THE CONDUCTOR

User manual

Jose Figueroa
Joel Legassie
Italo Hernandez
Cuauhtemoc Guerrero

REVISIONS

Rev.	Description	Date	Approved
A	Initial document	14/Nov./2023	Admin
B	Added revisions table and request files table	12/Dec./2023	Engineer 1

Table of contents

1. Introduction

Welcome to The Conductor Glove user manual, a guide designed to help you understand and use this innovative musical interface. The Conductor represents a fusion of technology and musical art, providing a new medium for expressing creativity through movement and gesture.

Purpose
The Conductor is designed to revolutionize musical interactions by allowing users to manipulate digital audio stations through intuitive gestures. It aims to bridge the gap between traditional electronic music expression and modern technology, offering a fluid and immersive experience for musicians, creators and enthusiasts.

Parameters
This user manual provides comprehensive information on the hardware, software and operational guidelines necessary to effectively use The Conductor. It describes the components, functionalities, setup procedures and some troubleshooting procedures to ensure a smooth and satisfying experience with the device.

Versions (software)
The Conductor latest code repository version is designed to enhance the user experience and functionality. This manual covers setup and usage instructions for the current software version for a complete understanding.

Concepts of the Conductor
At its core, The Conductor utilizes a blend of sensors, connectivity modules, and sophisticated neural networks to interpret physical gestures into musical commands using gravity vector differences. The glove's hardware components, including accelerometers, gyroscopes, time-of-flight sensors, and connectivity modules, work in harmony to capture and translate subtle movements into music-related actions.

This manual elucidates the fundamental concepts behind The Conductor's design, empowering users with the knowledge needed to effectively leverage its capabilities for musical expression.

2. Components

Hardware components.

The Conductor is a testament to technological innovation, integrating a series of hardware components meticulously selected to capture and translate subtle hand movements into musical expressions. Each component plays a crucial role in the glove's functionality, from the sensors that sense gestures to the connectivity modules that ensure seamless interaction. This fusion of hardware complexities forms the basis of an innovative musical interface, allowing users to manage their creativity through the subtlest of movements. Explore below to discover the array of hardware elements that harmonize to orchestrate a seamless connection between the artist and their music.

Table of components.

Component	Quantity	Description
Microprocessor	1	Unexpected Maker TinyS3 - ESP32-S3 Transceiver; 802.11 b/g/n (Wi-Fi), Bluetooth® 5 2.4GHz Evaluation Board
I2C Expander Module	1	TCA9548A I2C Switch Interface Evaluation Board
Regulator	1	Fixed 24V to 3.3V, 1.5V @ 150mA SOT-89-3 Linear Voltage Regulator (LDO) RoHS
Accelerometers/Gyroscope	4	VLGA-12 Attitude Sensor/Gyroscope ROHS *Range, Sampling & Power* • ±2, ±4, ±8, ±12, ±16g range • 16-bit resolution • 128 to 1024 Output Data Rate • 4 µA typical Standby current • Low typical active current *Simple System Integration* • I2C interface, up to 1 MHz • 2×2×0.92 mm 12-pin LGA package
Time of Flight Sensor	1	Adafruit VL53L1X Time of Flight Distance Sensor - 30 to 4000mm - STEMMA QT / Qwiic
Battery	1	YDL 3.7V 1000mAh 503450 Lipo Battery Rechargeable Lithium Polymer ion Battery Pack with PH2.0mm JST Connector

Screen	1	Graphic LCD Display Module White OLED - Passive Matrix I^{2}C, SPI 0.96" (24.38mm) 128 x 64

Software Components.

The Conductor's functionality is intrinsically linked to its software components, which play a key role in interpreting gestures into musical commands and ensuring seamless interaction with digital audio workstations (DAWs) or music software.

Gesture recognition algorithm
At the heart of the Conductor Glove is a sophisticated gesture recognition algorithm, further processed by a neural network. This software component analyzes data captured by the glove's accelerometer sensors to interpret intricate hand movements into specific values. It transforms subtle variations in gravity into unique identifiable gesture values.

Communication protocol
The communication protocol facilitates seamless interaction between the Conductor glove and the user's computer with compatible DAW software. It establishes a reliable connection that allows real-time transmission of gesture data from the glove to the software interface. This protocol ensures a sensitive and accurate translation of gestures into musical actions.

MIDI writing software
Acting as a bridge between the interpreted gestures and the MIDI language, this component quickly converts the data processed by the glove's sensors - with data from the neural network, accelerometers and time-of-flight sensor - into accurate MIDI messages. These messages, translated in real time, are then transmitted via the established communication protocol to the connected digital audio workstation (DAW), enabling immediate integration and control of various musical parameters.

User interface and customization
The software interface accompanying the Conductor Glove provides an easy-to-use platform for configuring settings, assigning gestures to specific musical functions, and customizing the user experience. Users can customize gesture assignments, assign commands, and adjust settings to suit their preferred workflow and musical style.
Integration with Digital Audio Workstations (DAWs)

The software components are designed to integrate seamlessly with popular DAWs and music production software. This integration allows users to control parameters, trigger events and manipulate music production elements directly from the Conductor, enhancing the creative process and expanding the possibilities for musical expression.

3. Getting Started

Charging

To charge The Conductor's battery attach the battery to the power cable on the parent board. Attach a USB-C cable to The Conductor and connect it to a usb power source. Turn the switch to the On position, w=]hile the battery is charging an orange light will come on near the USB-C port. Once the battery is fully charged the light will turn green.

Power On/Off

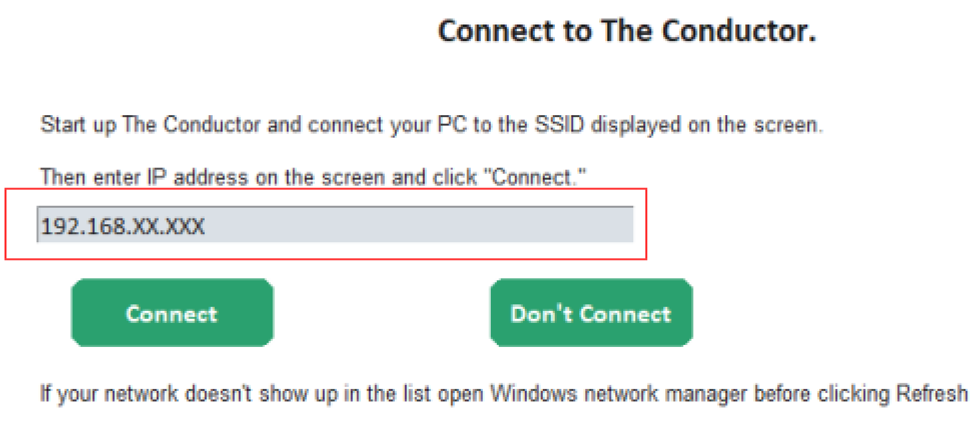
Push the switch on the front of The Conductor's main board to the right to turn on The Conductor.

Connecting to Wi-Fi

When The Conductor starts up it will create a WiFi network:

SSID name: TheConductor
SSID password: NoneShallPass
IP: 192.168.4.1

Connect your PC to The Conductor's network and start up the GUI interface. Enter the IP address above and click Connect to make a socket connection between your PC and The Conductor.



Connect to The Conductor.

Start up The Conductor and connect your PC to the SSID displayed on the screen.

Then enter IP address on the screen and click "Connect."

192.168.XX.XXX

If your network doesn't show up in the list open Windows network manager before clicking Refresh

Window to enter the conductor ip address.

Once The Conductor is connected to the PC you will have an opportunity to select a different WiFi network to connect to. Please note The Conductor can only connect to 2.4 GHz WiFi networks.

For more information on how to rename the network and change password see the Conductor's README github page theConductorCap/sections

4. Configuring The Conductor

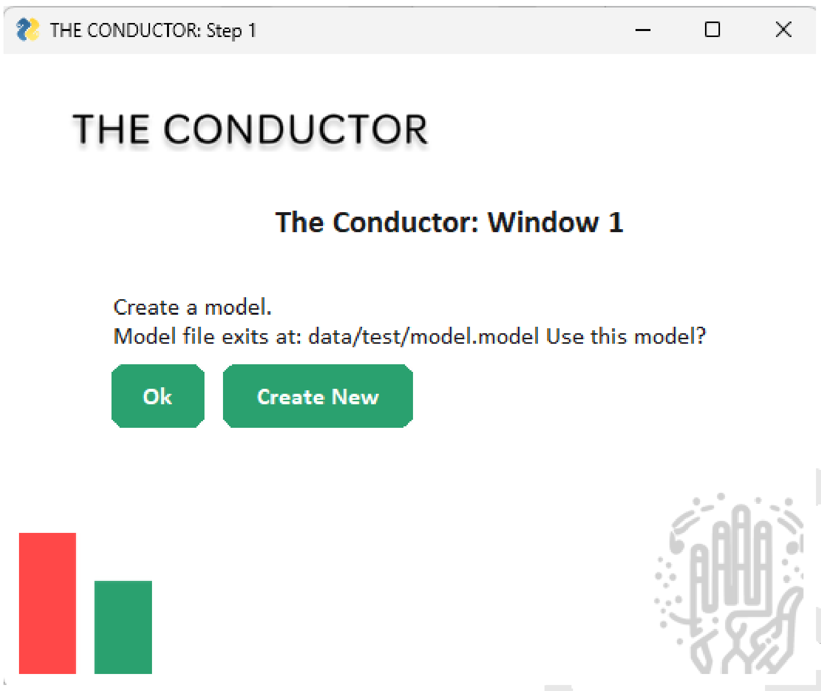
Setting Up the File System

Before using The Conductor to play music you need to train the gestures for each MIDI control you want to use. In order to store the gesture recognition data you must create a file system to contain the models.

The Conductor will automatically look for a folder called data/test inside the folder where The Conductor's PC app is saved. If you wish to use this location you must create this folder before using The Conductor app. Create a folder called "data" and then another folder inside data called "test". The data/test folder must be below the application's location in your file system like the example below:

ConductorApp >
> data
> test

In Window 1 of the GUI you will have the option of using this folder or choosing another folder.


THE CONDUCTOR: Step 1

THE CONDUCTOR

The Conductor: Window 1

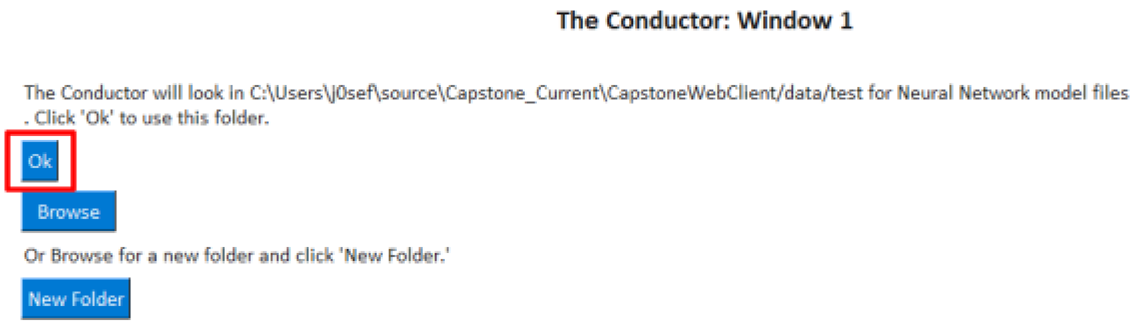
Create a model.
Model file exits at: data/test/model.model Use this model?

The Conductor GUI, create or choose an existing model.

5. Software Interface

Creating New Training Data

1. In the GUI Window 1 allows you to choose a new folder locate the data/test folder in you PC by selecting browse, or by accepting as the default location ../ConductorApp/data/test, once you have selected the location of your training data, press “Ok”



The Conductor: Window 1

The Conductor will look in C:\Users\j0sef\source\Capstone_Current\CapstoneWebClient/data/test for Neural Network model files . Click 'Ok' to use this folder.

Or Browse for a new folder and click 'New Folder.'

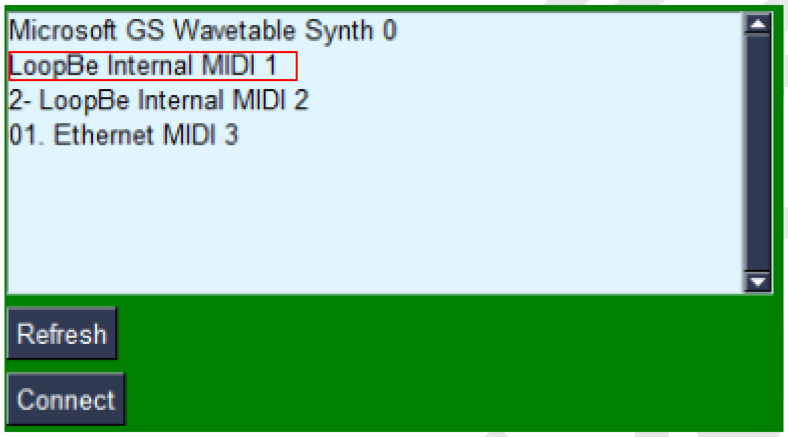
The Conductor: Window 1 creating new training data.

If there is no data already in the test folder the next window will ask you to confirm that you want to create new training data in that location.

2. Next, you can select how many hand positions you want to train, you can add a name for each position to keep track of them.

Creating MIDI Controls

1. In window 2 the first step of creating the MIDI controls is to connect the app with a virtual MIDIport. If you do not have a virtual MIDI port on your PC we recommend Loopbe1 Internal MIDI Port by nerds.de (Download ipMIDI, LoopBeAudio, Loopbe1, LoopBe30 (nerds.de)).



The Conductor GUI select MIDI device.

2. Select your virtual MIDI port and select “Connect” to open the virtual MIDI port.

3. In the next window you can select the beats per minute you want your MIDI controls to be sent at. Change this to the same BPM as you DAW and click Ok.

4. Next you can create the name for the MIDI control and press Ok. In the next page there are three options for the control.
    - Hold: The control will begin when the gesture is held for a threshold number of samples.
      For hold and transition controls you can select the gestures you want the control to activate on. For hold controls select the same gesture for the ON and OFF position, and for transition controls, select the two positions you want to transition to and from.
    - Transition: The control will begin when a transition between two selected positions is made.
    - No Action: The control will not produce any MIDI

    The sliders below each section is the gesture threshold and will determine how fast you want the gesture to be detected by the application (3 is recommended).

5. The next window selects the type of MIDI control you want to add.
    - Modulate: Create a low frequency oscillator for modulating MIDI CC messages. You can select the waveform, the rate and range of MIDI CC messages (from 0 to 127). The time of flight sensor on the palm of The Conductor will change the rate of the MIDI CC messages.
    - Arpeggiate: The MIDI control will take MIDI input from an external source and arpeggiate the notes sent to the app. The order of notes played can be set to ascending, descending, or random. The time of flight sensor on the palm of The Conductor will change the rate of notes played. You can also shift the octave that the notes will be played back in.
    - Time-of-Flight control: The time of flight sensor (Time-of-Flight) will determine the current value of a MIDI CC message. The larger the distance between the Time-of-Flight sensor and an object, the higher the MIDI CC message will be (from a range of 0 to 127)

Creating and Training Models

Once you have configured each MIDI control, Window 2.1 is used to either train or predict gestures.
To train your gestures:

1. Select the “Train button”.

2. Once you're ready to train your controls, click the “Go” button and a 3 second counter will begin.
3. Move your hand into the position you wish to train and wait for the position to complete its training.
4. Continue this process for each gesture until you have trained each gesture.
5. After you have trained each position you can check that your data/test folder is filled with all of your training files.

Using the Gestures for Music

Once you have trained each gesture you can select the Predict hand position button and press go to benign sending MIDI messages. The playback screen will show you which position is currently being held.

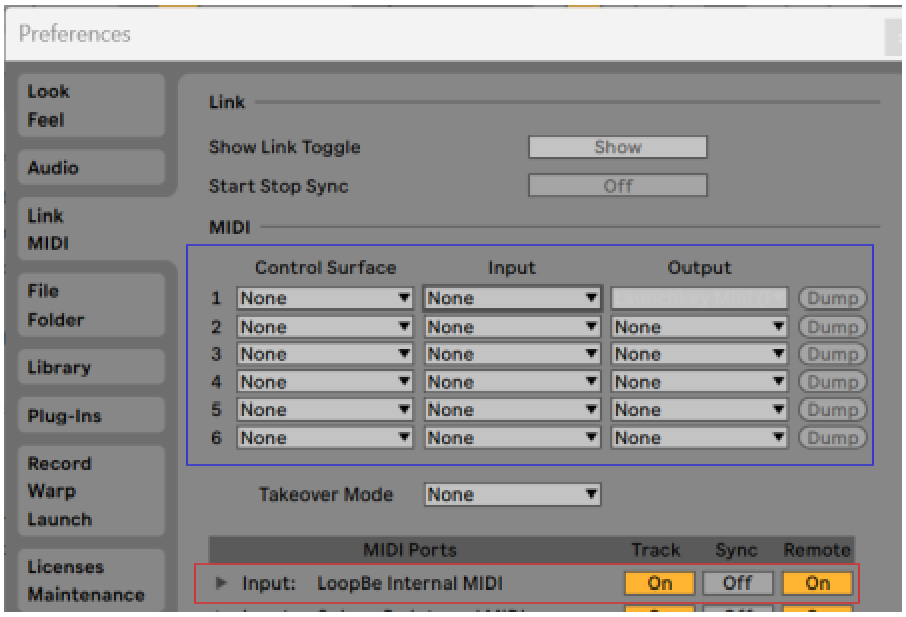
Uploading and Configuring Models

Once you have a trained model you can load it into the application again by browsing to the location of your test folder in Window 1.

6. Controlling Music

Sending MIDI to Ableton

1. Once playback has begun the MIDI arpeggiator will begin processing MIDI incoming messages and sending them to the selected virtual MIDI port. Make sure your Virtual MIDI port is turned on inside ableton by going to Preferences>MIDI >MIDI ports (see red box in picture below)



Preferences

Look Feel / Audio / Link MIDI / File Folder / Library / Plug-Ins / Record Warp Launch / Licenses Maintenance

Link

Show Link Toggle: Show

Start Stop Sync: Off

MIDI

	Control Surface	Input	Output	
1	None	None		Dump
2	None	None	None	Dump
3	None	None	None	Dump
4	None	None	None	Dump
5	None	None	None	Dump
6	None	None	None	Dump

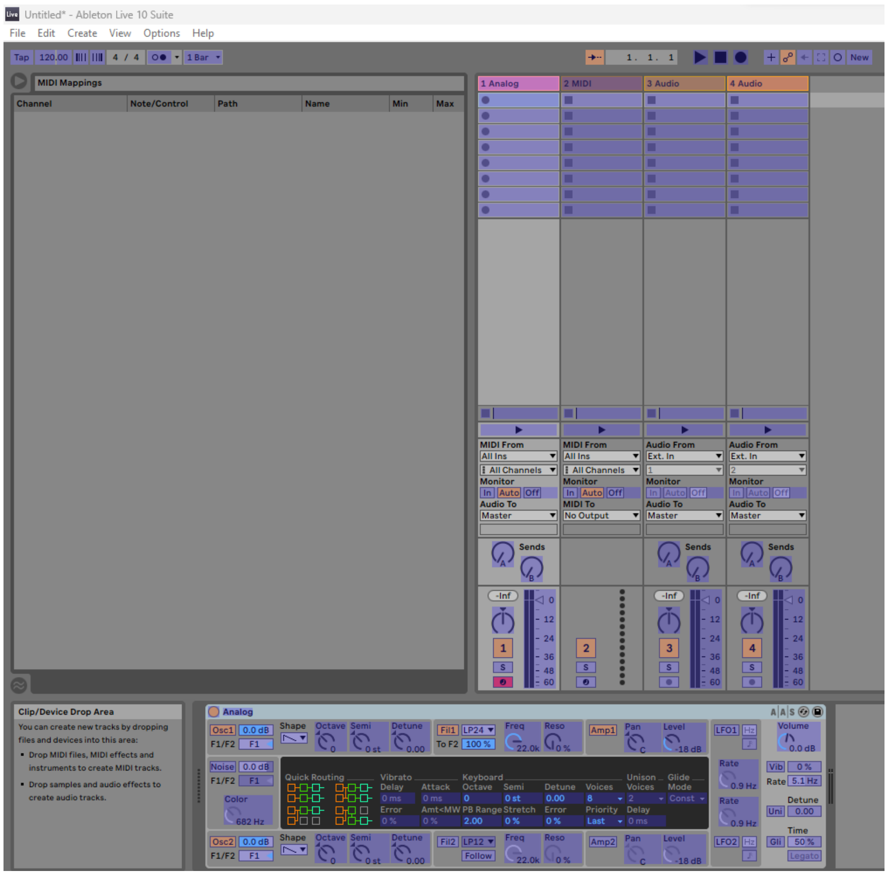
Takeover Mode: None

MIDI Ports	Track	Sync	Remote
▶ Input: LoopBe Internal MIDI	On	Off	On

Verify MIDI port is enabled in Ableton DAW.

In the previous image, make sure the MIDI device is used by The Conductor application and not connected to Ableton (the section in the blue box should be empty)

2. In Ableton map controls to controls, press CTRL + M to open MIDI mapping mode. In MIDI mapping mode all controls available to control are highlighted in purple.



Mapping controls to gestures in Ableton.

3. Click on the parameter in Ableton that you want to control then get ready to hold the gesture you want to map to Ableton.
4. Press Go and the parameter will be mapped to that specific parameter.
5. Press Ctrl + M to exit MIDI mapping mode.
6. The controls are now mapped and The Conductor is ready to perform.

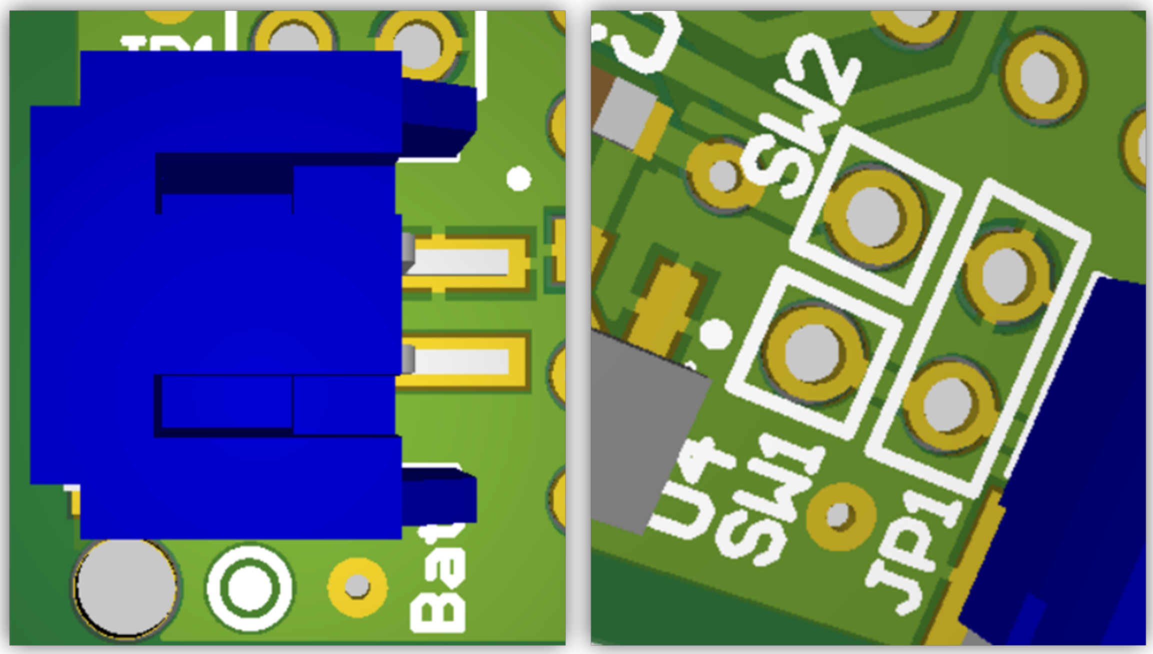
7. Troubleshooting

Hardware Troubleshooting

When The Conductor does not respond check the following common points of failure.

Handmade power JST cable (Pitch 2mm)

- Make sure the polarity is correct. It is recommended to apply continuity testing over the power pins as well as the switch.



(left) Power Pins. (right) Switch Pins

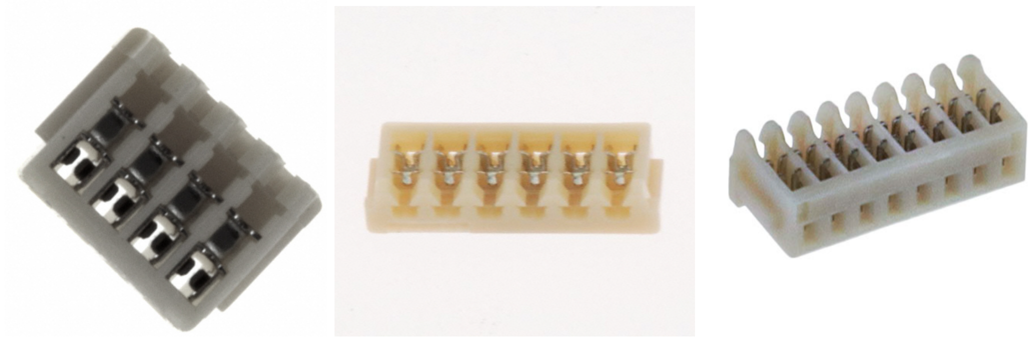
Handmade communication JST cables (Pitch 1mm)

The Conductor has different communications JST cables for the following components:

- Fingerboards (4 pins),
- Time of flight sensor (6 pins)
- LCD screen (8 pins)

To ensure correct continuity and communication follow this steps:

- Make a continuity testing over each line by using the PCB designs before being energized.
- Verify that the wires are properly connected to the connectors, if not replace the cable with a spare.
- Use the connectors recommended in our BOM. Different JST connectors can cause damage to the main board.


Different types of JST connectors.

Use the correct wiring gauge

The conductor follows the AWG standard for the correct handmade cables.

- Handmade power JST cable requires a 26.
- Handmade communication JST cables require 28, 30, or 32.

Through holes components

The Conductor uses mostly surface mounted parts. However there are three which must be soldered in place.

1) Micro controller (ESP32 TinyS3)
    - Check orientation of the development board.
    - Follow the specifications from the Datasheet about temperature for soldering.
    - Do not use headers for the current enclosure.

2) I2C Multiplexor
    - Check orientation of the development board.
    - Follow the specifications from the Datasheet about temperature for soldering.
    - Do not use headers for the current enclosure.

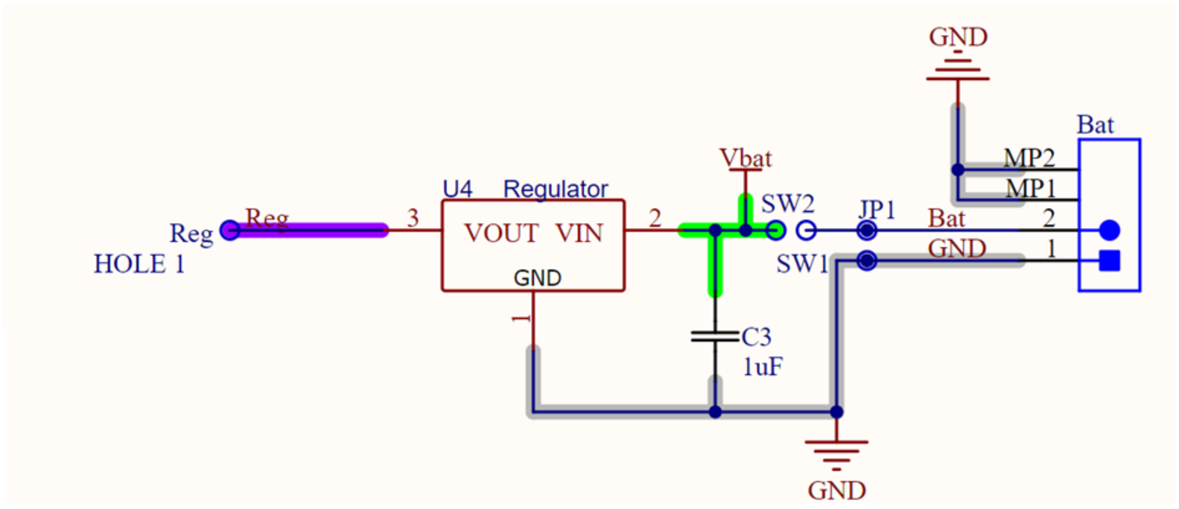
3) Switch
    - Protect the cables with Heat shrink or electrical tape.
    - Clean the board before and after the soldering.
    - Make a continuity testing after finishing soldering.

Extra Power Supply

The Conductor has a 150mA 3.3V voltage regulator that can be used to power external devices, such as a small display.
Consider the following steps for its well use:

- Check that input and output voltage regulators do not have continuity.

- Make a connection between sw1 and sw2.
- Use heat shrink for the cable.
- Apply continuity testing as before to the active components.



The Conductor regulator schematic

Software Troubleshooting

I am missing the required libraries or version of Python.
You can find the required libraries for running the application on the Conductor's github README section here https://github.com/theConductorCap/sections. The current version of Python used by the application is version 3.10.11

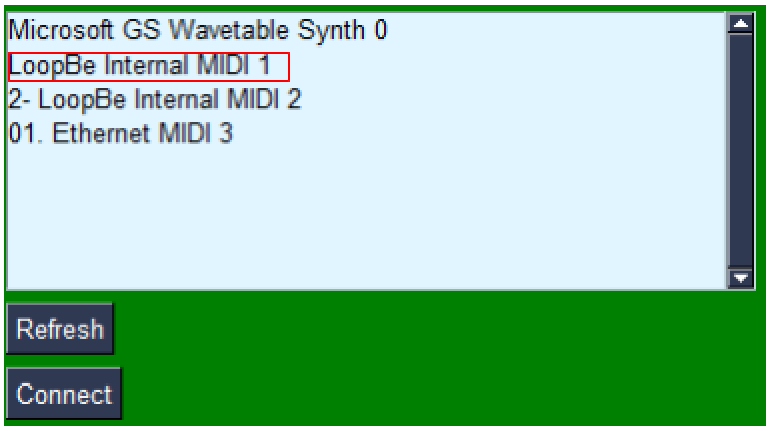
I can't connect The Conductor to the application.

- Ensure The Conductor is turned on, when The Conductor is on but not connected, the LED on the microcontroller should be green.
- Make sure your WiFi adapter is turned on for your PC, the Conductor should appear as "TheConductor" in your available networks by default
- You can reset The Conductor by selecting the reset button located on the microcontroller.

My training model did not work.
The gestures you train must be distinct enough from each other in order for the neural network to recognise individual hand positions. For best results your hand positions should have a difference of 45 degrees between the accelerometer's positions. Try to simplify your gestures until the neural network can recognize each distinct hand position.

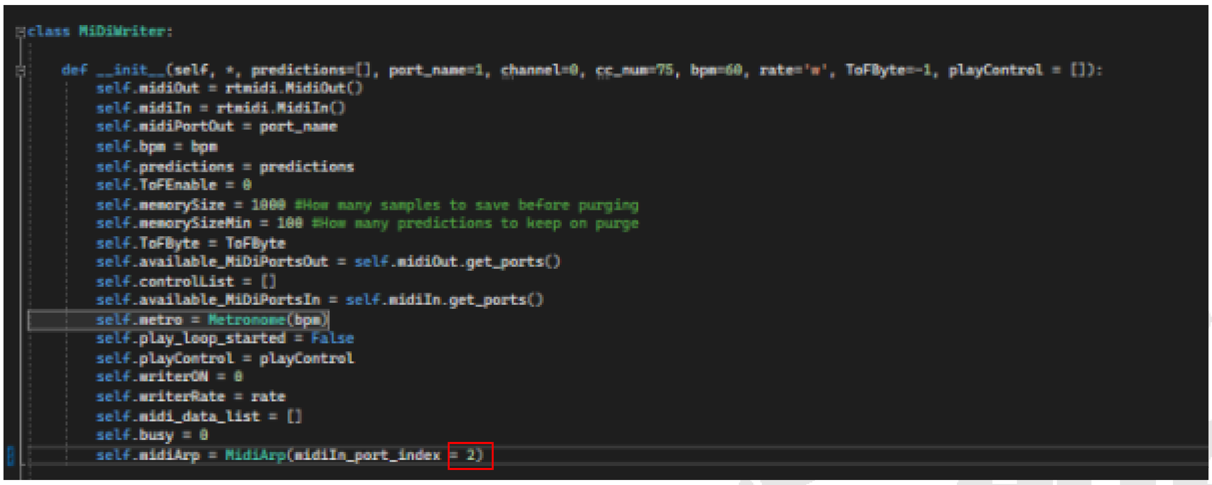
I can not send MIDI to my DAW.

- In order to send MIDI data to the DAW you need to select a virtual MIDI port to make sure you have downloaded a suitable virtual MIDI port and it has been selected inside the application.



Selection of virtual MIDI port.

- In order to monitor the MIDI messages being sent you can use a MIDI utility such as MIDIOX that will let you monitor incoming MIDI communications in your PC
- Ensure that the MIDI controller that you have connected to the Conductor Application is not already connected to your DAW, in order to modify which MIDI input device is connected to the application you can open the midiWriter.py file inside the project's code, search for 'midiIn_port_index' and modify to select the port your external MIDI controller is connected to when the program is initialized.



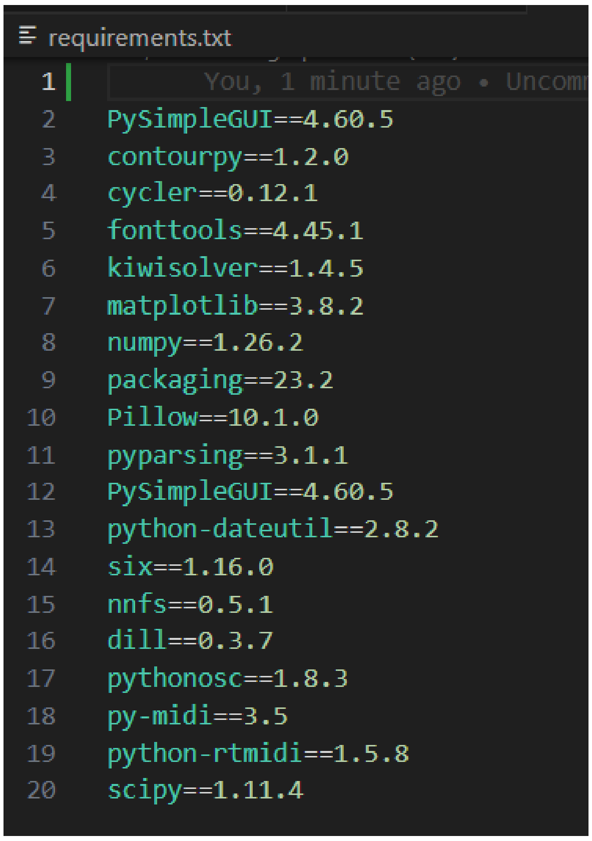
```python
class MiDiWriter:

    def __init__(self, *, predictions=[], port_name=1, channel=0, cc_num=75, bpm=60, rate='w', ToFByte=-1, playControl = []):
        self.midiOut = rtmidi.MidiOut()
        self.midiIn = rtmidi.MidiIn()
        self.midiPortOut = port_name
        self.bpm = bpm
        self.predictions = predictions
        self.ToFEnable = 0
        self.memorySize = 1000 #How many samples to save before purging
        self.memorySizeMin = 100 #How many predictions to keep on purge
        self.ToFByte = ToFByte
        self.available_MiDiPortsOut = self.midiOut.get_ports()
        self.controlList = []
        self.available_MiDiPortsIn = self.midiIn.get_ports()
        self.metro = Metronome(bpm)
        self.play_loop_started = False
        self.playControl = playControl
        self.writerON = 0
        self.writerRate = rate
        self.midi_data_list = []
        self.busy = 0
        self.midiArp = MidiArp(midiIn_port_index = 2)
```

midiWriter.py change variable midiIn_port_index to match selected MIDI input device.

8. Technical Specifications and Requirements

- Current version of The Conductor is capable of recognizing 15 unique gestures.
- Python-based code for data gathering, neural network interpretation, and MIDI translation require python interpreter 3.10.11 or newer
- Always refer to requirements.txt file on the source code, containing the required python libraries and versions.



requirements.txt

```
        You, 1 minute ago • Uncomm
PySimpleGUI==4.60.5
contourpy==1.2.0
cycler==0.12.1
fonttools==4.45.1
kiwisolver==1.4.5
matplotlib==3.8.2
numpy==1.26.2
packaging==23.2
Pillow==10.1.0
pyparsing==3.1.1
PySimpleGUI==4.60.5
python-dateutil==2.8.2
six==1.16.0
nnfs==0.5.1
dill==0.3.7
pythonosc==1.8.3
py-midi==3.5
python-rtmidi==1.5.8
scipy==1.11.4
```

Use this information as a reference, always check requirements.txt on source code repository..

- GUI (Graphical user interface) was tested on windows 10 and 11.

9. Support and Contact Information

For further assistance, contact our support team at support@theconductor.tech or visit our website www.theconductor.tech/support.

Appendix A: Glossary

AWG
American Wire Gauge (AWG) describes single-strand solid wire.

BPM

Beats per minute (BPM) represents the music tempo, and it is directly proportional to this one.

DAW
A Digital Audio Workstation (DAW) is a software application that allows users to produce, edit, record, and perform music.

GUI
A Graphic User Interface (GUI) is a user interface which makes interactions between electronic devices.

I2C

A Inter-Integrated Circuit (I2C) is a serial communication protocol used for connecting microcontrollers with peripherals such as sensors.

JST
The Japanese Solderless Terminal (JST) is a standard of electrical connection used widely across a spectrum of electrical applications.

LED
A light-emitting diode (LED) is a semiconductor device that emits light when an electric current is passed through it.

MIDI CC
MIDI CC are used for continuous control of musical parameters that do not contain note information.

MIDI Standard
Musical Instrument Digital Interface (MIDI) is a standard for controlling and communicating with electronic music instruments.

PC
Personal Computer (PC).

PCB
A printed circuit board (PCB) is the board with non-conductive material where it can be installed wires, electronic components, traces, and others.

RoHS
The Restriction of Hazardous Substances (RoHS) is a Directive on the restriction of the use of certain hazardous substances in electrical and electronic equipment.

SSID
Stands for Service Set Identifier, which is a 32-character sequence that uniquely identifies a wireless network (i.e. wifi network name).

Time-of-Flight
A sensor that detects distance.

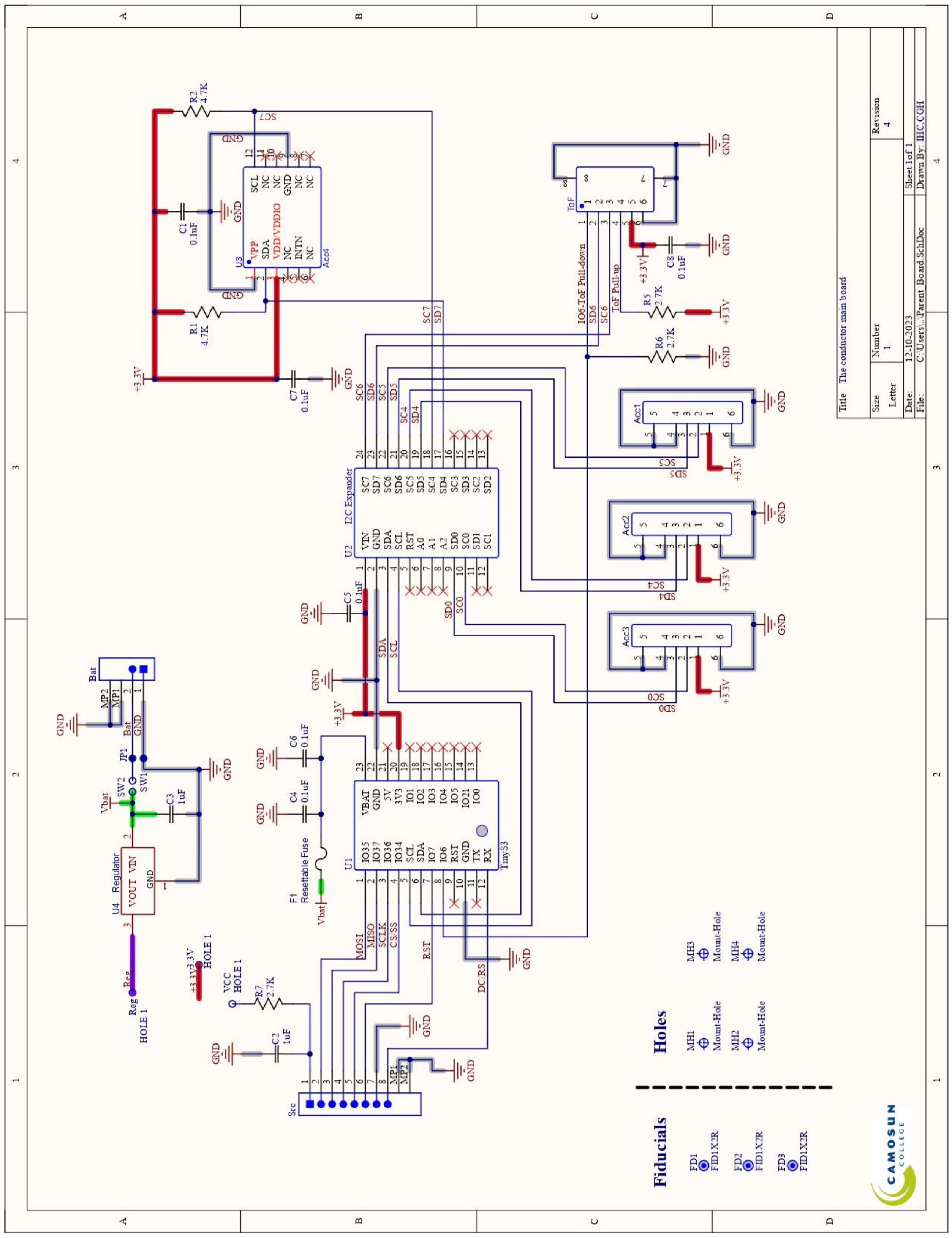
TinyS3
A development board in the ESP32 product line developed by Unexpected Maker.

USB-C
USB-C is an industry-standard connector for transmitting both data and power on a single cable.

WiFi
Wireless Fidelity (WiFi) uses radio waves to provide wireless.

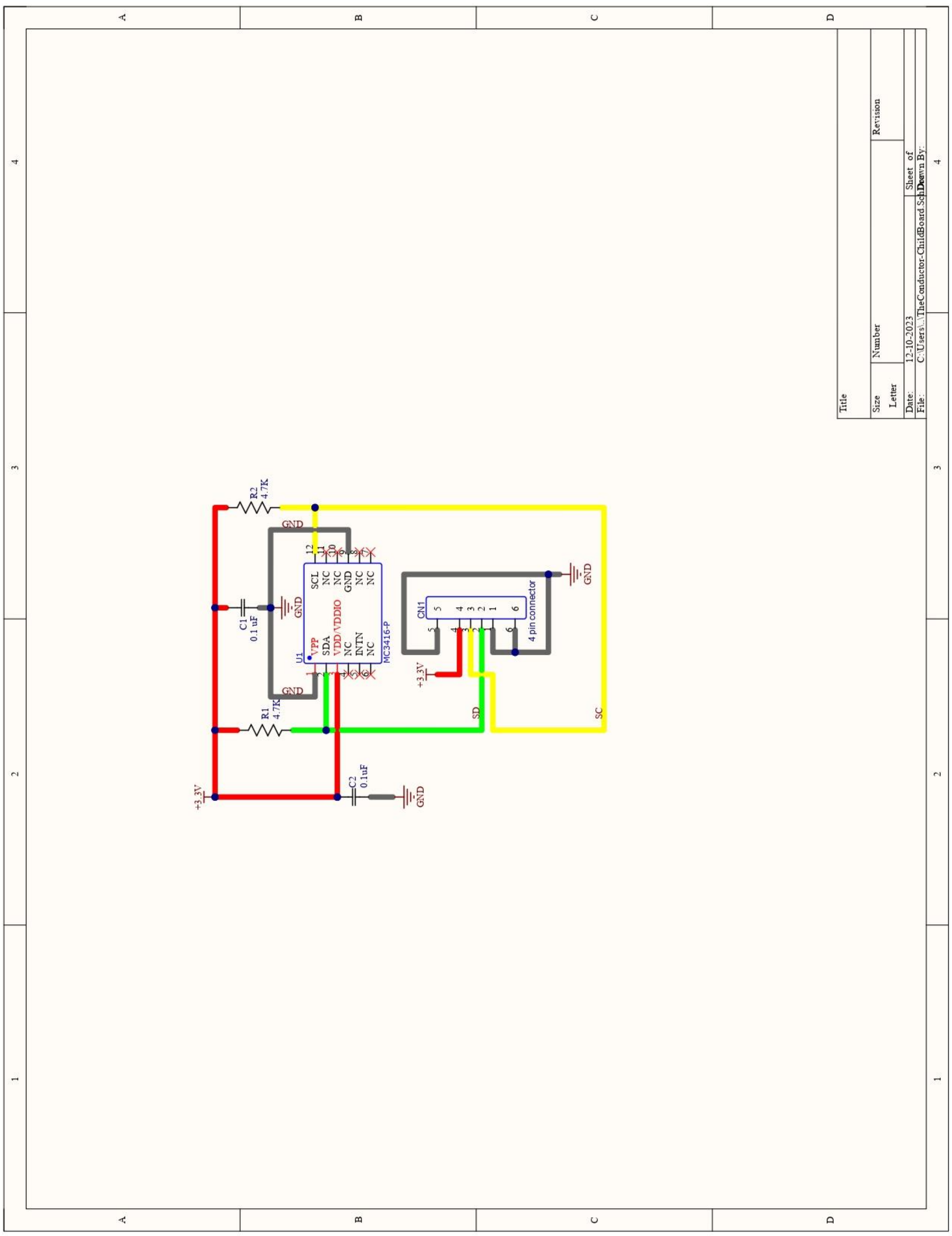
Appendix B: Schematics

Parent Board



Then Conductor's parent board version 2 schematic.

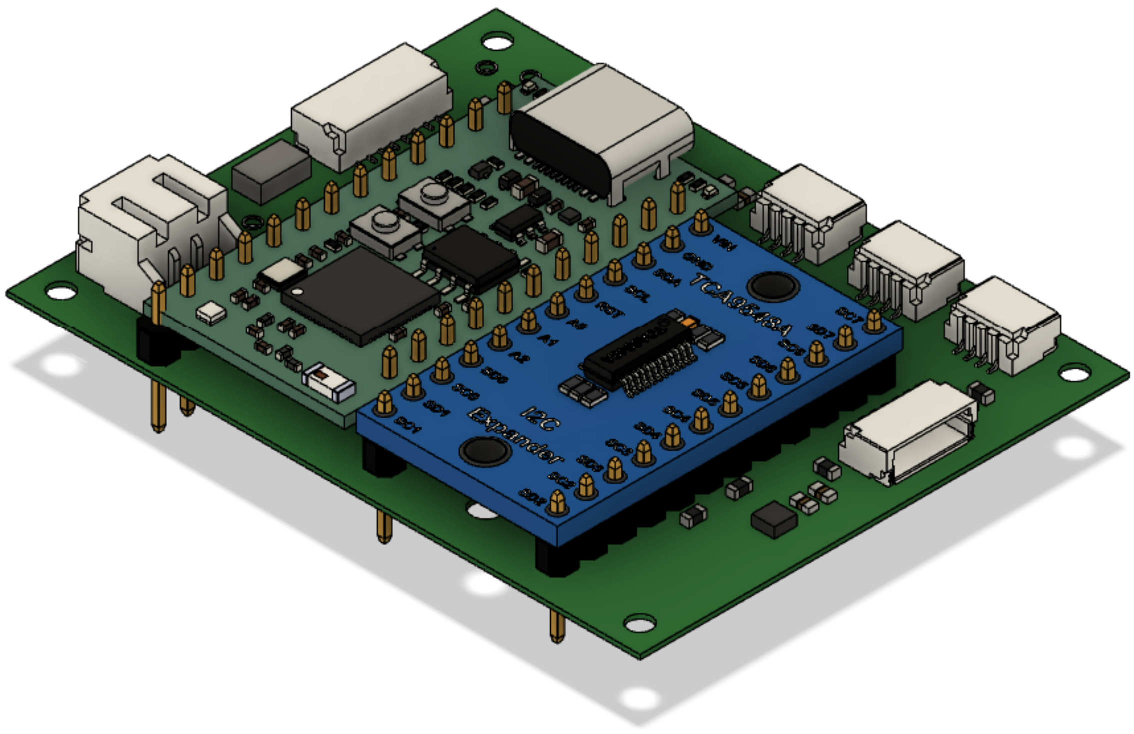
Child Board



The Conductor's child board version 2 schematic.

Appendix C: PCB's

Parent Board



Child Board