Raindrops on a Tin Roof: Exploring the Cyberinet

Matthew A. Bardin
- PHD Candidate

Louisana State University - Experimental Music & Digital Media

04/18/2023

1

The main event will begin soon.

Please take your seat.

Access the event program, various examples and music scores with this QR code.

Raindrops on a Tin Roof: Exploring the Cyberinet

Matthew A. Bardin
- PHD Candidate

Louisana State University - Experimental Music & Digital Media

04/18/2023

Augmented Instruments

WHAT IS THE CYBERINET?

What is an Augmented Instrument?

- Augmented Instruments are any instrument that has been altered or enhanced from its base capabilities in some way.
- This can be done in a variety of methods.
- Augmentation methods can be any combination of manual, automatic, acoustic, mechanical, or electronic.
- Many modern augmentations include electronic components, sensors and computers.

- Reid, Sarah, Sithi-Amnaui, Sara, & Kapur. Ajay. (2018). *Women who Build Things: Gestural Controllers, Augmented Instruments, and Musical Mechatronics*. Proceedings of the International Conference on New Interfaces for Musical Expression, 178-183.

Recent Eras of Augmented Instruments

Pre-20th Century

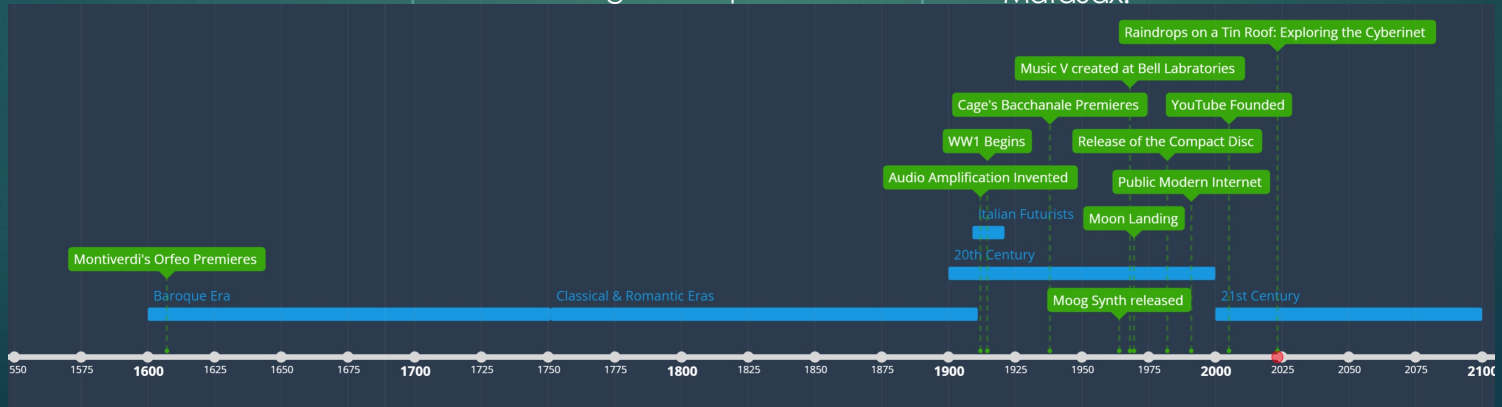
- Completely acoustic/mechanical.
- Generally looking to improve an instrument or crate a new one.
- Brass Mutes, Player Pianos, Calliope.

20th Century

- Early electronics improve over course of century
- Early computer augmentations begin to appear.
- Prepared Piano, Amplification, Effect Pedals, Recording Developments.

21st Century (so far)

- Largely computer-based.
- Exploring new timbres and instrument functionality with new technology.
- Cyberinet, MIGSI Trumpet, SABRe, Overtone Violin, MataSax.

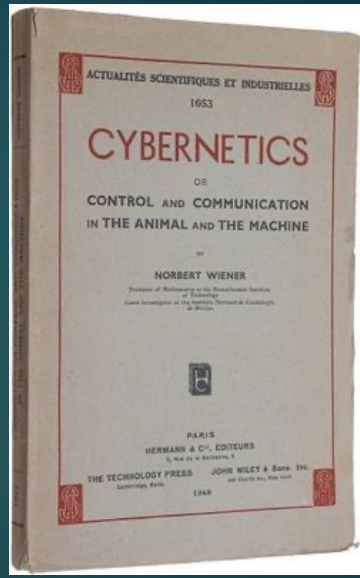


A General Historical Context. Visit 120years.net for a more detailed timeline.

Cybernetic + Clarinet

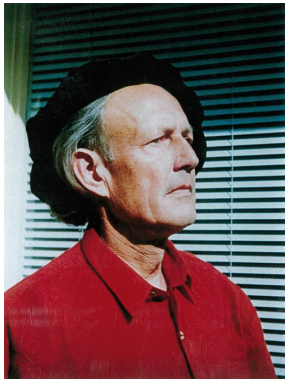
What are Cybernetics?

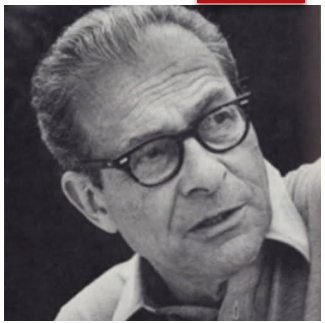
- Cybernetic: *The science of communications and automatic control systems in both machines and living things* – Oxford Dictionary
  - Definition ca. 1948 from the Greek "Kubernetes" meaning captain, pilot, conductor, etc.
  - Norbert Wiener first defined the term in his book *Cybernetics: Or Control and Communication in the Animal and the Machine*. It came from the concept of a sailor steering a ship, then seeing the ship move and responding to that in a feedback loop.
  - Later refined to: Circular causal and feedback mechanisms in biological and social systems.
  - Cyborg: portmanteau of cybernetic and organism. A being with organic and inorganic body parts. (ca. 1960)
  - Many crossovers and definition variances stemming from the core concept of automated control and feedback loops.
- The Cyberinet is used to collect performance data and automatically control the audio processing and other electronic elements of an electroacoustic performance.



Cybernetic Music

- Several composers have utilized cybernetics in some way in their music. A small sample includes:
- Roland Kayn *(1933-2011):*
  - Composer focusing on Cybernetic Concepts and electronics.
  - Mix of random and intentional aspects.
  - Allows the sound to grow and evolve on its own.
- Herbert Brün *(1918-2000):*
  - Composer and computer scientist.
  - Utilized early computing languages in the creation of new music.
  - Creates new timbre synthesis techniques similar to granular synthesis.
- Pauline Oliveros *(1932-2016):*
  - Composer and post-war experimental musician.
  - Inspired by Cybernetic concepts in the 1960's.
  - Led to experiments with sounds creating sounds, tape, and other methods of creating and listening to music.





How does the Cyberinet factor in?

- The Cyberinet is a sensor array designed to be easily adapted the instrument for electronic music without impeding the performance of the instrument.
- The Cyberinet can still be used as a regular clarinet barrel while powered off.
- Utilizes cybernetic principles (automatic control of various sound processes) to create an augmented instrument.
- Easily implement a feedback loop where the sound can effect the player making the sound.

Why use this Instrument?

- Easy to implement.
- Does not reduce the Clarinet's capabilities.
- Can easily add harmonic capabilities to the Clarinet.
- Intuitive controls.
- Can be used by performers, composers, and programmers.

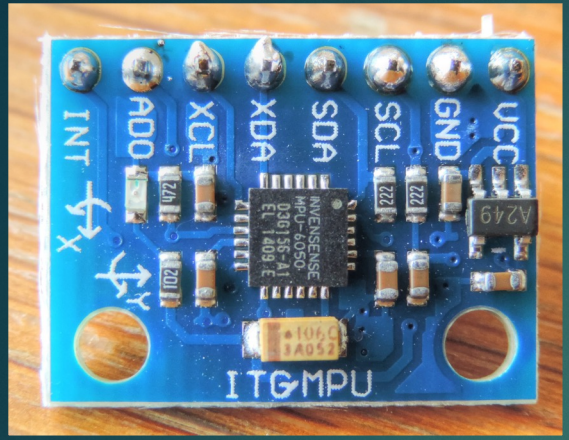
Creating the Cyberinet

OEM Components

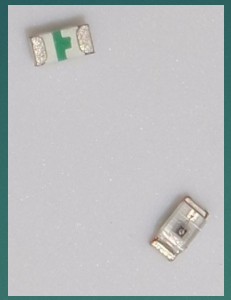
OEM Components are standardized to be compatible with each other.

- The Cyberinet is comprised of a main unit and smaller secondary unit.
- Main Unit:
  - 1200mAh Li-Po battery to power the device.
  - TP-4056 Type C board. (controls the charging and power distribution)
  - MPU-6050 board. (6 axis gyroscope and accelerometer)
  - SDP-31 Qwiic Connect board. (differential air pressure/flow and temperature)
  - ESP-32 DEVKIT board. (microcontroller and transmitter)
- Secondary Unit:
  - 2 momentary switches with independently programmable LEDs.

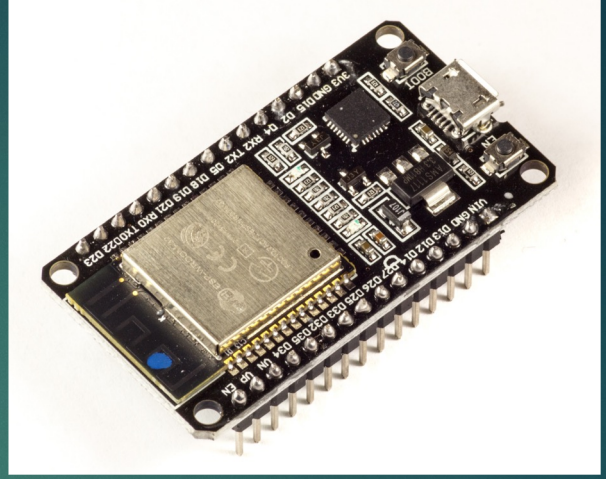
Also, the Cyberinet has 2 programmable LEDs, which are not OEM.



MPU 6050 (Gyro + Accelerometer)



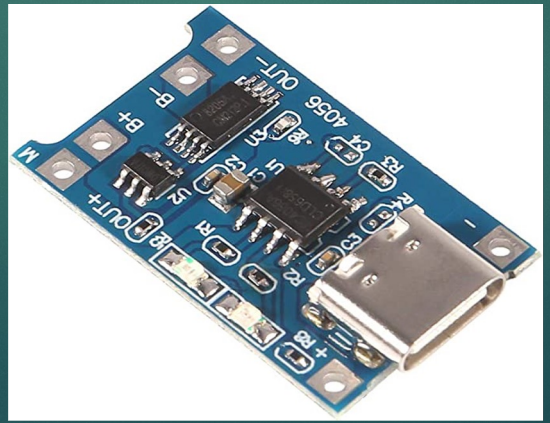
0602 Size SMD LEDs



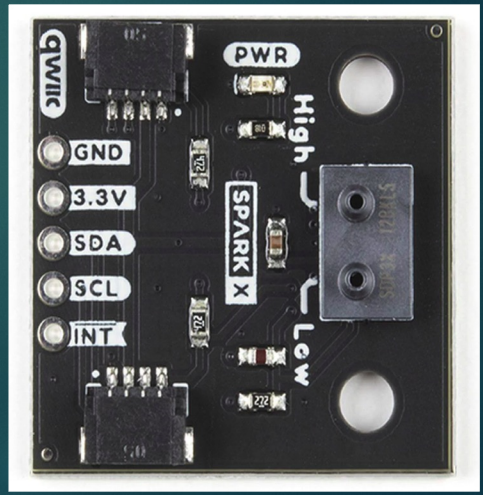
ESP-32 Microcontroller

Pushbuttons

TP-4056 (Power Distribution)

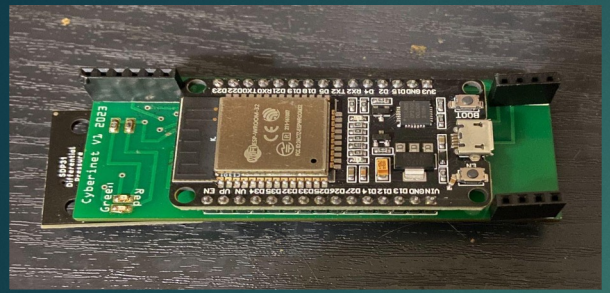


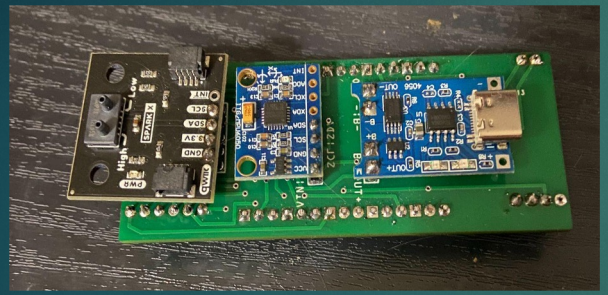
SDP-31 Differential Pressure Sensor



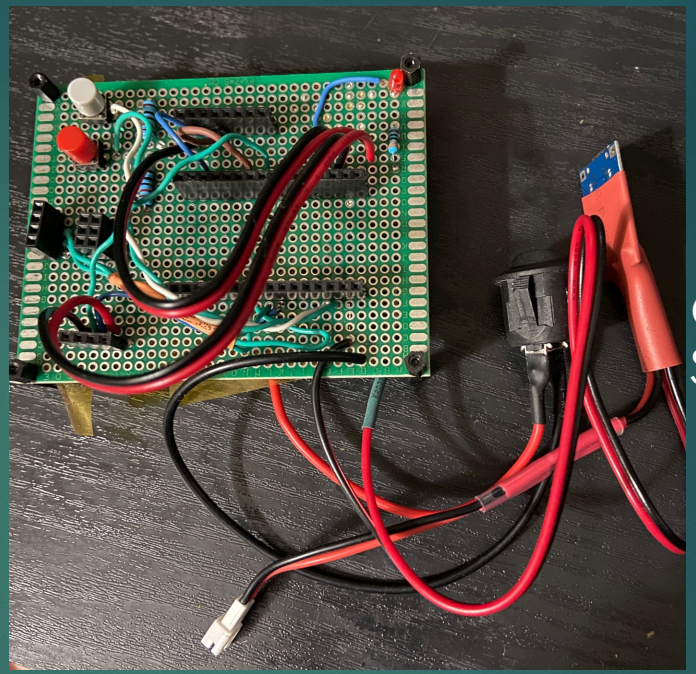
Version 1 Components

Top



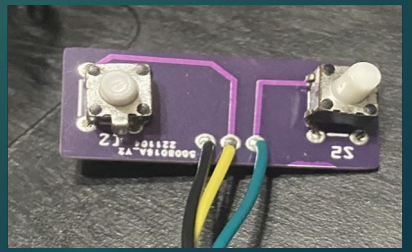


Bottom



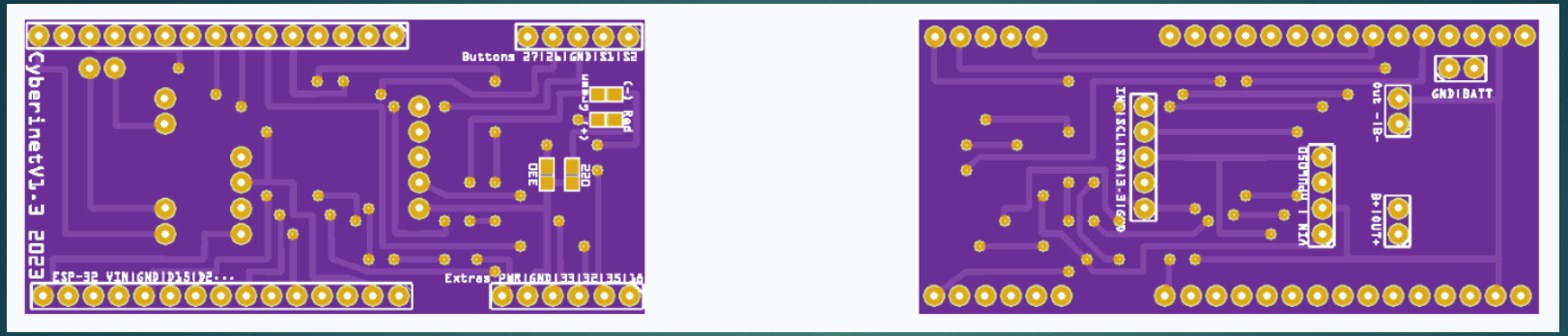
Original Prototype Socket Board

Buttons

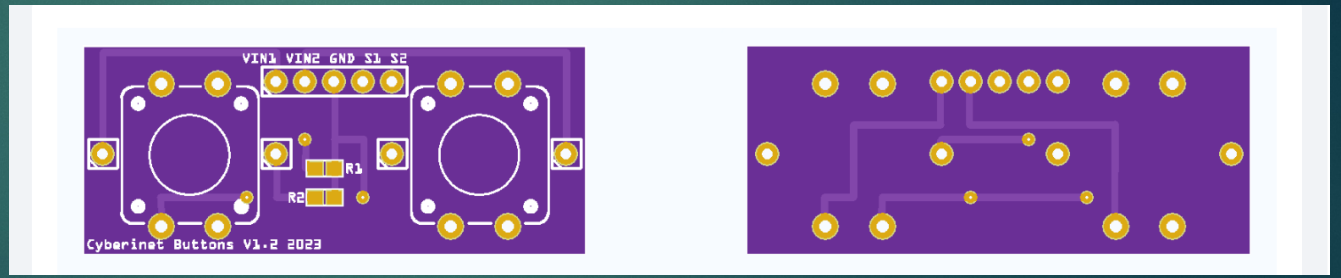


Main Unit & Button Expansion Prototypes

Main Board



Button Expansion



Version 1 Final PCB Diagrams

Future Optional add-ons

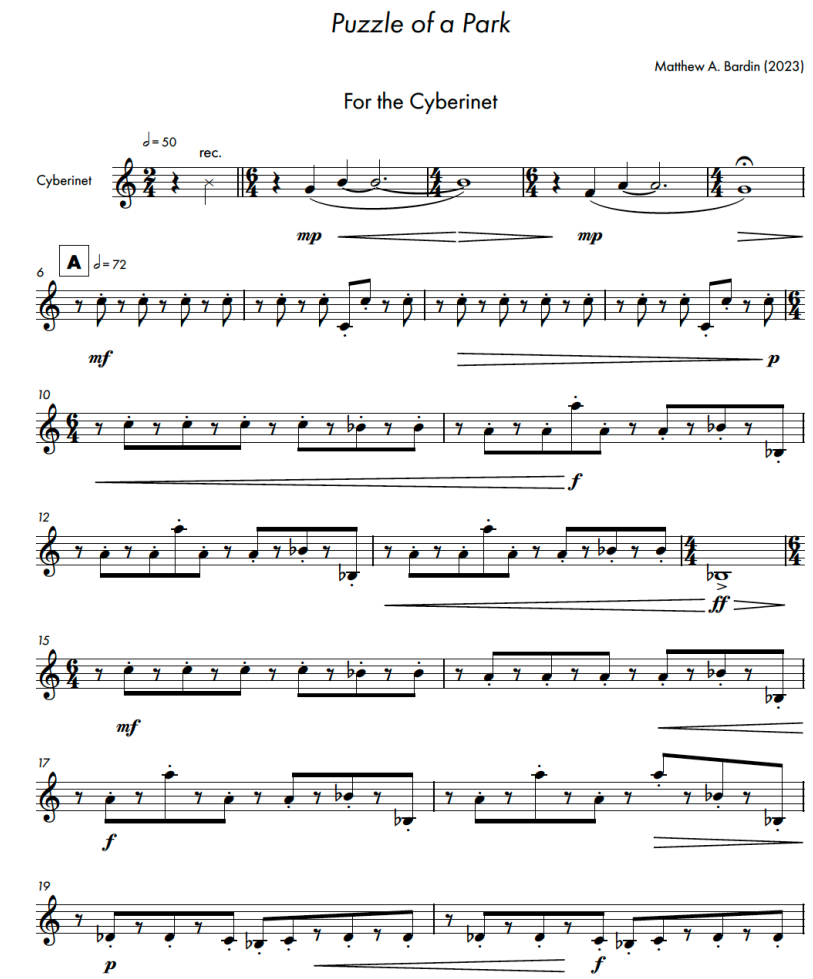
- To further expand the functionality of the Cyberinet, multiple optional expansion units contain a variety of other sensors and can be utilized through the same expansion port.
- These include:
  - Microphone
  - Finger pressure sensors
  - Secondary movement sensor
  - Speaker
- All of these are still in development, with the microphone closest to being functional.

Music

Puzzle of a Park (2023)

- All works heard today are the FIRST pieces written for this instrument.
- Written to utilize the optional button add-on.
- Performer triggers the on-stage microphone & playback of recordings with a single button.
- Functions similarly to a loop pedal.
  - Allows performer to multi-track and harmonize with themselves.

Adam Cope- Cyberinet

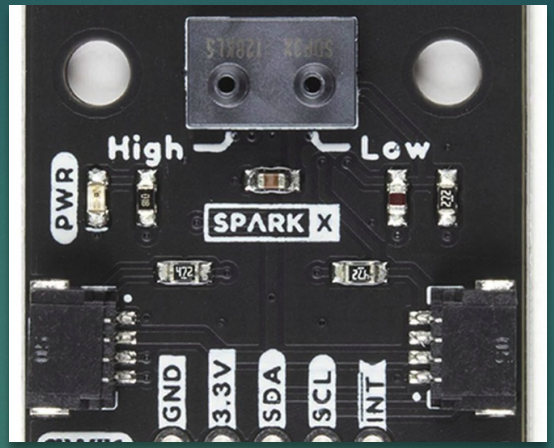


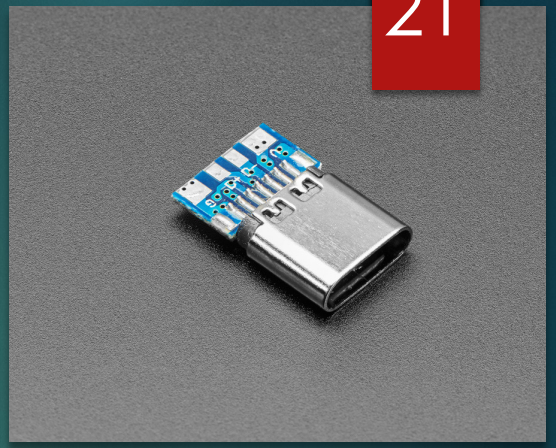
Puzzle of a Park opening page

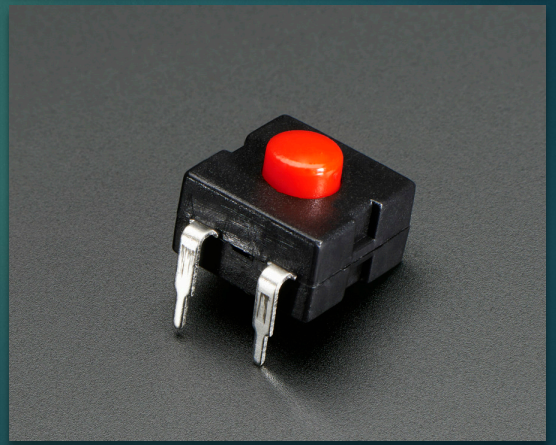
Assembling the Cyberinet

Assembling the Cyberinet

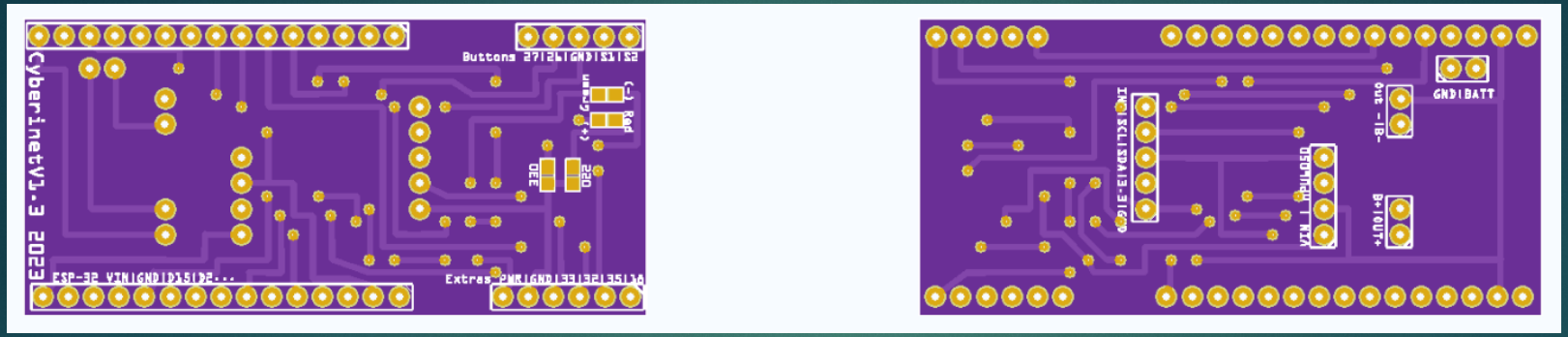
1:	Attach LEDs and Resistors (SMD Components) to the PCBs.
2:	Solder the USB-C and JST connectors. These are supported by the outer case.
3:	Solder the larger components directly to the PCBs. The microcontroller and SDP chip should be elevated with pin headers.
4:	Add the power switch. When done, connect the battery Glue/Sand as needed.



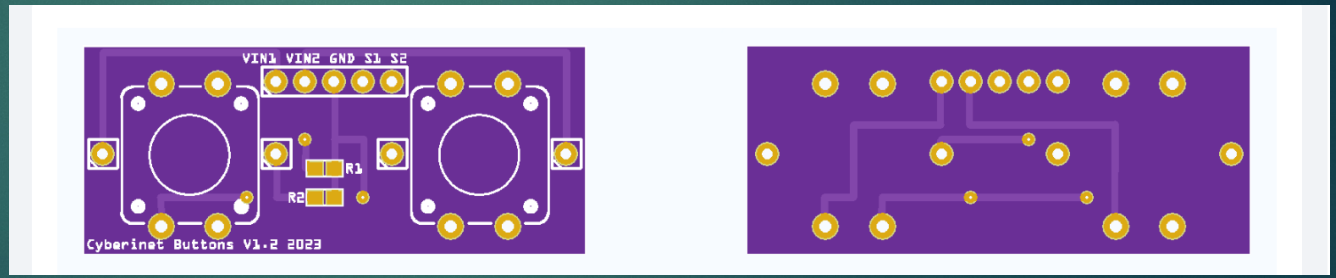




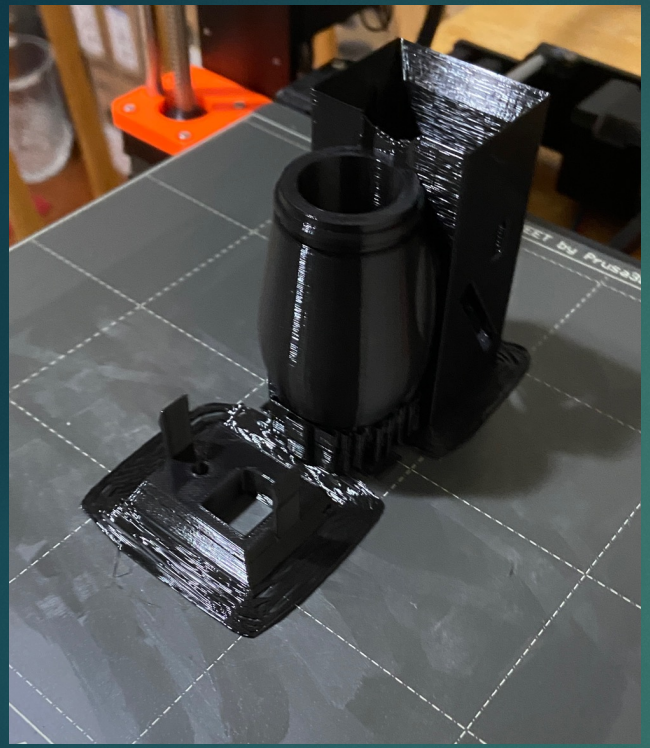
Main Board

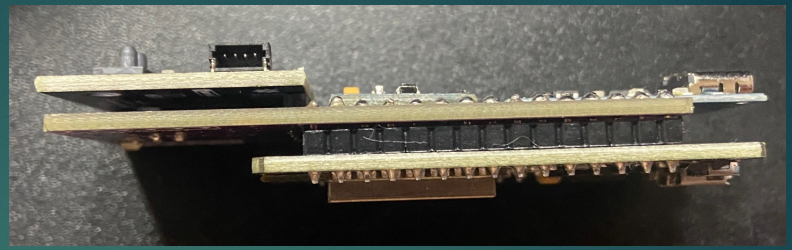


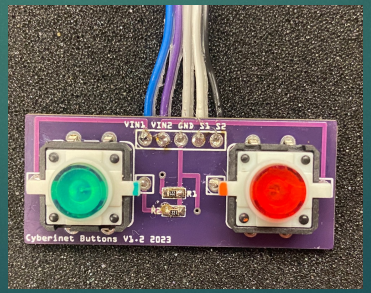
Button Expansion



Version 1 Final PCB Diagrams







Version 1 with Button Board

How the Cyberinet works

Collects data

- The Cyberinet's .ino code and attached sensors manage the collection of the performance data.
  - SDP31: Measures differential air pressure between the inside and outside of the instrument. Small tubes connect to the sensor to facilitate this. Also measures temperature.
  - MPU6050: Measures performer movement in 6 axis.
  - Buttons: triggerable by the performer.
  - Other Expansion Add-Ons.

```
void get6050() {

  ////////////Collect Data
  Wire.beginTransmission(MPU_ADDR);
  Wire.write(0x3B);  // wake up device
  Wire.endTransmission(true);

  Wire.requestFrom(MPU_ADDR, 6); //Request Accel Registers (3B - 40)
  while (Wire.available() < 6);
  AcX = Wire.read() << 8 | Wire.read(); // operator first bitshifts the sensor value by 8 bits,
  AcY = Wire.read() << 8 | Wire.read(); // then does a bitwise OR to compare to raw data
  AcZ = Wire.read() << 8 | Wire.read(); // enbd result is a combined original and bitshift

  Wire.beginTransmission(MPU_ADDR); //I2C address of the MPU
  Wire.write(0x43); //Starting register for Gyro Readings
  Wire.endTransmission();
  Wire.requestFrom(MPU_ADDR, 6); //Request Gyro/Accel Registers (43 - 48)
  while (Wire.available() < 6);
  GyX = Wire.read() << 8 | Wire.read();
  GyY = Wire.read() << 8 | Wire.read();
  GyZ = Wire.read() << 8 | Wire.read();


  ////////////Format Data
  rotX = GyX *  0.007633587786; // gravity and rotation offsets
  rotY = GyY *  0.007633587786;
  rotZ = GyZ *  0.007633587786;

  gForceX = AcX * 0.00006103515;
  gForceY = AcY * 0.00006103515;
  gForceZ = AcZ * 0.00006103515;


  ////////////Transmit Data
  digitalWrite(msgLED, HIGH); // turn on LED when when transmitting
  SerialBT.print("GyroX "); // label
  SerialBT.println(rotX); // value

  SerialBT.print("GyroY "); // label
  SerialBT.println(rotY); // value

  SerialBT.print("GyroZ "); // etc.
  SerialBT.println(rotZ);

  SerialBT.print("AccelX ");
  SerialBT.println(gForceX);

  SerialBT.print("AccelY ");
  SerialBT.println(gForceY);

  SerialBT.print("AccelZ ");
  SerialBT.println(gForceZ);
  digitalWrite(msgLED, LOW);// turn off led when done transmitting
}
```

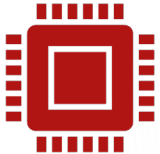
```
void getButtons() {
  ////////////Collect Data
  button1State = digitalRead(button1);
  button2State = digitalRead(button2);

  ////////////Transmit Data (every frame. CNET.receive Max Object filters out repetitions.)
  digitalWrite(msgLED, HIGH); // turn LED on when tranmitting data
  SerialBT.print("B1 "); // label
  SerialBT.println(button1State); // value
  digitalWrite(blLED, button1State); // on-board led state matches button state
  // repeat for other button
  SerialBT.print("B2 ");
  SerialBT.println(button2State);
  digitalWrite(blLED, button2State);
  digitalWrite(msgLED, LOW); // turn off led when done transmitting
}

void getAir() {
  ////////////Collect Data
  airFlow.readMeasurement(&diffPressure, &temperature);

  ////////////Transmit Data
  digitalWrite(msgLED, HIGH); // turn on LED when tranmitting
  SerialBT.print(F("AirP ")); // lebel
  SerialBT.println(diffPressure, 2); // value

  SerialBT.print(F("Temp "));
  SerialBT.println(temperature, 2);
  digitalWrite(msgLED, LOW); // turn of LED when done transmitting
}
```

Data Collection Code

Transmits wirelessly

The main unit's ESP-32 board the transmitting as well as the data collection.

By utilizing the Arduino library "BluetoothSerial" the DEVKIT can transmit serial communications through a Bluetooth connection. The device name is "CyberinetV1X" and currently requires no password to connect.



These values can be read by any serial terminal program on the host device. The Max objects utilize an additional node script to receive the data more reliably.

Performer Opinions

- We will take a few moments to ask Adam about his thoughts with using the device.
- What are your honest opinions on the usability, flexibility, and anything else you would like to mention?

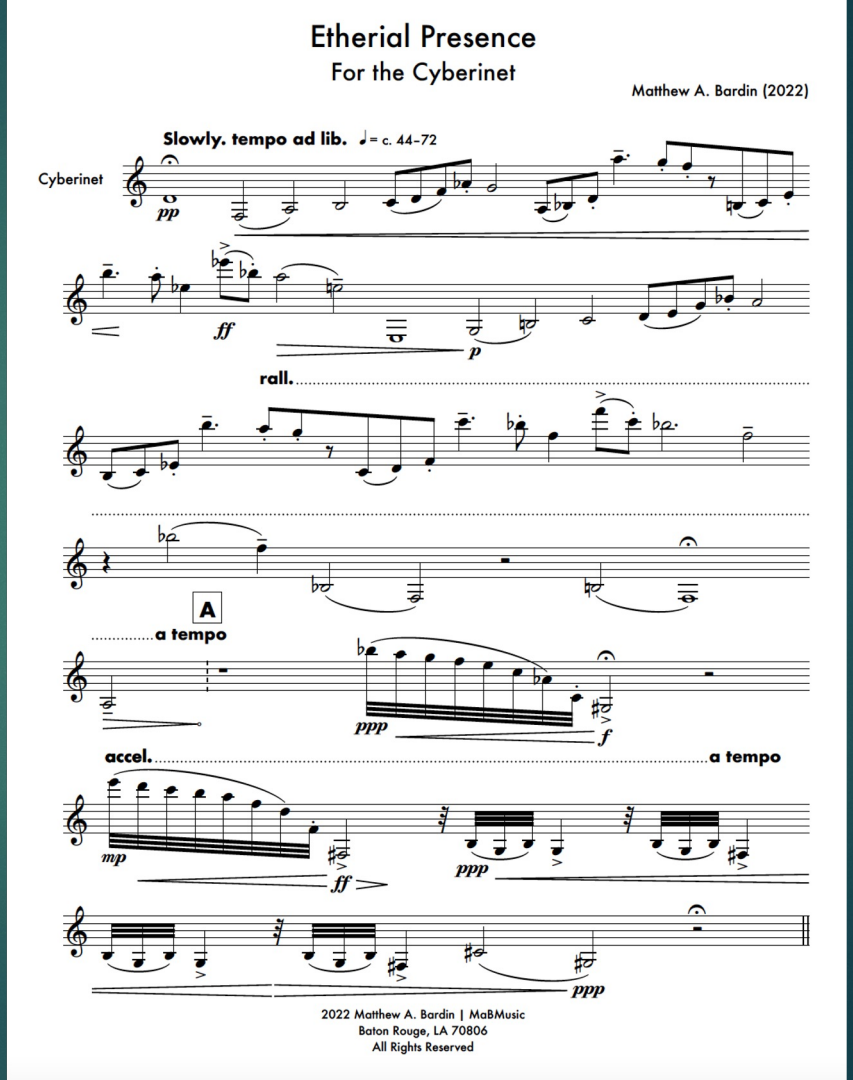
Thanks Adam!

Music

Ethereal Presence (2022)

- Utilizes gyroscope and airflow sensors.
- As the performer moves, the position of their instrument changes the pitch content of the accompaniment.
- Computer synthesis has 2 main textures:
  - Filtered pink noise to provide backing texture
  - Modified FM synthesis to provide accompaniment to the soloist
- Music alters between different styles to encourage a variety of performance movements.
- Wide tempo variety helps encourage a non-identical performances.

Adam Cope- Cyberinet

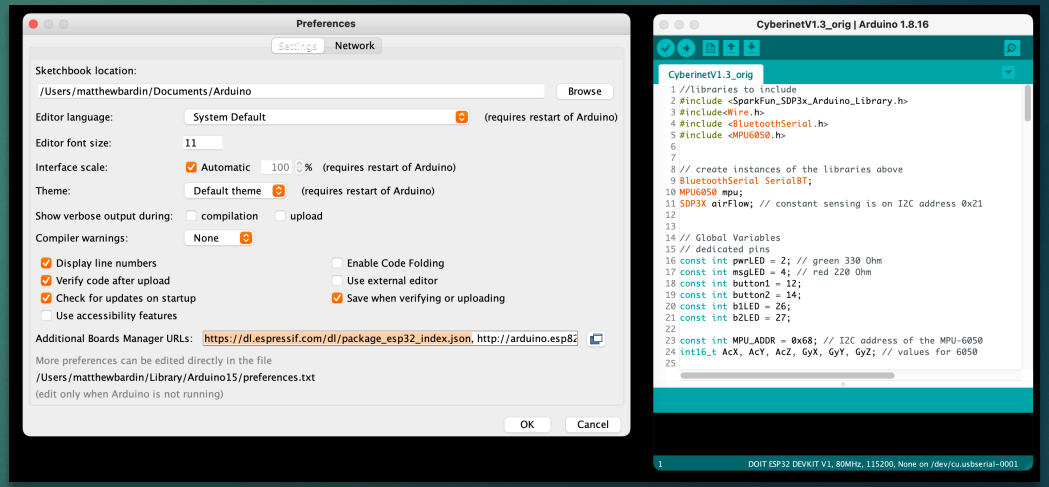


Ethereal Presence opening page

Computer Processing

Driver Installation

- A few drivers are needed to utilize the ireless communications.
  - ESP-32 Drivers: These can be easily installed via the Arduino IDE program.
  - Node and NPM: Can be installed from the Node website and managed via the Terminal.
- These extra steps only need to be run once, whenever it is the first time a computer has paired with a Cyberinet.
- NPM can be managed from the Max objects as well.
- Looking to streamline this step in future revisions.



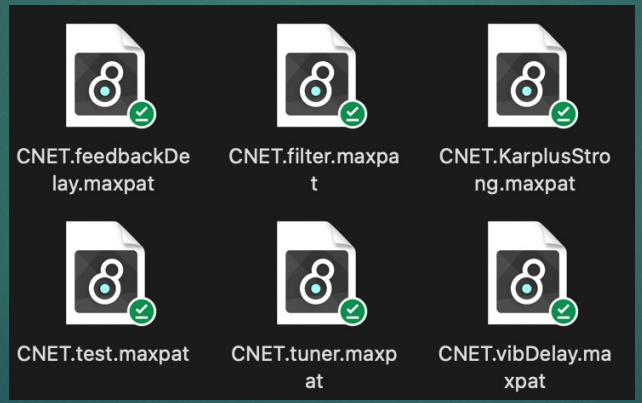
Processing/Manipulation in Max

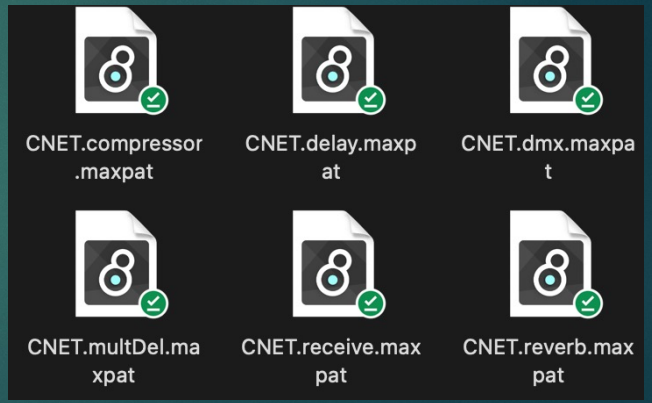
The main goal of the Cyberinet is to collect and transmit the sensor data.

From there it is largely up to the user to decide what they want to to with the data.

To help with that I have also created a series of Max patches that can be used to control various elements of a Max patch including:

- Reverb
- Various Delays
- Gain
- Debug testing tool
- Tuner
- Playback Control
- DMX Control
- More coming soon!





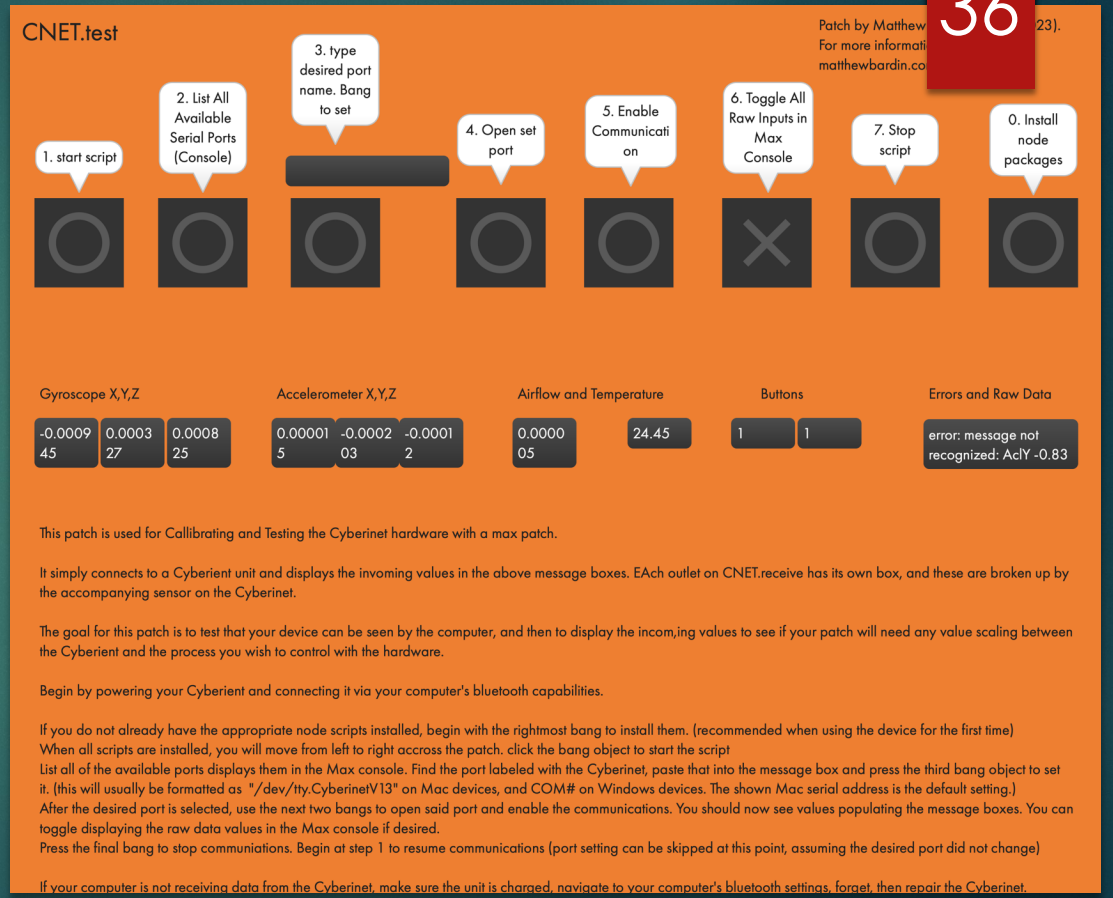
Early effects are simpler alterations to the Clarinet's sounds and software control. More complex uses to come as the hardware continues to be refined.

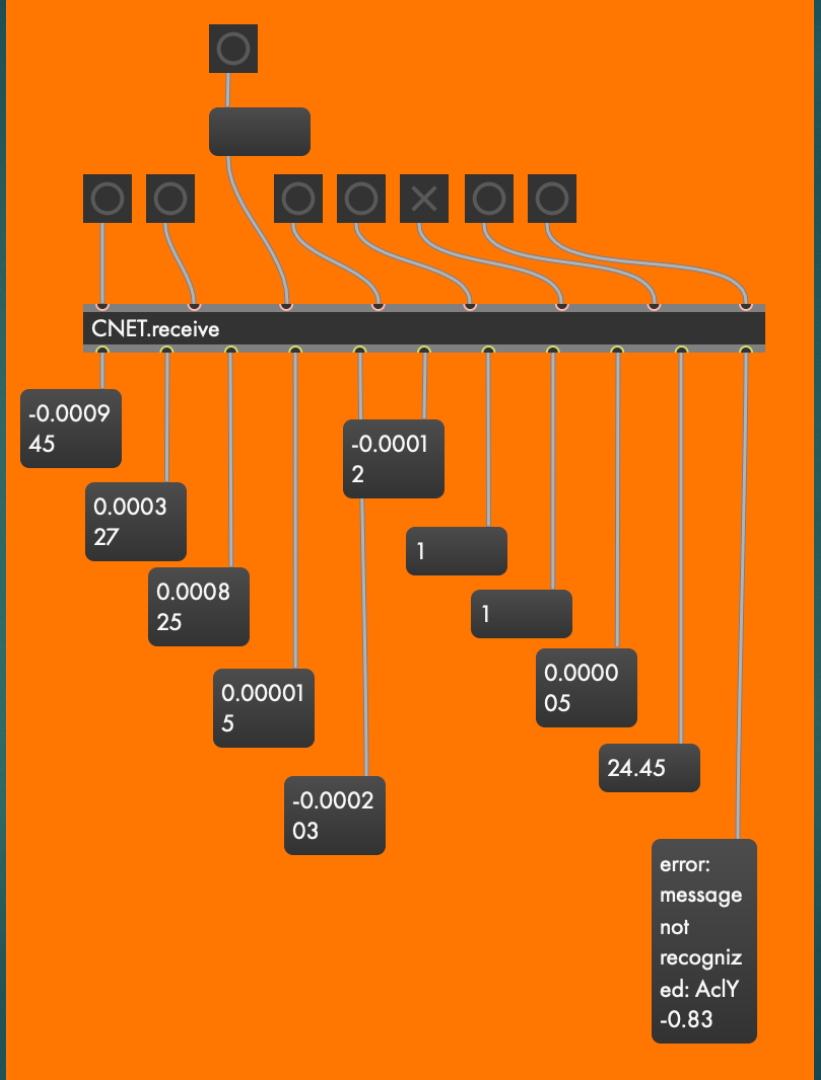
QR Code reminder

- You can find screenshots of a selection of max patches here.
- I'll keep this up for a few seconds if anyone needs it.

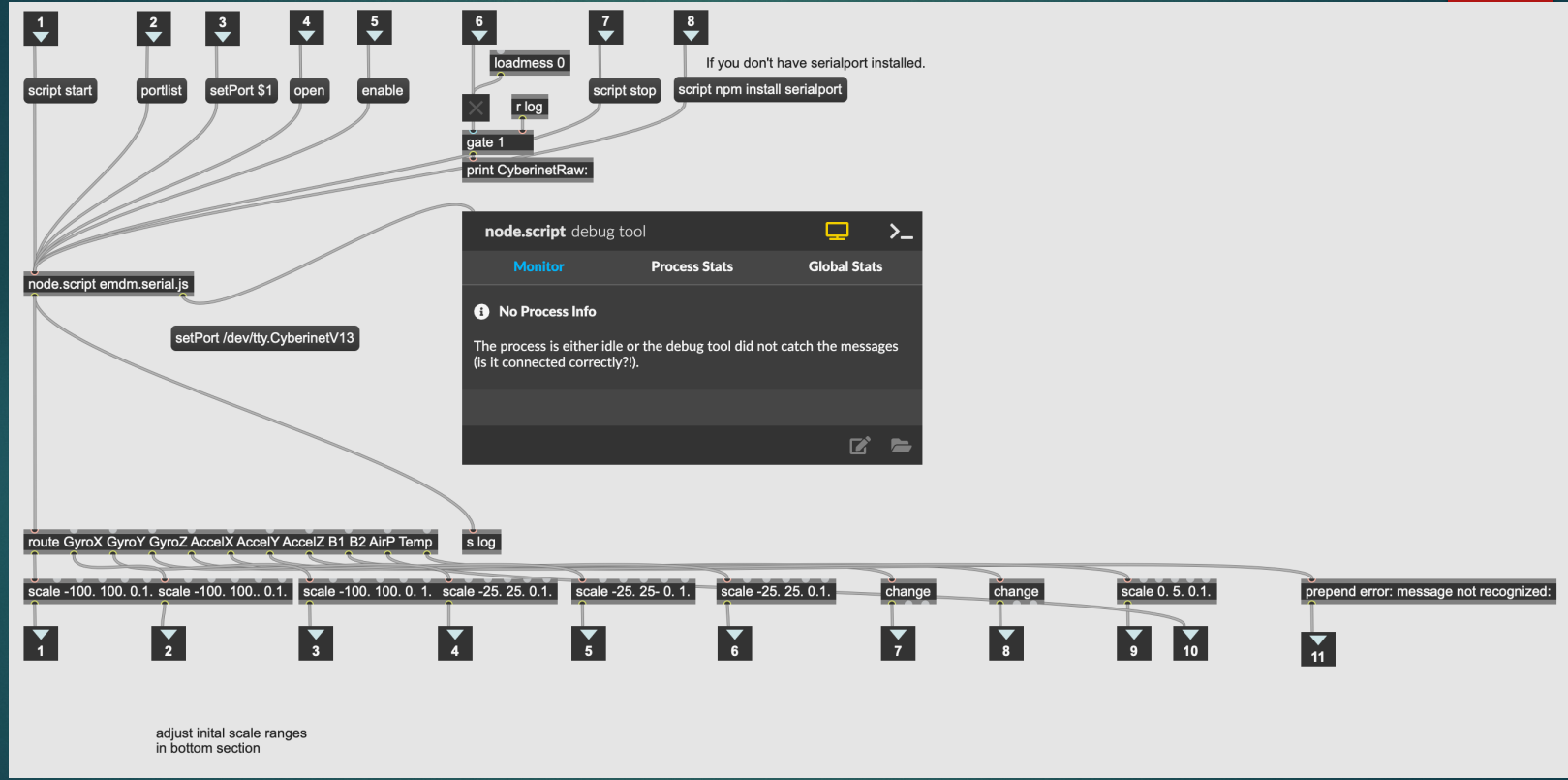
Max Patches Continued

- CNET.receive receives all the data from the Cyberinet, sorts and scales it for use, then outputs the numbers to unique object outlets.
- Port defaults to the expected MacOS formatting. If everything matches, steps 2 and 3 can be skipped.





CNET.receive



CNET.recieve patch inner workings

Sending data to other programs

- Data from the Cyberinet can be read by any Serial Monitor.
- Data is formatted for Max by default.
- Programs like Max can then convert the messages to different formats and transmit them to other devices and software as needed.

Online Resources

Github Repo

Contains all code, diagrams, and other information

github.com/mbardin/Cyberinet

Lecture Recital	documentation photos and slide updates	5 hours ago
Max Tools	draft of pan object	last week
Music Composition Scores	file name changes	yesterday
V_1.3	forgot to add file drafts to last push. uwu	yesterday
other documentation	documentation photos and slide updates	5 hours ago
.DS_Store	documentation photos and slide updates	5 hours ago
.gitattributes	Initial commit	5 months ago
.gitignore	updates	yesterday
Cyberinet Lecture Recital.pptx	documentation photos and slide updates	5 hours ago
LICENSE	Initial commit	5 months ago
Matthew Bardin Lecture Recital Progr...	working on presentation slides	5 days ago
README.md	slide and readme updates	5 hours ago

Code, Patches, Diagrams, & Resources.

Cyberinet units will be available for purchase at a future date.

All materials are open-source.

A person can build their own unit using the online resources.

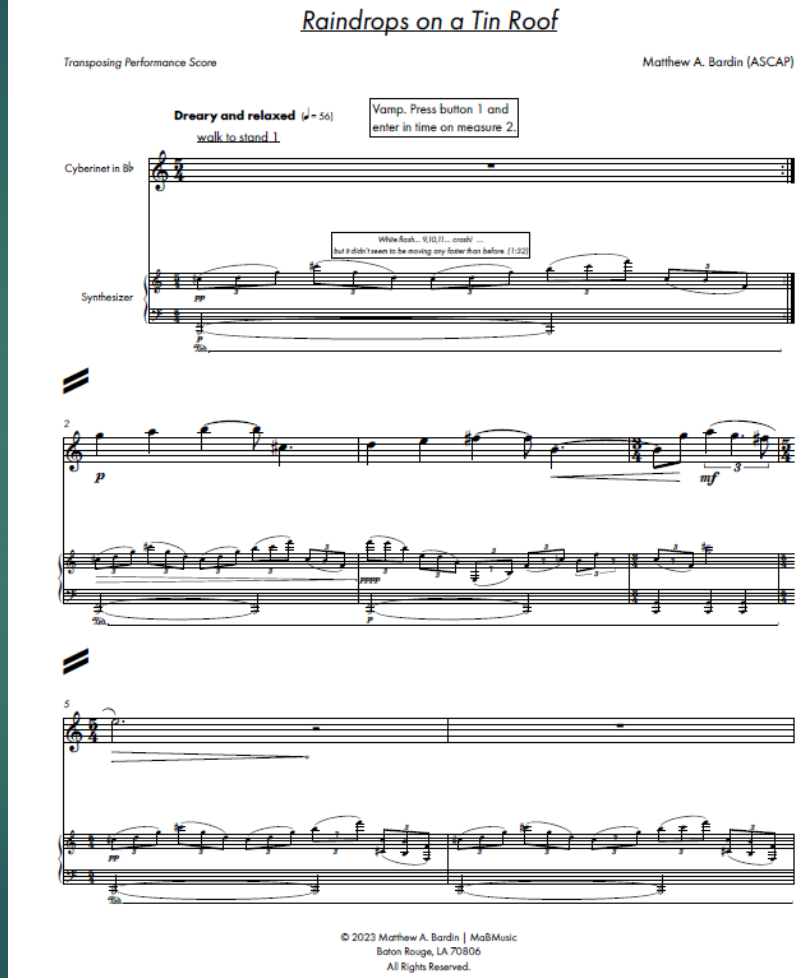
Music

Raindrops on a Tin Roof (2022-2023)

- Utilizes all the included sensors:
  - Gyroscope/Accelerometer
  - Airflow
  - Buttons
- The performer cues up various sound files and musical sections using the buttons.
- Other sensors are used to control the processing of the sounds.
- Sounds range from relaxing to extremely chaotic.
- Text by Matthew A. Bardin

Adam Cope - Cyberinet
Matthew Bardin - Narrator

Raindrops on a Tin Roof
opening page



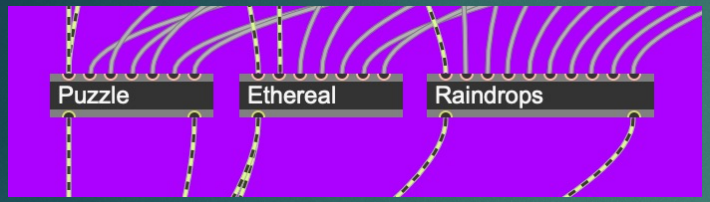
Future Directions

Potential Future Directions Include:

- Optimal serial communications and installations for Windows and Mac operating systems.
- Completing and refining the options for expansion units.
- Custom-made PCBs without OEM to make smaller units.
- OSC Formatting for easier multi-platform use cases.
- Permanent embedding within a clarinet.
- Making versions for other instruments.
- Password locking units to avoid miss-connections.

Final Notes

All the compositions are available in Max.



- Each of the patches used to make these performances has been created as an object in Max. Performers can download these from the GitHub repo for performance, or experiment with their own processing.

Scores, patches and other info available at matthewbardin.com

Questions?

Thank You for coming!