LabMaxPro SignalGenerator Firmware

This folder contains the firmware source code for the ***Heisenberg Simulator*** aka ***SignalGenerator***.

As of this writing, ***5/19/2015***, the software supports three instances of the hardware:

1. ***"LiTeensy*** – ***v1"*** the original board, used in conjunction with the standard test cable, the one with the two BNC connectors.

And two newer designs:

1. ***"DB25Teensy – v1"*** which adds a cable and DB25 connector that plugs directly to the *SSIM*.
2. ***“LiTeensy – v3”*** which adds a cable and connector for direct connection to the *PowerMax-Pro*.

# Installing the Tool Chain

The source code is Arduino based and building it requires a particular version of the Arduino tools that is further modified to support the *PJRC* ***Teensy*** device, *version 3.1.*

<https://www.pjrc.com/teensy/>

Thissource code specifically likely will ***NOT*** build with an arbitrary version of the Arduino tools.

The tool chain for building this software has been checked-in to SVN as part of the ***Phoenix Connection*** project. Within that project tree, the path to the tool chain is

*Phoenix Connection\trunk\ThirdParty\Arduino and Teensy support*

Follow the directions in the readme file therein.

# Building and Downloading

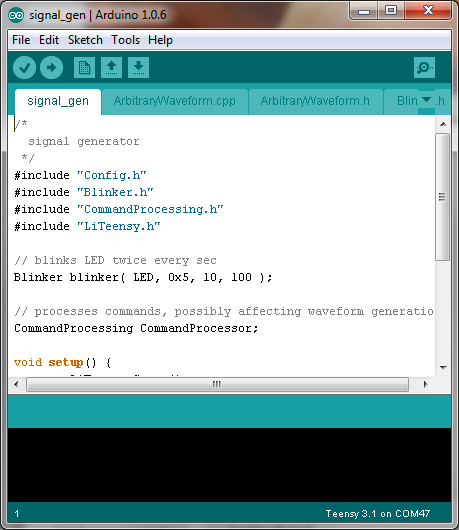
Once you’ve installed the tool chain, you can build the firmware and download it into a ***Teensy 3.1*** device.

The *Arduino* tool chain utilizes a *Java*-like convention in that an application must be located in a folder with the same name as the app. Within that folder, there must be an **.ino** file with the same base name as the containing folder.

Here we’re using the names **signal\_gen** and **signal\_gen.ino**. If you properly installed the tools, the **.ino** file should exhibit the distinctive Arduino icon within the folder:



Double-clicking on the **.ino** file should launch the IDE, as follows:



Normally the IDE loads all the source files in the folder into the window, one file per tab. There’s a combo-box pull-down (little down-arrow far right on the toolbar) to select from all the files.

A complete discussion of how to use this tool is beyond the scope of this document, though there’s a wealth of info on the web, starting with the [Arduino Home Page](http://www.arduino.cc/).

The tool buttons, left to right are:

* **Check** (Compile without downloading)
* **Compile** and download if no build errors
* **Create** New File
* **Open** a file
* **Save** a file that has been modified in the editor.

Under the “tools” menu, there’s a menu for selecting certain options.

* **Board:** selects the target environment. It must be Teensy 3.1 (which won’t be an option if you bollix the install).
* **Serial Port:** selects the port your device is connected to.
* **USB Type:** select *Serial*.
* **Serial Monitor:** if you turn this on, you will see a window which will contain output from the app when it is running. The window automatically is closed when you download and has to be manually re-opened after if you want to use it.

# Final Thoughts

Personally I think the IDE sucks. It is designed for tiny projects, consisting of just a few small files. It’s really not suitable for a project of this size, which is almost beyond its capacity. It should be possible to come up with a batch file that does this, but I never had the incentive or patience to figure out how. The IDE compiles and loads, so whaddayagonnado? It is what it is.

I always use my own favorite text editor to edit the source files, and use the IDE only to build and load the results. One PITA with this approach is that there is no way to tell the IDE to reload the source files. Easiest I’ve found is quit the IDE and re-launch it, and then re-build. It is possible to edit the source code within the IDE, but it’s has its own bizarre notions about auto-indent which I find intolerable.

If someone finds the time to come up with a makefile that builds and loads the code, I’d love it if they checked it into the source tree here.