KMAX V1730 DAQ SYSTEM

User Manual

October 2020 | 2nd edition

A customized data acquisition system based on sparrow corporations' Kmax and CAEN waveform digitizers, developed by the University of Kentucky

Danula Godagama
danula.godagama@uky.edu

Contents

Overview

KMAX V1730 DAQ system is a toolsheet program designed using sparrow corporation's Kmax framework. It's designed to provide a complete interface for data acquisition with state-of-the-art features like live histograms and pulse plotting. The DAQ system includes CAEN V1730 digitizers, struck SIS3153 VME controllers and scalars. The layout of the DAQ system is shown below. This document is prepared to provide necessary guidance to users of the toolsheet program.

The complete source code for this program is available as a GitHub repository at the following link. https://github.com/Danulagod/Kmax-V1730-DAQ-System

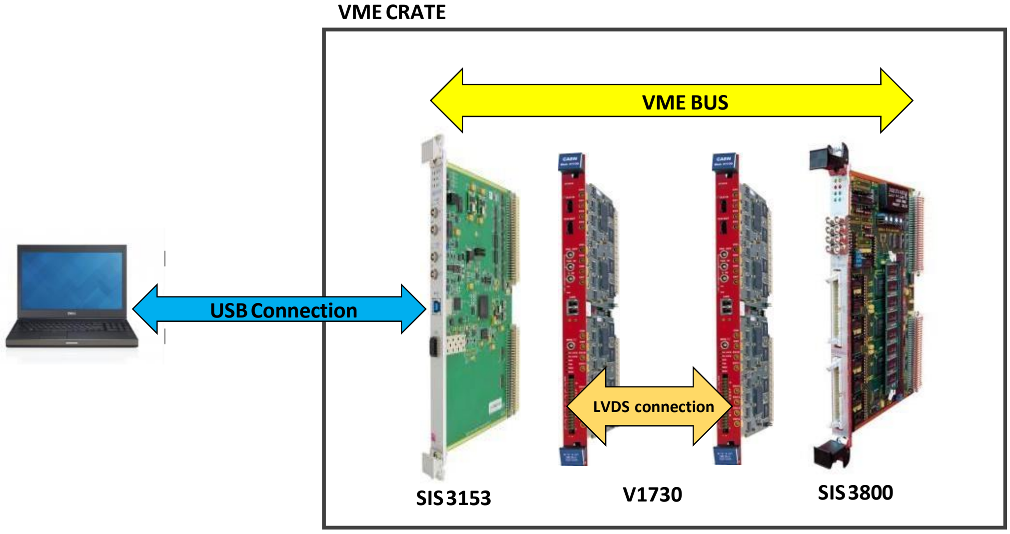

Figure 1: Layout of the DAQ system

Device installation

Installation of various modules in the VME crate should be done according to the manufacture’s specifications. The following are some general rules.

1) VME crate controllers must be installed in the 1st slot.

2) All modules should be pushed firmly against the VME slots and should be tightened with screws in the front panels of the modules before powering up the VME crate. Especially the struck SIS3153 controller is very hard to be installed in the WEINER VME 395 Mini crate.

3The proper function of the front panel LEDs of any module (digitizers or controllers) does not guaranty it's properly connected. Sometimes the following issue was observed. The front panel LEDs were lit up as described by the device user manuals, upon powering up the crate. But the modules were unable to perform VME operations because they were not properly connected to the back panel slots.

4) Unable to perform VME operations or abnormal behaviors during VME operations, most probably caused by connection issues between the modules and the back panel.

Kmax installation

Kmax framework is a Java-based application and requires the Java development kit 1.7 or later version to be installed on your computer. Then, the latest version of Kmax can be downloaded from the Sparrow website. The package comes as a zip file. It will create a folder called Kmax_Stuff once extracted. After that, the program can be opened by the terminal, CDing to Kmax_staff folder, and then executing the following command,

java -jar Kmax.jar

STRUCK SIS3153 driver installation

Kmax needs drivers to communicate with instrumentation devices. By default, Kmax does not come with drivers for struck SIS3153 VME controller.UK has developed a driver for SIS3153 to work alongside Kmax. Custom made drivers should be installed in proper locations before we can use the toolsheet in conjunction with the SIS3153 VME controller.

There are three components of the driver, and they should be installed in the following locations

1) SIS3153_Controller.mdr should be in Kmax_Stuff/Extentions/VME.MDRL

2) SIS3153.jar java driver file should be in Kmax_Stuff/Extensions/Linux-amd64

3) libSIS3153.so native driver file should be in Kmax_Stuff/Extensions/Linux-amd64

USB permissions

SIS3153 controller communicates with the host computer via a USB connection. If the user uses a non-administrator account of a Linux operating system, it might not have the necessary permissions to access the USB interface.

Do the following steps to set USB permissions.

type lsusb in the terminal. that will list all USB devices connected to the computer. Get the bus ID and device ID of the Struck SIS3153 device from the list. Then, type the following command as the administrator.

chmod 777 /dev/bus/usb/[bus ID]/[Device ID]

Note: The above procedure must be followed every time the user reconnects the USB port, reset the VME crate, or restart the DAQ computer. Alternatively, a USB rule can be included in the operating system granting users access to any STRUCK SIS3153 device. This is already done in UK's DELL DAQ laptop & GW's R540 server. Therefore, users have USB permissions by default.

Opening the toolsheet

Once the Kmax program started, the toolsheet can be opened using the file

menu. file>>Open Toolsheet and select V1730_DAQ_SIS3153.tlsh from the

dialog box or

file>>Open Recent Toolsheet>>V1730_DAQ_SIS3153.tlsh

Setting VME base addresses of the digitizers

SIS3153 controller communicates with other modules through the VME bus. all VME modules have a base address. It is usually set by the onboard rotary switches. Refer to the V1730 user manual for more information about setting the base address.

Then, these base addresses should be included in the toolsheet so that it can issue commands directed to digitizers. To do that, go to the editor panel of the toolsheet. Then, go to line 129 of the runtime code and type base addresses of each module as array elements of the array named BaseAddr_module. After that, save the toolsheet.

Compiling the toolsheet

The next step is to compile the toolsheet. The toolsheet can be compiled by just clicking the compile button in the toolbar. The status bar will show the message compile finished, following a successful compilation. Overwise error messages will be displayed.

Starting the toolsheet

The toolsheet program can be started by just clicking the go button in the toolbar. At this point, the toolsheet is ready to use. At the start, the toolsheet program will try to communicate with digitizer modules using the provided base addresses in line 129 of the runtime code. If that is successful, the toolsheet will issue commands to relocate the base address of each module, to enable the extended block transfer region. Default digitizer settings include a 4kB block transfer region which is easily overwhelmed by large event spaces. The block transfer region can be extended to 16mB which would require a base address relocation.

Please refer to the RelocateVMEAddress() method in the runtime code and the description of the register 0xEF00 in the V1730 register description manual(User manual UM 5118) for further information.

Control panel

The control panel is the part of the GUI that controls and monitors the data acquisition. The control panel is shown in the following figure. Run control section has Initialize, Start, and Stop buttons. The initialize button will prepare the application and the modules for acquisition and enable the start button. The start button will start the acquisition and enable the stop button. Finally, the stop button will stop the acquisition and enable the initialize button. This arrangement will prevent the user from disarranging the acquisition process by clicking the wrong button.

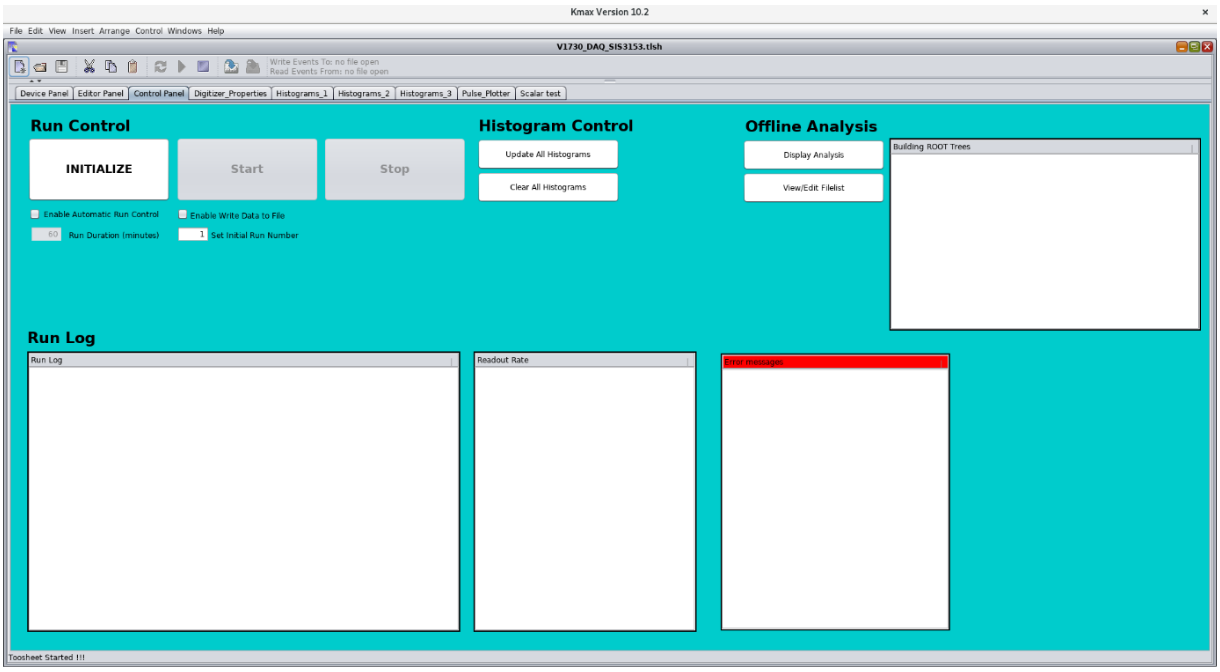

Figure 2: Control panel GUI

Setting the initial run number

Users should set the initial run number by using the set initial run number text box. The program will automatically increment the run number after that. At the start of the acquisition, if the program found an existing data file with the same run number, the program is designed to halt without overwriting data.

Saving data vs caching

If the user intends to save the data, enable write data to file checkbox should be enabled. If it's not enabled, data will be written to a temporary cache file. That option is suitable for test runs intended to observe the pulses and live histograms.

Automatic run control

The program has the option to automate the run control. Once this option is activated by enabling the enable automatic run control checkbox the program will stop the ongoing run and start a new run after a period set by the run duration textbox.

Data files

Data are saved in Kmax_Stuff/DATA folder as binary files. The DATA folder has subfolders for each digitizer module. Each module will have a data file for each run. The data files have the event structure described by the V1730 user manual. Please refer to the event structure section of the user manual (UM 2792) for more information.

Log files

The program keeps logs each run in an ASCII file. It creates a new log file for each day with the date as the file name. The log files are saved in the folder Kmax_Stuff/LOGS. The content of the current log file is displayed in the run log text pane of the control panel.

ROOT Tree files

The program generates ROOT based tree files from raw data which will be used in the offline analysis. The Tree files are generated using a C++ program named offline which is located at Kmax_Stuff/offline. Offline program performers charge integration and CFD algorithms on raw data and save the output in a tree. In addition to this, 48bit trigger time stamps are stored in the tree. Tree files are stored in Kmax_Stuff/Replays. The toolsheet program automatically runs the C++ program at the end of each run to generate the tree. The progress and the output messages from this process are displayed in text pane in the control panel. This runs as an external process and doesn't affect the data acquisition.

offline code is located at Kmax_Stuff/offline/offline.cxx. If the user made changes to this code, it needs to be recompiled. It can be compiled by typing the following command in the terminal.

```
g++ offline_back.cxx -o offline `root-config --cflags --glibs`
```

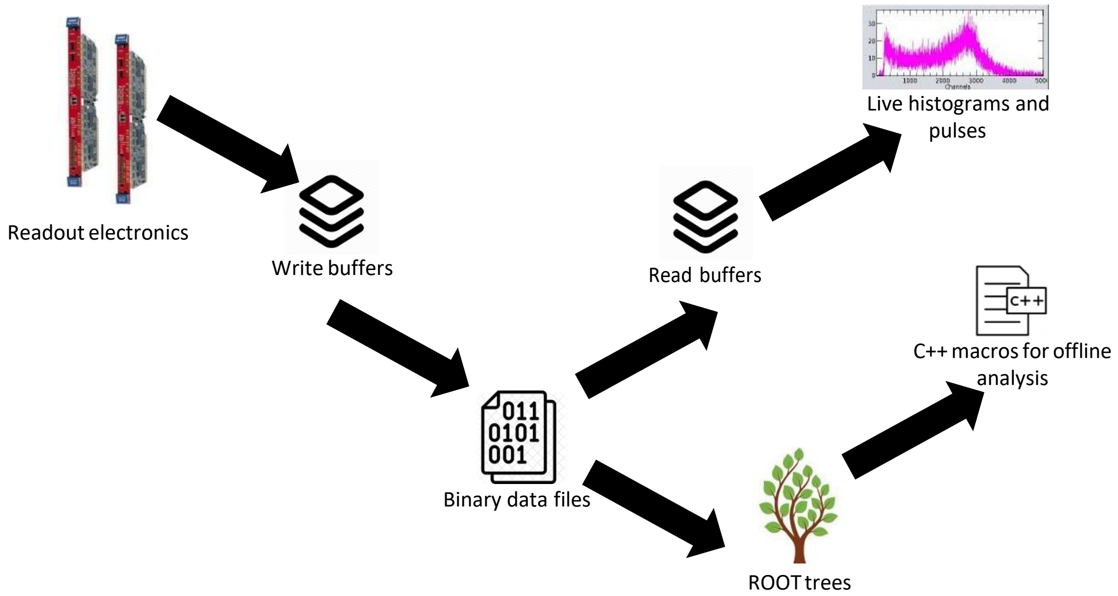

Figure 3: Data storage structure of the toolsheet program

Error messages

Fatal errors during the acquisition will be displayed in the message box in the control panel. The original purpose of this message box is to inform the developers about bugs. Therefore, during normal operations, no errors should be displayed there. Upon a display of an error message, it's best to stop the acquisition and investigate the issue before continuing.

Live histograms

The toolsheet program has several panels that contain histograms. Each panel has 16 histograms corresponds to 16 channels of a V1730 module. These histograms will automatically update every minute during the data acquisition (The user can change the update time at line 2216 in the runtime code). Also, the user can manually update and clear histograms using buttons in the control panel.

Pulse plotter

The toolsheet program has the live pulse plotting feature in which the user can observe digitized pulses from any channel. It can be enabled using the enable plotting checkbox in the pulse plotter panel. The user can observe up to 7 channels simultaneously. Channels can be selected using the channel selection combo boxes. Pulses will be displayed in individual plots on the left. These individual plots can be overlaid in the overlay plot on the right using the checkboxes next to the channel selection combo boxes. Plots display one pulse per second regardless of the trigger rate.

Digitizer properties

Digitizer properties tab provides a user-friendly GUI to access and edit all digitizer settings. Also, it allows saving the settings to the disk(as a config file) and reloading them for recurring use.

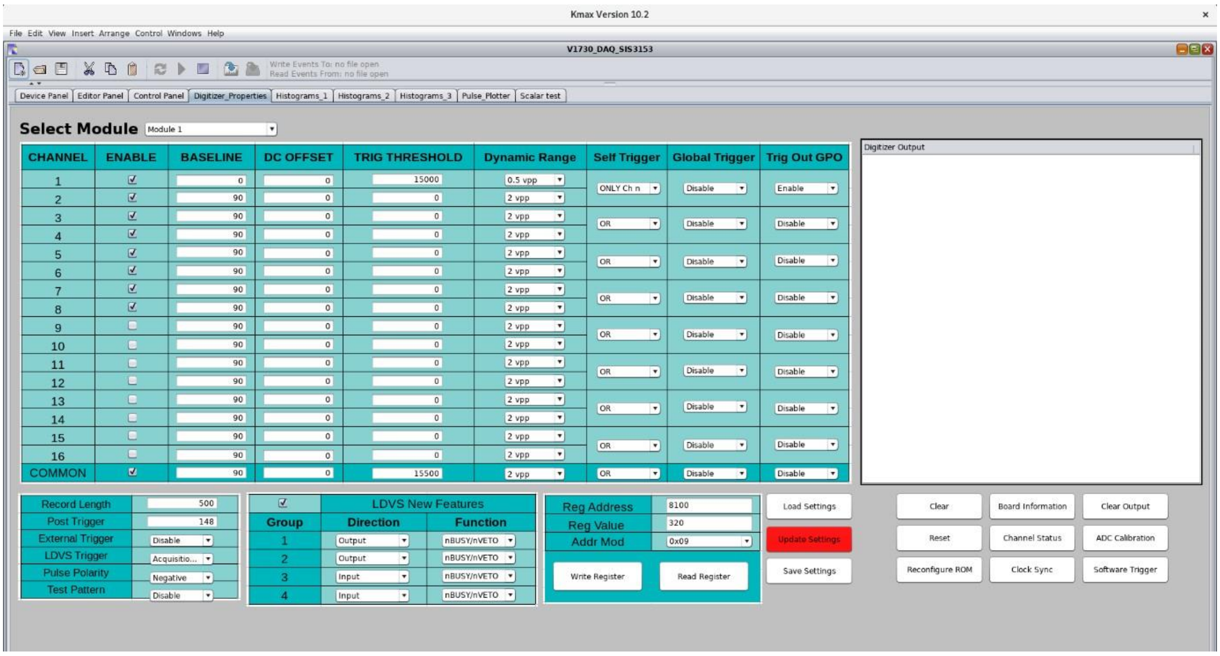

Figure 4:Digitizer properties GUI

Channel Enable

options are yes or no. when enabled channels are included in the readout

Trigger Threshold

to be set with LSB(least significant bits) units. This value is relative to the absolute zero

DC offset

DC offset value should be selected so that the entire pulse in the ADC dynamic range. ADC scale ranges from 0 to 16383 (2^{14}-1). But the DAC controlling the DC offset is 16 bits and therefore ranges from 0 to 65535. DAC settings close to 0 and 65535 are out of range of the channel ADC. The following figure shows the position of the ADC range with respect to the DC offset values.

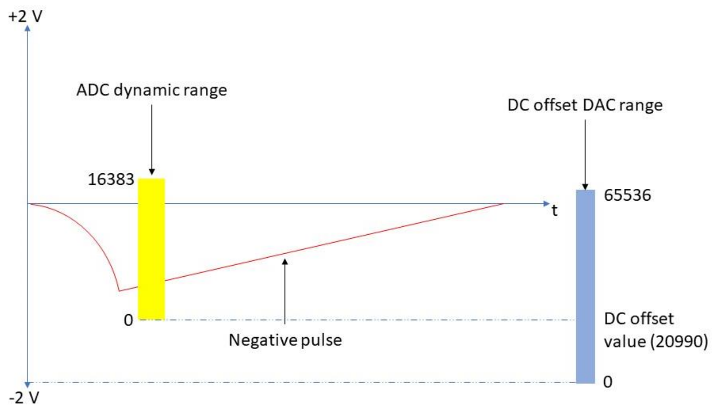

Figure 5:ADC dynamic range & DC offset

Dynamic Range
Options are 0.5 Vpp or 2.0 Vpp

Self Trigger
Channels are coupled in the trigger logic of V1730. That means 8 couples in a 16 channel V1730 and each couple can have one of the following options.
OR, AND, only channel n and only channel n+1

Global Trigger
Options are enabled or disabled. When enabled couple will be part of the global trigger which start the acquisition

Trigger Out GPO
Options are enable or disable. When enabled channels will be part of the trigout signal which will be propagated through the TRIGOUT port or LDVS trigger out. This signal can be used to create trigger logic for DAQ systems with multiple digitizers

Record Length
Set the number of samples in the acquisition window

Post Trigger
Set the number of post-trigger samples. This value must be less than the record length

External Trigger
Options are disabled, acquisition only, TRIGOUT only, acquisition and TRIGOUT

LVDS Trigger
Options are disabled, acquisition only, TRIGOUT only, acquisition and TRIGOUT

Pulse Polarity
Options are Positive or Negative

Once the settings have been set, the user needs to write the settings to the digitizer by pressing the update settings button. Also, the user can save settings in a configuration file by clicking the save settings button.

Synchronization between multiple digitizer modules

A single CEAN V1730 digitizer has 16 channels. But some nuclear physics experiments require more than 16 channels. Therefore, DAQ systems should be capable of facilitating multiple digitizer modules that run in synchronization. A properly synchronized collection of modules (with n modules) acts as a single module with 16xn channels.

The synchronization is obtained by achieving the following.

1) The same clock is distributed to all the modules.
2) Start and stop of acquisition done simultaneously in all the modules.
3) In case of one or more modules become busy hence can't accept triggers, veto the acquisition until all the modules become available.

Clock distribution among digitizer modules

V1730 digitizers come with a 50mHz internal oscillator. They also have input and output connectors to propagate clock signals in a daisy chain. The board clock can be synthesized using either the internal oscillator or a clock input signal. The modules have an onboard dip switch to select the clock source (Please refer to page 23 of the V1730 user manual).

When the same clock needs to be distributed among several digitizers, one module is selected as the master clock module. The master module uses its internal oscillator to generate the clock and outputs a 62.5 MHz clock signal through its CLKOUT connector. The 1st slave module receives the clock signal through its CLKIN connector and generates its board clock using the incoming signal. Also, it outputs the incoming 62.5 MHz signal through its CLKOUT connector. Subsequent slave modules should be daisy-chained to receive and output the incoming signal through its CLKIN/CLKOUT connectors.

There is a propagation delay of the clock signal. The delay is about 2.7 ns from one module to the subsequent module in the daisy chain. It's necessary to offset this delay by a phase logic loop (PLL) upgrade.

CAENUpgrader program can be used to perform PLL upgrades on digitizers. It's a GUI tool provided by CAEN to perform tasks like firmware upgrades and PLL reconfigurations on CAEN manufactured modules. The PLL upgrade is done by flashing the digitizer ROM with the correct binary file. These binary files come with the CAENUpgrader program (have the extension .rbf). The user should select the appropriate binary file at the time of the upgrade. The following steps demonstrate how to proceed with the upgrade.

CAENUpgrader program uses a dynamic library called CAENComm(libCAENComm.so) to communicate with the devices. The library is specifically designed to work with CAEN manufactured devices. In this DAQ system, the VME controller which provides the communication interface between the readout computer and the VME bus is made by a different manufacturer(STRUCK). In this case, the user should reimplement CAENComm library functions to support the SIS3153 controller. The re-implemented library is available to download at GitHub along with the source code and installation instructions.

1) Master module PLL upgrade

program PLL to generate board clock using the internal 50 MHz signal and propagate 62.5 MHz signal on CLKOUT.

CAENUpgrader -> Upgrade PLL ->v1730_in50_out62.5_delay_2.5ns.rbf

2) Slave modules PLL upgrade

program PLL to generate board clock using the incoming 62.5 MHz signal and propagate 62.5 MHz signal on CLKOUT.

CAENUpgrader -> Upgrade PLL ->v1730_in62.5_out62.5_delay_2.5ns.rbf

3) Power cycle the VME crate.

Turn off the create. Then, adjust the clock source dip switches(SW3) on digitizers.

set the switch to INT(internal) position in the master module.

set the switch to EXT(external) position in slave modules.

Connect the clock distribution cables and turn on the VME crate.

Upon a successful upgrade, PLL-LOCK LED should be on in all modules

and CLK-IN LED should be on in slave modules.

4) Check clock signals on the oscilloscope
Exit the CAENUpgrader program and open the DAQ toolsheet. (keep in mind that CAENUpgrader and the DAQ toolsheet both use the same dynamic library. Running both programs at the same time may cause errors). Using the digitizer properties tab in the toolsheet, program the digitizers to propagate the clock signal on the TRG-OUT connector (write 0x50000 to register 0x811C). Then, Connect the TRG- OUT to the oscilloscope. There should be no jitter between master and slave clock signals if the synchronization is successful.

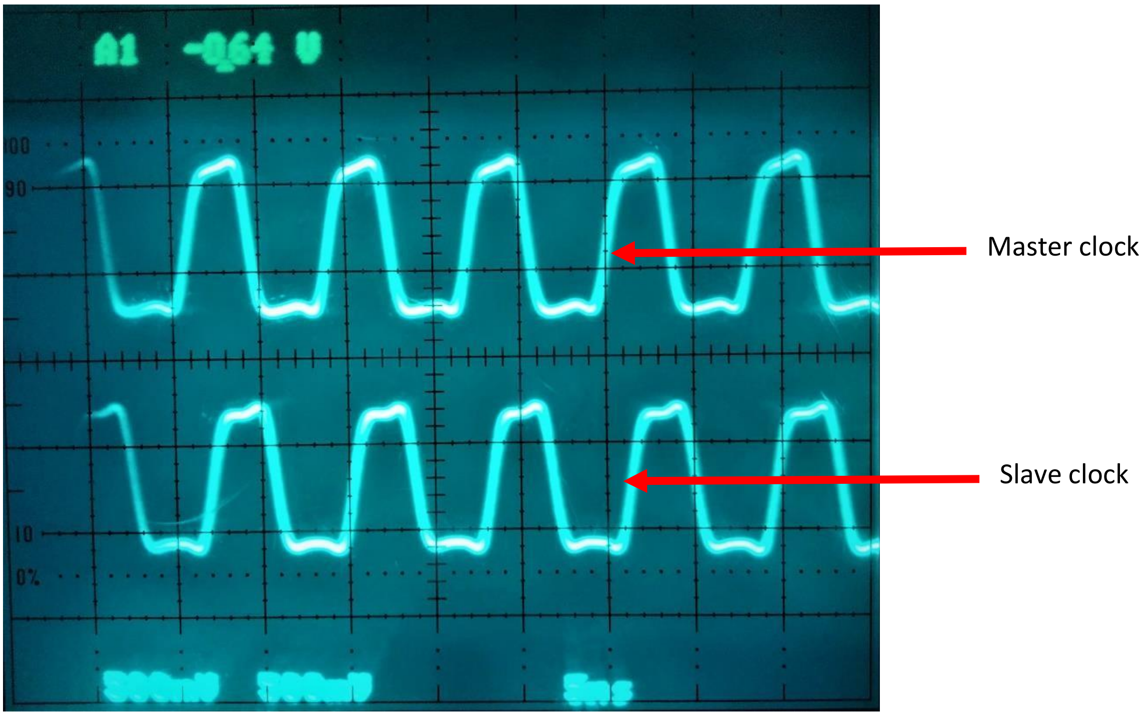

Figure 6:The oscilloscope view. Oscilloscope triggered by the master clock signal. No jitter between master and slave clock.

Simultaneous start and stop of the acquisition

All digitizers which run in synchronization mode should start and stop at the same time to ensure the event alignment. The acquisition can be started using either a software command or an LVDS signal or at the reception of the first trigger. Also, the start logic can be propagated through the LVDS connections or S-IN/TRG-OUT NIM connecters. It is necessary to program the digitizers to offset the propagation delay. Please refer to the V1730 register description for more information (description of the register 0x8170 in the document UM 5118). There are several ways to implement the start logic. Usually, it is specific to the DAQ system.

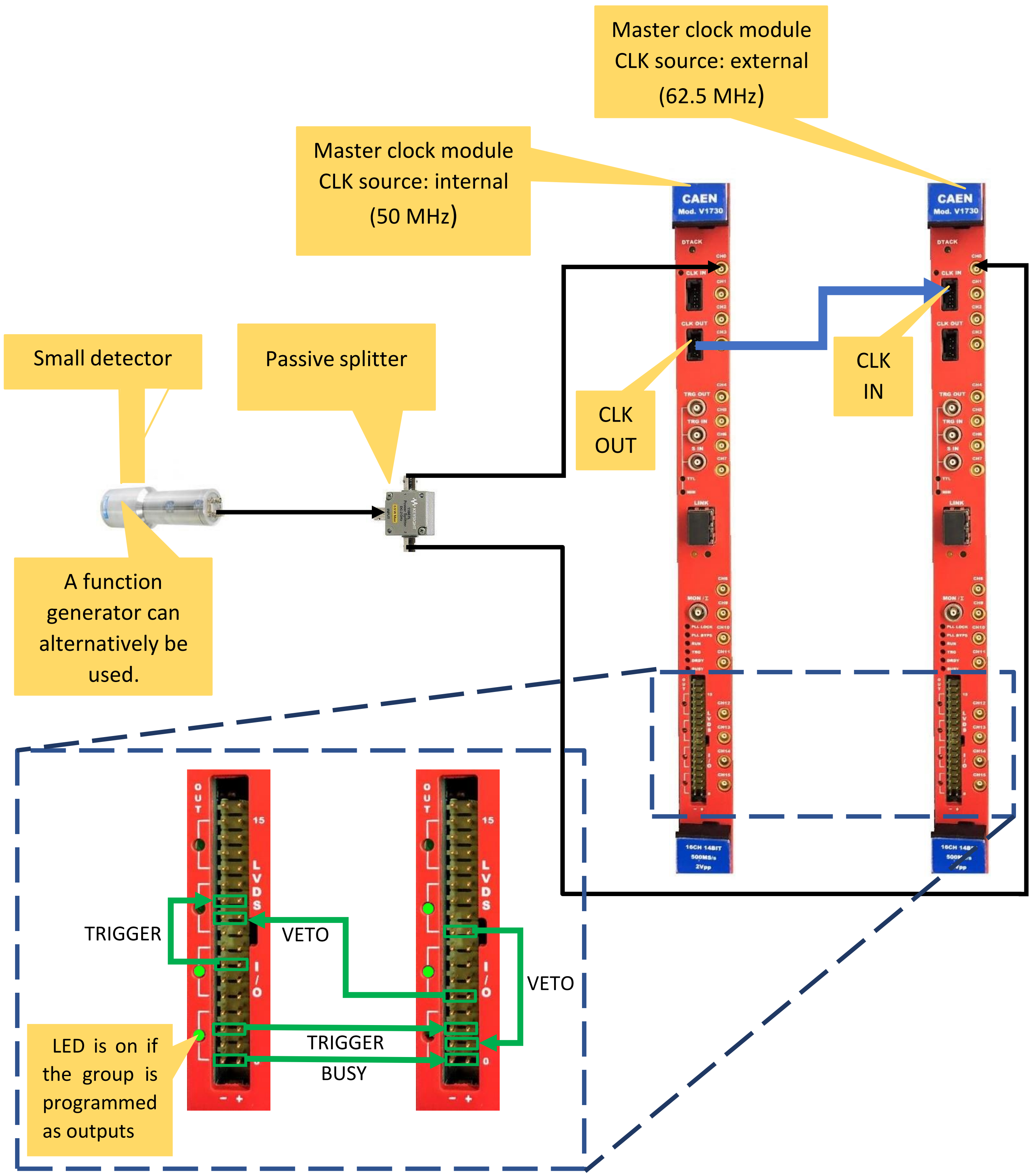

Figure 7:Layout of the multi-module synchronized DAQ system

In the above demonstration, two digitizers run in synchronization mode. The signal from a small detector is fed to channel 1 of each digitizer via a passive splitter. The first module generates a trigger request upon the input signal on channel 1 passes a programmable threshold. The trigger is then propagated to the second module as well as to the first module through LVDS connections. LVDS connections also propagate the BUSY and VETO signals to keep the event alignment. Digitizers start to capture the event upon the reception of the LVDS trigger. All digitizer settings for this setup are described in table 01 in the appendix.

SyncTest.C is a ROOT macro written to test the synchronization among multiple digitizers. It can be used to test data taken with the above digitizer configuration. It histograms the difference between trigger time tags (TTT), Time resolution, and charge deposition correlation between two digitizers. The macro output is shown in the following figure.

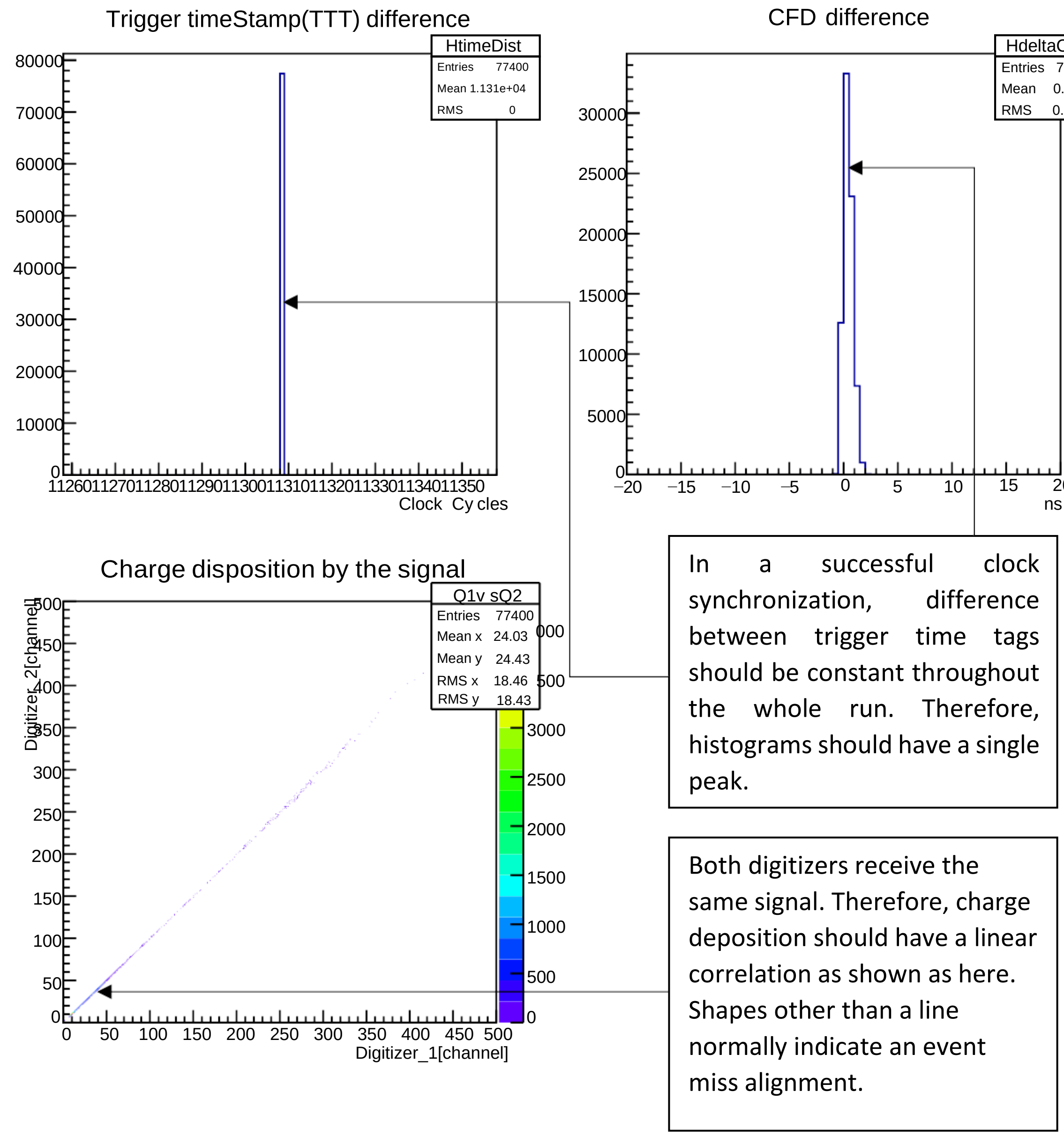

Figure 8:Output from the SyncTest macro

Appendix

Table 1:Complete digitizer settings for the demonstration

<table>
<tr><th colspan="2">Setting</th><th colspan="2">Module 1</th><th colspan="2">Module 2</th></tr>
<tr><td colspan="2">Enable</td><td colspan="2">Channel 1</td><td colspan="2">Channel 1</td></tr>
<tr><td colspan="2">DC offset</td><td colspan="2">0</td><td colspan="2">0</td></tr>
<tr><td colspan="2">Trig threshold</td><td colspan="2">15000</td><td colspan="2">0</td></tr>
<tr><td colspan="2">Dynamic range</td><td colspan="2">2.0 V</td><td colspan="2">2.0 V</td></tr>
<tr><td colspan="2">Channel self trigger</td><td colspan="2">Only channel 1</td><td colspan="2">-</td></tr>
<tr><td colspan="2">Channel global trigger</td><td colspan="2">All disabled</td><td colspan="2">All disabled</td></tr>
<tr><td colspan="2">Channel trigout trigger</td><td colspan="2">Enabled for channel 1</td><td colspan="2">All disabled</td></tr>
<tr><td colspan="2">Record length</td><td colspan="2">500</td><td colspan="2">500</td></tr>
<tr><td colspan="2">Post trigger</td><td colspan="2">148</td><td colspan="2">148</td></tr>
<tr><td colspan="2">External trigger</td><td colspan="2">Disabled</td><td colspan="2">Disabled</td></tr>
<tr><td colspan="2">LVDS trigger</td><td colspan="2">Acquisition only</td><td colspan="2">Acquisition only</td></tr>
<tr><td colspan="2">Pulse polarity</td><td colspan="2">Negative</td><td colspan="2">Negative</td></tr>
<tr><td colspan="2">Test pattern</td><td colspan="2">Disabled</td><td colspan="2">Disabled</td></tr>
<tr><td rowspan="5">LVDS settings</td><td>LVDS new features</td><td colspan="2">Enabled</td><td colspan="2">Enabled</td></tr>
<tr><td>1</td><td>Output</td><td>nBUSY/nVETO</td><td>Input</td><td>nBUSY/nVETO</td></tr>
<tr><td>2</td><td>Output</td><td>nBUSY/nVETO</td><td>Output</td><td>nBUSY/nVETO</td></tr>
<tr><td>3</td><td>Input</td><td>nBUSY/nVETO</td><td>Output</td><td>nBUSY/nVETO</td></tr>
<tr><td>4</td><td>Input</td><td>nBUSY/nVETO</td><td>Input</td><td>nBUSY/nVETO</td></tr>
<tr><td colspan="2">Clock source</td><td colspan="2">Internal</td><td colspan="2">External</td></tr>
<tr><td colspan="2">PLL file</td><td colspan="2">v1730_in50_out62.5_delay_2.5ns.rbf</td><td colspan="2">v1730_in62.5_out62.5_delay_2.5ns.rbf</td></tr>
</table>

Figure 9: SyncTest output of an unsuccessful synchronization. Double peaks indicate an issue related to the clock synchronization. Check the PLL configuration of the digitizers.

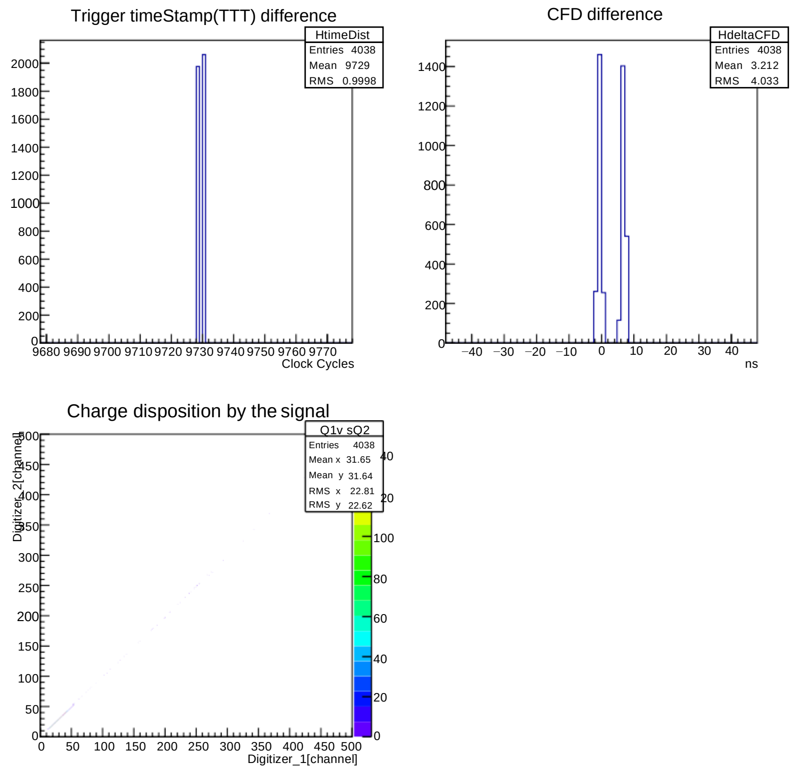

Figure 10: SyncTest output of an unsuccessful synchronization. Square like artifact in the 2D histogram indicate an event miss alignment. Check BUSY/VETO LVDS connections