Thor’s Arsenal

User Manual

Greenface Labs -- 2025

Table of Contents

[Description 5](#_Toc188725609)

[Features 6](#_Toc188725610)

[Installation 7](#_Toc188725611)

[Quick Start 9](#_Toc188725612)

[Details 11](#_Toc188725613)

[Open-Source Project 11](#_Toc188725614)

[General Operation 12](#_Toc188725615)

[Screen Saver 13](#_Toc188725616)

[Output Details 14](#_Toc188725617)

[Waveform Details 17](#_Toc188725618)

[Settings Details 18](#_Toc188725619)

[Bounce Function 19](#_Toc188725620)

[Remote Modes 21](#_Toc188725621)

[USB Direct Mode 21](#_Toc188725622)

[De-quarantining on the Mac 22](#_Toc188725623)

[WiFi (Bonkulator Only) 22](#_Toc188725624)

[Using The APP 24](#_Toc188725625)

[Output Select / Settings Buttons 25](#_Toc188725626)

[Waveform / Message Display 25](#_Toc188725627)

[Parameters 26](#_Toc188725628)

[Triggers / Group 27](#_Toc188725629)

[Slider Controls 29](#_Toc188725630)

[Macro Controls 30](#_Toc188725631)

[Draw Waveform Controls 32](#_Toc188725632)

[Terminal Mode 34](#_Toc188725633)

[Software Update 39](#_Toc188725634)

[Disaster Recovery 39](#_Toc188725635)

[Appendix 40](#_Toc188725636)

[Remote Commands That Don’t Take an Argument 40](#_Toc188725637)

[Remote Commands That Take an Argument 42](#_Toc188725638)

A close-up of a device

Description automatically generated

# Description

Thor’s Arsenal is a 14HP Eurorack module that generates signals that can be used to modulate voltage-controlled functions such as VCO, VCF, VCA, etc... Eight independent outputs can be triggered by any of four trigger inputs and can also be modulated by the two CV input signals.

It also has a special function canned *Bounce* that is a combination of DVM and Sample&Hold.

#### 8 Outputs

* In general, the output level follows the selected waveform.
* Waveforms can be from the factory list or user defined.
* Factory waveforms include ramp, sine, haystack and others.
* Waveforms can be scaled and offset by UI setting or by CV input.
* Waveform period can be set via UI or by CV input.
* Waveform repetitions can be set from 1 to 32,767.
* Waveform quantization can be enabled or disabled.
* Waveform can be randomized from 0-99%.
* Outputs can be programmed to be triggered or clocked by any of the trigger inputs.
* Outputs have LED level indicators

#### 4 Trigger/Clock Inputs

* Triggers activate Output and Bounce functions.
* When Output is set to External Clock mode, trigger inputs increment the waveform step.
* The trigger inputs respond to a positive-going signal or a button press.
* Each trigger can be independently enabled or disabled.
* An LED indicates when any attached output has been triggered and is active.

#### 2 CV Inputs

* Input Range: +/-10V (+/-5V for the Bonkulator).
* CV inputs can control many of the output parameters.
* LED level indicators.

## Features

* 128x64 two-color OLED graphic display
* Desktop and Terminal interfaces provide rich user experiences with the addition of computer resources like a keyboard, mouse, display and scripting. *The Bonkulator also has a WiFi interface that is similar to the GUI desktop app.*
* All settings are stored in FRAM memory so that they are retained when power is off
* Inputs and Outputs are protected against damage during typical use
* *Bonkulator output range is switchable: +/-5V or 0-10V (board models 4.0 and later)*
* *An Expansion module is available for the Bonkulator that provides an external USB interface with some interesting extra features. See “The BonkDaddy”.*

Thor’s Arsenal has a lighted Adjust Knob. It turns green when a function requires you to push the knob in order to execute the function. This is how you enter several Settings functions. The Adjust Knob may turn red. For instance, it does this to indicate recording status in User Waveforms. The knob may also turn orange. Currently this only happens during the boot sequence where the knob first turns red, then orange, then green and turns off when the boot sequence is complete.

# Installation

**Thor’s Arsenal** has an “either way” power connector. A 10-pin connector is typically used, but a 16-pin connector will work if the red line is lined up with either end of the mating connector.

A close-up of a circuit board

Description automatically generated

**For the Bonkulator**, care must be taken to connect the power to the right connector. The power connection is “either way”. Refer to the following image for the location of the power connector.

A close-up of a circuit board

Description automatically generated

# Quick Start

Thor’s Arsenal can be used in an endless variety of ways, but here is a great way to start exploring. These instructions are for use with a Moog Mother-32 but can be easily adapted to most modular synthesizers. The addition of an Echo module is highly recommended.

#### Mother-32 Set-up

* VCA Mode – ON (do not use the envelope generator)
* VCO Mod Amount – 50%
* VCO Mod Dest - Frequency
* VCF Mode – Low Pass
* VCF Mod Polarity – +
* LFO Wave – triangle
* LFO Rate – 30%
* Cutoff – 50%
* Resonance – 20%
* Sustain – Off
* Frequency – 50%
* Pulse Width – 50%
* VCO Wave - Pulse
* Octave – 4
* Note – 8

#### Thor’s Arsenal Set-Up

* Output 0 and Output 1 (common settings)
  + Waveform: Ramp
  + Init Delay: 00000
  + Active Time: 03010
  + Idle Time: 00000
  + Repeat: 00000
  + T0: Enabled
  + T1: Disabled
  + T2: Disabled
  + T3: Disabled
  + CV0: Off
  + CV1: Off
  + Range: +/-5V
  + Scale: see below
  + Offset: 000
  + Phase: 0
  + Randomness: 00
  + Quantize: No
  + Idle Value: 0.0
  + Clock: Internal
  + Trig Ctrl: Hold\_Off
  + Trig Holdoff: 5
* Output 0
  + Scale: 100
* Output 1
  + Scale: -100

Connect Thor’s Arsenal Output 0 to the M-32 VCO 1V/Oct input and the VCF Cutoff input. Connect Output 1 to the M-32 LFO Rate input. Press Trigger 0 on Thor’s Arsenal and you should hear a sound much like that found [here](https://www.greenfacelabs.com/posts/bonkulator-quick-start-example/). Experiment with the various Output parameters. Try increasing the Randomness on Output 0 to 20.

# Details

## Open-Source Project

Thor’s Arsenal is an Open-Source project that has an Arduino RP2040 at its core. The code is written in C++ and makes use of several public domain code libraries for managing the various hardware components such as the display and the FRAM non-volatile memory.

Because it is Open Source, in addition to being used by musicians and sound designers to create new and amazing sounds, Thor’s Arsenal can be used by students and entrepreneurs to create new, amazing modules. Links to the design repositories can be found on the Thor’s Arsenal web page.

## General Operation

The typical procedure for using Thor’s Arsenal involves choosing a set of waveforms that are needed for a particular application. Then a set of triggers are defined and the various parameters for each waveform are entered. Once configured, Thor’s Arsenal reacts to trigger signals, sending the waveform for each Output that was triggered.

* The Output is selected using the Output buttons.
  + Terminal uses the **‘f’** command. E.g. F3
  + The app has dedicated Function buttons.
  + Each Output has several parameters.
  + Waveforms are selected as one of the parameters
  + Choose from 7 pre-installed waveforms and 8 user-defined waveforms.
* Parameters are adjusted using the arrow keys and the Adjust knob.
  + Parameters are selected using the **up** and **down** arrows.
  + Digit to be adjusted is selected by using the **right** and **left** arrows.
  + When entering strings, holding the right or left arrow down for 2 seconds inserts (right-arrow) or deletes (left-arrow) a character.
  + Use the Adjust knob to increment and decrement the selected digit.
  + Parameter values may be directly entered using the app or Terminal Interface.
* Enable Triggers for each Output that is to be used
  + Outputs can be triggered by any or all of the four triggers.
  + Triggers T0 – T3 can be either ENABLED or DISABLED.
* Operation of each Output is triggered by a variety of methods.
  + Pressing the selected trigger button. (Manual trigger)
  + Receiving a low to high transition on the selected Trigger jack.
  + Terminal uses the ‘**!**’ key to trigger the selected trigger.
  + Web Interface has a dedicated Trigger button for each trigger.
* If an Output is already active and REPEAT is non-zero, receiving a trigger command will re-trigger the output.
* If an Output is already active and REPEAT is zero, receiving a trigger command will stop the output.
* Trigger signals merely advance the waveform when in External Clock mode.
* Many parameters can be modulated by the two input signals.
  + Set CV0 and CV1 to select the modulation target parameter (scale, offset, active time, idle value, randomness or off).
  + CV In jacks have a range of -10V to +10V (Bonkulator: -5V to +5V)
  + The sample period for the CV inputs is approximately 20ms. There is no filtering at the input to the converter, so frequencies above 5Hz will produce unpredictable modulation. Could be interesting…
  + If both CVs are specified, the effect is to cascade CV1 after CV0
* Pressing the Adjust Knob activates various functions of Thor’s Arsenal. The use of this button is context-sensitive and is only active within the Settings function. Pressing and holding this knob for longer than 2 seconds brings up the Settings function.

## Screen Saver

In order to maximize display life, the display will turn off after a set time. The default time is 15 minutes. The time can be set in the Settings function with a range of 1 to 9999 minutes.

When the screen saver activates, the screen will show an animation of falling stars. If no key is pressed within 15 minutes after the animation starts, the display will turn blank.

Press any button or send the ‘z’ command to bring the display back.

## Output Details

The Output signal ranges from +10.00V to -10.00V.

Bonkulator: The Output signal ranges from -5.00V to +5.00V or from 0 to 10.00V. (see Range setting)

When the Output isn’t active, the **Idle Value** is sent. Use Output parameters to select the Idle Value.

The app has a dedicated slider control for several parameters, and Terminal mode uses the *S, O* and *c* keys followed by a number. See the table of remote commands for more details.

Thor’s Arsenal Outputs have several parameters. Details for each parameter follow. A star(\*) next to the Output’s name indicates that output is not calibrated.

1. Waveform – These are the signals that can be assigned to the outputs
   1. Fifteen waveforms are available, seven are factory-installed and eight are user-defined. Waveforms are stored as 128 values that range from 0-1023. Depending on the chosen Active Time, 10 to 128 of these values will be used to create the output signal.
   2. The factory-installed waveforms are: Sine, Haystack, Ramp, Pyramid, Pulse, Maytag, and Toggle.
   3. The user-defined waveforms can be entered by the following methods:
      1. Manually via the front panel controls
      2. Drawing the waveform using the app.
      3. Via script using either Terminal mode or an app macro.
      4. Recorded from the CV0 input
      5. Imported from .wave files using the WavTool program.
2. Init Delay – The time in milliseconds to delay before sending the Waveform
   1. Range 0 to 65535 msecs
3. Active Time/Parts – The length in milliseconds of the waveform when in Internal clock mode or the number of samples to be used when in External clock mode
   1. Active Time ranges from 10 msecs – 65535 msecs
   2. Parts ranges from 10 to 128
   3. When in Internal clock mode, the Active Time affects the number of samples used and thus the sample rate.
      1. When the Active Time is a multiple of 128, the number of samples used will be the full 128.
      2. When the Active Time is less than 128, the number of samples will equal the Active Time.
      3. When the Active Time is greater than 128, the number of samples will be the result of a best-fit algorithm.
   4. When in External clock mode, the Delay and Repeat settings are ignored
4. Idle Time – The time in milliseconds to delay before repeating the Waveform
   1. Range 0 to 65535 msecs
5. Repeat – The number of times to repeat the waveform
   1. Range 1 to 65535
   2. Set Repeat to 0 to repeat forever.
6. T0 thru T4 – These are the Trigger/Clock inputs
   1. Each of the four triggers can be either Enabled or Disabled. That means that every output can be triggered by as many as four different trigger signals.
   2. Each trigger can be manually activated by pressing the associated button.
   3. Triggers are activated by a rising edge 5V signal at the input.
   4. In Internal clock mode, triggers re-trigger the trigger state of the output. So, if an output is Idle, receiving a trigger will activate the output. If the output is already active, a trigger will start sending the waveform from the beginning.
   5. When in External clock mode, every trigger merely sends the next output sample.
7. CV0 & CV1 – These are the control voltage inputs.
   1. They can be set to modulate the following Output parameters:
      1. Scale – Range .01x – 100x (Bonkulator: .1x – 10x)
         1. Signal gain is = 100.2Vi, where Vi is the input voltage.  
            This results in a gain range of .01 - 100 for an input range of -10V - +10V.
      2. Offset – Range +/-10V
         1. Output is offset by the input voltage.
      3. Active Time
         1. It modulates the Active Time by modifying the time increment for each step of the output wave. The input range is:  
            0-10V for increments of 1 ms - 512ms.
      4. Idle Value
         1. The output signal level follows the level of the input.  
            This parameter only applies when the trigger is active and Idle Time is greater than 0.
      5. Randomness
         1. An input range of 0-10V results in a randomness of 0-100 per cent of the signal's amplitude.
   2. A star (\*) next to the name indicates that input is not calibrated.
8. Scale
   1. Scale ranges from -100% to 100%
   2. Negative values invert the signal
9. Offset
   1. Range - +/-10.0V%
   2. Bonkulator Range - +/-5.0V%
10. Randomness – 0 to 100%
11. Quantize - Yes/No
    1. Quantize yes forces the output to match the 1V/oct pattern.
12. Idle Value
    1. The Idle Value is sent when the Output is not active.
    2. It is also sent during the Idle Time period when the output is active.
13. Clock
    1. Internal – The next sample is sent based on the Active Time setting.
    2. External – The next sample is sent on the rising edge of the enabled trigger.
14. Range (Bonkulator Only)
    1. +/-5V (Default)
    2. 0-10V (board models 4.0 and later)
15. Trigger Control
    1. Trigger Hold-off
       1. Suppresses triggers for set time after receiving a trigger.
       2. Range 0-65535ms
    2. Trigger Skip
       1. Skips set number of triggers
       2. Range 0-65535 triggers
    3. Trigger Density
       1. Randomly accepts a percentage of triggers
       2. Range 0-100%

## Waveform Details

Thor’s Arsenal has seven factory-installed and eight user-defined waveforms. The user-defined waveforms are managed in the Settings section where the initial names shown below may be changed.

1. Sine – Classic Sine Wave
2. Haystack – First half of Sine wave
3. Ramp – Asymmetrical sawtooth
4. Pyramid – Symmetrical sawtooth
5. Pulse – The Pulse is formed by the waveform transitioning from the Active Value to the Idle Value
   1. **Idle Time** must be non-zero for a pulse to form
   2. **Offset** is disabled for this waveform
   3. A variety of pulse shapes can be formed by adjusting **Scale** and **Idle Value**
6. Maytag – A random sequence
7. Toggle – Toggle creates a scaled and offset square wave that toggles with every transition of the trigger input
8. User Wave 0-7
   1. User Waves are managed in Settings
   2. Default is a flat line
   3. User Waves can be manually entered point-by-point or recorded at CV0
   4. User waves can be drawn and imported via the app.

## Settings Details

The Settings function displays information about Thor’s Arsenal as well as allowing some adjustments to various system parameters. The Bounce function also appears as one of the Settings parameters. Refer to the table below.

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Min** | **Max** | **Sub Menu** |
| Version | -- | -- |  |
| Name | -- | -- |  |
| Encoder Type | Normal | Reverse |  |
| WiFi | Enabled | Disabled | Press Activate |
| Screen Saver | 1 | 9999 |  |
| Waveforms | Wave 0 | Wave 7 | Press Activate |
| Reset | -- | -- | Press Activate |
| Calibrate | Inputs | Outputs | Press Activate |
| Bounce | 0 | 1 | Press Activate |

Notes:

1. **Version** shows the software version number. This is useful when communicating about bugs and questions.
2. **Name**
   1. The Name is displayed on boot-up and is displayed in the remote screens.
   2. This helps keep track of which unit is associated with which screen when there are multiple units installed.
3. **Encoder Type** takes the two types of rotary encoders into account. This makes them easier to purchase if one type is unavailable. If your Adjust knob is working opposite as it should, this is the setting that corrects it. (Bonkulator Only)
4. **WiFi** can be disabled here if you want to turn it off but retain your password. (Bonkulator Only)
5. **Screen Saver** prolongs the life of the OLED display. The screen will blank after the set number of minutes. Press any button to bring the display back.
6. **Waveforms** is where User Waveforms are managed. Each waveform contains 128 samples. Waveforms can be entered manually, drawn or imported using WiFi or the Desktop app, or recorded from CV0. Press Activate after selecting the waveform to bring up the Manage Waveform menu.
   1. **Name** – Enter a meaningful name for your waveform
   2. **Index** – For manual entry. Range 0:128
   3. **Value** – For manual entry. Range 0:4095 (see Output Range)
      1. Displays current value at the element pointed to by Index.
      2. New values are entered by pressing Activate.
      3. Pressing Activate also increments Index.
   4. **Sample Time** – This is the time between recording samples in milliseconds. Calculate using this formula: length of waveform to be sampled in milliseconds / 128
   5. **Record On** – The event that starts the recording of a waveform
      1. **Immediate** – Starts recording when Activated
      2. **Trigger** – Waits for a trigger on any trigger input
      3. **+Change** - Waits for a positive change in the voltage to start recording
      4. **-Change** - Waits for a negative change in the voltage to start recording
   6. **Note** – The app allows for point and click setting of the Index and Value. This greatly speeds up entering new waveforms.
7. **Reset** brings Thor’s Arsenal’s parameters back to their factory defaults. Note that, for the Bonkulator, WiFi parameters are unaffected.
   1. **All**
      1. Resets all of the below
   2. **Parameters** 
      1. Resets Output and Bounce parameters
   3. **User Waveforms**
      1. Resets name to User 0…
      2. Resets wave to Flat Line.
   4. **Input Corrections**
      1. Resets input calibration values to 0.
   5. **Output Corrections**
      1. Resets output calibration values to 0. (Bonkulator Only)
8. **Calibrate** is where the 8 output and 2 input signal paths are calibrated.
   1. **Inputs** – Requires DC voltage source, accuracy .1%
      1. First apply 0VDC to input and adjust Offset to get a correct reading.
      2. Apply 4VDC to input and adjust Scale to get a correct reading.
      3. Test other voltages to verify accuracy. (note that full scale is +/-4.967V)
      4. Set Calibrated to Yes when done.
   2. **Outputs** – Requires DC Voltmeter, accuracy .1% (Bonkulator Only)
      1. Adjust Offset until voltage at Output is as close to zero as possible.
      2. Set Calibrated to Yes when done.
   3. Note: The Bonkulator’s scale, and Thor’s Arsenal scale and offset are calibrated in hardware. See the assembly document in the GitHub archives.
9. **Bounce Function** – An advanced Sample & Hold + DVM (see below)

## Bounce Function

The Bounce function measures the voltage present at the selected CV Input when triggered and displays it to the user. It also scales and offsets the measured voltage and “bounces” the result to the selected output.

This creates a scalable sample and hold function.

All the triggering modes apply.

The Bounce function is accessed via the **Settings** menu. Select which input you want then press Activate to enter the Bounce function.

# Remote Modes

There are three Remote modes available.

1. USB Direct Mode
2. WiFi Mode (Bonkulator Only)
3. Terminal Mode

Important! On the Bonkulator, only connect USB to a powered-up system. Otherwise, the 5V power from the USB will try to power the entire rack. And that’s not good. So, please disconnect the USB cable before you turn off the power.

You can get around this by using the BonkDaddy which disconnects the USB power when Thor’s Arsenal is turned off.

**This is not an issue with Thor’s Arsenal.**

## USB Direct Mode

USB Direct Mode allows you to control Thor’s Arsenal via USB using a desktop app that emulates the unit’s front panel. It also has the added benefit of being able to display trigger events as they occur rather than having to request them as in the web app. USB Direct Mode is automatically enabled when the unit is selected via the app. However, USB Direct Mode may be controlled in these two ways:

* In Settings
  + USB Direct Enabled/Disabled
* Terminal Command
  + U0 - USB Direct Off
  + U1 - USB Direct On

Get the desktop app from the [website](https://www.greenfacelabs.com/bonkulator/#download_the_code). Unzip the folder into a convenient location and run the executable file contained in that folder. If you’re on a Mac, you’ll need to de-quarantine the app. See the section below if you need to do this.

Once you have the app running, you should see a screen like the following figure. Click on the line containing Thor’s Arsenal to enter the app.

A screenshot of a computer

Description automatically generated

### De-quarantining on the Mac

If you download the app on the Mac and try to run it, it will say that the app is damaged and can’t be run. This is an attempt by Apple to get all apps to be distributed through their store. Until this app is available there, you’ll need to de-quarantine it to run.

To do this, using Finder, navigate to the folder that contains the folder that contains the app. Select the folder that contains the app. Then click on Finder->Services->New Terminal at Folder.

This will open the Terminal. Once there, issue this command:

xattr -d com.apple.quarantine bonk\_cp\_app.app

The app will now be ready to use.

## WiFi (Bonkulator Only)

The Bonkulator is equipped with a 2.4GHz WiFi interface. When connected to your network, you can control it using the [Web Interface](https://www.greenfacelabs.com/bonkulator-control/) (see example screen below).

WiFi can be disabled in the Settings function. Make sure it is enabled before proceeding in the connection process.

#### Connecting to WiFi

1. Enter WiFi function and press Activate to scan for networks.
2. Use the up and down arrow keys to select your network. Press Activate.
3. Use the right and left arrow keys with the Adjust control to select each character of your password. Press Activate when your password is entered. Note: It is much easier in Terminal mode. Type your password preceded by $, then press Enter.

The Bonkulator will try to connect using the password that you entered. If successful, the screen will display the connection data. This includes the connection status, the IP it can be found at and the signal strength. The signal strength will also be displayed in the upper right corner of the display as 1-4 bars. The signal strength display is shown in most functions.

If the connection fails, The Bonkulator will give its best explanation of why it failed.

The Bonkulator remembers its connection and will reestablish it if the power is recycled.

Visiting the WiFi function again while connected will display the connection data. Press Activate to re-enter the scan for networks screen.

If The Bonkulator connects to the network, but the Web app has trouble connecting to The Bonkulator, it often helps to recycle the power to The Bonkulator. Try the Web app again after The Bonkulator restarts and re-connects to WiFi.

Important: The Bonkulator only operates via http: Browsers often try to force https: For this reason, it is necessary to remove the ‘s’ from the URL in the address bar of your browser.



Figure 1 Chrome Browser

## Using The APP

The Desktop app and The Bonkulator’s Web Interface were designed to emulate the physical front panel. However, because the app and the Web Interface enjoy the added functionality a computer brings, they have significant differences. Please refer to Fig. 2 below.

A screenshot of a computer

Description automatically generated Figure 2 Web Interface Example

The display is organized into several sections:

* Output Select / Settings Buttons
* Waveform / Message Display
* Parameters
* Triggers / Group
* Slider Controls
* Macro Controls
* Draw Waveform Controls (Not Shown)
* Skin Controls

### Output Select / Settings Buttons

The buttons on the left correspond to the Output select buttons on the front panel. Click these to set the parameters for each output.

The Settings function also has a button. This is the equivalent of pressing and holding the Adjust knob for 2 seconds.

Grouped Outputs are highlighted in purple as shown in Figure 3. (Outputs 0 and 1 are grouped in the example)

### Waveform / Message Display

The Waveform / Message Display either shows the output’s selected waveform or a message. Typically, the message is information about the selected parameter.

Use the Wave/Message button to switch between the displays.

A screenshot of a computer

Description automatically generated

Figure 3 Waveform / Message Display

### Parameters

The app makes it easy to adjust parameters. While you can still adjust parameters a digit at a time, you can now enter parameters from the keyboard by first double-clicking on the parameter then using the dedicated entry field.

Adjusting parameters digit by digit is still possible by using the buttons indicated by the smaller red arrow to select and increment or decrement the digit.

In that group of buttons, there is a Default Selected button. Clicking this button will return the selected parameter to its factory setting.

Below that button is the Activate button. (shown in its inactive state) It corresponds to the Adjust Knob push-button on the front panel. It changes color to match the color of the Adjust Knob. Click this button to activate the selected function.

Keyboard shortcuts that help navigating the parameters are displayed by clicking the “?” button.

A screenshot of a computer

Description automatically generated

Figure 4 Parameters

### Triggers / Group

The Triggers / Group area is a quick way to set the triggers as well as an excellent way to see the trigger settings at a glance. It also allows you to *group* Outputs. This is a feature that is not available from the front panel.

#### Triggers

Each of the eight outputs can be triggered by any of the four triggers by enabling that trigger for that output. Figure 5 shows Trigger 0 (T0) has Output 0 and Output 1 enabled. T1 only has Output 3 enabled. T3 has all outputs enabled except Output 0. And T2 has no outputs enabled.

Trigger the outputs by clicking on the trigger button (T0 – T3). Each output will do whatever it is programmed to do when it receives a trigger signal.

The trigger buttons indicate the state of the triggers regardless of the method of triggering.

The state of the triggers is indicated by the shade of the trigger buttons.

Semi-Transparent: Disabled (no outputs selected)

Dim: Inactive (outputs are selected, trigger is inactive)

Bold: Active (outputs are selected, trigger is active)

#### Grouping

The Group controls are the purple set of buttons pointed to by the smaller arrow in Figure 5. These allow you to control two or more outputs simultaneously by adjusting the parameters of any one of the grouped outputs.

The Grp button indicates the state of the group as well as allowing the user to temporarily disable grouping.

The state of the Group is indicated by the shade of the Grp button.

Semi-Transparent: Disabled (no outputs grouped)

Dim: Inactive (outputs are grouped, grouping is temporarily disabled)

Bold: Active (outputs are grouped, grouping is enabled)

A screenshot of a computer

Description automatically generated

Figure 5 Triggers / Group

### Slider Controls

The Slider Controls are a convenient way to adjust or quickly and clearly see certain commonly used parameters. The Slider controls offer 2 ways to easily adjust parameters.

**Use the slider** – Drag the slider to the new position or click on the new position on the slider.

**Use the numeric entry field** (example indicated by the green arrow) – Click on the field and enter the parameter directly from the keyboard.

When a parameter is being modulated by one of the CV inputs (indicated by the shorter red arrow), the corresponding slider (in this instance the Offset slider) is highlighted to show that the parameter is being modulated.

A screenshot of a computer

Description automatically generated

Figure 6 Slider Controls

### Macro Controls

The Macro controls allow you to capture the actions you take in a script that can be replayed later. Once a macro has been recorded, copy the contents of the Macro area and save it to a text file or other medium. Just paste the macro into the Macro area and press Play to execute the macro.

Once you learn the scripting language (see the reference later in this manual), you can enter macros directly into the macro area by typing them.

Macros can be a powerful tool for remembering patches. Enterprising users can even develop DB applications that manage macros.

Figure 7 shows a macro in the process of being recorded.

A screenshot of a computer

Description automatically generated

Figure 7 Macro Controls: Recording a macro

#### Take Snapshot

The Macro Controls also allow you to take a snapshot of all the parameters of the selected function. Figure 8 shows the result of taking a snapshot of Output 2. The resulting macro can be played or saved just like recorded macros.

Note that the example macro in Figure 8 selects function (Output) 2. It then goes through every parameter in turn setting that parameter’s value to the recorded value.

A screenshot of a computer

Description automatically generated

Figure 8 Take Snapshot

### Draw Waveform Controls

It is possible to draw your own waveforms in the Desktop and WiFi apps. Just select Settings->User Waveforms. Choose a user waveform from the dropdown menu. Then click Draw Waveform.

Use the mouse or touchpad to draw. Once you’re done, click Save Waveform. Or you may click Cancel Draw Waveform to abandon the edits.

Figure 8 shows the screen just after the waveform has been drawn and the Export button has been clicked. Waveforms can be Exported and Imported similar to Macros.

Use the Greenface Labs WavTool to convert .wav files into Thor’s Arsenal format. This puts the resulting data into the clipboard which you can paste into the User Waveform area. Click Import to import the waveform.

A screenshot of a computer

Description automatically generated

Figure 9 Draw Waveform Example

#### Importing Waveforms

Using the Wav2Bonk app, it is possible to import waveforms from .wav file. The example below shows the WavTool app positioned over the Desktop app. The contents of the clipboard have been pasted into the User Waveform text box and the Import button has been pressed. The imported waveform (fm7.txt) is shown in the Waveform display.

A screenshot of a computer

Description automatically generated

Figure 10 Impot Waveform Example

# Terminal Mode

Terminal mode is accessed via the USB-C connector on the front panel. (The Bonkulator uses the MicroUSB connector on the Arduino that is on the back side). Terminal Mode is always available and cannot be turned on or off.

Use a terminal emulation program like PuTTY that has VT-100 mode. Thor’s Arsenal is set to run at 115200 baud but will adapt to the PuTTY settings.

On the Mac use this procedure:

List ports:

ls /dev/tty.\*

Invoke screen:

screen <port> 15200

Quit screen:

ctrl-a ctrl-\

#### Terminal Example Screen

The Terminal screen essentially shows a copy of the user display with the addition of a product identification header that includes the software version and the unit’s name.

Below the product identification header is a status line that usually shows the state of the outputs. If an output is active, the status line will show which trigger activated that output.

Figure 11 shows Output 0, and that Output 0 has been triggered by Trigger 0.

A screenshot of a computer

Description automatically generated

Figure 11 – Example Terminal Interface Screen

#### Thor’s Arsenal Remote Commands

|  |  |  |  |
| --- | --- | --- | --- |
| **Command** | **Front Panel** | **Terminal** | **App** |
| Previous Parameter | ^ | [A | ^ |
| Next Parameter | v | [B | v |
| Next Digit | > | [C | > |
| Previous Digit | < | [D | < |
| Increment Digit | Adjust clockwise | u | Inc |
| Decrement Digit | Counter-clockwise | d | Dec |
| Settings Function | Press & Hold ADJ | \* | Settings |
| Disable Displays |  | Z |  |
| Restore Displays |  | z | Refresh Browser |
| Trigger Selected |  | ! | Use T0-T3 |
| Insert blank char | Press & Hold > | i | Use Param Input |
| Delete char | Press & Hold < | Back Space | Use Param Input |
| Go To Function | Press Output | fx | Click Fxn Name |
| Select Parameter |  | px | Click on Param |
| Select Digit# |  | :x |  |
| Enter Parameter |  | #xxx… | Use Param Input |
| Enter String |  | $ab… | Use Param Input |
| Set Default Value |  | Ix | Use Default Button |
| Idle Value |  | Vx |  |
| Offset | Use Adj | Ox | Use Offset slider |
| Scale | Use Adj | Sx | Use Scale slider |
| Randomness | Use Adj | Rx | Use Random slider |
| Set Phase |  | Px |  |
| Active Time | Use Adj | Ax | Use Active T slider |
| Group |  | Gx | Use Group Controls |
| Group Active |  | gx | Use Grp Active Button |
| Set/Examine DAC |  | =x |  |
| Quantization |  | Qx |  |
| Display Control |  | Jx |  |
| USB Direct |  | Ux |  |
| Select Trigger |  | tx |  |
| Clear Trigger |  | cx |  |
| Disable Selected Trig |  | T0x |  |
| Enable Selected Trig |  | T1x |  |
| Toggle Selected Trig |  | T2x |  |
| Disable All Triggers |  | T3 |  |
| Trigger All Triggers |  | T4 |  |
| Clear All Triggers |  | T5 |  |
| Trigger Report |  | T6 |  |
| Gen Macro Selected |  | M0 |  |
| Gen Macro Input Cal |  | M1 |  |
| Gen Macro Output Cal |  | M2 |  |
| Gen Macro User Wave |  | M1x |  |
| Dump Waveform |  | Dxy |  |
| Receive User Waveform |  | wx | Use Import Button |
| Print WiFi Status | Use WiFi Function | W | Use WiFi Function |
| Comment (for scripts) |  | / |  |

##### Command Parameter Details

* Go To Function – **fx** where x is the function number. The numbers 0 – 7 denote Outputs and 8 denotes Settings.
* Select Parameter – **px** where x is a single digit in the range of 0-n, and n = the number of parameters minus 1 for a chosen function.
* Select Digit# - **:x** where x is the digit number. Digit 0 is the most significant digit.
* Enter Parameter – **#xx…** The number of digits depends on the parameter. The range also depends on the parameter. The # is optional.
* Enter String – **$ab…** Start with $ followed by desired string. Terminate with carriage return.
* Set Default Value – **Ix** where x is the parameter number. (x isn’t implemented yet, use I without an argument to initialize the selected parameter)
* Idle Value – **Vx** where x is an integer that ranges from -1000 to +1000 (-5000 to +5000 for the Bonkulator). This number, divided by 100, is the actual output voltage.
* Offset – **Ox** (same as Idle Value)
* Scale – **Sx** where x ranges from -100 to +100. (Negative scales invert the waveform)
* Randomness – **Rx** where x ranges from 0 to 99. Randomness is relative to the waveform’s amplitude.
* Set Phase – **Px** where x is the phase angle in per cent. X ranges from 0 – 100.
* Active Time – **Ax** where x ranges from 10 to 65535.
* Group – **Gx** where x ranges from 0-255. Sets bitmap indicating which outputs are grouped. Output 0 is the least significant bit. Example: G129 groups Outputs 0 and 7.
* Group Active – **gx** where x can be either 0 or 1. Sets whether the group is active or not.
* Set/Examine DAC – **=x** where x ranges from 0 – 4095. This sets the Idle Value of the selected output. Entering = without an argument will print the current value of the DAC.
* Quantization – **Qx** where x can be either 0 or 1. Sets whether Quantization is active or not.
* Display Control – **Jx** where x can be either 0 or 1. Sets whether the OLED display is on or off. Turning off the OLED display noticeably speeds up response when using the terminal interface.
* USB Direct – **Ux** where x can be either 0 or 1. Sets whether the USB Direct is on or off.
* Select Trigger – **tx** where x denotes trigger number (0 – 3). This command is used in conjunction with the trigger commands that refer to the selected trigger.
* Clear Trigger – **cx** where x denotes Output 0-7. Stops output if triggered. Does not disable trigger.
* Disable/Enable/Toggle Selected Trigger – **Tyx** where x denotes Output 0-7 and y is 0, 1 or 2 to disable, enable or toggle respectively.
* Bulk Trigger Cmds - **Tx** where x is 3, 4, 5 and triggers, clears or disables all triggers respectively.
* Trigger Report – **T6** generates a report showing which triggers are enabled and/or are active.
* Gen Macro – **Mx** where x can be 0, 1 or 2. See [below](#_Generating_Terminal_Mode) section for more details.
* Gen Macro User Wave - **Mx** where x denotes wave number plus ten.
* Dump Waveform - **Dxy** where x denotes Output 0-7. This sends the waveform values to the terminal. If y is present, it dumps the reference waveform
* Receive User Waveform – **wx** where x denotes Output 0-7. It expects waveform values to follow.
* Comment – the **/** character causes the following text up until the carriage return to be ignored.

Notes:

Under the hood, the app and Web interface use the same command characters as the Terminal. Commands with yellow background require the user to press Enter when using a terminal.

Parameters and digits may be navigated in Terminal mode using the arrow keys.

Pro Tip: Print the two previous pages on both sides of a card stock sheet and laminate it for a handy reference!

Disable Displays - Z is used to speed up remote scripting.

Restore Displays – z restores display that has been disabled by command or has been turned off due to inactivity.

#### Generating Terminal Mode Macros

Terminal Mode offers its own version of parameter macros.

Software updates often contain new features that affect EEPROM allocation and thus will require initialization of the EEPROM when installed. This means that all parameters, user waveforms and calibration data will be lost and must be re-entered.

Terminal Mode Macros are a way of minimizing the impact of this condition. When a Macro is created, it will be sent to the terminal screen where it can be copied and pasted into a text file for later use.

To run the Macro, merely paste it into the Terminal screen. Do this after a software update and whenever you need to set parameters to a known state. It is recommended that Calibration data macros only be run after a software update or if calibration data is lost or corrupted.

Macros are initiated by the ‘M’ character followed by a number that signifies the operation.

* M0 generates a macro that contains the parameters for the currently selected function.
* M1 generates a macro that contains the Input calibration data
* M2 generates a macro that contains the Output calibration data
* M10 – M17 generate macros that contain the User Waveform data. The selected waveform will equal the entered number minus ten. Example: M15 denotes User Waveform 5.

# Software Update

When a software update becomes available, it will be posted on the website as a zipped package. There are two types of packages. One for Win and one for Mac. Each package contains the files needed to update the code. Unzip the one you need into an empty folder on your machine.

Follow these steps to update the code.

1. Note: If the update changes the EEPROM storage mapping, the unit will automatically initiate a Software Reset. Any stored settings will be lost, so make sure to save these settings before updating if they are important. See [Generating Parameter Macros](#_Generating_Parameter_Macros)
2. Skip steps 3-5 if Thor’s Arsenal
3. Turn off power to the rack and remove The Bonkulator.
4. Leaving the power cable connected, place The Bonkulator on a non-conductive surface. Make sure nothing touches The Bonkulator that might cause a short.
5. Re-power the rack.
6. Plug the appropriate USB cable into the unit.
7. Run update.bat or update.sh if Mac. It will show available COM ports. Choose by entering the number/name of the port.
8. If it connects properly, you will see that a new drive has been mounted. The code is copied to this drive. If the update succeeds, the unit will reboot as normal with the new software installed and the drive will be removed. You can see the new version number in the Settings function.

## Disaster Recovery

When things go wrong, as they sometimes do, Thor’s Arsenal might become unresponsive. When this happens, the program must be reset. Usually recycling the power\* will fix the problem, but occasionally things have gone so bad that the code must be reloaded.

There is a reset button on the front panel (Arduino for the Bonkulator). Pressing this twice quickly will force the unit into bootloader mode. You will see the yellow LED pulsate to indicate when the unit is in bootloader mode. The update process should work at this time.

If pressing the reset button doesn’t work, put a jumper onto J20 and press reset again. This should open a drive on your computer. When this happens, remove the jumper. Drag the file Blink.ino.elf.uf2 to the drive. Once loaded, you should see the Arduino blink some LEDs. The normal update process should now work.

Please contact Greenface Labs directly if this doesn’t work.

# Appendix

## Remote Commands That Don’t Take an Argument

1. **Case 'Z'**:
   * **Description**: Disables the display and terminal.
   * **Action**: Calls [ui.all\_off()](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) to turn off all UI elements.
2. **Command 'z'**:
   * **Description**: Restores the display.
   * **Action**: Calls [restore\_display()](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) to bring the display back to its previous state.
3. **Command '!'**:
   * **Description**: Activates a specific function. Or initiates a trigger from the selected trigger.
   * **Action**: Calls [activate()](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) to perform the activation.
4. **Command 'u'**:
   * **Description**: Adjusts the group upwards.
   * **Action**: Calls [adjust\_group(1)](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) to move the group up by one.
5. **Command 'd'**:
   * **Description**: Adjusts the group downwards.
   * **Action**: Calls [adjust\_group(-1)](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) to move the group down by one.
6. **Command '\*'**:
   * **Description**: Selects the settings function.
   * **Action**: Calls [select\_fxn(SETTINGS\_FXN)](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) to switch to the settings function.
7. **Command 'i'**:
   * **Description**: Inserts a space character.
   * **Action**: Calls [selected\_fxn->insert\_char(' ')](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) to insert a space character at the current position.
8. **Command 127**:
   * **Description**: Removes a character (typically the delete key).
   * **Action**: Calls [selected\_fxn->remove\_char()](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) to remove the character at the current position.
9. **Command 'A'**:
   * **Description**: Moves up a parameter in escape mode.
   * **Condition**: Only executes if [esc\_mode](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) is true.
   * **Action**: Calls [dec\_param\_num()](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) to decrement the parameter number.
10. **Command 'B'**:
    * **Description**: Moves down a parameter in escape mode.
    * **Condition**: Only executes if [esc\_mode](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) is true.
    * **Action**: Calls [inc\_param\_num()](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) to increment the parameter number.
11. **Command 'C'**:
    * **Description**: Moves right in escape mode.
    * **Condition**: Only executes if [esc\_mode](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) is true.
    * **Action**: Calls [inc\_dig\_num()](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) to increment the digit number.
12. **Command 'D'**:
    * **Description**: Moves left in escape mode.
    * **Condition**: Only executes if [esc\_mode](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) is true.
    * **Action**: Calls [dec\_dig\_num()](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) to decrement the digit number.

## Remote Commands That Take an Argument

Note: int\_param is the integer following the command. Example. Int\_param for ‘=403’ would be 403, and the command would be ‘=’

1. **Command '['**:
   * **Description**: Enters escape mode and sets the command to the next character in the input string.
   * **Action**: Sets [esc\_mode](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) to true and updates [cmd](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) to the second character of [in\_str](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
   * **Example:** [A sends the up arrow, which goes to the previous parameter.
2. **Command '='**:
   * **Description**: Handles idle value or DAC output based on the second character of the input string.
   * **Action**: If the command is ‘=I’ (second character is 'I'), it prints the idle DAC value of the selected output. Otherwise, it sets the DAC output to [int\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
   * **Example**: =403 sets the current output DAC to 403.
3. **Command ':'**:
   * **Description**: Updates the digit number and prints parameters.
   * **Action**: Calls [put\_dig\_num](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) with [int\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) and prints the parameters.
   * **Example**: :0 sets the digit number of the selected parameter to 0.
     1. Digit 0 is the MSD (most significant digit)
     2. The LSD (least significant digit) is the number of digits minus one.
     3. Setting the digit number to anything beyond the number of digits minus one just selects the LSD.
4. **Command 'f'**:
   * **Description**: Selects a function based on the integer parameter.
   * **Action**: Calls [select\_fxn](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) with [int\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
   * **Example**: f6 selects Output 6, and f8 selects Settings.
5. **Command 'p'**:
   * **Description**: Updates the parameter number and prints parameters.
   * **Action**: Calls [put\_param\_num](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) with [int\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) and prints the parameters.
   * **Example**: p0 selects parameter 0. If in an Output function this would select Waveform.
6. **Command 'c'**:
   * **Description**: Clears a trigger for the specified output.
   * **Action**: Calls [clear\_trigger](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) with [int\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) and the trigger number of [selected\_trigger](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
   * **Example**: c3 will stop the Output 3 waveform if active and return it to the Idle Value. It will not disable the trigger for that output.
7. **Command 't'**:
   * **Description**: Selects a trigger and optionally triggers it.
   * **Action**: Calls [select\_trigger](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) with [dig1](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html). If the third character of [in\_str](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) is '!', it triggers [selected\_trigger](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) and restores the parameter number.
   * **Example**: t3! Selects Trigger 3 and activates that trigger.
8. **Command 'w'**:
   * **Description**: Receives a user waveform.
   * **Action**: Calls [recv\_user\_waveform](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) with [in\_str](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
   * **Example**: w0,2080,2662,3503,… 128 values between 0 and 4095.
9. **Command 'I'**:
   * **Description**: Initializes the selected parameter and that parameter of every output that is in the current group. If the current function is not in a group, only the current selected parameter will be initialized.
   * **Action**: Calls [init\_group](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
   * **Example**: I initializes the Waveform to Sine when the selected parameter is Waveform.
10. **Command 'T'**:
    * **Description**: Modifies triggers based on [dig1](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
    * **Action**: Depending on the value of [dig1](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html), it updates, toggles, disables, or triggers all triggers, or displays a trigger report.
    * **Sub-Command 0:** Disable the selected trigger for the specified output.
      1. **Example:** T02 will disable the selected trigger for Output 2
      2. **Note:** use Command ‘t’ to select a trigger.
    * **Sub-Command 1:** Enable the trigger for the specified output.
      1. **Example:** T12 will enable the selected trigger for Output 2
      2. **Note:** use Command ‘t’ to select a trigger.
    * **Sub-Command 2:** Toggle the trigger for the specified output.
      1. **Example:** T22 will toggle the selected trigger for Output 2
      2. **Note:** use Command ‘t’ to select a trigger.
    * **Sub-Command 3:** Disable all triggers for all outputs.
      1. **Example:** T3 will disable all triggers. This will leave the Trigger buttons in the disabled state. Re-enable a trigger for an output to re-enable the trigger.
    * **Sub-Command 4:** Trigger all triggers.
      1. **Example:** T4 will trigger all enabled triggers for all outputs.
    * **Sub-Command 5:** Clear all triggers.
      1. **Example:** T5 will stop all outputs that have active waveforms and return those outputs to the Idle Value.
    * **Sub-Command 6:** Issue a Trigger Report.
      1. **Example**: T6 issues a Trigger Report
      2. **Note**: This command only works in Terminal Remote mode.
11. **Command 'U'**:
    * **Description**: Sets USB direct settings.
    * **Action**: Calls [settings\_put\_usb\_direct](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) with [int\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
    * **Example**: U0 turns USB Direct OFF. U1 turns it ON.
    * **Note**: Selecting a Port from the app will automatically turn USB Direct ON. There is no reason to use U1 in normal operation.
    * **Note**: If you should connect via Terminal while the module is still in USB Direct mode, you can turn it OFF and return to Terminal Mode by using command: U0
12. **Command 'J'**:
    * **Description**: Turns the front panel display on or off based on [int\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
    * **Action**: If [int\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) is 0, it turns the display off. If [int\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) is 1, it turns the display on.
    * **Example**: J0 turns off the display and J1 turns it back on.
13. **Command 'M'**:
    * **Description**: Generates macros to the Terminal screen based on [int\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
    * **Action**: Depending on the value of [int\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html), it generates different macros for input calibration, output calibration, user waveform parameters, or general parameters.
      1. **Sub-Command 1:** Generate Input calibration parameters macro.
      2. **Sub-Command 2:** Generate Output calibration parameters macro.
      3. **Sub-Commands 10 to 17:** Generate User waveform parameters macro.
      4. **Default:** Generate parameters macro for the currently selected function. (Output, Settings, etc.)
    * **Example**: M1 generates a macro that sets the input calibration values. This is good for remembering these values in case of a software update or Reset that would erase them.
    * **Example**: M14 would generate a macro for User Waveform 4
    * **Note**: The Default behavior occurs when M is anything other than the above sub-commands.
    * **Example:** M by itself or M paired with any integer above 17 while in an Output function would generate a macro for the current Output parameters.
14. **Command 'P'**:
    * **Description**: Updates the output phase if in an Output function.
    * **Action**: Calls [param\_put](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) with [int\_param % 100](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) and [OUTPUT\_PHASE](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html), then calls [update\_phase](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
    * **Example**: P50 shifts the selected waveform by 180 degrees. This command only works with the standard waveforms.
15. **Command 'D'**:
    * **Description**: Prints the values of one of the standard waveforms to the Terminal.
    * **Action**: Calls [dump\_waveform](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) with [dig1](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) and a boolean indicating if [dig2](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) is greater than 0.
    * **Example**: D1 prints a Waveform Report for Output 1’s selected waveform.
    * **Example**: D11 prints the *reference* waveform for Output 1's selected waveform. This is the waveform before it is modified by the parameters.
    * **Note**: Command D only works in Terminal mode.
16. **Command 'G'**:
    * **Description**: Sets the group. The parameter is interpreted as a binary number whose bit positions indicate which outputs are to be grouped.
    * **Action**: Updates the [group](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) variable with [int\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
    * **Example**: G3 groups Output 0 and Output 1
    * **Example**: G192 groups Output 6 and Output 7
    * **Note**: This works in Terminal mode. But, at present, there is no indication of grouped outputs like there is in the app.
    * **Note**: If no outputs are grouped, adjustments to the group merely adjust the selected parameter of the currently selected function.
17. **Command 'g'**:
    * **Description**: Activates or deactivates the group.
    * **Action**: Sets [group\_active](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) to true if [int\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) is greater than 0, otherwise sets it to false.
    * **Example**: g0 disables the group. Disabling the group unlinks the outputs assigned to the group without unassigning the outputs. This is a convenience when you want to adjust a parameter of a single output without affecting other outputs in the group. Issuing g1 will re-enable the group so that all outputs in the group are linked again.
18. **Command 'O'**:
    * **Description**: Sets the output offset for the group. The value ranges from -10.00V to +10.00V.
    * **Action**: Calls [put\_group](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) with [int\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) and [OUTPUT\_OFFSET](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
    * **Example**: O500 sets the Offset to +5V. O-123 sets the Offset to -1.23V.
    * **Note**: This command only affects the waveform. It does not affect the Idle Value.
19. **Command 'S'**:
    * **Description**: Sets the output scale for the group. The value ranges from -100 to +100. The units are per cent.
    * **Action**: Calls [put\_group](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) with [int\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) and [OUTPUT\_SCALE](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
    * **Example**: S-50 sets the scale of the output waveform to 50% full-scale. The negative sign inverts the signal around its middle point.
20. **Command 'A'**:
    * **Description**: Sets the active time for the group.
    * **Action**: Calls [put\_group](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) with [int\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) and [OUTPUT\_ACTIVE\_TIME](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
    * **Example**: A3423 sets the Active Time for the group to be 3423ms.
21. **Command 's'**:
    * **Description**: Reserved.
    * **Action**: Does nothing.
22. **Command 'R'**:
    * **Description**: Sets the randomness for the group. Randomness ranges from 0 to 99% of the selected signal level.
    * **Action**: Calls [put\_group](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) with [int\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) and [OUTPUT\_RANDOMNESS](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
    * **Example**: R43 sets the Randomness to 43%
23. **Command 'V'**:
    * **Description**: Sets the idle value for the group. The Idle Value is the voltage at the output when it isn’t active (sending a waveform). It ranges from plus full scale to minus full scale.
    * **Action**: Calls [put\_group](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) with [int\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) and [OUTPUT\_IDLE\_VALUE](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
    * **Example**: V500 sets the Idle Value to 5.00V.
24. **Command 'Q'**:
    * **Description**: Sets the quantize parameter for the group.
    * **Action**: Calls [put\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) with [int\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) and [OUTPUT\_QUANTIZE](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
    * **Example**: Q1 turns quantize on. Q0 turns it off.
25. **Command 'W'** (if [WIFI\_INSTALLED](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html)):
    * **Description**: Clears the terminal and prints WiFi status. This command does not take a parameter. It might in the future.
    * **Action**: Calls [clearTerminal](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) and [printWifiStatus](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
    * **Example**: If WiFi is installed (the device is a Bonkulator), W will print a WiFi status report.
26. **Command '$'**:
    * **Description**: Sets a string variable.
    * **Action**: Calls [put\_string\_var](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) with the URL-decoded substring of [in\_str](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
    * **Example**: $Mandy sets the string variable to Mandy.
    * **Note**: This only works on string variables. An example is the device Name in Settings.
    * **Note**: This is the preferred method for entering string variables in Terminal mode.
27. **Command '/'**:
    * **Description**: Comment character, does nothing.
    * **Action**: No action taken.
28. **Command '#'**:
    * **Description**: Updates a parameter and applies it to the waveform and graphs it if in output function.
    * **Action**: Calls [put\_param\_w\_offset](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) with [int\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html). If in output function, it applies parameters to the waveform and graphs it based on the parameter number.
    * **Example**: #780 updates the current parameter to 780.
    * **Note**: This works for numeric variables and also for select variables. For select variables just enter the index as the integer argument.
    * **Example**: #0 sets a select variable to the first menu item.
    * **Note**: The # is optional. Just entering a number will work. This command is reserved for apps that require a command character when building their instruction strings.
29. **Command ‘=’**:
    * **Description**: The = command is typically used for development but might be useful in scripting. It has two modes. I and O. I mode prints information to the app. Or it prints to the terminal if the app isn’t active. O mode takes a parameter and outputs it to the module. Currently the = command only deals with the DAC (digital-to-analog-converter) which determines the output voltage.
    * **I Mode**:
      1. **Description**: Reads the DAC setting given the output number.
      2. **Example**: =I3 reads the DAC setting for Output 3.
      3. **Example**: = by itself reads the DAC setting of the selected output.
    * **O Mode**:
      + **Description**: Sets the DAC given the output number.
      + **Example**: =O35643 sets the DAC for Output 3 to 5643.
      + **Example**: =5643 by itself sets the DAC for the selected output to 5643.

* **Command '=':**
  + Description: If the second character is 'I', print the idle value of the selected output.
  + Otherwise, output the DAC value.
* **Command 'M':**
* **Command '#':**
  + Set the parameter with an offset using the provided integer parameter.
  + If the selected function is an output function:
    - **Sub-Command OUTPUT\_SCALE:** Apply parameters to the waveform and graph the waveform.
    - **Sub-Command OUTPUT\_OFFSET:** Apply parameters to the waveform and graph the waveform.
    - **Sub-Command OUTPUT\_RANDOMNESS:** Apply parameters to the waveform and graph the waveform.
    - **Sub-Command OUTPUT\_QUANTIZE:** Apply parameters to the waveform and graph the waveform.

1. **Check if**[**in\_str**](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html)**is "=" or if the second character of**[**in\_str**](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html)**is 'I':**
   * If true:
     + Determine [temp\_int](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html):
       - If the second character of [in\_str](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) is 'I', convert the substring starting from the third character to an integer.
       - Otherwise, use [selected\_output\_temp](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
     + Constrain [temp\_int](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) to be within the range [0, NUM\_OUTPUTS - 1].
     + Create a string [temp\_string](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) that includes the DAC value for the output specified by [temp\_int](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
     + If USB direct mode is enabled:
       - Format [temp\_string](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) as HTML and include the output number.
       - Send the formatted string to USB.
     + Otherwise:
       - Print [temp\_string](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) to the terminal with the output number.
2. **If the first condition is false:**
   * Check if the second character of [in\_str](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) is 'O':
     + If true:
       - Convert the third character of [in\_str](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) to an integer and assign it to [selected\_output\_temp](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
       - Convert the substring starting from the fourth character to an integer and assign it to [temp\_int](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
     + Otherwise:
       - Use [int\_param](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) as [temp\_int](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
   * Call [dac\_out](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) with [selected\_output\_temp](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) and [temp\_int](vscode-file://vscode-app/c:/Users/shann/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) to output the DAC value.