AWESEM Acquisition Testbench

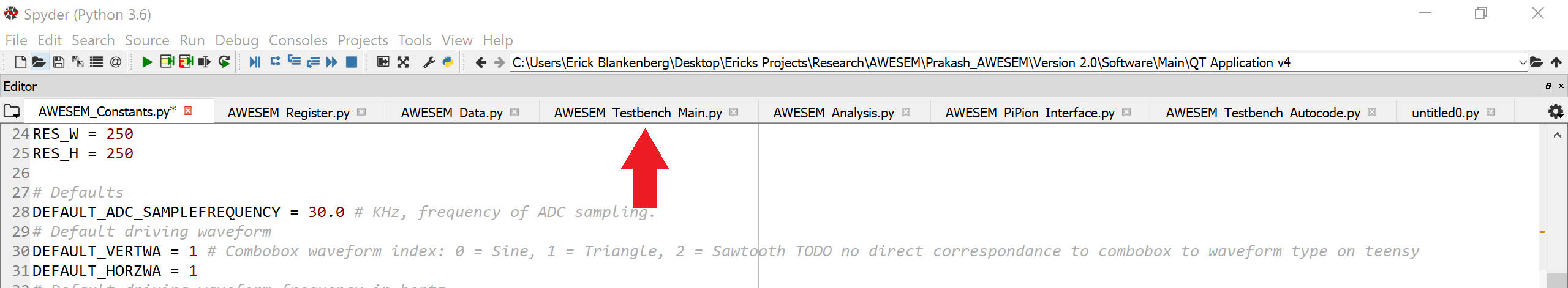
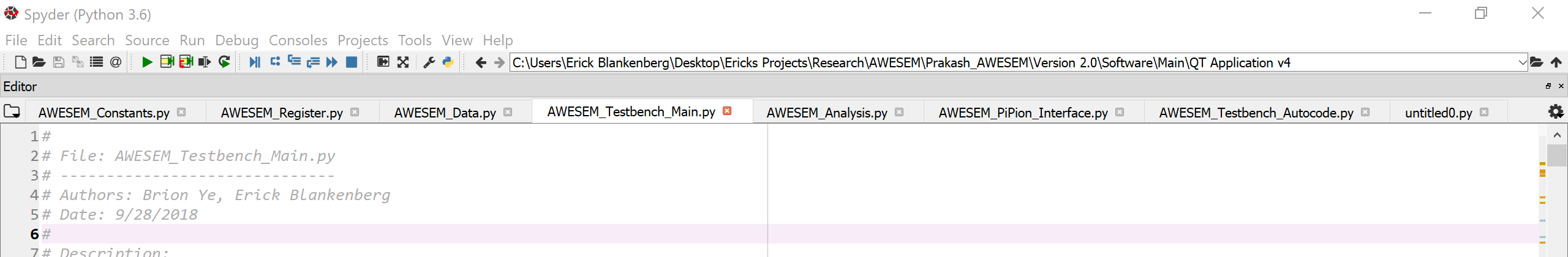
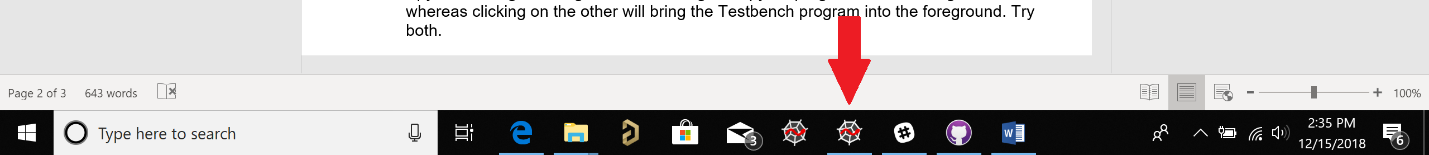
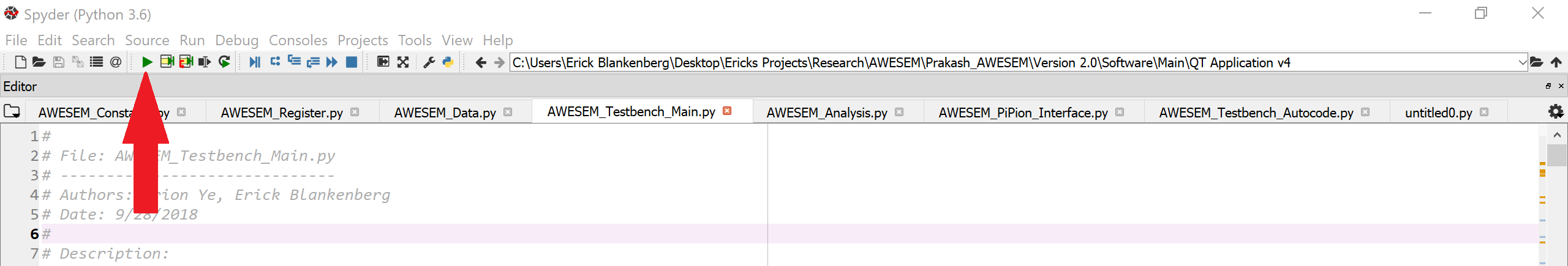
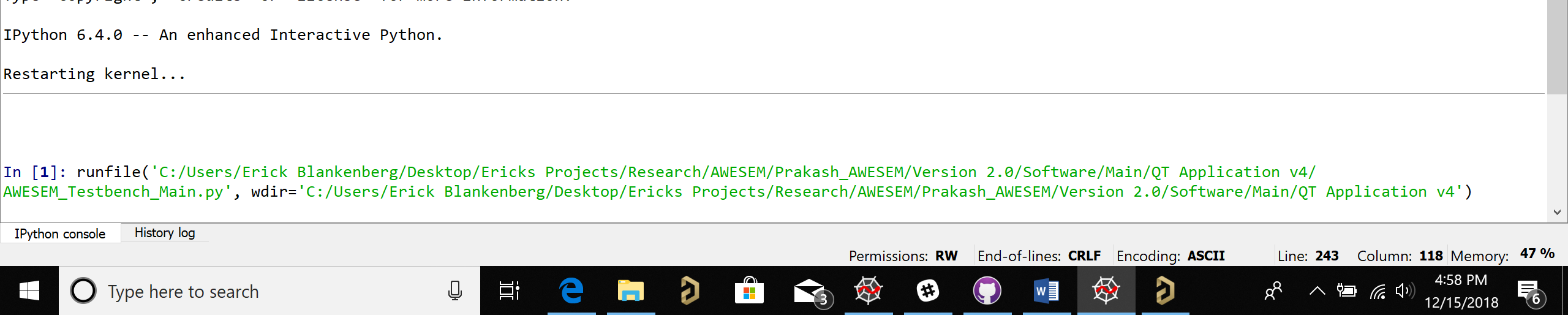
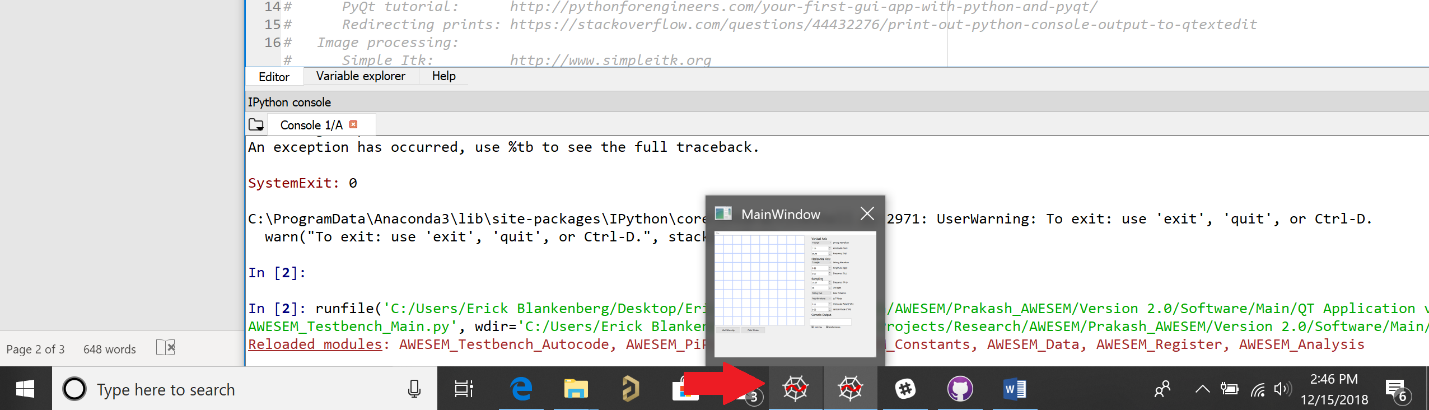
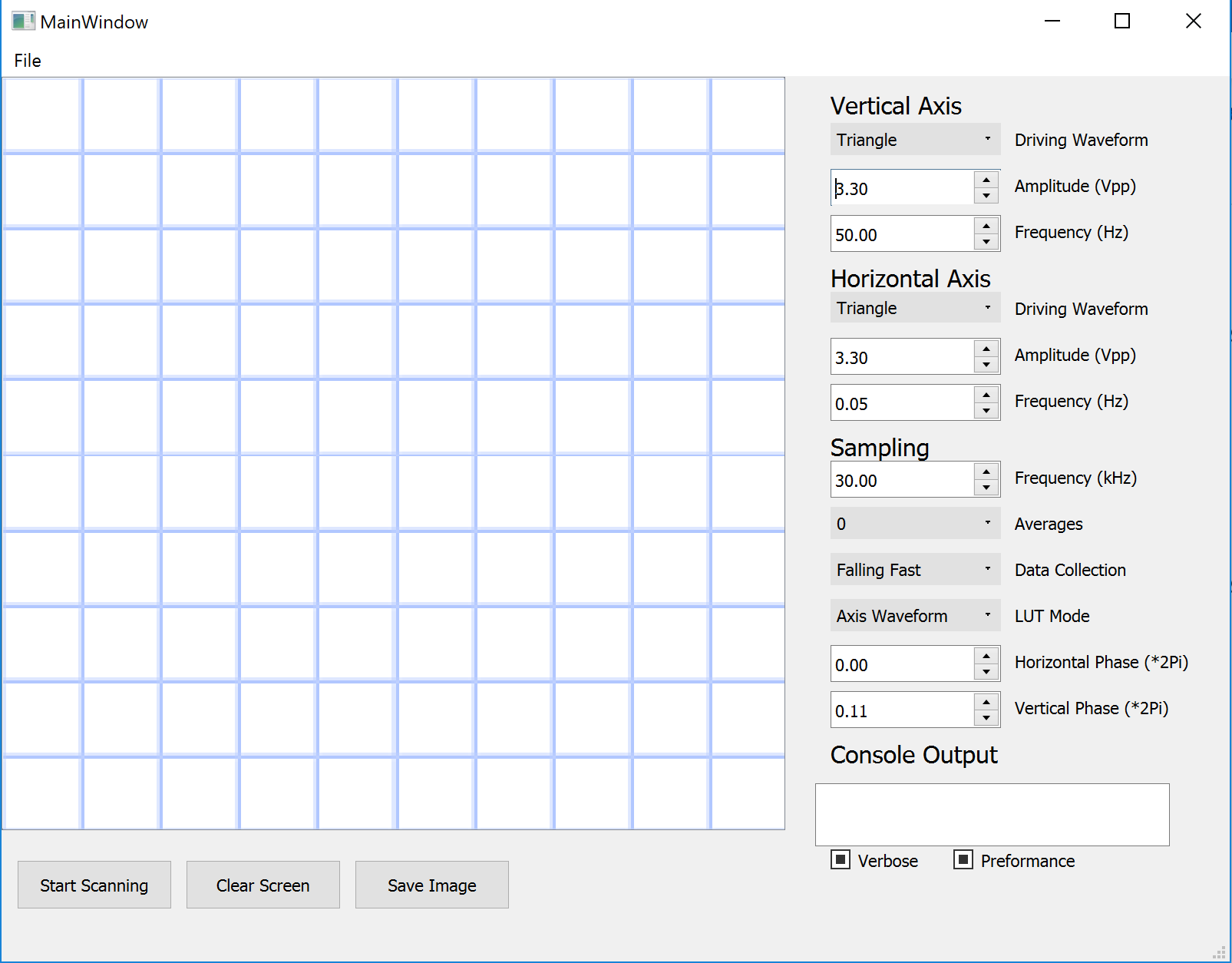
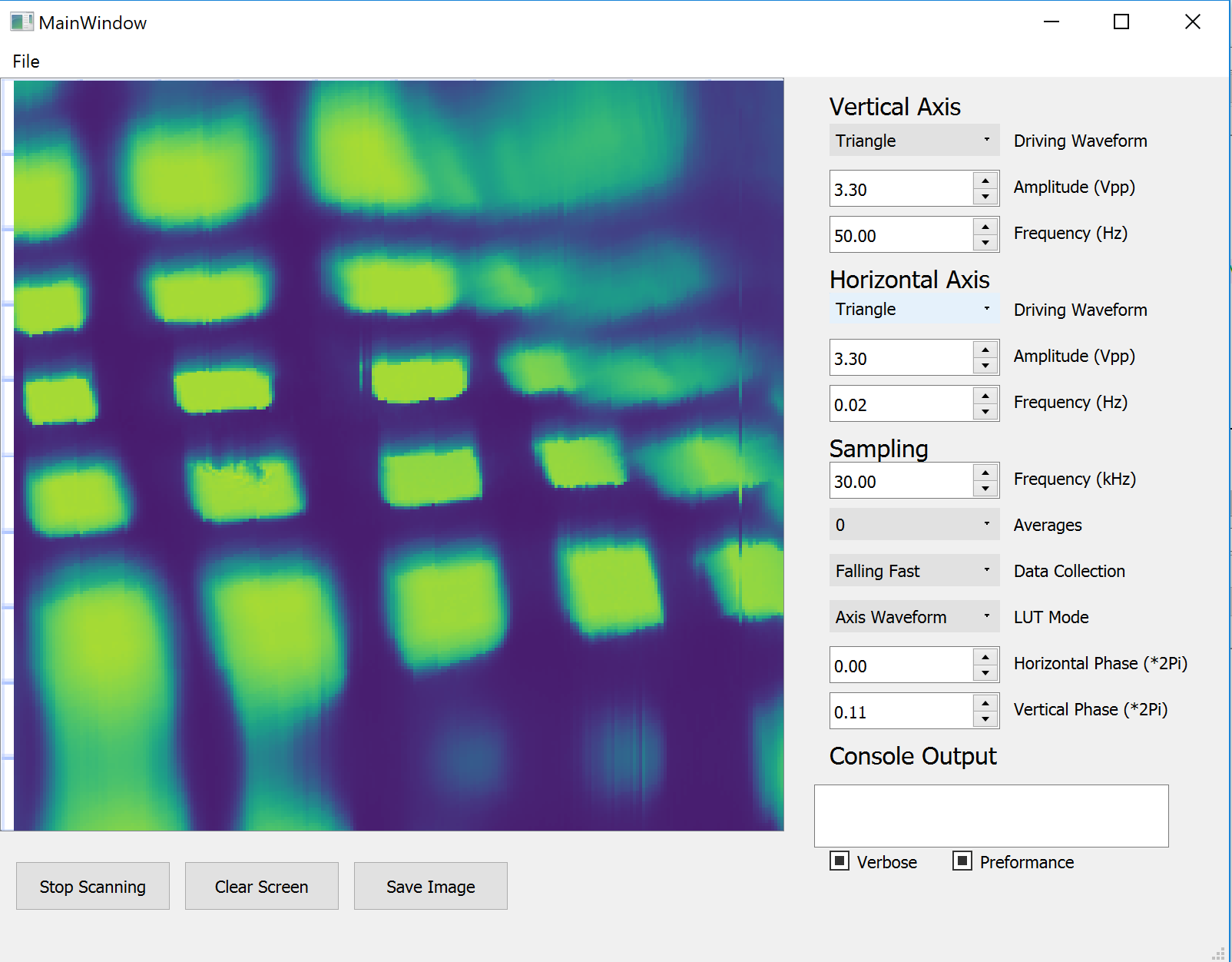
Github Repository: <https://github.com/Eblanken/Prakash_AWESEM.git>

PC Software Directory: Prakash\_AWESEM\Version 2.0\Software\Main\QT Application v4

MCU Directory: Prakash\_AWESEM\Version 2.0\Software\Drivers\PiPion

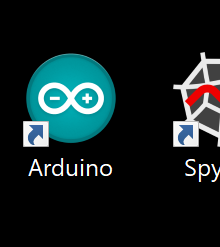
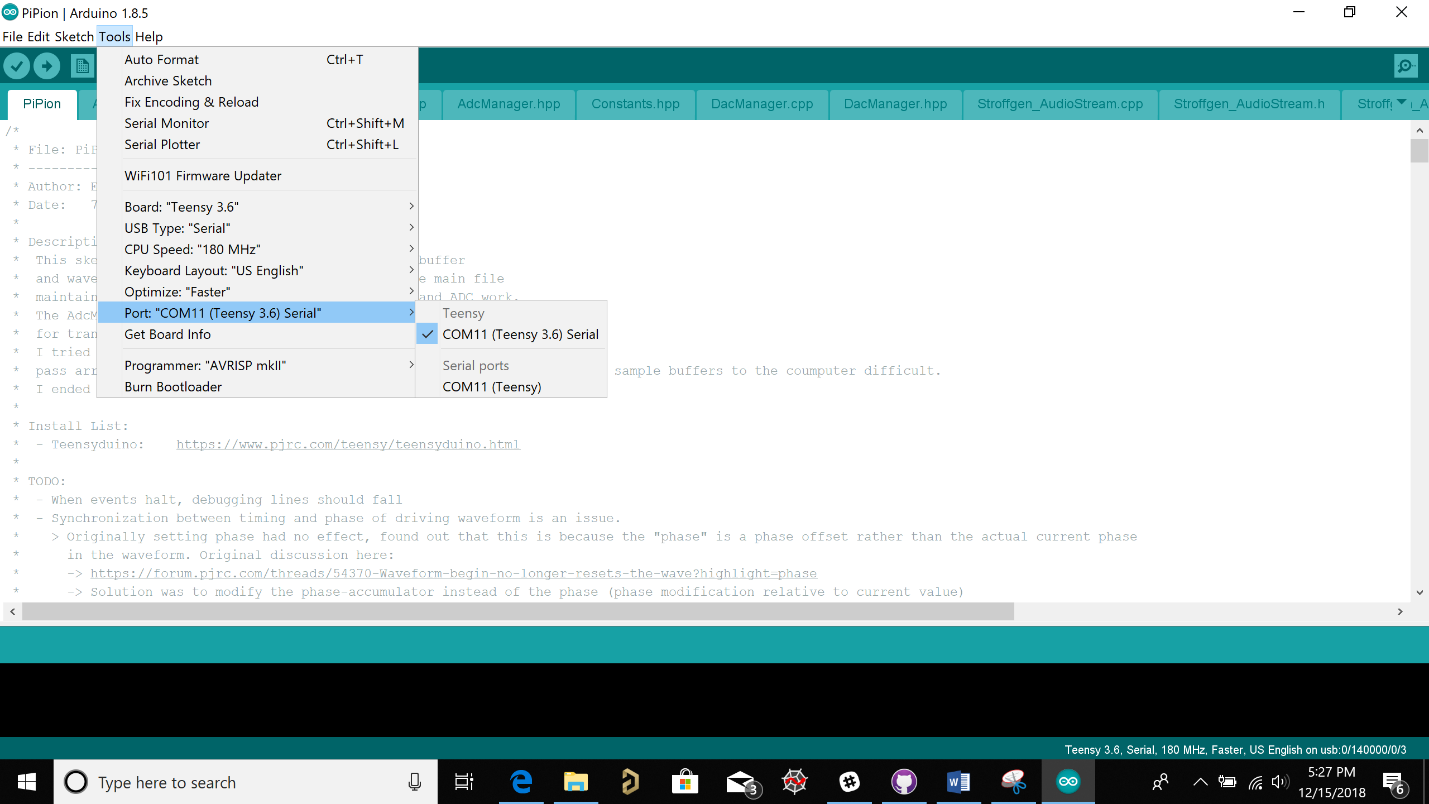
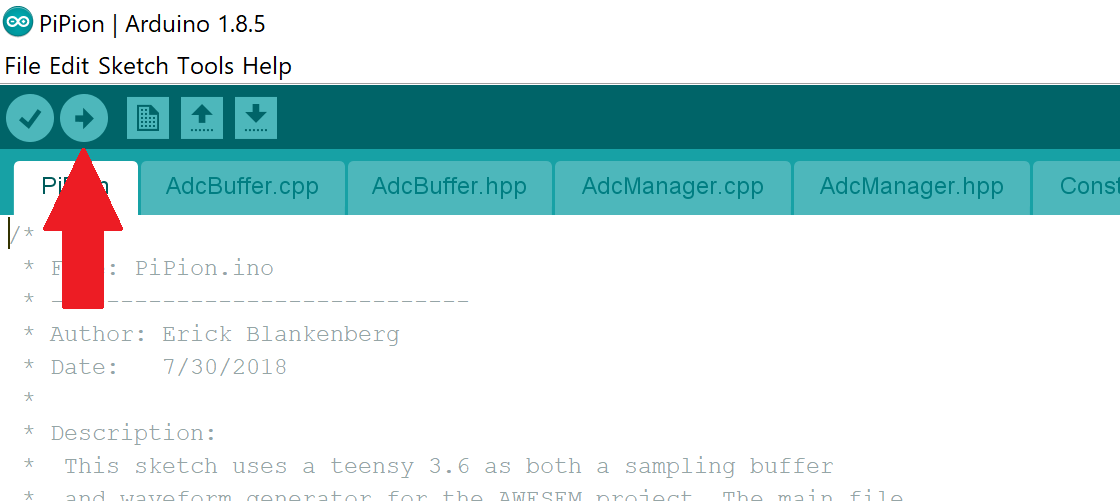
For support, call me at my cell at 1 (530) 524-6910 or at home at 1 (530) 241-7121

# Setup

1. Instructions assume computer is initially turned off and that all electrical connections with the micro-controller are disconnected. Please make sure that the computer is charging from the wall, if the computer runs low on battery it will enter power saving mode which will drastically reduce the performance of the Testbench program.
2. Setting up the computer:
   1. Open the laptop (Gigabyte Aero 14) and press the circular power button above the keyboard. While the computer is booting up the power button should glow white.
   2. On the windows 10 welcome screen, press enter to switch to the user login page
   3. The username and password are as follows:
      1. Username: “Erick Blankenberg”
      2. Password: “Pi31415926”
   4. Once the desktop loads, near the center of the desktop there is an application called “Spyder”. This is the development environment I have been using. Once you open the program it should load all of the files from the last session.
   5. Along the top of the program, there are several tabs, one for each file (ex. AWSEM\_Data.py). Click on the tab labelled “AWSEM\_Testbench\_Main.py”. 
   6. You should see that the code available in the editor has changed. We now need to set up the MCU.
3. Setting up the MCU:
   1. Find a micro USB cable that you know works with other devices and connect the Teensy 3.6 micro-controller to the computer on any USB port.
   2. Set up but do not connect any external support hardware, primarily the driving amplifiers and signal amplifier. Make sure that these will not deliver a potential of greater than 3.3v to any pins of the micro-controller at any time. The output DAC’s of the micro-controller used for driving the stage range from 0v to 3.3v with a neutral value of 1.65v. Please see he “Pinout” section for pin assignment.
   3. Use a multimeter to check that the output of both DACs of the micro controller are steady at 1.65 volts when first connected to the computer. If this is not the case you should first disconnect and reconnect the MCU from the computer. If this does not work, see the troubleshooting entry titled “Uploading MCU Firmware”
   4. Once you have verified that the external analog hardware is safe and that the micro-controller is in the expected initial state as above, connect the external analog signal input to the signal amplifier and the DAC outputs to the external driver amplifiers. Make sure that all grounds are common. Please see the “Pinout” section for details. We should now be able to run the program.
4. Running the program:
   1. Spyder should still be running from before on our desktop. If this is not the case either open the program from the desktop or find it running in the taskbar at the bottom of the screen. If you had to restart Spyder open the tab titled “AWESEM\_Testbench\_Main.py”. 
   2. Along the top of the Spyder IDE there are a few colored icons, click the green arrow icon to run the Testbench program. 
   3. The terminal along the bottom half of the screen should have a message “runfile(…..)”.
      1. If you get an error message indicating that communication initialization failed, then the following may have happened
         1. The MCU is not connected to the computer, check the cable
         2. The serial connection is already being used by another program, make sure that there is not another instance of the program running and that you do not have the Arduino IDE open with a serial monitor connection to the Teensy. If all else fails, restart the kernel as described in the troubleshooting section.
      2. If as soon as you launch the program and have not started scanning you see scrolling messages such as “error…missed X packets” then you may need to restart the kernel. See the troubleshooting section.
   4. The program is now running but may be hidden behind Spyder.
      1. If Spyder does not occupy half of the screen already, click anywhere along the top half of the Spyder program and drag your cursor to one side of the screen. Once your cursor hits the far edge you will see that a small shockwave animation occurs at the cursor and that a glassy rectangular outline occupies half of the desktop. Release the mouse. This should resize Spyder so that it occupies half of the monitor.
   5. If you cannot see the AWESEM Testbench program window, look at the taskbar at the bottom of the desktop. There appear to be two separate instances of Spyder running. Clicking on one will bring the Spyder program into the foreground whereas clicking on the other will bring the Testbench program into the foreground. Try both. You can hover your mouse over the icon in the taskbar to bring up a preview. 
   6. You should now be able to stream live data from the MCU, hit “Start Scanning” to begin.
5. Program Manual
   1. This is the program GUI:
      1. On Startup: 
      2. Capturing Data: 
   2. Sections:
      1. Buttons along lower edge:
         1. “Start Scanning”: Toggles whether the device is active. When not scanning DAC outputs will remain at 1.65v.
         2. “Clear Screen”: Will set image back to the original background grid.
         3. “Save Image”: Will create a copy of the current screen with the current timestamp in the “Captures” folder.
      2. Vertical Axis/Horizontal Axis:
         1. Amplitude (Vpp): Sets potential of driving waveform, maximum value is 3.3v
         2. Frequency (Hz): Sets frequency in hertz of driving waveform, minimum is 0.01, maximum is 200.
      3. Sampling:
         1. Frequency (kHz): Sets sampling frequency of ADC, target value is 40.0 but can be reduced if performance is bad.
         2. Averages: TODO Currently does not affect MCU, MCU currently uses 4 averages per sample.
         3. Data Collection: Restrict plotted data by expected behavior of fastest axis. Choices are “All”, which will plot all available data, “Rising Fast” and “Falling Fast” which will only plot data on the rising or falling edge of an assumed stage response that is the same as the driving function with the given phase offsets.
         4. LUT Mode: Determines reconstruction method. If “Axis Waveform” will assume stage response of same form as driving function with phase offset. If “Linear” will assume sawtooth response waveform and will plot “Modern Art”.
         5. Horizonal/Vertical Phase: Adjusts timing offset of inbound data to try to manually compensate for phase difference between driving waveform and real-world system response.
      4. Console Output:
         1. TODO Currently unused.

# Pinout

# Troubleshooting

1. Uploading MCU Firmware
   1. If the firmware on the MCU is out of date or if the programming button was reset, you will need to re-program the microcontroller. We can do that easily through the Arduino IDE.
   2. Make sure to close any instances of the Testbench program.
   3. On the desktop, you should see the Arduino icon near the center next to Spyder: 
   4. Once you open the IDE, it will load files from the last session. These should be the AWESEM files.
   5. Make sure that the Teensy is visible to the program and is the selected board. Go to Tools>Board and make sure that “Teensy 3.6” is selected. Then go to Tools>Port and make sure that a device is selected. The title of the device in the ports list should contain the term “Teensy”. 
   6. The firmware provided should be ready to upload, hit the right arrow icon to upload to the board. If you see an error message in the terminal below stating that the Teensy could not be put into a programming state, hit the white reset/program button on the top of the Teensy microcontroller.
2. Resetting the python kernel
   1. Occasionally when trying to start the program from Spyder issues may occur. The first and easiest potential fix is to reset the kernel.
   2. Make sure that any instances of the testbench program are closed.
   3. In Spyder, look near the terminal at the bottom of the window. There is a gear icon, click to expand it, press “Restart Kernal” and wait awhile until the terminal view responds. 