Senior Project I Report

MIDI Signals Manipulating Console for Live Multiple-Keyboard Performance

---

Project Advisor
Asst. Prof. Dr. Thitipong Tanprasert

Committee Members
Asst. Prof. Dr. Benjawan Srisura
A. Chayapol Moemeng

Submitted By
Kanyarat Nalucupchanchai
Napatsorn Kawaree
Taechasit Sarasitt

IT 4291 Senior Project I (1/2021)

Senior Project Approval

Project title:	Text Classification for Education Publication
Academic Year:	1/2021
Authors:	Tachasit Sarasitt (6110032) Napatsorn Kawaree (6111099) Kanyarat Nalucupchanchai (6115308)
Project Advisor:	Asst. Prof. Dr. Thitipong Tanprasert

The Senior Project committee's cooperation between the Department of Computer Science and Information Technology, Vincent Mary School of Science and Technology, Assumption University had approved this Senior Project. The Senior Project in partial fulfilment of the requirement for the degree of Bachelor of Science in Computer Science and Information technology.

Approval Committee:

…………………………..
(Asst. Prof. Dr. Thitipong Tanprasert)
Project Advisor

…………………………..
(Asst. Prof. Dr. Benjawan Srisura)
Committee Member

…………………………..
(A. Chayapol Moemeng)
Committee Member

Abstract

Nowadays, music concerts or live performances need good music equipment especially ones that make the performance turn out entertaining for the audience, have its unique point, and multiple keyboards is one of the most important equipment. Keyboard-type device that doesn't have a built-in sound source is called a MIDI controller. There are two types of MIDI controllers: performance controllers that generate notes and are used to perform music, and controllers that may not send notes, but transmit other types of real-time events. Many devices are some combination of the two types. Keyboard-type device needs to be connected to other devices that are the source of the sound, through MIDI protocol. Yet, existing musical instrument digital interfaces are designed for sound engineers, which are not easy to use for real-time performance by musicians of typical instruments, even by pianists. In this project, we studied the technologies involved with modern digital sound productions, then created an easy and futuristic musical instrument as a digital interface for multiple keyboards that supports live music performance, is not more complicated to operate than an organ, and utilizes a potentially unlimited repertoire of sounds.

Table Of Contents

Table Of Contents

Chapter 1: Introduction

MIDI is short for Musical Instrument Digital Interface. It’s a protocol that allows computers, musical instruments, and other hardware to communicate. MIDI is a communication standard that allows digital music gear to speak the same language. A MIDI keyboard or controller keyboard is typically a piano-style electronic musical keyboard, often with other buttons, wheels, and sliders, used for sending MIDI signals or commands over a USB or MIDI 5-pin cable to other musical devices or computers. MIDI keyboard is a very common music equipment among musicians because MIDI gives musicians the chance to edit performances note-by-note or entire sections of music through features like quantizing. MIDI can be used as a versatile tool for writing, recording, and performing. It gives musicians instant access to a universe of sounds they wouldn't have had access to before

In the general market, there are piano, organ, electone, These have their own uniqueness. Playing the organ and piano are similar in technique. Electone (with bass pedal legs, double keys) will have another way to play because the bass is played with the left foot and use the right hand to play on the upper keyboard and left hand on lower keyboard. The right foot controls the volume and the Rhythm box. It will be an electronic system with bass, drum, chords, buzzing with many different rhythms. This electronic keyboard is also popularly used as an instrument in a band because there are many musical instruments to choose from. However, an electronic organ is a tool for an organist only and it cannot create modern sounds unlike electone and the price is very high in the market. Our project is a MIDI designed for everyone or musician in particular to access and get their hands on a very convenient and easy-to-use and simple MIDI keyboard.

1.1 Problem Statement

In order to facilitate digital audio technology in providing flexible arrangement of sounds for a musician to perform in live mode, while keeping minimal dependence on specific hardware, a software is to be developed as a generic console of a multiple-keyboard musical instrument. The user can connect MIDI-controller keyboards to the PC running the software. The software provides user-friendly interface for selecting and mixing sounds, for selecting rhythm and tempo, for saving and restoring sounds arrangements as presets that can be recalled on the fly, and manages all the MIDI outputs to produce the actual sounds through Digital Audio Workstation. The user interface is expected to utilize touch-screen technology, so that all the buttons and continuous control devices can be made totally virtual.

1.2 Scope of the project

- The software will be developed using Python as programming language, Mido as MIDI library, and Qt designer as user interface design software.
- We use Tracktion Waveform Digital Audio Workstation (DAW) to turn MIDI signals into sound.
- The PC's specification must support ASIO audio standard or a compatible one. ASIO is required for fast real-time audio response to the MIDI signals.

Chapter 2: Related Work

According to our research on the relevant tasks of MIDI Signals Manipulating Console for Live Multiple-Keyboard Performance, there does not have any existing interface or similar work with our project that can really meet the needs of users or musicians. Therefore, we have adapted problems encountered in related works and similar to our work. Other file formats from research on related work have been listed below.

MIDI or Musical Instrument Digital Interface is a technical standard that describes a communications protocol, digital interface, and electrical connectors that connect a wide variety of electronic musical instruments, computers, and related audio devices for playing, editing, and recording music. No audio signals are sent via MIDI. Instead MIDI works as a digital signal (0s and 1s). A series of binary digits. Each instrument or computer understands and then responds to these 1s and 0s, which are combined into 8-bit messages supporting data rates of up to 31,250 bits per second.

2.1 Protocol of MIDI

It is a standard protocol invented in 1982 as a music communication system of electronic music devices such as computers, synthesizers, sequencers, sound modules and samplers, which use digital electrical signals to transmit data between connected devices to musical note. and various sound control settings.

The MIDI message is up to 3-bytes long and consists of 3 parts, status byte, data byte 1, and data byte 2. The status byte is mandatory, it describes the functionality of the message alongside with the channel it operates on. While the optional data byte indicates the detail of the message. The example of MIDI message is shown in figure 2.1 below.

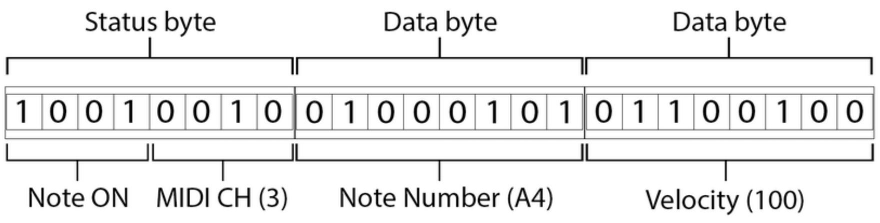


Figure 2.1: An example of MIDI message

2.2 Midi transport and MIDI standard

The original MIDI 1.0 Specification called for using a 5-Pin DIN cable to connect MIDI compatible devices, but today there are many different "transports" capable of carrying MIDI data and the specification for 5-Pin DIN has been updated. So MIDI 2.0 was there to change the musical life. MIDI 2.0 was released in 2020 by representatives Yamaha, Roli, Microsoft, Google, and the MIDI Association. Its significant update adds bidirectional communication while maintaining backwards compatibility. MIDI 2.0 has the new Universal MIDI Packet format for high-speed transports which supports both MIDI 1.0 and MIDI 2.0 voice messages. The Universal MIDI Packet is intended for high-speed transport such as USB and Ethernet and is not supported on the existing 5-pin DIN connections. System Real-Time and System Common messages are the same as defined in MIDI 1.0. This new packet format supports a total of 256 MIDI channels instead of 16 channels. The resolution is higher for data transmission as well as speed and control and increased transmission speed from 7 bits to 16 bits.

2.3. Waveform

Waveform audio or audio software is often used to mean the recorded sound itself. in order to distinguish it from structured audio like, MIDI data. Examples of waveform programs are Cakewalk, Sonar, Adobe, Tracktion, etc . These are digital audio workstations for sound engineering. It is an electronic device or application used for recording sound, editing and producing audio files such as songs, music pieces, speech or audio. But it is adequate for live performance because it has recording delay compensation, low latency monitoring and latency issues and it can not let the user play and manipulate the live performances on-the-fly. Furthermore, these software's user interfaces are too complex for live performance as well.

2.4 Related work

A related work is a well-known commercial program called Hauptwerk. Hauptwerk is a German computer program available from Milan Digital Audio designed to allow playback or live playback of pipe organ music using MIDI and recorded audio samples.

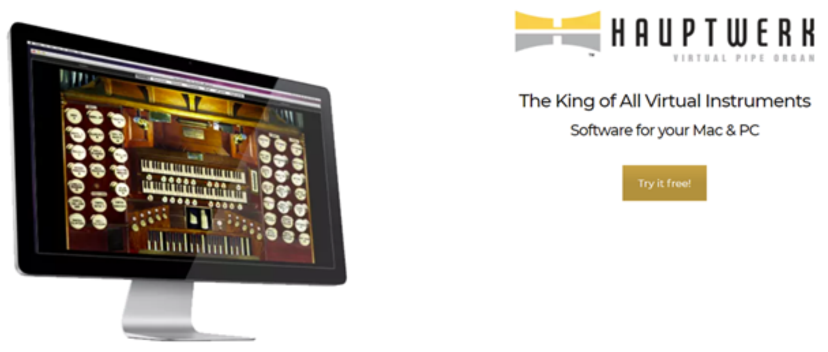

HAUPTWERK
VIRTUAL PIPE ORGAN
The King of All Virtual Instruments
Software for your Mac & PC
Try it free!

Figure 2.2: A related work (Hauptwerk)

The advantages of hauptwerk are perfect for church organists. It has many sounds provided in the program. the power of Hauptwerk, including expression, tremulants, crescendo, bass and melody couplers as well as a fully working combination system identical to the original organ and All people can access this program.

Yet, there are some disadvantages, they are only for a church organist, as each Hauptwerk software is an exact mock-up of a real church organ. All the sounds are fixed and thus Hauptwerk can not be used to create pop or rock music like what music is like in the modern world. Our project aim is to create an MIDI keyboard for users to be able to adjust volume and sounds to create different types of music.

Chapter 3: Proposed Methodology

3.1 Methodology

We developed it as a generic console of a multiple-keyboard musical instrument. The software is developed by using Python Language (Python3) and all methodology that we use for our project development, there are Mido, Qt, Qt designer, Tracktion Waveform, LoopMIDI and MIDI-OX.

3.1.1 Mido

We use the Mido (MIDI Objects for Python) library to interact with midi by treating midi as if it were an object in python. Mido is created by Ole Martin Bjørndalen and many other contributors. It is released as an open source library under the terms of MIT license. Mido can interact with midi by both real-time processing and file read/write operations, and has full support for all 18 messages defined by the MIDI standard. Mido has support for multiple backends, which are RtMidi (default), PortMidi, Pygame, rtmidi-python, Amidi. We use RtMidi as the backend in our software as it is recommended by the Mido developer.

The RtMidi, which Mido use it as backend, is a set of C++ classes which provides a concise and simple, cross-platform API (Application Programming Interface) for realtime MIDI input/output across Linux (ALSA & JACK), macOS / OS X (CoreMIDI & JACK), and Windows (MultiMedia System) operating systems

3.1.2 Qt

Qt is a cross-platform application development framework for desktop, embedded and mobile. Supported Platforms include Linux, OS X, Windows, VxWorks, QNX, Android, iOS, BlackBerry, Sailfish OS and others. Qt is also a graphical user interface (GUI) framework, a toolkit, that is used for developing software that can be run on different hardware platforms and operating systems. Qt makes it easy to develop software with native-looking (to the OS it is running on). For our project, Qt version 4.0 which generated from Qt Designer we used to develop a framework.

3.1.3 Qt Designer

Qt Designer is a tool for quickly building graphical user interfaces with widgets from the Qt GUI framework. It gives us a simple drag-and-drop interface for laying out components such as buttons, text fields, combo boxes and more. Here is a screenshot of Qt Designer on Windows:

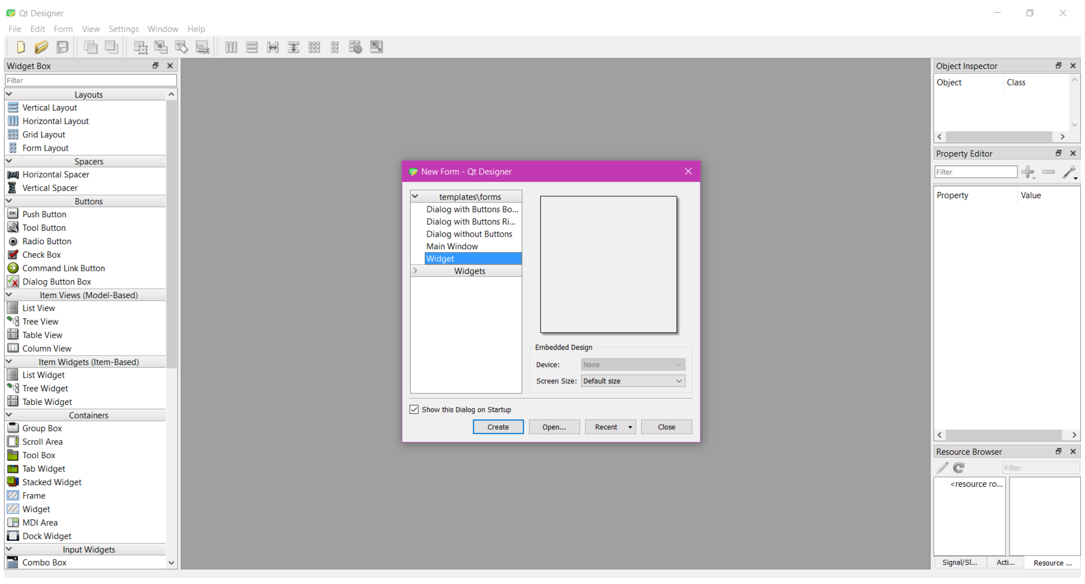


Figure 3.1: The screenshot of Qt Designer on Windows

In our project, we use Qt Designer together with Python because it is a dynamic language that lends itself well to rapid prototyping. Using its simple drag and drop interface, a GUI interface can be quickly built without having to write the code. The UI design and backend programming is done separately, which makes the software development process faster.

We use Python PyQt5 library to make the connection between the Qt file and Python program. The UI element in the Qt file has its own unique ObjectName, which we link it with its corresponding function in the Python program.

3.1.4 Tracktion Waveform

Tracktion Waveform is a Digital Audio Workstation (DAW) for recording and editing audio and playing MIDI. The software is cross-platform, running on Apple macOS, Microsoft Windows and Linux. Waveform is a rapidly evolving application specifically designed for the needs of modern music producers. Specializing in creative and inspirational workflows and avoiding features not explicitly needed to allow the app to remain surprisingly fun and intuitive. While other apps try to appeal to broad user groups, for example film score, live sound, performance and focus on music production.

The use of Waveform in our project is to turn the midi signal that has been manipulated from our software into audible sound, since our software is only responsible for midi manipulation. But in the real world applications, users can freely use other software as a midi receiver, or even hardware if it is supported.

3.1.5 LoopMIDI

LoopMIDI is an application that can create virtual loopback MIDI ports. It makes use of the virtualMIDI driver to create the virtual ports, but in addition to the virtual driver, it also offers a way to control and configure the ports.

The virtual loopback MIDI-ports is used to interconnect applications on Windows that want to open hardware-MIDI-ports for communication. The ports created are unique for each user and only exist while the loopMIDI-application is running. So if we log-off, the created ports cease to exist. LoopMIDI comes from the fact that the ports created with this tool will cease to exist once the app is closed. They'll be up and running only while loopMIDI is also running, so we will be able to avoid having the ports interfere with other apps and creating unwanted situations. Furthermore, the ports are unique for each user.

The loopMIDI plays an important role in our project because we can test our program without having to connect the hardware midi devices. Which facilitates software development.

3.1.6 MIDI-OX

MIDI-OX is a versatile utility that is great for troubleshooting faulty MIDI hardware devices. It also acts as a System Exclusive SysEx) librarian, which allows you to send (dump) and receive SysEx data. It can perform filtering and mapping of MIDI data streams. It displays incoming MIDI streams, and passes the data to a MIDI output driver or the MIDI Mapper. It also can generate MIDI data using the computer keyboard or the built-in control panel.

The role of MIDI-OX in our project is to test and debug the functionality of our software since it can read the data inside the midi messages. We use this software to make sure that our software is manipulating midi messages as it should.

Chapter 4: Design of The System (or Work)

4.1 System Design

For our project interface designing, we got an electone produced by Yamaha to be our inspired design. We use the Qt designer program to develop our user interface designing for Musical Instrument Digital Interface.

4.1.1 Inspired design

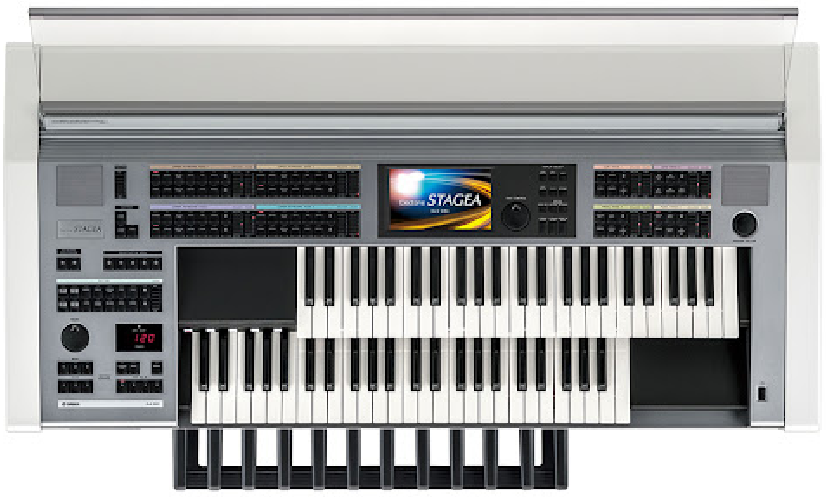


Figure 4.1: Yamaha electone STAGEA ELS-01C

In Figure 4.1, it is an electone produced by Yamaha and was released in the market in 2004. We use Yamaha electone stagea els 01 as our inspired design. we adjusted the design to our own design.

4.1.2 UI Prototype with Qt Designer

For our user interface designing, it is initially designed using a Top-Down approach. This is a design coupled with coding to get the desired results first. Then we will continue to improve the design customization. Our user interface designing is based on Qt. Designer program to design for users to be easy to use and convenient, fast, uncomplicated in use. The interface that we designed is developed in the following:

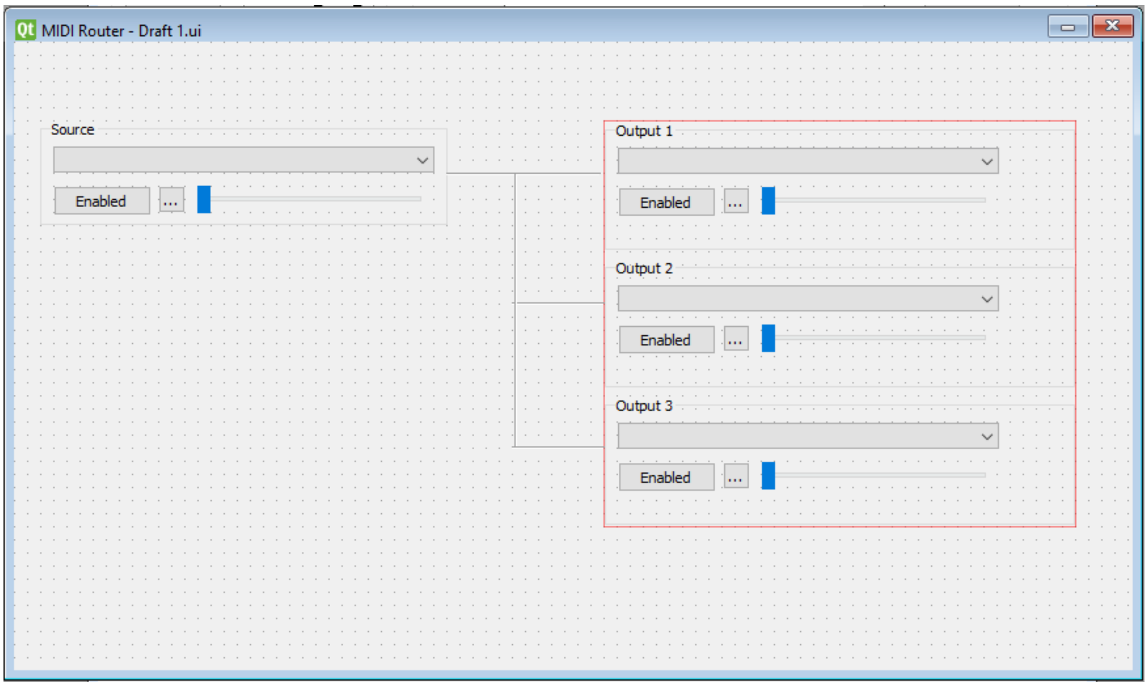


Figure 4.2: First design of Musical Instrument Digital Interface

In Figure 4.2, from the pictures you can see that we are starting to design little by little, starting with users being able to source and it can adjust the volume. The source refers to the Midi keyboard. The other side of the design is the part of output which is the receiver of the Midi.

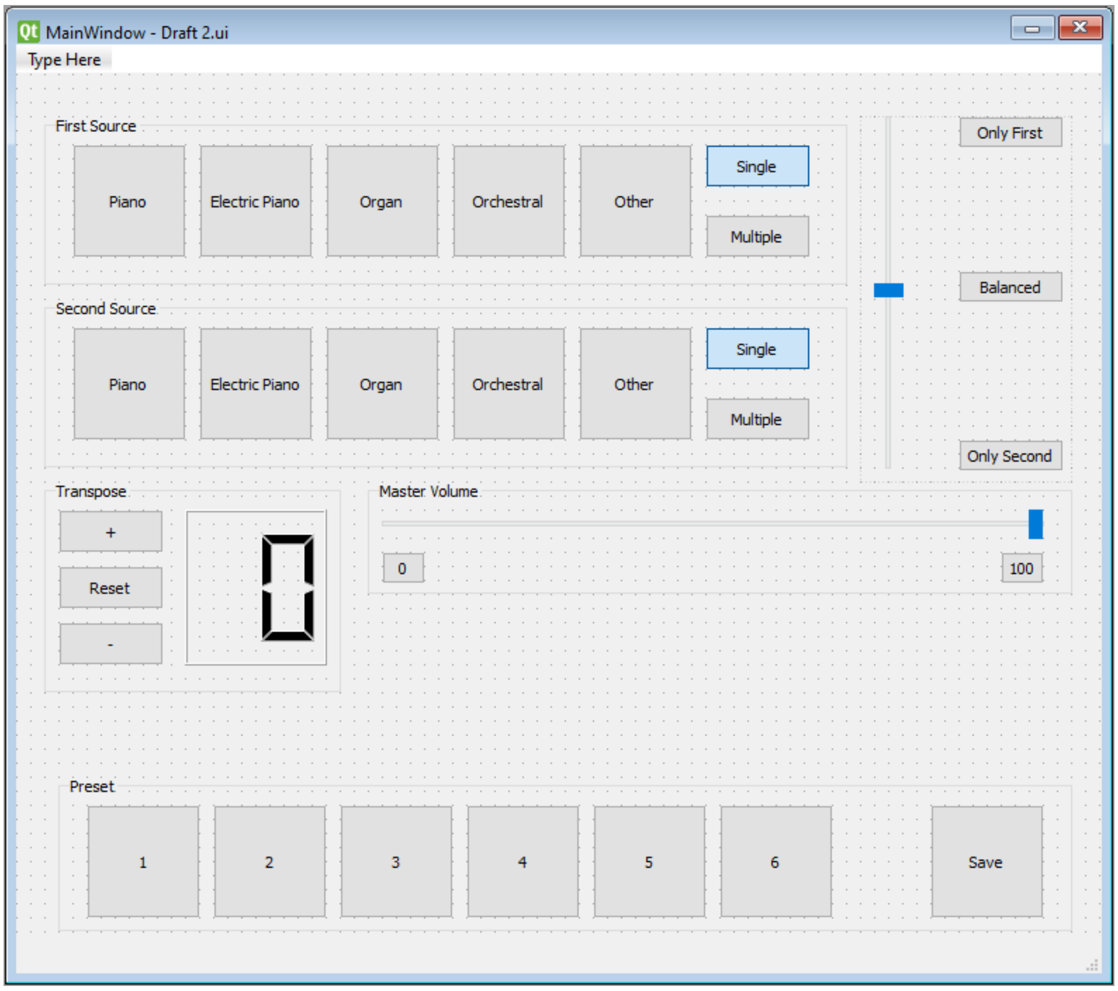


Figure 4.3: Second design of Musical Instrument Digital Interface

In Figure 4.3, we foresee that creating an example source is to allow users to choose and use it more conveniently. The source that we have provided Piano, Electric Piano, Organ, Orchestral and other. Single and Multiple that we have created to separate the work of the system. Single can open only one sound in the respective source section. The Multiple section will be able to open multiple sounds at the same time. In the balance section, it is not the balance of the sound level from left and right speakers, but the balance between source 1 and source 2. As for the volume level, we will adjust the volume of the Master Volume instead. Transpose is a note change feature, e.g. Transpose +1 = change from C to C#. Preset is for the system to remember the volume that we set and also we have a save button for saving preset.

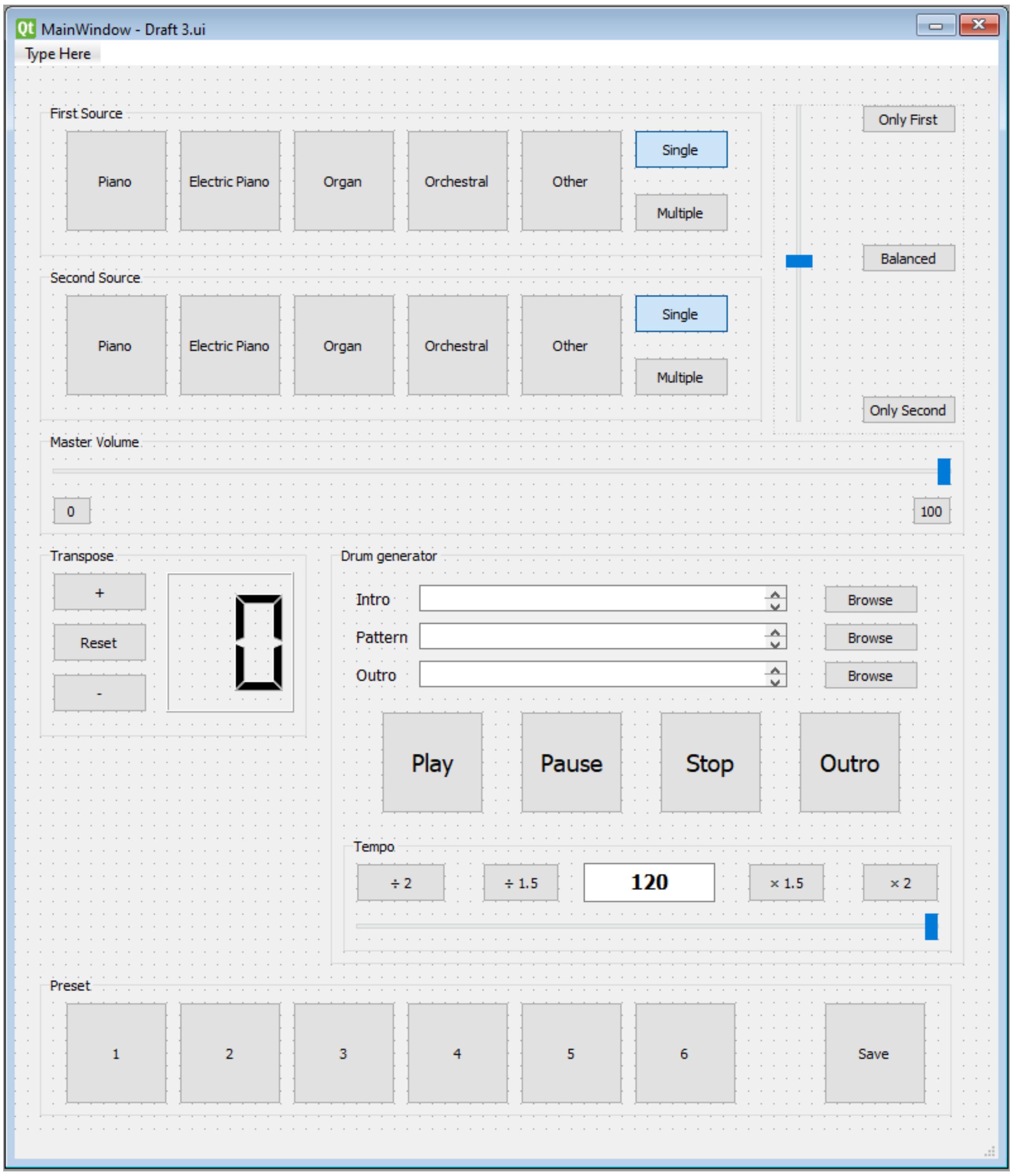


Figure 4.4: Third design of Musical Instrument Digital Interface

In Figure 4.4, From the 2nd design picture, we have added a Drum generator to import the sound file from outside and added 4 buttons which are Play button, Pause button, Stop button and Outro button. When we click on the Play button, there will play from the Intro section from the sound starting until the end and then play automatically at Pattern and outro respectively. We also added Tempo in order to adjust the slow or fast speed of the imported sound, the number is the tempo in the unit of beats per minute.

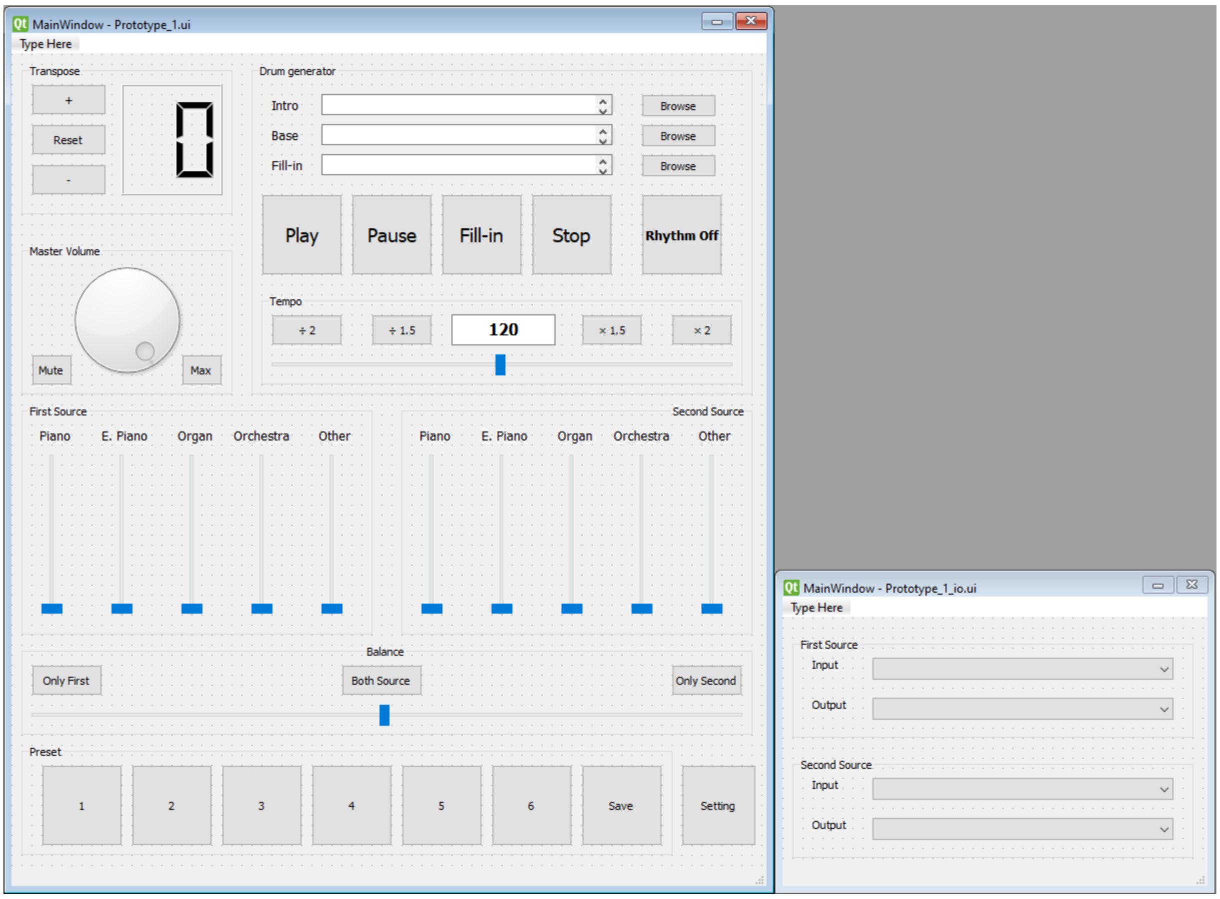
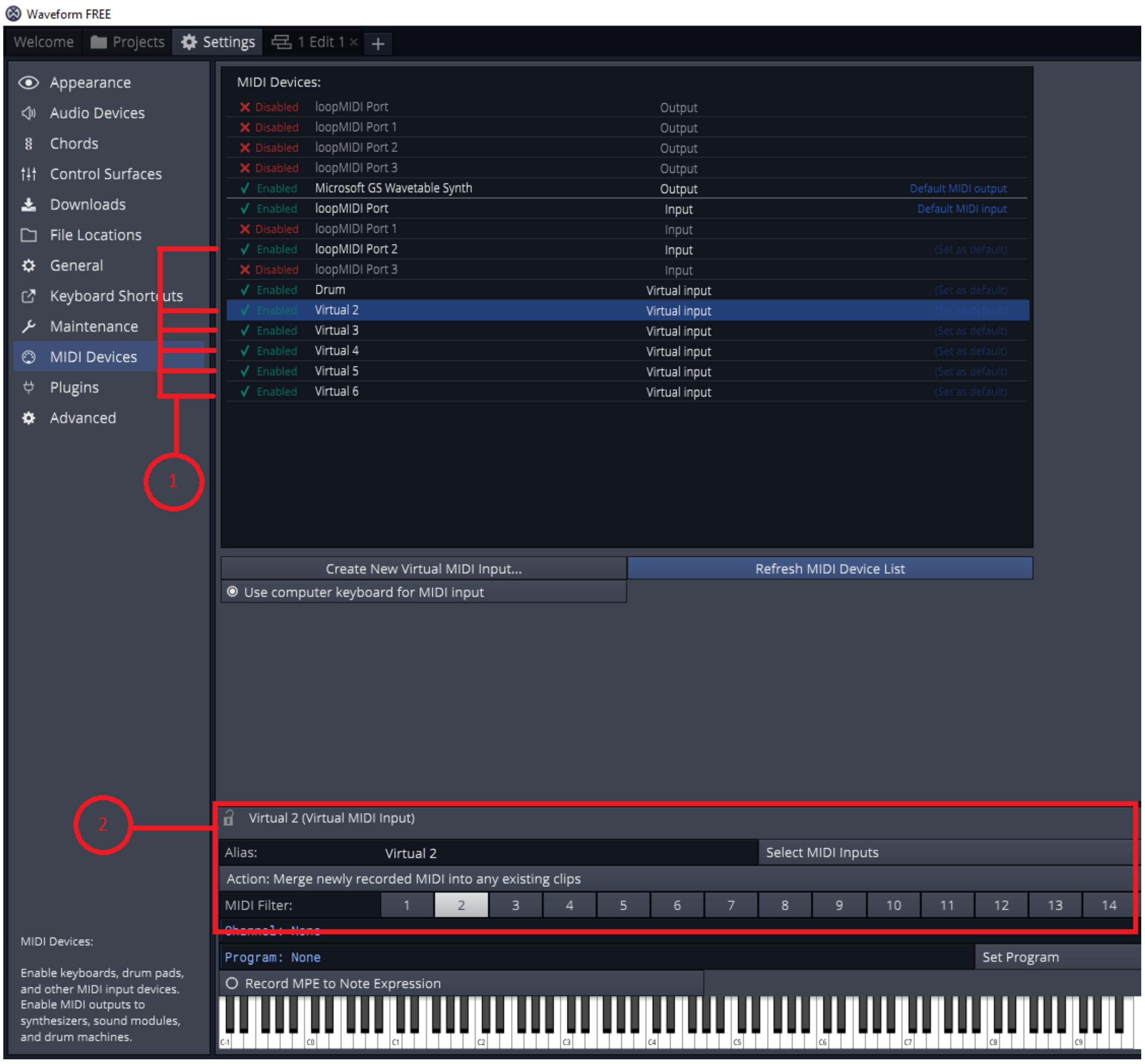


Figure 4.5: Fourth design of Musical Instrument Digital Interface

In Figure 4.5, the design of this stage was developed from the third design with modifications to the Master. Volume to be different in the design of the sound adjustment. We've added setting buttons, when we click, it will bring up the design window on the other side. The setting window is to choose which source receives and transmits midi signal from which port.

4.1.3 Waveform / Soundfonts

In order to test the software we developed with the real-world application, we use Waveform DAW as midi receiver software, and the open source Juicy SF audio plugin as a soundfont player. For the soundfront file, we use publicly available GeneralUser GS v1.44.



MIDI Devices:

✗ Disabled	loopMIDI Port	Output	
✗ Disabled	loopMIDI Port 1	Output	
✗ Disabled	loopMIDI Port 2	Output	
✗ Disabled	loopMIDI Port 3	Output	
✓ Enabled	Microsoft GS Wavetable Synth	Output	Default MIDI output
✓ Enabled	loopMIDI Port	Input	Default MIDI input
✗ Disabled	loopMIDI Port 1	Input	
✓ Enabled	loopMIDI Port 2	Input	(Set as default)
✗ Disabled	loopMIDI Port 3	Input	
✓ Enabled	Drum	Virtual input	(Set as default)
✓ Enabled	Virtual 2	Virtual input	(Set as default)
✓ Enabled	Virtual 3	Virtual input	(Set as default)
✓ Enabled	Virtual 4	Virtual input	(Set as default)
✓ Enabled	Virtual 5	Virtual input	(Set as default)
✓ Enabled	Virtual 6	Virtual input	(Set as default)

Virtual 2 (Virtual MIDI Input)

Alias: Virtual 2

Action: Merge newly recorded MIDI into any existing clips

MIDI Filter: 1 [✓ 2] 3 4 5 6 7 8 9 10 11 12 13 14

Channel: None

Program: None

MIDI Devices:

Enable keyboards, drum pads, and other MIDI input devices. Enable MIDI outputs to synthesizers, sound modules, and drum machines.

Figure 4.6: Waveform configuration screen

In our testing environment, all midi messages from our software are sent to “LoopMIDI Port 2”. Then it duplicated the message into the virtual interface 2 to 6 and drum interface -(1). In each virtual interface, the message is filtered to receive only the specific midi channel -(2), “Virtual 2” only receives midi channel 2, “Virtual 3” only receives midi channel 3, and so on. The drum channel is set to channel 10 to comply with General MIDI standards.

Chapter 5: Result

5.1 System Architecture

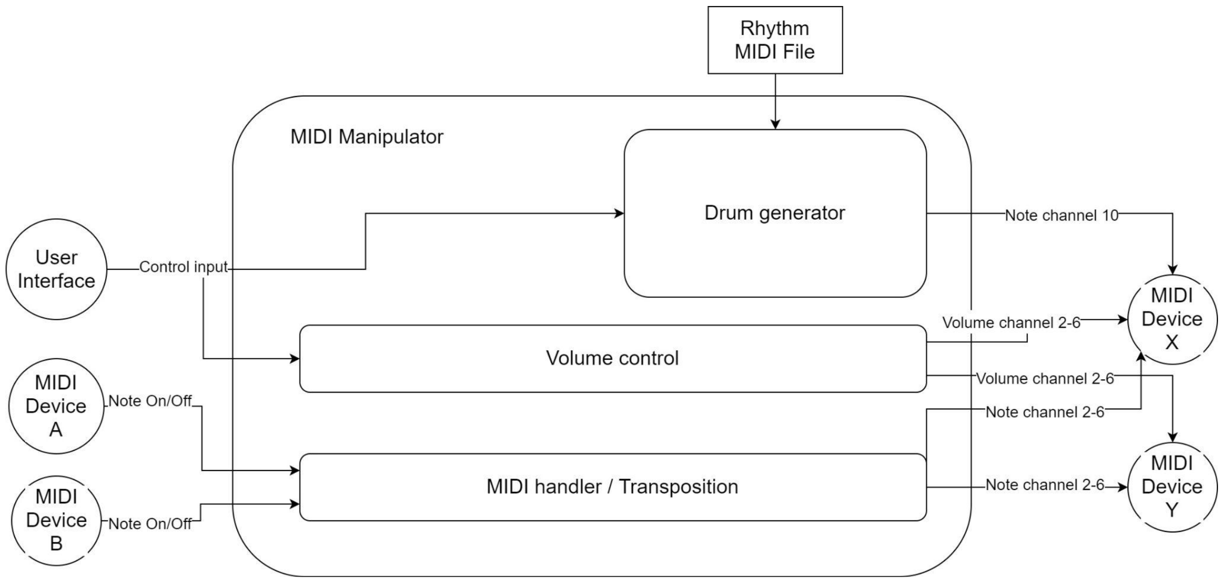


Figure 5.1: An overview of system architecture

Our software consists of 3 parts, Drum generator, Volume control module, and MIDI handler.

The Drum generator is the program that plays the percussion instruments pattern from the MIDI file. It is useful in the case of users needing the percussion sounds, drum beats, and patterns without having a drummer.

MIDI handler is responsible for duplicating and forwarding midi signals from source device to the destination device in real time. The transposition happens in the MIDI handler by modifying the midi NOTE ON/OFF message data according to the transpose set in software. MIDI handler function is operated by using the "callback" feature in mido library. Which is the function that only acts when the midi message is sent to the port that callback operates on. When the midi message arrives, it will be copied in for loop, set the channel according to each loop, set the transpose and then send it out. The functionality of MIDI handler is visualized in the flowchart below.

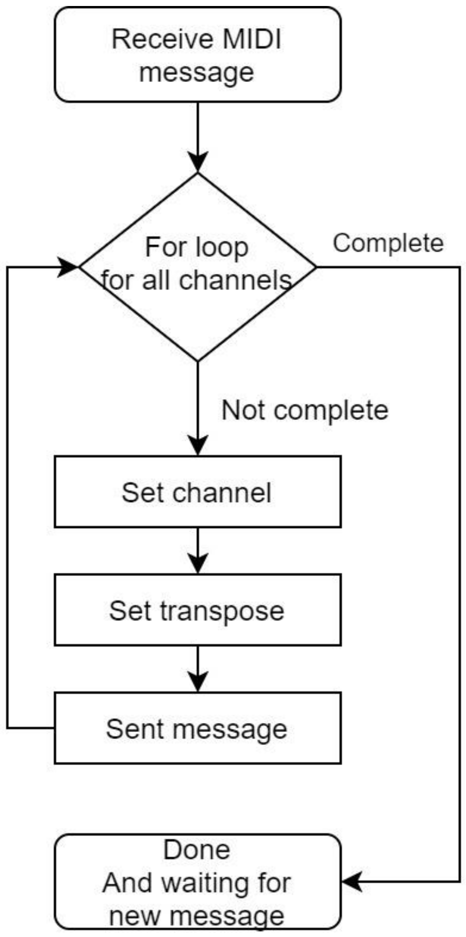


Figure 5.2: MIDI handler flowchart

Volume control is used to send the midi control (CONTROL/MODE CHANGE) message to the destination midi devices. The reason we have separate volume control instead of modifying the velocity data of midi NOTE ON/OFF is the velocity data greatly affects the sound characteristics, and not every instrument / soundfonts support for dynamic velocity.
The "slider" value is 0-127, which is the minimum / maximum value for midi messages. This function is called every time the volume-related UI is changed. The slider value will be modified by master and balance value before sending to the destination midi device.

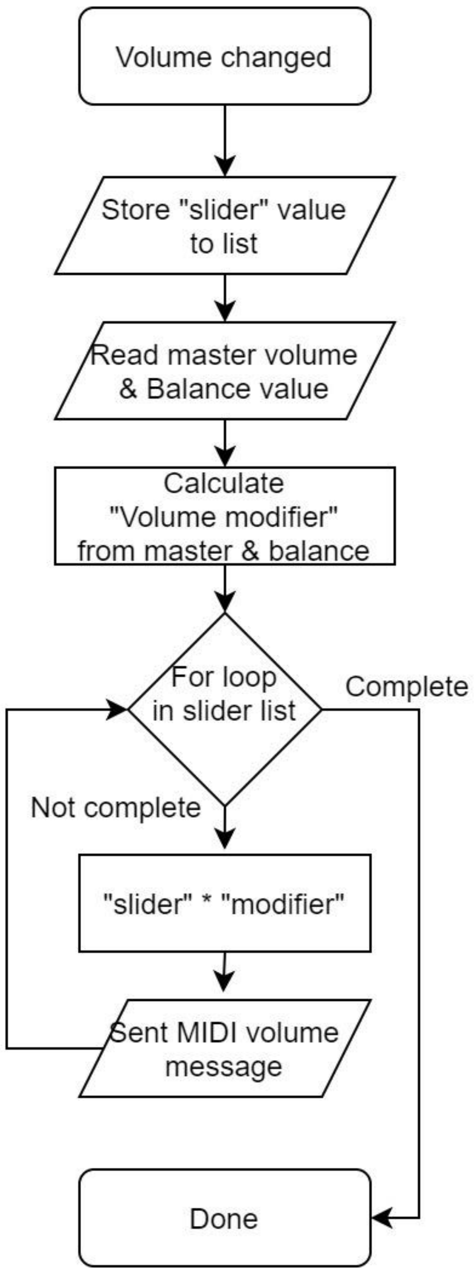

Volume changed
Store "slider" value to list
Read master volume & Balance value
Calculate "Volume modifier" from master & balance
For loop in slider list
Complete
Not complete
"slider" * "modifier"
Sent MIDI volume message
Done

Figure 5.3: Volume control flowchart

5.2 Installation Prerequisites

In order to use our software, users must have, or pass the prerequisite as described below.

1. The computer that runs this software should have a low-latency audio driver. In the case of Windows OS, it should install the ASIO driver and configure the sound generation software to use ASIO. Our software can operate without ASIO, but users may notice the added latency if they use the standard driver.
2. The computer that runs this software MUST have Python 3 installed. As well as Mido and PyQt 5 library. Which can be installed by using this following command.
    - pip install mido
    - pip install python-rtmidi
    - pip install PyQt5
3. Users MUST have at least 4 MIDI ports (physical or virtual) open and available before running our application. This is due to the fact that our MIDI handler will open the 2 pairs of ports when it operates.

5.3 User Interface and Function breakdown

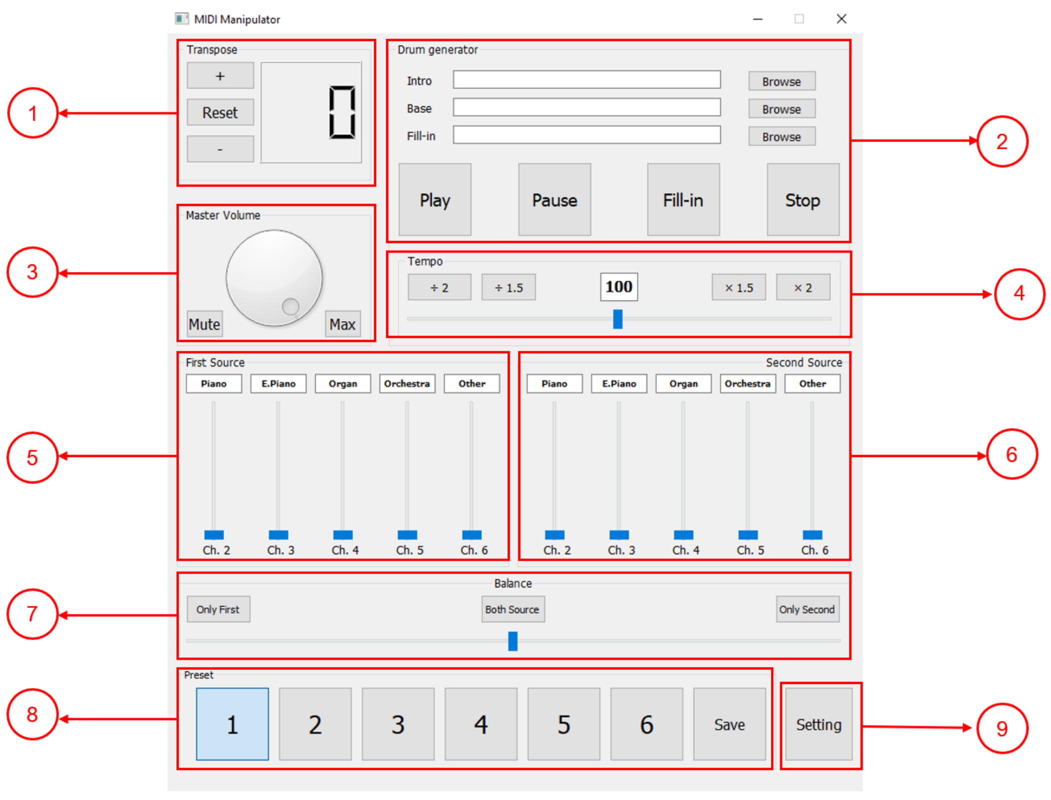


Figure 5.4: Final design of Musical Instrument Digital Interface, Main window

The volume control of each midi channel in each midi ports pair can be adjusted by using sound sliders in (5) and (6). By default, every channel is muted at the startup. This design requires users to increase the volume in each channel by themself to reduce the chance that the user accidentally sent the sound to the wrong channel.

Currently, the output channel for each port is fixed to channel 2 to 6, but we may allow users to change the channels in the future. The textbox above the sliders do not represent the sound type, since our software does not generate the sound. But rather is the note to the user which they can edit to match the instrument they set in midi destination.

The balance between First source and Second source can be set in (7). When the slider is placed at the center, both sources output the sound level they set in (5) and (6). But when moving the slider to the left, the sound volume of Second source will gradually decrease and reach zero when the slider is placed at the end of left. And vice versa.

If the user wants to instantly set the volume to only the first or second channel, or both channels at the same level. They can do that by pressing the button above the slider as well.

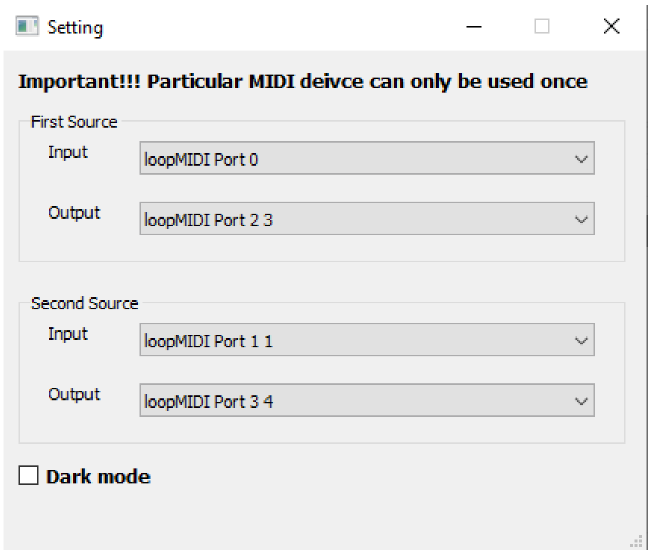
Master Volume (3) is used to adjust the sound for every channel before it sent out to destination midi device

Transposition (1) is the function that is used to change notes gradually by increase or decrease with the fixed value. For example, when setting the transpose to 1, it means the C note will transform into C Sharp. and when set to -1, C will transform into B. The range of transposition allowed by our software is -11 to +11.

Drum generator (2) is the function that can play Midi file, loop, and set the tempo of that particular file played. The program will play the intro file followed by the looped base, and it will play fill-in when the user clicks the fill-in button. The tempo of the drum generator is set by the slider in (4). If users want to increase or decrease the tempo by half or one fold, they can easily click the button above the slider.

The Preset system (8) allows the user to save the setting in (3), (5), (6), (7) into the preset. They can change the setting by just single clicking, which will be useful in the live-performance. They can save the setting into preset by clicking the Save button. But from this design, they cannot copy the preset from one to another.

Users can change the pair of midi input and output devices by clicking on the Setting button (9). Then it will pop up the new window, as shown in Figure 5.3 below.



Important!!! Particular MIDI deivce can only be used once

First Source
Input: loopMIDI Port 0
Output: loopMIDI Port 2 3

Second Source
Input: loopMIDI Port 1 1
Output: loopMIDI Port 3 4

☐ Dark mode

Figure 5.5: Final design of Musical Instrument Digital Interface, Setting window

Dark mode is also available for our software, it can be toggled from the setting. In our use case, dark mode is not just for the aesthetics, but also affects the functionality of the software. Dark mode can be especially useful in case the musician uses this software in a dark environment. For example, night restaurants. Which normal(light) mode will be too bright for that lighting conditions.

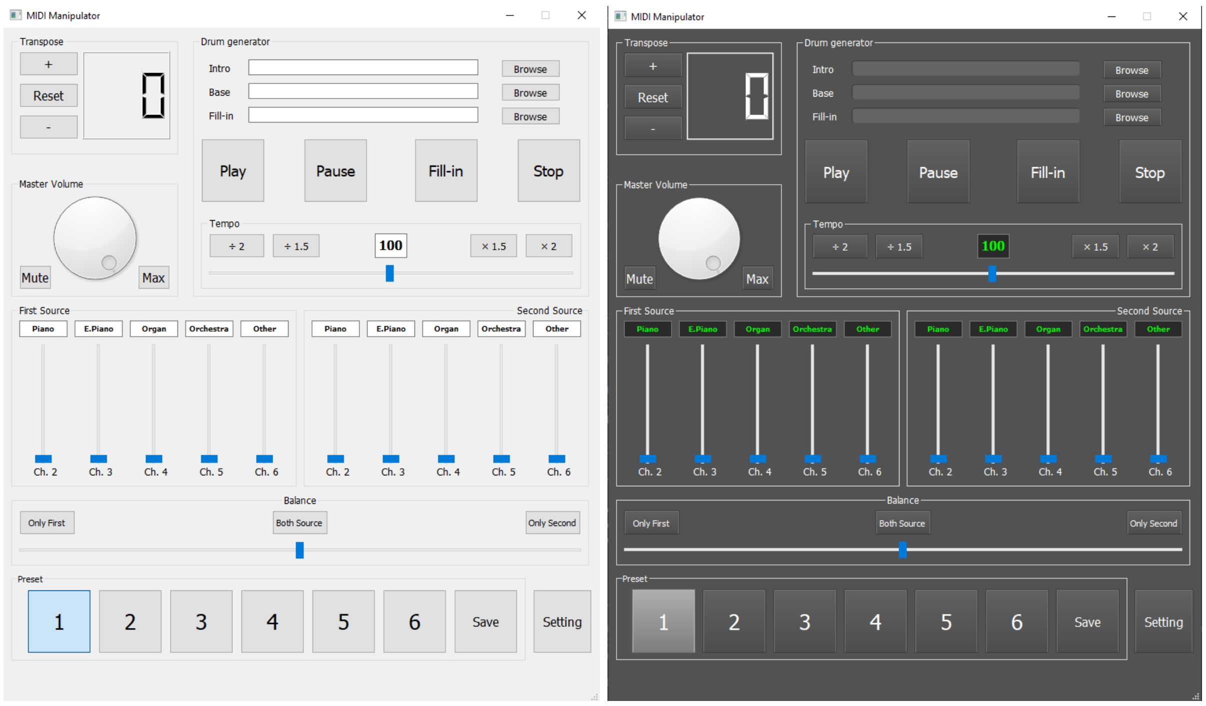


Figure 5.6: Comparison between light mode (left) and dark mode (right).

5.4 Source code

The source code, Qt file, and sample of rhythm MIDI file is hosted on GitHub.

https://github.com/Taechasit1001/AU_SeniorProject_1

Chapter 6: Conclusion

The purpose of this project was to create a MIDI keyboard using Waveform, Qt design and Python 3. The method is successfully used. Based on the analysis, it can be conducted that there are suggestions of what can be in the further stage. First, the design can be improved in the future to be more creative and friendlier looking. Secondly, as we mentioned in Chapter 5.3 User Interface and Function breakdown. The volume control and each midi channel has some function that we wish could be fixed in the future. The output channel of each port is fixed to channel 2 to 6. However, we may improve that by making them flexible for users.

References

[1] Peter Waiganjo Wagacha. Instance-Based Learning: *k*-Nearest Neighbour, 2003

[2] Tom Mitchell. *Machine Learning*. MIT Press and McGraw-Hill, 1997.

[3] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, UK, 1995. ISBN: 0 19 853864 2.

[4] *Tom Gruber <gruber@ksl.stanford.edu>*Short answer: An ontology is a specification of a conceptualization.

[5] Uche Ogbuji is a consultant and co-founder of Fourthought Inc., a consulting firm specializing in XML solutions for enterprise knowledge management applications. Boulder, Colorado, USA.

[6] Swift, Andrew. (May 1997), "A brief Introduction to MIDI", *SURPRISE*, Imperial College of Science Technology and Medicine.

[7] Milan Digital Audio: Related work (Hauptwerk)
https://www.hauptwerk.com/

[8] Yamaha Corporation.: System Design (Inspired design)
https://th.yamaha.com/th/products/musical_instruments/keyboards/electone/els-02c/index.html

[9] Bernardo Breve, Stefano Cirillo, Domenico DesiatoDepartment of Computer ScienceUniversity of Salerno84084 Fisciano (SA): An example of MIDI message
https://www.researchgate.net/publication/343709022_Perceiving_space_through_sound_mapping_human_movements_into_MIDI

[10] Ole Martin Bjørndalen by Read the Docs: Methodology (Mido)
https://mido.readthedocs.io/en/latest/

[11] Techopedia Inc.: Methodology (Qt)
https://www.techopedia.com/definition/13148/qt

[12] The Qt Company Ltd.: Methodology (Qt Designer)
https://doc.qt.io/qt-5/qtdesigner-manual.html

[13] Tracktion Software Corporation: Methodology (Tracktion Waveform)
https://www.tracktion.com/products/waveform-free

[14] Tobias Erichsen: Methodology (LoopMIDI)
https://www.tobias-erichsen.de/software/loopmidi.html
[15] Fort Wayne: Methodology (MIDI-OX)
https://www.sweetwater.com/sweetcare/articles/how-do-i-install-and-use-midi-ox-for-windows/
[16] Jamie O'Connell: Methodology (MIDI-OX)
http://www.midiox.com/