CTR2-MIDI

MIDI Mode

Manual

v3.00.01



Last Revision: November 4, 2026

Copyright 2024-26: Lynovation, LLC
All rights reserved

Updated to firmware v3.00.01

Revised sections for this version are highlighted in yellow

Contents

Introduction 4
MIDI Mode 4
Flex Mode 4
Legal Notice 6
How to use this manual 6
Change Log 6
v3.00.01: November 4, 2026 6
Quick Start Guide 7
Basic Concepts 8
Initial Startup 8
Connect to and Configure Your App 8
Configure Your *MIDI* 9
Keywords and Definitions 9
Hardware 10
Operating Modes 11
Switching Modes 11
MIDI Control Types 12
MIDI Button Pairs for Dial Button Mode 13
Two Maps 14
Connecting CTR2-MIDI to Your App 14
Remote Ham Radio 15
RemoteTx 15
AetherSDR 15
Marcus' Apps for iOS or MacOS 16
..... 17
CTR2 Controller Screen 17
Windows Applications 19
Thetis 19
SparkSDR 20
BTN# + Knob Options 20
Beep Mode (BTN1+Knob) 20
Speed Tuning Mode (BTN2+Knob) 21
Knob Map (BTN3+Knob) 21
Run CW Report (BTN4+Knob*) 21
Switch Modes (BTN5+Knob*) 21
Bluetooth Radio Power (BTN6+Knob) 22
CTR2-MIDI Controls 22
Dual-Function Buttons (BTNs) 22
Knob Functions 23
Knob Mode Indications 24
Paddle Input Mode 24
Application Mapping 25
Mapping for Remote Ham Radio 25
Keying Input 25
Knob Mode Control 26
Button Controls 26
Mapping for RemoteTx 27

PTT Input 27
Knob Mode Control 28
RemoteTx Knob Mapping 28
Mapping for AetherSDR 28
Mapping for iOS and MacOS Apps 29
Editing the CTR2 Controller Map 30
Mapping Functions to Controls 30
Using Paddles with the App's Keyer 31
Using a Straight Key or an External PTT Switch 31
Mapping for Thetis 32
Mapping Controls 32
Advanced Options 34
Operation with N1MM Logger+ 34
Troubleshooting CTR2-MIDI Firmware 35
Appendix A: CTR2-MIDI Terminal Menus 37
Device Names 37
Knob Mode 38
Pdl Mode 38
Beep Mode 38
Speed Tuning 38
Knob Map 38
Run CW Report 39
Edit Knob Map 39
Extended BTN Mode 41
Switch to CTR2-MIDI Flex Mode 42
Bluetooth Pwr 42
Maintenance Menu 42
Export Settings 42
Import Settings 43
Recalibrate Buttons 44
Reset to Factory Settings 44
Appendix B: Using CTR2-MIDI Firmware on CTR2-Micro 45
Flashing the Firmware 45
Operating the *Micro* with *MIDI* Firmware 46
LED Indications 46
Modified BTN#+Knob Functions 46
Appendix C: Installing or Updating CTR2-MIDI Firmware 47
Installing Firmware using ESPConnect (New Method) 47
Installing Firmware using EspressIF Flash Download Tool (Old Method) 50
Installing Firmware using Linux or Mac 51
Appendix D: Configuring Tera Term 53
Appendix E: Configuring Putty 54
Serial Session 54
Appendix F: Apple or Linux Terminal Programs 55
Appendix G: Using CTR2-MIDI with Non-MIDI Programs 56
Configuring CoyoteMIDI 57
Configure SmartSDR for Windows Frequency Control 57
Configure wfView Frequency Control 58
CoyoteMIDI Startup Options 59

Appendix H: Knob and Normal Button Assignment Worksheet 60
Appendix I: Knob and Extended Button Assignment Worksheets 61
SmartSDR for iOS/MacOS Worksheet Example 62
Thetis Worksheet Example 63
Appendix J: Change Log 64

Introduction

CTR2-MIDI (called the ***MIDI*** in this document) is a small radio controller designed for ham operators. It utilizes dual-boot firmware that allows you to boot in either **MIDI** controller firmware, or **Flex** controller firmware.



MIDI Mode

Booting into ***MIDI****'s* MIDI firmware allows you to control MIDI enabled radio control apps using Bluetooth MIDI, USB MIDI, or HID (Human Interface Device) control.

Flex Mode

Beginning with v2.00.00 firmware, you can choose to boot into **Flex** mode. In this mode, the controller connects directly to the network server in your Flex 6xxx/8xxx radio over WiFi allowing you to control it using the Flex API. This mode works with any version of SmartSDR, including the Windows version. It does not, however, support SmartLink for remote control.

Beginning with v3.00.00 firmware, **Flex** mode supports the new **CTR2 Relay** feature that was added to AetherSDR v26.10.3 and above. Just plug your ***MIDI*** into a USB port on the computer running AetherSDR and **CTR2 Relay** creates a virtual connection between ***MIDI*** and your radio.

Flex mode is covered in the CTR2-MIDI Flex Mode manual.

In **MIDI** mode, MIDI commands sent by the ***MIDI*** can be used to change frequency, mode, volume, CW speed, control the keyer and PTT, and many other functions in the app. The ***MIDI*** only sends commands to the app. It has no idea what the app or radio is doing. Function mapping must be done in the radio control app.

The app manages the user interface and the connection to the radio. The ***MIDI*** can be powered from the computer's USB port, and iOS device's accessory jack using an OTG adapter (also referred to as a Camera Adapter), a 5-volt cell phone charger, or a USB battery pack making the ***MIDI*** an idea companion for portable operation. You can turn the Bluetooth radio off to conserve battery power when using USB MIDI.



The ***MIDI*** can use USB MIDI when powered by the device running the app. If the ***MIDI*** is powered separately, you can use Bluetooth-LE MIDI to connect it to the app.

This photo shows the ***MIDI*** controlling SmartSDR for iOS while being powered by the phone though a Lightning OTG adapter. A charger or USB battery can be connected to the OTG adapter's Lightning power jack to power both devices. The Bluetooth radio can be turned off in this configuration to conserve battery power.

The ***MIDI*** includes the following features:

1. USB MIDI and Bluetooth-LE MIDI connections are supported.
2. Turn the Bluetooth radio off to save 50 mA of current draw for battery operation.
3. Three button/knob beep modes, off, normal, or long-press only.
4. Two proportional (speed) tuning speeds.
5. Two maps to customize the MIDI controls assigned to each **Knob** function.
6. Four knob modes control a total of 8 "Wheel" or "Slider" functions for tuning, volume control, etc. Yellow LEDs indicate the selected knob mode.
7. Six dual-mode pushbuttons. Each button can be assigned to the short-press and a long-press function.
    a. You can enable **Extended BTN mode** to expand the number of buttons available from 12 to 48. In ***Extended BTN mode***, each knob mode has its own set of 12 button functions that can be mapped to select modes, bands, or other app functions.
8. 3.5mm (1/8") stereo input jack allow you to use your paddles to control the keyer, straight key, or PTT in the app (not all apps support all modes).
9. Power the ***MIDI*** from any USB 5-volt power source; computer USB port, cell phone charger, USB battery, or through an OTG adapter on your mobile device.
10. Firmware updating is done through the USB-C port.
11. HID support for RemoteTx and OnlineRemoteRadio – the unit must be in **MIDI** mode to control these apps
12. Flex mode allows you to connect to a Flex radio directly, without a 3rd party app, using WiFi. It also support direct Flex radio control over USB when used with **CTR2 Relay** in AetherSDR.

Legal Notice

What would a manual be without a legal notice? Here goes…

- This is a hobby endeavor. Nothing is guaranteed! Use this device at your own risk!
- I will do my best to make sure you receive functioning hardware if you buy a CTR2-MIDI and will work with you if there is a problem with your unit on arrival.
- I cannot guarantee or warranty the hardware supplied in the kit (kits have been discontinued).
- I make no warranty that the firmware provided for CTR2-MIDI will perform up to your expectations or be suitable for your application. Software bugs are a fact of life and I try to find and correct all bug reports to the best of my ability ASAP.
- **MIDI** mode can be complicated to configure. Plan on spending some time learning how MIDI mode works. Everything you need to know should be in this manual.

How to use this manual

This manual should be used as a reference manual. An expanded Help system if you will. Items in the Table of Contents link to their write up in the manual. The main categories have short write ups describing the functions available in that section. I've tried to group things logically and have added hyperlinks so you can quickly jump to other sections.

As this document evolves, sections that have changed since the last update will be highlighted in yellow.

The version number of this manual will follow the latest released version number of the firmware.

Change Log

This manual covers the **MIDI** mode operation. For information on **Flex** mode, download the CTR2-MIDI Flex Mode manual.

v3.00.01: November 4, 2026

- Bug fix in **Flex** mode. **MIDI** mode not affected, just keeping version #s synced

v3.00.00: November 3, 2026

- Added support for AetherSDR's CTR2 Relay USB mode to **Flex** mode
- Removed "WiFi" from **Flex** mode references since CTR2 Relay mode allows a USB connection to your radio in AetherSDR
- Simplified this manual

v2.01.03: October 29, 2026

- MIDI Maps were not indicating the correct maps # on the terminal or the Morse sounder. The Map was switching, but the operator always saw it in Map #1.
- Added the Map# after the "M" Morse character on boot. This way, the operator knows which map they're operating in.

v2.01.02: September 29, 2026

- Renamed this manual from **CTR2-MIDI Operation Manual** to **CTR2-MIDI MIDI Mode Manual**.

v2.01.01a: April 15, 2026

- Added additional information and a warning about the Paddle/Key/PTT jack's function

v2.01.01: March 24, 2026

- Fixed PTT operation in Flex mode
- No changes to MIDI mode

v2.01.00b: March 7, 2026

- Added a new Quick Start Guide to help new users get their **CTR2-MIDI** up and running quickly

v2.01.00a: February 12, 2026

- Updated Appendix C: Installing or Updating CTR2-Flex Firmware to include ESPConnect as the primary method to flash firmware to the **CTR2-MIDI** on PCs and Macs (sorry, not Linux).

v2.01.00: February 3, 2026

- Not saving Map 2 configuration – fixed
- Added HID support for RemoteTx to firmware - browser based rig control
    - **NOTE: HID** and **RemoteTx** support only works in **MIDI mode**.

Changes to previous versions can be found in Appendix J.

Quick Start Guide

If you're like most hams, when you receive a new tool for your shack you just want to plug it in and try it out.

Unfortunately, the ***MIDI*** is not one of those tools!

Unless you plan on using your new ***MIDI*** in Flex mode, you'll need to configure the 3rd-party radio control app you're using to work with it. This is not as hard as it seems, and I have many pre-configured maps and spreadsheets available to get you started.

If you are planning on using your ***MIDI*** in Flex mode, download that manual here and follow its **Quick Start Guide**.

This manual will give you more information than you probably want to know. The goal of this section is to give you a basic understanding of the ***MIDI*** and get you connected so you can start learning about what the ***MIDI*** is all about.

Basic Concepts

Let's start with what the ***MIDI*** is and how it works with your 3rd-party app.

CTR2-MIDI Controller

CTR2-MIDI (***MIDI***) when operating in MIDI mode, is a simple MIDI controller that has been optimized for ham radio use and for 3rd-party radio control apps that support MIDI control. It includes a tuning knob that has 4 dual-function tuning modes (turn and press-and-turn), 6 dual-function pushbuttons (short and long-press), and a dual-function 3.5mm stereo input jack (for paddles or a straight key and PTT switch). All controls on the ***MIDI*** are "hardwired" to specific MIDI control #s, so very little configuration is required on the ***MIDI***.

3rd-Party App

The app is the real brains of the system. It controls the radio and provides the user with a user interface. There are many to choose from because they usually control radios from one manufacturer. For example, SmartSDR for iOS/MacOS controls a Flex radio, SDR-Control is for Icom radios, and Thetis is for many of the SDR radios from Apache Labs and the Hermes Lite 2. Generally, all of the controls in the app are "virtual" so radio functions are changed with a mouse or a touchscreen. The purpose of the ***MIDI*** is to provide physical controls for your radio. It does this by connecting it to a 3rd-party app.

MIDI Control

MIDI is a control protocol used for musical instruments. It's small and fast. MIDI sends Notes (buttons) and Wheel/Slider controls. Because each ***MIDI*** physical control is hardwired to a MIDI control #, you just need to "map" each function in the app to the control # sent by the ***MIDI*** when you execute one of its controls. All apps provide a method to map MIDI functions. I cover many of them here. I also provide many spreadsheets and app map files to get you started here.

Initial Startup

The first order of business is to power up your ***MIDI***. This is generally done by plugging the ***MIDI*** into a USB port on your computer. Do not plug it into an unpowered USB hub. Unpowered hubs cannot provide the power the ***MIDI*** requires when it's using the Bluetooth radio.

When the ***MIDI*** boots it will flash all of the LEDs. v2.xx units will sound an 'M' plus the currently selected map (1 or 2) in Morse. This indicates that the ***MIDI*** has booted into **MIDI mode**. A Windows computer will also "ding" when it registers the ***MIDI***. If the ***MIDI*** sounds an 'F', it's telling you that it booted into Flex mode. You'll want to switch it to **MIDI mode** for this guide. To do this, press-and-hold BTN5 (button #5, middle button on the right side), then press-and-release the knob switch, followed by releasing BTN5. The unit will sound 'M?' in Morse. Press-and-release the knob switch to accept the mode change, or any button to cancel.

Once the ***MIDI*** is powered up, connect it to your app.

Connect to and Configure Your App

First, connect ***MIDI*** to the device running your radio control app. The Connecting CTR2-MIDI to your App section has instructions for connecting to many of the popular apps. Open the apps MIDI editing menu

then execute a control on the ***MIDI*** to see which function it controls. I have map files and spreadsheets available for many other apps out there. Map files can be imported directly into your app. The spreadsheets allow you to manually edit your app's mapping. If your app isn't included in my files, you can use my spreadsheets as a guide to configure your app manually.

Configure Your *MIDI*

If you change the default mapping in your app, you may need to change the MIDI control type the ***MIDI*** uses for each knob control. There are basically two knob control types, **Wheel** and **Slider**. Wheel controls are used for continuously variable controls, such as VFOs. Slider controls are used for potentiometer type controls like volume, squelch, power, etc. The control type must be edited with a dumb terminal such as Putty or Tera Term connected to the ***MIDI***. You can find more information on changing control types here.

NOTE: If the VFO function (usually the default knob mode, all yellow LEDs OFF) doesn't control the VFO in your app:

- Check the mapping in the app to verify that turning the knob executes the VFO/Frequency function
- Change the MIDI control type in the **Edit Knob Map** menu using a dumb terminal. Control type **WheelA** is generally used with Marcus' apps, RHR, RemoteTx, and most other apps. **WheelB** is generally used with Thetis, SDR-Console, and some other Windows apps. **WheelB-r** is only used with piHPSDR.

Once you have the configurations done, it's time to start exploring what the ***MIDI*** can do!

Keywords and Definitions

As many are new to the concept of controlling radios with an external controller, it will help to define many of the keywords used in this document.

Bluetooth	A wireless technology used to connect two devices together
Bluetooth MIDI	A method that sends MIDI commands to an app running on another computer using Bluetooth
BLE	Bluetooth Low Power – An advanced type of Bluetooth communication that uses less power than conventional Bluetooth
CTR2	A product line from Lynovation.com that specializes in control devices for ham radio
CTR2-MIDI	A miniature radio controller based on the ESP32-S3 microcontroller. It uses Bluetooth or USB MIDI to connect 3rd party radio control apps that support MIDI control of their functions. Referred to as "the ***MIDI***" in this manual.
Flex Mode	A mode available in **CTR2-MIDI** firmware that uses WiFi (or USB with **CTR2 Relay** in AetherSDR) instead of Bluetooth to connect directly to, and control, a Flex radio with, or without a 3rd party radio control app
MIDI Mode	This is the original **CTR2-MIDI** firmware that uses *Bluetooth* or *USB MIDI*, or *USB HID*, to control a 3rd party radio control app. This manual covers this mode.
Extended BTN Mode	When *enabled*, the 12 button functions on the unit change with each *Knob Mode*. When *disabled*, the 12 button functions in the Knob Home mode are used in all knob modes. Both yellow LEDs flash when changing knob modes when *enabled*.

Extended Paddle Mode	When enabled, the Paddle Input Jack can be used for straight-key and PTT switch input instead of paddle input for the internal keyer. The green LED is lit when enabled.
HID	Human Interface Device – a USB protocol used to control RemoteTx and OnlineRemoteRadio. HID is more stable than MIDI to control browser-based applications.
Knob Mode	Refers to one of four modes available on the knob control – each knob mode has a *Push* and a *Push & Turn* function – Knob modes are indicated by the yellow LEDs
LAN	Local Area Network – generally a wired network contained within small, localized area such as a home, office, or your radio shack – It usually incorporates WiFi to connect wireless devices to the network
MIDI	A protocol originally designed to control musical instruments - It is simple, small, and fast
MIDI Button	A MIDI command that sends an On or Off command to control a parameter in a radio control app – also referred to as a MIDI Note command since it was originally used to send musical note command to instruments
MIDI Control	A MIDI command that allows sending a value to the radio control app to change a variable parameter such as tuning, volume, etc.
Radio Control App	A 3rd party program or application running on a computer, tablet, or phone that is used to control a radio
Serial Port	A connection that sends data to, or receives data from, a remote device serially – the serial communications can travel over a physical wire (RS-232) or be embedded in a virtual connection using *USB*
Terminal	A program such as Tera Term or Putty, sometimes called a *dumb terminal* that uses a serial port that allows you to interact with a device running a *terminal server*
Terminal Server	An application running on a device that allow it respond to key strokes sent from a *dumb terminal* – generally used to create a basic user interface to control a remote device
USB	Universal Serial Bus – a common interface available on all computers that enables communications between the computer and external devices
USB MIDI	A method used to send MIDI commands to an app using USB
WiFi	A wireless technology that allows remote devices to connect to a *LAN* network

Hardware



The ***MIDI*** uses a 60mm x 60mm x 20mm enclosure. This form factor is rugged and the perfect size for a busy operating desk or for portable operation.

As shown in the photo, the knob is the predominant feature on the face of the unit. When oriented with the USB-C connector and paddle jack as shown the two LEDs in the upper left corner show the knob mode with the lower LED having a value of 1 and the upper LED having a value of 2. Both LEDs will be off when the knob is in its **Home** mode. Short-pressing the knob steps you through knob modes **1, 2, 3**, and back to **Home**. In the photo above the knob is in mode 1.

The green LED flashes once every 5 seconds if the Bluetooth radio is turned off. It flashes once every 2 seconds when the unit is first powered up and the Bluetooth radio is turned on. It changes to two flashes every second when the unit is connected via Bluetooth-LE to the app. It is on all the time when you enter the *Extended* paddle mode.

The dual-function buttons (BTNs) are laid out around the circumference of the knob. BTN1 is on the upper left, just below the knob mode LEDs. The buttons are numbered counter-clockwise around the knob ending with BTN6 on the upper right. Short and long-press functions allow you to control two MIDI **Button** functions with each button.

You'll find the USB-C and 3.5mm (1/8") stereo Paddle/Key/PTT input jack on the top edge in the photo.

CAUTION: The 3.5mm jack is for paddle, key, and PTT inputs only. It is not designed for output. Do not connect this jack to your radio as this may damage the unit.

Operating Modes

CTR2-MIDI can be configured to operate in either the **MIDI** mode or in **Flex** mode. **MIDI** and **Flex** modes cannot be used at the same time.

MIDI mode allows you to control MIDI enabled 3rd party radio control software such as SmartSDR, SDR-Control, TS-Control, FT-Control, K4-Control for iOS/MacOS, AetherSDR and RimuSDR for any operating system, or Thetis and SDR-Console for Windows, using a Bluetooth or USB MIDI connection. It also works well with Remote Ham Radio, RemoteTx, and OnlineRemoteRadio browser-based controllers.

Flex mode, allows you to directly control your Flex 6xxx/8xxx/Aurora series radio using WiFi without the need of a 3rd party radio control app. **Flex** mode connects directly to the network server in the radio and works with any version of SmartSDR, including the Windows versions. It does not support SmartLink.

AetherSDR provides a USB connection option for **Flex** mode. Instead of using WiFi to connect to your radio, the **CTR2 Relay** feature in AetherSDR allows you to connect to the radio AetherSDR is controlling using USB instead of WiFi. **CTR2 Relay** also works when controlling your radio over a VPN link to AetherSDR.

Another option for remote operation is to switch to **MIDI mode** and use AetherSDR or one of Marcus' (DL8MRE) iOS/MacOS radio control apps. You can download the operation manual for **Flex** mode here.

Switching Modes

To switch between **MIDI** and **Flex** modes, press-and-hold **BTN5** (middle-right button), then press-and-release the knob switch. The unit will power down and restart in the new mode and sound either Morse "M" plus the current map # (MIDI) or "F" (Flex) to indicate the new mode.

The following sections describe **MIDI** mode. For information on using **CTR2-MIDI** in **Flex** mode, refer to the **CTR2-MIDI Flex manual**.

MIDI Control Types

Before discussing the ***MIDI's*** configuration, it's important to understand the types of MIDI controls ***MIDI*** supports. The table below summarizes each function.

Buttons	Application
Button controls	Buttons execute MIDI Note controls. Each button in ***MIDI*** is assigned to a MIDI Note #. The short-press function of buttons 1 through 6 are assigned to MIDI notes 1 to 6. The long-press function is assigned to MIDI notes 11 to 16. When you execute a button, the program sends a MIDI Note ON command followed by a MIDI Note OFF command to the app when you *release the button*.
Wheels & Sliders	**Application**
WheelA	This control is continuously adjustable and is centered at 64 counts. Frequency up/down control is initiated by sending a value above or below 64 depending on the direction and speed the dial is turned. The offset from 64 counts determines the turning speed. This control is used with Marcus' iOS/MacOS programs, RHR, and RemoteTx, AetherSDR, RimuSDR, and most modern control apps for frequency control.
WheelB	This control is similar to WheelA and is used with Thetis and many older Windows control programs like SDR-Console. This wheel sends 126 counts to decrement the frequency by one step and 1 to increment the frequency by one step. When the knob is quickly turned the firmware sends 10 commands for each step instead of one to speed up tuning.
WheelB-r	This control is the same as Wheel B except that 1 is sent to decrement frequency and 126 is sent to increment frequency. This control type was added to support the PI HPSDR control program.
SliderA	This control is similar to potentiometer. Its range is 0 to 127 counts. Turning the *Dial* control CCW reduces the count until 0 is reached. Turning it clockwise increases the count until 127 is reached.
SliderB	This control is similar to SliderA except that it is used specifically for center tuned controls like RIT and XIT. This control is centered at 64 counts. Decrementing below 64 counts applies a negative offset and incrementing above 64 counts sends a positive offset. When the value of this control reaches 64 when turning from either direction, ***MIDI*** beeps and pauses for 750 milliseconds. This allows you to find zero on the control.
Buttons Wheel Type	This control assigns a pair of MIDI buttons controls to send up and down button controls to the control program using the *Dial* or *Knob*. The lower control # is sent when the control is turned CCW and the upper control # is set when the control is turned CW. This is normally used for frequency or frequency tuning step up/down control. The control pair is determined by the control # as shown in the next table.

MIDI Button Pairs for Dial Button Mode

When the **Button** control type is selected for a **Dial** control, the MIDI button pair is determined by the Dial control #. Turning the knob CCW sends the lower MIDI command #, and turning it CW sends the upper MIDI command # from the table below.

Dial Ctrl #	Button Pair	Dial Ctrl #	Button Pair
100	40/41	104	48/49
101	42/43	105	50/51
102	44/45	106	52/53
103	46/47	107	54/55

* **Button 60** is shared with the last button in the **Button** menu.

EXAMPLE

To assign a knob control to increment or decrement the frequency tuning step in SmartSDR when you press and turn the knob in the Home knob mode, do the following:

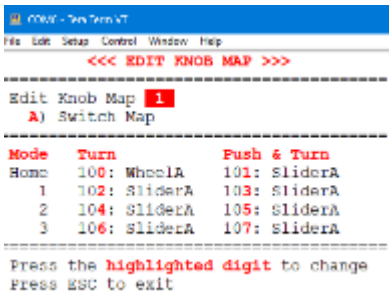
1. We'll use #101 in this example because it's MIDI control that's sent when you push and turn the knob in the **Home** knob mode.
2. Using the **Edit Knob Map** option in the terminal display, set the **Turn** and **Push & Turn** options to **Button**.
3. Open the **Edit Mapping** menu in the **CTR2 Controller Tool** in SmartSDR
4. Select control 103 on the **Dial control** then turn the dial CCW. MIDI button 42 should be highlighted in the **Edit Mapping** menu in SmartSDR.
5. Select the *Tune Step Previous* function from the function list.
6. Turn the dial CW. MIDI button 43 should highlight in the **Edit Mapping** menu in SmartSDR.
7. Select the *Tune Step (1,20,100,1000)* function from the function list.

Now, turning the knob for Dial control 101 will increment or decrement the frequency tuning step in SmartSDR.

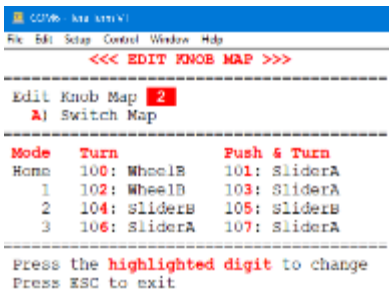
Two Maps

MIDI provides two configuration maps. Each map can be configured for a specific app. The factory default has map #1 setup for radio control apps that use MIDI control **WheelA** for VFO control. Most modern apps use this to control the VFO because it allows for speed proportional tuning. Map #2 is set up for older apps and uses **WheelB** for VFO control. Older control apps, like Thetis and SDR-Console, use this MIDI control type. It only indicates the direction the wheel is turning, not the speed.

The configuration of each map must be set with a dumb terminal using the **Edit Knob Map** option. The following screenshots show the default configuration of each map. To change a setting, just press the highlighted digit.



Knob Map #1



Knob Map #2

You can use a dumb terminal to switch between these maps, however, it's easier to do it on ***MIDI*** itself.

To switch maps, press-and-hold BTN3 (lower-left button) then press-and-release the knob button. ***MIDI*** will sound the new map setting in Morse. "M1" when map #1 is active, and "M2" when map #2 is active.

When you boot ***MIDI*** in **MIDI** mode, it will also sound "M" plus the current map # in Morse.

You can check the current settings ***MIDI*** is using by starting the **Morse Report**. To start this report, press-and-hold BTN4 (bottom-right button) the press-and-release the knob button.

Connecting CTR2-MIDI to Your App

This section describes how to connect the ***MIDI*** to various apps. Once you are connected to the app, see the CTR2-MIDI Controls section to understand how the ***MIDI*** controls work with the app.

NOTE: To use USB MIDI, you must use a USB-C data cable. Many USB-C cables are charge only cables and don't have data lines for communication. If you're using a Windows computer you should hear a beep from the computer when it registers the USB port on CTR2-Dial when you plug it in. If you don't hear the beep, try another cable.

Remote Ham Radio

To connect the ***MIDI*** to Remote Ham Radio, plug the ***MIDI*** into a USB port on your computer then select the **CTR2-MIDI** option in the RHR console's **Tool** menu. The app is pre-configured for the ***MIDI*** so it's plug-and-play. You can see how the ***MIDI's*** controls are mapped to RHR in the Remote Ham Radio Configuration section.

NOTE: RHR requires a Chrome, Edge, or Opera based browser.

NOTE: *Disable* the Extended BTN mode when using RHR.

RemoteTx

RemoteTx is a browser based remote control system. Instead of connecting to the browser using MIDI commands, ***MIDI*** connects to **RemoteTx** as an HID (Human Interface Device). This eliminates most of the problems users experience with MIDI control in browser, especially when refreshing a page.

NOTE: RemoteTx is a subscription service and requires a Chrome, Edge, or Opera based browser.



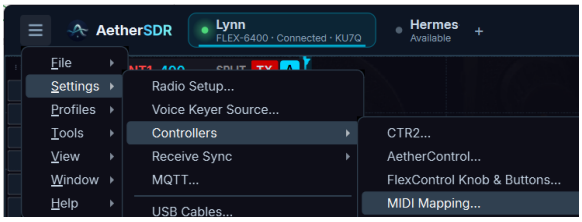
To connect the ***MIDI*** to RemoteTx, plug the ***MIDI*** into a USB port on your computer and start the **RemoteTx** web page. Select the *Radio* button at the top of the page and click the **VFO** button. Next select the **XIAO_ESP32S3** device from the dropdown list.

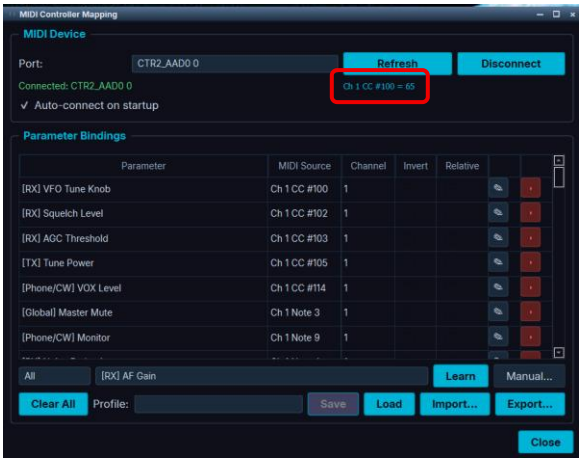
NOTE: *Disable* the Extended BTN mode when using **RemoteTx**. Mapping for this app is covered in the **Mapping for RemoteTx** section below.

AetherSDR

AetherSDR supports both MIDI control and direct radio control using WiFi or USB through its **CTR2 Relay** interface. **CTR2 Relay** is discussed in the **CTR2-Flex Manual**.

To configure the MIDI interface, select the **Settings -> Controllers -> MIDI Mapping…** panel.





With the **MIDI Controller Mapping** panel displayed, plug your ***MIDI*** into a USB jack on your computer the click the **Refresh** button. Your ***MIDI*** should appear on the **Port** list. Select it and press **Connect**.

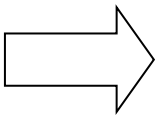
Once connected, rotating the knob or pressing a button on ***MIDI*** will display the executed control's MIDI control # just below the **Refresh** button as shown above. Instructions for loading configuration maps and manually mapping controls can be found in the Mapping for AetherSDR section.

Marcus' Apps for iOS or MacOS

Enable Permissions

If you plan on using Bluetooth with Marcus' apps on your **iOS** device, open the app's **Settings** window and allow Marcus' apps to access Bluetooth.

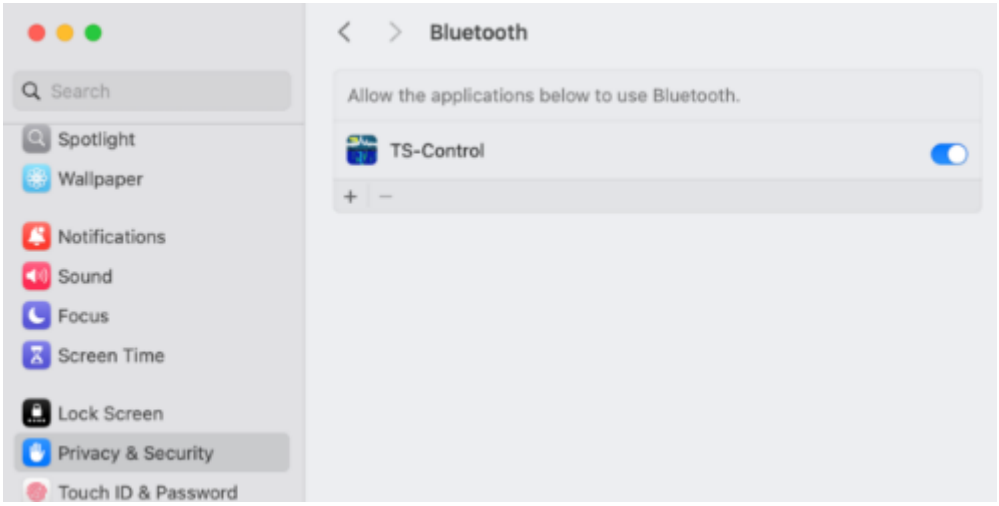






You can also allow the app to use your iCloud Drive to save maps in this window.

MacOS has a similar permission setting. It can be found by navigating to **Apple -> System Settings -> Privacy & Security -> Bluetooth.** The screenshot below shows the *TS-Control* app has permission to use Bluetooth.



Select the CTR2-MIDI Device

In Marcus' apps, select the **CTR2 Controller** device in the **Tools** menu (not the **CTR2 Dial Controller Tool**). The **Tools** menu will look different in each app so just look for the **CTR2 Controller**.

Here's an iOS and MacOS example of the **Tools** menu. I've highlighted the **CTR2 Controller** option in each menu…



iOS Devices



MacOS Devices

CTR2 Controller Screen

Once you select **CTR2 Controller** from the **Tools** menu a screen like this will appear. It may look different depending on the Apple device you're using but it has basically the same information.



Make sure your ***MIDI*** is powered up. The green LED should be flashing once every two seconds if the Bluetooth radio is on, or once every 5 seconds if it is off.

USB Wired Connection



To use the ***MIDI*** with a wired USB connection on Mac's and newer iOS devices, connect the ***MIDI*** to your device using a USB data cable, select **XAIO_ESP32-S3** from the **Device** menu, and click the **Enabled** button. The **Status** should change to **Connected**. Press the **Edit Mapping** button and press a button or turn the knob to verify the app is receiving MIDI commands.

NOTE: To use a wired USB connection with an older iOS device (with a Lightning connector) you'll need to purchase a Lightning OTG (On The Go) adapter such as the one shown here. These are also referred to as "Camera Adapters" on Amazon. The adapter I use has two USB-A connectors on it. It also has a Lightning port so you can connect a wall charger or battery to power your phone and ***MIDI***.

Bluetooth-LE Connection

To use Bluetooth-LE MIDI, make sure the Bluetooth radio is on (green LED flashes every 2 seconds) then click the **Find CTR2 Bluetooth-LE Device** link to initiate a search for your ***MIDI***. The search will fail the first time you try to access a new device. If you get a failure notice, close the failure popup notification, **wait 10 seconds**, then click the **Find** link again. Once it registers your ***MIDI*** (your ***MIDI's*** device name is written on the bottom label if you bought an assembled unit) click the **Enabled** button to start the Bluetooth connection. You don't need to select the ***MIDI's*** name on the **Device** list unless you have two or more ***MIDI*** units available. (You can only use one ***MIDI*** at a time but you can use a separate wired USB MIDI device such as a DJ2GO along with the ***MIDI***.)

NOTE: v1.02 firmware gives you to option of turning off the Bluetooth radio. If the radio is off the green LED flashes once every **5 seconds**. To turn it on press-and-hold BTN6 then press-and-release the knob. You should hear "BLE 1" in Morse code then the unit will reboot so it can turn the radio on.

If you can't get the app to connect to ***MIDI*** you probably haven't allowed the app to use Bluetooth (see Enable Permissions above). If you still can't get it to connect, download **MIDI-Wrench** or **midimittr** for iOS or **Conji** for the Mac from the App store. They are free apps used to troubleshoot BLE MIDI issues. Connect to the ***MIDI*** with one of these apps first and then try to connect to the ***MIDI*** again in the radio control app.

MacOS Bluetooth Connection Problems

Follow these steps to connect the ***MIDI*** to your Mac using Bluetooth:

1. Connect your ***MIDI*** to a USB port on your Mac.
2. In the **CTR Controller** screen, check the **Enabled** button then click **Find CTR2 Bluetooth-LE Device** at the top of the screen.
3. The Mac may report that it can't find a Bluetooth-LE device. Wait 10 seconds the click **Find CTR2 Bluetooth-LE Device** again.
4. The Mac should report that it has found a Bluetooth device named **CTR2xxxx** where xxxx is the address of your ***MIDI***.
5. Select **Xiao_ESP32S3** on the **Device** list to use USB MIDI while connected to the Mac.
6. Select **CTR2xxxx** on the **Device** list to use Bluetooth MIDI.
7. You may need to change the state of the **Enabled** checkbox to get the ***MIDI*** to connect.
8. You can now unplug you ***MIDI*** from the Mac and use it remotely with Bluetooth.

NOTE: Bluetooth can introduce significant delays (up to 15 milliseconds) when using the paddles to control the keyer in your app. Use of wireless Bluetooth headsets can also cause connection issues. Unless you really need to use Bluetooth, I recommend using USB to connect to your device.

Windows Applications

Thetis

Windows only supports USB MIDI connections. When you connect the ***MIDI*** to your Windows computer's USB port it will automatically register as a MIDI device and be assigned a virtual COM port. Windows does not support Bluetooth-LE MIDI (at least not very well).

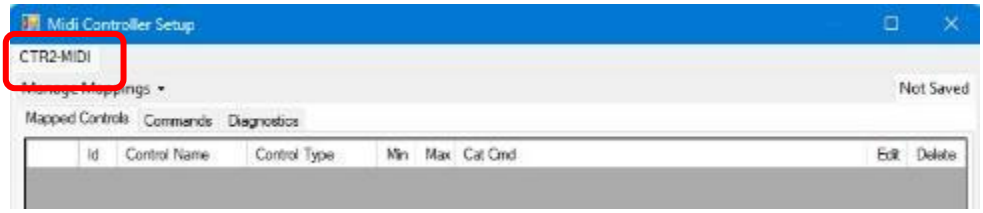


NOTE: USB MIDI requires a USB-C data cable. USB-C power cables will not work!

Most Windows apps will need to be configured (i.e. mapped). **Thetis** is used as an example here but connecting to and mapping controls in other apps should be similar. You will need to experiment with your app's mapping to determine how its map is configured.

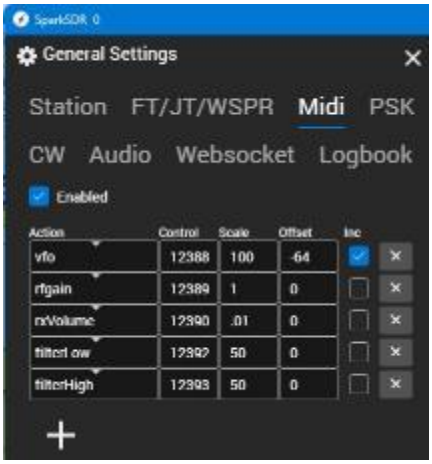
Start by selecting the **Settings -> Settings** menu on the main page in Thetis then select the **Serial/Network/MIDI CAT** tab. Click the **MIDI** tab as shown above.

Next, click the **Configure MIDI** button to open the **Midi Controller Setup** page. The program should go through an initialization sequence and end up on this page if it finds the ***MIDI*** on USB. The tab at the top should be labeled **CTR2-MIDI** or possibly **CTR2xxxx**. If the initialization fails, double-check that you are using a USB-C data cable, not a charge cable and try again.



Additional information on mapping MIDI controls in **Thetis** can be found here.

SparkSDR



SparkSDR (https://www.sparksdr.com/) has limited support for MIDI control. It doesn't support buttons, only the knob. To map the ***MIDI****'s* knob functions to this program, first plug your ***MIDI*** into a USB port on your computer and open SparkSDR. Once you have the program running, click on the **General Settings** menu and select **Midi**. You should see **CTR2-MIDI 0** as a MIDI controller and turning the knob should display MIDI messages from the controller. To map these messages to a control in the app simply press the **+** icon and enter the **Control** number shown in the **Last midi message:** window in the **Control** column. Then select an **Action** and set the **Scale** and **Offset** required for that control. You may need to play with these values to get the control action you want. Check the **Inc** box if you want the value from ***MIDI*** to be added to the current setting of the control. This is only used with the **vfo** control. This screenshot shows the settings I use.

BTN# + Knob Options

Now that you have the ***MIDI*** connected to your app, it's time to go over the various options available in the ***MIDI***. The options below are accessed by pressing-and-holding the indicated BTN then pressing-and-releasing the knob switch. You can also change these options with a dumb terminal.

Beep Mode (BTN1+Knob)

You can control when the ***MIDI*** beeps. Normally, it beeps whenever you press a button or the knob. This lets you know that you've changed something in the app. You may prefer the ***MIDI*** remains quiet all the time or only beep when you long-press a button or the knob.

To change the **Beep** mode press-and-hold BTN1 then press-and-release the knob. The new **Beep** mode will be announced in Morse code as follows:

- "B 0" = **Beep** mode is off. All LEDs will briefly flash when you long-press a button or the knob.
- "B 1" = **Beep** mode is normal. The ***MIDI*** will beep with every button or knob press.
- "B 2" = **Beep** mode is long-press. The ***MIDI*** will beep only when you long-press a button or the knob.

Speed Tuning Mode (BTN2+Knob)

Speed (proportional) tuning is supported when **WheelA** is selected as the MIDI control assigned to the app's tuning function. You can select from **On**, **Fast**, or turn speed tuning **Off**.

To change the Speed mode press-and-hold BTN2 then press-and-release the knob. The new speed mode will be announced in Morse code as follows:

- "S 0" = speed tuning is off
- "S 1" = speed tuning is set to normal speed
- "S 2" = speed tuning is set to fast speed

Knob Map (BTN3+Knob)

MIDI supports two knob maps. This allows you to configure the ***MIDI*** for different apps depending on your needs. **Map 1** is the default and is configured to work with Marcus' iOS/MacOS apps and most modern radio control apps such as AetherSDR and RimuSDR. It uses **WheelA** for tuning. **Map 2** is configured for some older control apps such as Thetis which uses **WheelB** for VFO A and VFO B tuning control.

To change maps, press-and-hold BTN3 then press-and-release the knob. The new map will be announced in Morse code as follows:

- "M 1" = **Map 1** is active
- "M 2" = **Map 2** is active

Run CW Report (BTN4+Knob*)

To hear a Morse report of all of the option setting press-and-hold BTN4 then press-and-release the knob. You can stop the report by pressing any button.

NOTE: You can change the speed of the CW sending the report by turning the knob *while the report is running.*

***** When ***MIDI*** firmware is running on *Micro* hardware, *long-press* **BTN1** then press-and-release the knob.

Switch Modes (BTN5+Knob*)

To switch between **MIDI** and **Flex** modes press-and-hold **BTN5** (middle-right button) then press-and-release the knob switch. The unit will power down and reboot into the new mode. When it boots into **MIDI** mode, a Morse "M" plus the current map #, will be sounded. When it boots into **Flex** mode, a Morse "F" will be sounded. If you have a terminal connected to the unit, press any key to open the home menu. See the **CTR2-MIDI Flex operation manual** for information on this mode.

***** When ***MIDI*** firmware is running on *Micro* hardware, *long-press* **BTN2** then press-and-release the knob.

Bluetooth Radio Power (BTN6+Knob)

You can turn off the Bluetooth radio on the ***MIDI*** if you are using a wired USB connection and are not running ***MIDI*** firmware on *Micro* hardware. This is particularly handy when powering the ***MIDI*** from a battery-operated device such as an iPhone or iPad. When the radio is off the ***MIDI*** draws around 43 mA. When it is on the current draw is around 93 mA.

To toggle the power on the Bluetooth radio press-and-hold BTN6 then press-and-release the knob. The radio's state will be announced in Morse code as follows:

- "BLE 0" = Bluetooth radio is off
- "BLE 1" = Bluetooth radio is on

NOTE: This will change to "BT0" and "BT1" in future versions of CTR2-MIDI firmware.

Once the Morse announcement completes the ***MIDI*** will start beeping once per second. This tells you that the ***MIDI*** is waiting for you to cycle the power to change the radio setting. The radio can only be enabled or disabled during the startup process.

If the Bluetooth radio is off the green LED will flash once every 5 seconds. If it is on, but not connected to your app, the green LED flashes once every 2 seconds. When connected to your app the green LED flashes twice every second.

CTR2-MIDI Controls

The ***MIDI*** sends MIDI commands to the app when the buttons are pressed or the knob is turned. This section will describe the various controls available.

Dual-Function Buttons (BTNs)

Buttons are laid out counter-clockwise around the knob starting with BTN1 on the upper left and ending with BTN6 on the upper right of the knob.

Buttons send the MIDI **NoteOn** command when the button is released.

Each button has two functions defined by the amount of time you hold the button down.

Short-pressing (i.e. less than 1 second) a button produces one beep (if Beep mode is set to Normal) when pressed and sends the associated MIDI **NoteOn** command when released. The MIDI **NoteOn** control # will be from 1 to 6 depending on which button was pressed.

Long-pressing a button (holding it down longer than 1 second but less than 3 seconds) produces another beep to indicate when it enters long-press mode if Beep mode is set to *Normal* or *Long-Press*. If **Beep** mode is *Off* all the LEDs will briefly flash when entering long-press mode. Upon button release the associated MIDI **NoteOn** command is sent. The **NoteOn** control will be from 11 to 16 depending on which button was pressed.

NOTE: Holding a button longer than 3 seconds cancels the button press and sounds a third low frequency beep if **Beep** mode is *Normal* or *Long-Press*. There is no indication if **Beep** mode is *Off*.

MIDI **NoteOn** commands can be mapped in the app to toggle functions such as *Band Up, Band Down, Mode Up, Mode Down*, etc.

NOTE: The ***MIDI*** does not receive updates from the controls it's mapped to in the app. If you change a value with the ***MIDI,*** then change it manually in the app, the next time you change it in the ***MIDI*** it will send the new value based on the last value the ***MIDI*** sent, not the value you manually set in the app.

Knob Functions

The knob has four modes. Each mode has a turn and a press and turn function for a total of eight functions. knob modes are selected by short-pressing the knob. The yellow LEDs indicate the selected mode. Both LEDs will be off in the knob's **Home** mode.

The knob can be used for tuning or changing parameters like volume, bandwidth, CW speed, etc. There are several MIDI control types: **WheelA**, **WheelB**, and **WheelB-r** are used for tuning (depending on the app). Most other controls use **SliderA** which acts like a potentiometer. **SliderB** is used for RIT and XIT controls. It beeps at the ends of the range and it beeps and pauses when you hit the middle of the range (where RIT and XIT are off). Although not generally used, there is a **Button** option. This can also be used for tuning if your app supports up and down buttons for frequency control.

You must define the MIDI control you want the ***MIDI*** to use for each knob function with a terminal program.

MIDI supports two knob maps so you can configure one map for RHR, Marcus' iOS/MacOS apps, AetherSDR, RimuSDR, etc., and the other map for Thetis or other apps. Press-and-hold **BTN3** then press-and-release the knob to toggle the knob map. The active knob map will be indicated by sounding either "M1" or "M2" in Morse. This is also sounded in Morse when the unit boots in **MIDI** mode.

NOTE: The ***MIDI*** always sends commands on MIDI Channel 1.

Knob Mode Indications

The ***MIDI*** supports four knob modes. Each mode has a *turn* and *push & turn* function. The knob mode is indicated by the yellow LEDs as shown in the table below. The ***MIDI*** sends the MIDI *Control Change* command (CC) # indicated in the table below for each mode.

> **NOTE:** The function your radio control app actually executes is determined by the MIDI mapping in your app.

Knob Mode	LED	Beep Tone on Selection	MIDI Control Change
Home	○ ○	Single high frequency beep	Knob Turn = **CC 100** Push & Turn = CC **101**
1	○ ●	Single low frequency beep	Knob Turn = **CC 102** Push & Turn = CC **103**
2	● ○	Two low frequency beeps	Knob Turn = **CC 104** Push & Turn = CC **105**
3	● ●	Three low frequency beeps	Knob Turn = **CC 106** Push & Turn = CC **107**

Long-pressing the knob while in Mode 1, 2, or 3 will return you back to the knob **Home** mode (all LEDs off) and sounds a single high frequency beep if **Beep** mode is set to *On* or *Long-Press*. Long-pressing the knob in the knob **Home** mode toggles the **Paddle Mode**.

Paddle Input Mode

The ***MIDI*** has two **Paddle Modes,** *Normal* and *Extended*. These modes allow you to map different MIDI control functions to the paddle input jack depending on your needs. The **Paddle Mode** is indicated by the green LED. The table below explains the two modes.

Paddle Input	Green LED	Beep Tone On	Description
Normal Mode	○	Two tone High>Low beep	If Extended BTN mode is ***disabled***, the left paddle controls MIDI 20 and the right paddle controls MIDI 21. If it is ***enabled***, the left and right paddle are assigned to MIDI 96 and MIDI 97. Map these to **Trigger CW Left Paddle** and **Trigger CW Right Paddle**. Not all apps have these options.
Extended Mode	●	Two-tone Low>High beep	If Extended BTN mode is ***disabled***, the left paddle controls MIDI 30 and the right paddle controls MIDI 31. If it is ***enabled***, the left and right paddle are assigned to MIDI 98 and MIDI 99. You can map these to **Trigger CW Straight Key** and **PTT Push** to give you a straight key and PTT control in this mode. Or, you can map them the same as Normal mode to disable the **Extended Paddle** mode. Not all apps have these options.

To toggle **Paddle Mode,** long-press-and-release the knob while in the **Home** knob mode (both yellow LEDs off).

If your app doesn't support keyer or PTT input control, or you don't use them, you can map the paddle controls to other functions your app supports.

CAUTION: The 3.5mm jack is for paddle, key, and PTT inputs only. It is not designed for output. Do not connect this jack to your radio as this may damage the unit.

Application Mapping

CTR2-MIDI, running in **MIDI** mode, is a MIDI controller. It has no concept of radios, or how to control them. It only sends MIDI commands based on the physical control you execute. Therefore, it must be used with a 3rd party radio control app that supports MIDI control. How the ***MIDI*** works with your radio is determined by the MIDI mapping within that radio control app. The sections below show how several radio control apps can be mapped to the ***MIDI***. You're not limited to these configurations. Feel free to develop your own mapping based on your operating style.

You can download spreadsheets and application maps for the **CTR2-MIDI Configuration Files** page on the Lynovation web site. The spreadsheets will assist you with configuring the MIDI control types that are assigned to the knob controls in **CTR2-MIDI** and you can import the app maps directly into your app to configure it's MIDI mapping.

NOTE: The two maps in **CTR2-MIDI** are used to configure the knob control types. They do not affect the button mapping. Extended BTN mode controls how many button functions are available and their MIDI Button assignments.

Mapping for Remote Ham Radio

To use the ***MIDI*** with Remote Ham Radio, open the **Tools** menu in the RHR console and select **CTR2-MIDI**.

The ***MIDI****'s* controls are pre-mapped in RHR as follows:

Keying Input

You can plug your paddles or a straight key and PTT switch into the 3.5mm jack on the ***MIDI***. To use paddles, set the **Paddle Mode** to **Normal** (long-press-and-release the knob in the knob **Home** mode until the green LED turns *off*). To use your straight key and a PTT switch, wire the straight key to TIP and Shield and the PTT switch to RING and Shield of a 3.5mm plug and insert it into the 3.5mm jack on the ***MIDI***. Set the **Paddle Mode** to **Extended** by long-pressing the knob while in the knob **Home** mode until the green LED turns *on*.

Knob Mode Control

To select a knob mode, short-press-and-release the knob until the yellow LEDs indicate the mode you want. The knob has a *turn* and *push & turn* function for each mode. The modes and their functions are listed below. I don't know if you can change this mapping in the program.

ENCODER ACTIONS		
LEDS	**User Input**	**Action Taken**
○ ○	Home (turn)	Change VFO
	Home (push & turn)	Slice Volume
○ ●	Mode 1 (turn)	NR Level
	Mode 1 (push & turn)	NB Level
● ○	Mode 2 (turn)	VOX Level
	Mode 2 (push & turn)	*Not used*
● ●	Mode 3 (turn)	Keyer Speed
	Mode 3 (push & turn)	*Not Used*

Button Controls

Each button (BTN) has two functions.

Short-press the button (< 1 second) to access the primary function.

Long-press the button (> 1 second) to access the secondary function. Thos table lists the functions available.

NOTE: Do not ***enable*** **Extended BTN mode** when using RHR because it changes the MIDI Button # assigned to the long-press button functions.

BUTTON ACTIONS	
User Input	**Action Taken**
BTN1	CW Mode
BTN1 (long-press)	Toggle NR
BTN2	SSB Mode
BTN2 (long-press)	Toggle NB
BTN3	FT8
BTN3 (long-press	FT4
BTN4	Toggle PTT
BTN4 (long-press)	Toggle VOX
BTN5	Band Down
BTN5 (long-press)	*Not Used*
BTN6	Band Up
BTN6 (long-press)	*Not Used*

Mapping for RemoteTx

Important: HID and RemoteTx support does not work in **Flex** mode

To use the ***MIDI*** with RemoteTx, click **VFO** in the *Radio* page.



A list of USB devices on your computer will open. Select the **XIAO_ESP32S3** device then press **Connect**. The **VFO** button should turn red, indicating that the ***MIDI*** is connected to this browser session.



The ***MIDI****'s* buttons are pre-mapped in **RemoteTx** as shown here.

The large knob controls the VFO frequency. Push and turn it, or push the knob once to move to knob mode #1 (bottom LED lit) to control the RIT frequency. See RemoteTx Knob Mapping for information on configuring RIT frequency control.

NOTE: Do not ***enable*** **Extended BTN mode** when using **RemoteTx** because it changes the MIDI Button # assigned to all of the button functions.



PTT Input

RemoteTx only supports PTT control. It does not support a CW keyer. You can plug your straight key or PTT switch into the 3.5mm jack on the ***MIDI***. Both TIP and RING of the jack are mapped to PTT in the app. Leave the **Paddle Mode** set to **Normal** (green LED off). If it is on, long-press-and-release the knob while in the knob **Home** mode until the green LED turns *off*.

Knob Mode Control

To select a knob mode, short-press-and-release the knob until the yellow LEDs indicates the mode you want. The knob has a *turn* and *push & turn* function for each mode. The modes and their functions are listed here. As of this version, only VFO frequency and RIT frequency adjustments are supported.

ENCODER ACTIONS		
LEDS	**User Input**	**Action Taken**
○	Home (turn)	Change VFO
○	Home (push & turn)	Change RIT Freq
○	Mode 1 (turn)	Change RIT Freq
●	Mode 1 (push & turn)	<Not used>
●	Mode 2 (turn)	<Not used>
○	Mode 2 (push & turn)	<Not used>
●	Mode 3 (turn)	<Not used>
●	Mode 3 (push & turn)	<Not used>

RemoteTx Knob Mapping

In order to control the RIT frequency with the knob, you'll need to edit the control map for the ***MIDI*** in a dumb terminal such as Putty or Tera Term.



Once you connect your terminal to the ***MIDI***, press the **E** key on the terminal to open the **Edit Knob Map** menu. Once this menu is open, press key **1** until **WheelA** appears next to **101:**. Next, press key **2** until **WheelA** appears next to **102:**. This sets the **Home** press & turn, and knob mode 1 (bottom yellow LED on) to use the **WheelA** MIDI control type for these knob actions.

When you're finished, 100, 101, and 102 should be set to **WheelA**.

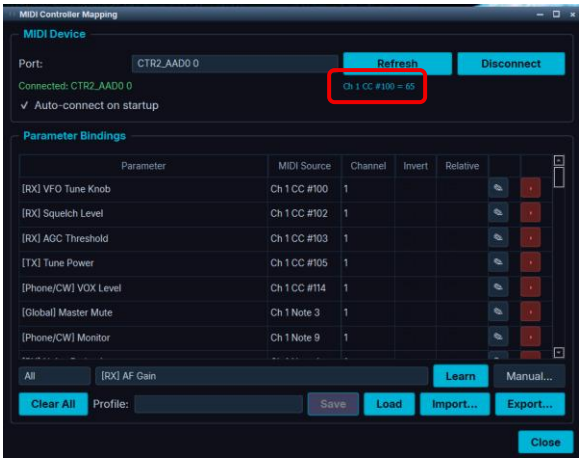
Mapping for AetherSDR

AetherSDR is a newcomer to the radio control scene. It's primarily designed for Flex radio control, but does support Icom and some ANAN and Hermes radios in an experimental mode.

AetherSDR is unique because it supports both MIDI and WiFi/USB connections to control your Flex radio. MIDI control is only available for experimental radio modes.

You can use the WiFi or USB connection modes provided by **CTR2 Relay** when ***MIDI*** is in **Flex** mode. For more information about this mode, see the **CTR2-MIDI Flex Mode manual**.

To configure AetherSDR for use with ***MIDI's*** **MIDI** mode, connect ***MIDI*** to AetherSDR using the process described in the Connect to AetherSDR section above.



Once the **MIDI Controller Mapping** panel is open, press the **Connect** button to connect to ***MIDI***. Turning the knob or pressing a button will display the MIDI control # for the control you changed below the **Reset** button.

Click the **Import** button to import the AetherSDR MIDI map from the AetherSDR folder in the CTR2-MIDI Configuration file.

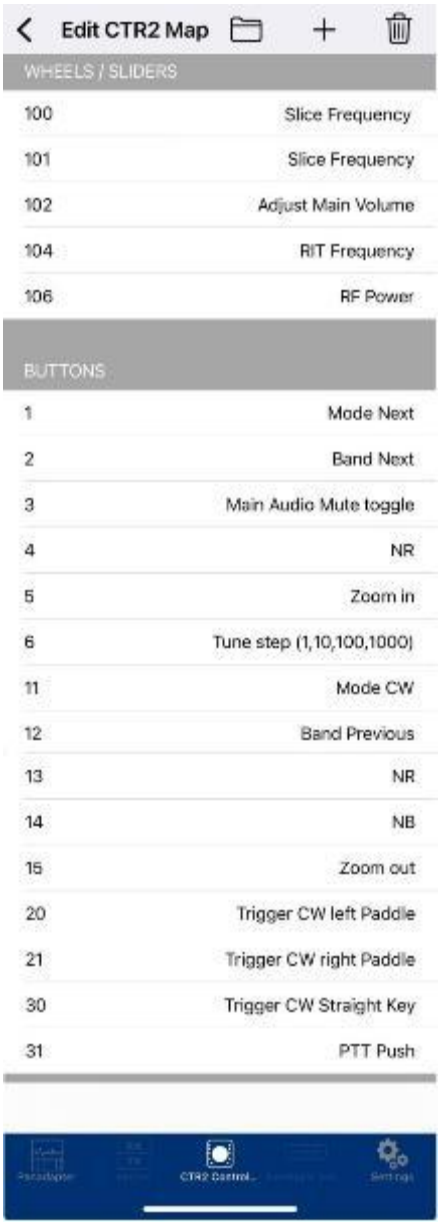
To manually edit the function assignments, click on the **Learn** text box to display a list of functions that you can map. Select a function then press the **Learn** button. Next, execute the knob or button you want to assign to that function. Once you have configured ***MIDI's*** controls, enter a file name in the **Profile** text box then press the **Save** button.

Mapping for iOS and MacOS Apps

The map on Marcus' apps is preconfigured for the ***MIDI*** when you select the **CTR2 Controller** device from the **Tools** menu. You can change the mapping to fit your needs.

NOTE: The example below is based on using the ***MIDI*** with Extended BTN mode ***disabled***. If you ***enable*** **Extended BTN mode**, you can select the **CTR2 Dial Controller** device in the **Tools** menu instead. This device is mapped to support the additional MIDI Button controls available on **CTR2-Dial** and on **CTR2-MIDI** in **Extended BTN** mode. You'll probably want to edit the MIDI map in the app to fine-tune your preferences.

The default configuration for the ***MIDI's*** **Map 1** defines the first primary knob control (**CC 100**) as **WheelA**. This control is typically used for frequency control. Other knob functions are defined as MIDI **SliderA** (no center/edge beeps). You can change these assignments using a terminal program.



Editing the CTR2 Controller Map

Pressing the **Edit Mapping** button in the **CTR2 Controller** page of Marcus' app opens the **Edit CTR2 Map** page.

The map is preconfigured when you select the **CTR2 Controller** the first time as shown here. You can edit it to fit you needs.

Press a button or turn the knob to highlight its function on the map.

Mapping Functions to Controls

To change a mapped function press or click the control line and choose a function from the dropdown menu.

NOTE: Your ***MIDI*** will not be able to control your app unless functions are mapped!

Example: Short-pressing **BTN1** on the ***MIDI*** always sends the **Button 1** command (unless **Extended BTN mode** is ***enabled***). This button is mapped to *Mode Next* function in SmartSDR which increments the mode on the active slice. Long-pressing **BTN1** on the ***MIDI*** always sends **Button 11** (again, unless **Extended BTN mode** is ***enabled***). This is mapped to *Mode CW* in SmartSDR which tells it to set the radio's mode to CW.

Once you have your map configured press the folder button at the top of the display to save your map.

Using Paddles with the App's Keyer

Marcus' iOS/MacOS apps include a keyer that you can control with the paddles connected to the ***MIDI***.

> **NOTE:** If you're using Bluetooth-LE MIDI you may notice a difference in how the paddles work with the app's keyer as opposed to a hardwired paddle/keyer. The latency in the Bluetooth-LE connection causes up to 15 milliseconds of delay. This will throw your "fist" timing off, especially in Iambic modes where an extra element may be added. It may take some practice to get use to the timing change. If this is a problem for you consider using USB MIDI instead.

Follow these steps to configure the app and the ***MIDI*** for paddle keying:

- In the app, map **Button 20** to **Trigger CW Left Paddle** and **Button 21** to **Trigger CW Right Panel**. (Use **Button 96** and **97** if **Extended BTN mode** is ***enabled***). ***You*** can swap this mapping if your paddles are wired backwards.
- Plug your paddles into the **Paddle In** jack on CTR2-MIDI.
- Verify that the **Paddle Mode** is in ***Normal*** *mode* on the ***MIDI*** (the green LED is off). To toggle this mode, long-press-and-release the knob while in the **Home** knob mode (both yellow LEDs off). You should hear two tones (high to low frequency) and the green LED will turn off.
  NOTE: If **Paddle Mode** is in ***extended*** mode the green LED will be on and the paddles will control MIDI **Buttons 30** and **31** (or 98 and 99 if **Extended BTN mode** is ***enabled***).
- Set the radio's mode to CW
- In SmartSDR, press the **View** menu in the Panadapter display and select **CWX Panel**. You can adjust speed and other setting here.
- In SDR-Control, select the **CW Keyer** option in the **Tools** menu to configure the keyer. DO NOT enable the MIDI device in this panel, this is done in the **CTR2-MIDI Controller** panel.
- Pressing either paddle will now activate the keyer and key the radio.

Using a Straight Key or an External PTT Switch

Straight Key Wiring

If you want to use a straight key with the MIDI, map **Button 30** (or 98 if **Extended BTN mode** is ***enabled***) to **Trigger CW Straight Key** in the app then wire your straight key to the TIP and SHIELD of a 3.5mm (1/8") stereo jack and insert it into the **Paddle In** jack on the ***MIDI***. Do not use a mono jack as the shield will ground the RING lead on the **Paddle In** jack.

PTT Switch Wiring

If you want to use an external PTT switch with the ***MIDI***, map **Button 31** (or 99 if **Extended BTN mode** is ***enabled***) to **PTT Push** in the app and wire your PTT switch to the RING and SHIELD of a 3.5mm (1/8") stereo jack and insert it into the **Paddle In** jack on the ***MIDI.***

NOTE: You can wire your straight key and external PTT switch to the same stereo plug.

NOTE: Re-map your radio control app if you need to reverse the paddle wiring or change which jack input is used for your straight key or PTT switch.

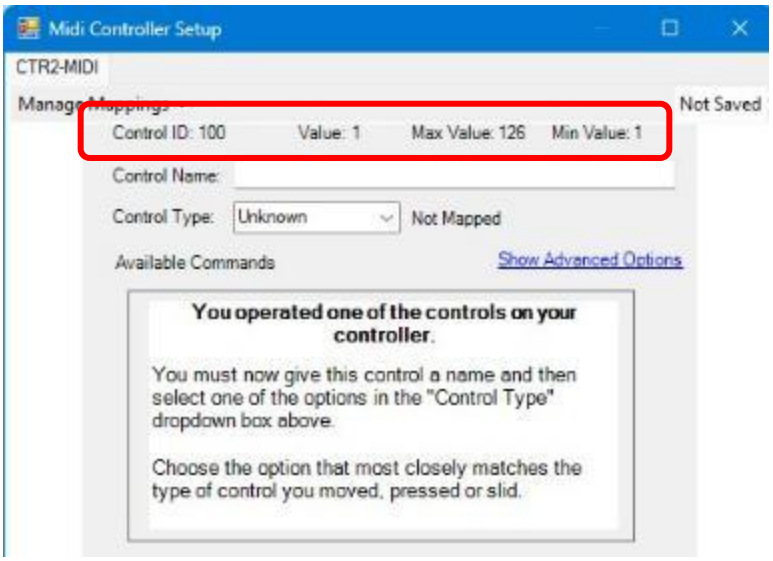
Long-press-and-release the knob while in the **Home** knob mode (yellow LEDs off) to toggle the **Paddle Mode** from **Normal** to **Extended**. You should hear two tones (low to high frequency) and the green LED will light when in **Extended** paddle mode.

Mapping for Thetis

Thetis is a little different than RHR and iOS/MacOS apps in that it uses the ***MIDI's*** **WheelB** control instead of **WheelA** for frequency tuning. **Map 2** is pre-configured with **WheelB** for tuning. Press-and-hold BTN3 then press-and-release the knob to switch maps.

WheelB-r reverses the direction of the tuning knob and is primarily used with the PI HPSDR app.

With the **Setup -> Serial/Network/Midi CAT** window open select the **MIDI** tab and click the **Configure MIDI** button. The program will initialize ***MIDI*** using USB.



NOTE: If the program doesn't initialize the ***MIDI*** check to make sure you are using a USB-C data cable, not a USB-C charge cable.

Mapping Controls

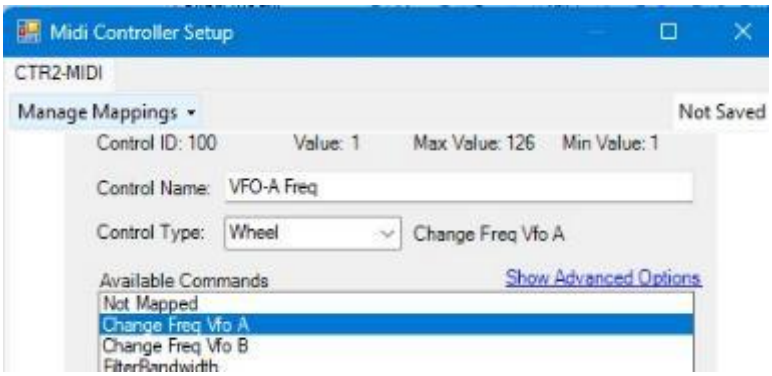
Press a button or turn the knob and the window at the left will open.

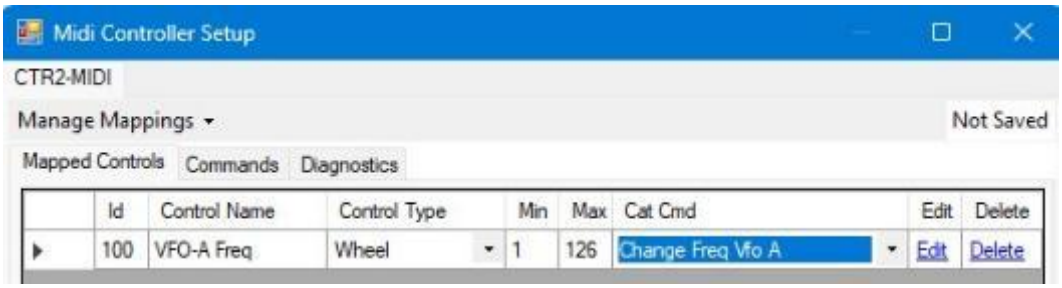
In this example I turned knob when the ***MIDI*** was set to the **Home** knob mode so it is controlling **Change Control (CC) 100**.

NOTE: When mapping knob controls you need to turn the knob fully CCW until you get to 1 and then fully CW until you get to 127 to set the Max and Min range values.

Next, I give the control a name. I'll call it **VFO-A Freq** and select **Change Freq VFO A** from the **Available Commands** list.

Click the **Done** and then the **Save** button to return to the **Manage Mappings** page.





You've mapped your first control! Now follow the same procedure to the other 7 knob functions and the 12 button functions.

Here's a screenshot of the mapping I currently have in Thetis. Don't be afraid to experiment with the settings so you can find the combination that works best for you.

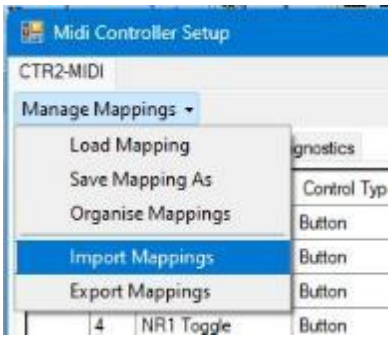
NOTE: When using the Thetis ***Wheel*** *control type* you must [set the knob type on the ***MIDI*** to **WheelB**](#).



When you're finished mapping controls make sure to click the **Save** button in the **Midi Controller Setup** page and **Apply** before you leave the **Setup** page so these functions will be restored the next time you run Thetis.

If you ***enable*** **Extended BTN mode** you can map additional button controls in Thetis to select specific modes, bands, and other features in the app.

Once you've completed your map you can export it to share with others. You can also import a map from someone else by clicking the **Manage Mappings** tab at the top of the **Midi Controller Setup** display. I've included a copy of my map in the firmware zip file.



Advanced Options

Operation with N1MM Logger+

You can use the ***MIDI*** as an interface to use N1MM's CW keyer with SmartSDR or SDR-Control for iOS. To do this, follow the instructions on the N1MM Interfacing Basics webpage to build a key interface circuit. This will require a USB serial adapter with a keying interface circuit wired to its RTS output. Connect the output of the keying interface circuit to the TIP and SHIELD of the ***MIDI****'s* Paddle input jack. Next, map the Left Paddle input on the ***MIDI*** to **Trigger CW Straight Key**. With this setup you can control your radio using the N1MM Logger+ and use keyboard keying in N1MM Logger+.

If you wire your paddle in parallel with the N1MM+ straight key you can switch between you paddles and the N1MM+ keyer by switching Extended Paddle Mode (long-press-and-release the knob while in the **Home** knob mode – both yellow LEDs off).

Unfortunately, this configuration does not allow N1MM Logger+ to control your radio remotely.

If you're looking to have N1MM Logger control your radio remotely you might consider using CTR2-Micro to remotely control and key your radio with SmartSDR or SDR-Control providing the user interface. Instructions for this can be found in the CTR2-Micro operation manual under the **Advanced Options** section.

Troubleshooting CTR2-MIDI Firmware

The table below helps you troubleshoot problems with CTR2-MIDI firmware.

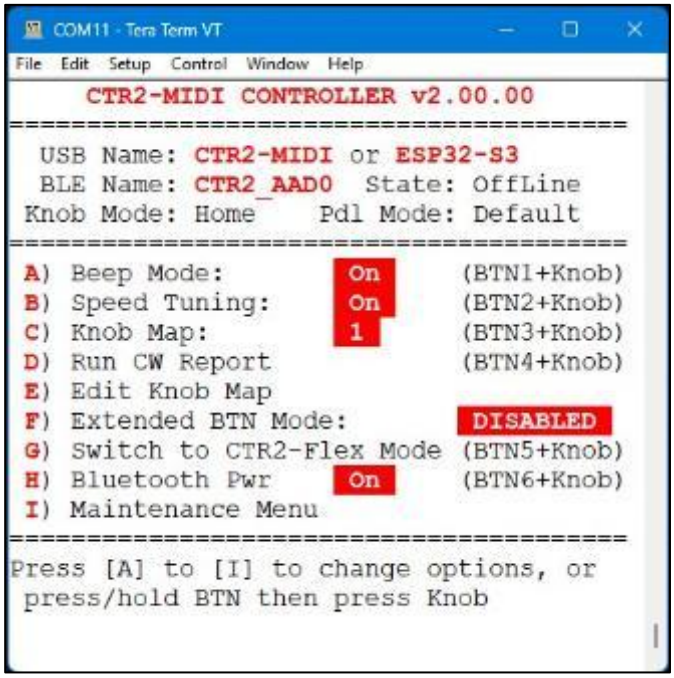
Issue	Solution
I can't connect the app to the CTR2-MIDI using USB MIDI	1) Make sure you are using a USB-C data cable. Many USB-C cables are power only. These will not work with USB MIDI. 2) If using a mobile device (iPhone, iPad, Android, etc) you must use an OTG adapter for your device. This adapter provides a USB connection for an external device. Newer iPhones and iPads with USB-C connectors do not required an OTG adapter.
I can't connect the app to the CTR2-MIDI BLE device	Make sure the Bluetooth radio is turned on and the app has permission to use Bluetooth. The green LED will flash once every 2 seconds when the radio is on. If it's off (green LED flashes once every 5 seconds), press-and-hold BTN6 then press-and-release the knob to turn it on. Follow the instructions in the Connecting CTR2-MIDI to the App and Enable Permissions sections.
I have to connect CTR2-MIDI every time I start the app.	CTR2-MIDI will stay connected the app on your iOS or Mac device with Bluetooth-LE as long as it remains powered up. If you remove power from CTR2-MIDI you'll need to open the **Tools -> CTR2 Controller** window and click the **Find Bluetooth-LE MIDI Device** button then click **Enable** to reconnect it to the app. When connecting with USB MIDI it should say connected even through a power cycle.
My Flex radio starts sending a string of Dits or Dahs when I select CW mode	This seems to be a bug SmartSDR v6.9.15. If you press the paddles when the app is not in CW mode but the CWX panel is open that paddle press is buffered in the app. When you change to CW mode from another mode CWX triggers the last paddle press you made and keeps sending it until you press the paddle again, turn off CW mode, or close the CWX panel. This should be fixed in the latest update of SmartSDR.
The ***MIDI*** connects to my device but does not control anything	1) Verify that you have mapped each ***MIDI*** control to a function in your app. 2) Verify that you are in the correct **Extended BTN mode** for the mapping in your app.
Frequency tuning is not working	1) Make sure you have the right knob map selected. Press-and-hold BTN3 then press-and-release the knob to switch maps. 2) Use **WheelA** for RHR and Marcus' iOS and MacOS apps. Use **WheelB** for Thetis and other Windows apps. **WheelB-r** reverses the tuning direction and is used with PI HPSDR. 3) Make sure you have the knob mode set to the mode you programmed frequency control to. I use the knob's **Home** mode for frequency tuning. 4) Make sure you have the correct MIDI command mapped to the frequency control in your app.
How can I tell when I'm at the center (0 Hz) when controlling RIT or XIT?	Use the **SliderB** control for RIT and XIT controls. This control beeps and pauses tuning for about ½ second when the control reaches its center position (64 counts) and beeps twice when reaching the upper or lower end of tuning. **SliderA** does not beep at center or at the edges. Its best for most other knob or dial controls.

Pressing an BTN executes the wrong parameter	1) Verify that the MIDI mapping is correct in the app. If it is, you may need to be recalibrate the BTN ADC counts. Connect a terminal to the unit's virtual serial port and select the **Recalibrate Buttons** option to start the recalibration process. 2) Verify that you are in the correct **Extended BTN mode** for the mapping in your app.
Speed tuning is not working	Speed sensitive tuning (tuning step changes as you tune faster) must be set to **Normal** or **Fast** in the ***MIDI****'s* settings. Press-and-hold **BTN2** then press-and-release the knob to step through the speed settings. Morse output S0=Off, S1=Normal, S2=Fast. You may also have the wrong knob map selected on the ***MIDI***. Press-and-hold BTN3 then click the knob to toggle maps.
Paddles don't key the radio	1) Verify the radio is in CW mode 2) Verify the keyer is in **Breakin** mode and **Sidetone** in **On** in the P/CW panel 3) Verify the **Paddle Mode** is in **Normal** mode (green LED is off) 4) Verify MIDI Buttons 20 and 21 are mapped to **Trigger CW Left Paddle** and **Trigger CW Right Paddle** 5) Open the CWX Panel in the app's **View** menu 6) If you're using a Flex radio, cycle the power on the radio… it might be in CW Zombie mode.
Paddles are reversed	Remap MIDI Buttons 20 and 21 (or 96 and 97 if you're using the **Extended BTN mode**) in your app to swap Left and Right paddle assignments
No sidetone in SmartSDR when keying	Open the left pop-out window in the Panadapter display on the app and click the **Audio** menu. Set the **Local Audio Monitor** slider to 0.
Slow response or timing issues with keyer	Connecting multiple Bluetooth devices to your iOS device (i.e. CTR2-MIDI and a BT headset) may affect the app's keyer response to paddle input. consider using USB MIDI with an OTG adapter on your mobile device.
You can connect CTR2-MIDI to RemoteTx using the **XIAO_ESP32S3 HID** object, but it doesn't control the app	Make sure CTR2-MIDI is set to the **MIDI mode**, not the **Flex mode**. The ***MIDI*** does not send HID reports in the **Flex mode**. The ***MIDI*** will send **M** or **F** in Morse when it boots to let you know which mode it is in (MIDI or Flex). To check which mode you are in after booting, you can connect a terminal to the ***MIDI***, or press-and-hold BTN5**,** then press-and-release the encoder. If you hear **M?** in Morse (i.e. "Do you want to go to **MIDI mode?**), you are in **Flex mode**. Press the encoder again to accept and switch to **MIDI** mode.

Appendix A: CTR2-MIDI Terminal Menus

This appendix is for the **CTR2-MIDI** terminal menus.

Many of the ***MIDI**'s* options can be changed using a "hold button and knob press" sequence. If you want to edit the knob maps, toggle Extended BTN Mode, Export or Import setting files from your PC, recalibrate the buttons, or reset the ***MIDI*** to factory settings, you'll need to connect a terminal program such as Putty or Tera Term to the ***MIDI**'s* virtual USB serial port. Apple and Linux users can use the built-in terminal program on their computer. Information on connecting a terminal program can be found in **Appendix D**, **E**, and F.



Once you connect the terminal to ***MIDI*** press any key on the terminal's keyboard to open the ***MIDI**'s* terminal display shown here.

Select an option by pressing the key associated with that option. For instance, to cycle through the **Beep** modes, press the [a] key.

The various options in the terminal **Home** menu display are described next.

Device Names

When connecting the ***MIDI*** to the device running the radio control app using the USB-C connector it will show up on the device using one of two device names. On Windows it shows up as **CTR2-MIDI**. On Apple iOS and MacOS devices it shows up as **XIAO_ESP32-S3**.

When you connect to an Apple device using Bluetooth-LE MIDI the ***MIDI*** will show up as **CTR2_** plus a four-digit hexadecimal number derived from the ***MIDI**'s* MAC address. This naming convention gives every ***MIDI*** a different Bluetooth-LE device name so you can use more than one ***MIDI*** on an Apple device, (but not at the same time!)

NOTE: I write the four-digit address on the bottom label of all assembled ***MIDIs**.*

The BLE **State** on the terminal display will show the connection status of the Bluetooth-LE MIDI connection. The green LED on the ***MIDI*** will flash twice once a second when the ***MIDI**'s* Bluetooth-LE is online. If the Bluetooth radio is turned off the status will show *BLE Off*.

> **NOTE:** BLE MIDI is not supported on Windows.

Knob Mode

The knob mode will be shown here. You can step through the knob modes by pressing and releasing the knob. The yellow LEDs will show the selected mode.

Pdl Mode

When **Pdl mode** is set to ***default***, the ***MIDI*** sends MIDI controls 20 and 21 for the left and right paddles (or 96 and 97 if Extended BTN mode is ***enabled***). These are usually mapped to the *Trigger CW Left paddle* and *Trigger CW Right paddle* functions in the iOS/MacOS apps. When the *Extended* **Paddle mode** is selected (by long-pressing the knob in the knob's **Home** mode – yellow LED are off) MIDI controls 30 and 31 (or 98 and 99 if Extended BTN mode is ***enabled***) are sent with the left and right paddles. These can be mapped to the *Trigger CW Straight Key* and *PTT Push* functions in the SmartSDR app. **Not every app has keyer or straight key MIDI controls.**

Beep Mode

Press **A** (or press-and-hold **BTN1** then press-and-release the knob) to step though the **Beep** modes. When **Beep** is *off* the LEDs will momentarily flash when you press a control long enough to enter its long-press function. When **Beep** is *normal* pressing any button or the knob will sound a beep. If **Beep** is set to *Long-Press* the ***MIDI*** will only beep when you press a control long enough to enter its long-press function.

The selected beep mode will be sent in Morse, **B0**= Beep mode Off, **B1**= Beep mode Normal, **B2**= Beep mode Long-Press.

Speed Tuning

Press **B** (or press-and-hold **BTN2** then press-and-release the knob) to step through the speed (proportional) tuning settings for **WheelA**. There are three options, *Off, Normal,* and *Fast*. The selected tuning mode will be sent in Morse, **S0**= Speed tuning Off, **S1**= Speed tuning Normal, **S2**= Speed tuning Fast.

Knob Map

Press **C** (or press-and-hold **BTN3** then press-and-release the knob) to toggle between **Knob Map 1** and **Knob Map 2**.

The selected map will be sent in Morse, **M1**= Map 1 is active, **M2**= Map 2 is active.

Run CW Report

Press **D** (or press-and-hold **BTN4** then press-and-release the knob) to have the ***MIDI*** report all of the option settings in Morse code. You can press any button during the report to silence it.

The report uses the following format:

M	**M** represents MIDI mode. In **Flex** mode the report starts with **F**
B{0, 1 or 2}	**Beep** mode, 0=off, 1=on, 2= Beep for long-press only
S{0, 1, or 2}	**Speed Tuning** mode, 0=off, 1= normal, 2= fast
M{1 or 2}	**Map #**
BT{0 or 1}	**Bluetooth Radio** mode, 0=off, 1=on

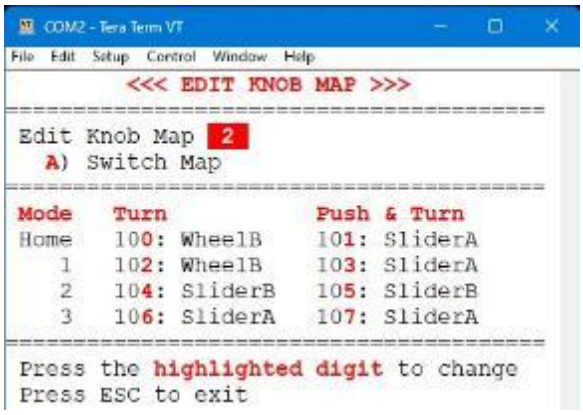
Edit Knob Map

Press **E** to open the **Edit Knob Map** display. Here you can change the MIDI control type assigned for each knob function. Press the **A** key to switch between the two maps.

Knob Map 1 is typically used with RHR and Marcus' iOS/MacOS apps. **Knob Map 2** is typically used with Windows apps such as Thetis and SDR-Console.

Press the **Esc** key to exit this screen.





Four knob *Modes* with *Turn* and *Push & Turn* controls (for a total of 8 functions) are available on the knob. *Turn* mode controls are executed when you turn the knob, and are shown in the left column. They are even numbered starting at 100 (the MIDI **Control Change** command number). The *Push & Turn* controls are executed when you press and turn the knob, and are shown in the right column. They are odd numbered starting at 101.

> **IMPORTANT:** You can only change the MIDI control type in the ***MIDI*** map. The actual radio parameter that each control changes must be mapped to the MIDI control # sent by the ***MIDI*** for each function.

Each knob function supports one of the following MIDI control types. To change the MIDI control type assigned to a knob function, press the highlighted hotkey (the red digit in the control #) on the terminal's keyboard.

- **Button** – in this mode turning the knob CCW or CW sends a MIDI **NoteOn {Button #}** for each knob tick based on the table below. There are 24 knob ticks per revolution. These commands can be used to change the frequency on iOS/MacOS apps when they are mapped to the **Button** *Frequency Down* and *Frequency Up* in the app.

Turn	CCW Button #	CW Button #	Push & Turn	CCW Button #	CW Button #
100	40	41	101	42	43
102	44	45	103	46	47
104	48	49	105	50	51
106	52	53	107	54	55

- **SliderA** mode emulates a potentiometer. It sends a MIDI **Control Change** (aka **CC**) value for each knob tick to the app indicating the position of the slider. Turning the knob CCW decrements the count value down to 0 and turning the knob CW increments the count value up to 127. The ***MIDI*** remembers the last count position through power cycles. This mode is used for controls that would use physical knobs or sliders such as volume, squelch, power, etc.
- **SliderB** is similar to **SliderA** except it adds "audible detents". The ***MIDI*** will beep once at the center (64 counts) and twice at the lower and upper limits of the control (0 and 127 counts). Knob changes are blocked for 600 milliseconds after the center beep to give you time to stop turning. **SliderB** is useful when using the knob for RIT and XIT tuning as the center beep indicates when RIT or XIT is turned off and the edge beeps indicate when you have reached the end of the tuning range.
- **WheelA** is similar to the **Slider** controls in that it uses a MIDI **CC** command to send a value to the app. This control has an imaginary center position at 64 counts. Turning the knob counter-clockwise sends a value below 64 for each knob tick. Turning it clockwise sends a value above 64 for each knob tick. This control is speed sensitive. Turning the knob slowly sends a value of +/- 1 count from center. Turning it moderately fast (one revolution every two seconds) sends a value of +/- 10 counts from center. Turning it fast (about 1 revolution per second) sends a value of +/- 50 counts from center. ***RHR and Marcus' iOS/MacOS apps use this scheme for frequency control and the frequency change per count depends on the Tune Step selected in the app.***
- **WheelB** is a variant of **WheelA**. Unlike **WheelA** this control has no center position, it only tells the app which direction the knob is being turned. Turning the knob counter-clockwise sends a value of 126 for each knob tick. Turning it clockwise sends a value of 1 for each tick. ***Thetis and SDR-Console use this scheme for frequency control. Frequency change per tick is based on the Tune Step set in the app.***
- **WheelB-r** is the same as **WheelB** except the tuning direction is reversed. This MIDI type is usually only used with the PI HPSDR app.

Extended BTN Mode

Press **F** to toggle the **Extended Button** mode. When **Extended BTN** mode is ***disabled*** the 12 MIDI Button controls remain the same for all knob modes. When **Extended BTN** mode is ***enabled***, the MIDI Button control sent is determined by the current knob mode. The chart below shows the BTN# / MIDI Button assignments. The user must map the MIDI Button controls in their radio control app to execute the desired functions in the app.

WARNING: You must change the MIDI mapping in the radio control app if you change the **Extended BTN** Mode.

The following table shows the BTN and Paddle Input Jack to MIDI # assignments when **Extended BTN** mode is ***disabled***. Worksheets are included in Appendix H for documenting these assignments.

BTN	Short-Press Function	Long-Press Function
1	MIDI 1	MIDI 11
2	MIDI 2	MIDI 12
3	MIDI 3	MIDI 13
4	MIDI 4	MIDI 14
5	MIDI 5	MIDI 15
6	MIDI 6	MIDI 16

Paddle Input Mode MIDI Assignments	
Normal	Extended
TIP -> MIDI 20	TIP -> MIDI 30
RING -> MIDI 21	RING -> MIDI 31

The following table shows the BTN and Paddle Input Jack to MIDI # assignments when **Extended BTN** mode is ***enabled***. Worksheets are included in Appendix I for documenting these assignments.

Knob Mode	BTN	Short-Press	Long-Press	Knob Mode	BTN	Short-Press	Long-Press
Home ○ ○	1	MIDI 1	MIDI 25	2 ● ○	1	MIDI 13	MIDI 37
	2	MIDI 2	MIDI 26		2	MIDI 14	MIDI 38
	3	MIDI 3	MIDI 27		3	MIDI 15	MIDI 39
	4	MIDI 4	MIDI 28		4	MIDI 16	MIDI 40
	5	MIDI 5	MIDI 29		5	MIDI 17	MIDI 41
	6	MIDI 6	MIDI 30		6	MIDI 18	MIDI 42
1 ○ ●	1	MIDI 7	MIDI 31	3 ● ●	1	MIDI 19	MIDI 43
	2	MIDI 8	MIDI 32		2	MIDI 20	MIDI 44
	3	MIDI 9	MIDI 33		3	MIDI 21	MIDI 45
	4	MIDI 10	MIDI 34		4	MIDI 22	MIDI 46
	5	MIDI 11	MIDI 35		5	MIDI 23	MIDI 47
	6	MIDI12	MIDI 36		6	MIDI 24	MIDI 48

In **Extended BTN** mode The Paddle Input Jack is mapped to the following MIDI #s based on the Paddle Input Mode:

Paddle Input Mode MIDI Assignments	
Normal	Extended
TIP -> MIDI 96	TIP -> MIDI 98
RING -> MIDI 97	RING -> MIDI 99

Switch to CTR2-MIDI Flex Mode

Press **G** (or press-and-hold **BTN5** then press-and-release the knob) to switch to **Flex** mode. An acknowledgement screen will open if you have a terminal connected and ***F?*** will be sent in Morse. You can press **Y** on the terminal or press-and-release the knob switch to execute the switch.

Bluetooth Pwr

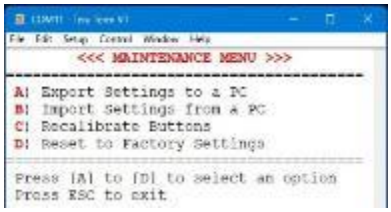
Press **H** (or press-and-hold **BTN6** then press-and-release the knob) to toggle the Bluetooth radio's power. When powering the ***MIDI*** from a portable device such as an iPhone or iPad you can use USB MIDI (with the appropriate cable or OTG adapter) and turn the power off on the Bluetooth radio. This will cut the current consumption from about 93 mA to 43 mA.

Pressing **H** will open an acknowledgment screen. If you enter **Y** the ***MIDI*** will send the new state of the Bluetooth radio in Morse (**BT0**=Off, **BT1**=On) then reboot.

NOTE: The Bluetooth power option is not available when **CTR2-MIDI** firmware is running on a **CTR2-Micro** hardware because the *Micro* hardware does not support USB MIDI and must use Bluetooth MIDI.

Maintenance Menu

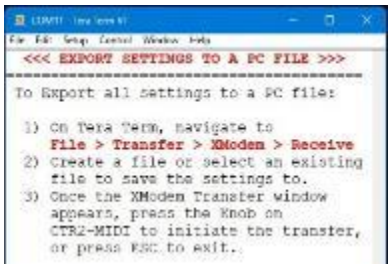
The **Maintenance** menu is where you can backup and restore the settings on your unit, recalibrate the BTNs, and reset your unit to factory settings.



NOTE: You need to use a terminal program that supports XModem such as Tera Term to **Export** and **Import** settings files. It will be used here to explain the process.

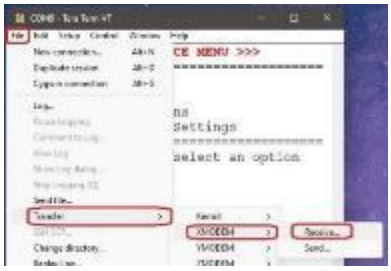
Export Settings

Press **A** to export the settings on your unit to a file on your computer. The terminal will display instructions on how to export your settings.



NOTE: This screen will time-out after 60 seconds if a transfer hasn't been started. You can also press any BTN on the unit to exit this screen.

Next, setup Tera Term to receive your data. Select **File->Transfer->XModem->Receive…** and enter the name of the settings file you wish to create. If you enter an existing file name, you'll be asked if you want to overwrite it.



I'll create a file named *test1.txt* for this example. You can use any valid file name and extension for the file name.

NOTE: I recommend using the .txt extension since the file to be created is a text file and can be edited with a text editor.

Once you've entered the file name, the **XModem Receive** dialog box will appear. At this point, Tera Term is waiting for the unit to send its settings.



>>> Press-and-release the knob on the *MIDI* to start the transfer.

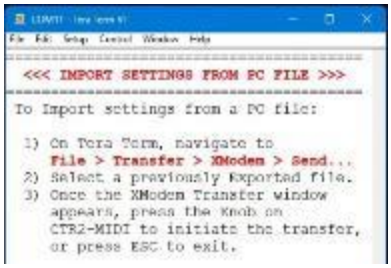
Once the transfer completes, you'll be returned to the **Maintenance** menu.

NOTE: Your WiFi and Radio IP settings are encrypted in the settings file for security.

Import Settings

Press **B** to import a previously exported settings file from your computer. This option uses **XModem Send** method to send the file.

To initiate the **Import**, press **B** to start the process. The screen at the right will appear giving you instructions on how to proceed.



NOTE: This screen will time-out after 60 seconds if a transfer hasn't been started. You can also press any BTN on the unit to exit this screen.

Next, select **File->Transfer->XModem->Send…** in Tera Term, then select a previously exported file from the file list to send to the ***MIDI***.



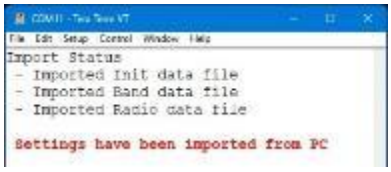
In this example, I've selected a file named *test1.txt*.

The **XModem Send** dialog box will appear indicating that Tera Term is waiting for the unit to indicate that it is ready to receive the file.



>>> Press-and-release the knob on the *MIDI* to initiate the transfer.

Once the transfer is complete, an **Import Status** screen will be briefly displayed indicating the status of each file imported. If any of the files indicate the import failed, try the **Import** function again.



The program will return to the **Maintenance Menu** once the transfer has completed.

Recalibrate Buttons

Each button on the ***MIDI*** presents a specific voltage to the ADC (analog/digital converter) on the processor. This allows the processor to determine which button is pressed.

To ensure the highest repeatability, the buttons should be calibrated to compensate for manufacturing tolerances.

Press **C** on the **Maintenance** menu to start the calibration process.

Once the **BTN CALIBRATION** menu appears, press each button, one at a time, in sequence, starting with **BTN1** and working your way around to **BTN6**. This records the ADC values for each button. Once you've finished, the calibration screen will update and then return back to the **Maintenance** menu.



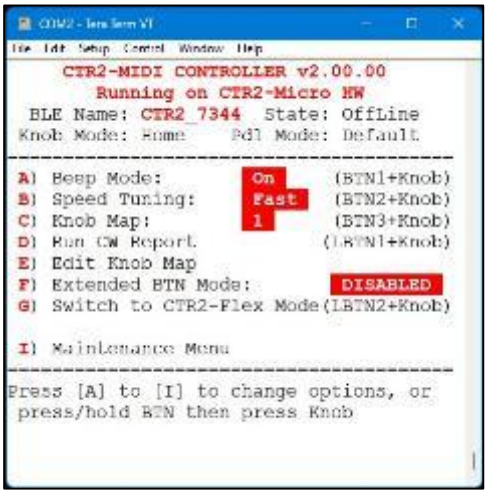
Reset to Factory Settings

Press **D** on the **Maintenance** menu to reset the unit back to factory settings. You will need to recalibrate the buttons, and re-enter your WiFi credentials, radio IP address, and call sign.

NOTE: If you don't want to lose your settings, Export them to your PC before resetting back to factory. After the reset, import them back into the ***MIDI*** using the Import function. You'll also want to export your settings if you plan on erasing the flash memory when updating the firmware.

Appendix B: Using CTR2-MIDI Firmware on CTR2-Micro

The CTR2-MIDI firmware can be flashed to CTR2-Micro. This gives current *Micro* users access to the ***MIDI*** features without having to buy new hardware.



NOTE: Because of the way the buttons are sampled on **CTR2-Micro** hardware (with the ESP32-C3 processor), you must recalibrate the buttons when switching between **CTR2-Micro** firmware and **CTR2-MIDI** firmware.

The terminal display for ***MIDI*** firmware running on the *Micro* is similar to the normal ***MIDI*** display except the **USB Name** and the Bluetooth Power option is not shown because these options are not available on the *Micro.*

There are several limitations to using the ***MIDI*** firmware on the *Micro*:

- **CTR2-Micro** hardware only supports Bluetooth-LE MIDI and WiFi. The ESP32-C3 processor in the *Micro* doesn't have the hardware to support USB MIDI.
- Once you have flashed the ***MIDI****'s* firmware on your *Micro* most of the features you use on the *Micro* will no longer be available. This includes the following:
    - WiFi is not supported in **MIDI** mode. To control a Flex radio over WiFi, switch to **Flex** mode. **Flex** mode does not support CTR2-Voice
    - Serial CAT is not supported so you can't control other radios with the *Micro*
    - The *Micro's* keyer is not supported so you can't use a terminal as a keyboard keyer or use any CW message buffers you had set up in the *Micro*. **Flex** mode does have a built-in keyer and has its own set of CW message buffers.
    - Physical Key and PTT Output are not supported; however, **Flex** mode supports Key and PTT over WiFi
    - Favorite Frequency and Previous Frequency lists are not supported
    - The web browser interface is not supported
    - You must reflash the **CTR2-Micro** firmware on your *Micro* to restore its normal features
- The *Micro* only has three buttons so you can only map 6 MIDI **Buttons** in the app (or 24 if you enable Extended Button Mode).
- The *Micro* only has one LED. This LED flashes to indicate the current mode of the unit. It does not indicate long-press mode if **Beep** mode is turned *Off*.
- CW Report and Switch Mode options (**BTN4+Knob** and **BTN5+Knob**) are accessible using *long-press* **BTN1+Knob** and **BTN2+Knob** sequences.

Flashing the Firmware

To flash the ***MIDI*** firmware on to your *Micro,* unzip the four binary files from the ***MIDI*** firmware distribution zip file into a unique folder then follow the instructions in **Appendix B**.

You will need to complete the **BTN Calibration** on the *Micro* to save the BTN ADC counts in the ***MIDI****'s* initialization file. This only needs to be done once.

> **NOTE:** The ***MIDI*** firmware maintains its own file structure therefore the settings you had for the *Micro* firmware are not lost. Just reflash the *Micro* firmware to your *Micro* to restore your *Micro* to its normal operation – **do not Erase the flash memory** in the Flash Download tool.

Operating the *Micro* with *MIDI* Firmware

MIDI firmware can run on *Micro* hardware, with a few differences. You will need to go through the same steps to connect the *Micro* to your iPad and you will need to map the controls on the *Micro* to the app. The main difference is that you will only have 6 MIDI **Button** functions (two on each BTN) and all of the knob modes (24 if you enable Extended BTN Mode).

LED Indications

The big difference between the ***MIDI*** and the *Micro* is with the LED indicators (or lack thereof). The *Micro's* status is indicated by the flash sequence of the status LED.

The table below summarizes these flash sequences and the beeps associated with them.

Knob Mode	Flash Sequence	Buzzer Tone when entering this mode	Description
Home	One long flash	One high frequency beep	Primary knob sends **Control Change (CC) 100** Push and turn sends **CC 101**
Mode 1	One short flash	One low frequency beep	Primary knob sends **CC 102** Push and turn sends **CC 103**
Mode 2	Two short flashes	Two low frequency beeps	Primary knob sends **CC 104** Push and turn sends **CC 105**
Mode 3	Three short flashes	Three low frequency beeps	Primary knob sends **CC 106** Push and turn sends **CC 107**

The **Paddle Mode** determines the rate of the LED flash.

- When **Paddle Mode** is in **Normal** mode (controlling MIDI **Buttons 20** and **21)** the LED flashes the status at 2 second intervals
- When **Paddle Mode** is in **Extended** mode (controlling MIDI **Buttons 30** and **31**) the LED flashes the status at 1 second intervals

Modified BTN#+Knob Functions

Since the *Micro* hardware only has three physical buttons the BTN#+Knob options are slightly modified.

The BTN4+Knob function to play the report or change PTT modes is accessed by *long-pressing* **BTN1+Knob**.

The BTN5+Knob function to switch modes is accessed by *long-pressing* **BTN2+Knob**.

The BTN6+Knob function to toggle WiFi (in Flex mode) is accessed by *long-pressing* **BTN3+Knob**.

Appendix C: Installing or Updating CTR2-MIDI Firmware

As of v2.00.00, the ***MIDI's*** firmware is distributed as a single BIN file. This simplifies the installation process and reduces the possibilities of entering the wrong offset address for individual BIN files. The address of the single BIN file always starts at **0x0**.

Kits and assembled **CTR2-MIDIs** have the firmware already installed on them but as you might expect, changes will be made to the program over time. To install the latest MIDI firmware on your device, follow these steps:

1. Determine which version of the firmware you need for your device. If your device is a CTR2-MIDI (with six BTNs) you'll need the **S3** firmware (for the ESP32-S3). If you are running CTR2-MIDI on a CTR2-Micro (with three BTNs) you'll need the **C3** firmware (for the ESP32-C3).

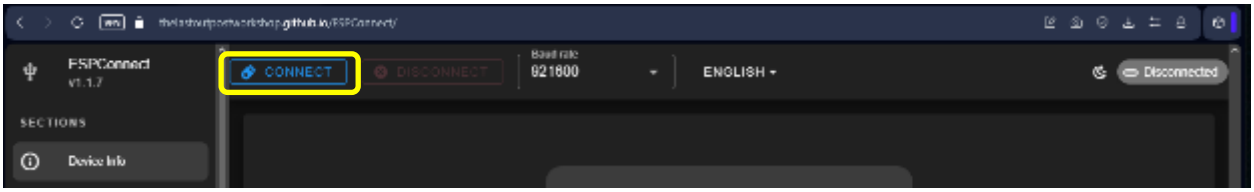
2. Download and unzip the appropriate **CTR2-MIDI** firmware from my web site or the CTR2 Group. Unzip it into a different folder than where you store other CTR2 firmware update files.

Installing Firmware using ESPConnect (New Method)

By far, the easiest method to install or update firmware on an ESP32 process is by using ESPConnect, an open source ESP32 management project.

ESPConnect is a browser-based tool that must be opened on a Chromium-based browser that supports Web Serial, such as Chrome, Edge, or Opera. It will run on a Windows PC or an Apple Mac. Unfortunately, Linux doesn't support Web Serial in any browser.

To start, open **ESPConnect** here: https://thelastoutpostworkshop.github.io/ESPConnect/



Next, plug your **CTR2-MIDI** unit into a USB port on your computer.

NOTE: Do not connect it to an unpowered USB hub.

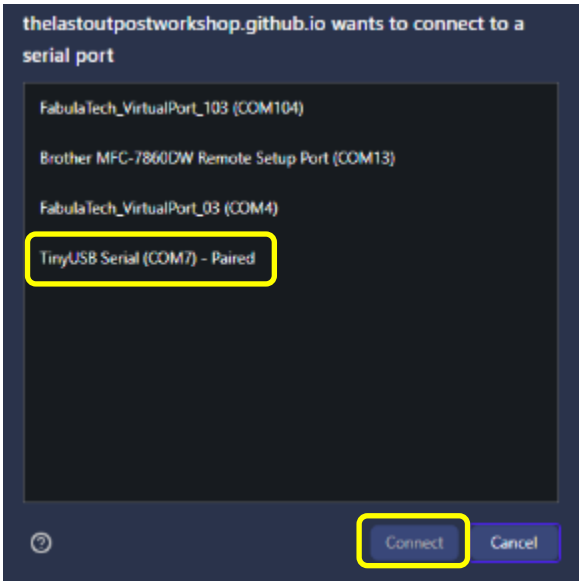
Click the **Connect** button.

A device list will pop up showing the available USB devices on your computer.

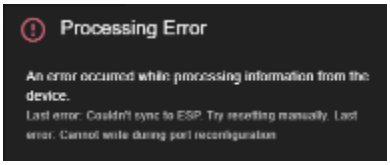
On a Windows PCs, **CTR2-MIDI** will be shown as a **TinyUSB Serial** device, as shown here.

On Macs, **CTR2-MIDI** units will be listed as a **XIAO_ESP32S3** device.

Select the **CTR2** device and click the **Connect** button.



A popup warning will appear telling you that an error occurred. Some errors may tell you to press the BOOT button on the processor to put it into bootloader mode. <u>You do not need to do this.</u>



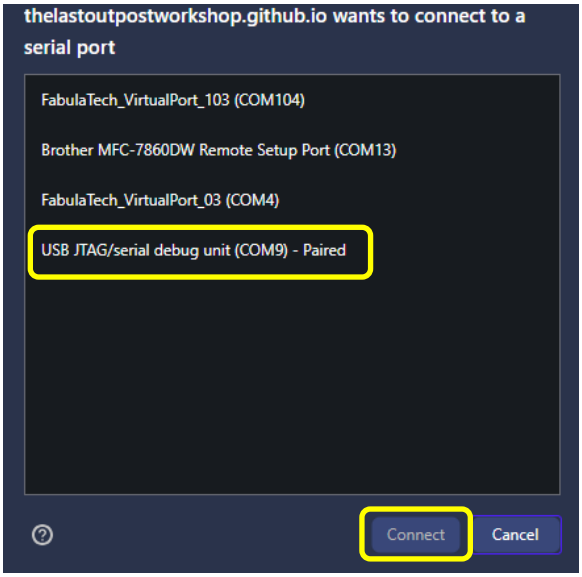
This is the normal response when trying to connect to a **Tiny USB serial** device.

Click the **Connect** button on the home page again.

This time, the **TinyUSB serial** (or **XIAO_ESP32S3** on Macs) will not be shown on the list. Instead, a new device, **USB JTAG/serial debug unit** will be shown (on both PCs and the Mac).

This is the bootloader port and the unit is now ready to program.

Select the **USB JTAG/serial debug unit** device then click **Connect**.



Once you're connected to your **CTR2** unit, the left menu items on the page will be enabled.

<<< WARNING >>>

ESPConnect is a powerful device editor. Many of the functions can cause problems with **CTR2** firmware if you change them. If you don't know what a functions does, DON'T CHANGE IT!

Click on the **Flash Tools** menu item and scroll down to the **Flash Firmware** section.

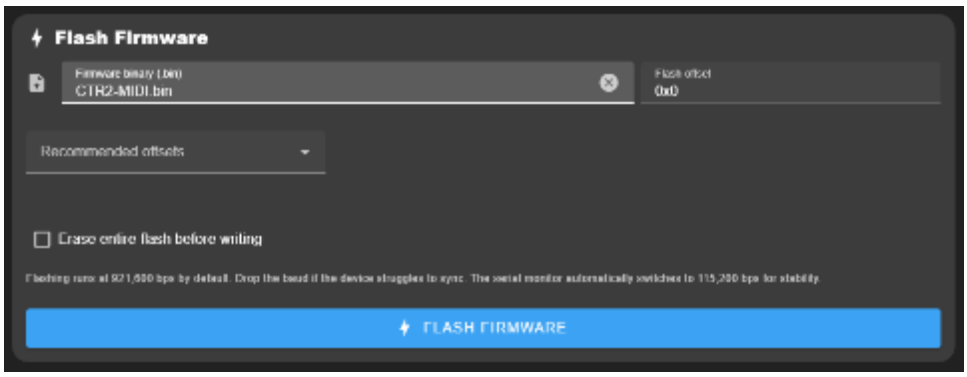
Click the **Firmware binary (.bin)** text box.



A file navigation screen will appear. Navigate to where you unzipped the firmware zip file from the firmware update page and click on the **CTR2-Flex.bin** file.

Your display should now look like this:

Leave the **Flash offset** set to **0x0** and uncheck the **Erase entire flash before writing** checkbox.



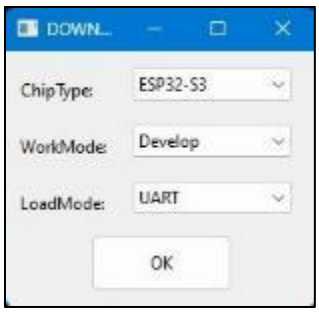
Click the **FLASH FIRMWARE** button to start the flash process.

Once the process completes, cycle the power on your **CTR2-MIDI** unit. The version # will be displayed in a dumb terminal when the unit boots. The version # should match the version # from the firmware zip file.

This completes the firmware update process.

Installing Firmware using EspressIF Flash Download Tool (Old Method)

If you choose to use the EspressIF Flash Download tool, proceed with these steps after you have downloaded the firmware from https://ctr2.lynovation.com/download-ctr2-midi-firmware/



1. Download and open the EspressIF Flash Download Tool. When it starts, select the **Chip Type** your unit is using (*ESP32-S3* for **CTR2-MIDI** or *ESP32-C3* for **CTR2-Micro** hardware). Leave **WorkMode** set to *Develop* and **LoadMode** set to *UART.*

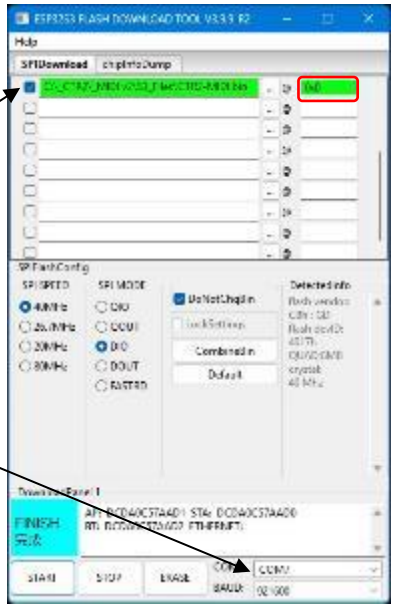
   NOTE: A browser-based flash download tool is also available. See this page on my web site for more information.

2. Map the **CTR2-MIDI.bin** file that you unzipped from the **CTR2-MIDI_v2.xx.xx** firmware distribution file into the downloader tool.

3. Check the checkbox on the left of the **CTR2-MIDI.bin** filename as shown. This tells the program to flash this file.

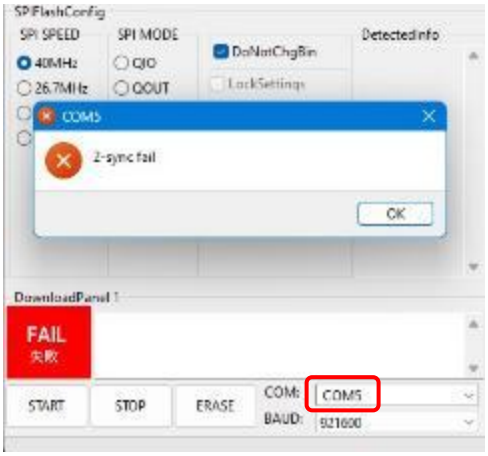
   Set the **COM:** port to the port your computer assigned to **CTR2-MIDI** and set the **Baud** to **921600**.

   NOTE: You must use a USB-C data cable. USB-C charge-only cables (supplied with may devices) will not work.

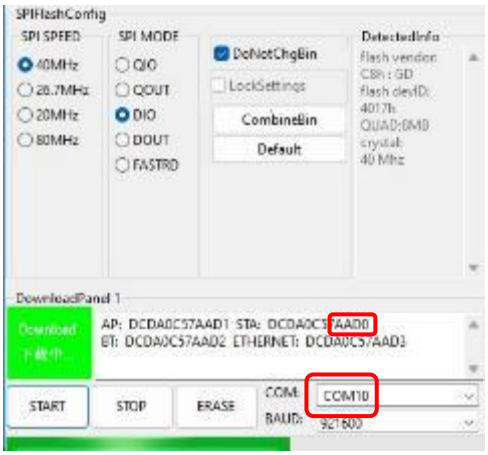


IMPORTANT NOTE FOR ESP32-S3

The ESP32–S3 processor in CTR2-MIDI has two USB UARTs. The first UART (COM 5 in this example, yours will be different) is active when you initially power up the unit and is the COM port you connect to with a terminal program. ***When you select this port the Flash Download Tool will fail and pop up a 2-sync fail window when you press either Start or Erase.*** Press **OK** to clear the alarm window.



Next, drop down the **COM:** list again. You will notice that the initial COM port (COM 5 in this example) is not on the list. You'll also notice a new COM port has been added to the list. On my computer it shows up as COM 10 (your computer will show a different port #). This is the second USB UART on the processor and is used for programming. Select this new COM port.



NOTE: If you don't see the second COM port you will need to press the **PROGRAM** button while you power up the ***MIDI*** to force it into program mode. See the Manually Switch the Processor to Program Mode section for more information on this button. For this reason, I highly recommend using the new process described above to flash your unit.

NOTE: The ESP32-C3 processor in **CTR2-Micro** only has one UART so the above steps are not required. Just select the COM port for the *Micro* when flashing CTR2-MIDI firmware to it.

4. Click the **START** button to start the download.

   NOTE: The unit's Bluetooth ID is based on the last four digits of the Station MAC address that is displayed in the top line of the info box once the flash process starts. In this case it is AADO (circled in the screenshot above). The Bluetooth ID of this unit will be **CTR2_AADO**. Your unit will have a different ID#.

5. Once the download is complete, cycle the power on the unit to start the new MIDI firmware.

   NOTE: The ESP32-S3 processor will revert back to its initial COM port (COM 5 in this example) after the reboot.

NOTE: Clicking the **ERASE** button will erase the entire flash memory including the setup file. This will reset any custom MIDI control types you have selected for **MIDI** mode, and all custom settings in **Flex** mode. It also resets the BTN ADC values back to default values. You will need to recalibrate the buttons when you start the program and edit the maps on the ***MIDI*** (or Import a previously Exported settings file) to restore your custom settings. MIDI Mapping in your app is not affected.

NOTE: If you are flashing the ***MIDI*** firmware to a **CTR2-Micro** (-C3 processor) erasing the flash will also delete the configuration files used with the normal *Micro* firmware.

Installing Firmware using Linux or Mac

Mac users should use the new method using ESPConnect, described above. Linux users have no other option, other than to follow the procedure below.

A script file is supplied in the firmware update zip file. This script file can be used in a Linux or Mac environment if you don't have access to a Windows computer.

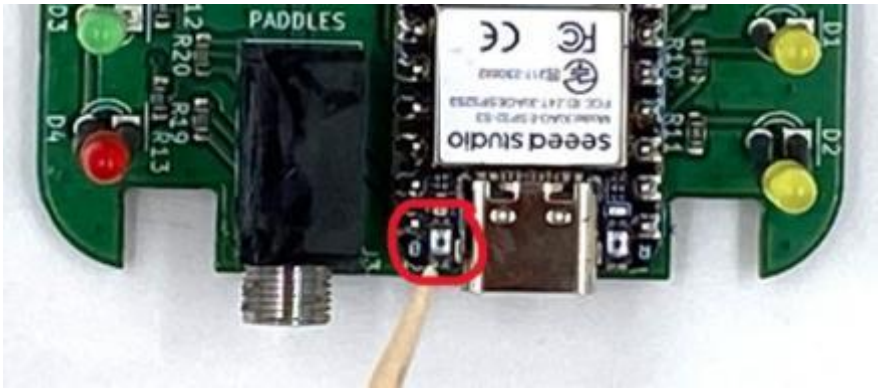
Instructions for using this script file are include in the CTR2-Micro operation manual in **Appendix B**.

> **NOTE:** A browser-based flash download tool is also available. See this page on my web site for more information.

The firmware that allows USB MIDI control, changes the way the virtual COM port works on the ESP32-S3. One COM port is used in the normal operating mode and another COM port is used for flashing the firmware. In order to flash new firmware to the unit you must force the ESP32-S3 to switch from the normal operating mode to programming mode.

Manually Switch the Processor to Program Mode

1) Power the unit off.
2) Press-and-hold the **PROGRAM** button down on the processor board then apply power to the board. Yes, it's that little black dot on the square silver pad the toothpick is pointing to in this photo!



A notch has been cut in the enclosure next to the USB-C connector to allow you to access this button without opening the enclosure.

It's still hard to do especially if your eyes aren't as good as they use to be. Insert the toothpick only about 6mm (1/4") and gently press down. You will feel a light click. If you insert the toothpick too far it will rest on the chip LED in back of the switch and will not press the button. Once you have the button pressed power up the unit. If it's in programming mode the normal LED boot sequence will not run. Use a terminal program to find the Program UARTs COM port assignment.



Once the unit is in Programming mode, edit and run the script to flash the firmware.

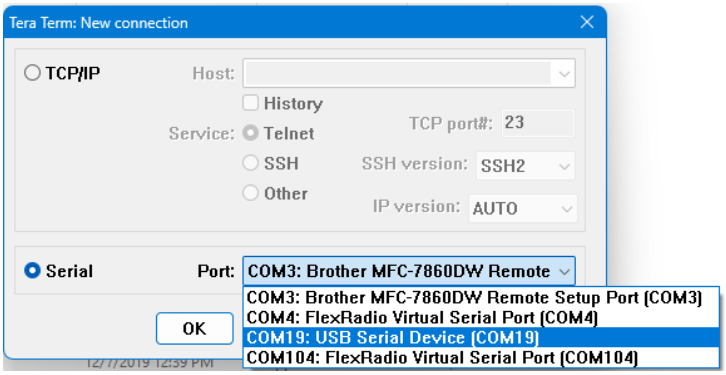
NOTE: To use the script, you must set it as an executable file and grant permission in you OS to access the **DIALOUT** group. You'll also need to edit the file to add the serial port ID and the path to the CTR2-MIDI.BIN file.

Appendix D: Configuring Tera Term

Tera Term is the simplest terminal program to get running for a serial connection.

If you search for Tera Term you will find a lot of garbage with malware attached to it. I've downloaded a clean copy of Tera Term v4.106 and posted it in the CTR2 Group Tera Term Install File section. You can download it here. As far as I know, Tera Term is only available for Windows.

When you first open Tera Term you'll be presented with the **Tera Term New connection** window. Simply select the **Serial** radio button, select the COM port Window's assigned to your Micro when you plugged it in, and click the **OK** button.



Since you are connecting to a USB serial port there is no need to set the baud rate. It will run at USB speed regardless of the baud setting.

That's it! Tera Term will connect to the ***MIDI***. Press any key to open the configuration display.

You can change the terminal size in the **Setup** menu. Select **Terminal…** Set the **Terminal Size** to *41 x 20.* The ***MIDI****'s* terminal interface was designed for this size.

While in the **Terminal…** settings verify the **New-line** options are set to *CR* for both **Transmit** and **Receive** and the **Terminal ID** is set to *VT100*.

You'll probably want to change the font size and colors. These are also changed in Tera Term's **Setup** menu. Select **Display** to change the font and background colors to your liking. Select **Font** to change the font and font size. I like *Courier New*, *Regular*, and *14-point size*. Your preferences may differ.

Once you have the program configured the way you like, select the **Setup->Save Setup…** menu and save your configuration. If you use the default file name, TERATERM.INI the program will automatically start a Telnet session using the COM port you selected above when it opens. This provides one-click access to your ***MIDI***.

Appendix E: Configuring Putty

Putty is a terminal program that can be configured for a variety of needs. The ***MIDI*** only supports serial connections. This section describes how to configure the program to interface with the ***MIDI***.

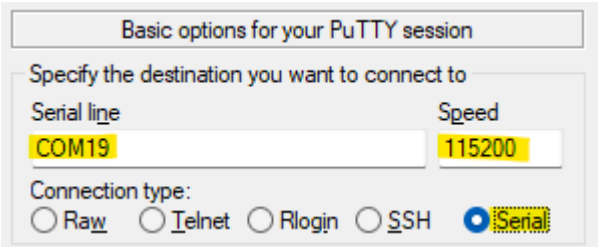
Download Putty for Windows from https://www.putty.org/. It's also available for Linux at https://www.ssh.com/academy/ssh/putty/linux and for Mac at https://www.ssh.com/academy/ssh/putty/mac.

You'll need to connect to the ***MIDI*** using its USB serial port in order to configure it.

Serial Session

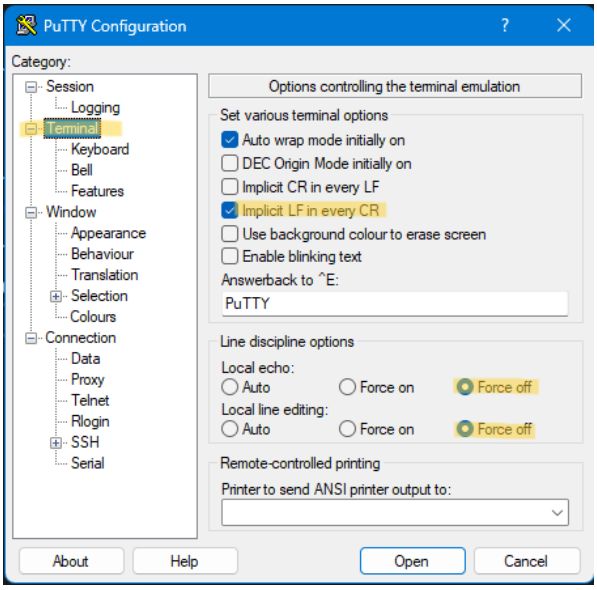
Select **Serial** then set the **Serial Line** to the COM port you found in the Device Manager and set **Speed** (Baud Rate) to 115200.

NOTE: Since this is a USB serial port the **Speed** (baud rate) doesn't matter. Data will be sent at USB speeds regardless of the **Speed** setting.

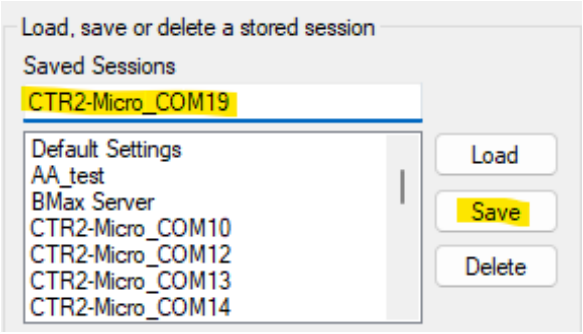


Next, select the **Terminal** item and set the **Implicit LF in Every CR** to on, and **Local Echo,** and **Local Line Editing** to **Force Off**.

You can change the window size under the **Window** item. Set the **Columns** to *41* and the **Rows** to *20*.



Once this has been done, return to the **Session** menu item, enter a name for this session and click the **Save** button. This allows you to easily re-open this session with just a couple of clicks.



If you right-click on the Putty icon in the Windows toolbar the last few sessions you had open will be displayed. Just select the one you want to open it.

You can adjust the display colors on the **Windows->Colours** menu item. The Micro uses the **Bold** attribute to highlight the ***hotkeys*** and other items. I like to set the **Background** color to blue and the **Bold** color to yellow but you can find the colors that work for you. After you get a color combination you like return to the **Session** menu and **Save** the session.

Appendix F: Apple or Linux Terminal Programs

The Apple Mac and Linux have built-in terminal programs so there is no need to install a separate app. To connect your ***MIDI*** to a terminal session, use the following process.

First, list your current serial ports without the ***MIDI*** plugged in.

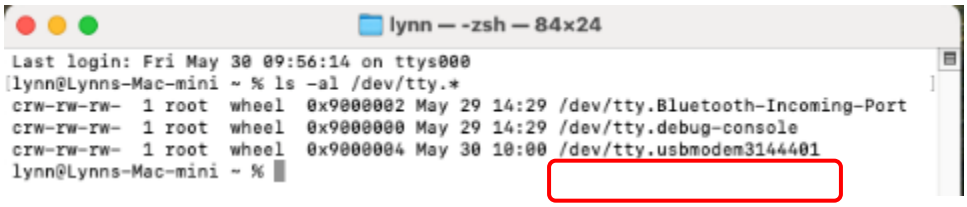
- On the Mac open **Applications/Utilities/Terminal.app**. On Linux open the terminal program supplied by your distro.
- On the Mac, enter ***ls -l /dev/tty.usb****, on Linux, enter ***ls /dev/tty**** This will return a list of all known serial ports.
- Next, plug the ***MIDI*** into the computer's serial port and execute the command above again. This is easily done by pressing the *Up*-arrow key.
- Compare the new list with the old list. The *Flex's* serial port ID will appear on just the new list. For Mac users the serial port ID format will be **/dev/tty.usbserialxxxxx** where **xxxxx** is a unique device ID #. Linux users will see something like **/dev/ttyACMx** or **/dev/ttyUSBx**, where **x** is a unique # for that port.

NOTE: If a new virtual serial port is not created when you plug your Micro into your PC make sure you are using a USB-C cable that supports data. Many USB-C cables only provide power to the remote device.

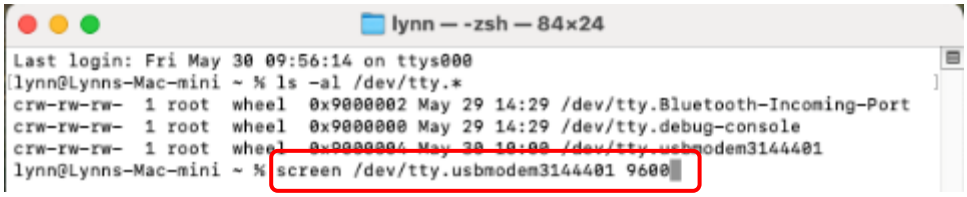
Once you know the ***MIDI****'s* USB serial port ID, write it on the label on the bottom of the unit using a fine-tipped permanent marker for future reference. Put a piece of transparent tape over the label to seal the ink so it doesn't rub off (it's not as permanent as you think). You can always remove the tape if you want to change what's written on the label.

Once you have the serial port ID, enter the following: **screen {serial port ID} 9600**. Include the complete device description (i.e. **/dev/ttyxxxxxxx**) for the **serial port ID**. This will open the serial port using 9600 baud in a terminal session. The following screenshots demonstrate these steps.

1. Get the list of serial devices on your computer. We're looking for the **usb** device.

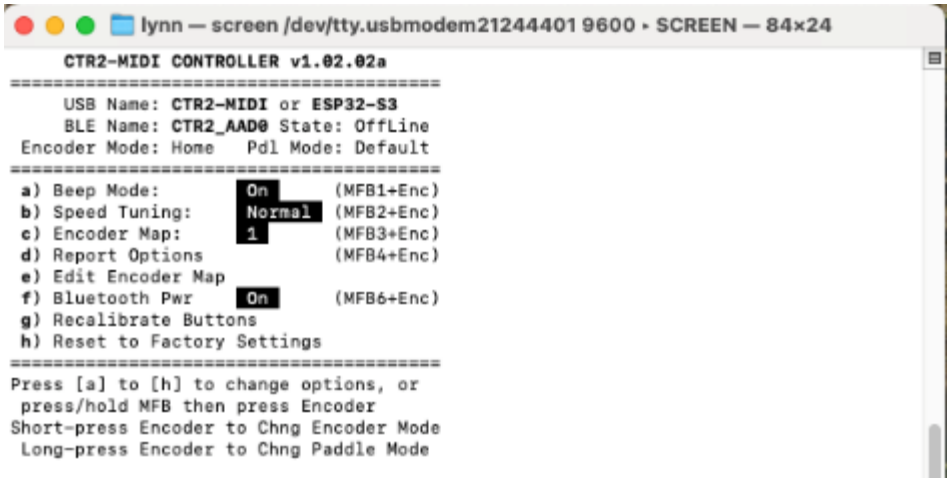


2. Open the **screen** utility using the **usbmodem3144401** device. Your USB device will have a different #.



3. The Apple and Linux terminals don't offer a lot of customization, and they don't support XModem transfers for exporting and importing setting file backups.

   This is the ***MIDI**'s* terminal display on a Mac.



Appendix G: Using CTR2-MIDI with Non-MIDI Programs

MIDI is not limited to programs that support MIDI controls. Third-party MIDI translator apps such as CoyoteMIDI can receive MIDI commands from the ***MIDI*** and convert them to other actions such as mouse scroll or keystroke actions. MIDI translator programs are available for Windows, Mac, and Linux.

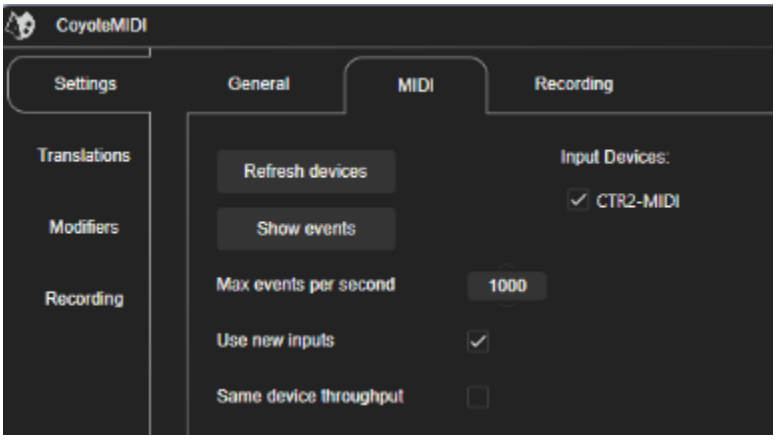
These apps allow the ***MIDI*** to be used as a physical tuning knob with radio control programs such a SmartSDR for Windows, wfView, and WebSDR that don't normally support external tuning knobs. And it doesn't stop there. These apps also allow the ***MIDI*** to be used with any program that accepts mouse scrolls or keystroke for control. For instance, it makes a great jog wheel for Audacity.

There are a couple of drawbacks to using a translation app instead of native MIDI control. First, like your mouse or keyboard, the app you want to control must be "in focus" to receive the mouse scrolls or keystrokes from the translation app. Second, most programs don't have a lot of control functions tied to

mouse scroll or keystroke actions. For instance, SmartSDR for Windows only accepts mouse scroll actions for frequency control and wfView only supports the J and K keys for frequency control.

Configuring CoyoteMIDI

This section will cover configuring CoyoteMIDI to interface with SmartSDR for Windows and wfView. The translator you choose will have a different setup procedure but they're all basically the same.



Before you start, plug the ***MIDI*** into a USB port on your computer. It will automatically register as a MIDI device. Make sure you are in the **Home** knob mode (all yellow LEDs are OFF. It any of them are on, press-and-release the knob switch until they are all off. This put the knob in its **Home** knob mode.

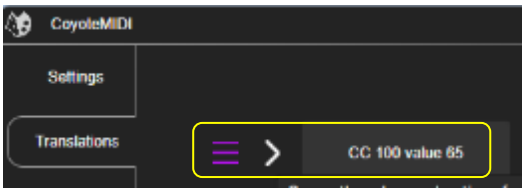
Once you've installed CoyoteMIDI open the app and navigate to the **Settings -> MIDI** tab. **CTR2-MIDI** should show up on the **Input Devices:** list. Click the checkbox to enable it.

Configure SmartSDR for Windows Frequency Control

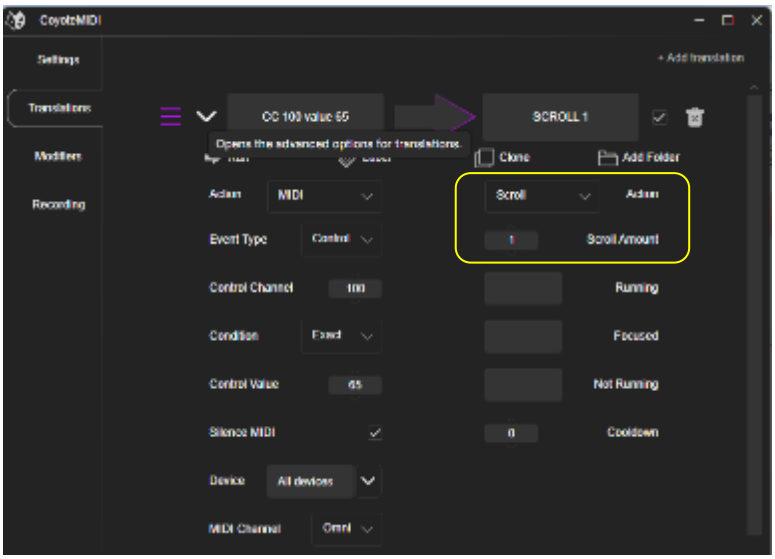
Next, select the **Translations** tab and click the **+ Add Translation** button in the top-right corner.

The first example uses the **Home** knob mode on the ***MIDI*** (MIDI control #100). This mode uses the **WheelA** MIDI control where a value of 65 represents a clockwise turn and 63 represents a counter-clockwise turn of the knob.

Click the box with **Click to record** then turn the knob on the ***MIDI*** *slowly* clockwise. You should see **CC 100 value 65** appear in the left box. If you don't see this, make sure all yellow LEDs are off. Hover the mouse over this entry to show the violet menu and **>** icons.



Click the **>** icon to view the details of this command. I've already set this command to be translated to the **SCROLL 1** mouse command.



The options on the left should be as shown. If not, edit them to match what's shown here.

To configure the **SCROLL** 1 output, click the **Action** dropdown box and select **Scroll**. Next, set the **Scroll Amount** to **1** so each tick of the clockwise turn of the knob produces on scroll tick.

That's it. You configured your first translation. Click the **+Add Translation** button again and follow the same process to program the counter-clockwise knob turn. In this case, set the **Scroll Amount** to **-1**. This will send the reverse mouse scroll.

Now open SmartSDR for Windows and the frequency should change as you change the knob on the ***MIDI***.

Configure wfView Frequency Control

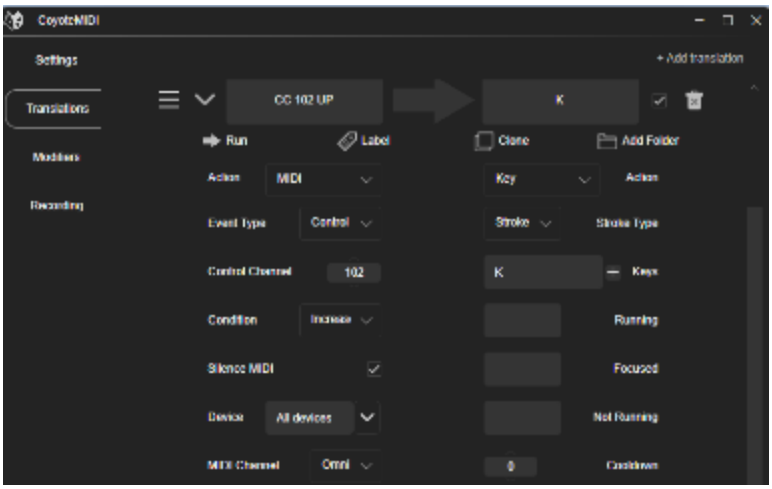
You can set up translations for many other programs the same way. Next, we'll go over setting up keystroke "J" and "K" to control the frequency in wfView.

Press-and-release the knob on the ***MIDI*** to switch to **Knob Mode 1**. The bottom yellow LED should be ON and the top yellow LED should be OFF. In this mode, the knob sends MIDI control #102.

NOTE: This knob uses the **SliderA** control type. This type operates like a potentiometer and has a range of 0 to 127 counts. To use this control type, we'll look for the value increasing or decreasing to determine the direction the knob is turning. You can edit the MIDI control type in a terminal if you want to change it to **WheelA**. If you do this, follow the setting in the first example.

Click the **+Add Translation** button to add another translation.

Click the box that says **Click to record** then turn the knob on the ***MIDI*** clockwise. You should see **CC 102 value xx**, where xx is a number between 1 and 127.



Hover the mouse next to this box then click the **>** icon to show the details.

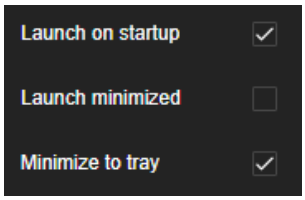
On the left column, click the **Condition** dropdown box and select **Increase**. This action will trigger when the MIDI value is increasing. The title in the top box will change to **CC 102 UP**.

In the **Action** dropdown box in the right column select **Key** and in the **Stroke Type** select **Stroke**. Finally, in the **+ Keys** box, enter **K**. This is the key used to increment the frequency in wfView.

Next, click the **+ Add Translation** button again, click the new box showing **Click to record** and turn the ***MIDI****'s* knob counter-clockwise. Now, follow the same process as outlined above except select **Decrease** in the **Condition** box and enter the letter **J** in the **+ Keys** box. wfView uses the **J** key to decrement the frequency. I'm not aware of any other keystrokes wfView supports. If there are more they can easily be added.

CoyoteMIDI Startup Options

There are a couple of options in the **Settings -> General** menu that you might want to set if you regularly use the ***MIDI*** and CoyoteMIDI with your apps. If you select **Launch on startup** and **Minimize to tray** options CoyoteMIDI will automatically start when you boot your computer and will be available in the Windows tray if you need to change anything.



Appendix H: Knob and Normal Button Assignment Worksheet

This worksheet can be used to document the knob and button assignments when CTR2-MIDI is in *Normal* button mode. An Excel version of this worksheet is available here..

CTR2-MIDI Normal-Button Worksheet

App: _____

Use this sheet to document your radio control App's MIDI mapping
Knob Modes: Upper # is Turn, Lower # is Push & Turn
Button Modes: Upper # is Short-press, Lower # is Long-press

Excel files can be fouind here
Firmware v2.00.00

All Knob Modes



Button Assignments	
MIDI #	Function
1	
11	
2	
12	
3	
13	

Button Assignments	
Function	MIDI #
	6
	16
	5
	15
	4
	14

Knob Assignments			
Mode	Function	MIDI #	Ctrl Type*
Home		100	
P&T		101	
1		102	
P&T		103	
2		104	
P&T		105	
3		106	
P&T		107	

NOTE: Change App MIDI mapping when changing between Normal and Extended Button modes.

Control Types	Descripton
SliderA	Potentiometer type
SliderB	Centerr Beep
WheelA	VFO - SmartSDR
WheelB	VFO-Thetis
WheelB-R	VFO-PiSDR

* Set Control Type in MIDI's Map

Paddle Input Assignments	
Normal Paddle Mode	
MIDI #	Function
20	Left Paddle
21	Right Paddle
Extended Paddle Mode	
30	Staight Key
31	MOX

Appendix I: Knob and Extended Button Assignment Worksheets

Use this worksheet to record the *Extended* button assignments you have mapped in your radio control app. An Excel version of this worksheet is available here.

CTR2-MIDI Extended Button Worksheet

App: _____

Use this sheet to document your radio control App's MIDI mapping
Knob Modes: Upper # is Turn, Lower # is Push & Turn
Button Modes: Upper # is Short-press, Lower # is Long-press

Excel files can be found here

Firmware v2.00.00

Knob Home

Knob Assignments	
MIDI #	Function
100	
101	

Button Assignments	
MIDI #	Function
1	
25	
2	
26	
3	
27	

Button Assignments	
Function	MIDI #
	6
	30
	5
	29
	4
	28

Knob #2

Knob Assignments	
MIDI #	Function
104	
105	

Button Assignments	
MIDI #	Function
13	
37	
14	
38	
15	
39	

Button Assignments	
Function	MIDI #
	18
	42
	17
	41
	16
	40

Knob #1

Knob Assignments	
MIDI #	Function
102	
103	

Button Assignments	
MIDI #	Function
7	
31	
8	
32	
9	
33	

Button Assignments	
Function	MIDI #
	12
	36
	11
	35
	10
	34

Knob #3

Knob Assignments	
MIDI #	Function
106	
107	

Button Assignments	
MIDI #	Function
19	
43	
20	
44	
21	
45	

Button Assignments	
Function	MIDI #
	24
	48
	23
	47
	22
	46

Paddle Input Assignments	
MIDI #	Function
96	
97	
98	
99	

Knob Control Types	Notes
SliderA	Common for most
SliderB	RIT, XIT
WheelA	VFO in SmartSDR
WheelB	VFO in Thetis
WheelB-R	VFO in FlSDR

Set Control Type in MIDI's Map

SmartSDR for iOS/MacOS Worksheet Example

This spreadsheet shows an example of the mapping you can use in SmartSDR for iOS/MacOS.

CTR2-MIDI Extended Button Worksheet

SmartSDR for iOS/MacOS

Use this sheet to document your radio control App's MIDI mapping
Knob Modes: Upper # is Turn, Lower # is Push & Turn
Button Modes: Upper # is Short-press, Lower # is Long-press

Excel files can be found here

Firmware v2.00.00

Knob Home

Knob Assignments	
MIDI #	Function
100	Slice Frequency **
101	Slice Volume

Button Assignments	
MIDI #	Function
1	Mode Next
25	Mode Prev
2	Band Next
26	Band Prev
3	Main Audio Mute
27	DAX Toggle

Button Assignments	
Function	MIDI #
Tune Step	6
Zoom In	5
Zoom Out	29
NR	4
NB	28

Knob #2

Knob Assignments	
MIDI #	Function
104	NR Level
105	NB Level

Button Assignments	
MIDI #	Function
13	80 Meters
37	160 Meters
14	40 Meters
38	60 Meters
15	20 Meters
39	30 Meters

Function	MIDI #
6 Meters	18
10 Meters	17
12 Meters	41
15 Meters	16
17 Meters	40

Knob #1

Knob Assignments	
MIDI #	Function
102	AGC-T
103	Zoom Slice

Button Assignments	
MIDI #	Function
7	Mode CW
31	DigL
8	Mode LSB
32	DigH
9	Mode USB
33	SAM

Button Assignments	
Function	MIDI #
Mode RTTY	12
Open DX Cluster Tool	36
Mode FM	11
Open Logbook Tool	35
Mode AM	10
Open FT8 Tool	34

Knob #3

Knob Assignments	
MIDI #	Function
106	CW Speed
107	RIT Freq *
19	Macro 1
43	Macro 5
20	Macro 2
44	Macro 6
21	Macro 3
45	Macro 7

Function	MIDI #
Log QSO	24
Slice Set Tx	48
RIT	23
XIT	47
Macro 4	22
Macro Stop	46

Paddle Input Assignments	
MIDI #	Function
96	Trigger Left Paddle
97	Trigger Right Paddle
98	Trigger Straight Key
99	MOX

Knob Control Types	
SliderA	*
SliderB	**
WheelA	***
WheelB	****
WheelB-R	Set Control Type in MIDI's Map

Notes
Common for most
RIT, XIT
VFO in SmartSDR
VFO in Thetis
VFO in PiSDR

Thetis Worksheet Example

This spreadsheet shows an example of the mapping you can use in Thetis.

CTR2-MIDI Extended Button Worksheet

Thetis

Excel files can be found here

Firmware v2.00.00

Use this sheet to document your radio control App's MIDI mapping
Knob Modes: Upper # is Turn, Lower # is Push & Turn
Button Modes: Upper # is Short-press, Lower # is Long-press

Knob Home

Knob Assignments	
MIDI #	Function
100	VFO A ...
101	Volume A

Button Assignments	
MIDI #	Function
1	Rx1 Mode Next
25	Rx1 Mode Prev
2	Band Up
26	Band Down
3	Mute On/Off
27	RX2 Mute On/Off

Button Assignments	
Function	MIDI #
Tune Step Up	6
Tune Step Down	30
Zoom Inc	5
Zoom Dec	29
Tuner On/Off	4
Start On/Off	28

Knob #2

Knob Assignments	
MIDI #	Function
104	RIT
105	XIT

Button Assignments	
MIDI #	Function
13	80 m
37	160 m
14	40 m
38	60 m
15	20 m
39	30 m

Function	MIDI #
6 m	18
VFO Swap	42
10 m	17
12 m	41
15 m	16
17 m	40

Knob #1

Knob Assignments	
MIDI #	Function
102	VFO B ...
103	Volume B

Button Assignments	
MIDI #	Function
7	CW
31	DigL
8	LSB
32	DigU
9	USB
33	SPEC

Button Assignments	
Function	MIDI #
RX EQ On/Off	12
TX EQ On/Off	36
FM	11
	35
AM	10
SAM	34

Knob #3

Knob Assignments	
MIDI #	Function
106	Drive Level
107	Panadapter Zoom

Button Assignments	
MIDI #	Function
19	Macro 1
43	Macro 5
20	Macro 2
44	Macro 6
21	Macro 3
45	Macro 7

Function	MIDI #
NR1 On/Off	24
NR2 On/Off	48
Filter Narrower	23
Filter Wider	47
Macro 4	22
Macro 8	46

Paddle Input Assignments	
MIDI #	Function
96	Trigger Left Paddle
97	Trigger Right Paddle
98	Trigger Straight Key
99	MIDI Straight Key

Knob Control Types	
SliderA	.
SliderB	..
WheelA	...
WheelB	....
WheelB-R	.....

Notes
Common for most
RIT, XIT
VFO in SmartSDR
VFO in Thetis
VFO in PiSDR

Set Control Type in MIDI's Map

Appendix J: Change Log

v2.00.02a: January 23, 2026

- Added a section for **RemoteTx** app setup and support
- Fixed several links to the CTR2-MIDI Flex manual.
- Added link to the **CTR2-MIDI** configuration files

v2.00.02: December 26, 2025

- Additional debouncing refinements to minimize random code elements on paddles
- Enable red Tx LED when keying in MIDI mode, even when Bluetooth is not connected
- Additional refinements to **Flex** mode – see the CTR2-MIDI Flex manual for that firmware

v2.00.01: November 10, 2025

- Added paddle release debounce to fix issues with cheaper paddles adding extra code elements
- Improved keyer Bug mode so it works properly now
- The **CTR2-MIDI firmware is now distributed as a single BIN file** named **CTR2-MIDI.bin**. The instructions for flashing the single BIN file have been updated in Appendix C.

v2.00.00: November 3, 2025

- Added a new option to run **CTR2-MIDI in Flex mode**, similar to the dual-boot mode in **CTR2-Flex/CTR2-Dial** firmware
    - **Flex** mode was ported from CTR2-Flex firmware. It allows you to connect **CTR2-MIDI** directly to your Flex radio over your local WiFi network. This mode supports all versions of SmartSDR, including Windows versions. Radio control is done directly with the radio using the Flex API, not MIDI, so no 3rd party app is required for this mode. **Flex** mode does not support SmartLink.
    - A separate operation manual for **CTR2-MIDI Flex** mode can be downloaded here.
- Added **Extended BTN** mode – in this mode, each knob mode has 12 button functions, for a total of 48 buttons.
- Added a new Maintenance Menu that adds the ability to Export and Import settings to a file on your PC for backup.
- Added Appendix H: Knob and Normal Button Assignment Worksheets for normal BTN mode.
- Added Appendix I: Knob and Extended Button Assignment Worksheets for **Extended BTN** mode.
- Added a new Keywords and Definitions section
- When toggling the Bluetooth radio on/off CTR2-MIDI automatically reboots. No need to power the unit down to reboot it.
- Replaced all references to the term “encoder” in this manual and in the firmware with the term “knob”.
- Replaced all references to the term “MFB” (i.e. Multi-Function Button) in this manual and in the firmware with the term “BTN”.

v1.02.01b – May 30, 2025

- Added Appendix F with information on using the **screen** command in Mac and Linux for a terminal interface
- Added a new link to download Tera Term. Get it from the CTR2 Group IO File folder here.

v1.02.01a – May 23, 2025

- Added a note about using a USB-C data cable for USB MIDI

v1.02.01 – May 9, 2025

- Added Appendix G: Using CTR2-MIDI with Non-MIDI Programs with examples of how to use CTR2-MIDI with programs like SmartSDR for Windows that do not support MIDI control

v1.02.01 – February 2, 2025

- Added **WheelB-r** to reverse dial direction for PI HPSDR app
- Added **Speed Tuning** to **WheelB** and **WheelB-r**
- Smoothed transition to speed tuning
- Changed Morse when changing Bluetooth On/Off from BLE0/1 to BT0/1
- Added additional information about connecting CTR2-MIDI to your Mac using Bluetooth
- Fixed incorrect instructions for selecting the files in EspressIF Flash Download Tool.
- Added information on mapping ***MIDI*** to the SparkSDR app
- Added a screenshot of the Mac Bluetooth permission screen
- Added additional information about Enabling Permissions in a Mac to use Bluetooth
- Updated the description of wiring the straight key and PTT switch to the MIDI's **Paddle In** jack
- Added a troubleshooting note about Flex Radio Zombie mode (no power out in CW mode when remote keying)

v1.02.00 – September 30, 2024

This update adds many new functions to the ***MIDI***:

- Three beep modes – *Off, Normal*, and *Long-Press*. Press-and-hold **BTN1** then press-and-release the knob to step through the beep modes.
- The ability to enable or disable speed (proportional) turning on **WheelA** control. Press-and-hold **BTN2** then press-and-release the knob to step through the speed modes, *Off, Normal,* and *Fast*.
- Support for two MIDI knob maps. Use one for iOS/Mac operations and one for Windows operations. Press-and-hold **BTN3** then press-and-release the knob to switch the maps.
- The ability to turn the Bluetooth radio off for battery operation. Press-and-hold **BTN6** then press-and-release the knob to turn the radio on or off.
- A Reset to Factory Settings option when all else fails. You must connect a terminal to the ***MIDI*** to reset the settings to factory.

v1.01.00f: September 3, 2024

- Added more information to Connecting CTR2-MIDI to the App section for connecting to Marcus' apps

v1.01.00e: August 20, 2024

- Updated the Connecting CTR2-MIDI to the App section to include new information on connecting to Marcus' apps
- Moved Configuring the MIDI Firmware to Appendix A.

v1.01.00d: May 22, 2024

- Updated the Installing using Linux or Mac section with information about switching COM ports for flashing new firmware after v1.01.00.

v1.01.00b and v1.01.00c: May 10 and 12, 2024

- Rewrote parts in the manual to clarify the configuration requirements of the ***MIDI*** and the app.

v1.01.00a: May 8, 2024

- Updated the Appendix B: Installing and Updating CTR2-MIDI Firmware section to include new information about flashing firmware to the ESP32-S3 in CTR2-MIDI.

v1.01.00: April 29, 2024 – Major Update Release

- Added USB MIDI support
    - Once v1.01.00 firmware has been installed in CTR2-MIDI you'll need to follow **Step 1** in Appendix B: Installing or Updating CTR2-MIDI Firmware section for future updates.
- Select from MIDI **Button**, and two types of **Sliders** and **Wheels** for each knob function in the terminal page
- Redesigned terminal page
    - Hotkeys select the MIDI function for each knob function
    - Displays saved ADC count assigned to each BTN
    - Displays ADC count when an BTN is pressed
    - Displays MIDI channel, control, and value when a control is executed
- Added Appendix D and E with information on using Tera Term and Putty

v1.00.05 – April 7, 2024

- Added a reference to the **Conji** app for MacOS to help with connecting the ***MIDI*** to a Mac computer.

v1.00.05 – March 26, 2024

- Revised button read code to reduce false triggers
- Added Show/Hide BTN Counts option in terminal set up
- Added *No Sidetone* solution to Troubleshooting table

v1.00.04 – March 23, 2024

- Green LED now flashes to indicate unit is powered up (CTR2-MIDI hardware only)

v1.00.03 – March 21, 2024

- Added additional information on importing the .map file
- Added additional information on connecting CTR2-MIDI to the app
- Added additional troubleshooting help

v1.00.03 – March 10, 2024

- Add beep types to tables
- Save knob values to initialization file

v1.00.02 – March 8, 2024

- Added Troubleshooting section

v1.00.02 – March 7, 2024

- Shortened the ***MIDI's*** device name from CTR2-MIDI_xxxx to CTR2_xxxx
- Removed **Paddle PTT** mode and replaced it with **Paddle Mode**
    - When the green LED is off the paddle inputs control MIDI Buttons 20 and 21
    - When the green LED is on the paddle inputs control MIDI Buttons 30 and 31
- Removed PTT latching logic that as associated with **Paddle PTT** mode
- The red LED now lights when either paddle is pressed
- Redesigned terminal screen

v1.00.01 – March 6, 2024

- Added an option in the terminal display to enable/disable **Paddle PTT** mode
- Added BTN ADC values to the terminal display

v1.00.00m – March 3, 2024

- Beta release for interface development

v1.00.00 – December 22, 2023

- Initial release of CTR2-MIDI firmware for CTR2-Micro