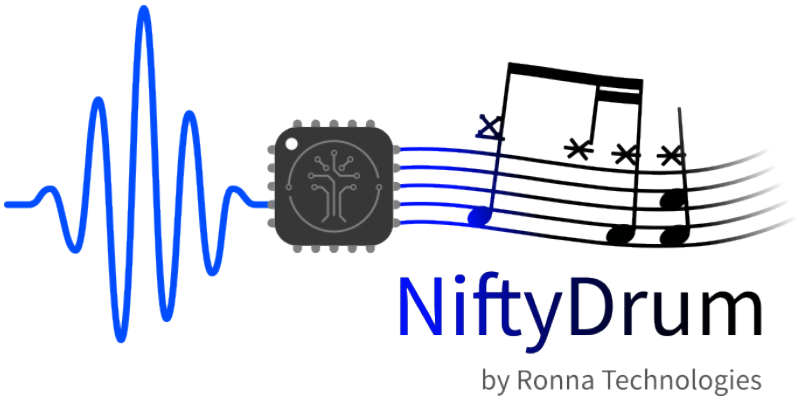
NiftyDrum

Official Documentation

Ronna Technologies

Table of contents

1. About NiftyDrum



1.1 Description

NiftyDrum is a trigger-to-MIDI conversion module that transforms piezo and FSR sensor inputs into MIDI messages. Connect up to 9 piezo sensors and 1 FSR (Force Sensing Resistor) to the dedicated terminal blocks, then receive MIDI data via USB-C connection.

1.2 How It Works

NiftyDrum delivers high-level MIDI performance in 4 easy steps:

- **Connect sensors**: Attach up to 9 piezo sensors and 1 FSR to the terminal blocks
- **Plug in**: Connect to your DAW, Raspberry Pi, or drum module via USB
- **Configure**: Use the web-based GUI to adjust trigger parameters, MIDI mapping, and velocity curves
- **Play**: Notes are transmitted instantly with imperceptible latency

1.3 Specifications

1.3.1 Hardware

- **Piezo inputs**: 9 channels
- **FSR input**: 1 channel (hi-hat controller)
- **Connector type**: Terminal blocks
- **USB interface**: Type-C
- **Dimensions**: 65 × 56.5 mm

1.3.2 Performance

- **Latency**: <2.5 ms
- **Sample rate**: >10 kHz
- **Velocity resolution**: 127 levels (full MIDI range)

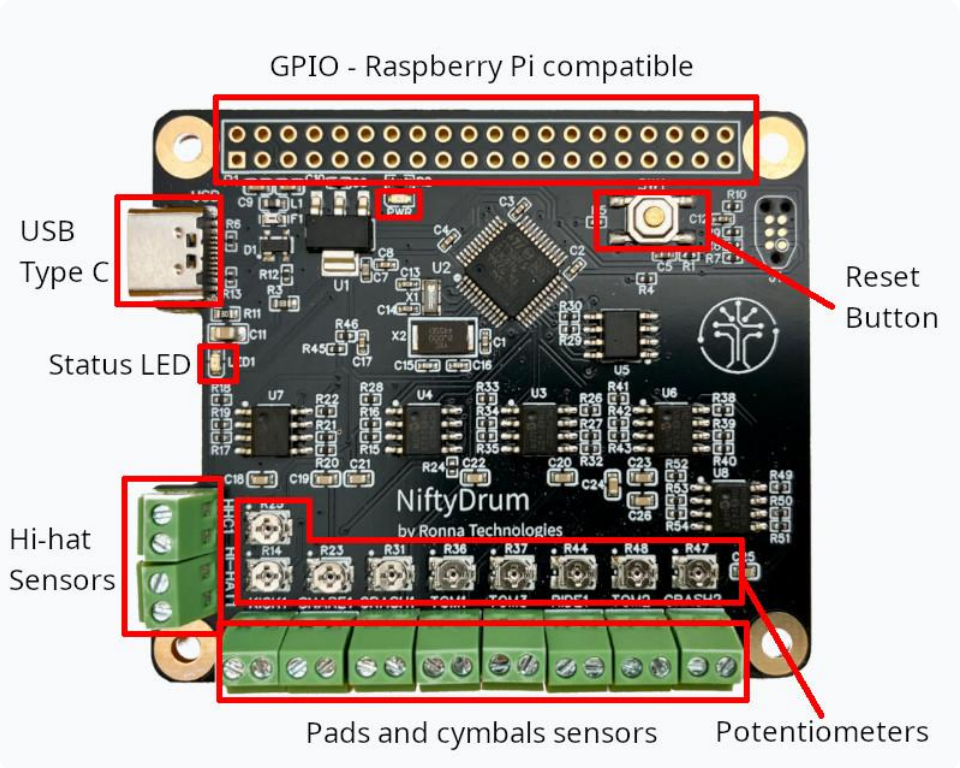
1.3.3 Software

- **Platform support**: Windows, macOS, Linux
- **User interface**: Web-based application
- **Firmware updates**: Via USB
- **MIDI output**: Note messages and Control Changes (CC)

2. The Board

2.1 Description

The NiftyDrum board is shown in the following image.



This board features the following interfaces:

- Terminal blocks for sensor inputs
- USB Type-C port for laptop or PC connectivity
- 9 potentiometers for sensitivity adjustment
- 2 LEDs
- Reset button
- Raspberry Pi-compatible GPIO header
- 4 mounting holes

2.2 Connecting Sensors

The board provides 9 piezo inputs, supporting up to 9 single-zone pads, as well as 1 FSR input.

2.2.1 Hi-Hat Sensors

On the left side of the board, two dedicated terminal blocks are reserved for hi-hat sensors:

- **Top terminal block**: Connects to an FSR (Force Sensing Resistor) sensor for hi-hat controller input
- **Bottom terminal block**: Connects to a piezo sensor for hi-hat cymbal trigger

2.2.2 Standard Pads and Cymbals

The remaining eight terminal blocks, located at the bottom of the board, are for connecting regular pads and cymbals. While the board labels indicate the default firmware assignments, these inputs are fully customizable.

2.2.3 Important Notes

- For all terminal blocks, the ground pin is positioned on the **left-hand side**
- Nine onboard potentiometers enable hardware-level sensitivity adjustments for maximum flexibility
- If unsure about sensitivity settings, leave potentiometers at their midpoint for balanced performance

2.3 MIDI Outputs

The board offers two methods for transmitting MIDI notes and Control Changes:

- **USB-C port**: Outputs USB MIDI messages
- **GPIO UART pins**: Raspberry Pi GPIO-compatible interface

2.4 USB-C Port

Using NiftyDrum as a USB device is the recommended method for receiving MIDI messages. This configuration enables:

- Integration with DAW software for high-quality sound output from your laptop
- Control and configuration via the official app

2.5 Raspberry-Pi hat form factor

The board is designed with a Raspberry Pi 4 HAT form factor, ensuring seamless integration.

2.6 App Features

The official app provides comprehensive control over your NiftyDrum board:

- Customize MIDI note assignments for each trigger
- Design custom velocity curves per trigger
- Adjust advanced parameters including gain, threshold, scan time, mask time, and decay
- Update board firmware to the latest version

3. The App

3.1 Configure NiftyDrum

> ⚠ **Warning**
>
> This app is designed **only for configuration purposes**. While running, it continuously transfers data between your PC and the board. For optimal performance, **close the app before playing**.

NiftyDrum is fully configurable, allowing you to adjust parameters like scan time, mask time, decay, threshold, etc. To simplify customization, a dedicated desktop application is available, compatible with Windows, Linux, and macOS.

Below are all the different commands the app can send to the board.

3.1.1 General Board Commands

Command	Description
Reset	Restart the board in bootloader mode
Serial number	Retrieve the board's unique serial number
Version	Retrieve the current firmware version
Save current parameters	Persist current settings to the board
Load parameters	Load previously saved board parameters
Factory reset	Reset all parameters to factory defaults

3.1.2 Trigger Parameters (Per Trigger, Including Hi-Hat Cymbal)

Parameter	Description
Set/get velocity curve	Adjust or retrieve the velocity response curve
Set/get threshold	Configure the trigger activation threshold
Scan time	Set/get the trigger scan time
Mask time	Set or adjust the trigger mask time
Decay time	Adjust the decay time of the trigger
Gain	Adjust the gain level of the trigger
MIDI Note	Assign the MIDI note for the trigger

3.1.3 Hi-Hat Pedal Parameters

Parameter	Description
Update interval	Set the hi-hat pedal update frequency
Noise threshold	Ignore pedal changes below this value
Pedal offset	Determine if the hi-hat is fully closed
Velocity threshold	Set the velocity threshold for foot chick

3.1.4 How the App Works

The app simplifies customization by organizing everything logically: instruments are selected via a drop-down, while MIDI notes and velocity curves are managed separately from trigger settings for a cleaner, more efficient setup.

3.2 How to Install the App

he app is available for Windows, macOS, and Linux and can be downloaded directly from the official NiftyDrum.com website. Follow the OS-specific instructions provided on the site.

3.2.1 Windows

On Windows, the app is distributed as a `.zip` file, so no installation is required, simply extract and run it. Note that, if that's not already done, you will have to install the Microsoft Visual C++ Redistributable package.

3.2.2 Linux

For Linux, the app is packaged as a `.deb` file. You can install it using your preferred package manager or by running the following command in a terminal:

Ubuntu 22.04 Ubuntu 24.04 debian

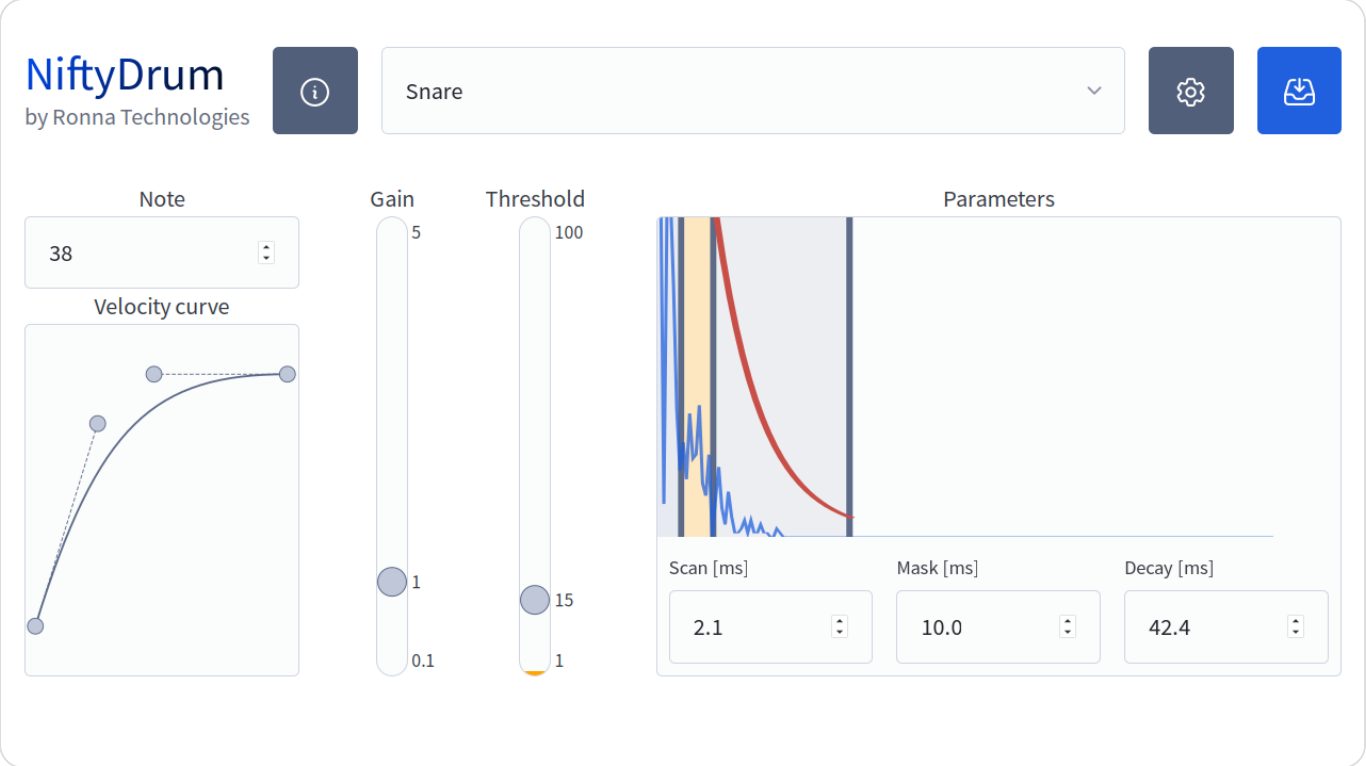
```
sudo apt install --reinstall ./NiftyDrum-1.0.0-Ubuntu-22.04.deb

sudo apt install --reinstall ./NiftyDrum-1.0.0-Ubuntu-24.04.deb

sudo apt install --reinstall ./NiftyDrum-1.0.0-Linux.deb
```

3.2.3 MacOS

The MacOS version of the app is provided as a `.zip` file. Just extract it and run the application. If you're using an Apple Silicon Mac, you may be prompted to install Rosetta the first time you launch the app.

3.3 Piezo Trigger Configuration

Customize each piezo trigger by selecting your desired instrument from the dropdown menu. The screenshot below illustrates the process for configuring the snare drum.



On the left side, you can assign the MIDI note for the pad, such as note 38 for the snare. Below the note input, you'll find the velocity curve editor, which uses Bézier controls for precise adjustments. You can drag and drop the endpoints and the two middle control points to shape the curve according to your needs. The horizontal axis reflects the raw MIDI velocity detected by the sensor, while the vertical axis shows the velocity value transmitted over USB. This setup lets you fine-tune the responsiveness and dynamics of your triggers.

Moving from left to right, you can adjust the trigger gain within a range of 0.1 to 5. This allows you to boost the input sensitivity of the trigger, enhancing the volume of ghost notes when the velocity curve alone isn't sufficient. It's important to note that the gain is applied after a strike has been detected, so it doesn't impact the threshold setting.

Next is the threshold setting, which sets the minimum signal level the piezo must exceed for a hit to register. This ensures only intentional strikes are detected, effectively filtering out unwanted noise and preventing false triggers.

3.4 Hi-Hat Pedal Configuration

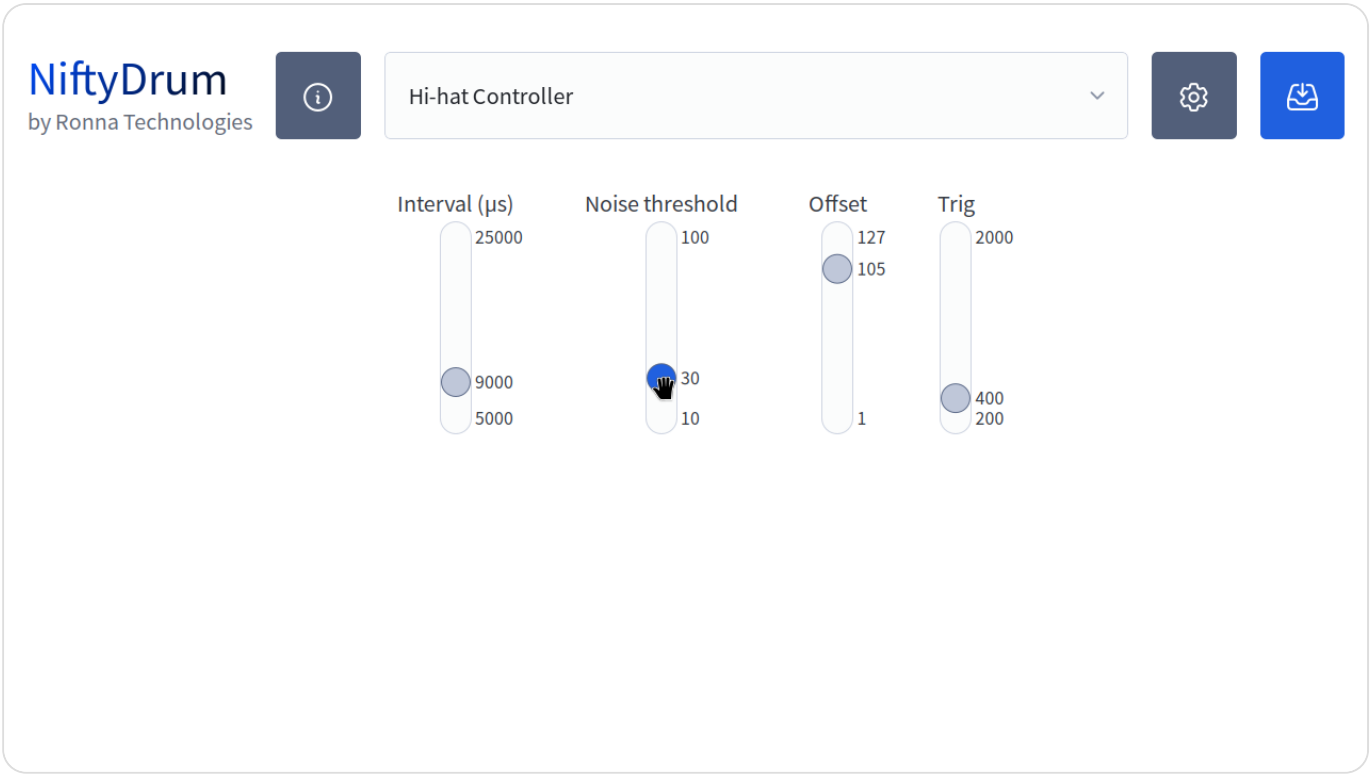
The last item in the dropdown menu is the **hi-hat pedal configuration**. Unlike other inputs, it does not include standard parameters like scan time or mask time.

> **Hi-Hat Implementation Status**
>
> The hi-hat implementation is currently **experimental**. The app settings may evolve in future updates.

3.4.1 Parameters

The hi-hat pedal configuration includes four key parameters:

Parameter	Description
Update interval	Sets the frequency at which the hi-hat pedal updates.
Noise threshold	Ignores pedal changes below this value to filter out noise.
Pedal offset	Determines whether the hi-hat is fully closed.
Velocity threshold (Trig)	Sets the sensitivity threshold for triggering a foot chick note.



3.4.2 CC Message Behavior

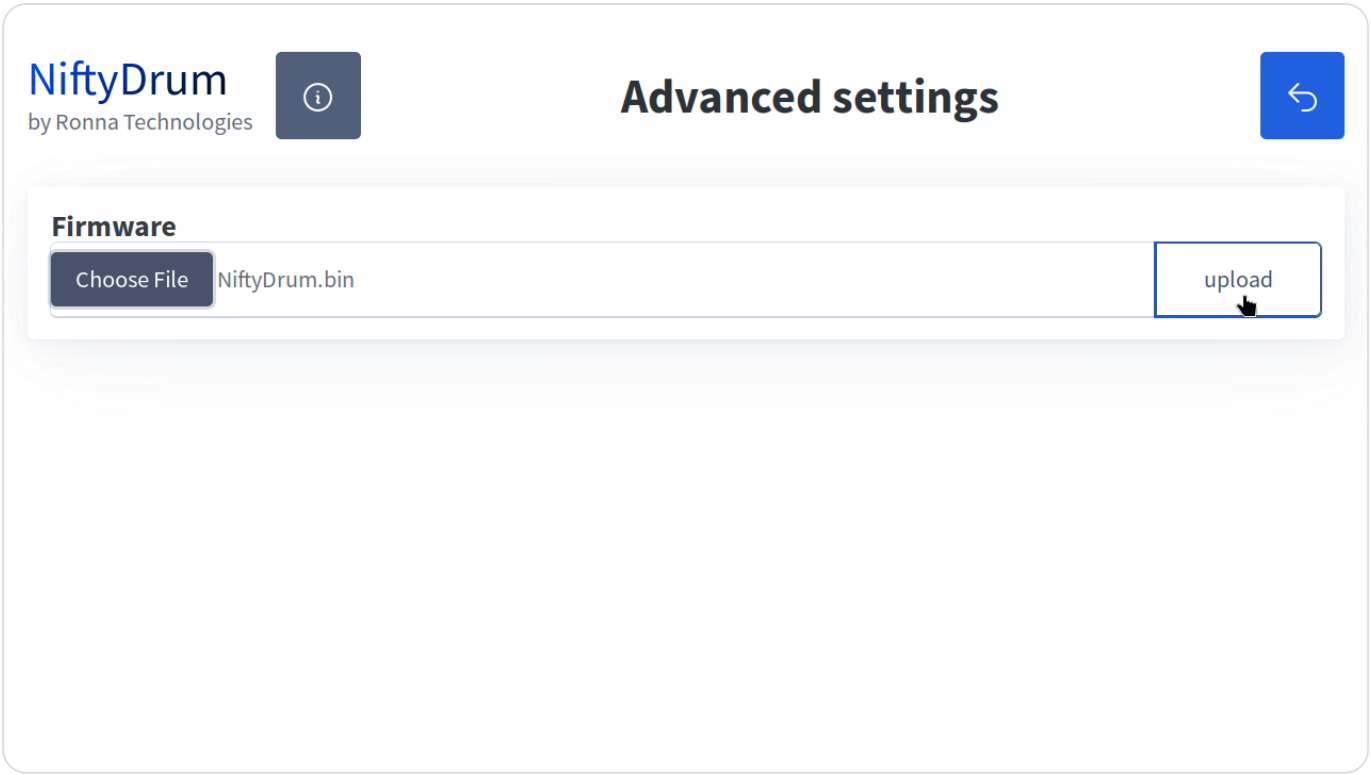
- The **Update interval** defines the minimum delay (in microseconds) between two CC messages, typically set to **10ms**.
- The **Noise threshold** sets the minimum change in value required to send a CC message.

3.4.3 Foot Chick Note Trigger

The **Pedal offset** and **Velocity threshold (Trig)** determine whether a foot chick note is sent, with the latter acting as a sensitivity threshold.

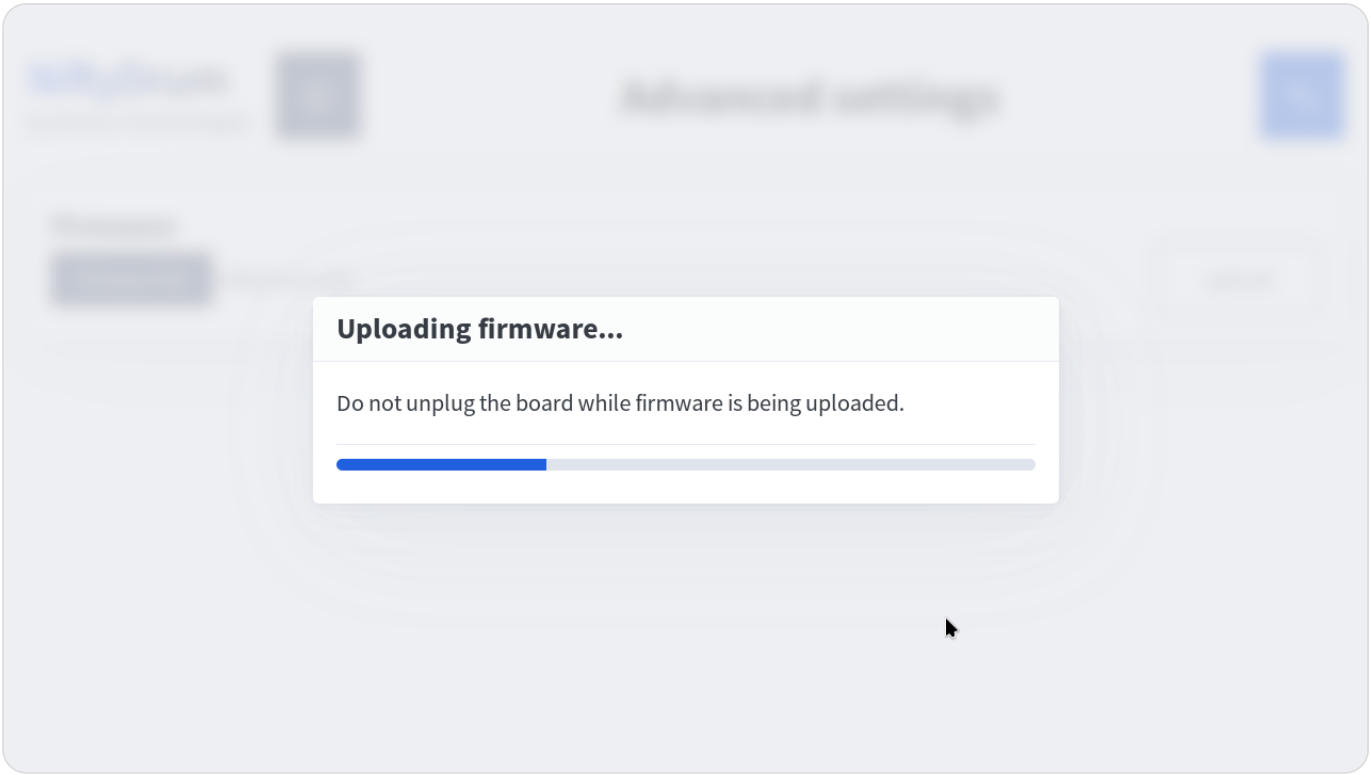
3.5 Firmware Upgrade

The app also supports **firmware upgrades** for your board.



Here is how to upgrade the firmware:

- From the app's main screen, click the **"gear" icon** to access the **Advanced Settings** screen.
- Select a firmware file (if downloaded as a `.zip`, **unzip it first**).
- Ensure the file has a `.bin` extension and is an **officially supported firmware**.
- Click the **`Upload`** button and wait for the process to complete.



> ⚠ **Warning**
>
> **Do not unplug the board** during the firmware upload.

> 🔥 **Troubleshooting**
>
> If the firmware upload fails or the board is accidentally unplugged. Download tkg-flash . **Double-press the reset button** on the board. Upload the firmware via terminal:

Unix Windows

```
./tkg-flash NiftyDrum.bin

./tkg-flash.exe NiftyDrum.bin
```

Once the upload is complete, the board will **restart automatically**, and the app will **reconnect** to it.

4. Serial Protocol

4.1 Overview

The NiftyDrum board establishes a **virtual serial port over USB** when connected to a computer. It also features a **USART interface** on the Raspberry Pi-compatible GPIO port.

> ⚠ **USART Voltage Limit**
>
> The USART pins are **3.3V only**. Exceeding this voltage may damage the board.

4.2 Command Syntax

All commands begin with a forward slash (`/`). Commands that require arguments follow the format:

```
/command arg1 arg2 arg3
```

4.2.1 Special Commands

Special commands do not require parameters. They either modify the board's behavior or provide specific information.

Command	Description
`/reset`	Reboots the board in bootloader mode.
`/factory_reset`	Resets all settings to factory defaults.
`/sn`	Returns the board's serial number.
`/version`	Returns the current firmware version.

4.2.2 Parameters Commands

The board uses **on-board EEPROM** to persist parameters. The serial protocol provides the following commands to save and load these parameters.

Command	Description
/load params all	Loads all parameters from EEPROM.
/save params all	Saves all current parameters to EEPROM.

4.2.3 Pad Parameters

Each pad supports **six configurable parameters**:

- Threshold
- Gain
- Scan time
- Mask time
- Decay
- MIDI note

> **Instrument Placeholder**
>
> The examples below use `snare`. Replace it with `hihat`, `kick`, `crash1`, `tom1`, `tom3`, `ride`, `tom2`, or `crash2` as needed.

Getters

Use the following commands to **retrieve** pad parameter values:

Command	Description	Example Reply
/get snare threshold	Gets the trigger threshold.	15
/get snare gain	Gets the sensor gain.	1.5
/get snare scan	Gets the scan time (µs).	3000 (3ms)
/get snare mask	Gets the mask time (µs).	20000 (20ms)
/get snare decay	Gets the decay time (µs).	60000 (60ms)
/get snare note	Gets the MIDI note assigned to the pad.	36

Setters

Use the following commands to **set** pad parameter values:

Command	Description	Example Usage
/set snare threshold	Sets the trigger threshold.	/set snare threshold 15
/set snare gain	Sets the sensor gain.	/set snare gain 1.5
/set snare scan	Sets the scan time (µs).	/set snare scan 3000
/set snare mask	Sets the mask time (µs).	/set snare mask 20000
/set snare decay	Sets the decay time (µs).	/set snare decay 60000
/set snare note	Sets the MIDI note for the pad.	/set snare note 36

> ⚠ **Value Validation**
>
> The board **does not validate** input values. Incorrect values may cause unexpected behavior.

4.3 Hi-Hat Controller Parameters

The hi-hat controller behaves differently.

5. Arduino