MIDI Controller Sequencer

Team: Devin Alexander (dbalexa2),
Martin Lamping (mdl3),
Nathan Zychal (nzycha2)
ECE 445 Project Proposal -- Spring 2019
TA: Christopher Horn

Table of Contents

1 Introduction

1.1 Objective

Music has undoubtedly been an important part of human development. Every known culture, past and present, has expressed some form of music or melodies. Theories suggest that music has been around from dates as early as 500 A.D. [6]. The four widely accepted eras of music are, the Baroque Era, the Classical Era, the Romantic Era, and the Contemporary Era [7][8]. Music variety and accessibility increased dramatically toward the end of the Romantic Era/beginning of the Modern Era (~1880) with the creation of the gramophone [10] and start of radio broadcasting [9]. Radio broadcasting had allowed for music to reach nearly 40 percent of American households [1]. In comparison to other fields such as the sciences, music development had been relatively slow. The 20th century not only led to inventions that could play music such as the gramophone, but also music recording [3] and later distribution. The use of sheet music was largely limited to that of the middle and upper class because the people using sheet music needed to be able to sing, play and read music [5]. Accessibility and the creation of music has always been an issue and continues to be in modern day.

Modern day software interfaces give a solution to the above issue of music accessibility and creation; however, we believe a dynamic *physical* solution has not been implemented. Our goal is to create an easy to use physical interface for prototyping and designing music and melodies for the modern-day producer. We will be creating a sequencer that works based off of MIDI communication. The “steps” in each sequence will play a sound given a pitch and root note, and pitches will either be set on a continuous or quantized scale. In order to create a quantized scale, the voltage from each motorized potentiometer must be read into an ADC. The frequency for each step will be set from positions of the motorized potentiometers and the note will eventually be output of the device.

1.2 Background

There have been various attempts at creating a dynamic sequencer with many features in modern day (particularly software based DAWs). There are far fewer hardware-based implementations of sequencers. Software based sequencer implementations (included in DAWs) have been implemented by big name companies such as Sony or Yamaha, however they are not the most popular. Instead, recording software such as FL Studio, Ableton, or Logic Pro are more commonly used and have in some cases been ported to physical synthesizers [2].

1.3 High-Level Requirements

• Motor Controllers must be used to allow for correct functioning of motorized potentiometers.
•Data must be communicated between devices (ex. from the ADC to the output of the system) through SPI, I2C or indirectly(ex. voltage levels,etc...)
•Appropriate power must be supplied multiple sequencer modules(3.3, 5, or 12 volts)

2 Design

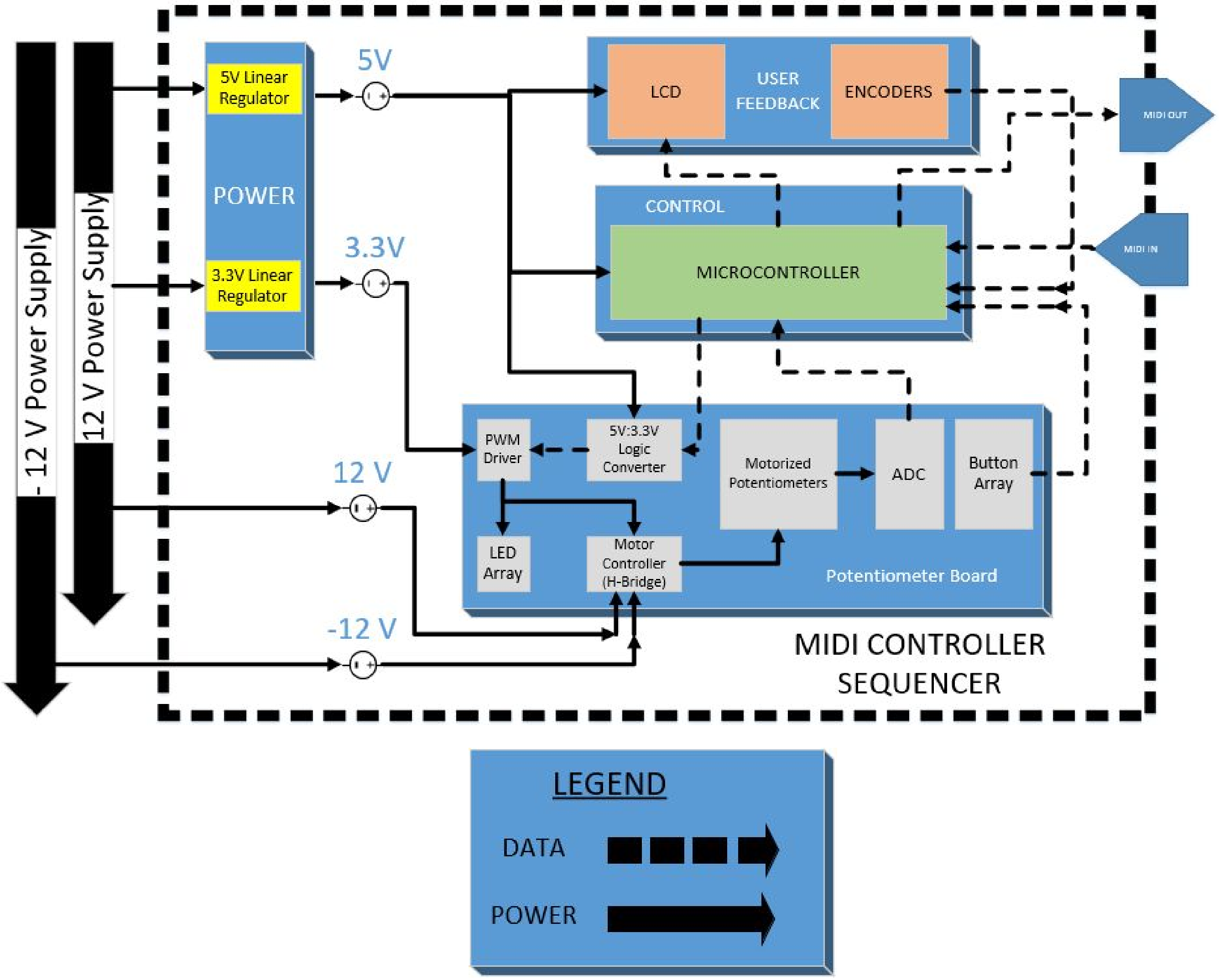

Figure 1 (Block Diagram): The figure above shows the block diagram implementation of the MIDI controller sequencer.

Input(s): Midi In **Output(s):** Midi Out
Notes: A *maximum* of 16 motorized potentiometers may be used. Functions controlled by buttons include but are not limited to tempo and randomization.

2.1 Block Diagram Description: The block diagram will meet the high level requirements mentioned in the introduction. The motor controllers are connected directly to the potentiometers and SPI communication protocol will be used to optimize data transfer rates. The arrows above indicate which sections of the sequencer will need to communicate with each other and over what protocol they will do so. Different power(3.3, 5, or 12 volts) will be routed to the appropriate module depending on its logic(or lack thereof). The 16 motorized potentiometers will be used to input specific tones which will be transferred through the ADC. The ADC's output will be read and processed by the microcontroller and MIDI data will be output. The LCD will show current user settings and processed music information

2.2 Physical Design: A 4u rack unit(4 * 1u) with a faceplate will be used to house all the PCB's internal to the sequencer and also hold the motorized potentiometers and LCD screen. It will also be retrofitted to house the buttons and rotary encoders. (Dimension(s): 4u= 7 inches of rack unit height=177.8mm).

Note: This section of the project will be considered a reach goal. We are unsure if the final dimensions of the PCB will be exactly what we have approximated.

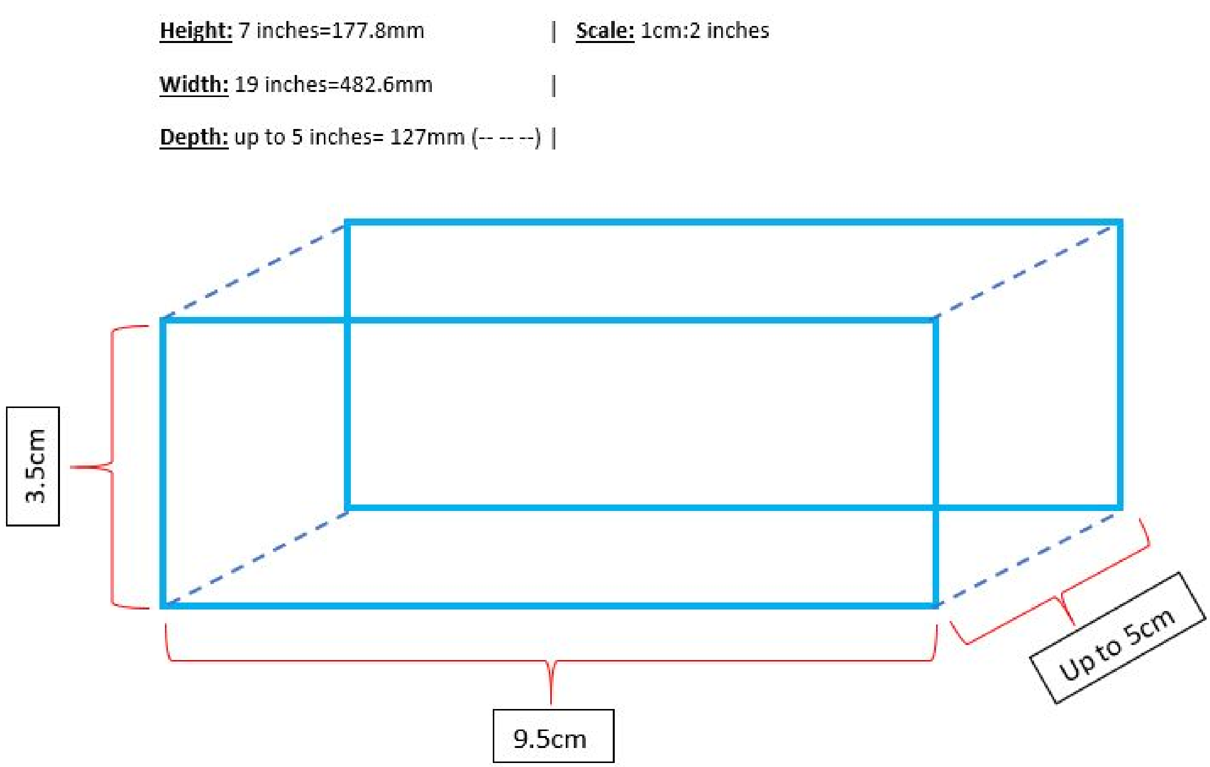

Figure 2(Physical Design Measurements): Potential Spatial Requirements

Figure 3 (4u Rack Unit Without Faceplate): Potential housing unit

Note: *If* a faceplate is designed it will be retro-fitted to hold all the motorized potentiometers, LCD screen, buttons, and encoders.

Functional Overview and Block Requirements

2.3 Power

2.3.1 Power Supply

This module is external to our design. We will be using a pre-existing power supply block to provide power to the system.

Requirement(s): Must provide +/- 12V, unless we determine otherwise.

2.3.2 Linear Voltage Regulators

Two linear voltage regulators are required to step down the 12V from the power supply. This is necessary because a few components will need to be supplied with 5V and 3.3 V power.

Requirement(s):
The 5 volt linear regulator will be providing power to most of the user interface ICs on the board, and the 3.3V linear regulator will be providing most of the power to ICs that are responsible for providing power to the Motor Controller ICs and LED drivers.
For the 5V Linear Voltage Regulator, we will be using a LM7805 and will need it to be rated for use of a maximum of 500mA. For the 3.3V Linear Voltage Regulator, we will be using a LM7833 and it will need to be rated for 1A.

2.4 User Feedback

2.4.1 LCD
The LCD will show the current states of user input such as the current note, octave,scale type, root note, tempo(clock rate), and duration. The size of this LCD display will be made of (4 rows by 20 columns) that can print off (9 row x 5 column) ascii/sprites in each space.

Requirement(s):
The LCD should be capable of either displaying all the listed data either at one time, or must cycle through the relevant information by the press of a button. The LCD will be running off of Hitachi hd44780 IC that is onboard the LCD module.

Since it requires the use of parallel pin data requirements and we need to save IC digital pins for use of connecting to other modules, we will be using a Parallel to I2C BUS converter chip to free up digital pin requirements. This will be soldered into place on the LCD module pins and we will be using I2C to connect to the microcontroller.

2.4.2 Encoders
The rotary encoders will be used to control the selection of scale, octave range, quantization of pitches (in particular key), and potentially additional parameters. These encoders will change options that display on the LCD screen and will only change the parameter when the button is pressed inwards.

Requirement(s):
We will be using encoders that utilize 5 pins per encoder. These encoders will be 20 detent (# of steps for complete 360 degree turn) encoders and the shaft that is rotated is also a button if pressed inwards. The 5 pins consist of (5V power, GND, DATA OUT 1, DATA OUT 2, and BUTTON). The microcontroller will be utilizing three digital pins for each of the encoders with 5 V and GND shared between other components.

2.5 Control

2.5.1 Microcontroller

We will be using an ATMEGA2560 as the microcontroller. The microcontroller's main purpose will be to control and operate the system as a whole. It will need to complete various tasks, some of which are timing critical. It will be the brains of the MIDI controller. It will use a host of protocols to talk to ICs on this device.

Note: We are still looking into using a dedicated IC to translate the USB data into serialized binary data that the microcontroller uses. It would be placed on the same PCB as the microcontroller. Otherwise we are going to have an external usb uart to serial programmer/debugger.

Requirement(s):
Interrupts: will be used for timing critical tasks to ensure that data is handled appropriately when it is generated/requested.

Crystal Oscillator: will run off a 5V linear regulator output and will oscillate at 16.000 MHz and provide a clock pulse rate to the microcontroller.

I2C: will be used to communicate with the PWM Drivers, the LCD display, and potentially the ADCs.

SPI: communication protocol will potentially be used to communicate with the ADCs that we are going to use if we don't use I2C.

UART: will be used for interfacing with the MIDI Ports. Two UART ports will be used total (1 for the MIDI IN and 1 for the MIDI OUT).

Digital Pins: will be used as ways to interface with buttons, encoders, and any SPI slave select pins.

2.6 Potentiometer Board

2.6.1 Motorized Potentiometers

The user will be able to control the movement of the potentiometer sliders either manually or through the use of rotary encoders. This will in effect change the tone data that will be output through MIDI.

<u>Requirement(s):</u>
The wiper positions on the motorized potentiometers will need to move to specific localized positions along the potentiometers linear movement range. These positions will be used to denote pitches in the MIDI output.

The potentiometers may need to have the noise that is generated by surrounding motors filtered out of the output voltage before going to the ADC for sampling.

The potentiometers will need to move at a speed slow enough so that users aren't harmed, yet also fast enough so that the movement is fluid. We may need to implement a PID algorithm to model the movement and send the corresponding signal to the PWM generator.

2.6.2 ADC

We will use 2 x 8 channel ADCs to sample the voltage signal from the linear potentiometers to cover all of the 16 potentiometers on the potentiometer board.

<u>Requirement(s):</u>
We will need to ensure that the noise from the motors is not present in the sampled signal. The noise may be mitigated by use of analog low pass filters or other methods.

They will need to be placed in strategic points on the potentiometer board in order to allow use of easy trace placement to connect wipers to inputs at the ADCs.

They will need to provide accurate and up to date values to they can be used in real time to move the motors to the correct positions. So sampling rate will need need to be fast enough to provide enough data to the microcontroller.

2.6.3 PWM Driver (LED Driver)

We will be using PCA9685PW as the IC to provide PWM power signals for controlling the LEDs as well as the PWM signals that will be used by the H-bridge in the motor controller.

<u>Requirement(s):</u>
The LED driver will need to provide pulses that are able to drive LEDs at a maximum of 20mA per channel. These PWM signals will also be responsible for providing the H-Bridge with a duty cycle at which to drive the motors in either a positive channel or negative direction.

The IC will be powered off of a 3.3V power supply.

2.6.4 H-Bridge (Motor Driver)

We will be using the TB6612FNG as the H-Bridge for this project. Each H-Bridge (motor driver) will be used to control the movement of 2 motors at the same time. It will as close as it can be to two motors so we can maximize the power efficiency.

Requirement(s):

The requirements for operation of the motor drivers are essentially to take a PWM signal from the PWM Driver and and pulse either the positive or negative bridge to control the motor in the respective forward or negative direction. It will be provided a +12V and -12V supply for either forward or reverse motor movements.

The IC will be powered off of a 3.3V power supply.

2.6.5 Button Array

Functions implemented by the buttons may include but are not limited to randomization and power.

Note: The encoders will also have buttons, but that is covered in the encoder section.

Requirement(s):

There will be 16 buttons on the potentiometer board for use of selecting steps to modify.

There will also be various buttons on the LCD PCB that will be used to select/navigate various midi modification options.

We will need to debounce signal coming from the button. Most likely a low pass filter. Software debouncing will need to be done potentially.

2.6.6 LED Array

LEDs will be used as indicators, placed at various locations on the faceplate of the rack unit. They will indicate the current potentiometer(s) that is(are) currently being used

Requirement(s):

RGB LEDs will do just fine for this application. We would like for them to be able to change color based on different parameters being modified. These will need to clearly indicate to the user what parameter is being modified (when in edit mode). It will also *need to clearly indicate to the user when a step is occuring in the sequence when in normal operation mode (not in edit mode).*

2.6.7 <u>5V : 3.3V Logic Converter</u>
We will need to change the voltage that the I2C communicates over coming from the microcontroller logic level of 5V to a logic level of 3.3V that the PWM driver will use as to not fry any traces in the IC. This will be accomplished by using a 2 channel Logic level converter.

<u>Requirement(s):</u>
It needs to convert 5V to 3.3V in both directions of flow.

<u>2.7 Ports</u>

2.7.1 <u>MIDI Ports (IN & OUT)</u>
The MIDI Ports will provide the means of speaking MIDI to external controllers and devices.

<u>Requirement(s):</u>
The physical MIDI connectors will need to be securely fastened to the panel in order to sustain at least a badly misaligned attempt to plug in a MIDI cable without breaking the fastening to the interface.

They will need to be clearly marked to denote which is the input and which is the output port.

These ports will be connected to the microcontroller on 2 of the 4 dedicated UART ports on the ATMEGA2560 since MIDI is a very time sensitive protocol. This is possible as the ATMEGA2560 has 4 dedicated UARTs on it.

The MIDI output from this will contain MIDI note on/off/duration/velocity data and will also have a tempo output.

The MIDI input will only be used for syncing with external tempo data and nothing else.

2.8 <u>Risk Analysis</u>
The block that poses the greatest risk to the successful completion of the project would certainly be the potentiometer board. The potentiometer board will hold all the motorized potentiometers, selection buttons, display LEDs, potentiometer voltage ADCs

as well as the Motor controller, LED Driver and logic converter. The potentiometer board is, by far, the largest PCB, with the greatest number of parts, that must be designed and soldered precisely to work correctly. The EMI produced by the motors may also interfere with the ADC readings [4]. This EMI will need to be correctly detected and handled appropriately to ensure it doesn't cause the potentiometers to position wobble. This is another obstacle that must be overcome to successfully implement our potentiometer block. Various voltage levels must be supplied to this PCB and circuit protection must also be devised to protect both the user and the more difficult sourced components like the motorized potentiometers.

3 Ethics and Safety

3.1 Safety

Synthesizers/sequencers generally operate at +/- 15 volts [11], these voltage levels are considered a safe range for users to operate the device at. In fact, our groups' synthesizer implementation will run at an even smaller voltage, with a maximum operation voltage of around +12 volts. Even with the device operating at a low voltage level, there is always a potential for risk. To mitigate the risk of electrical shock, the device inputs and outputs, and even some passive elements will be clearly labeled. An on/off switch *may* also be built in to the device with a led response to keep users informed of the sequencers current status.

3.2 Ethics

Ethics do not play a huge part in the consideration of our project. The only readily apparent ethical dilemma that may need to be considered is the way in which our device is used. The types of music, sounds, and melodies, a user chooses to produce can have an effect on a listener's behavior.

Citations

[1]

"A Chronology of AM Radio Broadcasting 1900-1960." *Earliest Known Uses of Some of the Words of Mathematics (S)*, jeff560.tripod.com/chrono1.html.

[2]

"Evolving DAWs." *The History Of Korg: Part 1 |*, Sound On Sound, 1 Feb. 2019, www.soundonsound.com/reviews/evolving-daws.

[3]

Fabry, Merrill. "First Recorded Sound: Scott, Edison and History of Invention." *Time*, Time, 1 May 2018, time.com/5084599/first-recorded-sound/.

[4]

Getz, Robin and Bob Moecker. "Understanding and Eliminating EMI in Microcontroller Applications." Literature Number: SNOA382. Texas Instruments. August 1996. http://www.ti.com/lit/an/snoa382/snoa382.pdf

[5]

"How Was Musical Notation Invented? A Brief History | WQXR | New York's Classical Music Radio Station." *WQXR*, www.wqxr.org/story/how-was-musical-notation-invented-brief-history/.

[6]

Martins, Ralph. "Was 'Earliest Musical Instrument' Just a Chewed-Up Bone?" *National Geographic*, National Geographic Society, 15 Dec. 2017, news.nationalgeographic.com/2015/03/150331-neanderthals-music-oldest-instrument-bones-flutes-archaeology-science/.

[7]

Paterson, Jim. "Classical Music Periods." *Hans Zimmer - Overview of the Film Composer and His Music*, www.mfiles.co.uk/classical-periods.htm.

[8]

Paterson, Jim. "Composer Timelines for Classical Music Periods." *Hans Zimmer - Overview of the Film Composer and His Music*, www.mfiles.co.uk/composer-timelines-classical-periods.htm.

[9]

"Radio: The Internet of the 1930s." *American RadioWorks*, www.americanradioworks.org/segments/radio-the-internet-of-the-1930s/.

[10]

"The Gramophone | Articles and Essays | Emile Berliner and the Birth of the Recording Industry | Digital Collections | Library of Congress." *Planning D-Day (April 2003) - Library of Congress Information Bulletin*, Victor, www.loc.gov/collections/emile-berliner/articles-and-essays/gramophone/.

[11]

“What Is Voltage? | The Synthesizer Academy.” *Synthesis Tutorial*, synthesizeracademy.com/what-is-voltage/.