CTR2-MIDI

Operation Manual

v1.00.03

Last Revision: March 21, 2024

Updated to firmware v1.00.03

Revised sections for this version are highlighted in yellow

Contents

Introduction

CTR2-MIDI is a custom MIDI controller designed for ham operators using the iPad and Mac radio control apps developed by Marcus Roskosch, DL8MRE. It currently supports MIDI over Bluetooth LE (BLE). MIDI over USB may be available in the future.

MIDI commands can be used to change frequency, mode, volume, CW speed, control the keyer and PTT, and many other functions in the app.

MIDI outputs are sent directly to the radio control app using BLE. No network or physical connections are required and CTR2-MIDI can be powered from the iOS device's accessory jack, a 5 volt cell phone charger, or a USB battery pack making CTR2-MIDI the idea companion for portable operation.

This photo shows the *MIDI* being powered by an iPhone though a Lightning/USB adapter. A charger or USB battery can be connected to the adapter to power both devices.

The **MIDI** includes the following features:

1. Four encoder modes that provides control of 8 parameters. Each mode controls two parameters: 1) normal encoder action, and 2) press and turn encoder action. Short-press the encoder to change its mode. The current mode is indicated by two yellow LEDs in the upper left corner of the unit.
2. Connect your paddles to the 3.5mm (1/8") stereo jack and use your paddles to control the keyer in the app. The paddle input can also be used to control PTT when in voice modes.
3. Six dual-mode pushbuttons. Each button can be assigned a short and long press function.
4. No configuration required. Just plug it into a USB power source and map the controls in the app.
5. Easy firmware updating through the USB-C port.

IMPORTANT NOTICE: CTR2-MIDI *is not Marcus's product and he offers no support for it*. He was kind enough to include MIDI BLE control in his programs. Please do not waste his valuable time with troubleshooting questions or asking for additional features. If you have questions, contact me. My email is good on QRZ.com.

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 the assembled unit 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.
- 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.

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.

Feel free to contact me if you have question about a certain feature or have ideas for future improvements. I love to get feedback on my work. My email addre1ss is good on QRZ.com.

Change Log

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 encoder values to initialization file

v1.00.02 – March 8, 2024

- Added Troubleshooting section

v1.00.02 – March 7, 2024

- Shortened the MIDI 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 MFB 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

Changes to previous versions can be found in Appendix C.

What is CTR2-MIDI?

Background

CTR2-MIDI (*MIDI*) was derived from CTR2-Micro (*Micro*), a full featured radio controller and memory keyer in a very small form factor. The *Micro* can control a wide range of radios but it has a limited user interface and generally requires a computer, tablet, or cell phone to display its settings, although it can be used with CTR2-Voice to provide an auditable interface for blind and sight-limited operators. The *Micro* has some limitations when used to control a Flex radio remotely because it doesn't support SmartLink and requires port 4992 on the Flex radio to be forwarded through the station's router, creating a security risk if not managed properly.

Since the *Micro*'s introduction many users asked if it was possible to connect the *Micro* directly to, and control, another radio control program such as Marcus Roskosch's SmartSDR or SDR-Control for iOS apps. Being able to do this would solve the security issues with the Flex radio and would provide a better Flex user experience because of the more robust user interface.

A New Product is Born

After discussions with Marcus on the best way to interface to his programs we settled on using the BLE MIDI interface. BLE stands for Bluetooth Low Energy and MIDI is a control protocol generally used to control musical instruments. It's simple and fast and can be used for other control purposes. Many radio control programs like SmartSDR for iOS have MIDI support built-in as an interface for remote control and many hams use off-the-shelf USB MIDI controllers to control their radios through these programs. But these controllers can be expensive and are not designed to accept paddle input to control the app's keyer.

The *Micro* seemed to be a natural fit. Unfortunately the ESP32-C3 controller used in the *Micro* doesn't support MIDI on its USB connection to the computer (that would be too easy!). It does support MIDI over Bluetooth LE (BLE), so that is good. However, when I added the BLE library to an already loaded, relatively slow (160 MHz) single-core processor it was more than the *Micro* could handle.

To get BLE MIDI running reliably on the ESP32-C3 processor I had to strip out almost all of the other hardware features built into the *Micro* like WiFi and high-precision ISR based timers for the keyer. The result is a single purpose program called **CTR2-MIDI**. This firmware is available for current CTR2-Micro owners who wish to use the *Micro* as a MIDI controller. See Appendix A for information on using CTR2-MIDI firmware on CTR2-Micro.

While CTR2-MIDI firmware runs on the *Micro's* hardware it is somewhat limited because the *Micro* only has three buttons and one LED. I have designed new hardware for the CTR2-MIDI firmware, and of course I call it CTR2-MIDI. The new hardware provides six dual-function buttons and has additional LEDs to show the encoder mode, paddle mode, and paddle status. An ESP32-S3 processor was chosen for the CTR2-MIDI hardware. This processor is dual-core and runs faster than the –C3 variant (240 MHz) but most importantly, it has USB hardware that should allow CTR2-MIDI to eventually support USB MIDI. This is important because Window's doesn't handle Bluetooth LE MIDI devices properly. Having USB MIDI would allow the CTR2-MIDI to be used with Windows based programs that support MIDI inputs.

Hardware

CTR2-MIDI is designed to use the same enclosure that is used with the *Micro*. This enclosure is the perfect size for a busy operating desk or for portable operation.

As shown in the photo, the encoder knob is the predominant feature on the face of the unit. When oriented with the USB-C connector and paddle jack on the edge away from you, the two LEDs in the upper left corner show the encoder status 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 encoder is in its Home mode. Short-pressing the encoder steps you through mode 1, 2, and 3 and the LEDs indicate the current mode. In this photo the encoder is in mode 1.

The multi-function buttons (MFBs) are laid out around the circumference of the knob. MFB1 is on the upper left, just below the encoder mode LEDs. The buttons are numbered counter-clockwise around the knob ending with MFB6 on the upper right.

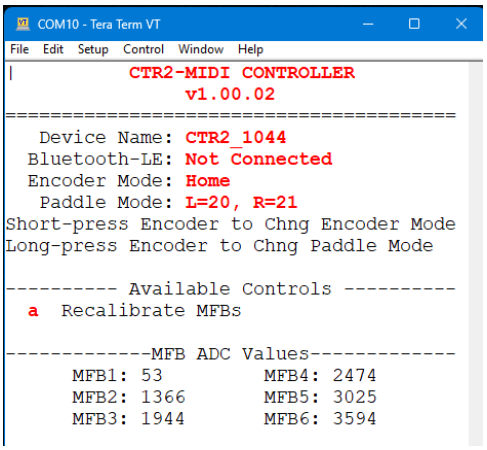
> **NOTE:** The button layout is different from the buttons on the *Micro* where MFB1 is the bottom-left button.

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

Buzzer Volume Control Option

CTR2-MIDI does not come with a volume control for the internal buzzer. Provision has been made on the PCB board to install one if you desire. Just cut JP1 and add RV1, a 50 kOhm potentiometer to the board. You'll also need to drill a 1/8" hole in enclosure for access.

Configuring the MIDI Firmware for First Use



All assembled *MID'sI* come pre-programmed and tested and do not require user configuration.

If you built your *MIDI* from a kit you will need to connect it to a computer running a terminal program such as Tera Term or Putty to register the ADC values for the MFBs. You can find information in **CTR2-Micro Operation Manual** **Appendix D** and **E** on configuring and connecting to a terminal program. Once you are connected press any key to open the display.

There is only one option in the terminal display. Pressing [**a**] opens the MFB Calibration display. You will be prompted to press each button in sequence, starting with MFB1 (the upper most button on the left side of the encoder knob) and progressing counter-clockwise about the encoder knob. This allows the unit to record the voltage each button presents to the ADC (A/D converter). These voltages are specific to your unit and are displayed at the bottom of page.

> **NOTE:** Each button must be pressed within 10 seconds or the process will fail.

> **NOTE:** If the *MIDI* loses the MFB calibration data it will sound "?" in Morse code when you press any MFB or press the encoder. To recalibrate the button voltages connect a terminal program to the *MIDI's* virtual USB serial port then press [**b**] to start the calibration process.

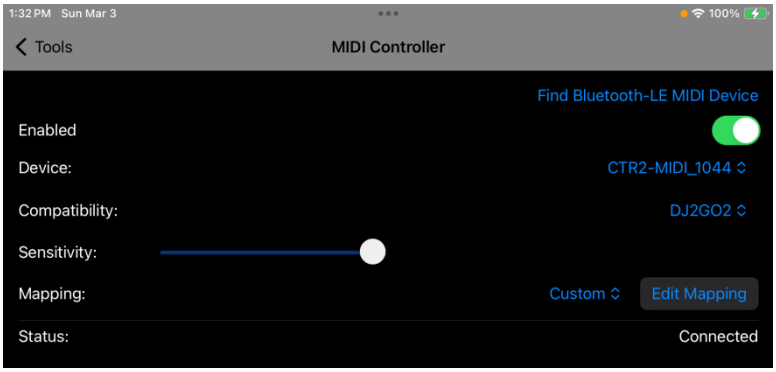
Connecting the App to CTR2-MIDI

Each CTR2-MIDI has a unique device name. They all start with **CTR2-MIDI_** followed by the last four digits of that units' MAC address. To connect CTR2-MIDI to your app simply enable Bluetooth on your iPad or iPhone and open the **MIDI Controller** option under **Tools** in the app.

> **NOTE:** Future versions of the apps will have a dedicated **CTR2-MIDI** option in **Tools**.

In the **MIDI Controller** screen select **CTR2_xxxx** in the **Device** list then slide the **Enabled** switch to the right.

If **CTR2_xxxx** does not appear on the **Device** list, or it doesn't



connect when turning the **Enabled** switch On press the **Find Bluetooth-LE MIDI Device** button at the top of the screen. You may need to press this button several times with the **Enabled** switch in either On or Off position for **CTR2_xxxx** to appear in the **Device** list.

If you can't get the app to connect to *MIDI*, download **MIDI-Wrench** from the App store. It's a free app used to troubleshoot BLE MIDI issues. Connect to the *MIDI* in this app then try to connect again in the radio control app.

> **NOTE:** BLE MIDI devices do not show up in the **Bluetooth** section in the device's **Settings** panel because BLE MIDI is not a standard interface device like a head set or keyboard.

Next, select **DJ2GO2** in the **Compatibility** dropdown and select **Custom** in the **Mapping** dropdown.

Once **Connected** appears on the **Status** line, press the **Edit Mapping** button to enter mapping mode.

CTR2-MIDI Functions

The MFBs and encoder on CTR2-MIDI have predefined functions (i.e. MIDI commands). To use CTR2-MIDI with the app you must tell it what you want it to do with each command from CTR2-MIDI. This is done through mapping.

> **NOTE:** CTR2-MIDI always sends commands on MIDI Channel 1.

Multi-Function Buttons (MFBs)

MFBs are laid out counter-clockwise around the encoder knob with MFB1 on the upper left and MFB6 on the upper right of the knob.

MFB1 through MFB6 generate two MIDI **NoteOn** commands when released. The first has Velocity set to 127 (i.e. On) and the second, sent a few milliseconds later, has Velocity set to 0 (i.e. Off). This is interpreted by the host program as a button press and release sequence.

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

Short-pressing (i.e. < 1 second) a button gives one audible beep on press and generates a **NoteOn 1** to **NoteOn 6** press/release sequence (depending on the button) when released.

Long-pressing (holding a button for longer than 1 second) gives one audible beep on press and a second beep after the long-press timeout. Upon button release a **NoteOn 11** to **NoteOn 16** press/release sequence is sent.

> **NOTE**: Holding a button longer than 4 seconds cancels the button press and sounds a third beep.

Buttons can be assigned to toggle commands in the app's map such as **Band Up**, **Band Down**, **Mode Up**, **Mode Down**, etc.

Encoder Mode

The encoder on CTR2-MIDI can control up to 8 functions. These are divided into four groups, or *modes* numbered 0 to 3, with 0 being the Home mode. The yellow LEDs in the upper left corner of the controller indicate which encoder mode is selected when the encoder mode is not in the Home mode. The lower LED has a value of 1 and the upper LED has a value of 2. No yellow LEDs are lit when the encoder is in the Home mode.

> **NOTE:** Turning the encoder in the Home encoder mode sends **NoteOn** press/release commands that should be mapped to **Frequency Up** or **Frequency Down** button commands. All other encoder modes send **Control Change** commands with a value indicating the setting change. These controls can be assigned to variable controls in the app such as **Volume, NB Level**, **NR Level**, 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 value it last sent, not the value you set in the app.

Encoder modes are shown in the table below.

Encoder Mode	LED	Beep Tone	Description
Home	○ ○	Single high frequency beep	Encoder Left = **NoteOn 40** (Freq down) Encoder Right= **NoteOn 41** (freq up) Press & Turn = **Control Change 1**
1	○ ●	Single low frequency beep	Encoder Turn = **Control Change 2** Press & Turn = **Control Change 3**
2	● ○	Two low frequency beeps	Encoder Turn = **Control Change 4** Press & Turn = **Control Change 5**
3	● ●	Three low frequency beeps	Encoder Turn = **Control Change 6** Press & Turn = **Control Change 7**

All encoder modes except the Home mode sends one MIDI Control Change Command when the encoder is turned left or right and it sends another Change Control Command when the encoder is pressed and held down then turned left or right. CTR2-Micro users will recognize these as the same methods used for tuning the frequency of the radio and for changing the tuning digit.

Short-pressing the encoder changes the encoder mode. The yellow LEDs on CTR2-MIDI will change to indicate the encoder's mode and the unit will emit one to three short low frequency beeps to indicate modes 1 to 3 or a single high frequency beep to indicate the Home mode.

Long-pressing the encoder while in Mode 1, 2, or 3 will return you back to the Home encoder mode (all LEDs off) and emit a single high frequency beep.

Paddle Mode

The *MIDI* has two **Paddle Modes, Normal** and **Extended**. These modes allow you to map different functions to the paddle input jack on the *MIDI* 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	Description
Normal Mode	○	Two tone Low>High beep	Left paddle controls **NoteOn 20**, Right paddle controls **NoteOn 21** Map these to **Trigger CW Left Paddle** and **Trigger CW Right Paddle**
Extended Mode	●	Two-tone High>Low beep	Left paddle controls **NoteOn 30**, Right paddle controls **NoteOn 31** 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 map t30 and 31 the same as 20 and 21 to disable the extended mode

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

Beep Tones

Two frequencies are used to signal the unit's mode: Low beep = 650 Hz and High beep =750 Hz. The beep tones are sounded when the unit enters the new mode. The table below summarizes the beep tones that you will hear when you make a change.

Action	Beep Tone	Mode
Short-press encoder	1 High beep	Home Encoder mode – no yellow LEDs lit – tune frequency and press & turn to change **CC#1**
Short-press encoder	1 Low beep	Encoder mode #1 – yellow LED 1 lit - turn changes **CC#2** and press + turn changes **CC#3**
Short-press encoder	2 Low beeps	Encoder mode #2 – yellow LED 2 lit – turn changes **CC#4** and press + turn changes **CC#5**
Short-press encoder	3 Low beeps	Encoder mode #3 – yellow LED 1 and 2 lit - turn changes **CC#6** and press + turn changes **CC#7**
Long-press encoder when in Extended Encoder mode	1 High beep	Return to Home Encoder mode – no yellow LEDs lit
Long-press encoder when in Home Encoder mode	High beep followed by a Low beep	Enter Extended paddle input mode where paddles control MIDI **Buttons #30** and **#31**
Long-press encoder when in Extended Encoder mode	High beep followed by a Low beep	Enter Normal paddle input mode where paddles control MIDI **Buttons #20** and **#21**

Mapping in the App

Each of the above MIDI controls can be mapped in the app to control various features. For instance, the MFBs can be mapped to change modes or bands, or enable features such as antenna tuners or filters. The eight encoder controls can be mapped to change values such as volume, RIT, XIT, RF Power, etc.

Importing the CTR2-MIDI.map file

A copy of the map I use is included in the *MIDI's* firmware ZIP file. You can import this map into your iOS device by pressing the **File** icon on the top of the **Edit MIDI Control Map** panel in the app.

In order to do this, you must allow the app to use the iCloud drive on your device. Once this is done a folder for the app is created on the iCloud drive. Copy the *CTR2-MIDI.map* file from the zip file to this folder then click **Import Mapping from iCloud Drive**. You can also export your settings to this drive to save a back up copy.

> **NOTE:** There is a bug in the app. Once you have imported a map you must edit one of the control settings before the app will save the map settings otherwise the settings will be removed when you navigate away from the MIDI tool.

Editing the MIDI Controller Map

Pressing the **Edit Mapping** button in the **MIDI Controller** page opens the **Edit MIDI Controller Map** page.

The map will be empty when you open it for the first time. To register each MFB on the *MIDI,* short-press and long-press each one to add it to the list. Next, step through each encoder mode then turn the encoder and press and turn the encoder.

Short-press MFBs are mapped to **Buttons 1** to **6**. Long-presses are mapped to **Buttons 11** to **16**.

> **NOTE:** Turn the encoder left and right while in the Home encoder mode (both yellow LEDs off) to register **Buttons** 40 and 41. Assign these to the **Frequency Down** and **Frequency Up** functions in the app.

To map a function to each control just press the control line and choose a function from the dropdown menu.

The screenshot at the right shows the mapping I'm currently using on my unit. I map similar items to the same button.

For instance: Short-pressing MFB1 controls **Button 1** which tells SmartSDR to increment the mode on the radio. Long-pressing MFB1 controls **Button 11** which tells SmartSDR to set the radio's mode back to CW.

Short-pressing MFB2 controls **Button 2** which increments the band. Long-pressing it (**Button 12)** decrements the band.

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

NOTE: After mapping the **Paddle Modes** for **Buttons 30 and 31** make sure you turn **Extended Paddle Mode** off (long-press the encoder) if you want to use the normal paddle mode for keying the keyer.

Using Paddles with the App's Keyer

The SmartSDR for iOS app includes a keyer that you can control with the paddles connected to CTR2-MIDI.

> **NOTE:** 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 Dit may be added. It may take some practice to get use to the timing change.

To control the keyer, follow these steps:

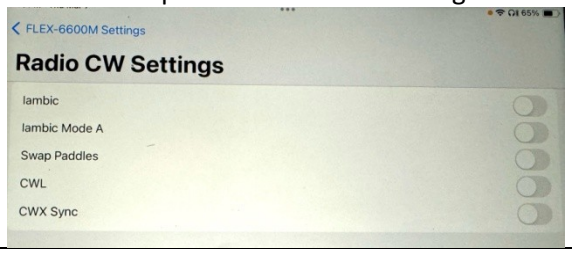
- Map **Button 20** to **Trigger CW Left Paddle** and **Button 21** to **Trigger CW Right Panel**. You can change this mapping if they are wired backwards.
- Plug your paddles into the **Paddle In** jack on CTR2-MIDI.
- Verify that the **Paddle Mode** is *in **Normal** mode* on the *MIDI* (the green LED is off). To toggle this mode, long-press the encoder while in the Home encoder mode (both yellow LEDs off). You should hear two tones (high to low frequency) and the green LED will turn off. *If **Paddle Mode** is in **Extended** mode the paddles will control MIDI **Buttons 30** and **31**.*
- Set the radio's mode to CW
- Press the **View** menu in SmartSDR's Panadapter display and select **CWX Panel**. You can adjust speed and other setting here.
- Pressing either paddle will now activate the keyer and key the radio.

Using a Straight Key or PTT Switch

If you want to use a straight key or an external PTT switch with the *MIDI*, map **Button 30** or **31** to **Trigger CW Straight Key** and **PTT Push** then wire the straight key and PTT switch to the Tip and Ring of a stereo plug and insert it into the **Paddle In** jack on the *MIDI.*

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

Troubleshooting

Issue	Solution
I can't connect to the CTR2-MIDI BLE device	Follow the instructions in the Connecting the App to CTR2-MIDI section. You may need to click both the **Find Bluetooth-LE MIDI Device** and the **Enabled** switch several times in different combinations to get the app to find the *MIDI.* You also may need to install the **MIDI-Wrench** app to get CTR2-MIDI recognized.
I have to connect CTR2-MIDI every time I start the app.	CTR2-MIDI will stay connected the app on your iOS or Mc device as long as it remains powered up. If you remove power from CTR2-MIDI you'll need to open the **Tools-MIDI Controller** option to reconnect it to the app.
Radio starts sending a string of Dits or Dahs when I select CW mode	This seems to be a bug in the app. 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.
Encoder doesn't tune radio's frequency	Make sure the encoder is set to its Home mode (all yellow LEDs off) and that MIDI Buttons 40 and 41 are mapped to Frequency Down and Frequency Up in the app.
Paddles don't key the radio	1. Verify the radio is in CW mode 2. Verify the **Paddle Mode** is in **Normal** mode (green LED is off) 3. Verify MIDI Buttons 20 and 21 are mapped to **Trigger CW Left Paddle** and **Trigger CW Right Paddle** 4. Open the CWX Panel in the app's View menu
Paddles are reversed	Remap MIDI Buttons 20 and 21 to swap Left and Right paddle assignments
Pushbutton operates the wrong function	1. Verify the button is mapped correctly 2. If the wrong MIDI Button is being selected open a terminal connection to the *MIDI* and recalibrate the button ADC values
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. To fix this problem, click the **Flex 6xxx** button on the bottom of the display, select the **CW** item to open the Radio CW Settings window and disable **CWX Sync**. 

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

As mentioned above, 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.

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

- 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 so you can't control the Flex radio with just the *Micro* or use 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 transmit message buffers you had set up in the *Micro*
    - Key and PTT Output is not supported
    - Favorite Frequency and Previous Frequency lists are not supported
    - The web browser interface is not supported
    - Reflash the *Micro* firmware on your *Micro* to restore its normal features
- The *Micro* only has three buttons so you can only map 6 **Buttons** in the app
- The *Micro* only has one LED. This LED flashes to indicate the current mode of the unit

Flashing the Firmware

To flash the *MIDI* firmware on to your *Micro,* follow the instructions in **Appendix B**.

If you built your *MIDI* from the kit, you will need to connect a terminal to your *MIDI* and go through the **MFB Calibration** sequence to save the MFB voltages in the *MIDI's* initialization file. This only needs to be done once.

> **NOTE:** The *MIDI* firmware maintains its own file structure so settings you had when running the *Micro* firmware are not lost. Just reflash the *Micro* firmware to your *Micro* to return it to normal.

Operating the *Micro* with *MIDI* Firmware

The *Micro* operates exactly like the *MIDI* when running the *MIDI* firmware. 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. You will have 6 **Button** functions (two on each MFB) and all of the encoder modes.

The big difference between the *MIDI* and the *Micro* is with the LED indicators (or lack thereof).

On the *Micro* the single LED flashes to indicate the mode of the controller. The table below summarizes these flash sequences and the beeps associated with them.

Encoder Mode	Flash Sequence	Buzzer Tone	Description
Home	One long flash	One high frequency beep	Turning encoder changes frequency. Press and turn encoder changes **Control Change 1 (CC1)**.
Mode 1	One short flash	One low frequency beep	Turning encoder changes **CC2**. Press and turn encoder changes **CC3**.
Mode 2	Two short flashes	Two low frequency beeps	Turning encoder changes **CC4**. Press and turn encoder changes **CC5**.
Mode 3	Three short flashes	Three low frequency beeps	Turning encoder changes **CC6**. Press and turn encoder changes **CC7**.

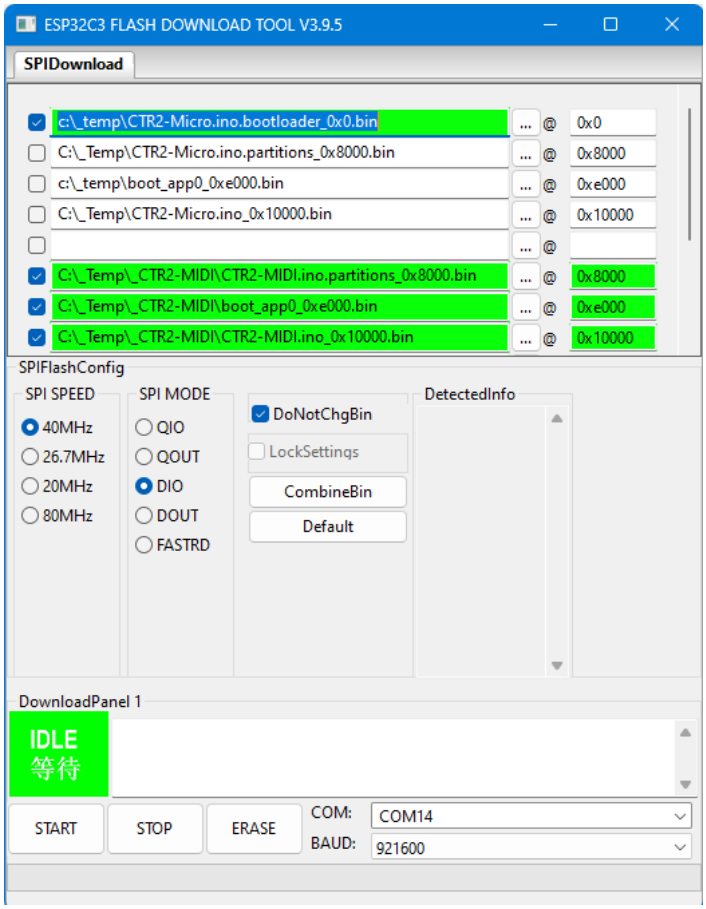
The **Paddle Mode** determines the rate of the LED flash.

- 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

Appendix B: Installing or Updating CTR2-MIDI

Kits and assembled CTR2-MIDIs have the firmware already installed on them but inevitably changes will be made to the program over time. To install the latest MIDI firmware into CTR2-Micro follow these steps:

1. Download and unzip the CTR2-MIDI firmware from **my web site.** Unzip it into a different folder than where you store the Micro's firmware update files.
2. Open the **EspressIF Flash Downloader Tool** as described in the **CTR2-Micro Operation Manual** under **Appendix B: Loading and Updating Firmware**. When it starts, select the **ESP32-S3 Chip Type** and USB **LoadMode**. (If installing on a CTR2-Micro, select the **ESP32-C3 c**hip type).
3. Map the four .BIN files in the CTR2-MIDI firmware distribution file into the downloader tool. **NOTE:** The address for each file is embedded in its file name. Enter this address in the address field to the right.
4. Select the checkboxes for the four files shown here.



5. Hold the encoder down and cycle the power on CTR2-MIDI. This puts CTR2-MIDI into program mode.
6. Set the COM: port to the port assigned to CTR2-MINI and set the Baud to 921600.
7. Click the **Start** button to start the download.

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

Installing using Linux or Mac

A script file is also 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**.

To use the script file you must change it to an executable file and edit it to work with your file system. Instructions to this are in the CTR2-Micro manual.

Appendix C: Change Log

Previous change log entries will be archived here.