Pixie Scan Setup Guide

S. V. Paulauskas

December 13, 2012

This guide is intended to aid in the installation of the latest version of the pixie_scan analysis software developed at The University of Tennessee at Knoxville for use with XIAs Pixie-16 electronics. The software is currently only maintained for Linux based systems. Those using Mac and Windows based systems are welcome to make a port, but should not expect support from the developers of this software.

1 Acquiring the software

The latest version of the software is always maintained on github. This version should always be maintained in a stable, working condition. No experimental or potentially volatile features will be included in these releases. Various processors for the detectors being developed will be added as these codes mature.

NOTE: The software requires the ORNL upak libraries, you will need to choose the appropriate software from Link 4 in Section 7.

NOTE 1: The usage of the high resolution timing algorithms, developed for VANDLE, require the GNU Scientific Library. This package is usually available through the standard package repositories. These algorithms can be applied to various scintillators with minimal changes.

1.1 Downloading using git

To download the software via git, the git version control software must be installed on the computer. This is generally available through standard package repositories.

1.1.1 Primary Method

To obtain the code using git issue the following commands. **NOTE:** This command will created a folder called "pixie_scan" in the current directory. If you already have a folder with this name use the alternative method.

Listing 1: Commands used to download code via git-clone

```
\$ git clone git://github.com/pixie16/pixie_scan.git
```

1.1.2 Alternate Method

First, create a directory with the desired name. Then enter the newly created directory and issue the following commands.

Listing 2: Alternate git commands

```
\$ git init
\$ git remote add origin git://github.com/pixie16/pixie_scan.git
\$ git pull origin master
```

More information about using git can be found in Link 6 and 5 in Section 7.

2 Installing the ORNL upak software

Information on how to install this software can be found in Link 4 of Section 7. Be sure that you obtain the proper software for your computer's architecture. If you do not have the proper sources or precompiled binaries, the analysis software **WILL NOT** work properly. If you are not installing this software yourself, contact your system administrator about the currently installed version.

3 Preparing the bash environment

If the ORNL upak software was installed to the standard directories (/usr/hhirf and /usr/acq2/lib) then you can skip Listing 3. However, if the ORNL software is not installed in the standard locations you will need to tell pixie_scan where to look for it. The lines in Listing 3 and 4 should be added to your computer's ".bashrc" file.

Listing 3: Exporting the ORNL paths

```
\$  export HHIRF_DIR=<full path>/hhirf
\$  export ACQ2_LIBDIR=<full path>/acq2/lib
```

The default FORTRAN compiler used for the software is g77. If you are using gfortran you will need to export an additional variable:

Listing 4: Exporting the FORTRAN Compiler

```
\$  export HHIRF_GFORTRAN=1
```

NOTE: The FORTRAN compiler used to compile the software **must** be the same as the version used to compile the ORNL upak libraries.

4 Change options in the Makefile

There are various options at the top of the Makefile. One should choose the appropriate Pixie revision for the data being analyzed. Additionally if the high resolution time analysis is not needed one should comment the PULSEFIT flag. Keeping this flag could potentially cause the scan to be slow as this is a memory intensive process. **Remember** to "make clean" whenever you change options in the Makefile.

5 Choose the necessary Processors

In the file "src/DetectorDriver.cpp" the desired detector processors should be added to the "vecProcess" vector. These definitions are located in the method "DetectorDriver()". There are some default processors currently loaded in the configuration, these can be used as templates for adding your own. To remove a processor from the analysis either delete or comment the line. The available processors are all in the "src/" directory with the title <X>Processor.cpp. It is generally good practice to remove any processors that are not needed for your analysis. This can potentially remove headaches down the road.

In addition to the various detector processors, there are also analyzers, which work on traces. These are added to the analysis in the same manner as the processors. The exception being the analyzers related to the high resolution timing. These are controlled via the Makefile.

NOTE: None of the processors or analyzers that are in the "#if defined" or "#if def" preprocessor commands should be edited, commented, or otherwise molested.

6 Setup the Configuration

All of the configuration files for the analysis are now contained in the "config" directory. This directory contains sub-directories for specific setups. The analysis will use the directory specified in the configuration file ".config". To see what configuration is currently in use

```
\$ cat .config
```

To change the configuration use

```
\$ echo ``config/<desiredConfig>'' > .config
```

This path need not be to a folder in the current directory, as long as it contains all of the necessary configuration files. Possibly the easiest thing to do is to copy one of the current configurations to a new folder (config/¡your config¿). You can then edit these files to reflect the setup that you need.

6.1 The Map File

The old style of map file (map.txt) is no longer supported. Any old map files need to be updated to the new "map2.txt". The new version of the map file makes use of wild cards. This enables one to set up large groups of detectors easily. In addition, there are special "tags" that one can assign to a detector type. For example, if a detector does not have a calibration one can tag with the "uncal" tag. This detector will not need to be added to the "cal.txt" file.

6.2 The Cal File

The file "cal.txt" remains mostly unchanged. The notable exception is that detectors with the "uncal" tag no longer need to be listed here.

6.3 The Tree Correlator

The new file "TreeCorrelator.xml" handles the logic related to the new correlation . For more information related to how to set this file up for your experiment consult the file "treeCorrelatorNotes.pdf" in the "doc/" directory.

If this aspect of the analysis software is not needed one can safely remove all entries between the "<TreeCorrelator>" and "</TreeCorrelator> entry.

6.4 The Timing Cal File

If the desired analysis does not use any VANDLE detectors, this file can be safely ignored. Otherwise this file should be setup to handle the proper timing offsets to be used in the VANDLE analysis.

6.5 High Resolution Timing Analysis

If one is using the high resolution timing analysis. The "traceDelay" and "traceOffset" parameters in the "timingConstants.txt" file should be modified to reflect the proper values for your data. Otherwise the timing analysis will not be able to locate the waveform for the analysis.

7 References

1. `https://github.com/`
2. `https://github.com/pixie16`
3. `https://github.com/pixie16/pixie_scan`
4. `ftp://ftp.phy.ornl.gov/pub/upak/Linux/`
5. `http://rogerdudler.github.com/git-guide/`
6. `http://schacon.github.com/git/everyday.html`

8 Tested systems

The software has successfully been used on the following Unix based systems.

- Arch Linux x86_x64; kernel: 3.6.9-1
- CentOS 6.2 x86_x64
- Fedora 14-17 x86_x64
- MaxOS X
- openSUSE 10.2 x86_64; kernel: 2.6.18-2-34

- RedHat x86_x64
- Ubuntu 10.04-12.04 x86_64 and x32

9 Questions? Bugs? Comments?

Github has a good way to communicate questions/bugs/comments to the code maintainers.

10 Contact information

Project Leader Robert Grzywacz <rgrzywac@utk.edu>

Dev and Repo Maintenance Stan Paulauskas <spaulaus@utk.edu>

Dev Krzysztof Miernik <miernikka@ornl.gov>

Dev Miguel Madurga <mmadurga@utk.edu>