



University of British Columbia  
Electrical and Computer Engineering  
Circuit Analysis I  
ELEC201

## Using the BB2Scope and Gen2BB Boards

Copyright © 2019-2020, Jesus Calvino-Fraga. Not to be copied, used, or revised  
without explicit written permission from the copyright owner.

### Table of Contents

Introduction .....	2
Features and Limitations .....	2
Requirements for Using BB2Scope and Gen2BB boards.....	3
BB2Scope Board Signal Connections .....	4
Gen2BB Signal Connections .....	4
Setup for Quick Testing.....	5
BB2Scope Operation .....	6
Trigger Operation .....	7
Horizontal Axis .....	8
Vertical Axis.....	9
Changing Graph Plot Settings .....	10
DSP option (Low Pass Filter).....	11
Parametric (XY) Plots .....	12
Math Plot.....	13
Using the Cursors for Differential Measurements .....	14
Controlling the outputs of the Gen2BB board .....	15
The Matplotlib Navigation Toolbar.....	16
Keyboard Shortcuts .....	17
On Screen Voltage Display .....	18
Auto Trigger with Moving Average.....	19
Removing Broken Wires from the Connectors.....	20
Saving and Retrieving the BBScope configuration .....	21
Examples .....	22
Half Wave Rectifier.....	22
555 Timer A-stable Oscillator .....	24
Colpitts Oscillator.....	25
I vs. V Curve Tracer.....	27
Running BB2Scope on Different Operating Systems.....	29
Running BB2Scope on Linux.....	29
Running BB2Scope on macOS.....	30
Running BB2Scope on Windows using the Official Python Distribution.....	33

# Introduction

As the COVID19 pandemic continues to disrupt the world in 2020, we at UBC have been greatly affected as well. With online teaching mandatory, we are confronted with the inaccessibility of electronics laboratory facilities required to successfully teach the lab portion of the Circuit Analysis I course. For this reasons a couple of electronics boards were designed during the summer of 2020 in order to facilitate the completion of laboratory experiments by ELEC201 students at home. These boards are an USB oscilloscope, the BB2Scope board and a function generator, the Gen2BB board. The boards were manufactured during the summer of 2020 and shipped to the students taking the course so that they can complete the lab work at home.

## Features and Limitations

The BB2Scope USB oscilloscope includes several features found in many commercially available products, including:

- 2 channels.
- Adjustable gain and offset.
- DC and AC modes.
- Auto and Normal modes.
- Adjustable trigger: source, level, and edge.
- YT and XY modes.
- Math trace.
- Python GUI, with cursors, visual trigger indication, zoom/navigation, and on-screen display of signal frequency and channel voltages.

Due to hardware limitations and in order to keep the design as simple and inexpensive as possible, BB2Scope has these two limitations:

- The sampling frequency is 500 kHz per channel. This limits the practical maximum frequency of the signals applied to the oscilloscope to about 50 kHz.
- The input voltage range is limited to  $\pm 32\text{V}$  maximum.

These limitations are not relevant for the laboratory experiments to be carried out in ELEC201.

The Gen2BB board has these features:

- 2 output channels.
- $\pm 11\text{V}$  peak maximum amplitude, typical.
- Independent control of wave amplitude and offset.
- Sine, square, triangle, and ramp outputs.
- Very low output impedance. The Gen2BB board can be used as a transformer replacement. Both single and center tap transformers can be simulated.
- Configurable default power-on configuration.
- +12V, -12V, and +5V power supply outputs with over-current protection.
- Controlled using the BB2Scope board.

The maximum frequency the Gen2BB board can deliver is about 20 kHz, while the minimum frequency is about 14 Hz. These limitations should not be an issue for the laboratory experiments to be carried out in ELEC201.

## Requirements for Using BB2Scope and Gen2BB boards.

To use the BB2Scope and Gen2BB boards we need the following:

- The BB2Scope and Gen2BB boards.
- A reasonably recent computer running Windows 10. That said the setup has been tested successfully on 2009 laptop running Windows 7 and on Windows 8. The setup also works on Linux and macOS. For more information about getting BB2Scope working on these operating systems, check the section '[Running BB2Scope on Different Operating Systems](#)' in page 29.
- Python 3.x with the packages 'matplotlib', 'pyserial', and 'scipy' installed. Other packages may be needed as well if they don't come pre-installed with the used Python 3 distribution. Optionally you can download a Python distribution which comes with all the required packages installed. The one used by the author for the development of the GUI software of the BB2Scope board is WinPython. It can be downloaded from:

<https://winpython.github.io/>

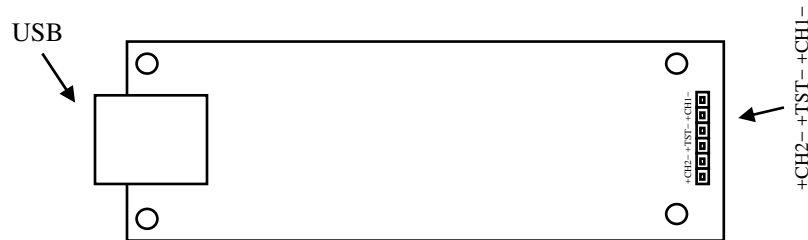
- The distribution used at the time of writing this document was "*WinPython64-3.8.1.0cod*". If you install WinPython, remember to run the program "*WinPython Control Panel.exe*" as an administrator, click the 'Advanced' tab, and then "register distribution" so files with the '.py' and '.pyw' extensions are associated with the installed distribution of Python. An alternative way of installing Python 3.x on Windows that doesn't use WinPython is described in the section '[Running BB2Scope on Different Operating Systems](#)' in page 29.

**WARNING: Matplotlib in Python 3.9 seems to be broken and doesn't install properly. Install an earlier version of Python instead.**

- An USB A-B cable.
- A 12V DC power adapter. This adapter should be capable of delivering at least 1A.
- The script "*BB2Scope.pyw*" which provides the front end GUI for the oscilloscope.
- The USB driver. Once the BB2Scope is connected to the computer, the USB driver should be installed automatically by Windows 10. If that is not the case, the driver would have to be installed manually. The driver can be downloaded from:

<https://www.ftdichip.com/Drivers/VCP.htm>

## BB2Scope Board Signal Connections



BB2Scope has two connectors. The big metallic connector to the left as shown in the figure above is the USB connector. The female header connector to the right is the signals connector which the following signals are available:

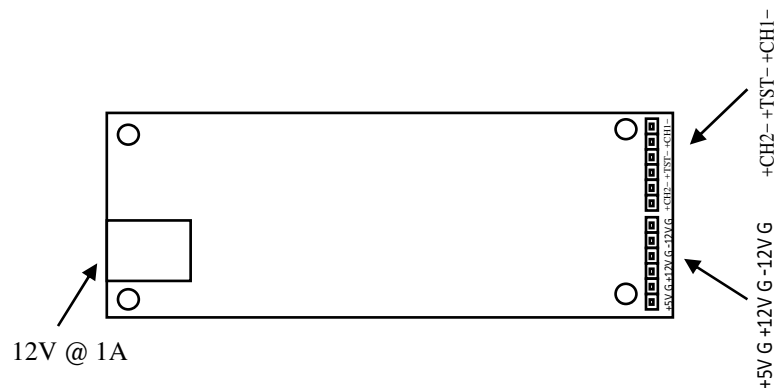
**CH1:** The input voltage for channel 1.

**TST:** The test signal output. This signal is currently used to send commands to the Gen2BB board.

**CH2:** The input voltage for channel 2.

The '-' terminal of the three signals are connected together and to ground.

## Gen2BB Signal Connections



Gen2BB has three connectors. The big connector to the left as shown in the figure above is the 12V power adapter input. The female header connector to the top right is the signals connector which the following signal outputs/input:

**CH1:** The generator output voltage 1.

**TST:** The test signal input. This signal is currently used to receive commands to configure the output voltages CH1 and CH2.

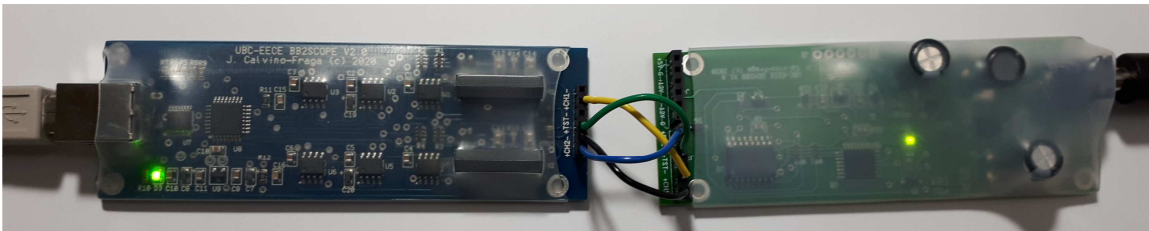
**CH2:** The generator output voltage 2.

The ‘-’ terminals for all the signals are connected together and to ground.

The female header connector to the bottom right is the voltages connector with the following voltages available: +5V, +12V and -12V. The terminals marked as ‘G’ are common ground to all the output voltages.

## Setup for Quick Testing

If you want to test the basic functionality of both the BB2Scope and Gen2BB boards, connect them as shown in the picture below.



Only four wires are required between the two boards:

**Wire 1:** Connect ‘+’ of TST in BB2Scope to ‘+’ of TST in Gen2BB (green wire).

**Wire 2:** Connect ‘-’ of any signal in BB2Scope to ‘-’ of any signal in Gen2BB (black wire).

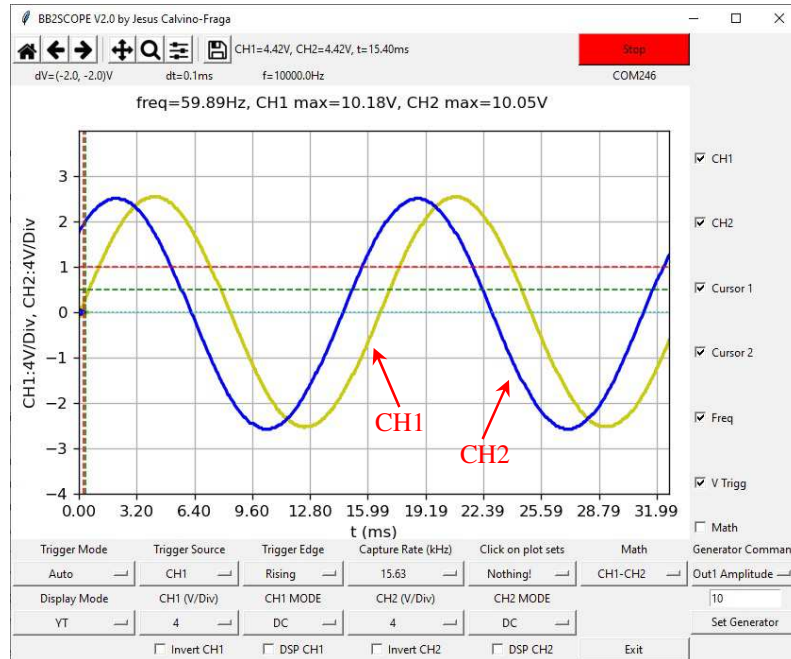
**Wire 3:** Connect ‘+’ of CH1 in BB2Scope to ‘+’ of CH1 in Gen2BB (yellow wire).

**Wire 4:** Connect ‘+’ of CH2 in BB2Scope to ‘+’ of CH2 in Gen2BB (blue wire).

Also connect an USB cable from the computer to the BB2Scope board and the 12V power adapter to the Gen2BB board.

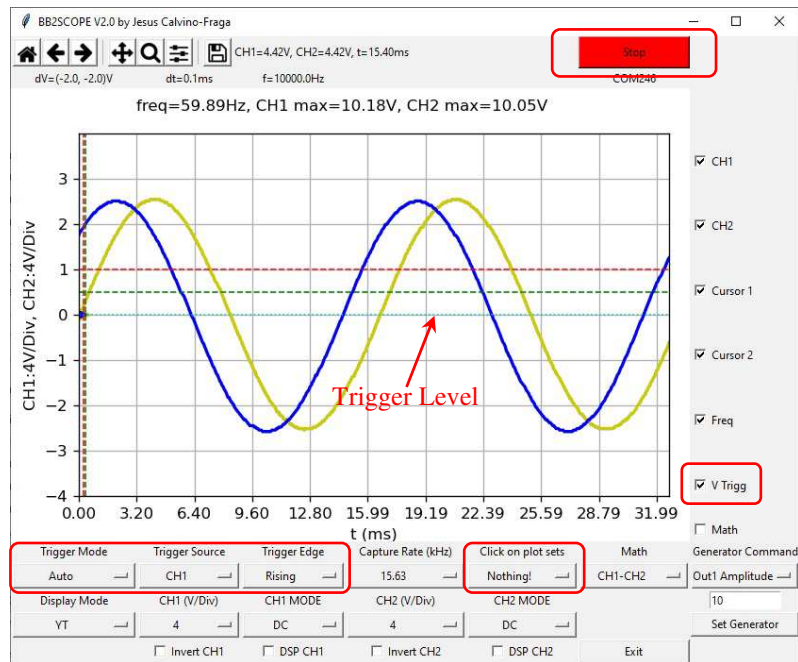
## BB2Scope Operation

The basic operation of the BB2Scope and Gen2BB board can be explored using the connections described above. Run the 'BB2Scope.pyw' Python 3.x script. The following screen is displayed:



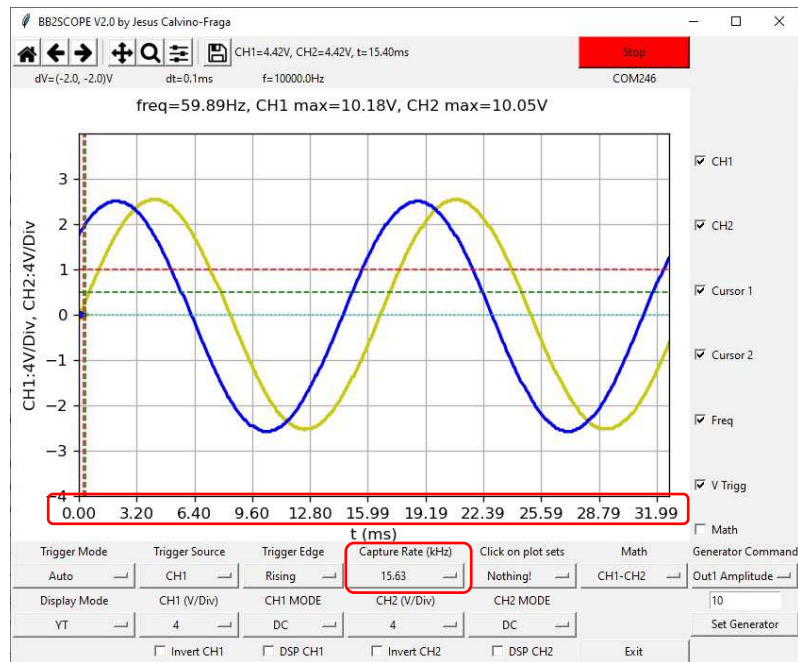
The figure above shows the two channel waveforms: yellow is for channel 1 and blue is for channel 2. These waveforms are updated in real time every 100ms.

## Trigger Operation



The buttons used for trigger operation are shown in the figure above. The ‘**Trigger Mode**’, ‘**Trigger Channel**’, and ‘**Trigger Edge**’ option menus control the type, source, and edge for the oscilloscope trigger source in a similar manner to the oscilloscopes in the lab. The trigger level is displayed as a thin dotted blue line in the graph, as indicated by the red arrow. To change the trigger level, select the option ‘**V Trigg**’ in the ‘**Click on plot sets**’ button, and then click on the plot to set the new trigger level. To hide/show the trigger level line, use the select box ‘**V Trigg**’ on the right of the figure. The red ‘**Stop**’ push button at the top is used to disable any further triggering and preserve the current plot display graph.

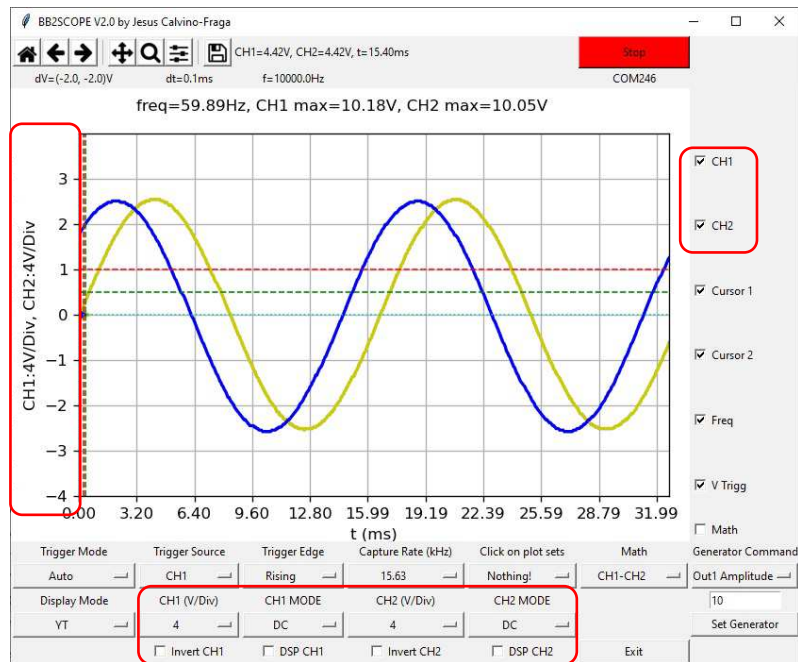
## Horizontal Axis



The figure above shows the button that controls the horizontal axis. The **'Capture Rate'** push button allows for the selection of different capture rates, which determines the scale of the horizontal axis.

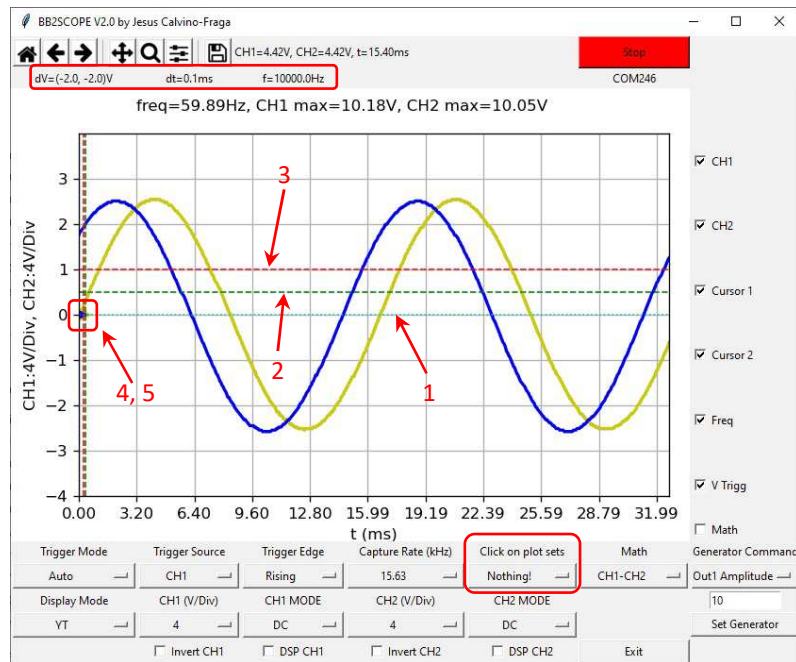


## Vertical Axis



The buttons and check boxes indicated in the figure above control how the vertical axis for each channel is displayed. The '**CHx (V/Div)**' button selects the vertical scale for the corresponding channel. The '**CHx MODE**' selects the type of coupling for the corresponding channel, either 'DC' coupling or 'AC' coupling are available. The '**Invert CHx**' check box will multiply the corresponding channel by  $-1.0$  when selected. The '**DSP CHx**' check box, when selected, applies some digital signal processing (low pass filter) to the corresponding signal before displaying it in the graph plot. The two check boxes to the right named '**CH1**' and '**CH2**' are used to either show or hide the corresponding channel trace in the plot graph.

## Changing Graph Plot Settings

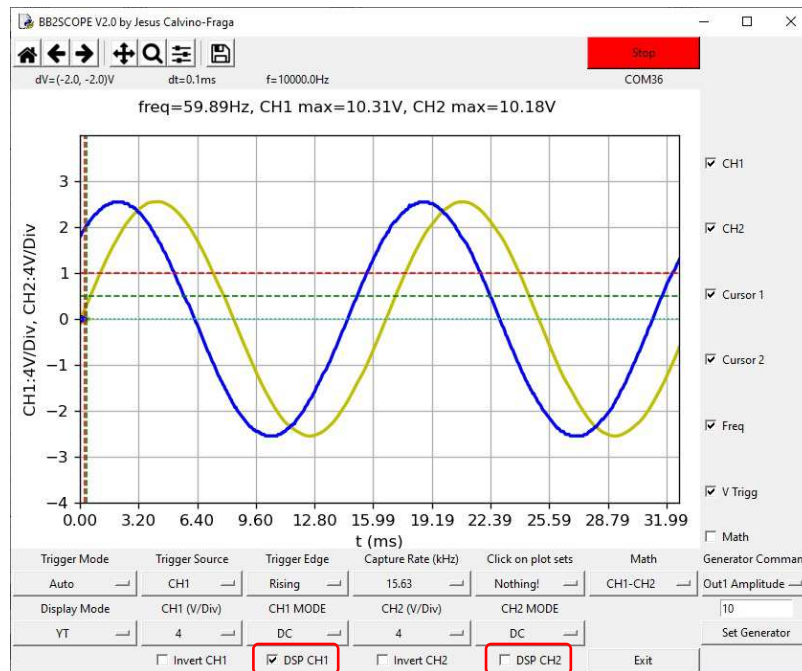


The ‘Click on plot sets’ option menu tells the BB2Scope script what happens when the plot display screen is clicked on:

1. If the option ‘**V trigg**’ is selected, clicking the screen sets the trigger voltage level. The trigger voltage level is displayed as a thin cyan dotted line in the screen. (#1 red arrow in the figure above)
2. If the option ‘**Cursor 1**’ is selected, clicking the screen sets the cursor 1 position. The cursor 1 position is displayed as two intersecting thin red dashed lines in the screen. (#2 red arrow in the figure above)
3. If the option ‘**Cursor 2**’ is selected, clicking the screen set the cursor 2 position. The cursor 2 position is displayed as two intersecting thin green dashed lines in the screen. (#3 red arrow in the figure above)
4. If the option ‘**Zero CH1**’ is selected, clicking the screen sets the zero position for channel 1. The zero position for channel 1 is displayed as a yellow triangle marker in the left vertical axis (4, 5 arrow in the figure above).
5. If the option ‘**Zero CH2**’ is selected, clicking the screen sets the zero position for channel 2. The zero position for channel 2 is displayed as a blue triangle marker in the left vertical axis (4, 5 arrow in the figure above)
6. If the option “**Nothing!**” is selected, nothing happens when clicking the plot graph screen. This option must be selected when using the pan and zoom options from the Matplotlib tool menu pushbuttons. This option is selected automatically after a few seconds if nothing is clicked in the plot graph.

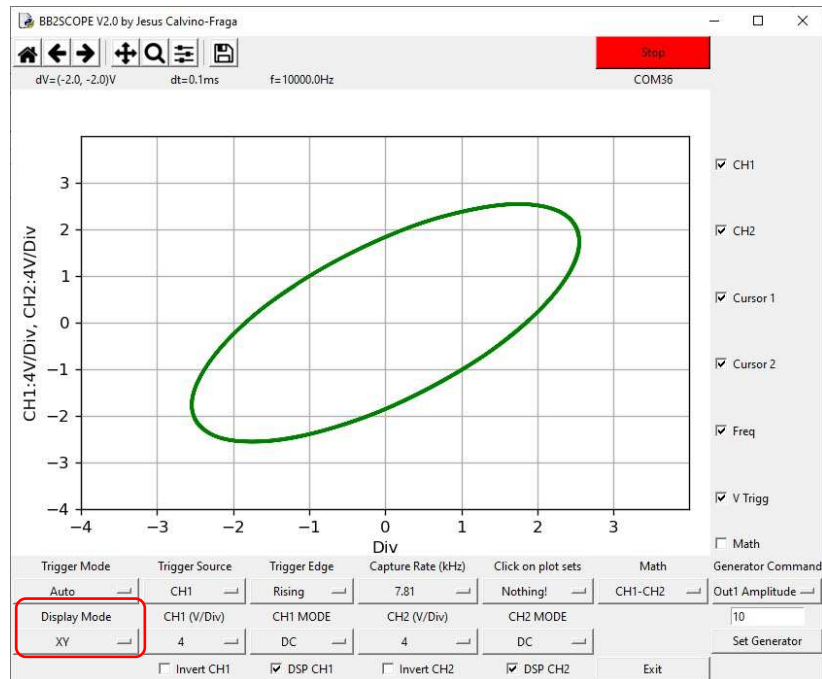
## DSP option (Low Pass Filter)

BB2Scope offers the option of applying Digital Signal Processing (DSP) to the channel traces before they are displayed. The DSP takes the form of a second order low pass Butterworth filter with a cutoff frequency  $1/10^{\text{th}}$  of the sampling frequency. The figure below shows the effect of enabling this option.



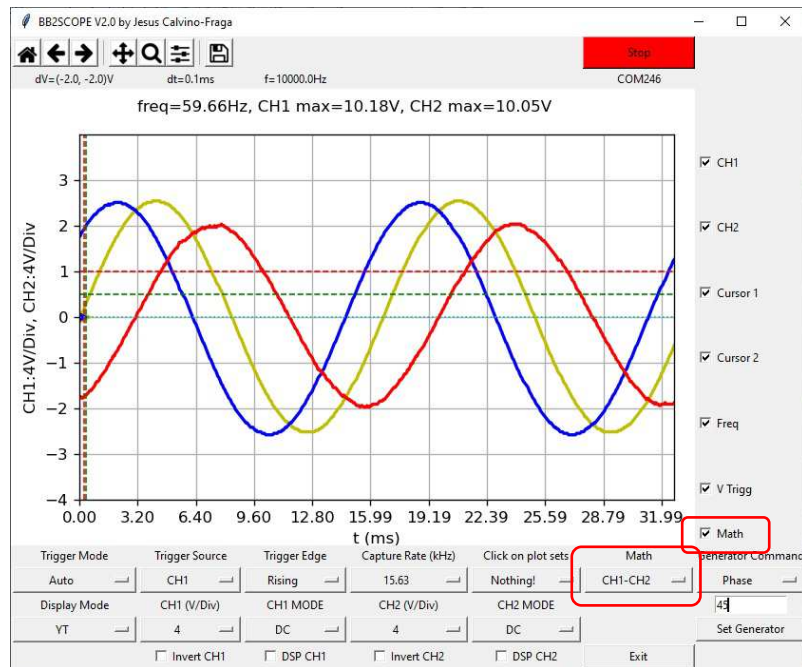
In the figure above a low pass filter is selected for CH1 by checking the tick box 'DSP CH1'. All the higher frequency noise is smoothed out as compared with the signal for CH2 which has 'DSP CH2' unselected. Care must be taken when using this option as it may distort the displayed waveform. This is particularly significant for signal with high harmonic content like square and pulse waves. The DSP option works best for sine waves that are displayed in the screen showing just a couple of complete periods.

## Parametric (XY) Plots



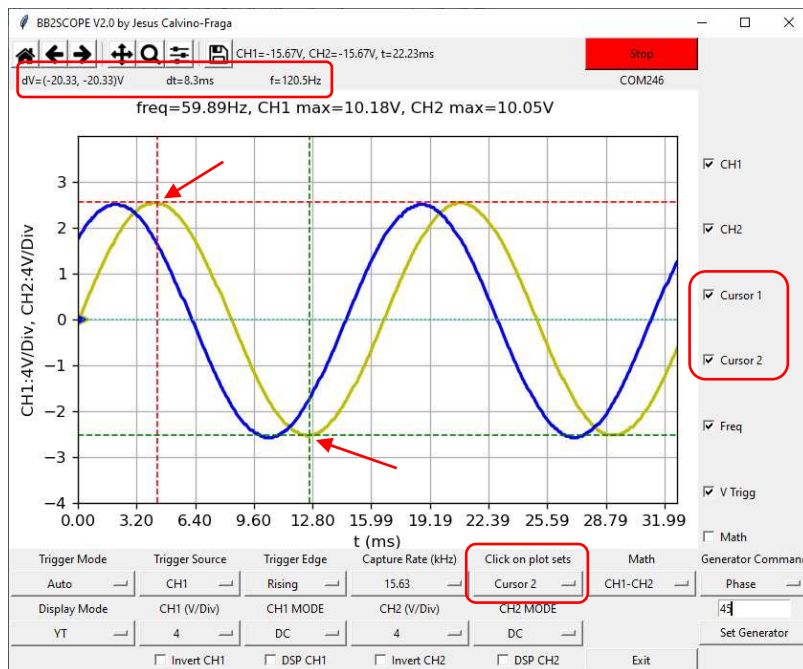
BB2Scope can display parametric figures when the ‘**Display Mode**’ selection button is set to either ‘**XY**’ or ‘**YX**’. In ‘**XY**’ mode channel 1 is used as the X axis input and channel 2 is used as the Y axis input. In ‘**YX**’ mode channel 2 is used as the X axis input and channel 1 is used as the Y axis input. For the example plot above both the CH1 and CH2 signals are sine waves with a 10 Vp amplitude @ 60Hz. The phase difference between the signals is 45 degrees. Both signals were generated using the Gen2BB board.

## Math Plot



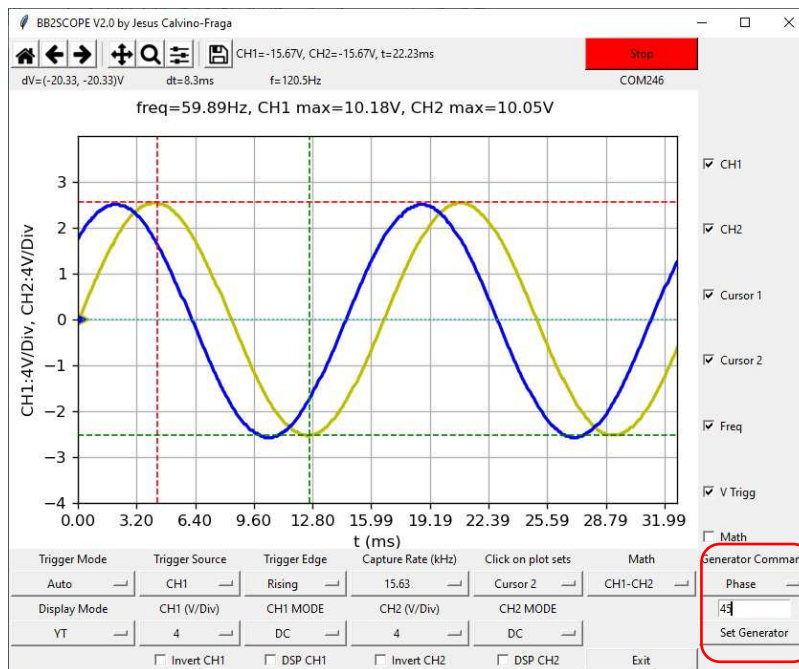
A third plot graph can be displayed in the screen which is the result of simple arithmetic operation with the signals CH1 and CH2. The math graph is displayed when the 'Math' tick box is selected on the left side of the screen. The 'Math' selection button allows for 'CH1-CH2', 'CH2-CH1', and "CH1+CH2". The math option is useful when a differential voltage wave form needs to be displayed.

## Using the Cursors for Differential Measurements



By using the 'Click on plot sets' selection box, cursors 1 and 2 can be positioned in the screen as desired. Underneath the Matplotlib toolbar the differential measurements are displayed: '**dV**' which shows the difference in voltage between cursor 1 and cursor 2 (the first number in parenthesis is **dV** for channel 1, the second number is **dV** for channel 2); '**dt**' which shows the difference in time between cursor 1 and cursor 2; and '**f**' which shows the inverse of '**dt**' when '**dt**' is not zero. The cursors can be hidden or displayed by selecting or deselecting the tick boxes on the right side of the screen.

## Controlling the outputs of the Gen2BB board

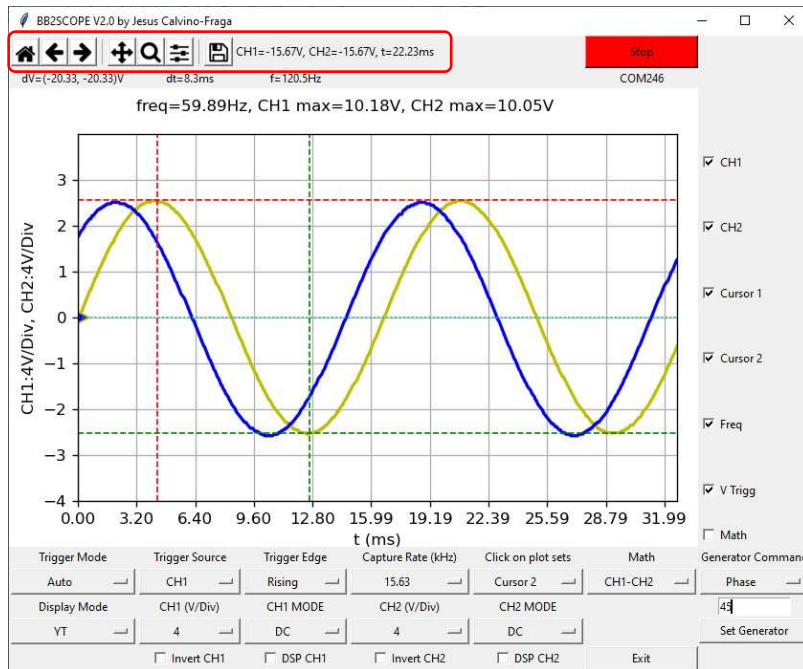


There are several commands that allow for the configuration of the Gen2BB board output. **For this to work the +TST- pins of the BB2Scope board MUST be connected to the +TST- pins of the Gen2BB board.** The commands are available after pressing the 'Generator Command' button. These commands include, for example, wave shape (sine, square, etc.), frequency, phase difference, amplitude, DC offset, etc. Some commands take a parameter, for example when setting the amplitude. The parameter is obtained from the input box below the 'Generator Command' button. Other commands don't require a parameter, for example when changing the wave shape. Once a command is selected and its parameter set, press the 'Set Generator' button to make the change. These are the commands that can be sent to the Gen2BB board:

- Generator Command
- Out1 Amplitude
  - Out2 Amplitude
  - Out1 Offset
  - Out2 Offset
  - Frequency
  - Phase
  - Out1 Sine
  - Out2 Sine
  - Out1 Square
  - Out2 Square
  - Out1 Triangle
  - Out2 Triangle
  - Out1 Ramp
  - Out2 Ramp
  - Save
  - Restore
  - On
  - Off
  - None

The 'Save' command will set the current configuration of the Gen2BB board as the power-on default. The 'Restore' command will restore the Gen2BB to its power-on default configuration. The 'Set Generator' button must be pressed in order for the commands to take effect. The 'factory' power-on setup of the Gen2BB is set to: both outputs as sine waves, the amplitude is 10 V<sub>peak</sub>, the frequency is 60 Hz, and the phase difference is 45 degrees.

## The Matplotlib Navigation Toolbar



The navigation toolbar on the top left of the screen is a standard part of the Matplotlib used by Python. A detailed description of its operation is available by following this link:

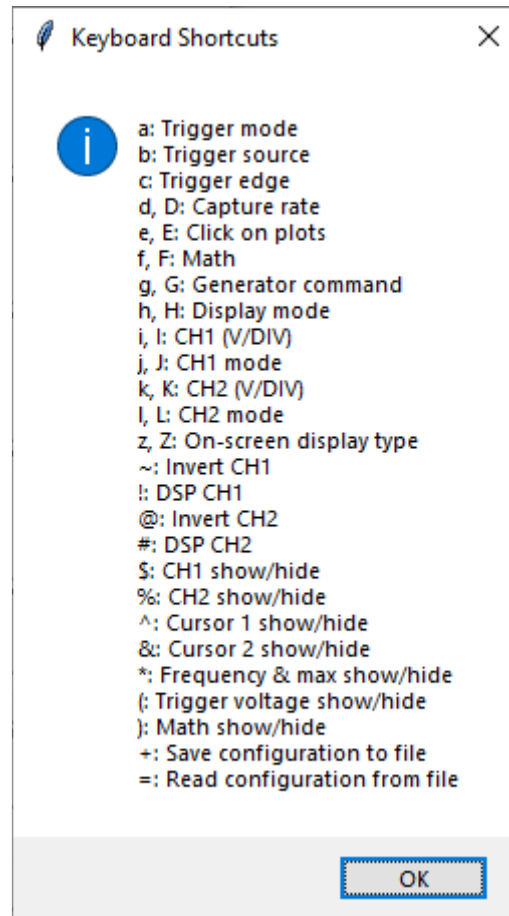
[https://matplotlib.org/3.1.1/users/navigation\\_toolbar.html](https://matplotlib.org/3.1.1/users/navigation_toolbar.html)

The displayed numerical values have been overridden so that the position of the arrow cursor shows the voltages of both channels and the time instead of plain X-Y coordinates as in the original navigation toolbar.



## Keyboard Shortcuts

Every BB2Scope command is available via single key shortcut. The shortcuts are arranged, more or less, in alphabetical order. To see the keyboard shortcut assignments either hover with mouse over a pushbutton or tick box or press key '?'. After pressing '?', a list of shortcuts is displayed as shown in the figure below. For option menu buttons, the lower case version of the shortcut selects the next option in the menu while the upper case version selects the previous option in the menu.

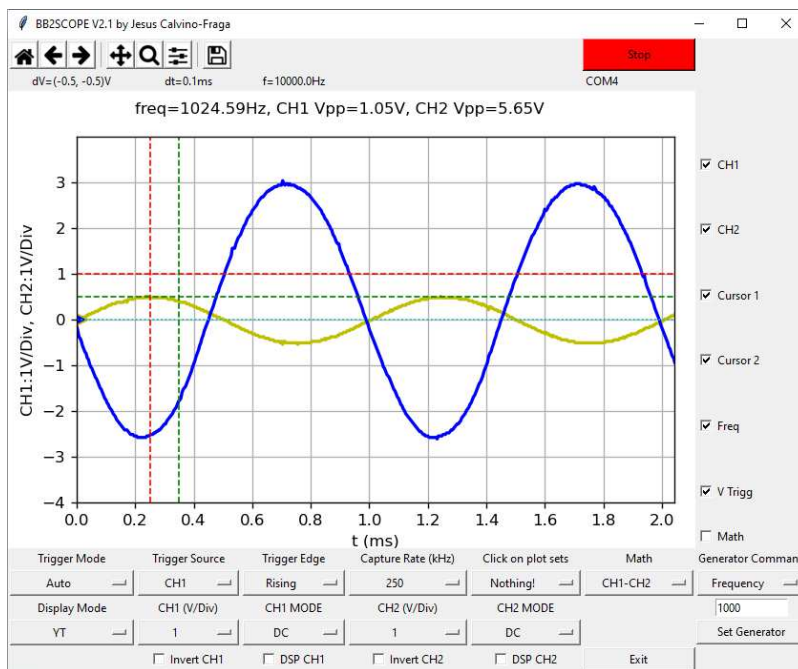


## On Screen Voltage Display

The keyboard shortcuts 'z' and 'Z' can be used to change the type of on-screen displayed voltages. These are the options available:

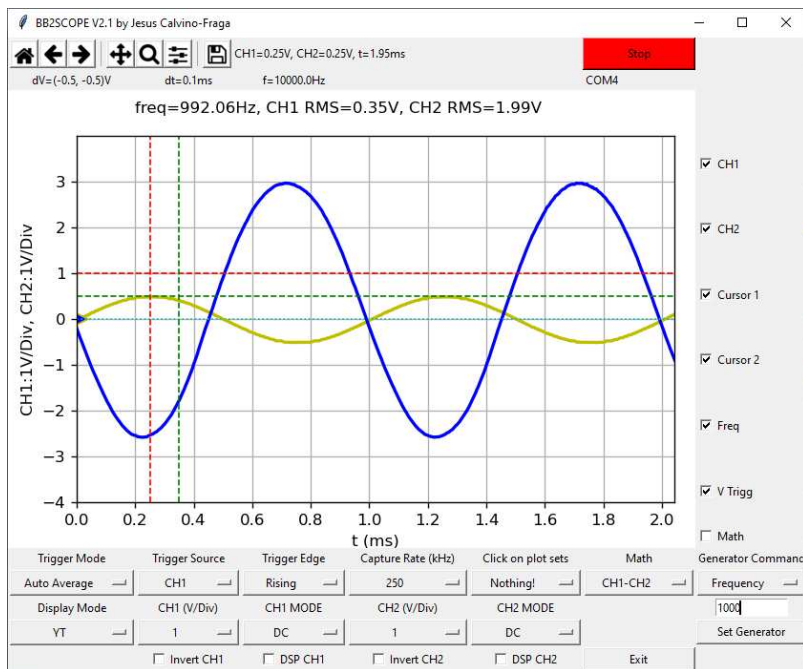
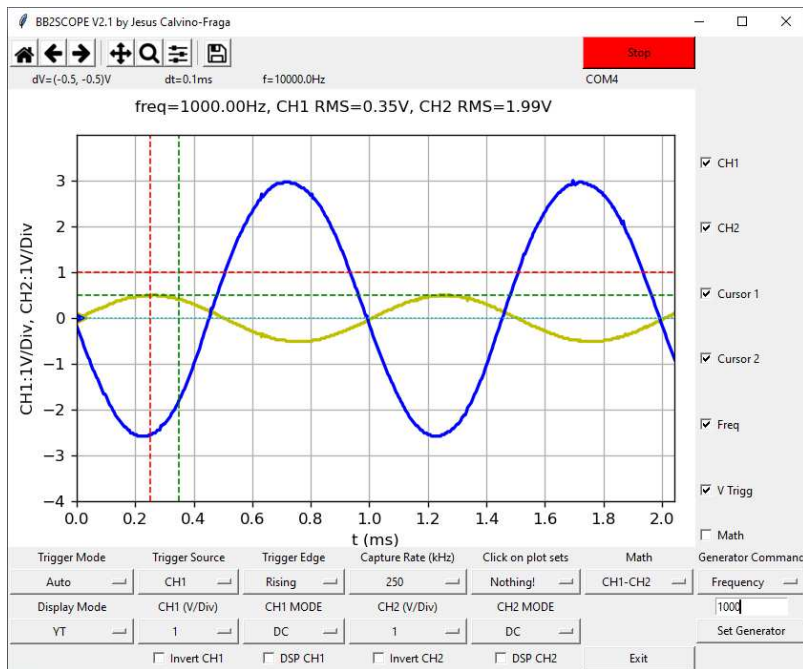
1. 'max': Display the maximum voltage of a waveform.
2. 'min': Display the minimum voltage of a waveform.
3. 'Vpp': Display the peak-to-peak voltage of a waveform.
4. 'Ave': Display the average voltage of a waveform.
5. 'RMS': display the root-mean-squared voltage of a waveform.

The figure below shows the on screen voltages displaying the peak-to-peak voltages for both channels. Every time 'z' is typed, the displayed voltage type changes on the screen.



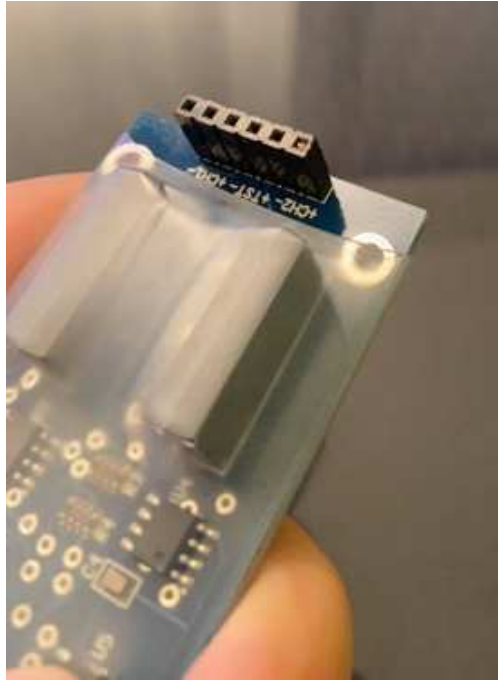
## Auto Trigger with Moving Average

For periodic signals that can trigger capture consistently it is possible to use the 'Auto Average' trigger mode. In this mode a moving average of the captured waveforms is displayed. This mode greatly improves the appearance of noisy signals as shown in the two screen captures below. The first screen capture uses the regular 'Auto' mode. The second screen used the 'Auto Average' mode; notice how all the small glitches are averaged out as more samples of the waveform are captured.



## ***Removing Broken Wires from the Connectors***

Occasionally a hook-up wire may break inside one of the connectors on the BB2Scope or Gen2BB boards as shown in the figure below.

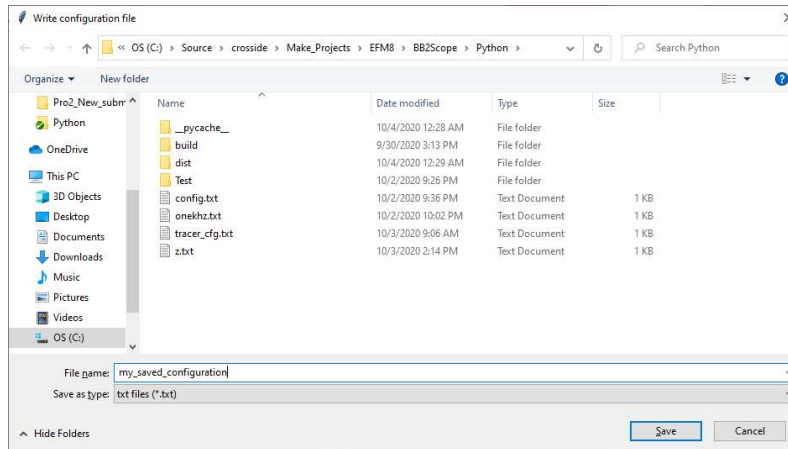


To remove the broken wire, use a blade to carefully lift the plastic housing of the connector without breaking it or bending its pins. For example a pocket knife can be used to wedge the housing out. It takes just a few seconds, be careful. With the metal contacts exposed remove the broken wire. Finally, put the plastic cover back and press firmly in place. The figure below shows the connector of the BB2Scope with the plastic housing removed.

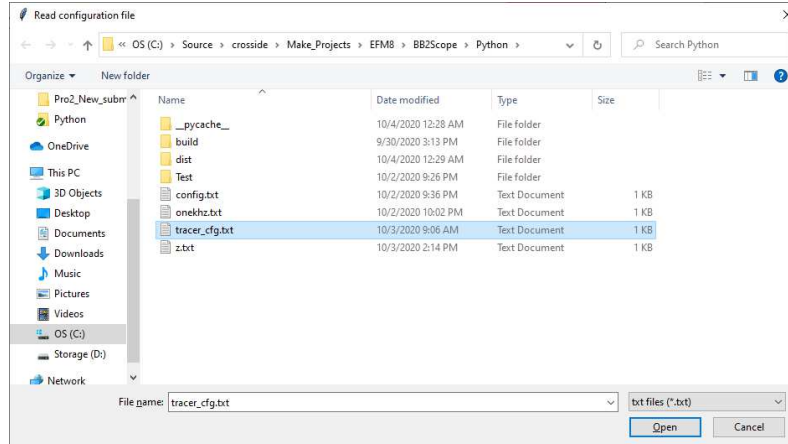


## ***Saving and Retrieving the BBScope configuration***

BB2Scope allows saving/retrieving its current configuration setup to/from a file stored in your computer. This feature is accessible via the key shortcuts '+' and '='. To save the current configuration setup to a file press the '+' key. A 'save' popup window is displayed:



Similarly, to load the configuration from a previously saved file, press the '=' key:

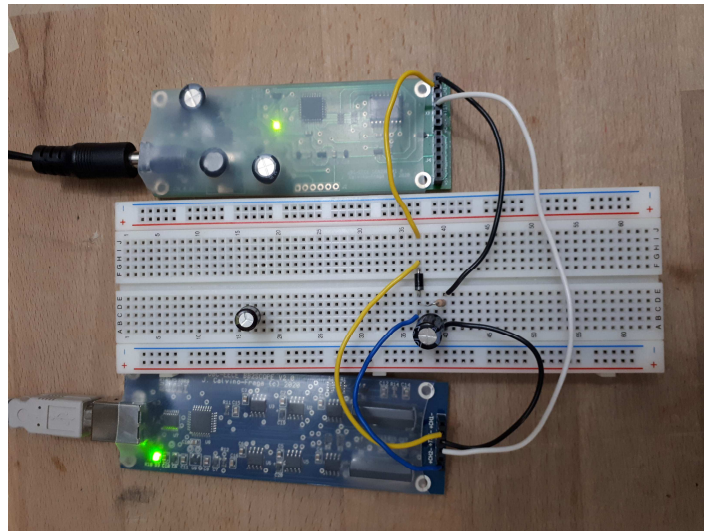


## Examples

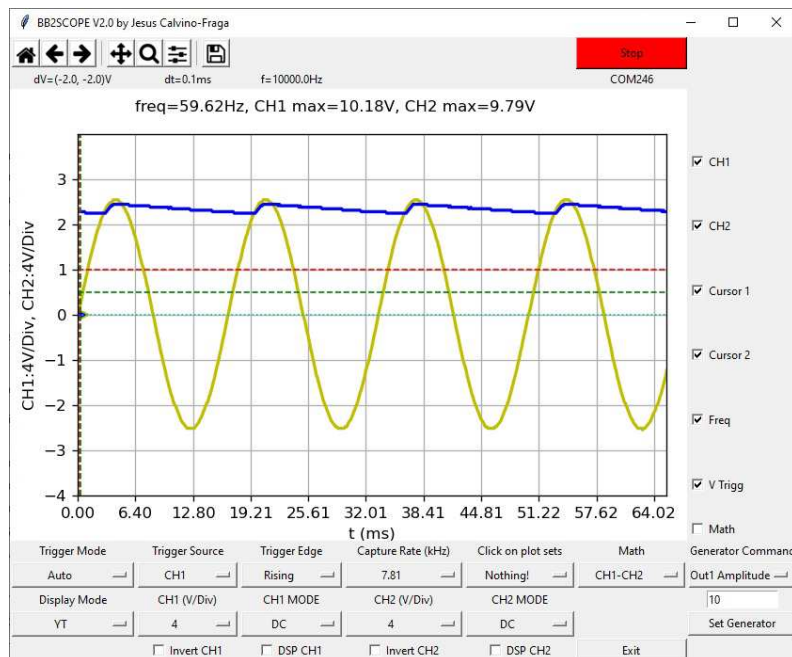
In order to illustrate practical uses for the BB2Scope and Gen2BB boards several examples are included in the sections that follow.

### *Half Wave Rectifier*

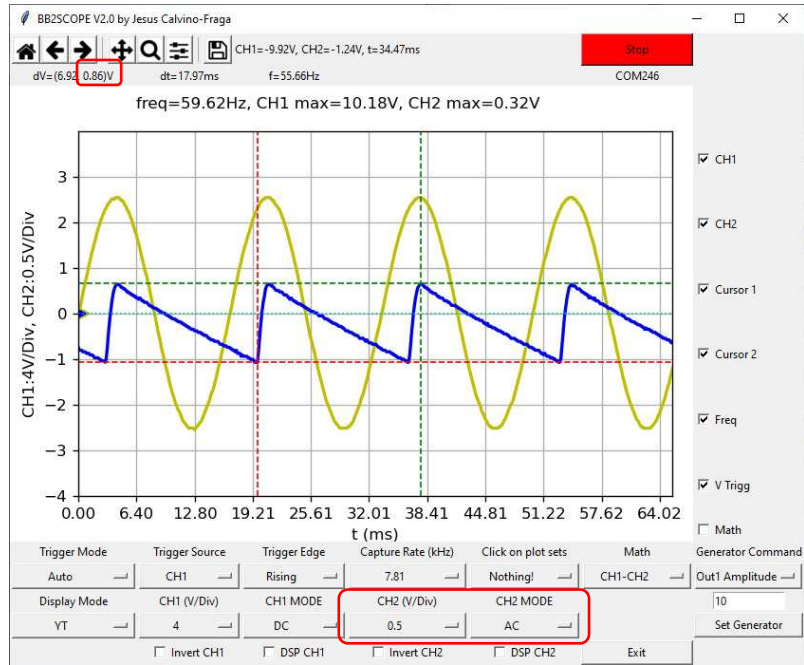
The figure below shows a bread-boarded half-wave rectifier circuit with filter capacitor. The diode is the MBR150, the electrolytic capacitor value is  $100\mu\text{F}$  (50V), and the resistor is  $2\text{k}\Omega$ . Output 1 of the Gen2BB board is used as the AC input for the circuit ( $10\text{Vp}$  @  $60\text{Hz}$ ).



BB2Scope's CH1 is used to display the input wave form, while CH2 is used to display the output waveform. The results are displayed in the figure below.



It can be observed that the output waveform in CH2 is the sum of a DC voltage and an AC ripple voltage. We are often interested on measuring accurately the ripple voltage. To perform this measurement we change the coupling of CH2 via the '**CH2 MODE**' button from '**DC**' to '**AC**'. Next, we adjust the CH2 gain to amplify the ripple voltage as needed, and use the cursors to measure the differential voltage as shown in the figure below.



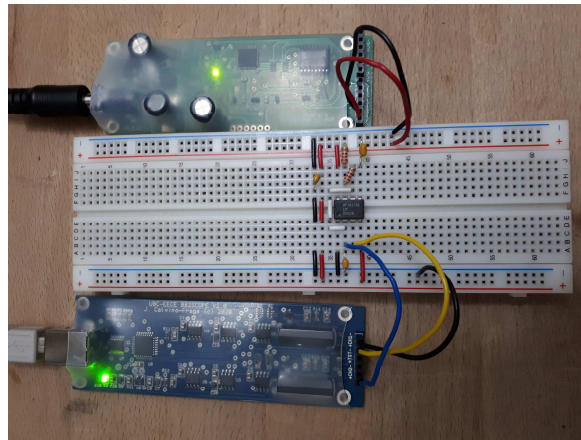
As we can observe from the figure above, the ripple voltage is approximately 0.86V peak to peak.



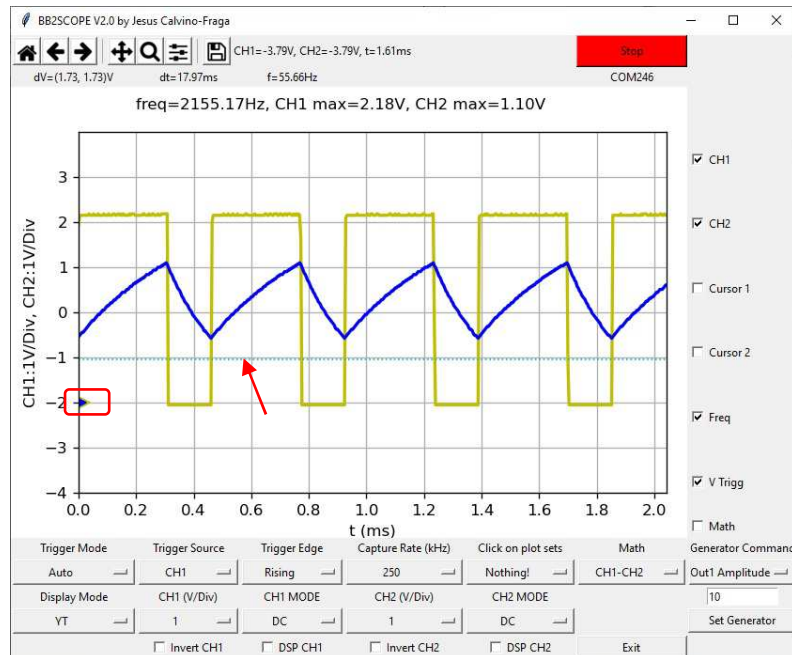
## 555 Timer A-stable Oscillator

The picture of the bread-boarded circuit below shows a 555 timer configured as an A-stable oscillator ( $R_1=R_2=2.2\text{k}\Omega$ ;  $C=0.1\mu\text{F}$ ). The power for the 555 timer IC (approximately 5V) is obtained from the Gen2BB board +5V output. The schematic diagram of this circuit can be found here:

<https://ohmslawcalculator.com/555-astable-calculator>



The figure below was obtained while the 555 timer A-stable oscillator was connected to BB2Scope board. CH1 shows the output of the timer (IC pin 3), while CH2 shows the voltage at the timing capacitor (IC pin 2).

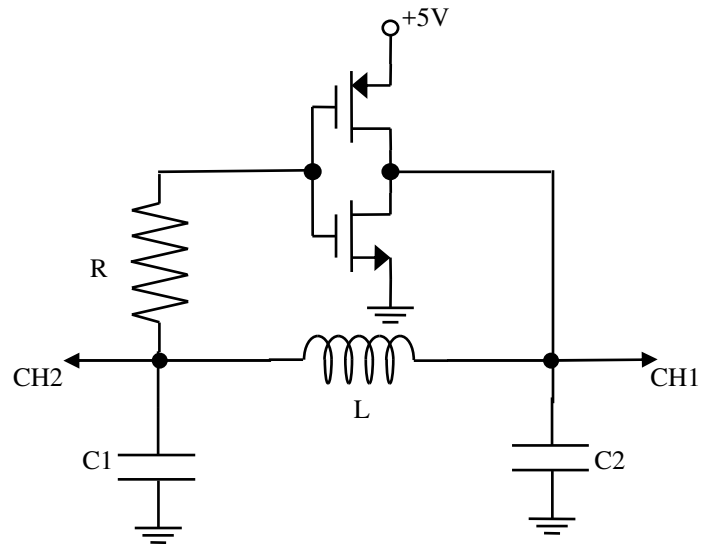


The zero position for both CH1 and CH2 channels have been adjusted in order to take advantage of the dynamic range of the scope while the gain of both channels is set to 1 V/Div. Notice also that the trigger level has been moved to in between the voltage levels of the CH1 signal; this is done in order to obtain a steady plot graph.

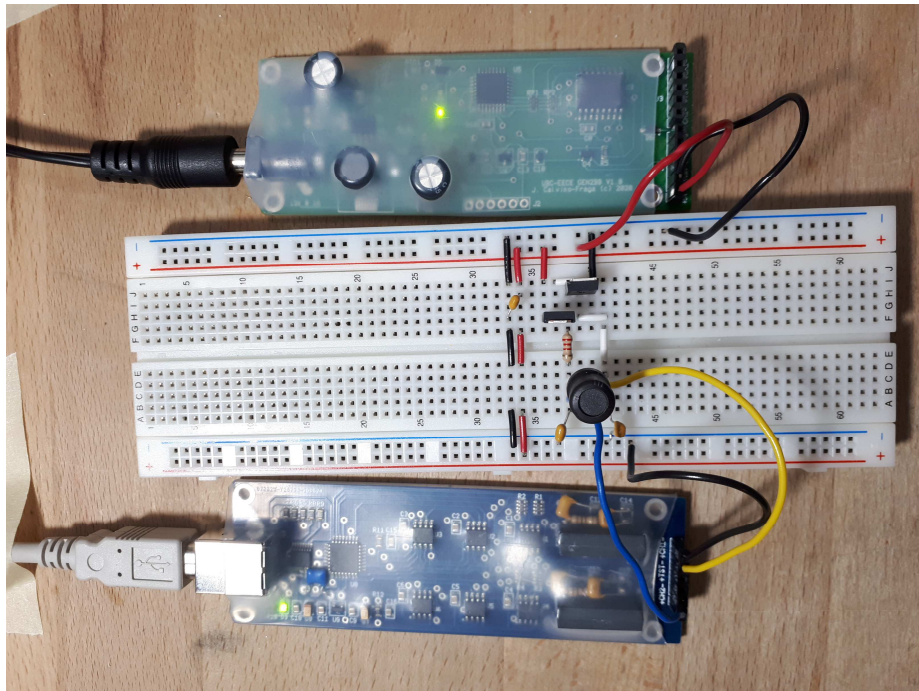


## Colpitts Oscillator

The picture below shows the circuit diagram of a Colpitts oscillator using a discrete CMOS inverter. For the circuit shown  $C1=C2=0.1\mu\text{F}$ ;  $L=75\text{mH}$ ;  $R=2.2\text{k}\Omega$ . The PMOS transistor is the NTD2955-1G and the NMOS transistor is the NTD3055L104-1G. The discrete CMOS gate is powered with the +5V output of the Gen2BB board.



The picture below shows the bread-boarded implementation of the circuit above.



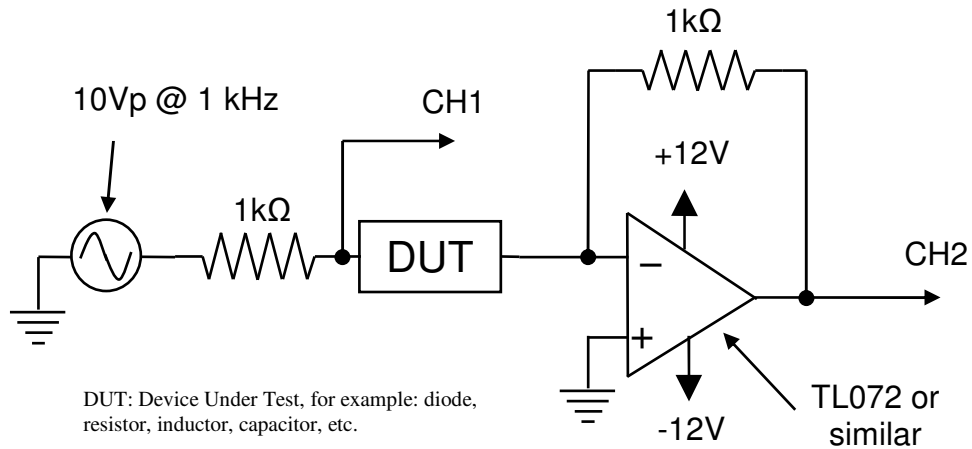
The figure below shows the voltage at the output of the CMOS inverter using CH1 and the voltage at the resistor before the input of the CMOS inverter using CH2.



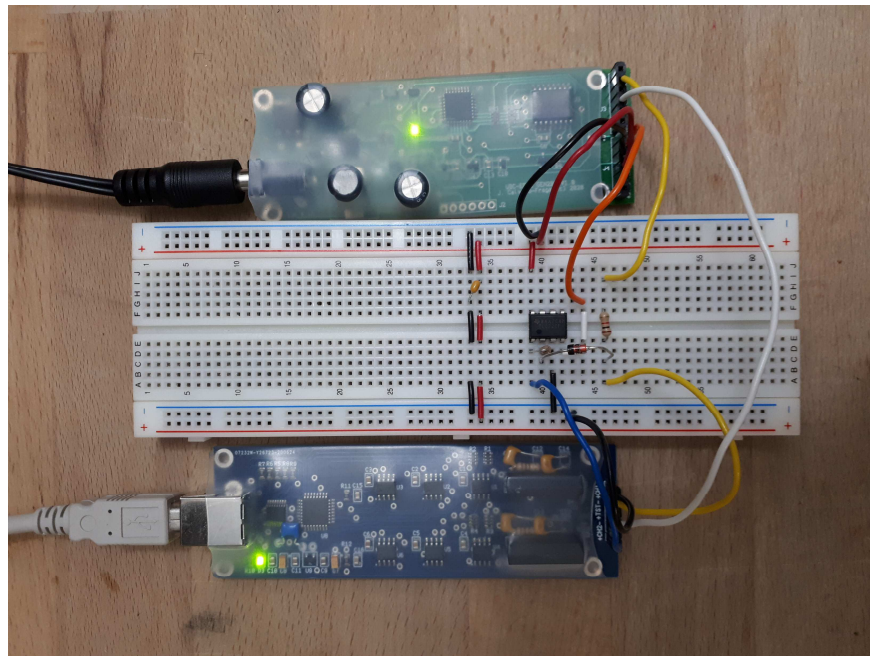
From the figure above we can see that the circuit is oscillating at around 2873 Hz and that we obtain a clean and undistorted sine wave.

## I vs. V Curve Tracer

A simple I vs. V parametric curve tracer can be implemented using the following circuit diagram:

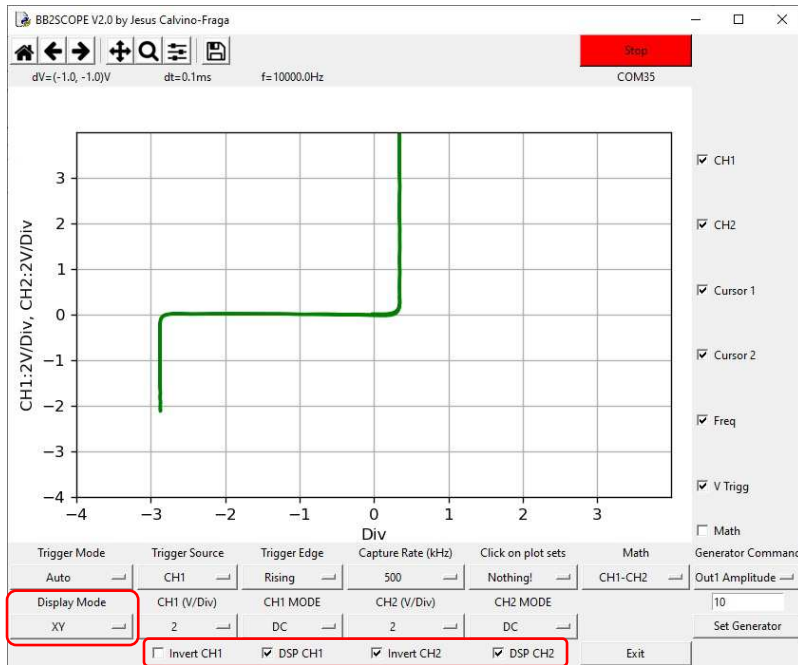


The 10Vp @ 1 kHz sine wave as well as the +12V and -12V power supplies are provided by the Gen2BB board. CH1 is used to display the voltage across the DUT, while CH2 is used to display the current through the DUT. The operational amplifier is configured as an inverting current to voltage converter: 1.0 mA through the DUT is displayed in CH2 as -1.0V. The assembled bread-boarded circuit is shown in the figure below with a 1N4734A zener diode as the DUT.



The figure below shows the 'I vs. V' parametric curve for a 1N4734A zener diode when the 'Display Mode' is changed to 'XY'. Both the forward voltage drop of around 0.7V and the reverse zener voltage of around 5.8V can be clearly observed. Notice that the polarity of CH2 has been inverted. This is required

because the current to voltage converted implemented in the circuit above is inverting ( $1.0 \text{ mA} \rightarrow -1.0 \text{ V}$ ). Also the DSP option for both channels is selected which results in a cleaner graph plot.



# Running BB2Scope on Different Operating Systems

## Running BB2Scope on Linux

The BB2Scope.pyw was verified to run on Debian Linux LXDE. The full description of the Linux system used is:

Linux VBdebian 4.9.0-6-amd64 #1 SMP Debian 4.9.88-1+deb9u1 (2018-05-07) x86\_64 GNU/Linux  
Desktop: LXDE  
Session: lightdm-xsession

You must be a member of group 'dialout' in order to access the 'USB to serial' device that communicates with the BB2Scope board. Something like this may do the job:

```
$ sudo gpasswd --add ${USER} dialout
```

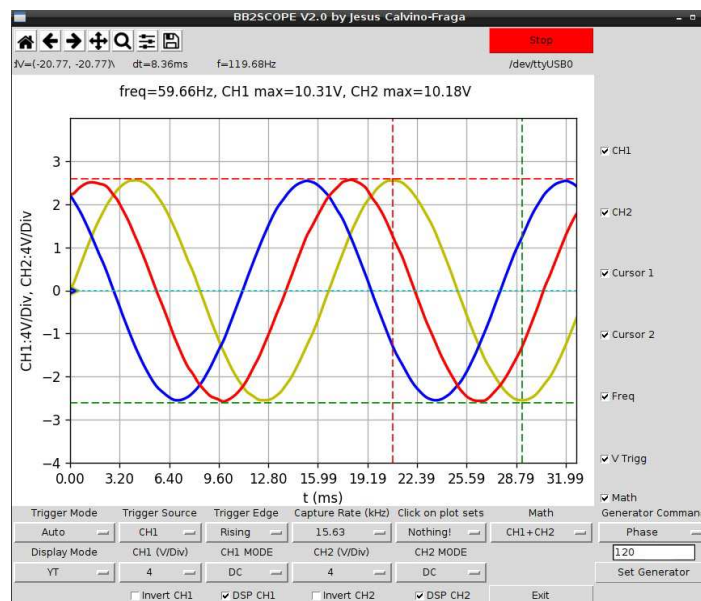
To install the required Python3 packages on Debian Linux you can try using apt-get. In other distributions of Linux you can try the standard Python package manager 'pip'.

```
$ sudo apt-get update  
$ sudo apt install python3-serial  
$ sudo apt-get install python3-matplotlib  
$ sudo apt-get install python3-scipy
```

To start the script:

```
$ python3 BB2Scope.pyw
```

The figure below shows the 'BB2Scope.pyw' script running on Linux. Notice that the communication with the BB2Scope board is through '/dev/ttyUSB0'. On Linux there is no need to install the USB driver as it is built into the Linux kernel.



## Running BB2Scope on macOS

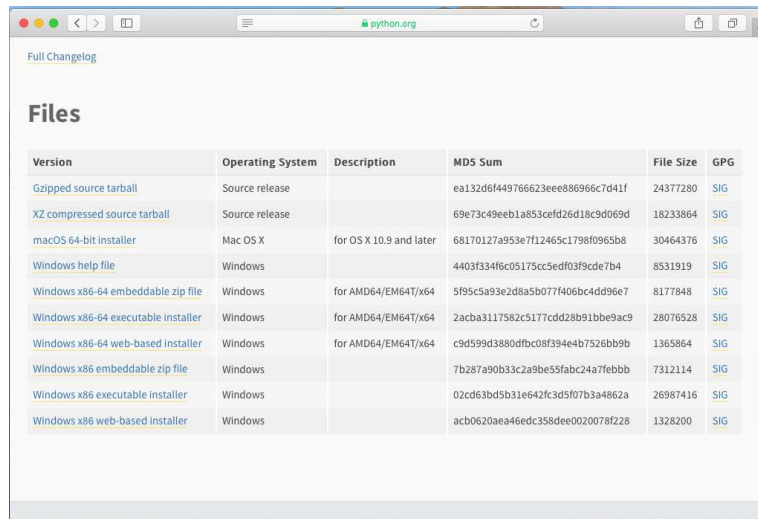
The “BB2Scope.pyw” has been verified to run on macOS 10.15 Catalina. One problem that you may encounter is that USB serial port support seems to be broken on some releases of this OS. If the computer is not connecting to the BB2Scope board, you may have to install the macOS VCP drivers from here:

<https://www.ftdichip.com/Drivers/VCP.htm>

The BB2Scope board was connected to two different computers running macOS 10.15 and both seemed to work properly with the built in USB driver. The script was verified to work also with macOS 10.11 El Capitan on a Mac Book Pro from 2010.

Python 2.7 is included with macOS 10.15 Catalina. This version of Python is deprecated and will not work with the “BB2Scope.pyw” script. There are many web pages that explain how to install Python3 for macOS in somehow convoluted ways. I just downloaded the latest release of Python 3 for MacOS from the official Python link below and installed it by clicking on the downloaded file.

<https://www.python.org/downloads/mac-osx/>



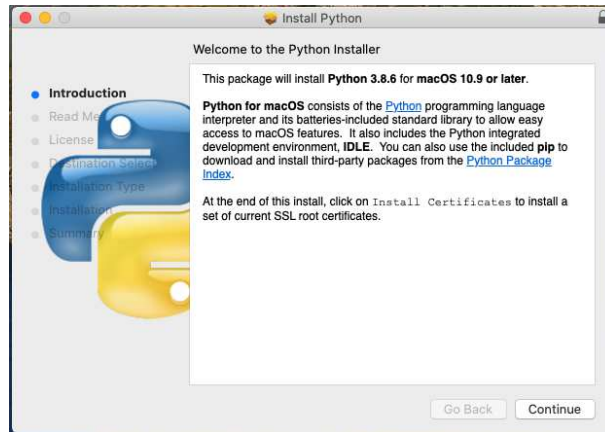
Version	Operating System	Description	MD5 Sum	File Size	GPG
<a href="#">Gzipped source tarball</a>	Source release		ea132d6f449766623eee886966c7d41f	24377280	<a href="#">SIG</a>
<a href="#">XZ compressed source tarball</a>	Source release		69e73c49eeb1a853cfd26d18c9d069d	18233864	<a href="#">SIG</a>
<a href="#">macOS 64-bit installer</a>	Mac OS X	for OS X 10.9 and later	68170127a953e7f12465c1798f0965b8	30464376	<a href="#">SIG</a>
<a href="#">Windows help file</a>	Windows		4403f34f6c05175cc5edf03f9cde7b4	8531919	<a href="#">SIG</a>
<a href="#">Windows x86-64 embeddable zip file</a>	Windows	for AMD64/EM64T/x64	5f95c5a93e2d8a5b077f406bc4dd96e7	8177848	<a href="#">SIG</a>
<a href="#">Windows x86-64 executable installer</a>	Windows	for AMD64/EM64T/x64	2acba3117582c5177cdd28b91bbe9ac9	28076528	<a href="#">SIG</a>
<a href="#">Windows x86-64 web-based installer</a>	Windows	for AMD64/EM64T/x64	c9d599d3880dfbc08f394e4b7526bb9b	1365864	<a href="#">SIG</a>
<a href="#">Windows x86 embeddable zip file</a>	Windows		7b287a90b33c2a9be55fab24a7febbb	7312114	<a href="#">SIG</a>
<a href="#">Windows x86 executable installer</a>	Windows		02cd63bd5b31e642fc3d5f07b3a4862a	26987416	<a href="#">SIG</a>
<a href="#">Windows x86 web-based installer</a>	Windows		acb0620aea46edc358dee0020078f228	1328200	<a href="#">SIG</a>

The link you are looking for is “macOS 64-bit installer”. In my case it downloaded “python-3.8.6-macosx10.9.pkg”.



Click the downloaded package to install.





After the installation of Python 3, the following required packages need to be installed using pip via a 'Terminal' window:

```
% pip3 install pyserial
% pip3 install matplotlib
% pip3 install scipy
```

This is how it looked like after installing the package "pyserial".

```
suso@susos-iMac ~ % pip3 install pyserial
Collecting pyserial
  Downloading pyserial-3.4-py2.py3-none-any.whl (193 kB)
    | 193 kB 2.4 MB/s
Installing collected packages: pyserial
Successfully installed pyserial-3.4
WARNING: You are using pip version 20.2.1; however, version 20.2.3 is available.
You should consider upgrading via the '/Library/Frameworks/Python.framework/Versions/3.8/bin/python3.8 -m pip install --upgrade pip' command.
suso@susos-iMac ~ %
```

The procedure is the same for the other two packages. (As suggested in the yellow warning the pip installer was upgraded as well, but this is not strictly necessary)

```
suso@susos-iMac ~ % pip3 install matplotlib
Collecting matplotlib
  Downloading matplotlib-3.3.2-py3.8-cp38-cp38-macosx_10_9_x86_64.whl (28.9 MB)
    | 28.9 MB 18.6 MB/s
Requirement already satisfied: numpy>=1.14.5 in /Library/Frameworks/Python.framework/Versions/3.8/lib/python3.8/site-packages (from matplotlib) (1.19.2)
Installing collected packages: matplotlib
Successfully installed matplotlib-3.3.2

suso@susos-iMac ~ % pip3 install scipy
Collecting scipy
  Downloading scipy-1.5.2-cp38-cp38-macosx_10_9_x86_64.whl (28.9 MB)
    | 28.9 MB 18.6 MB/s
Requirement already satisfied: numpy>=1.14.5 in /Library/Frameworks/Python.framework/Versions/3.8/lib/python3.8/site-packages (from scipy) (1.19.2)
Installing collected packages: scipy
Successfully installed scipy-1.5.2
suso@susos-iMac ~ %
```

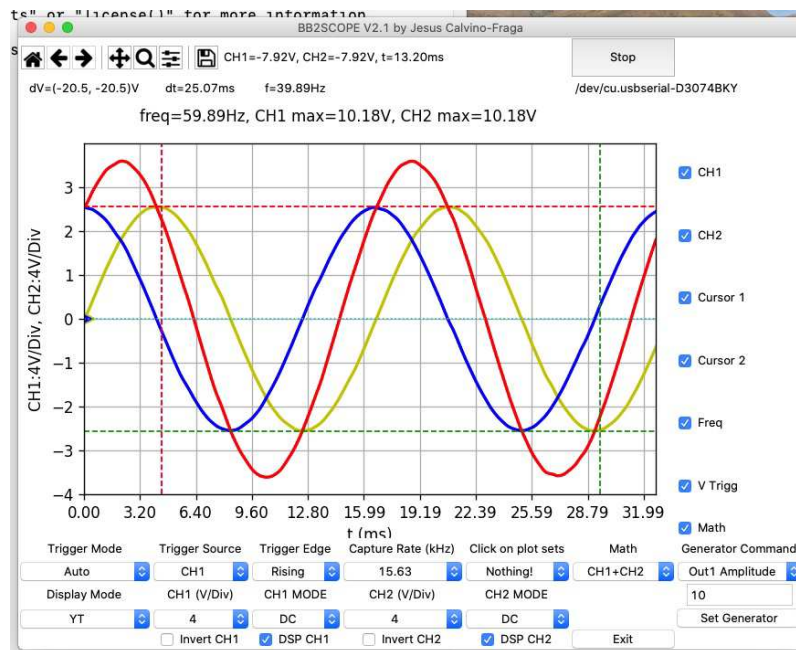
Finally, “BB2Scope.pyw” was started using the command shown below in the same ‘Terminal’ window.

**% python3 BB2Scope.pyw**

```
Work - Python BB2Scope.pyw - 80x24
Requirement already satisfied: numpy>=1.14.5 in /Library/Frameworks/Python.framework
work/Versions/3.8/lib/python3.8/site-packages (from scipy) (1.19.2)
Installing collected packages: scipy
Successfully installed scipy-1.5.2
suso@susos-iMac ~ % ls -l
total 0
drwx-----+ 3 suso  staff   96 25 Sep 19:55 Desktop
drwx-----+ 3 suso  staff   96 25 Sep 19:55 Documents
drwx-----+ 4 suso  staff  128 27 Sep 12:16 Downloads
drwx-----@ 55 suso  staff 1760 27 Sep 12:06 Library
drwx-----+ 4 suso  staff  128 27 Sep 11:56 Movies
drwx-----+ 4 suso  staff  128 27 Sep 11:57 Music
drwx-----+ 4 suso  staff  128 27 Sep 12:06 Pictures
drwxr-xr-x+ 4 suso  staff  128 25 Sep 19:55 Public
drwxr-xr-x 3 suso  staff   96 27 Sep 11:53 Work
suso@susos-iMac ~ % cd Work
suso@susos-iMac Work % ls -l
total 80
-rwxr-x--- 1 suso  staff 38055 27 Sep 11:53 BB2Scope.pyw
suso@susos-iMac Work % python3 BB2Scope.pyw
Unable to create basic Accelerated OpenGL renderer.
Unable to create basic Accelerated OpenGL renderer.
Core Image is now using the software OpenGL renderer. This will be slow.
```

If you double click the “BB2Scope.pyw” script in ‘Finder’ the Python editor ‘IDLE’ opens it. You can run the script from ‘IDLE’ by clicking on the ‘Run’ menu.

The result of running the “BB2Scope.pyw” script is shown below. For this particular BB2Scope board, the communication is through ‘/dev/cu.usbserial-D3074BKY’. (This name will be different but similar for each BB2Scope board.) The port name turned out to be a bit wider that the name for the serial port given in both the Windows and Linux operating systems, so the space used to display this information was increased in version 2.1 of the “BB2Scope.pyw” script.





## Running BB2Scope on Windows using the Official Python Distribution

The “BB2Scope.pyw” has been verified to run on Windows 7, 8.1, and 10. The driver for BB2Scope board is normally installed automatically after connecting it to the computer. If the computer is not connecting to the BB2Scope board, you may have to install the Windows VCP drivers manually from here:

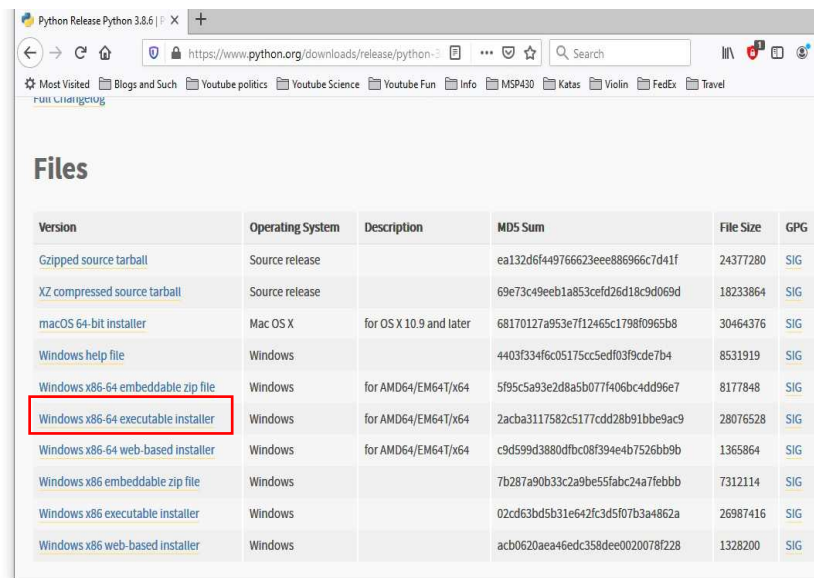
<https://www.ftdichip.com/Drivers/VCP.htm>

In order to run the “BB2Scope.pyw” in your Windows computer you’ll need Python 3.x installed. This is an alternative method of installing Python 3 instead of using WinPython described earlier.

**WARNING: On Windows 10 Python 3 can be ‘installed’ using the ‘Windows Store’. DON’T USE THIS METHOD. It may mess up your Python installation.**

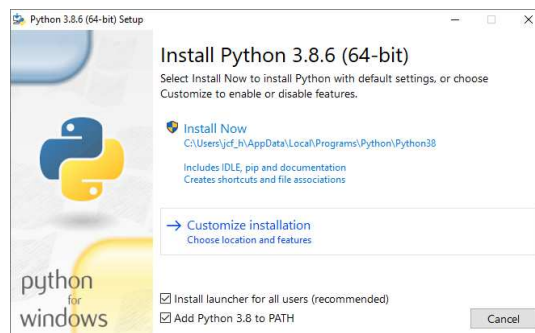
The first step is to download the Python installer from the official web site:

<https://www.python.org/downloads/windows/>



Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		ea132d6f44976623eee886966c7d41f	24377280	SIG
XZ compressed source tarball	Source release		69e73c49eeb1a853cfd26d18c9d069d	18233864	SIG
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	68170127a953e7f12465c1798f0965b8	30464376	SIG
Windows help file	Windows		4403f34f6c05175cc5edf03f9cde7b4	8531919	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	5f95c5a93e2d8a5b077f406bc4dd96e7	8177848	SIG
<b>Windows x86-64 executable installer</b>	Windows	for AMD64/EM64T/x64	2acba3117582c5177cdd28b91bbe9ac9	28076528	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	c9d599d3880dfbc08f394e4b7526bb9b	1365864	SIG
Windows x86 embeddable zip file	Windows		7b287a90b33c2a9be55fabc24a7febbb	7312114	SIG
Windows x86 executable installer	Windows		02cd63bd5b31e642fc3d5f07b3a4862a	26987416	SIG
Windows x86 web-based installer	Windows		acb0620aea46edc358dee0020078f228	1328200	SIG

The file downloaded at the time of writing this document was “python-3.8.6-amd64.exe”. Run the executable file just downloaded. Make sure to click ‘Add Python 3.8 to PATH’:



Wait for the installer to finish. Now open a command window and install the required packages using pip:

```
cmd.exe - Shortcut
Microsoft Windows [Version 10.0.18363.1082]
(c) 2019 Microsoft Corporation. All rights reserved.

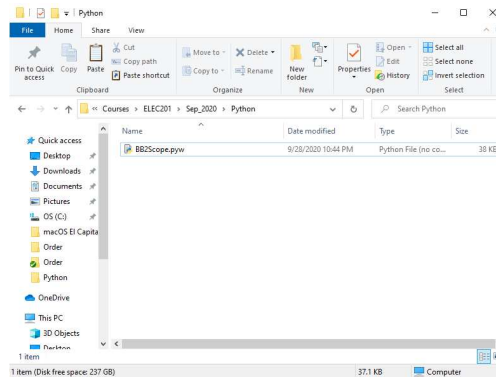
C:\Windows\System32>pip install pyserial
Collecting pyserial
  Using cached pyserial-3.4-py2.py3-none-any.whl (193 kB)
Installing collected packages: pyserial
Successfully installed pyserial-3.4
WARNING: You are using pip version 20.2.1; however, version 20.2.3 is available.
You should consider upgrading via the 'c:\programs\python38\python.exe -m pip install --upgrade pip' command.

C:\Windows\System32>
```

(Notice that I changed the installation folder from the suggested default, to 'c:\programs\python38', but that is not necessary.). You'll need to install the packages 'pyserial', 'matplotlib', and 'scipy':

```
C:\Windows\System32> pip install pyserial
C:\Windows\System32> pip install matplotlib
C:\Windows\System32> pip install scipy
```

Now, navigate to your "BB2Scope.pyw" script on Windows Explorer and double click it:



The program starts as displayed below, assuming the given signals are applied to both CH1 and CH2:

