Instruments That Advance The Art

Pixie-4 Express

LabVIEW Demo Interface Manual

Version 4.32

February 1, 2017

Hardware Revision: B

Software Revision: 4.32

XIA LLC
31057 Genstar Rd
Hayward, CA 94544 USA
Email: support@xia.com
Tel: (510) 401-5760; Fax: (510) 401-5761
http://www.xia.com/

Information furnished by XIA LLC is believed to be accurate and reliable. However, no responsibility is assumed by XIA for its use, or for any infringements of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of XIA. XIA reserves the right to change hardware or software specifications at any time without notice.

Table of Contents

Safety

Please take a moment to review these safety precautions. They are provided both for your protection and to prevent damage to the Pixie module and connected equipment. This safety information applies to all operators and service personnel.

Specific Precautions

Observe all of these precautions to ensure your personal safety and to prevent damage to either the Pixie module or equipment connected to it.

Power Source

The Pixie module is powered through a PXI Express (PXIe) chassis. Please refer to the chassis manual for the correct AC voltage connections. The chassis must be powered down to insert and remove the module.

User Adjustments/Disassembly

To avoid personal injury, and/or damage, always turn off power before accessing the Pixie module’s on-board switches and jumpers.

Detector and Preamplifier Damage

Because the Pixie module does not provide power for the detector or preamplifier there is little risk of damage to either resulting from the Pixie module itself. Nonetheless, please review all instructions and safety precautions provided with these components before powering a connected system.

Voltage Ratings

Signals on the analog inputs (gold SMA connectors) must not exceed ± 3.5V. Exceptions apply for certain attenuation and termination settings, see Appendix.

Signals on the digital inputs (gold MMCX connector and 10-pin 2mm har-link connector) must not exceed 3.3V.

Servicing and Cleaning

To avoid personal injury, and/or damage to the Pixie module or connected equipment, do not attempt to repair or clean these units. These modules are warranted against all defects for one (1) year. Please contact the factory or your distributor before returning items for service.

Warranty Statement

XIA LLC warrants that this product will be free from defects in materials and workmanship for a period of one (1) year from the date of shipment. If any such product proves defective during this warranty period, XIA LLC, at its option, will either repair the defective products without charge for parts and labor, or will provide a replacement in exchange for the defective product.

In order to obtain service under this warranty, Customer must notify XIA LLC of the defect before the expiration of the warranty period and make suitable arrangements for the performance of the service.

This warranty shall not apply to any defect, failure or damage caused by improper uses or inadequate care. XIA LLC shall not be obligated to furnish service under this warranty a) to repair damage resulting from attempts by personnel other than XIA LLC representatives to repair or service the product; or b) to repair damage resulting from improper use or connection to incompatible equipment.

THIS WARRANTY IS GIVEN BY XIA LLC WITH RESPECT TO THIS PRODUCT IN LIEU OF ANY OTHER WARRANTIES, EXPRESSED OR IMPLIED. XIA LLC AND ITS VENDORS DISCLAIM ANY IMPLIED WARRANTIES OF MERCHANTABILITYOR FITNESS FOR A PARTICULAR PURPOSE. XIA'S RESPONSIBILITY TO REPAIR OR REPLACE DEFECTIVE PRODUCTS IS THE SOLE AND EXCLUSIVE REMEDY PROVIDED TO THE CUSTOMER FOR BREACH OF THIS WARRANTY. XIA LLC AND ITS VENDORS WILL NOT BE LIABLE FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES IRRESPECTIVE OF WHETHER XIA LLC OR THE VENDOR HAS ADVANCE NOTICE OF THE POSSIBILITY OF SUCH DAMAGES.

Contact Information:

XIA LLC
31057 Genstar Rd.
Hayward, CA 94544 USA

Telephone:	(510) 401-5760
Downloads:	http://support.xia.com
Hardware Support:	support@xia.com
Software Support:	support@xia.com

1 Introduction

This document describes the LabVIEW demo interface for the Pixie-4, Pixie-4 Express and Pixie-500 Express modules. While fully functional to operate the modules, it is mainly intended as an example for users who want to integrate the Pixie software into a larger LabVIEW based data acquisition system. For a full description of the modules' functions and features, please refer to the appropriate User Manuals. In particular, references in this manual to sections 4 and higher imply these sections are located in the Pixie-4 Express User Manual.

Section 2 is intended to give a quick overview of installation and getting started with the Pixie Viewer. A description of all VI is found in section 3. For full details of the advanced controls in the VIs, please consult the Online Help manual released as part of the software distribution. It mainly describes the Pixie Viewer (based on Igor Pro), but the controls are generally named the same.

As a demo interface, the LabVIEW VIs do not show the full range of controls and functions. These can be added as necessary; it is always possible to set parameters in the Pixie Viewer, save to the .set file, and specify that file in LabVIEW for the module.

As far as the interface is concerned, the difference between the different module types manifests in these issues:

- The interface automatically rounds user specified times (e.g. trigger filter length) to closest multiples of the base digitization or processing time (2ns, 8ns or 13.33ns).
- When viewing captured waveforms, the time unit has to be set correctly in the VI by the user
- Firmware files for all module types have to be present; the software picks the right files to use based on the module type detected.

2 Setting Up

2.1 Installation

2.1.1 Hardware Setup

The Pixie-4 Express and Pixie-500 Express modules can be operated in any standard 3U PXIe chassis, using PXIe or hybrid slots. The Pixie-4 can be operated in any standard CompactPCI or PXI chassis, using only plain PXI slots, not PXIe hybrids. To communicate between modules a backplane adhering to the PXI(e) standard must be present.

To begin, place the embedded computer (or remote controller) in the system slot of your chassis. Place the Pixie modules into any free slots with the chassis still powered down, then power up the chassis (Pixie modules are not hot swappable). If using a remote controller, be sure to boot the host computer *after* powering up the chassis[1].

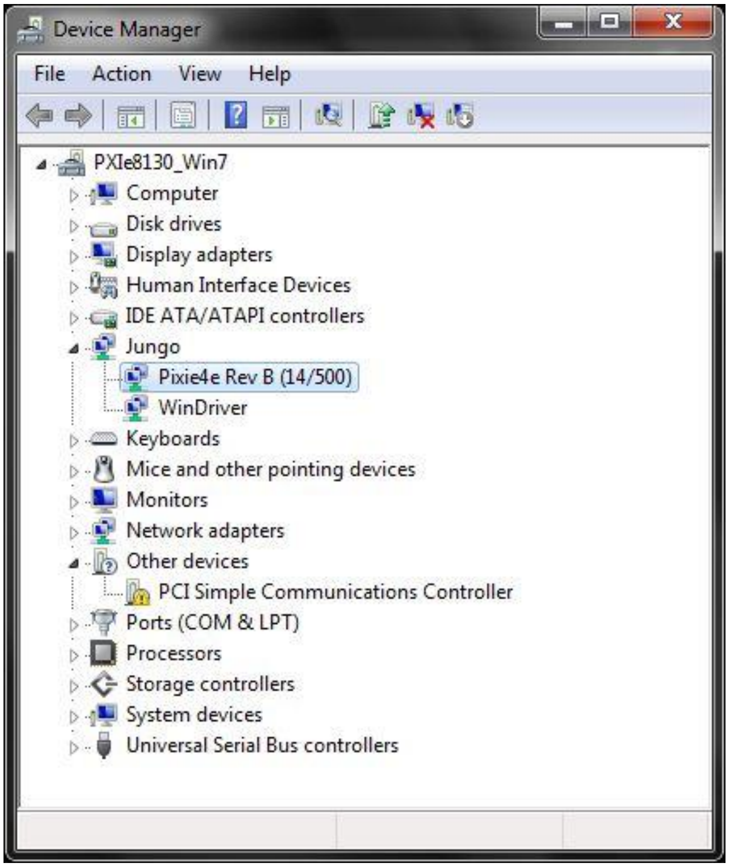
2.1.2 Hardware Drivers

When the host computer is powered up the first time (after installing the controller and Pixie modules in the chassis) it will detect new hardware and attempt to find the appropriate drivers. (A Pixie module will be detected as a new device every time it is installed in a new slot.) While there is no required order of installation of the driver software, the following sequence is recommended:

1. If you have a remote controller, first install the driver software for the controller itself. Otherwise, skip to step 4.
   Unless directed otherwise by the manufacturer of the controller, this can be done with or without the controller and Pixie-4 modules installed in the host computer and/or chassis. If the modules are installed, ignore attempts by Windows to install drivers until step 7.
   NI controllers come with a multi-CD package called “Device Driver Reference CD”. For simplicity it is recommended to install the software on these CDs in the default configuration.
2. Unless already installed, power down the host computer, install the controller in both the host computer and chassis, and power up the system again (chassis first).
3. Windows will detect new hardware (the controller) and should find the drivers automatically. Verify in Window’s device manager that the controller is properly installed and has no “resource conflicts”.
4. Install LabVIEW, version 8.5 or higher
5. Install the Pixie software provided by XIA (see section 2.1.3)

[1] In some systems, “scan for hardware changes” in the Windows device manager may detect and install a remote chassis when the PC was booted first.

6. Unless already installed, power down the host computer and install the Pixie modules in the chassis. Check the input settings for the appropriate signal termination: 50 Ω or 5 kΩ (see section 10.1 for details). Then power up the system again (chassis first).
7. Windows will detect new hardware (the Pixie modules) and should find the drivers automatically. If not, direct it to the “drivers” directory in the Pixie software distribution installed in step 5.
8. A) Pixie-4: Verify in Window’s device manager that Pixie-4 modules are properly installed as “PLX Custom (OEM) PCI 9054 Boards (##)” and have no “resource conflicts”. Currently, the driver must be version 6.5. ## refers to the operating system: 32 for 32bit or 64 for 64 bit.
   B) Pixie-4 Express: Verify in Window’s device manager under Jungo that Pixie-4 Express modules are properly installed as “Pixie4e Rev B (bb/rrr)” and have no “resource conflicts”. bb refers to the ADC precision (16 for 16bit ADCs), rrr refers to the ADC rate (125 for 125 MSPS).
   C) Pixie-500 Express: Verify in Window’s device manager under Jungo that Pixie-500 Express modules are properly installed as “Pixie500e Rev B” and have no “resource conflicts”. [2]



[2] On 64 bit Windows, the module name is Pixie-4e and Pixie-500e (with a dash).

2.1.3 Pixie LabVIEW User Interface

To “install” the Pixie LabVIEW interface (called Pixie Viewer), first run the Pixie-Viewer setup program that installs the Igor based Pixie Viewer. This will also install all necessary hardware drivers of the correct 32/64bit type. Then copy the complete LabVIEW folder from the installation CD (or zip file) to the folder C:\XIA\Pixie4e, or to a custom folder.

The communication from LabVIEW with the hardware goes through two levels of dlls: level 1 is for hardware I/O for the PCI or PCIe interface (PLX or Jungo/WinDriver) and level 2 is for module operations (XIA/Pixie): These dlls depend both on the OS 32/64bit type and the application 32/64bit type, resulting in 3 possible cases (actions not automatically performed by the installer in bold):

- For 32bit Windows and 32bit LabVIEW,
  - use 32bit hardware drivers
  - copy Drivers\PlxApi650.dll to Windows\System
  - copy Drivers\x86\wdapi1120.dll to Windows\System32
  - link LabVIEW VIs to Drivers\x86\Pixie4DLL.dll
- For 64bit Windows and 32bit LabVIEW,
  - use 64bit hardware drivers
  - copy Drivers\PlxApi650.dll to Windows\System
  - copy Drivers\amd64\wdapi1120.dll to Windows\System32
  - link LabVIEW VIs to Drivers\amd64\Pixie4DLL.dll
- For 64bit Windows and 64bit LabVIEW,
  - use 64bit hardware drivers
  - copy Drivers\LV64\PlxApi650_x64.dll to Windows\System
  - copy Drivers\LV64\wdapi1120_6464.dll to Windows\System32
  - link LabVIEW VIs to Drivers\LV64\Pixie4DLL.dll

The last step of linking LabVIEW Vis to the Pixie dll is

LabVIEW usually tries to find the best match for the Pixie dll automatically. It may remember the last location (from development at XIA), look for the dll in the folder of the local VI, or find it somewhere else. To avoid ambiguity, it is best to remove any unused dll of the wrong type from the installation. (Copies cab ne located in subfolders of …\Drivers and in subfolders of …\PixieClib.)

The LabVIEW distribution contains several LabVIEW Virtual Instruments (VIs) and eight subfolders (Configuration, Doc, Drivers, DSP, Firmware, MCA, PixieClib, and PulseShape). Make sure this folder organization is preserved, as the software expects files to be in those folders. Feel free, however, to add folders and subfolders at your convenience.

Functional (programming) or cosmetic (appearance) changes to any of the VIs may be saved by clicking on File -> Save from the top menu. This action saves the current state of the interface. This same action must be performed on all open VIs to save their status.

Generally, for any action in the VIs to be executed, the user first has to click the button (or other widget), then click on the “run” button at the top of the panel.

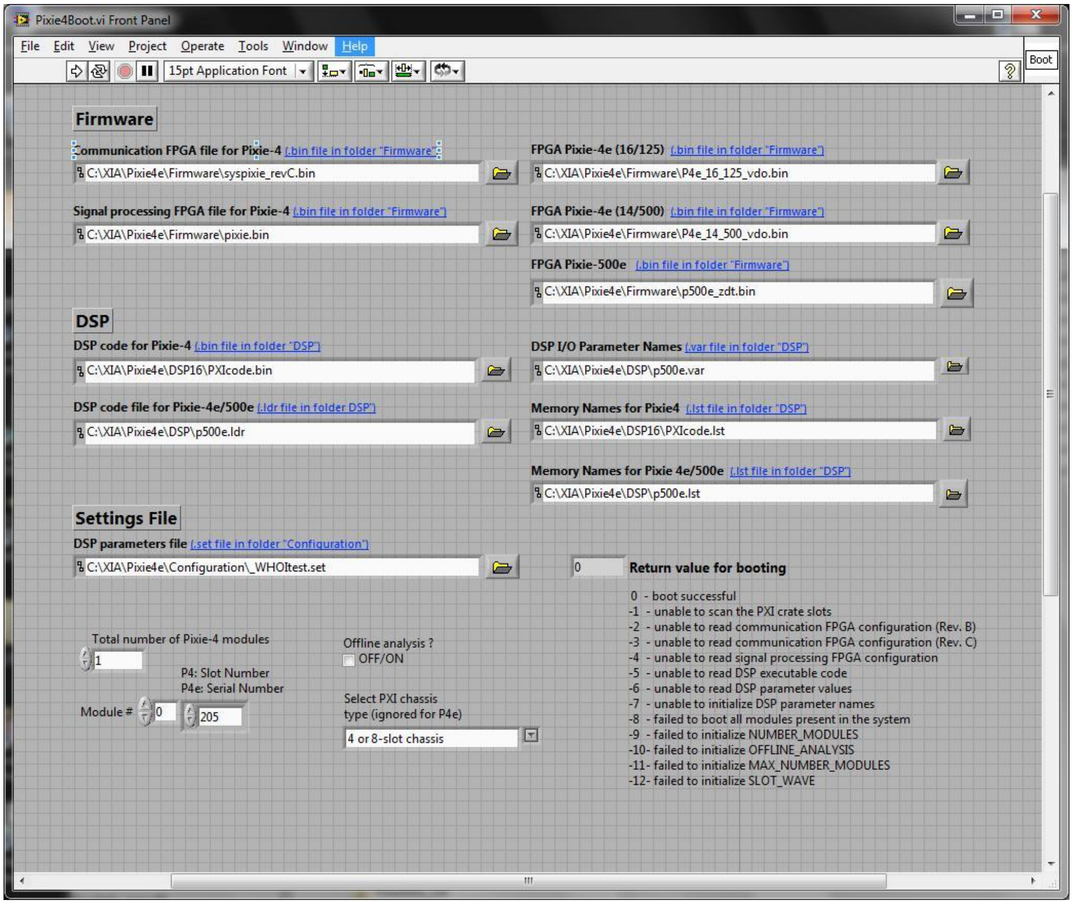
Error and debug messages are collected in a file “PIXIEmsg.txt”, usually located in the last folder a VI was started from or accessed by a VI.

2.2 Getting Started

2.2.1 Booting (PIXIE4BOOT.vi)

To initiate communication with a Pixie module, double-click on the file PIXIE4BOOT.VI in the LabVIEW folder. Prior to booting the Pixie module, the correct paths to the firmware, DSP files and settings files must be set (blue underlined text suggests a likely locations for the pertinent files) as shown in Figure 2.1. The procedure to change and save the defaults settings is

1. Ctrl-M to enter the VI edit mode
2. Edit the text fields to insert the correct paths
3. Choose Edit->Make Current Path Default in the Edit menu
4. Save Pixie4Boot.vi


Pixie4Boot.vi Front Panel
File Edit View Project Operate Tools Window Help
15pt Application Font
Boot

Firmware

Communication FPGA file for Pixie-4 (.bin file in folder "Firmware")
C:\XIA\Pixie4e\Firmware\syspixie_revC.bin

Signal processing FPGA file for Pixie-4 (.bin file in folder "Firmware")
C:\XIA\Pixie4e\Firmware\pixie.bin

FPGA Pixie-4e (16/125) (.bin file in folder "Firmware")
C:\XIA\Pixie4e\Firmware\P4e_16_125_vdo.bin

FPGA Pixie-4e (14/500) (.bin file in folder "Firmware")
C:\XIA\Pixie4e\Firmware\P4e_14_500_vdo.bin

FPGA Pixie-500e (.bin file in folder "Firmware")
C:\XIA\Pixie4e\Firmware\p500e_zdt.bin

DSP

DSP code for Pixie-4 (.bin file in folder "DSP")
C:\XIA\Pixie4e\DSP16\PXIcode.bin

DSP code file for Pixie-4e/500e (.ldr file in folder DSP")
C:\XIA\Pixie4e\DSP\p500e.ldr

DSP I/O Parameter Names (.var file in folder "DSP")
C:\XIA\Pixie4e\DSP\p500e.var

Memory Names for Pixie4 (.lst file in folder "DSP")
C:\XIA\Pixie4e\DSP16\PXIcode.lst

Memory Names for Pixie 4e/500e (.lst file in folder "DSP")
C:\XIA\Pixie4e\DSP\p500e.lst

Settings File

DSP parameters file (.set file in folder "Configuration")
C:\XIA\Pixie4e\Configuration_WHOItest.set

Total number of Pixie-4 modules: 1

Module # 0
P4: Slot Number
P4e: Serial Number
205

Offline analysis ?
□ OFF/ON

Select PXI chassis
type (ignored for P4e)
4 or 8-slot chassis

0 Return value for booting

0 - boot successful
-1 - unable to scan the PXI crate slots
-2 - unable to read communication FPGA configuration (Rev. B)
-3 - unable to read communication FPGA configuration (Rev. C)
-4 - unable to read signal processing FPGA configuration
-5 - unable to read DSP executable code
-6 - unable to read DSP parameter values
-7 - unable to initialize DSP parameter names
-8 - failed to boot all modules present in the system
-9 - failed to initialize NUMBER_MODULES
-10- failed to initialize OFFLINE_ANALYSIS
-11- failed to initialize MAX_NUMBER_MODULES
-12- failed to initialize SLOT_WAVE

Figure 2-1: The PIXIE4BOOT VI

In addition to the paths, the total number of Pixie-4 modules must be entered, as well as the PXI slot number for each Pixie-4 module or serial number for each Pixie-4 Express and Pixie-500 Express module. This action is executed by entering the slot or serial number for module 0, toggling the module number, and entering the slot or serial number for the next module and so on. Lastly, the chassis type must be selected. If you want to run the software without a chassis or module attached, check **Offline Analysis**.

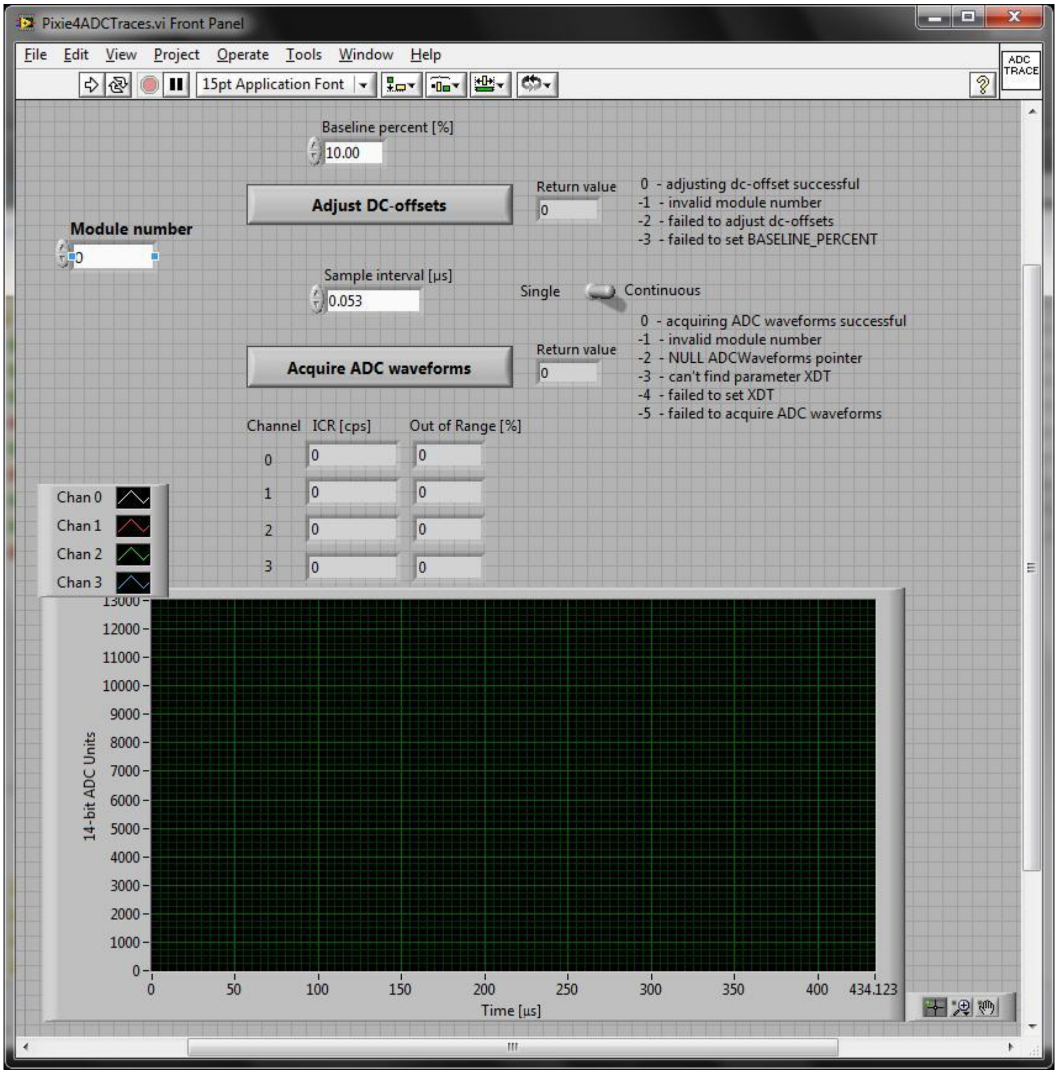
The Pixie-4 is booted by clicking the run button (arrow) in the top panel[3]. A series of return values indicates the boot status - a return value of 0 indicates a successful boot sequence (but also the default state before booting). Otherwise, refer to the troubleshooting section for possible solutions.

2.2.2 View ADC traces (PIXIE4ADCTRACES.vi)

With the Pixie module booted it is strongly suggested that the ADC waveforms be inspected to ensure that the sampled waveforms fall into the ADC voltage range before data is acquired.

Waveforms can be viewed by launching PIXIE4ADCTRACES.VI (Figure 2.2). Executing **Acquire ADC waveforms** displays traces from all 4 channels spanning 8192 time steps**.** **Sample interval** sets the time between successive steps. The toggle widget allows the user to switch between **Single** trace or **Continuous** trace acquisition. As each channel may have slightly analog different gains and/or offsets when a module is booted, **Adjust DC-Offsets** automatically sets a common DC-offset level for each channel (default: 10%)

[3] All subsequent action issued from any of the VI modules require that the run button be pressed for the command to be processed. This additional step is implied in all further descriptions.



Baseline percent [%] 10.00

Adjust DC-offsets

Return value 0

0 - adjusting dc-offset successful
-1 - invalid module number
-2 - failed to adjust dc-offsets
-3 - failed to set BASELINE_PERCENT

Module number 0

Sample interval [µs] 0.053

Single Continuous

0 - acquiring ADC waveforms successful
-1 - invalid module number
-2 - NULL ADCWaveforms pointer
-3 - can't find parameter XDT
-4 - failed to set XDT
-5 - failed to acquire ADC waveforms

Acquire ADC waveforms

Return value 0

Channel	ICR [cps]	Out of Range [%]
0	0	0
1	0	0
2	0	0
3	0	0

Figure 2-2: The PIXIE4ADCTRACES VI

Waveforms should be comfortably within ADC range, i.e. baseline at about 10%, pulses starting with a rising edge, and not clipped at the upper end of the ADC range. If any of the pulses are inverted, **Trigger positive** in the Register A tab of the PIXIE4SETUSERPAR VI should be toggled (Section 3.2). If pulses go out of range, adjust the **Gain** in the **Analog Signal Conditioning** tab in the PIXIE4SETUSERPAR VI, then again execute **Adjust DC-Offsets**. You can also manually specify an offset in the **Analog Signal Conditioning** tab. Signals that still fall out of the ADC range must either be terminated differently or the detector gain reduced.

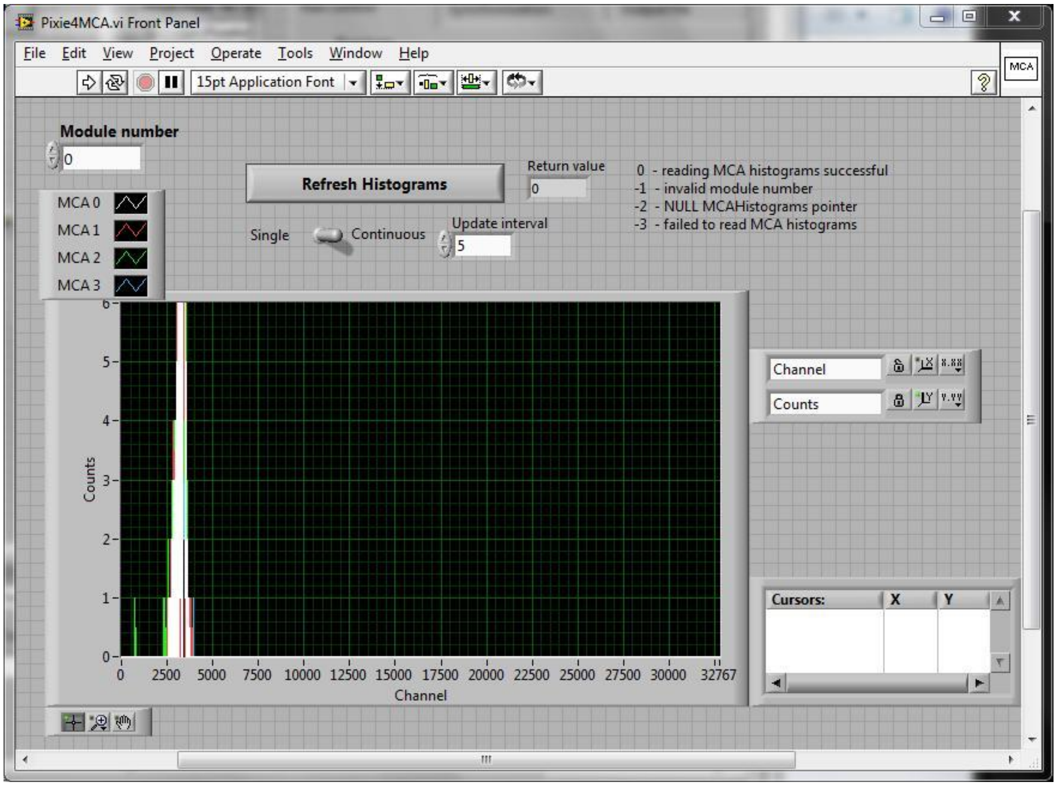
Generally speaking, changes to parameters made in the PIXIE4SETUSERPAR VI should follow this sequence: First ensure panel values are identical to the values present in the module's DSP by executing **Upload**. Then change values and apply them to the DSP by executing **Download** at the bottom of this same VI .When changes are applied, values are rounded or limited according to the module's valid range, so the final step should be to execute **Upload** again to see true values used by the module. At any point modified DSP parameter settings may be saved to file. To do so, launch the PIXIE4SAVEDSPPARTOFILE VI. It is advisable to create a new file name to avoid overwriting the default settings file.

2.2.3 Data Acquisition

Once the ADC traces have been properly adjusted, the pulse decay time constant has to be specified. In PIXIE4SETUSERPAR VI, click on the **Decay time** tab and enter an estimated preamplifier exponential RC decay time for **Tau**. Save settings as described above.

To start a data acquisition run, open the PIXIE4DAQ VI. Click on the **Run control** tab, set **Run Type** to “0x301: MCA Mode”, **Poll time** to .1 second, and **Run time** to 30 seconds or so, then click on the run button.

To view the result, open the PIXIE4MCA VI and execute **Refresh Histogram**.



Module number: 0

Return value: 0

0 - reading MCA histograms successful
-1 - invalid module number
-2 - NULL MCAHistograms pointer
-3 - failed to read MCA histograms

Update interval: 5

Figure 2-3: The PIXIE4MCA VI

This brief overview is intended to quickly get the user up and acquiring data with the Pixie module. Undoubtedly there are user parameters settings such as energy filter rise time and flat top that may need adjustment to optimize your results. These parameters, along with other a host of logical control parameters are described in the next section.

3 Navigating the LabVIEW VIs

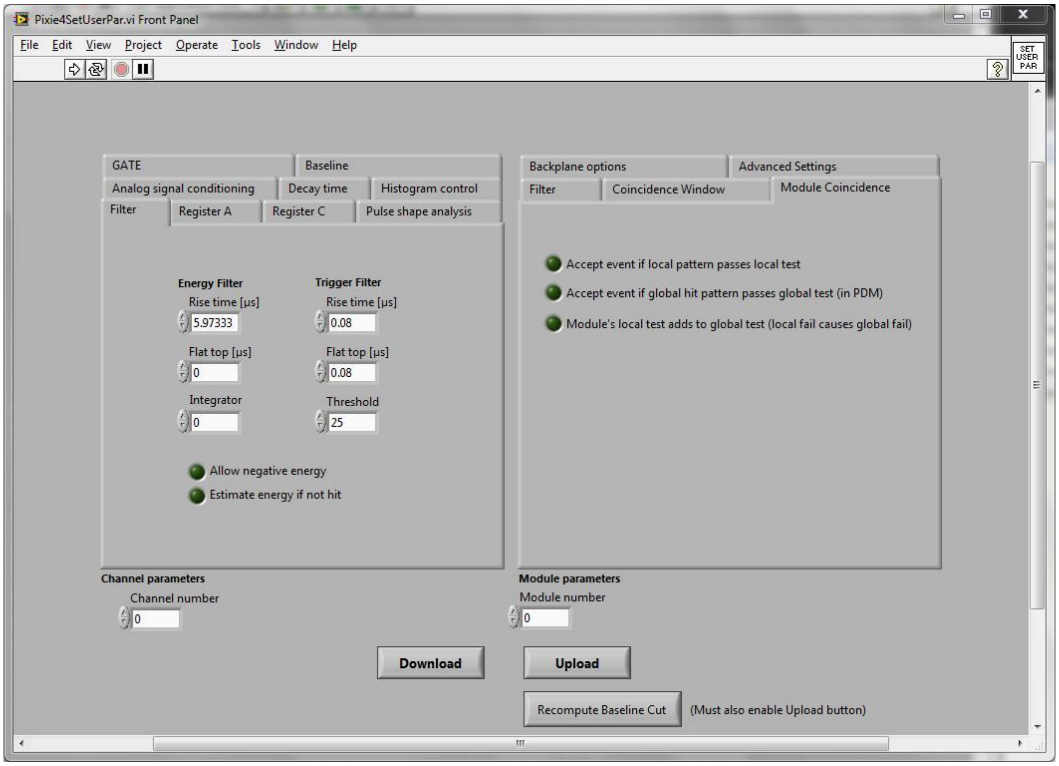
3.1 Overview

The LabVIEW interface consists of eight VIs, each of which can be independently launched on an as-needed basis. Loosely speaking they can be grouped into three categories:

- Setup Control
    - PIXIE4BOOT.vi,
    - PIXIE4ADCTRACES.vi,
    - PIXIE4SETUSERPAR.vi
    - PIXIE4SAVEDSPPARTTOFILE.vi
- Run Control
    - PIXIE4DAQ.vi
    - PIXIE4DAQE.vi
- Results
    - PIXIE4MCA.vi,
    - PIXIE4PULSESHAPE.vi
    - PIXIE4READRUNSTATISTICS.vi

3.2 Setup Group - PIXIE4SETUSERPAR.vi

All user parameters are accessed via the PIXIE4SETUSERPAR VI. Figure 3.2 is an overall view of the VI with tabs so designated as to give the user some idea of their functionality. The PIXIE4SETUSERPAR VI is itself divided into two subpanels: channel (left) versus module/system parameters (right).



Energy Filter

Rise time [µs] 5.97333

Flat top [µs] 0

Integrator 0

Trigger Filter

Rise time [µs] 0.08

Flat top [µs] 0.08

Threshold 25

● Allow negative energy

● Estimate energy if not hit

● Accept event if local pattern passes local test

● Accept event if global hit pattern passes global test (in PDM)

● Module's local test adds to global test (local fail causes global fail)

Channel parameters

Channel number 0

Module parameters

Module number 0

(Must also enable Upload button)

Figure 3-1: PIXIE4SETUSERPAR VI.

User parameters in the PIXIE4SETUSERPAR VI are available to tailor the Pixie module to the particular requirements of a given experimental setup. As befits a digital device, all parameter settings are stored in a settings file. This file is separate from LabVIEW VIs and provides flexibility by saving and restoring different settings for different detectors and applications. At boot time the parameters from the DSP parameter file specified in the PIXIE4BOOT VI are automatically downloaded to the module. After the module is booted, changes to any of the user parameter are applied to the module(s) by executing **Upload**, user input, **Download**, **Upload** (to verify) on the PIXIE4SETUSERPAR VI. Unless these changes are committed by writing them to a configuration file using the PIXIE4SAVEDSPPARTOFILE VI they will be lost on the next module reboot. Should an entirely different DSP parameter file be desired, a reboot is required.

When a user enters a parameter value for one channel, the same numerical value is automatically copied to all channels (for the sake of convenience; the rationale being that the most commonly encountered experimental setup is multiple channels of the same type of instrument). However, the value entered for that channel *only* is written to the DSP when executing **Download** with this channel selected. For example, if the **Tau** parameter in the **Decay Time** tab is changed for channel 0 and module 0, this same tau value will be copied to the **Tau** text entry boxes of all channels and modules. Executing **Download** would then download this revised tau value to the DSP only for channel 0 and module 0. If this same tau value happens to be appropriate for all channels and modules, then the user must toggle through each combination of channel and module and follow this same procedure. In a similar vein, if a separate tau value is required for some or all of the channels and modules then the user must again toggle through all combinations of channels and modules, entering a different tau value each time, executing **Download** before moving on to the next channel. Confusion may arise if the user were to toggle back to a channel for which the tau value has already been committed to the DSP since the most recently entered value will instead appear in the **Tau** text entry box. Nonetheless, one can confirm that the correct value for that channel was entered by executing **Upload**.

We now proceed with a detailed description of the functionality of each of the tabs in the PIXIE4SETUSERPAR VI. We start by describing the functionality of the tabs in the left panel before doing the same for tabs in the right panel. Before proceeding, we note that associated functionality for many of the tabs described in this section are found in tabs **Register A** and **Register C**. This will be noted on a case-by case basis.

3.2.1 Filter Tab

The **Filter** tab contains controls to set the trigger and energy filter parameters. These two filters are described in the following subsections

3.2.1.1 Trigger filter

The **Trigger filter** column defines the trigger filter times and **threshold**. Except for the threshold, the trigger settings have rarely to be changed from their default values.

The threshold value corresponds to ¼ of the pulse height in ADC steps, e.g. with a threshold of 20, triggers are issued for pulses above 80 ADC steps. This relation is true if the trigger filter **rise time** is large compared to the pulse rise time and small compared to the pulse decay time. A pulse shape not meeting these conditions has the effect of raising the effective threshold. The threshold value is scaled with the trigger filter **rise time**, therefore it is not limited to integer numbers. The **Register A** tab provides advanced trigger filter functionality via the buttons

- **Respond to group triggers only** to capture waveforms based on distributed, not local trigger,
- **Good channel** to enable or disable a channel,
- **Read always** to force read out of channels even in absence of a hit,
- **Enable trigger** to allow a channel to trigger acquisition (itself and other channels), and
- **Trigger positive** to invert incoming signals (trigger on inverted waveforms).

3.2.1.2 Energy filter

The **Energy filter** column contains the settings for the energy filter and the subsequent energy computation. These settings are paramount for obtaining the best possible energy resolution with a Pixie system. The energy filter **rise time** (or peaking time) essentially sets the trade-off between throughput and resolution: longer filter **rise times** generally improve the resolution (up to a certain optimum) but reduce the throughput because more time is required to measure each pulse. The **Integrator** parameter modifies the energy computation. For a detailed description of the filter operation see Section 6 of the user manual.

3.2.2 Pulse shape analysis tab

The **Pulse Shape Analysis** tab contains the controls to set the length and pre-trigger delay of the waveforms to be acquired. Advanced options include parameters for online pulse shape analysis

3.2.3 GATE Tab

The *GATE* tab, in tandem with tabs *Register A* and *Register C* contains the controls to set the window for gating acquisition with external signals. For a detailed description of the GATE and VETO operation, see sections 6.6.2 and 7.4 of the user manual.

3.2.4 Tau Tab

The pulse decay time **Tau** is used to compensate for the decay of a previous pulse in the computation of the pulse height. The user can enter a known value or use the waveforms from the PIXIE4ADCTRACES.VI to estimate it.

3.2.5 Analog signal conditioning Tab

The **Analog signal conditioning** tab has two text entry boxes whose values can be changed through the increment widget or by manual entry. The two settings are

- **Gain (V/V)** which sets the analog gain before digitization, and
- **Offset (V),** which directly sets the offset voltage.

3.2.6 Coincidence Window Tab

The **Coincidence** tab contains the controls to set the acceptable hit pattern, and the coincidence window after validation during which channels can contribute to the hit pattern. There is a check box for each possible hit pattern. For example, if the check box with pattern 0100 is checked, events with a hit in channel 2 and no others are accepted. Selecting multiple check boxes accepts combinations of hit patterns, e.g. any event with exactly one channel hit. For a detailed description of the coincidence operation, see section 7.2.1 of the user manual.

3.2.7 Module Coincidence Tab

The **Module Coincidence** tab is used to set parameters that affect the system as a whole. Examples are trigger distribution between modules, coincidence settings between modules, and the operation of the Pixie-4's front panel input. Controls for coincidences between modules are

- Accept event if the local pattern passes the local test (default),
- Accept event if the global hit pattern passes the global test, and
- Module's local test adds to global test (local fail adds to global fail).

See sections 7.2.2 and 7.6.2 of the user manual for details.

3.2.8 Histogram Tab

The **Histogram** tab contains basic functionality for histogram accumulation. There are two self-explanatory text boxes

- Minimum energy, to set the minimum energy to be histogrammed (bin 0 = min. energy)
- Binning factor to combine 2^N bins, which reduces the total number of bins but does not change the energy range in the spectrum.

3.2.9 Baseline Tab

The **Baseline** tab contains the controls for modifying baseline measurements. Section 6.4 addresses the role that the baseline plays in noise minimization. Once averaging values are input an improved estimate of the baseline cut can be obtained by highlighting the **Recompute Baseline Cut** button on the bottom right of the VI.

3.2.10 List Mode Spill Settings Tab

Pixie-4 only: The radio button **32 buffers** is used in conjunction with the PIXIE4DAQ VI and dumps 32 buffers to file in a single write operation. This mode results in much faster readout and lower dead time. The default is a single buffer.

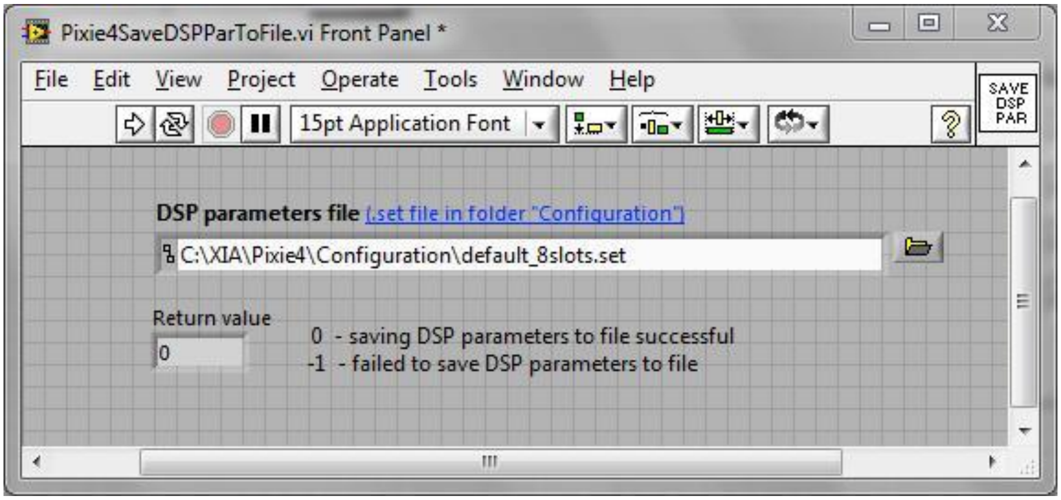
For explanation of the other options, please see the user manual and the online help.

3.2.11 Advanced Features

For explanation of all the other, advanced options, please see the user manual and the online help.

3.3 Setup Group - PIXIE4SAVEDSPPARTOFILE.vi

Changes to DSP parameters that are to be preserved for future runs must be saved to file using the PIXIE4SAVEDSPPARTOFILE VI as shown in Figure 3.9. A reboot via the PIXIEBOOT VI applies the parameters from the file to the module.


Pixie4SaveDSPParToFile.vi Front Panel *
File Edit View Project Operate Tools Window Help
15pt Application Font
SAVE DSP PAR

DSP parameters file (.set file in folder "Configuration")

C:\XIA\Pixie4\Configuration\default_8slots.set

Return value

0

0 - saving DSP parameters to file successful
-1 - failed to save DSP parameters to file

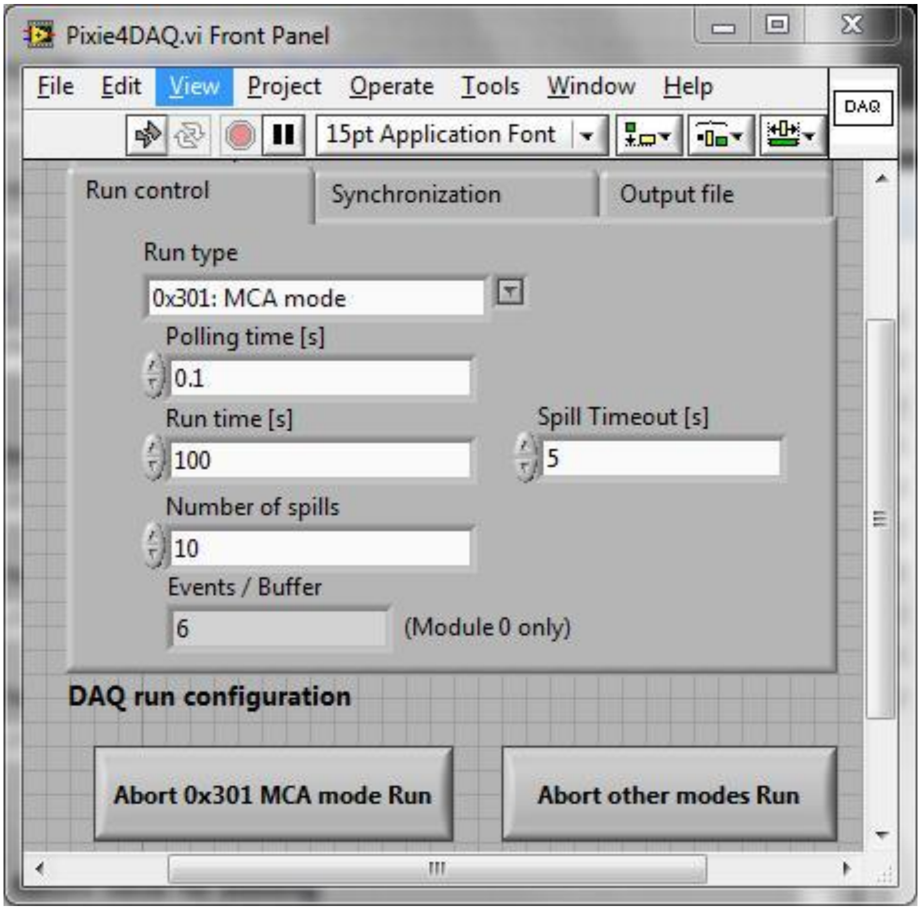
Figure 3-2 The PIXIE4SAVEDSPPARAMETERSTOFILE VI

3.4 Run Control - PIXIE4DAQ.vi

There are two Run Control VIs:

1. PIXIE4DAQ.vi for MCA runs and Pixie-4 list mode Run Types 0x100-103,
2. PIXIE4DAQE.vi for Pixie-4 Express and Pixie-500 Express list mode Run Type 0x400.

Both controls are essentially equivalent in controlling the data acquisition, but PIXIE4DAQE.vi is much simpler internally, making use of the interrupt driven data acquisition in the Pixie-4 Express architecture.



Run type: 0x301: MCA mode

Polling time [s]: 0.1

Run time [s]: 100

Spill Timeout [s]: 5

Number of spills: 10

Events / Buffer: 6 (Module 0 only)

DAQ run configuration

Figure 3-3: The PIXIE4DAQ VI

3.4.1 Run control tab

The **Run Control** tab in the PIXIE4BOOT VI defines the settings for data acquisition. This tab includes

- **Run type** to select MCA or list mode runs – refer to Section 4 for a detailed description.
- **Polling time** to set the polling time (period for checking if list mode data is available for readout and/or run time is reached),

- **Run time** to specify the duration of the run,
- **Number of spills** to specify the number of spills in list mode runs. (In list mode runs, data is accumulated in on-board memory until full, at which time it is read out by the host PC. Each readout is termed a spill. The number of spills thus sets the amount of data to collect.)
- **Spill Timeout** to set a timeout for each spill

3.4.2 Synchronization Tab

The controls include

- **Simultaneously stop/start modules** to stop/start all modules at the same time, and
- **Synchronize clocks** to synchronize acquisition between modules to set all times to zero at runstart.

This functionality is described in Section 7.2 of the user manual.

3.4.3 Output File Tab

In this tab, output files and paths are specified

- **Base name, Run number**
  The compete filename is formed from base name and 4-digit run number, e.g. test002.bin. File extensions are .bin and .dat for list mode data, .mca for spectra, .set for settings
- **MCA Data Path** to specify the location of the MCA files. If the default install location is C:XIA\Pixie4\ the MCA files are placed in C:\XIA\Pixie4\MCA
- **List Mode Data Path** to specify the location of the list mode files. If the default install location is C:XIA the list mode files are placed in C:\XIA\Pixie4\PulseShape

3.4.4 Data record options tab

The controls include

- **Auto increment run number** to increment the run number after each run,
- **Auto store spectrum data as binary .mca file** to write the MCA spectrum to file after each run,
- **Auto store setting in .set file after run** to record the settings after each run,
- **Auto store list mode data into .dat file** to convert the binary list mode data into a text file, and
- **New files after every xxx spills** to write a a file and start a new run after the specified number of spills.

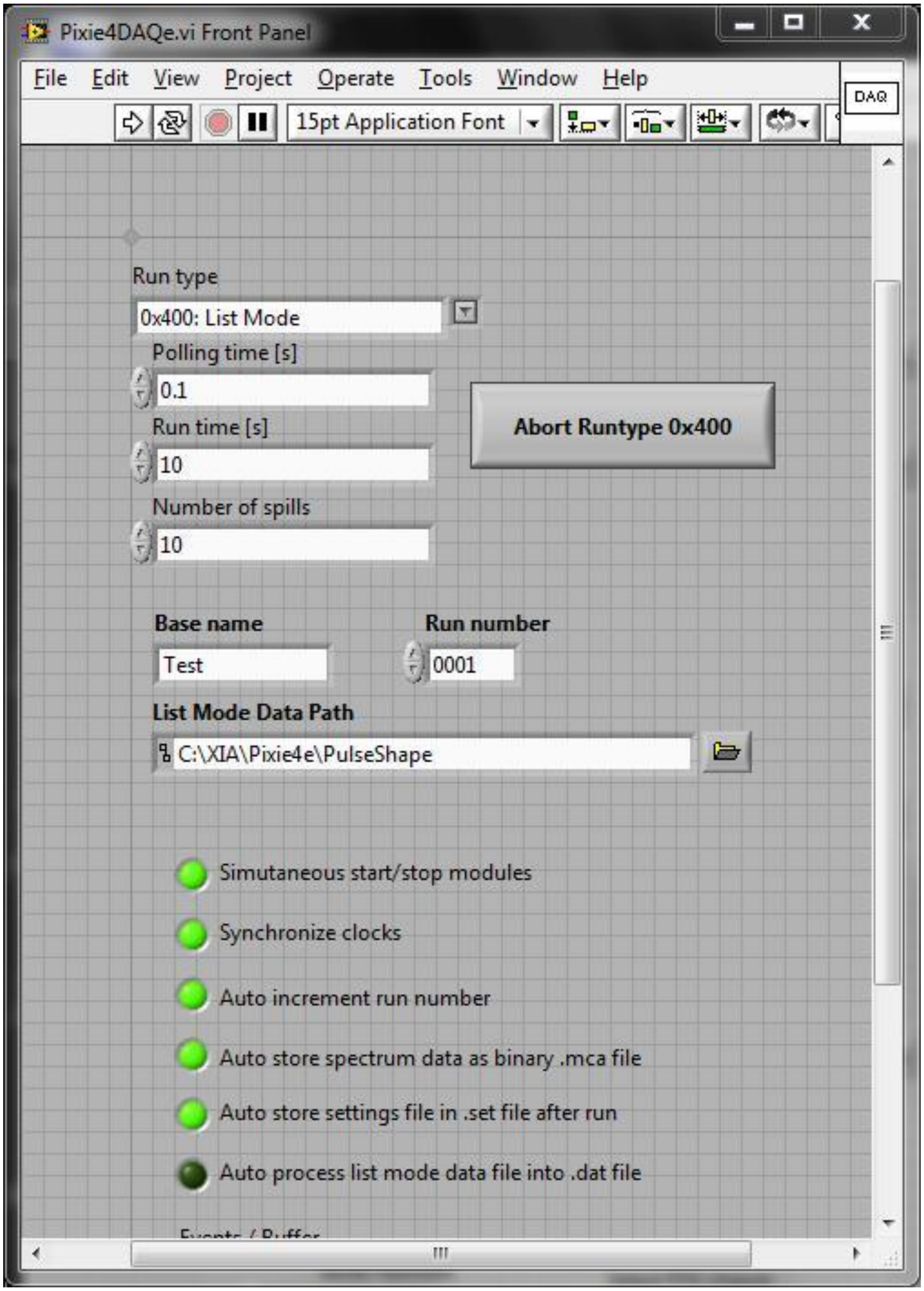
When taking long data acquisitions, it may be beneficial to break up the run into smaller sub runs. This helps to save data in case of power failure or system crashes, since only the most recent sub run is lost. Also list mode files tend to get large and unwieldy for analysis in longer runs.

3.5 Run Control - PIXIE4DAQE.vi

The PIXIE4DAQE VI has essentially the same controls as the PIXIE4DAQ VI, though simplified to the extent that

- Only list mode run type 0x400 is supported
- No multi-file runs are supported
- No Spill timeout is supported
- List mode data file names have the extension .b##, with ## being the 2-digit module number.

For a detailed description of the differences in the run types, please see the user manual.



Run type: 0x400: List Mode

Polling time [s]: 0.1

Run time [s]: 10

Number of spills: 10

Base name: Test

Run number: 0001

List Mode Data Path: C:\XIA\Pixie4e\PulseShape

● Simutaneous start/stop modules

● Synchronize clocks

● Auto increment run number

● Auto store spectrum data as binary .mca file

● Auto store settings file in .set file after run

○ Auto process list mode data file into .dat file

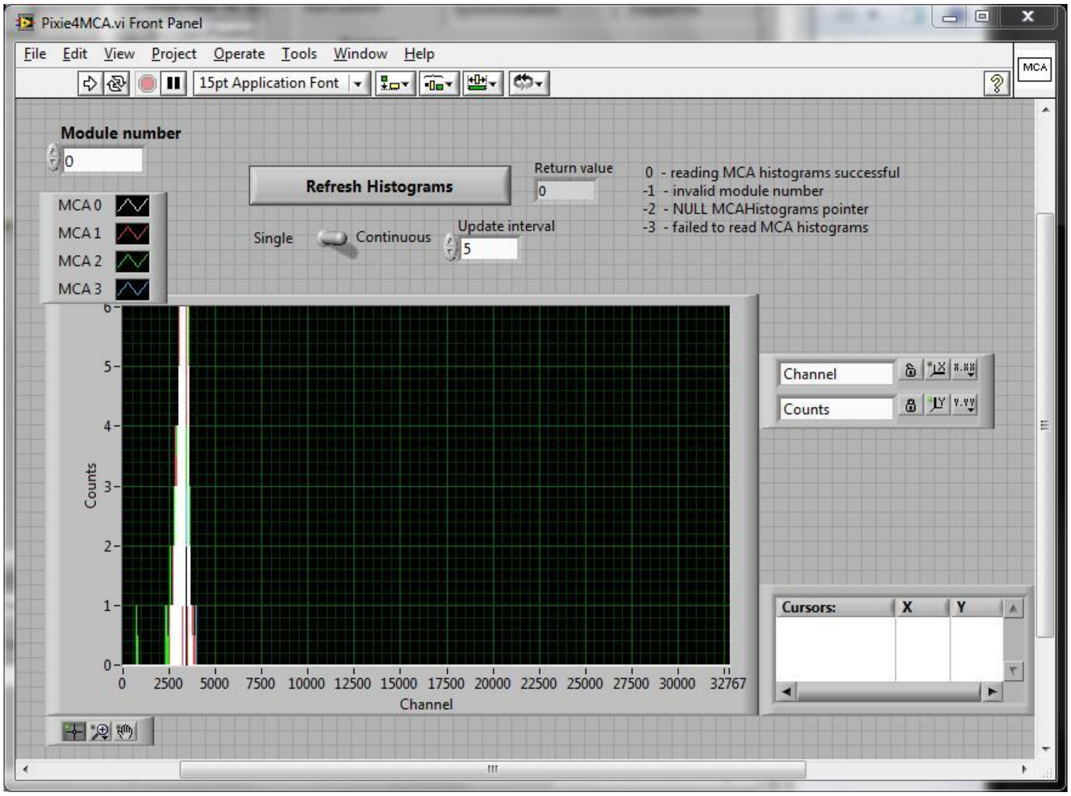
Figure 3-4: The Pixie4DAQe VI

3.6 Results Group

The Results group includes VIs to view the output data from the data acquisition in detail. It includes the Pixie4MCA, the Pixie4PulseShape and Pixie4ReadRunStatistics VIs.

3.6.1 MCA Spectrum (Pixie4MCA.vi)

The Pixie4MCA display shows the spectra accumulated in on-board memory. To observe the spectrum in real time the Pixie4MCA VI must be open and running simultaneously with the Pixie4DAQ VI. Alternatively, the MCA spectrum can be viewed after the run has ended. The spectrum is updated at a refresh interval set by **Update interval**. The MCA spectrum graph shows the MCA histograms for all four channels. The display can be rescaled by overwriting any of the text labels on the x- or y-axes. Multiple cursors can be inserted by right clicking in the **Cursor** box, and choosing **Create Cursor**. Once clicked, a cursor labeled *Cursor 0* is created and the user is prompted for *x* and *y* positioning. A second cursor can be similarly created. With the region of interest expanded the cursors can be dragged to either side of the peak to read off the FWHM. This VI operates only on a module's MCA memory. The saved .mca file (32bit unsigned integer binary numbers) can be imported into any suitable application.



Module number: 0

Return value: 0

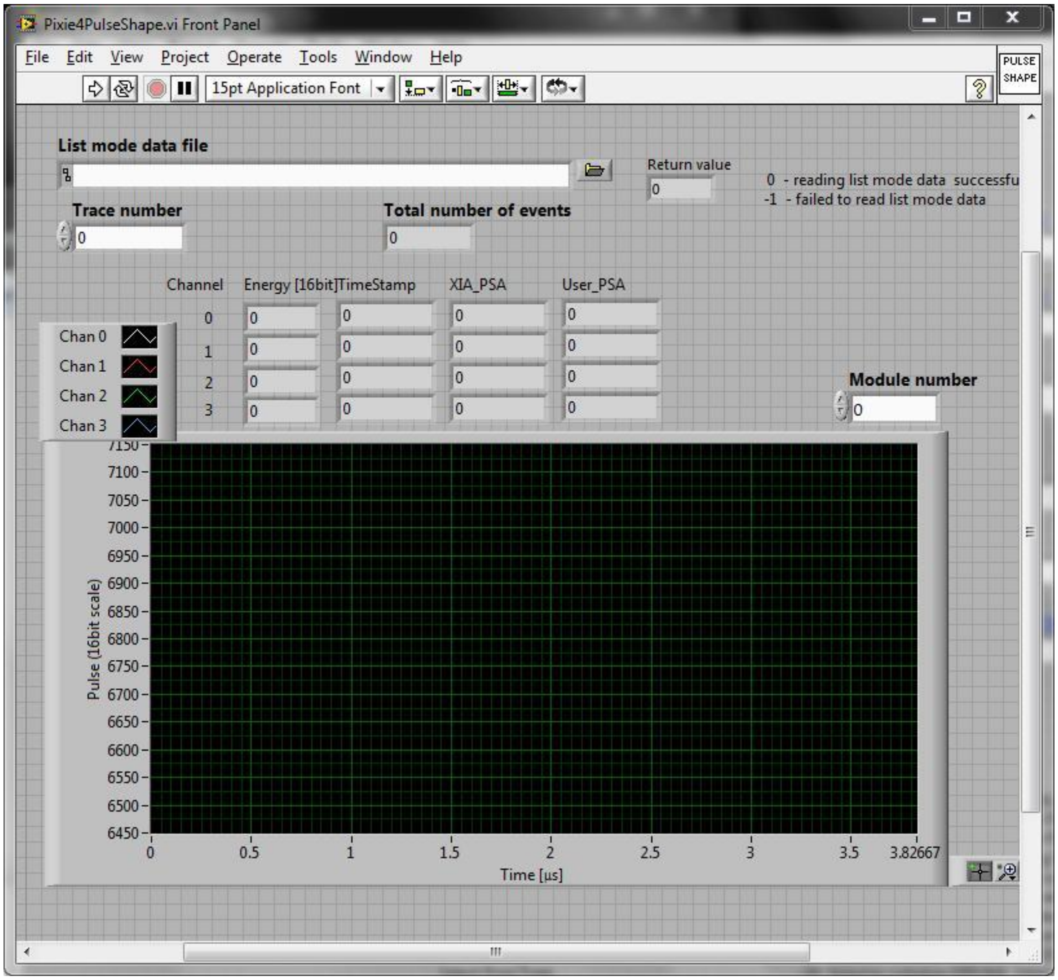
Update interval: 5

0 - reading MCA histograms successful
-1 - invalid module number
-2 - NULL MCAHistograms pointer
-3 - failed to read MCA histograms

Figure 3-5: The Pixie4MCA VI.

3.6.2 List Mode Traces (PIXIE4PULSESHAPE.vi)

The PIXIE4PULSESHAPE display shows data from the binary list mode files (.bin or .b##). If waveforms were collected, they are shown in the graph section of the panel. Event and channel header information – energy, time stamps, and hit patterns as described in section 4.1.2 – are shown in the fields above the graph section. As opposed to the PIXIE4MCA VI the PIXIE4PULSESHAPE VI operates only on saved output files. Data can only be viewed once a list mode run has terminated. However, one can open any .bin or .b## file, not just the one acquired during the last run. After specifying a data file, you can select an event to view by entering its number in the **Trace Number** field. To show the correct time units on the plot, be sure to select the appropriate **ADC sampling rate**.



List mode data file

Return value 0

0 - reading list mode data successfu
-1 - failed to read list mode data

Trace number 0

Total number of events 0

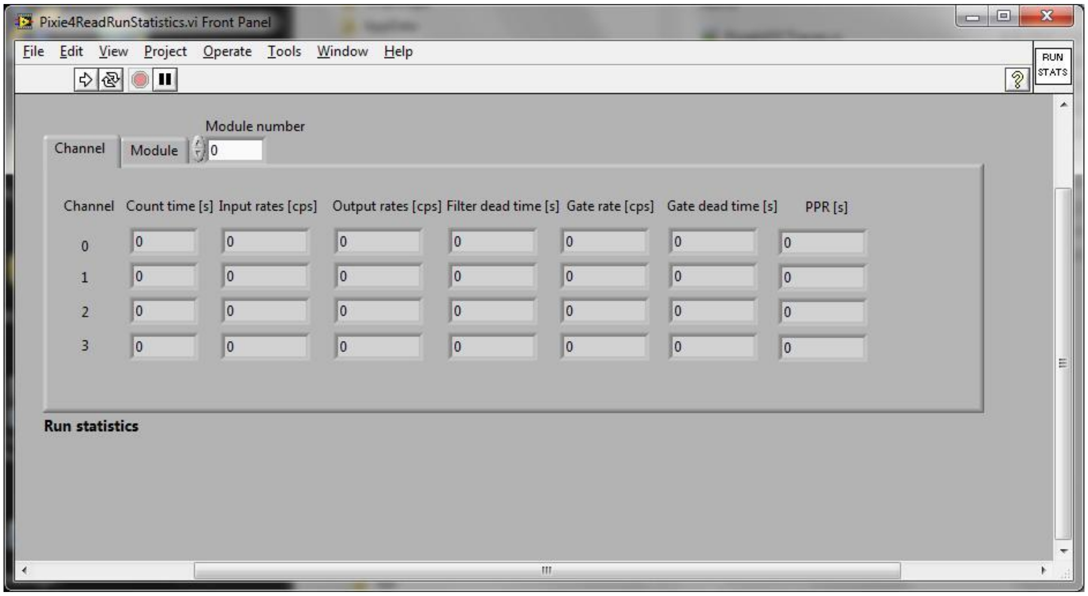
Channel	Energy [16bit]	TimeStamp	XIA_PSA	User_PSA
0	0	0	0	0
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0

Module number 0

Figure 3-6: The PIXIE4PULSESHAPE VI.

3.6.3 Run Statistics (Pixie4ReadRunStatistics.vi)

The Pixie4ReadRunStatistics VI shows the live times and count rates measured by the Pixie-4. The numbers can be updated by clicking the Run button. Otherwise, the statistics shown are that from the end of the run. For a detailed description of the definition of these values, see section 6.6. Figure 3.7 shows the Pixie4ReadRunStatistics VI



Module number: 0

Channel	Count time [s]	Input rates [cps]	Output rates [cps]	Filter dead time [s]	Gate rate [cps]	Gate dead time [s]	PPR [s]
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0

Figure 3-7: The Pixie4ReadRunStatistics VI

Appendices

Appendix A: LabVIEW Software Development

Introduction

The Pixie-4 LabVIEW demo interface was developed with National Instruments^{TM} LabVIEW 8.5. It has not been tested with other versions of LabVIEW, so compatibility with those versions can not be guaranteed at this time.

The Pixie-4 LabVIEW control software was built upon a Dynamic Link Library (dll) compiled from the Pixie C-Library code. The C library source code is distributed as part of the general software release, in sub-folder PixieCLib. A special wrapper layer was added to the dll compile so that LabVIEW can make direct calls to it. This layer is contained in the files `pixie4VI_DLL.c` and `pixie4VI_DLL.c.` The make file in the PixieCLib distribution can be called with option `–dlllib` to recompile the dll.

The dll thus contains the API functions described in the Pixie Programmer’s manual and additional wrapper function specifically for the LabVIEW Vis. These wrapper functions are described below.

Pixie-4 C-Library LabVIEW Wrapper Functions

There are currently 20 wrapper functions built into the Pixie-4 DLL. They are shown in the table below. These functions are generally straightforward calls to Pixie C library API functions, so the best documentation is the source code in `pixie4VI_DLL.c` combined with the Programmer’s manual to see which API function is called and what it is doing.

Function Name	Intended Usage
Pixie_Download_FileNames	Download file names from the host to the DLL
Pixie_Boot_Modules	Boot all the Pixie modules in the system
Pixie_Adjust_DCOffsets	Adjust DC-offsets in one Pixie-4 module
Pixie_Acquire_ADCWaveforms	Acquire untriggered ADC waveforms from one Pixie-4 module
Pixie_Start_MCARun	Start or resume MCA run in all Pixie-4 modules
Pixie_Start_ListModeRun	Start or resume list mode run in all Pixie-4 modules, run types 0x100-103
Pixie_Start_ListModeRun_400	Start or resume list mode run in all Pixie-4 modules, run type 0x400
Pixie_Check_RunStatus	Check run status in one Pixie-4 module, MCA run or list mode run 0x100-103
Pixie_Check_RunStatus_400	Check run status in one Pixie-4 module, list mode run 0x400
Pixie_Check_RunData*	Special polling task to return both number of spills recorded so far and new list mode data since the last poll
Pixie_Get_New_LMdata*	Special polling task to return new list mode data since the last poll, reading from file
Pixie_Stop_Run	Stop data acquisition run in all Pixie-4 modules

Pixie_Stop_Run_400	Stop data acquisition run in all Pixie-4 modules, run type 0x400-403
Pixie_Read_MCAHistograms	Read MCA histograms from the external memory in one Pixie-4 module
Pixie_Read_8KofMCAHistograms*	Read first N words of MCA histograms from the external memory in one Pixie-4 module
Pixie_Save_MCAHistograms	Read MCA histograms from the external memory in all Pixie modules and save the histograms to a file in binary format
Pixie_Save_ListModeData	Read list mode data from the external memory in all Pixie modules and save it to a file in binary format
Pixie_Save_Settings	Read DSP parameters from all Pixie modules and save these parameters to a file in binary format
Pixie_Write_User_Par	Write user parameter values to one Pixie-4 module
Pixie_Read_User_Par	Read user parameter values from one Pixie-4 module
Pixie_Read_ListModeData	Parse and read list mode data from a list mode data file

Table 1: Pixie-4 C-Library wrapper functions that were built into the Pixie-4 DLL. Functions marked with * are experimental.

LabVIEW Development Recommendations

For LabVIEW code development, it is recommended to set up the Pixie-4 Express with the Igor interface, save the settings to a new file, and load them into the LabVIEW interface by booting with the new file. This should bring the module into exactly the same state as in the Igor interface.

However, for the Pixie-4 Express it is no longer possible to operate Igor and LabVIEW interfaces in parallel. The PCIe drivers lock the “handle” for the device and the other interface can not boot the module.

Appendix B: LabVIEW Interface Changes from Version 2.6x

Previous releases of the LabVIEW demo interface for the Pixie-4 were based on software release 2.6x. By adding support for the Pixie-4 Express and Pixie-500 Express, the following changes were made to the interface. For full detail of the changes, please see the Programmer’s Manual and the “Pixie-4 Programmer’s Manual Differences”.

- In Pixie4Boot.VI, the number of boot file names was increased to 16. Only 10 are actually used, they all must be present (even those specific for a particular module type that is not in the system).
- In Pixie4Boot.VI, serial numbers must be specified for modules, not PXI slot numbers
- Pixie-4 Express and Pixie-500 Express modules use Run Type 0x400 for list mode data runs. A new VI for that run type was created (PixieDAQe.vi). It calls the new wrapper functions with suffix “_400”. These wrappers consistently require full Run Type and file name as arguments, the older wrappers did not as this was not required. The new wrappers are equivalent in function to the old wrappers. PixieDAQe.vi is simplified from the older PixieDAQ.vi since no cycles of check

run status – read data – resume run are required in Run Type 0x400; the readout of spills is handled by the interrupts in the C library and the LabVIEW program only polls for the number of spills saved so far.

- List mode data is organized as one file per module. The data format of the file is different from the Pixie-4 .bin files (see User Manual). The VI to view list mode data is unchanged, since the API functions to process list mode data files return equivalent data for the same function call.
  The API also provides a translator routine from new data format to old format so that existing data processing code can still be used. The function is not shown in a VI (yet).
- A number of new options and parameters have been added to Pixie4SetUserPar.vi. For details of the new parameters, please see the other manuals.
- In Pixie4ReadRunStatistics.vi, the parameter “live time” has been renamed to “count time” as the old name lead to misunderstandings.
- All other panels are unchanged

Please Note:

Settings files from version 2.6x are not compatible with version 4.xx. It is recommended to use one of the default files provided and change parameters to the old values. Most parameters are the same, but several have been expanded, removed or added, so that locations in the file shifted.