CAEN
Tools for Discovery

Electronic Instrumentation

User Manual UM2784

CAENDigitizer LabVIEW Library

LabVIEW VI for CAEN Digitizers high level management

Rev. 1 - 15 January 2015

Purpose of this User Manual

This User Manual contains the full description of the CAENDigitizer LabVIEW library.

Change Document Record

Date	Revision	Changes
11 April 2013	00	Initial release
15 January 2015	01	Added support to Demos, updated DPP functions, added HV VI for high voltage management

Symbols, abbreviated terms and notation

ADC	Analog to Digital Converter
DPP	Digital Pulse Processing
FFT	Fast Fourier Transform
FSR	Full Scale Range
OS	Operating System
SBC	Single Board Computer

Reference Document

[RD1] UM1935 - CAENDigitizer User & Reference Manual

[RD2] GD2783 – First Installation Guide to Desktop Digitizers & MCA

[RD3] Technical Information Manual of V1718 and VX1718 VME – USB2.0 Bridge

[RD4] Technical Information Manual of A3818 PCI Express Optical Link Controller

[RD5] Technical Information Manual of A2818 PCI Optical Link Controller

[RD6] UM1934 - CAENComm User & Reference Manual

[RD7] AN2472 - CONET1 to CONET2 migration Application Note

[RD8] UM2606 – DT5780 Digital MCA

[RD9] UM3188 - DT5790 Digital Pulse Analyzer User Manual

[RD10] UM2580 – DPSD User Manual

[RD11] UM2085 – DPCI User Manual

[RD12] UM3182 – MC^{2}Analyzer User Manual

All documents can be downloaded at: http://www.caen.it/csite/LibrarySearch.jsp

CAEN S.p.A.
Via Vetraia, 11 55049 Viareggio (LU) - ITALY
Tel. +39.0584.388.398 Fax +39.0584.388.959
info@caen.it
www.caen.it

© CAEN SpA – 2015

Disclaimer

No part of this manual may be reproduced in any form or by any means, electronic, mechanical, recording, or otherwise, without the prior written permission of CAEN SpA.

The information contained herein has been carefully checked and is believed to be accurate; however, no responsibility is assumed for inaccuracies. CAEN SpA reserves the right to modify its products specifications without giving any notice; for up to date information please visit www.caen.it.

Index

Purpose of this User Manual 2
Change Document Record 2
Symbols, abbreviated terms and notation 2
Reference Document 2
Index 4
List of Figures 6
List of Tables 6
1 Introduction 7
Drivers & Libraries 8
Drivers 8
Libraries 8
Installation 10
Return codes 11
2 Communication 12
OpenDigitizer 12
CloseDigitizer 13
GetInfo 13
Reset 14
WriteRegister 14
ReadRegister 14
Calibrate 15
Interrupt configuration 15
Set / GetInterruptConfig 16
IRQWait 17
VMEIRQWait 17
VMEIRQCheck 18
IACKCycle 18
VMEIACKCycle 19
RearmInterrupt 19
Data Readout 20
ClearData 20
DisableEventAlignedReadout 20
Set / GetMaxNumEventsBLT 21
ReadData 22
GetNumEvents 23
GetEventInfo 23
DecodeEvent 24
LoadDRS4CorrectionData 25
Enable/Disable DRS4Correction 25
3 Trigger configuration 26
SendSWtrigger 26
Set / GetSWTriggerMode 26
Set / GetExtTriggerInputMode 27
Set / GetChannelSelfTrigger 27
Set / GetGroupSelfTrigger 28
Set / GetChannelGroupMask 28
Set / GetChannelTriggerThreshold 29
Set / GetGroupTriggerThreshold 29
Set / GetChannelPulsePolarity 30
Set / GetRunSynchronizationMode 30
Set / GetIOLevel 30
Set / GetTriggerPolarity 31
Set / GetGroupFastTriggerThreshold 31
Set / GetGroupFastTriggerDCOffset 32
Set / GetFastTriggerDigitizing 32
Set / GetFastTriggerMode 33
Set / GetDRS4SamplingFrequency 33
Set / GetOutputSignalMode 33

4 Acquisition 34
Set / GetChannelEnableMask 34
Set / GetGroupEnableMask 34
SWStartAcquisition 35
SWStopAcquisition 35
Set / GetRecordLength 35
Set / GetPostTriggerSize 36
Set / GetAcquisitionMode 36
Set / GetChannelDCOffset 37
Set / GetGroupDCOffset 37
Set / GetDESMode 38
Set / GetAnalogMonOutput 38
Set / GetAnalogInspectionMonParams 39
Set / GetEventPackaging 39
Set / GetZeroSuppressionMode 40
Set / GetChannelZSParams 41
Acquisition example 42
5 DPP specific VIs 45
Set / GetDPPPreTriggerSize 45
GetDPPEvents 46
DecodeDPPWaveforms 47
SetDPPEventAggregation 48
Set / GetNumEventsPerAggregate 48
Set / GetMaxNumAggregatesBLT 49
SetDPPParameters 50
Set / GetDPPAcquisitionMode 51
Set / GetDPPTriggerMode 52
Set / GetDPP_PHA_VirtualProbe 53
Set / GetDPP_PSD_VirtualProbe 54
Set / GetDPP_CI_VirtualProbe 55
6 HV specific VIs 56
Read_HV_Status 56
Read_HV_VMon 57
Read_HV_IMon 57
Read_HV_VExt 57
Read_HV_TRes 58
Set / Get_HV_Power 58
Set / Get_HV_VSet 58
Set / Get_HV_ISet 59
Set / Get_HV_RampUp 59
Set / Get_HV_RampDown 59
Set / Get_HV_VMax 60
Set / Get_HV_PWDownMode 60
Set / Get_HV_MonitorMode 61
Get_HV_ChannelsInfo 61
Reset_HV 62
7 DEMOs 63
Digitizer Demo for firmware standard 63
DPP-PSD / DPP-CI Demos 66
DPP-PHA Demo for 724 series, DT5780, and DT5781 70
8 Examples of communication settings 73
Example No.1 73
Example No.2 74
Example No.3 75
Example No.4 77
Example No.5 78
9 Technical support 79

List of Figures

Fig. 1.1: Hardware and Software layers 9
Fig. 4.1: Acquisition example control panel 42
Fig. 4.2: Block Diagram Part #1.....43
Fig. 4.3: Block Diagram Part #2.....44
Fig. 4.4: Block Diagram Part #3.....44
Fig. 8.1: Connection example no.1. 73
Fig. 8.2: Connection example no.2. 74
Fig. 8.3: Connection example no.3. 75
Fig. 8.4: A2818 network scheme. 76
Fig. 8.5: Connection example no.4 77
Fig. 8.6: Connection example no.5 78

List of Tables

Tab. 1.1: Return codes table 11

1 Introduction

CAEN has developed a family of Sampling ADCs modules with different form factors (VME, NIM and Desktop). They all provide the possibility to be handled and readout by a host PC via different communication link.

CAENDigitizer is a library of functions specifically designed for the digitizer families running both standard firmware and DPP (Digital Pulse Processing) firmware. The library allows the user to connect to the digitizer, program it and manage the data acquisition: simple readout programs can be made with no need to know the details of the registers and the event data format. Indeed the CAENDigitizer library implements a common interface for higher software layers, masking the details of the physical channel and its communication protocol.

A specific version of CAENDigitizer library in C is available for Windows and Linux 32 and 64-bit platforms, and it is documented in **[RD1].**

Drivers & Libraries

Drivers

In order to deal with the hardware, CAEN provides the drivers for all the different types of physical communication interfaces featured by the specific digitizer and compliant with Windows and Linux OS:

- **USB 2.0 Drivers for NIM/Desktop** boards are downloadable on CAEN website (www.caen.it) in the "Software/Firmware" tab of the digitizer web page (**login required**).

Note: Windows OS USB driver installation for Desktop/NIM digitizers is detailed in **[RD2]**.

- **USB 2.0 Drivers for V1718** CAEN Bridge, required for the VME boards interface, is downloadable on CAEN website (www.caen.it) in the "Software/Firmware" tab of the V1718 web page (**login required**).

Note: For the installation of the V1718 USB driver, refer to the User Manual of the Bridge (**[RD3]**).

- **Optical Link Drivers** are managed by the A2818 PCI card or the A3818 PCIe card. The driver installation package is available on CAEN website in the "Software/Firmware" area at the A2818 or A3818 page (**login required**)

Note: For the installation of the Optical Link driver, refer to the User Manual of the specific Controller (**[RD4]**, **[RD5]**).

Libraries

The CAENDigitizer library is based on a set of middleware software also required by CAEN software tools for a correct functioning. These libraries, including also demo and example programs, represent a powerful base for users who want to develop customized applications for the digitizer control (communication, configuration, readout, etc.):

- **CAENVMELib** is a set of ANSI C functions which allows the user to manage and configure the CAEN Bridges and Controllers V1718/VX1718 (VME-USB2.0 Bridge), V2718/VX2718 (VME-PCI/PCIe Optical Link Bridge), and A2818/A3818 (PCI/PCIe-CONET Controller).

  The CAENVMElib installation package is available on CAEN website in the 'Download' area at the CAENVMELib Library page. Reference document: **[RD3]**.

- **CAENComm** library manages the communication at low level (read and write access). The purpose of the CAENComm is to implement a common interface to the higher software layers, masking the details of the physical channel and its protocol, thus making the libraries and applications that rely on the CAENComm independent from the physical layer. Moreover, the CAENComm is based on CAENVMELib and it requires the CAENVMELib library (access to the VME bus) even in the cases where the VME is not used. This is the reason why **CAENVMELib has to be already installed on your PC before installing the CAENComm**.

  The CAENComm installation package is available on CAEN website in the 'Download' area at the CAENComm Library page. Reference document: **[RD6]**.

Currently, the CAENComm, and so the CAENDigitizer, supports the following communication interfaces (see **Fig. 1.1**):

- PC → USB → Digitizer (either Desktop or NIM models)
- PC → USB → V1718 → VME → Digitizers (VME models only)
- PC → PCI (A2818) → CONET → Digitizers (all models)
- PC → PCI (A2818) → CONET → V2718 → VME → Digitizers (VME models only)
- PC → PCIe (A3818) → CONET → Digitizers (all models)
- PC → PCIe (A3818) → CONET → V2718 → VME → Digitizers (VME models only)

CONET (Chainable Optical NETwork) indicates the CAEN proprietary protocol for communication on Optical Link. Refer to **[RD7]** for useful information.

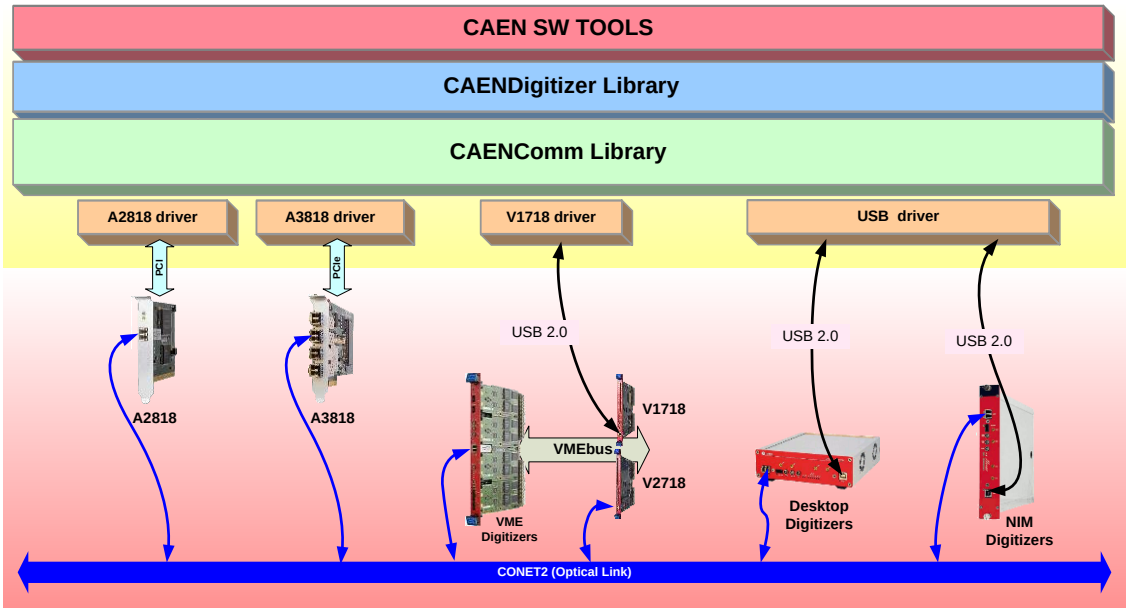


Fig. 1.1: Hardware and Software layers

Installation

CAENDigitizer LabVIEW library works on Windows OS, 32-64 bit, and requires the third-party software LabVIEW 2009 or higher.

Before installing CAENDigitizer LabVIEW library, perform the following steps:

- **Make sure** that your **hardware** (Digitizer and/or Bridge, or Controller) is **properly installed** (refer to the related User Manual for hardware installation instructions).
- **Make sure** you **have installed the driver** for your OS and the physical communication layer to be used. Driver installation packages are downloadable on CAEN website (**login required**) as reported in the **Drivers** paragraph.
- **Make sure** you **have installed the required CAEN libraries** CAENVMELib and CAENComm.

Then:

- **Download the CAENDigitizer LabVIEW installation package** compliant with your OS from CAEN website under the 'Download' area at the CAENDigitizer Library page (**login required**):

  Home / Products / Firmware/Software / Digitizer Software / Software Libraries / CAENDigitizer Library

Note: at this stage, if the required libraries still haven't been installed, it is possible to download them by clicking on the red link under the library packet link.

- **Extract files** to your host.
- **Launch the installer** and **follow** the instructions in the **installation wizard**.

Note: Installation of the CAENDitizer LabVIEW library also includes a "Demos" folder with a set of VI demo for acquisition with standard and DPP firmware, which are explained in this User Manual.

Return codes

Error code	Value	Meaning
CAEN_DGTZ_Success	0	Operation completed successfully
CAEN_DGTZ_CommError	-1	Communication error
CAEN_DGTZ_GenericError	-2	Unspecified error
CAEN_DGTZ_InvalidParam	-3	Invalid parameter
CAEN_DGTZ_InvalidLinkType	-4	Invalid Link Type
CAEN_DGTZ_InvalidHandler	-5	Invalid device handler
CAEN_DGTZ_MaxDevicesError	-6	Maximum number of devices exceeded
CAEN_DGTZ_BadBoardType	-7	Operation not allowed on this type of board
CAEN_DGTZ_BadInterruptLev	-8	The interrupt level is not allowed
CAEN_DGTZ_BadEventNumber	-9	The event number is bad
CAEN_DGTZ_ReadDeviceRegisterFail	-10	Unable to read the registry
CAEN_DGTZ_WriteDeviceRegisterFail	-11	Unable to write into the registry
CAEN_DGTZ_InvalidChannelNumber	-13	The Channel is busy
CAEN_DGTZ_ChannelBusy	-14	The channel number is invalid
CAEN_DGTZ_FPIOModeInvalid	-15	Invalid FPIO Mode
CAEN_DGTZ_WrongAcqMode	-16	Wrong acquisition mode
CAEN_DGTZ_FunctionNotAllowed	-17	This function is not allowed for this module
CAEN_DGTZ_Timeout	-18	Communication Timeout
CAEN_DGTZ_InvalidBuffer	-19	The buffer is invalid
CAEN_DGTZ_EventNotFound	-20	The event is not found
CAEN_DGTZ_InvalidEvent	-21	The event is invalid
CAEN_DGTZ_OutOfMemory	-22	Out of memory
CAEN_DGTZ_CalibrationError	-23	Unable to calibrate the board
CAEN_DGTZ_DigitizerNotFound	-24	Unable to open the digitizer
CAEN_DGTZ_DigitizerAlreadyOpen	-25	The Digitizer is already open
CAEN_DGTZ_DigitizerNotReady	-26	The Digitizer is not ready to operate
CAEN_DGTZ_InterruptNotConfigured	-27	The Digitizer has not the IRQ configured
CAEN_DGTZ_DigitizerMemoryCorrupted	-28	The digitizer flash memory is corrupted
CAEN_DGTZ_DPPFirmwareNotSupported	-29	The digitizer DPP firmware is not supported in this lib version
CAEN_DGTZ_NotYetImplemented	-99	The function is not yet implemented

Tab. 1.1: Return codes table

2 Communication

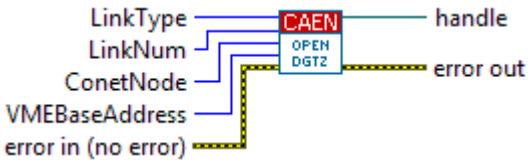
The functions described in this chapter allow the user to open and close the connection with the digitizer as well as to get the board information such as the serial number, the model, the firmware revision, etc. To open one board is necessary to set the physical communication channel from the PC to the device (as already indicated in the introduction). Once the device is opened, the function returns a **handle** that becomes the unique identifier of that device; any access operation to the device (except for VME IRQ management) will take place according to its handle, thus not invoking the physical channel.

OpenDigitizer

Desktop and NIM versions can be directly handled via USB, just connecting the digitizer to the host PC via the USB cable (the USB driver is available on Digitizer web page). In case of optical link, the communication is handled by the auxiliary board A2818/A3818, that manages the CAEN protocol communication CONET2. When using VME boards it is possible to directly connect via optical link, or to use the communication bridges V1718 (for USB) or V2718 (for optical link).

Description
Opens the digitizer and gets the device handle. See the examples in Chapter **Examples of communication settings** for the different types of communication channels and the relevant parameters.

LabVIEW VI



Arguments

Name	Type	Description
`LinkType`	I32	Indicates the physical communication channel. It can be CAEN_DGTZ_USB (either direct connection or VME through V1718), CAEN_DGTZ_OpticalLink (A2818/A3818 -> Optical Link, either direct connection or VME through V2718). **Note:** functions CAEN_DGTZ_PCI_OpticalLink, CAEN_DGTZ_PCIE_OpticalLink, and CAEN_DGTZ_PCIE_EmbeddedDigitizer are now deprecated.
`LinkNum`	I32	Link number: in case of USB, the link numbers are assigned by the PC when you connect the cable to the device; it is 0 for the first device, 1 for the second and so on. There is not a fixed correspondence between the USB port and the link number. For the CONET, the link number indicates which link of A2818 or A3818 is used; Link index start from 0 (1st Optical link port in the 1st slot used). It is not known a priori which is the first slot used (it depends on the motherboard of the PC used.). **IMPORTANT NOTE: if A2818 and A3818 are installed together, the A2818 have the lowest index assigned**.
`ConetNode`	I32	The CONET node identifies which device in the Daisy chain is being addressed. The node is 0 for the first device in the chain, 1 for the second and so on. In case of USB, *ConetNode* must be 0.
`VMEBaseAddress`	U32	VME Base Address of the board (rotary switches setting) expressed as a 32-bit number. This argument is used only for the VME models accessed through the VME bus and **MUST BE 0** in all other cases.
`handle`		Pointer to the handler returned by the open function

Note
To open multiple connections the user must use one subVI for each connection he wants to open.

CloseDigitizer

Description
This function closes the digitizer.

LabVIEW VI

Arguments

Name	Type	Description
handle		Device handler

GetInfo

Description
The function reads from the board some information such as serial number, model, number of channels, firmware release and other parameters of the device.

LabVIEW VI

Arguments

Name	Type	Description
handle		Device handler
Board Info		The structure containing the Board Info filled by the CAEN_DGTZ_GetInfo

BoardInfo Fields

Name	Type	Description
ModelName		Model name: for example “V1724”
Model		See Type Def CAEN_DGTZ_BoardModel.ctl
Channels		Number of channels
FormFactor		Format Factor (VME, NIM, Desktop); see Type Def CAEN_DGTZ_BoardFormFactor.ctl
FamilyCode		Family (ADC type); see Type Def CAEN_DGTZ_FamilyCode.ctl
ROC_FirmwareRel		Firmware Revision of the FPGA on the mother board (ROC); for example “01.02”
AMC_FirmwareRel		Firmware Revision of the FPGA on the daughter board (AMC)
SerialNumber		Serial number of the board
PCB_Revision		PCB Revision number
ADC_NBits		Number of bits of the ADC
DPPFirmware		On-board firmware information: – *NotDPPFirmware* in case of standard firmware – *DPPFirmware_CI/PHA/PSD* in case of DPP firmware – *DPPFirmwareNotSupported* in case of obsolete firmware – *DPPFirmware_unknown* in case of firmware not compliant

Reset

Description
This function resets the Digitizer. All internal registers and states are restored to default.

LabVIEW VI

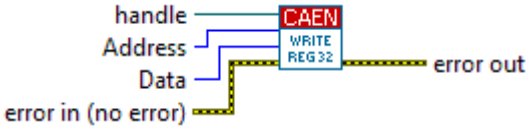
Arguments

Name	Type	Description
handle		Device handler

WriteRegister

Description
Generic write access to one register of the digitizer. The CAENDigitizer library provides specific functions for most of the parameters settings; in case there is not a specific function for accessing a particular register or the user wants to force the writing of a datum, this function makes it possible to perform a direct access to the registers. It is worth noticing that overwriting of some settings may cause inconsistencies on the operations.

LabVIEW VI



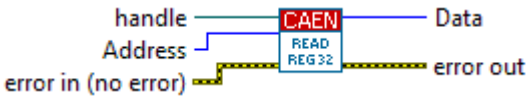
Arguments

Name	Type	Description
handle		Device handler
Address	U32	Register address. For the VME access, this is the lower 16 bit part of the VME address bus
Data	U32	32-bit data to write

ReadRegister

Description
Generic read access to one register of the digitizer (see **WriteRegister** for more details)

LabVIEW VI



Arguments

Name	Type	Description
handle		Device handler
Address	U32	Register address. For the VME access, this is the lower 16-bit part of the VME address bus
Data	U32	Data read from the board (32 bit)

Calibrate

Description
Enable the ADC calibration for x730/x731. For the x751/761 this function is automatically implemented in the **OpenDigitizer** function.

LabVIEW VI

Arguments

Name	Type	Description
handle		Device handler

Interrupt configuration

All digitizers can generate interrupt requests (IRQ) to the PC to the occurrence of a particular condition: if the memory contains at least Ne events ready for reading, where Ne is a programmable parameter.

This allows to create programs that build the process of readout (read access to the memory buffer) on interrupts: they perform passive wait cycles, until they are awakened by the driver at the arrival of an interrupt from the digitizer; at such point , the process can read data, aware to find at least Ne events in memory, without having to check in advance the presence of data, as in the case of the readout based on polling.

The readout based on the interrupts is therefore more efficient, in terms of employment of the PC resources, compared to the one based on polling.

The interrupt requests are transferred from the digitizer to the PC via the optical link, in one of the following ways:

- Direct connection to the optical link (all models): the digitizer sends the interrupt request on the optical link to the A2818 PCI or A3818 PCIe connected to the PC, and these, in their turn, assert the interrupt request on the PCI bus or PCIe respectively. In this case, the interrupt request coming to the PC is uniquely associated with the digitizer which sent it.
- Connection via VME bus: in this case, the digitizer asserts the interrupt request on the VME bus on one of the 7 IRQ lines, and this request is detected by the VME master (V2718), which sends it via optical link to the PC, in the same manner described above. In this case, since the lines IRQ [7 .. 1] of the VME are shared with all modules on VME bus, it is necessary to identify the module that sent the request, as explained farther.

Note: interrupts cannot be used in case of communication via USB (either directly or through V1718 and VME)

Set / GetInterruptConfig

Description
Enable / Disable the digitizer to generate an interrupt request when the memory contains at least Ne events ready for reading, where Ne is the parameter event_number.

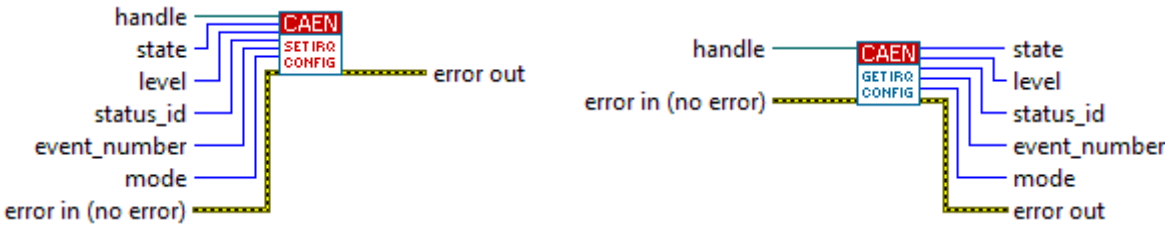
- In the case of VME models, the IRQ level to be activated on VME bus can be set from 1 to 7;
- in the case of the optical link, level should be 1.

The status_id, according to the specifications of the VME bus, is the value returned by the card during the interrupt acknowledge cycle and allows the operator to see which digitizer has asserted the interrupt request on the VME bus; in the programming stage, the user must set different status_id values for each digitizer. In the case of the optical link, the status_id is meaningless.

The mode parameter sets the interrupt release policy of the digitizer: in particular, **Roak** (Release On Acknowledge) mode foresees that the request is issued immediately after the interrupt acknowledge cycle (IACK), while in the case of **Rora** (Release on Register Access) mode, the interrupt request is not released by the digitizer until the user accesses a particular registry to disable it; in the case of the digitizer, the release occurs by setting to zero the level in the VME Control register, by calling the "Set" function of **Set / GetInterruptConfig** with status = disabled.

The methods Rora and Roak, arising from the VME specifications, are implemented also in the CONET protocol of the optical link, with the exception that the Interrupt Acknowledge cycle with CONET is required only to release the interrupt and not to identify the device that has generated it, since this information is already determined from the handle

LabVIEW VI



Arguments

Name	Type (Set)	Type (Get)	Description
handle			Device handler
state	U32	U32	Enable/Disable
level	U8	U8	VME IRQ Level (from 1 to 7). Must be 1 for direct connection through CONET
status_id	U32	U32	32-bit number assigned to the device and returned by the device during the Interrupt Acknowledge
event_number	U16	U16	If the number of events ready for the readout is equal to or greater than *event_number*, then the digitizer asserts the interrupt request
mode	U32	U32	Interrupt release mode: *CAEN_DGTZ_IRQ_MODE_RORA* (release on register access) or *CAEN_DGTZ_IRQ_MODE_ROAK* (release on acknowledge)

IRQWait

Description

Once set up the digitizer to generate an interrupt request by the function described above, the reading process can enter a state of passive waiting to be woken up as the interrupt request from the digitizer which is communicating with (the one identified uniquely from the handle passed as a parameter), is sent. This function is valid only for direct connection to link optical digitizer, in the case of communication via the VME, use **VMEIRQWait**. The timeout parameter indicates the maximum waiting time before being forced to wake up even without interrupt. In this case, the value returned by the function is 18.

LabVIEW VI

Arguments

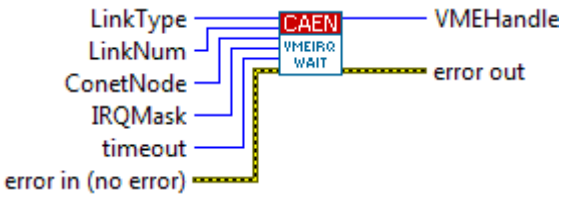
Name	Type	Description
handle		Device handler
timeout	U32	Timeout (max. wait time) in ms

VMEIRQWait

Description

This function, as the one described above, implements the passive waiting from which the waking occurs up in response to an interrupt request from the digitizer. The main difference is that in this case, the digitizer asserts a IRQ (1 to 7) on the VME bus and this is transferred to the PC by the master VME V2718. Since other digitizers could be on the VME bus (and therefore different handles that identify them within the program), and each one can generate interrupts, even on the same IRQ line, the management of interrupts cannot take place through the handle of the digitizer (which cannot be uniquely associated with the request arrived at the PC) but must be performed through the handle of the master VME V2718 which is the unique collector of interrupt requests to the PC. Once awakened from the waiting status, the process of reading can understand what digitizer has actually sent the request via the interrupt acknowledge cycle.

LabVIEW VI



Arguments

Name	Type	Description
LinkType	U32	Indicates the physical communication channel used to connect the CAEN VME bridge that handles the interrupts on the VME bus. It can be `CAENComm_USB` for the V1718 or `CAENComm_PCI_OpticalLink` for the A2818 -> Optical Link -> V2718) or `CAENComm_PCIe_OpticalLink` (same as A2818 but using A3818)
LinkNum	I32	Link number: in case of USB, the link numbers are assigned by the PC when you connect the cable to the device; it is 0 for the first device, 1 for the second and so on. There is not a fixed correspondence between the USB port and the link number. For the CONET, the link number indicates which A2818 or A3818 is used; also in this case, it is not known a priori which PCI/PCIe card is assigned to which number.
ConetNode	I32	The CONET node identifies which device in the Daisy chain is being addressed. The node is 0 for the first device in the chain, 1 for the second and so on. In case of USB, ConetNode must be 0.
timeout	U32	Timeout (max wait time) in msec
VMEHandle	I32	Device handler of the CAEN VME Bridge that received the interrupt request

VMEIRQCheck

Description

This function allows to read the status of interrupt requests on the VME bus (IRQ1-7) and, for this reason, the handle to be passed is the VME master one, not the digitizer one. This function can only be used for digitizer that communicate via the VME bus. The purpose of this function is almost exclusively for debugging.

LabVIEW VI

Arguments

Name	Type	Description
VMEHandle	I32	Device handler of the VME bridge that handles the interrupts
Mask	U8	Mask of the IRQ lines read from the VME bus (1=IRQ active, 0=IRQ not active)

IACKCycle

Description

This function performs an interrupt acknowledge cycle on the digitizer identified by the handle. This function can only be used for direct communications via optical links; in case of communication via the VME, it should be used **VMEIACKCycle** described farther. Although in the case of direct connection to the optical link there is not need to identify the digitizer that generated the interrupt request, the IACK cycle is still executed in the case of mode ROAK (release on acknowledge) to release the request

LabVIEW VI

Arguments

Name	Type	Description
handle		Device handler of the digitizer
board_id	I32	Data (status_id) returned by the digitizer that asserted the interrupt request.

VMEIACKCycle

Description

This function performs an interrupt acknowledge cycle to know the board_id of the board that raised an interrupt. As described previously, in the case of interrupt requests on the VME bus, it is not possible to know in advance which digitizer asserted a certain IRQ line. Indeed, it could also happen that a line is asserted by any other slave on the VME bus with which no communication is established. For this reason, when the reading process on hold in a specific IRQ is awakened, it must perform an interrupt acknowledge cycle to see which one generated the interrupt. The identification is as follows: during acknowledge cycle (which is very similar to a read cycle), the slave that caused the interruption puts on his bus status_id, actually the value previously programmed by the user through the "Set" function of **Set / GetInterruptConfig** function. In the case of multiple cards having different values of the programmed status_id, the user will be able to figure out who sent the request, and then which one is to be read. It should be noted that in the case of multiple cards on the bus (even inhomogeneous), the interrupt management must be centralized, as the acknowledge cycle should be performed only once. It is therefore not recommended (although possible) to have more process waiting on the same IRQ line.

LabVIEW VI

Arguments

Name	Type	Description
`VMEHandle`	I32	Device handler of the CAEN VME bridge that handles the interrupts
`level`	U8	IRQ level (from 1 to 7) on which to perform the interrupt acknowledge cycle
`board_id`	I32	Data (status_id) returned by the digitizer that asserted the interrupt request

RearmInterrupt

Description

Rearm the Interrupt.

LabVIEW VI

Arguments

Name	Type	Description
`handle`		Device handler

Data Readout

The data reading from the memories of the digitizer is done through BlockRead cycles (although it is possible also to run cycles to read each buffer). In the case of direct communication via USB or optical link, the protocol that manages the blocks transfer is CAEN proprietary and therefore there are no ambiguities or special options to be decided. Conversely, if reading takes place through the VME bus, since the standard provides different types of access and not all VME masters support all modes (or do it differently), the reading mode may need to be adapted according to the master features. The library foresees the use of master CAEN V1718 and V2718 and the readout mode is optimized for these modules.

ClearData

Description
This function Clears the data stored in the buffers of the Digitizer.

Note: this function is automatically run at the StartAcquisition. Do not use during an acquisition unless you are aware that the data has to be cleared.

LabVIEW VI

Arguments

Name	Type	Description
handle		Device handler

DisableEventAlignedReadout

Description
By default, in the data transfer from the memory of the digitizer to the PC, regardless of the type of link used, events are aligned: the digitizer stop the transfer after transferring an integer number Ne of events, where Ne is user programmable through the "Set" function of **Set / GetMaxNumEventsBLT**, even if the user has requested the transfer of more data. In the case of communication via USB and optical links, the premature termination of the transfer is foreseen by the protocol; instead, for the VME Block Transfer, the transfer is interrupted by the digitizer asserting the bus error (if enabled, see above).

LabVIEW VI

Arguments

Name	Type	Description
handle		Device handler

Set / GetMaxNumEventsBLT

Description

Concerning the Digitizers running the standard firmware, this function sets/gets the maximum number of events for each transfer. Regardless of the type of link, during a block transfer cycle, the digitizer stops the transfer after a predetermined number of events (or when the memory is empty). The greater the number of events transferred (and thus the size of the block read), the greater the efficiency of the readout, since the protocol overhead is smaller. In contrast, higher values for **MaxNumEventsBLT** imply the need to allocate a memory buffer for very large the readout.

Note: If using DPP-PHA, DPP-PSD or DPP-CI firmware, you have to refer to the **SetDPPEventAggregation** function.

LabVIEW VI

Arguments

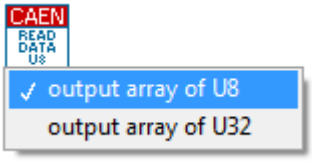
Name	Type (Set)	Type (Get)	Description
handle			Device handler
numEvents	U32	U32	Maximum number of events to transfer in a BlockRead

ReadData

Description
This function performs a block transfer of data from the digitizer to the computer. The size of the block is a function of the programmed parameters. The block can contain one or more events, that can be transferred through *buffers*. The function returns in *bufferSize* the size of the data block read from the card, expressed in bytes.

LabVIEW VI (*polymorphic*)

Use the selector menu to choose the VI according to the type of the output *buffer* parameter.



Note: Make sure to select the proper output paramenter according to the input VI. For example, usually firmware standard VI GetNUmEvents/DecodeEvent/GetEventInfo requires arrays of U8, while DPP VI GetDPPEvents/DecodeDPPWaveform requires arrays of U32).

Arguments

Name	Type	Description
handle		Device handler
mode	U32	CAEN_DGTZ_SLAVE_TERMINATED_READOUT_MBLT = 0 CAEN_DGTZ_SLAVE_TERMINATED_READOUT_2eVME = 1 CAEN_DGTZ_SLAVE_TERMINATED_READOUT_2eSST = 2 CAEN_DGTZ_POLLING_MBLT = 3 CAEN_DGTZ_POLLING_2eVME = 4 CAEN_DGTZ_POLLING_2eSST = 5
buffer	U8 / U32	Readout buffer
bufferSize	U32	Size of the data block read from the board (expressed in bytes)

Note:
CAEN_DGTZ_SLAVE_TERMINATED_READOUT_MBLT for VME accesses:
In this case the digitizer is programmed to assert the VME Bus Error during a Block Transfer cycle to prematurely end the cycle when it no longer has data to transfer or has completed the transfer of the maximum number of events planned (see *BLT_EVENT_NUM* register, or **Set / GetMaxNumEventsBLT** function). This use of the Bus Error, though not specifically provided by the VME standard for this purpose, it is actually very common. However, some VME masters have a Bus Error management not suitable for this purpose.
CAEN_DGTZ_POLLING_MBLT for VME accesses:
The VME Bus Error generation is disabled, the transfer always continues until the completion of the number of bytes required and, if there are no data to be transferred, the digitizer will insert filler words (0xFFFFFFFF)

GetNumEvents

Description

This function scans the readout buffer and gets the number of events contained in the data block previously read by the **ReadData** function. The number of events is returned in the parameter *numEvents*.

Note: If using DPP-PHA, DPP-PSD or DPP-CI firmware, you have to refer to the **GetDPPEvents** function.

LabVIEW VI

Arguments

Name	Type	Description
`handle`		Device handler
`buffer`	U8	Readout buffer
`numEvents`	U32	Number of events contained in the readout buffer
`indexEvents`	U32	Array of the event indexes in the readout buffer

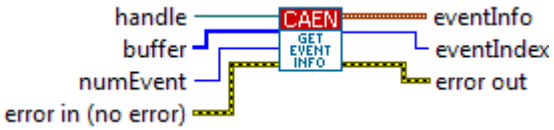
GetEventInfo

Description

This function retrieves the information (trigger time stamp, event number, channel mask, etc.) associated to one event contained in the readout buffer. This function reads the header of the *numEvent* event in the buffer, fills the eventInfo structure and returns in the *eventIndex* its index. This index will be passed to the **DecodeEvent** function described below.

Note: If using DPP-PHA, DPP-PSD or DPP-CI firmware, you have to refer to the **GetDPPEvents** function.

LabVIEW VI



Arguments

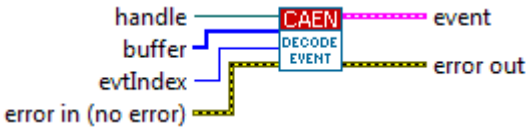
Name	Type	Description
`handle`		Device handler
`buffer`	U8	Readout buffer
`numEvent`	I32	Number of the requested event in the readout buffer (0 is the first event in the buffer)
`eventInfo`		The structure that contains the information about the requested event. The allowed fields of this structure (type U32) are: – *EventSize* – *BoardId* – *Pattern* – *ChannelMask* – *TriggerTimeTag* See the digitizer User Manual for a detailed description.
`eventIndex`	I32	The index of the requested event data in the readout buffer

DecodeEvent

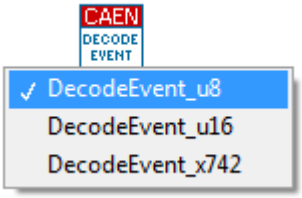
Description
Each type of digitizer has a different event data format. This function decodes (unpacks) the data of a specified event and fills the event structure containing the data of each channel (i.e. the waveform and/or other parameters in case of DPP) separately.

Note: If using DPP-PHA, DPP-PSD or DPP-CI firmware, you have to refer to the **GetDPPEvents** function.

LabVIEW VI (*polymorphic*)



Use the selector menu to choose the proper VI according to the *event* parameter.



Note: the menu options are driven by the specific digitizer model (DecodeEvent_u8 used by V1721 and V1731, DecodeEvent_x742 dedicated to 742 series).

Arguments

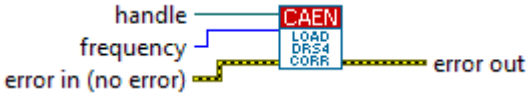
Name	Type	Description
`handle`		Device handler
`buffer`	U8	Readout buffer
`evtIndex`	I32	Index to the event data in the readout buffer (this is the index returned by the **GetEventInfo** function).
`event`		The decoded event structure with the following fields: **Event_u8** U32 ChSize U8 DataChannel **Event_u16** U32 ChSize U16 DataChannel **Event_x742** U8 GrPresent DataGroup: U32 ChSize SGL DataChannel U32 TriggerTimeTag U16 StartIndexCell

LoadDRS4CorrectionData

Description
Regarding the x742 series, in order to compensate for unavoidable construction differences in the DRS4 chips, a data correction is required (for details, please refer to the User Manual of the board). This function loads the correction parameters stored on board, while a **DecodeEvent** function is then needed to apply them. The correction parameters to load depend on the operating sampling frequency.

Note: to be used only with x742 series.

LabVIEW VI



Arguments

Name	Type	Description
`handle`		Device handler
`frequency`	U32	The DRS4 sampling frequency.

Enable/Disable DRS4Correction

Description
Enables/disables the data correction in the x742 series.

Note: to be used only with x742 series.

Note: If enabled, the data correction through the **DecodeEvent** function only applies if a **LoadDRS4CorrectionData** has been previously called, otherwise the **DecodeEvent** runs the same, but data will be provided out not compensated.

LabVIEW VI

Arguments

Name	Type (Enable)	Type (Disable)	Description
`handle`			Device handler

3 Trigger configuration

The acquisition in the digitizer is ruled by the trigger, which is a signal that decides when to start the acquisition window and save samples of the ADC or the values of interest calculated on line (DPP) in the digitizer memory.

The digitizer can have the following trigger sources: External Trigger (digital signal from the panel), Software Trigger (write access to the specific register), Self-Trigger Channel (internal signal generated by a digitizer channel under certain conditions, for example when the input signal exceeds a programmable threshold).

All trigger sources can be enabled or not to generate the acquisition trigger for the channels. Similarly, it is possible to decide what triggers should participate in the generation of the Trigger Output (NIM or TTL digital output of the digitizer panel). Trigger Output can not necessarily coincide with the acquisition trigger: for example, in order to trigger multiple cards at once, as one of their channel has “auto triggered“; for this purpose, the auto triggering channel is used only to generate the Trigger Outputs (but not for the acquisition trigger); all Trigger Outputs are ORed externally to the cards and the resulting signal is sent in parallel to all cards Trigger Inputs, which are programmed to enable only the Trigger Input to generate the acquisition Trigger.

Note: in digitizer series x740, the auto trigger channel is divided into two levels: each 8-channel group generates a “group local trigger”, given by the OR a of channel triggers enabled to generate them. The group triggers, in their turn, may participate or not to generate the acquisition trigger and / or trigger output.

SendSWtrigger

Description
This function sends a Software trigger to the Digitizer. The SW trigger can be used to save an acquisition window on all channels at the same time and/or to generate a pulse on the Trigger Output of the board, according to the SW trigger mode set by the “Set” function of the **Set / GetSWTriggerMode**.

LabVIEW VI

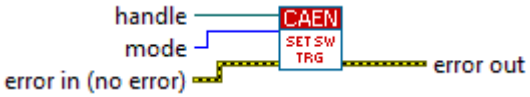
Arguments

Name	Type	Description
handle		Device handler

Set / GetSWTriggerMode

Description
This function decides whether the trigger software should only be used to generate the acquisition trigger, only to generate the trigger output, or both.

LabVIEW VI



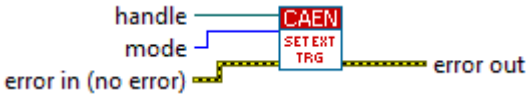
Arguments

Name	Type (Set)	Type (Get)	Description
handle			Device handler
mode	U32	U32	SW Trigger mode: CAEN_DGTZ_TRGMODE_DISABLED = 0, CAEN_DGTZ_TRGMODE_EXTOUT_ONLY = 2, CAEN_DGTZ_TRGMODE_ACQ_ONLY = 1, CAEN_DGTZ_TRGMODE_ACQ_AND_EXTOUT = 3,

Set / GetExtTriggerInputMode

Description
This function decides whether the external trigger should only be used to generate the acquisition trigger, only to generate the trigger output, or both.

LabVIEW VI



Arguments

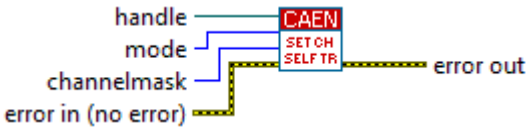
Name	Type (Set)	Type (Get)	Description
handle			Device handler
mode	U32	U32	External Trigger mode CAEN_DGTZ_TRGMODE_DISABLED = 0, CAEN_DGTZ_TRGMODE_EXTOUT_ONLY = 2, CAEN_DGTZ_TRGMODE_ACQ_ONLY = 1, CAEN_DGTZ_TRGMODE_ACQ_AND_EXTOUT = 3,

Set / GetChannelSelfTrigger

Description
This function decides whether the trigger of a channel should be used only to generate the acquisition trigger, only to generate the trigger output, or both.

For the x740 series, use the **Set / GetGroupSelfTrigger** function.

LabVIEW VI



Arguments

Name	Type (Set)	Type (Get)	Description
handle			Device handler
mode	U32	U32	Channel Self Trigger mode CAEN_DGTZ_TRGMODE_DISABLED = 0, CAEN_DGTZ_TRGMODE_EXTOUT_ONLY = 2, CAEN_DGTZ_TRGMODE_ACQ_ONLY = 1, CAEN_DGTZ_TRGMODE_ACQ_AND_EXTOUT = 3,
channelmask	U32	-	(only for Set): the function applies only to those channels that have the relevant bit in the mask equal to 1
channel	-	U32	(only for Get): channel for which the mode is get

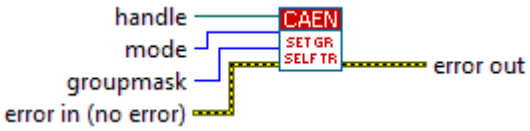
Set / GetGroupSelfTrigger

Description

This function is valid only for the x740 series. In fact, in this type of digitizer, the channels are grouped 8 by 8. The trigger properties are referred to the groups and cannot be set individually channel by channel. Each group of 8 channels generates one single self-trigger which is the OR of the 8 self-triggers in the group (with a programmable trigger enable mask, see next function). The group self-trigger can generate the acquisition trigger for the board and/or a pulse on the Trigger Output.

Note: to be used only with x740 series.

LabVIEW VI



Arguments

Name	Type (Set)	Type (Get)	Description
handle			Device handler
mode	U32	U32	Group Self Trigger mode: CAEN_DGTZ_TRGMODE_DISABLED = 0, CAEN_DGTZ_TRGMODE_EXTOUT_ONLY = 2, CAEN_DGTZ_TRGMODE_ACQ_ONLY = 1, CAEN_DGTZ_TRGMODE_ACQ_AND_EXTOUT = 3,
groupmask	U32	-	(only for Set): the function applies only to those groups that have the relevant bit in the mask equal to 1
group	-	U32	(only for Get): group for which the mode is get

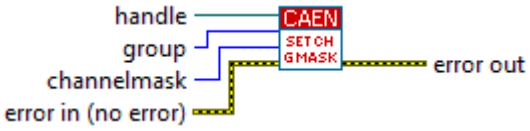
Set / GetChannelGroupMask

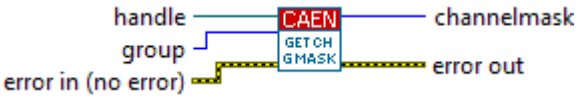
Description

This function decides which channels in a group of 8 participate to the generation of the self-trigger of that group. The self-trigger is the OR of the channels enabled by this function that are above the threshold. **WARNING:** the channels that are not connected must be disabled here, otherwise it may happen that one channel has a DC offset higher than the threshold and it keeps the OR always active.

Note: to be used only with x740 series.

LabVIEW VI





Arguments

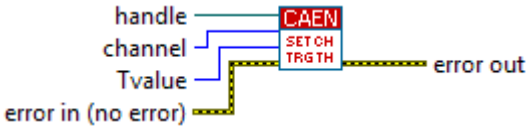
Name	Type (Set)	Type (Get)	Description
handle			Device handler
group	U32	U32	Group for which the mask is set
channelmask	U32	U32	Channels Trigger mask for the group (8 bits)

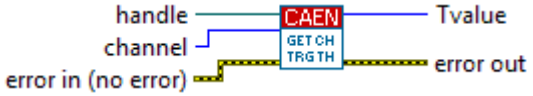
Set / GetChannelTriggerThreshold

Description
This function sets the Trigger Threshold for a specific channel. The threshold is applied to the digital signal after the ADC and it is expressed in ADC counts. The user should take care of the DC offset adjust when converting the digital threshold in the corresponding voltage level on the analog input.

For the x740 series, use the **Set / GetGroupTriggerThreshold** function. For the DPP firmware, use the **SetDPPParameters** function.

LabVIEW VI





Arguments

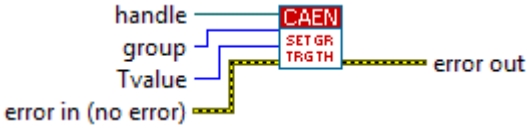
Name	Type (Set)	Type (Get)	Description
handle			Device handler
channel	U32	U32	Channel to set
Tvalue	U32	U32	Threshold value (in ADC counts)

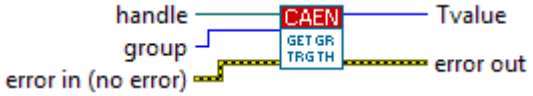
Set / GetGroupTriggerThreshold

Description
This function sets/gets the Trigger Threshold for a specified group of channel. The threshold is common to the 8 channels in the group. See the **Set / GetChannelTriggerThreshold** function for further details.

Note: to be used only with x740 series.

LabVIEW VI





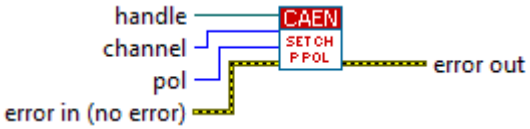
Arguments

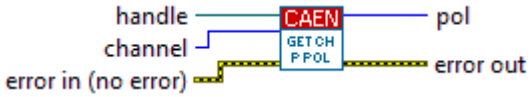
Name	Type (Set)	Type (Get)	Description
handle			Device handler
group	U32	U32	Group to set
Tvalue	U32	U32	Threshold value

Set / GetChannelPulsePolarity

Description
Sets/gets the value of the pulse polarity for the specified channel.

LabVIEW VI





Arguments

Name	Type (Set)	Type (Get)	Description
handle			Device handler
channel	U32	U32	The channel to set/get information for
pol	U32	U32	Value of the pulse polarity

Set / GetRunSynchronizationMode

Description
Sets/gets the run synchronization mode of the digitizer, used to synchronize an acquisition on multiple boards.

LabVIEW VI

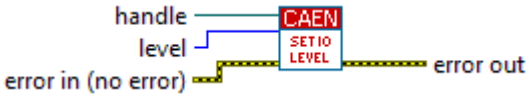
Arguments

Name	Type (Set)	Type (Get)	Description
handle			Device handler
mode	U32	U32	The run synchronization mode to set/get

Set / GetIOLevel

Description
Sets/gets the I/O level.

LabVIEW VI



Arguments

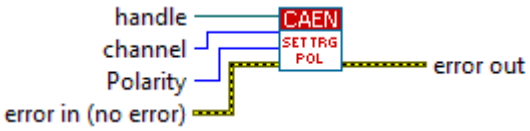
Name	Type (Set)	Type (Get)	Description
handle			Device handler
level	U32	U32	The I/O level of the digitizer to set/get

Set / GetTriggerPolarity

Description
Sets/gets the trigger polarity of a specified channel.

Note: not to be used with DPP firmware.

LabVIEW VI



Arguments

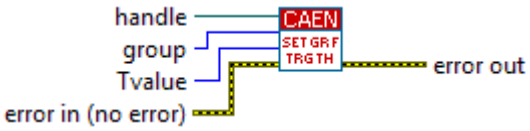
Name	Type (Set)	Type (Get)	Description
handle			Device handler
channel	U32	U32	Selects the channel to set/get the trigger polarity for
Polarity	U32	U32	The polarity of the trigger to set/get

Set / GetGroupFastTriggerThreshold

Description
Sets/gets the threshold value on TRn input (used as external trigger) for the local trigger generation in x742 series. As the threshold is an hardware threshold (input of a programmable 16-bit DAC, whose voltage output goes to a comparator), it is not easy to set and the user can refer to the board User Manual for setting examples.

Note: to be used only with x742 series.

LabVIEW VI



Arguments

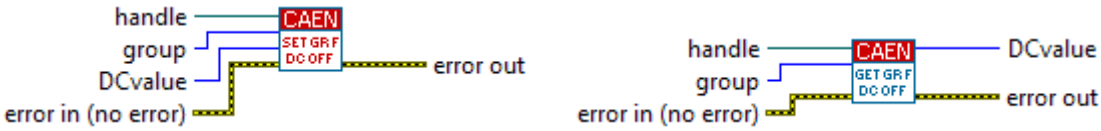
Name	Type (Set)	Type (Get)	Description
handle			Device handler
group	U32	U32	The channels group the threshold is applied to
Tvalue	U32	U32	The value of the TRn threshold to set/get

Set / GetGroupFastTriggerDCOffset

Description
Regarding the x742 series, sets/gets the TRn signal DC offset when it is sampled in the DRS4 chips in order to make positive, negative or bipolar input signals to be compliant with the DRS4 input dynamics. The DC offset also affects the TRn when used as trigger, in this case it relates to the threshold setting above described (please refer to the board User Manual for setting examples).

Note: to be used only with x742 series.

LabVIEW VI



Arguments

Name	Type (Set)	Type (Get)	Description
handle			Device handler
group	U32	U32	The channels group the DC offset is applied to
DCvalue	U32	U32	The value of the TRn DC offset to set/get

Set / GetFastTriggerDigitizing

Description
Regarding the x742 series, enables/disables (set) the presence of the TRn signal in the data readout as well as allows for checking the status of the setting (get).

Note: to be used only with x742 series.

LabVIEW VI

Arguments

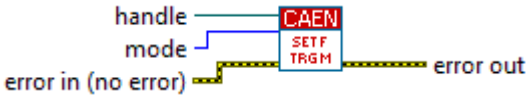
Name	Type (Set)	Type (Get)	Description
handle			Device handler
enable	U32	U32	The enable flag to set/get

Set / GetFastTriggerMode

Description
Enables/disables (set) the TRn input as local trigger in x742 series, as wells allows for checking the status of the setting (get).

Note: to be used only with x742 series.

LabVIEW VI



Arguments

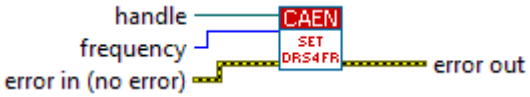
Name	Type (Set)	Type (Get)	Description
handle			Device handler
mode	U32	U32	The fast trigger value to set/get.

Set / GetDRS4SamplingFrequency

Description
Regarding the x742 series, sets/gets the sampling frequency of the DRS4 chips which sample the input analog signal and the fast trigger signal.

Note: to be used only with x742 series.

LabVIEW VI



Arguments

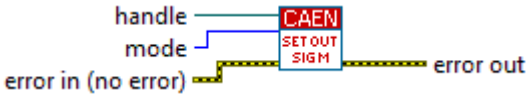
Name	Type (Set)	Type (Get)	Description
handle			Device handler
frequency	U32	U32	The sampling frequency value to set/get: *CAEN_DGTZ_DRS4Frequency_t* See the LoadCorrectionData function in **[RD1]**.

Set / GetOutputSignalMode

Description
Sets/gets the signal to be provided out over the TRG-OUT output channel in the x742 series.

Note: to be used only with x742 series.

LabVIEW VI



Arguments

Name	Type (Set)	Type (Get)	Description
handle			Device handler
mode	U32	U32	The output signal mode to set/get.

4 Acquisition

Set / GetChannelEnableMask

Description
This function enables/disables the channels for the acquisition. Disabled channels don't give any trigger and don't participate to the event data.

For the x740 and x742 series, use the **Set / GetGroupEnableMask** function

LabVIEW VI

Arguments

Name	Type (Set)	Type (Get)	Description
handle			Device handler
mask	U32	U32	Enable Mask. Bit n corresponds to channel n. Please, refer to the User Manual of each specific board for the allowed number of channels.

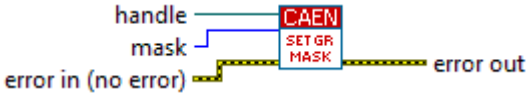
Examples
If you want to enable channel 0 and channel 1, the binary bit mask is 11 which corresponds to 0x3. If you want to enable only channel 1, the binary bit mask is 10, which corresponds to 0x2.

Set / GetGroupEnableMask

Description
This function enables/disables the groups for the acquisition. This function is valid only for the x740 and x742 series. Disabled groups don't give any trigger and don't participate to the event data. The 8 channel in a group are all enabled/disabled according to the relevant bit in the enable mask.

Note: to be used only with x740 and x742 series.

LabVIEW VI



Arguments

Name	Type (Set)	Type (Get)	Description
handle			Device handler
mask	U32	U32	Enable Mask. Bit n (with 0 <= n <= 7) corresponds to group n.

Examples
If you want to enable group 0 and group 1, the binary bit mask is 11 which corresponds to 0x3. If you want to enable only group 1, the binary bit mask is 10, which corresponds to 0x2.

SWStartAcquisition

Description

This function starts the acquisition in a board using a software command. When the acquisition starts, the relevant RUN LED on the front panel lights up. It is worth noticing that in case of multiple board systems, the software start doesn't allow the digitizer to start synchronously. For this purpose, it is necessary to use to start the acquisition using a physical signal, such as the S-IN or GPI as well as the TRG-IN-TRG-OUT Daisy chain. Please refer to Digitizer manual for more details on this issue.

LabVIEW VI

Arguments

Name	Type	Description
handle		Device handler

SWStopAcquisition

Description

This function stops the acquisition in a board using a software command.

LabVIEW VI

Arguments

Name	Type	Description
handle		Device handler

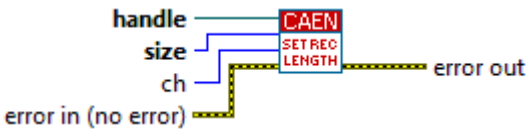
Set / GetRecordLength

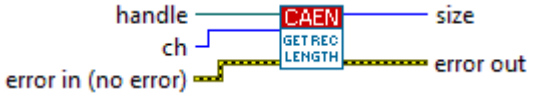
Description

This function sets the size of the acquisition window, that is the number of samples that belong to it. Due to the way the samples are written into the memory (more samples are put in parallel), there is a specific granularity of the record length depending on the board model. For example, in the x720 series, the samples are written 4 by 4, hence the record length must be a multiple of 4. Please, refer to the User Manual of the specific board for the granularity value. The function accepts any value for the parameter size and then takes the closest value multiple of the granularity. The function **GetRecordLength** returns the exact value.

Note: Each time the record length is changed, the post-trigger must be updated (through the *SetPostTriggerSize*).

LabVIEW VI





Arguments

Name	Type (Set)	Type (Get)	Description
handle			Device handler
size	U32	U32	The size of the record (in samples) to set/get
ch	I32	I32	INT value corresponding to the channel index. **Used only for digitizers running DPP firmware, in particular DPP-PSD and DPP-CI**.

Set / GetPostTriggerSize

Description
This function sets the post trigger size, that is the position of the trigger within the acquisition window. The size is expressed in percentage of the record length. 0% means that the trigger is at the end of the window, while 100% means that it is at the beginning.

Note: The post-trigger must be updated each time the record length is changed (through the *SetRecordLength*).

LabVIEW VI

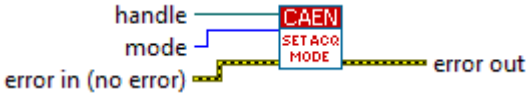
Arguments

Name	Type (Set)	Type (Get)	Description
handle			Device handler
percent	U32	U32	Post trigger as percentage of the record length

Set / GetAcquisitionMode

Description
Gets/Sets digitizer acquisition mode.

LabVIEW VI



Arguments

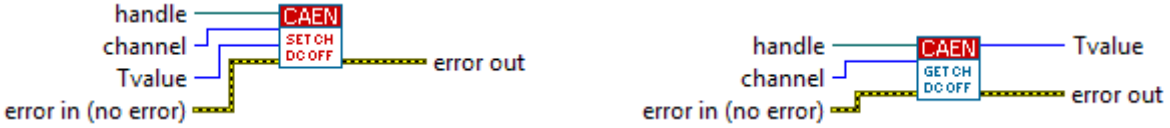
Name	Type (Set)	Type (Get)	Description
handle			Device handler
mode	U32	U32	The acquisition mode: *CAEN_DGTZ_AcqMode_t*. CAEN_DGTZ_SW_CONTROLLED = 0 CAEN_DGTZ_S_IN_CONTROLLED = 1 See the Set/GetAcquisitionMode function in **[RD1]**.

Set / GetChannelDCOffset

Description

This function sets the 16 bit DAC that adds a DC offset to the input signal in order to adapt it to the dynamic range of the ADC. By default, the DAC is set to middle scale (0x8000) which corresponds to a DC offset of –Vpp/2, where Vpp is the voltage range (peak to peak) of the ADC. This means that the input signal can range from –Vpp/2 to +Vpp/2. If the DAC is set to 0x0000, then no DC offset is added and the range of the input signal goes from 0 to +Vpp. Conversely, when the DAC is set to 0xFFFF, the DC offset is –Vpp and the range goes from –Vpp to 0. The DC offset can be set on channel basis except for the x740 in which it is set on group basis; in this case, you must use the **Set / GetGroupDCOffset** functions.

LabVIEW VI



Arguments

Name	Type (Set)	Type (Get)	Description
handle			Device handler
channel	U32	U32	Value corresponding to the channel index. Use -1 for all channels
Tvalue	U32	U32	DAC value (from 0x0000 to 0xFFFF)

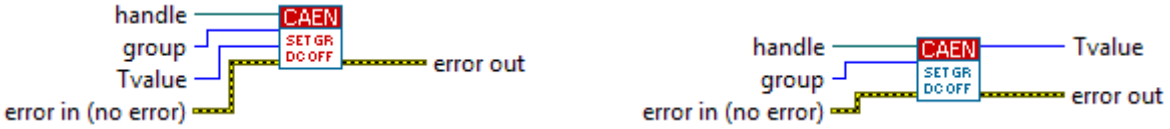
Set / GetGroupDCOffset

Description

The same as Set/Get ChannelDCoffset, but in this case it is applied to the groups of the x740 series.

Note: to be used only with x740 series.

LabVIEW VI



Arguments

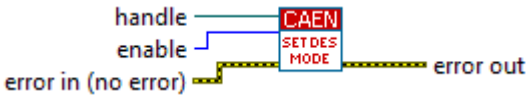
Name	Type (Set)	Type (Get)	Description
handle			Device handler
group	U32	U32	Value corresponding to the group index. Use -1 for all groups.
Tvalue	U32	U32	DAC value (from 0x0000 to 0xFFFF)

Set / GetDESMode

Description
This function enables or disables the Dual Edge Sampling mode, that is the channel interleaving option to double the sampling frequency. This option is available in the x731 and x751 series only.

WARNING: when the DES mode is enabled, only the odd channels (for the x751) or the even channels (for the x731) will work; the other channels must be left unconnected.

LabVIEW VI



Arguments

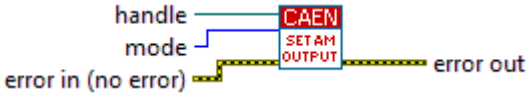
Name	Type (Set)	Type (Get)	Description
handle			Device handler
mode	U32	U32	Enable/disable mode: *CAEN_DGTZ_EnaDis_t*. CAEN_DGTZ_ENABLE = 1 to enable the DES mode, CAEN_DGTZ_DISABLE = 0 to disable the DES mode. See the Set/GetDesMode function in **[RD1]**.

Set / GetAnalogMonOutput

Description
Sets/Gets the signal to output on the Analog Monitor Front Panel output in VME digitizers running standard firmware.

Note: the function is not supported by V1742 and digitizers running DPP firmware.

LabVIEW VI



Arguments

Name	Type (Set)	Type (Get)	Description
handle			Device handler
mode	U32	U32	Analog Monitor Mode: *CAEN_DGTZ_AnalogMonitorOutputMode_t*. CAEN_DGTZ_AM_TRIGGER_MAJORITY = 0 (Trigger Majority Mode), CAEN_DGTZ_AM_TEST = 1 (Test Mode), CAEN_DGTZ_AM_ANALOG_INSPECTION = 2 (Analog Inspection Mode), CAEN_DGTZ_AM_BUFFER_OCCUPANCY = 3 (Buffer Occupancy Mode), CAEN_DGTZ_AM_VOLTAGE_LEVEL = 4 (Voltage Level Mode). See the Set/GetAnalogMonOutput function in **[RD1]**.

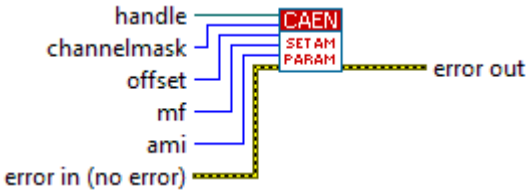
Supported digitizers and permitted AM modes

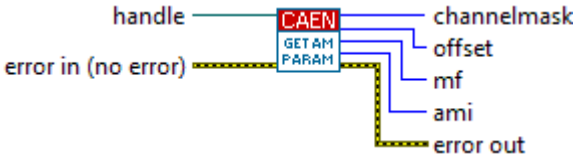
Digitizer	0	1	2	3	4
V1720-V1721-V1731-V1740-V1751	X	X		X	X
V1724	X	X	X	X	X

Set / GetAnalogInspectionMonParams

Description
Sets/Gets the Analog Inspection Monitor parameters for a V1724 digitizer running standard firmware.

LabVIEW VI





Arguments

Name	Type (Set)	Type (Get)	Description
handle			Device handler
channelmask	U32	U32	channel enable mask
offset	U32	U32	DC Offset for the analog output signal
mf	U32	U32	Multiply factor
ami	U32	U32	Invert Output

Set / GetEventPackaging

Description
This function allows to enable or disable the Pack 2.5 mode of V1720/DT5720 Digitizers

LabVIEW VI

Arguments

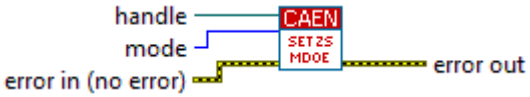
Name	Type (Set)	Type (Get)	Description
handle			Device handler
mode	U32	U32	Enable/Disable the Pack 2.5 mode

Set / GetZeroSuppressionMode

Description

This function sets/gets the Zero Suppression mode for x720, x721, x731, x724 digitizer models running standard firmware.

LabVIEW VI



Arguments

Name	Type (Set)	Type (Get)	Description
`handle`			Device handler
`mode`	U32	U32	Zero Suppression Mode: *CAEN_DGTZ_ZS_Mode_t*. CAEN_DGTZ_ZS_NO = 0 (no Zero suppression), CAEN_DGTZ_ZS_INT = 1 (Full Suppression based on the integral of the signal), CAEN_DGTZ_ZS_ZLE = 2 (Zero Length Encoding), CAEN_DGTZ_ZS_AMP = 3 (Full Suppression based on the signal amplitude). See the Set/GetZeroSuppressionMode function in **[RD1]**

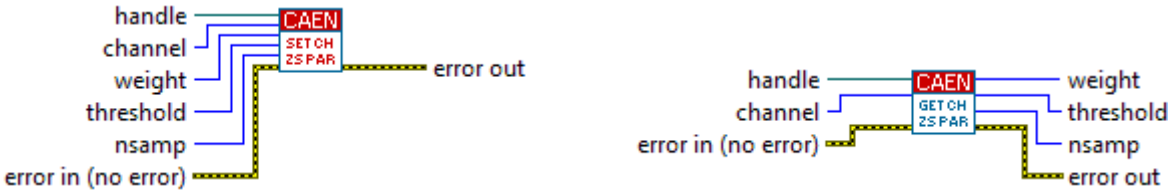
Supported digitizers and permitted zero suppression modes

Digitizer	0	1	2	3
`x720`	X		X	X
`V1721/V1731`	X		X	X
`x724`	X	X	X	X

Set / GetChannelZSParams

Description
Sets/Gets Zero Suppression parameters for a specific channel in the supported digitizers (see the table in the **Set / GetZeroSuppressionMode**functions).

LabVIEW VI



Arguments

Name	Type (Set)	Type (Get)	Description
`handle`			Device handler
`channel`	U32	U32	Channel to which the ZS settings are applied. Use -1 for all channels
`weight`	U32	U32	Zero Suppression weight*(*)*: *CAEN_DGTZ_ThresholdWeight_t*. **Used in "Full Suppression based on the integral of the signal" supported only by x724 series.** CAEN_DGTZ_ZS_FINE = 0 (Fine threshold step; the threshold is the *threshold* parameter), CAEN_DGTZ_ZS_COARSE = 1 (Coarse threshold step; the threshold is *threshold* × 64). See the Set/GetChannelZSParams function in **[RD1]**. For "Full Suppression based on the signal amplitude" and "Zero Length Encoding" algorithms, the value of *weight* doesn't affect the function working.
`threshold`	I32	I32	Zero Suppression Threshold to be used depending on the ZS algorithm*(*)*.
`nsamp`	I32	I32	Number of samples to be used by the ZS algorithm*(*)*.

*(*)*Refer to the digitizer User Manual for definition and representation.

Acquisition example

This section is intended to provide a simple example on how to use the CAENDigitizer LabVIEW library. The example works with any digitizers running standard firmware, apart the 740 and 742 series. More specific examples for both standard firmware and DPP firmware can be found on Chapter **DEMOs**.

The example codes are located in the *Examples* folder of the main library folder. By default, this corresponds to:

C:\Program Files\CAEN\Digitizers\LabView

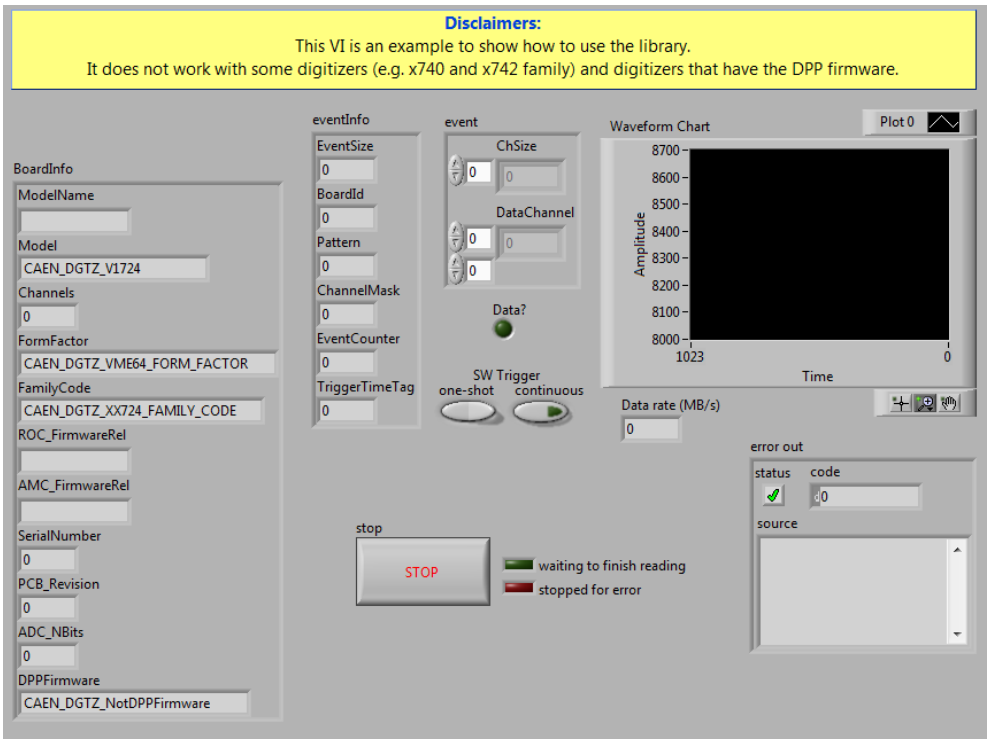


Fig. 4.1: Acquisition example control panel

This acquisition example allows the user to perform the following functions:

- open the device;
- set some of the acquisition parameters;
- start and stop the acquisition;
- plot the waveform;
- issue a software trigger.

It is not possible to save data. The example can be modified by the user for his specific needs.

Once connected to the board, the program displays the information into the *BoardInfo* section (left side) and starts automatically the acquisition. If the board is in self-trigger mode, the *eventinfo* and *event* sections update, and the *Waveform Chart* plots the input pulse. The *SW Trigger* (single or continuous) allows the user to force the board to trigger.

The code is structured into three main parts designed as follows:

Part #1: Open the device, Reset and GetInfo functions.

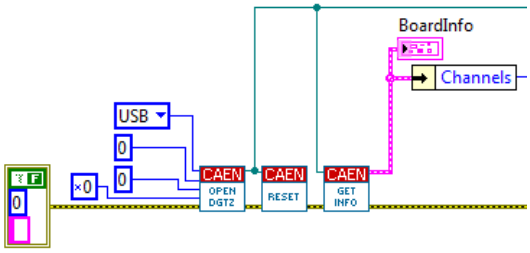


Fig. 4.2: Block Diagram Part #1

Part #2: Acquisition parameters: some of the settings are individual by channel, other are common to all channels. The functions involved are: SetMaxnumBLT, SetRecLength, SetChannelMask, SetChannelSeflTrigger, SetSWTrigger; loop over the N channels of the functions SetChannelDCOffset, SetChannelTriggerThreshold, SetChannelPulsePolarity; finaly there is the SetAcquisitionMode function.

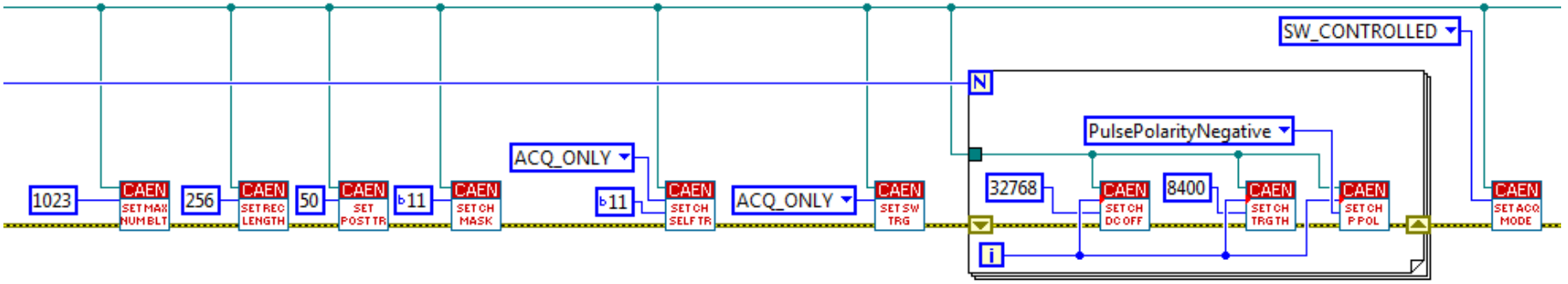


Fig. 4.3: Block Diagram Part #2

Part #3: Acquisition loop.

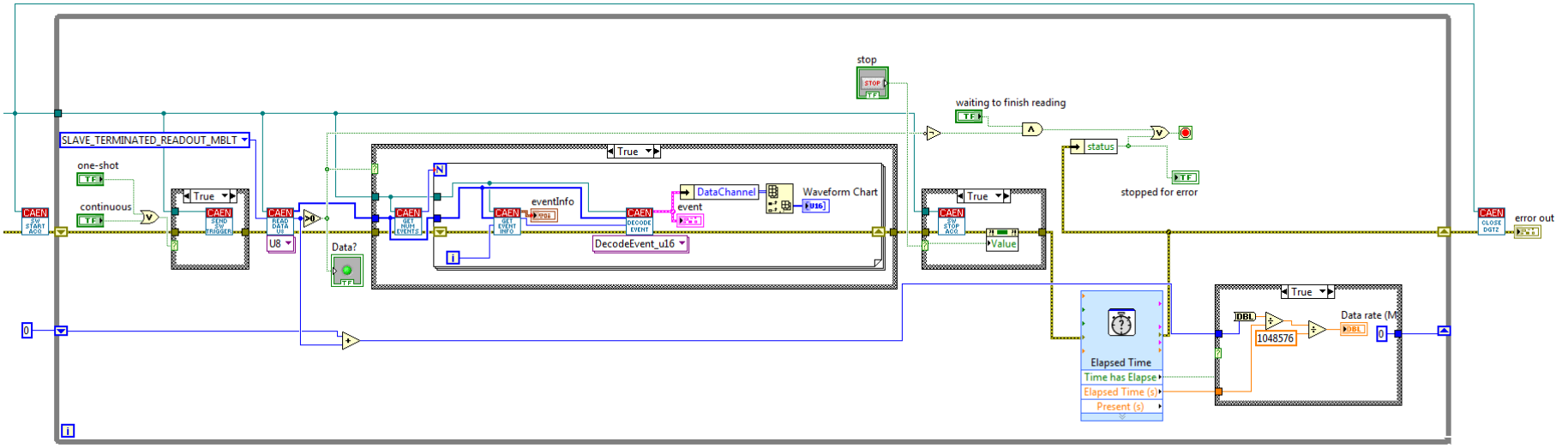


Fig. 4.4: Block Diagram Part #3

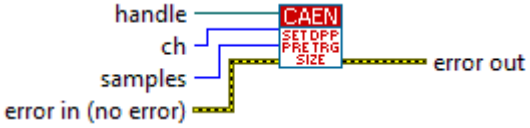
5 DPP specific VIs

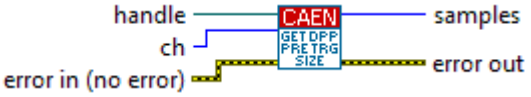
In order to handle acquisitions with the DPP firmware (PHA, PSD, CI), the LabVIEW VIs described in this chapter can be used.

Set / GetDPPPreTriggerSize

Description
Sets/gets the pre-trigger size, which is the portion of acquisition window visible before a trigger.

LabVIEW VI





Arguments

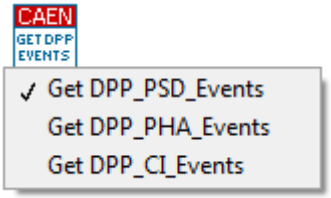
Name	Type (Set)	Type (Get)	Description
handle			Device handler
samples	U32	U32	The size of the record (in samples)
ch	I32	I32	The channel whose pre-trigger has to be set/get

GetDPPEvents

Description
Decodes and returns all the DPP events stored in the acquisition buffers.

LabVIEW VI (*polymorphic*)

Use the selector menu to choose the proper VI according to the *events* parameter.



Note: the menu options are driven by the specific DPP firmware (DPP-PSD for 720, 751 and 790 series, DPP-PHA for 724 and 780 series, DPP-CI for 720 series). The DPP-PSD and DPP-PHA for 730 are not yet managed by the CAENDigitizer LabVIEW library.

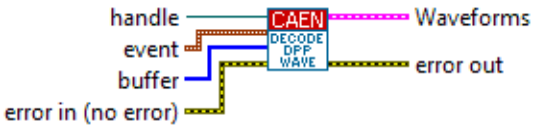
Arguments

Name	Type	Description
`handle`		Device handler
`buffer`	U32	Readout buffer
`events`		The decoded event structure with the following fields: **DPP-PSD_Events** U32 Format U32 TimeTag U32 ChargeShort U32 ChargeLong U32 Baseline U32 Pur U32 Waveforms_index U32 Extras **DPP-PHA_Events** U32 Format U64 TimeTag U16 Energy I16 Extras U32 Waveforms_index **DPP-CI_Events** U32 Format U32 TimeTag I16 Charge I16 Baseline U32 Waveforms_index
`numEventsArray`	[I32]	Array of *int* which will contain the number of events found per channel

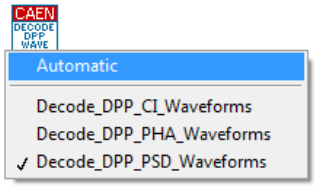
DecodeDPPWaveforms

Description
Decodes the waveforms of an event.

LabVIEW VI (*polymorphic*)



Use the selector menu to choose the proper VI according to the *event* parameter, or set to "Automatic".



Arguments

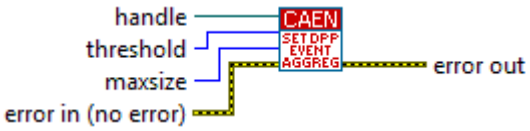
Name	Type	Description
`handle`		Device handler
`event`		The decoded event structure provided out by the **GetDPPEvents** function.
`buffer`		Readout buffer
`waveforms`		Structure of the waveform list with the following fields: **DPP_PSD_Waveforms** I32 Ns U8 DualTrace U8 anlgProbe U8 dgtProbe1 U8 dgtProbe2 U16 Trace1 U16 Trace2 U8 DTrace1 U8 DTrace2 U8 DTrace3 U8 DTrace4 **DPP_PHA_Waveforms** U32 Ns U8 DualTrace U8 VProbe1 U8 VProbe2 U8 VDProbe I16 Trace1 I16 Trace2 U8 DTrace1 U8 DTrace2 **DPP_CI_Waveforms** U32 Ns U8 DualTrace U8 anlgProbe U8 dgtProbe1 U8 dgtProbe2 U16 Trace1 U16 Trace2 U8 DTrace1 U8 DTrace2 U8 DTrace3 U8 DTrace4

SetDPPEventAggregation

Description
Sets event aggregation parameters.

Note: This function is to be used only after the record length parameter has been set (by the "Set" function of the **Set / GetRecordLength**).

LabVIEW VI



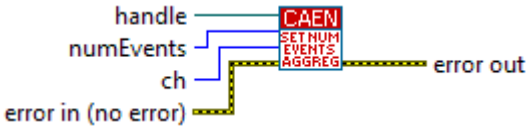
Arguments

Name	Type	Description
`handle`		Device handler
`threshold`	I32	Specifies how many events can be accumulated into the board memory before they are available for readout. A low number maximizes responsiveness, since data are read as soon as it is stored in memory, while a high number maximizes efficiency, since fewer transfers are made. **It is recommended to set the value 0** to let the library choose the best value depending on acquisition mode and other parameters.
`maxsize`	I32	Specifies the maximum size (in bytes) of the event buffer on the PC side. This parameter might be useful in case the computer has very low RAM. Normally **it is recommended to set the value 0** to let the library choose the appropriate value automatically.

Set / GetNumEventsPerAggregate

Description
Sets/Gets the number of events that each aggregate will contain.

LabVIEW VI



Arguments

Name	Type (Set)	Type (Get)	Description
`handle`			Device handler
`numEvents`	U32	U32	Number of events oper aggregater.
`ch`	I32	I32	INT value corresponding to the channel index (required for DPP-PSD and DPP-CI, ignored by DPP-PHA).

Set / GetMaxNumAggregatesBLT

Description
Sets/Gets the maximum number of aggregates per each transfer.

Note: with DPP-PHA, DPP-PSD and DPP-CI, also the *maxsize* parameter of **SetDPPEventAggregation** can be used.

LabVIEW VI

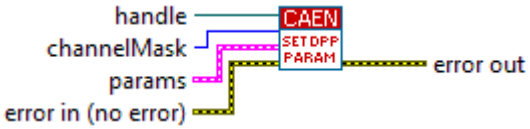
Arguments

Name	Type (Set)	Type (Get)	Description
`handle`			Device handler
`numAggr`	U32	U32	Maximun number of aggregates per transfer

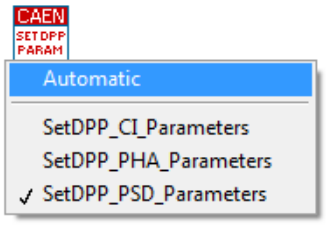
SetDPPParameters

Description
Sets DPP configuration parameters for DPP-PHA, DPP-PSD or DPP-CI.

LabVIEW VI (*polymorphic*)



Use the selector menu to choose the proper VI according to the *params* parameter, or set to "Automatic".



Arguments

Name	Type	Description
`handle`		Device handler
`channelMask`	U32	A bit mask indicating to which channels the DPP parameters are applied
`params`		Structure of the DPP parameters: *CAEN_DGTZ_DPP_PHA_Params_t* M - Decay Time (ns) m - Trapezoid Flat Top (ns) k - Trapezoid Rise Time (ns) ftd - Flat Top Delay (ns) a - Trigger Filter smoothing factor (allowed values are 1, 2, 4, 8, 16 and 32) b - Input Signal Rise time (ns) thr - Trigger Threshold (LSB) nsbl - Number of Samples for Baseline Mean (allowed values are from 0 to 6, where 0 = 0, 1 = 16, 2 = 64, 3 = 256, 4 = 1024, 5 = 4096, 6 = 16384) nspk - Number of Samples for Peak Mean (allowed values are from 0 to 3, where 0 = 1, 1 = 4, 2 = 16, 3 = 64) Pkho - Peak Hold Off (ns) blho - Base Line Hold Off (ns) otrej - N/A (you can use zero) trgho - Trigger Hold Off (ns) twwdt - RTD Window Width (When 0, the RTD is disabled) trgw - N/A dgain - Input Digital Gain (allowed values are from 0 to 3, where 0 = 1, 1 = 2, 2 = 4, 3 = 8) decimation - Decimation (allowed values are from 0 to 3, where 0 = 0, 1 = 2, 2 = 4, 3 = 8) enf - Trapezoidal Gain (default value to use 1) *CAEN_DGTZ_DPP_PSD_Params_t* blthr – Baseline Threshold to be defined when nsbl = 0 (Baseline Fixed) (LSB) bltmo – N/A trgho – Trigger hold-off (samples) thr – Trigger Threshold (LSB) selft – Self-Trigger (options: 0 = Disabled, 1 = Enabled) csens – Charge Sensitive (options for x720: 0= 40, 1 = 160, 2= 640, 3 = 2560 fC/LSB; options for x751: 0 = 20, 1 = 40, 2 = 80, 3 = 160, 4 = 320, 5 = 640 fC/LSB) sgate – Short Gate Width (samples) lgate – Long Gate Width (samples) pgate – Gate Offset (samples)

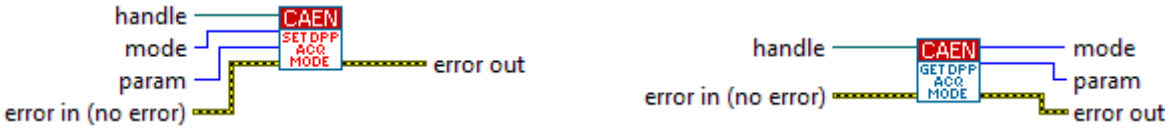
		tvaw – Trigger Validation Acceptance Window (samples) nsbl – Number of Samples for Baseline Mean (options for x720: 0 = FIXED, 1 = 8, 2 = 32, 3 = 128; options for x751: 0 = FIXED, 1 = 8, 2 = 16, 3 = 32, 4 = 64, 5 = 128, 6 = 256, 7 = 512) trgc – must be set to 1 = CAEN_DGTZ_DPP_TriggerConfig_Threshold, the configuration CAEN_DGTZ_DPP_TriggerConfig_Peak is no longer available purh – select Pile-Up option: 0 =CAEN_DGTZ_DPP_PSD_PUR_DetectOnly, 1 = CAEN_DGTZ_DPP_PSD_PUR_Enabled purgap – set the Pile-Up Rejection GAP value (LSB) *CAEN_DGTZ_DPP_CI_Params_t* blthr – Baseline Threshold to be defined when nsbl = 0 (Baseline Fixed) (LSB) bltmo – N/A trgho – Trigger hold-off (samples) thr – Trigger Threshold (LSB) selft – Self-Trigger (options: 0 = Disabled, 1 = Enabled) csens – Charge Sensitive (options 0= 40, 1 = 160, 2= 640, 3 = 2560 fC/LSB) gate – Gate Width (samples) pgate – Gate Offset (samples) tvaw – Trigger Validation Acceptance Window (samples) nsbl – Number of Samples for Baseline Mean n (options for x720: 0 = FIXED, 1 = 8, 2 = 32, 3 = 128) trgc – must be set to 1 = CAEN_DGTZ_DPP_TriggerConfig_Threshold, the configuration CAEN_DGTZ_DPP_TriggerConfig_Peak is no longer available. For a complete description of the parameters refer to each specific DPP User Manual.

Note: one sample corresponds to 4 ns for x720, and 1 ns for x751.

Set / GetDPPAcquisitionMode

Description
Sets/gets the DPP acquisition mode.

LabVIEW VI



Arguments

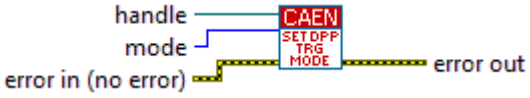
Name	Type (Set)	Type (Get)	Description
`handle`			Device handler
`mode`	U32	U32	The DPP acquisition mode to set/get: *CAEN_DGTZ_DPP_AcqMode_t*. CAEN_DGTZ_DPP_ACQ_MODE_Oscilloscope = 0: enables the acquisition of the samples of the digitized waveforms, CAEN_DGTZ_DPP_ACQ_MODE_List = 1: enables the acquisition of time stamps and energy values in case of DPP-PHA, or charge in case of DPP-CI and DPP-PSD, CAEN_DGTZ_DPP_ACQ_MODE_Mixed = 2: enables the acquisition of both waveforms, energies or charges, and time stamps. See the Set/GetDPPAcquistionMode function in **[RD1]**.
`param`	U32	U32	The acquisition data to retrieve in acquisition: *CAEN_DGTZ_DPP_SaveParam_t*. CAEN_DGTZ_DPP_SAVE_PARAM_EnergyOnly = 0, CAEN_DGTZ_DPP_SAVE_PARAM_TimeOnly = 1, CAEN_DGTZ_DPP_SAVE_PARAM_EnergyAndTime = 2, CAEN_DGTZ_DPP_SAVE_PARAM_ChargeAndTime = 4, CAEN_DGTZ_DPP_SAVE_PARAM_None = 3. See the Set/GetDPPAcquistionMode function in **[RD1]**.

Set / GetDPPTriggerMode

Description
Sets/gets the DPP Trigger mode.

Note: to be used only with DPP-PSD and DPP-CI enabled firmware.

LabVIEW VI



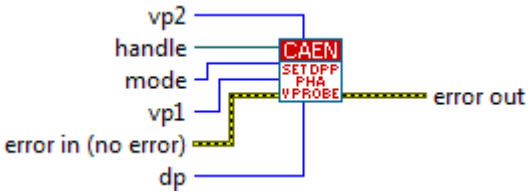
Arguments

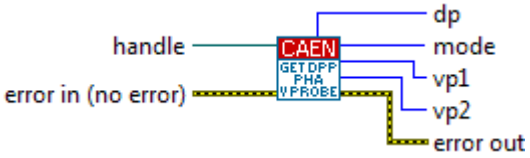
Name	Type (Set)	Type (Get)	Description
`handle`			Device handler
`mode`	U32	U32	Concerning SetDPPTriggerMode, it is the desired trigger mode which can be set: *CAEN_DGTZ_DPP_TriggerMode_t*. CAEN_DGTZ_DPP_TriggerMode_Normal, CAEN_DGTZ_DPP_TriggerMode_Coincidence. Concerning GetDPPTriggerMode, it is the current trigger mode.

Set / GetDPP_PHA_VirtualProbe

Description
Set/gets the information about the output signal of the DPP-PHA acquisition mode.

LabVIEW VI





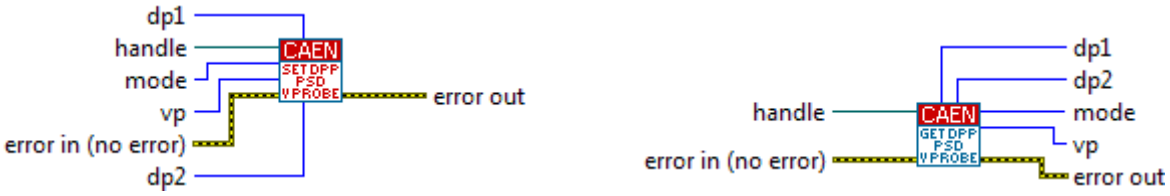
Arguments

Name	Type (Set)	Type (Get)	Description
`handle`			Device handler
`mode`	U32	U32	The Virtual Probe mode to set/get: *CAEN_DGTZ_DPP_VirtualProbe_t*. CAEN_DGTZ_DPP_VIRTUALPROBE_SINGLE = 0, CAEN_DGTZ_DPP_VIRTUALPROBE_DUAL = 1.
`vp1`	U32	U32	The Virtual Probe1 mode to set/get: *CAEN_DGTZ_DPP_PHA_VirtualProbe1_t*. CAEN_DGTZ_DPP_PHA_VIRTUALPROBE1_Input = 0, CAEN_DGTZ_DPP_PHA_VIRTUALPROBE1_Delta = 1, CAEN_DGTZ_DPP_PHA_VIRTUALPROBE1_Delta2 = 2, CAEN_DGTZ_DPP_PHA_VIRTUALPROBE1_trapezoid = 3.
`vp2`	U32	U32	The Virtual Probe2 mode to set/get: *CAEN_DGTZ_DPP_PHA_VirtualProbe2_t*. CAEN_DGTZ_DPP_PHA_VIRTUALPROBE2_Input = 0, CAEN_DGTZ_DPP_PHA_VIRTUALPROBE2_S3 = 1, CAEN_DGTZ_DPP_PHA_VIRTUALPROBE2_DigitalCombo = 2, CAEN_DGTZ_DPP_PHA_VIRTUALPROBE2_trapBaseline = 3, CAEN_DGTZ_DPP_PHA_VIRTUALPROBE2_None = 4 See the Set/GetDPP_PHA_VirtualProbe function in **[RD1]**.
`dp`	U32	U32	The Digital Probe mode to set/get: *CAEN_DGTZ_DPP_PHA_DigitalProbe_t*. CAEN_DGTZ_DPP_PHA_DIGITAL_PROBE_trgWin = 0, CAEN_DGTZ_DPP_PHA_DIGITAL_PROBE_Armed = 1, CAEN_DGTZ_DPP_PHA_DIGITAL_PROBE_PkRun = 2, CAEN_DGTZ_DPP_PHA_DIGITAL_PROBE_PURFlag = 3, CAEN_DGTZ_DPP_PHA_DIGITAL_PROBE_Peaking = 4, CAEN_DGTZ_DPP_PHA_DIGITAL_PROBE_TVAW = 5, CAEN_DGTZ_DPP_PHA_DIGITAL_PROBE_BLHoldoff = 6, CAEN_DGTZ_DPP_PHA_DIGITAL_PROBE_TRGHoldOff = 7, CAEN_DGTZ_DPP_PHA_DIGITAL_PROBE_TRGVal = 8, CAEN_DGTZ_DPP_PHA_DIGITAL_PROBE_ACQVeto = 9, CAEN_DGTZ_DPP_PHA_DIGITAL_PROBE_BFMVeto = 10, CAEN_DGTZ_DPP_PHA_DIGITAL_PROBE_ExtTRG = 11. Check the specific meaning of the probes in the DPP-PHA User Manual **[RD12]**

Set / GetDPP_PSD_VirtualProbe

Description
Sets/gets the information about the output signal of the DPP-PSD acquisition mode.

LabVIEW VI



Arguments

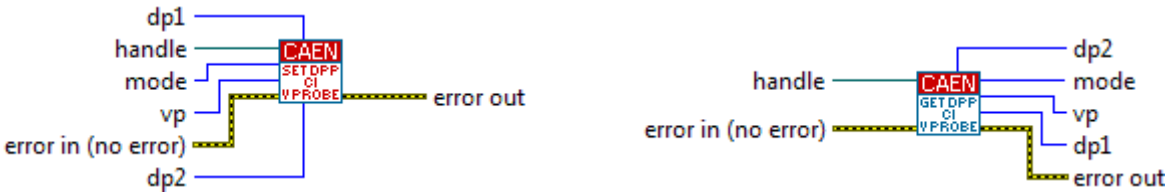
Name	Type (Set)	Type (Get)	Description
handle			Device handler.
mode	U32	U32	The Virtual Probe mode to set/get: *CAEN_DGTZ_DPP_VirtualProbe_t*. CAEN_DGTZ_DPP_VIRTUALPROBE_SINGLE = 0, CAEN_DGTZ_DPP_VIRTUALPROBE_DUAL = 1. See the Set/GetDPP_PSD_VirtualProbe function in **[RD1]**.
vp	U32	U32	The Virtual Probe to set/get: *CAEN_DGTZ_DPP_PSD_VirtualProbe_t*. CAEN_DGTZ_DPP_PSD_VIRTUALPROBE_Baseline = 0, CAEN_DGTZ_DPP_PSD_VIRTUALPROBE_Threshold = 1 See the Set/GetDPP_PSD_VirtualProbe function in **[RD1]**. **NOTE:** ignored for x751; VirtualProbes are always Input and Baseline
dp1	U32	U32	The Digital Probe1 to set/get: *CAEN_DGTZ_DPP_PSD_DigitalProbe1_t*. **Digital Probes Types:** CAEN_DGTZ_DPP_PSD_DIGITALPROBE1_R6_ExtTrg = 11, /* x720 only */ CAEN_DGTZ_DPP_PSD_DIGITALPROBE1_R6_OverThr = 12, CAEN_DGTZ_DPP_PSD_DIGITALPROBE1_R6_TrigOut = 13, CAEN_DGTZ_DPP_PSD_DIGITALPROBE1_R6_CoincWin = 14, CAEN_DGTZ_DPP_PSD_DIGITALPROBE1_R6_PileUp = 15, CAEN_DGTZ_DPP_PSD_DIGITALPROBE1_R6_Coincidence = 16, CAEN_DGTZ_DPP_PSD_DIGITALPROBE1_R6_GateLong = 17, /* x751 only */
dp2	U32	U32	The Digital Probe2 to set/get: *CAEN_DGTZ_DPP_PSD_DigitalProbe2_t*. **Digital Probes Types:** CAEN_DGTZ_DPP_PSD_DIGITALPROBE2_R6_GateShort = 11, CAEN_DGTZ_DPP_PSD_DIGITALPROBE2_R6_OverThr = 12, CAEN_DGTZ_DPP_PSD_DIGITALPROBE2_R6_TrgVal = 13, CAEN_DGTZ_DPP_PSD_DIGITALPROBE2_R6_TrgHO = 14, CAEN_DGTZ_DPP_PSD_DIGITALPROBE2_R6_PileUp = 15, CAEN_DGTZ_DPP_PSD_DIGITALPROBE2_R6_Coincidence = 16,

Set / GetDPP_CI_VirtualProbe

Description
Sets/gets the information about the output signal of the DPP-CI acquisition mode.

Note: this function is supported only by DPP-CI firmware from release 3.4_130.16 on.

LabVIEW VI



Arguments

Name	Type (Set)	Type (Get)	Description
handle			Device handler
mode	U32	U32	The Virtual Probe mode to set/get: *CAEN_DGTZ_DPP_VirtualProbe_t*. CAEN_DGTZ_DPP_VIRTUALPROBE_SINGLE = 0, CAEN_DGTZ_DPP_VIRTUALPROBE_DUAL = 1. See the Set/GetDPP_CI_VirtualProbe function in **[RD1]**.
vp	U32	U32	The Virtual Probe to set/get: *CAEN_DGTZ_DPP_CI_VirtualProbe_t*. CAEN_DGTZ_DPP_CI_VIRTUALPROBE_Baseline = 0. See the Set/GetDPP_CI_VirtualProbe function in **[RD1]**.
dp1	U32	U32	The Digital Probe1 to set/get: *CAEN_DGTZ_DPP_CI_DigitalProbe1_t*. CAEN_DGTZ_DPP_CI_DIGITALPROBE1_R22_ExtTrg = 4, CAEN_DGTZ_DPP_CI_DIGITALPROBE1_R22_OverThr = 5, CAEN_DGTZ_DPP_CI_DIGITALPROBE1_R22_TrigOut = 6, CAEN_DGTZ_DPP_CI_DIGITALPROBE1_R22_CoincWin = 7, CAEN_DGTZ_DPP_CI_DIGITALPROBE1_R22_Coincidence = 9,
dp2	U32	U32	The Digital Probe2 to set/get: *CAEN_DGTZ_DPP_CI_DigitalProbe2_t*. CAEN_DGTZ_DPP_CI_DIGITALPROBE2_R22_OverThr = 5, CAEN_DGTZ_DPP_CI_DIGITALPROBE2_R22_TrgVal = 6, CAEN_DGTZ_DPP_CI_DIGITALPROBE2_R22_TrgHO = 7, CAEN_DGTZ_DPP_CI_DIGITALPROBE2_R22_Coincidence = 9,

6 HV specific VIs

Among the different digitizer family of CAEN, the DT5780 and DT5790 desktop digitizers integrate also two High Voltage (HV) channels. In this chapter the specific functions to control the HV are described.

Read_HV_Status

Description
This function allows the user to read the status of a HV channel. The status is represented both as integer value, as well as an array of bool, where each bit returns true or false, according to the following table.

Note: the user must set in advance the IVMon monitor from function **Set / Get_HV_MonitorMode**.

LabVIEW VI

Arguments

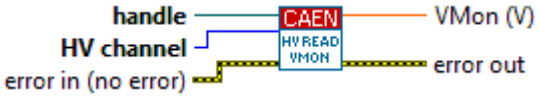
Name	Type	Description
handle		Device handler
HV channel	U32	The HV channel
Status	U16	The integer word identifying the HV channel status
Status bit		Array of Boolean bits, which have the following meaning: Bit 0 = 1 -> ON Bit 1 = 1 -> Ramp UP Bit 2 = 1 -> Ramp DOWN Bit 3 = 1 -> OVER CURRENT (IMON > ISET) Bit 4 = 1 -> OVER VOLTAGE (VMON > VSET + 2%) Bit 5 = 1 -> UNDER VOLTAGE (VMON < VSET - 2%) Bit 6 = 1 -> MAX VOLTAGE (VOUT > VMAX) Bit 7 = 1 -> MAX CURRENT (IOUT > Absolute Max Iout) Bit 8 = 1 -> TEMPERATURE WARNING (TEMP > 80ºC) Bit 9 = 1 -> OVER TEMPERATURE (TEMP > 125ºC) Bit 10 = 1 -> DISABLED (Active External Inhibit) Bit 11 = 1 -> CALIBRATION ERROR Bit 12 = 1 -> Resetting Bit 13 = 1 -> Going Off Bit 14 = 1 -> MAX POWER (OUTPUT POWER > 4W) Bit 15 = 1 -> FAN SPEED HIGH

Read_HV_VMon

Description
This function allows the user to read the monitoring voltage of a specific HV channel.

Note: the user must set in advance the IVMon monitor from function **Set / Get_HV_MonitorMode**.

LabVIEW VI



Arguments

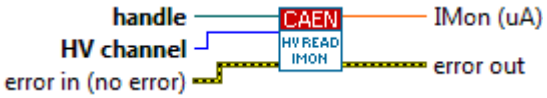
Name	Type	Description
handle		Device handler
HV channel	U32	The HV channel
VMon	DBL	Value in Volt of the voltage for the specific HV channel

Read_HV_IMon

Description
This function allows the user to read the monitoring current of a specific HV channel.

Note: the user must set in advance the IVMon monitor from function **Set / Get_HV_MonitorMode**.

LabVIEW VI



Arguments

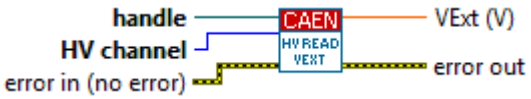
Name	Type	Description
handle		Device handler
HV channel	U32	The HV channel
IMon	DBL	Value in uA of the current for the specific HV channel

Read_HV_VExt

Description
This function allows the user to read the external voltage of a device connected to the PREAMP connectors. Refer to **[RD8]** and **[RD9]** for more details about the PREAMP pin-out.

Note: the user must set in advance the ExtMon monitor from function **Set / Get_HV_MonitorMode**.

LabVIEW VI



Arguments

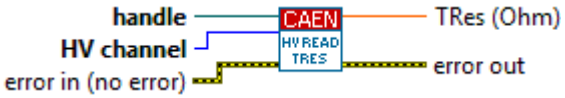
Name	Type	Description
handle		Device handler
HV channel	U32	The PREAMP channel
VExt	DBL	Value in Volt of the voltage of the external device connected to the PREAMP connector

Read_HV_TRes

Description
This function allows the user to read the value of the resistance from temperature probe of type pt100/pt1000. The probe has to be connected to the PREAMP connector. Refer to **[RD8]** and **[RD9]** for more details about the PREAMP pin-out.

Note: the user must set in advance the ExtMon monitor from function **Set / Get_HV_MonitorMode**.

LabVIEW VI



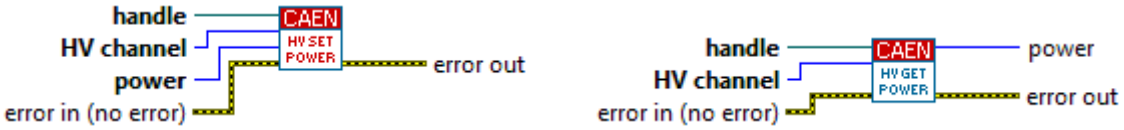
Arguments

Name	Type	Description
handle		Device handler
HV channel	U32	The PREAMP channel
TRes	DBL	Value in Ohm of the resistance of the temperature probe

Set / Get_HV_Power

Description
The Set function allows the user to provide the power ON/shut down command to the board. In case of Power ON the HV channel will reach the voltage value set on function **Set / Get_HV_VSet**. The Get function returns the power status.

LabVIEW VI



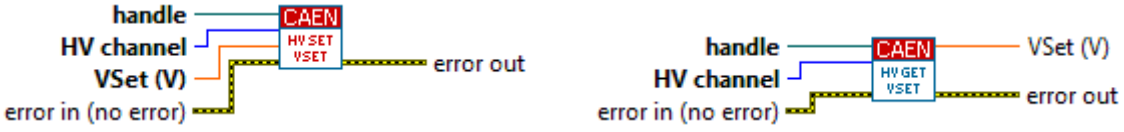
Arguments

Name	Type (Set)	Type (Get)	Description
handle			Device handler
HV channel	U32	U32	The HV channel
power	U32	U32	Power status: 1 = ON, 0 = OFF

Set / Get_HV_VSet

Description
This function allows the user to set the desired value of voltage for a specific HV channel or to retrieve the corresponding information.

LabVIEW VI



Arguments

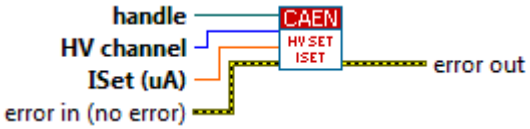
Name	Type (Set)	Type (Get)	Description
handle			Device handler
HV channel	U32	U32	The HV channel
VSet	DBL	DBL	Value in Volt of the voltage for the specific HV channel

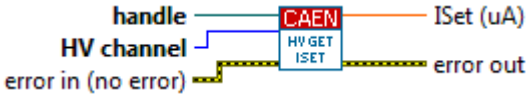
Set / Get_HV_ISet

Description

This function allows the user to set the maximum value of current that the HV channel might supply. In case of overcurrent the channel will give an OVCURR error and it automatically shuts down.

LabVIEW VI





Arguments

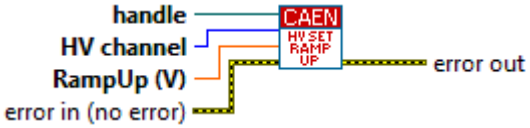
Name	Type (Set)	Type (Get)	Description
handle			Device handler
HV channel	U32	U32	The HV channel
ISet	DBL	DBL	Value in uA of the current for the specific HV channel

Set / Get_HV_RampUp

Description

This function allows the user to set the desired value of voltage/sec of voltage ramp up for a specific HV channel. The Get function retrieves the corresponding information.

LabVIEW VI



Arguments

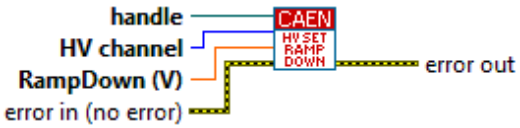
Name	Type (Set)	Type (Get)	Description
handle			Device handler
HV channel	U32	U32	The HV channel
RampUp	DBL	DBL	Value in Volt/s for the voltage ramp up of the specific HV channel

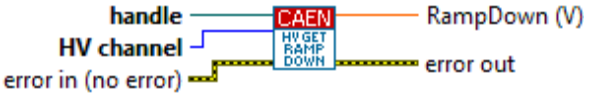
Set / Get_HV_RampDown

Description

This function allows the user to set the desired value of voltage/sec of voltage ramp down for a specific HV channel. The Get function retrieves the corresponding information.

LabVIEW VI





Arguments

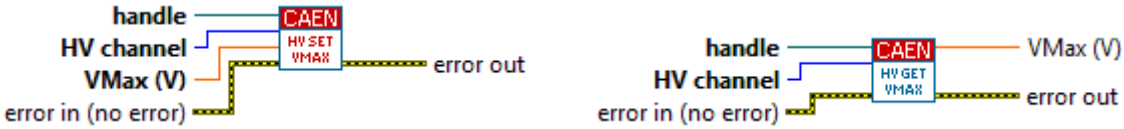
Name	Type (Set)	Type (Get)	Description
handle			Device handler
HV channel	U32	U32	The HV channel
RampDown	DBL	DBL	Value in Volt/s for the voltage ramp down of the specific HV channel

Set / Get_HV_VMax

Description

This function allows the user to set the maximum value of voltage, beyond that the HV channel gives the MAXV error and automatically shuts down. The Get function retrieves the corresponding information.

LabVIEW VI



Arguments

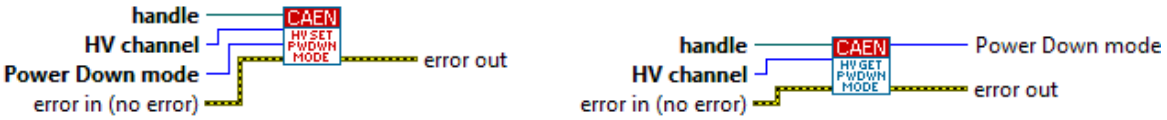
Name	Type (Set)	Type (Get)	Description
handle			Device handler
HV channel	U32	U32	The HV channel
VMax	DBL	DBL	Value in Volt of the maximum voltage for the specific HV channel

Set / Get_HV_PWDownMode

Description

This function allows the user to set/get the power down mode, choosing between RAMP and KILL. In the RAMP mode the HV channel turns off with the voltage/sec speed as defined in the **Set / Get_HV_RampDown** function. In the KILL mode the HV channel turns off the voltage instantaneously. In case of alarm (OVCURR, OVTEMP, MAXPW bit status) or External Inhibit (DISABLE bits status) the HV channel automatically shuts down with the set Power Down mode.

LabVIEW VI



Arguments

Name	Type (Set)	Type (Get)	Description
handle			Device handler
HV channel	U32	U32	The HV channel
Power Down mode	U32	U32	Power Down mode: 0 = Ramp, 1 = Kill

Set / Get_HV_MonitorMode

Description
This function allows the user to set/get the HV monitor mode, choosing between IVMon or ExtMon.

LabVIEW VI

Arguments

Name	Type (Set)	Type (Get)	Description
handle			Device handler
HV channel	U32	U32	The HV channel
Monitor mode	U32	U32	Monitor Mode: 0 = IVMon, 1 = ExtMon

Note: according to the programmed HV monitor mode, the following board registers will change their specific value.

Register	IVMon	ExtMon
0x1n40 (n=2 forHV channel 0; n=3 for HV channel 1)	VMon value	VExt value
0x1n38 (n=2 forHV channel 0; n=3 for HV channel 1)	Status	A639 Firmware Release
0x1n44 (n=2 forHV channel 0; n=3 for HV channel 1)	IMon value	TRes value

Get_HV_ChannelsInfo

Description
This function gives to the user the HV channels list. For each channel the function retrieves the Max, Min, and Res value for the VSet, ISet, RampUp, RampDown, and VMax settings.

Note: Res is the resolution step for the specific parameter.

LabVIEW VI

Arguments

Name	Type	Description
handle		Device handler
Num HV Channels	I32	The HV channel numbers
HV Channels Info		Infos of the HV Channels Cluster VSet: Min, Max, Res ISet: Min, Max, Res RampUp: Min, Max, Res RampDown: Min, Max, Res VMax: Min, Max, Res

Reset_HV

Description

This function reset the HV channels to the following default configuration:

- VSet = 0 V
- ISet = ISetMAX / 10
- RampUp = RampUpMAX / 10
- RampDown = RampDownMAX / 10
- VMax = VMaxMAX
- PWDownMode = Ramp

LabVIEW VI

Arguments

Name	Type	Description
handle		Device handler
HV channel	U32	The HV channel

7 DEMOs

This chapter provides some demo LabVIEW GUI to handle with digitizers with standard and DPP firmware. The user can take those LabVIEW codes for free and modify them to implement specific functionalities.
For any information refer to the support mailing list at Chapter **Technical support**.

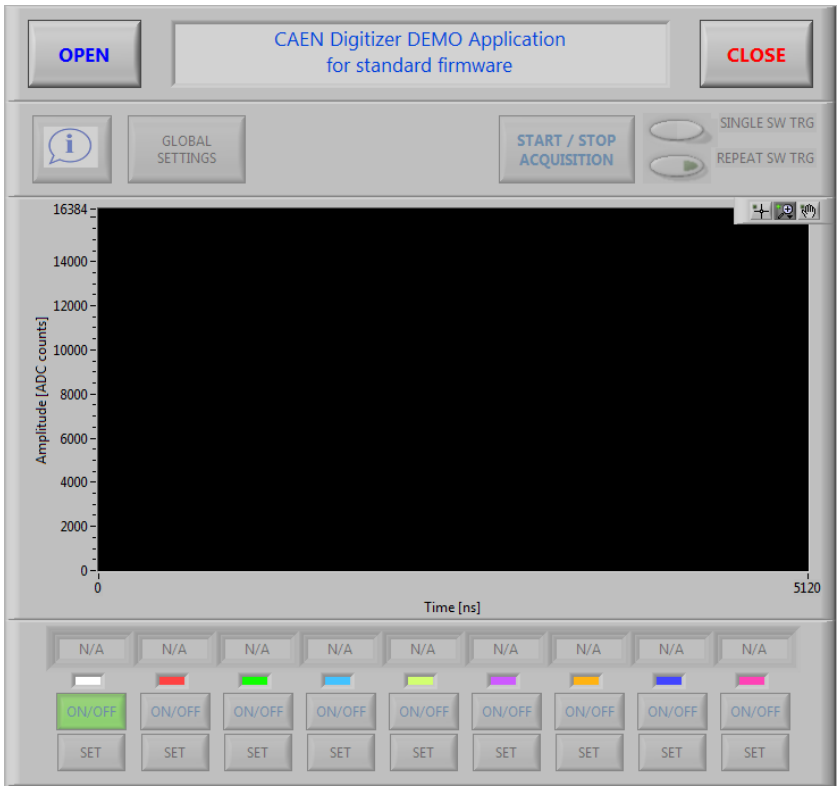
Digitizer Demo for firmware standard

The following demo is compliant with digitizers running standard firmware. The demo allows the user to configure the board and to acquire and visualize the waveforms. The following functionalities are not implemented:

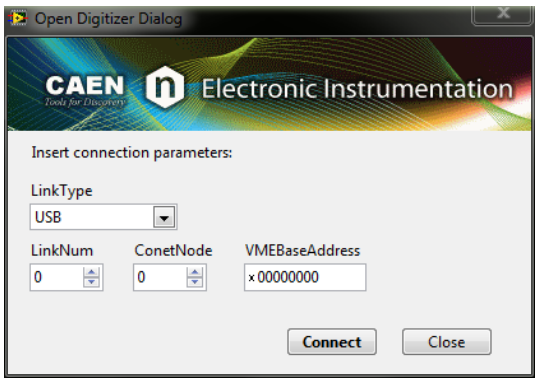
- Save and reload the configuration file;
- Save of data event

The demo can manage up to eight channels per board. In case of x730 the demo can manage the first eight channels, while for x740 and x742 the user can select one group of eight channels.

The main graphical interface appears as in the following figure:

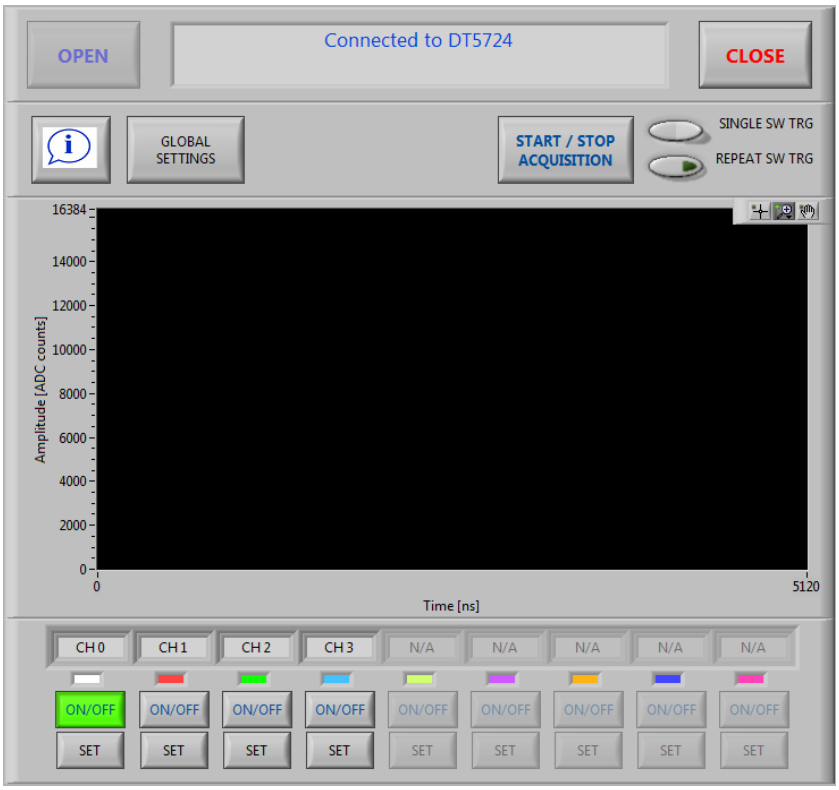


1. Press **OPEN** to connect the demo software to the digitizer. The **"Open Digitizer Dialog"** window will appear.

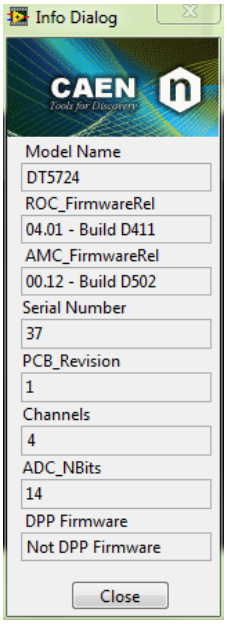


Set the correct connection parameters and press **"Connect".**

2. If the connection succeeded the following window will appear: only the available channels will be active in the GUI. In the following example we connected to a DT5724 which has 4 channels.
   In case of x740 and x742 a pop-up will appear asking for the group to acquire. In case of x730 it is possible to select only the first eight channels.

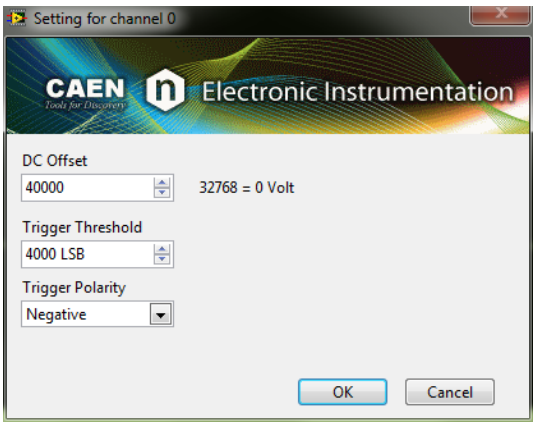


3. Press the **INFO** button to retrieve a summary of the board and firmware info



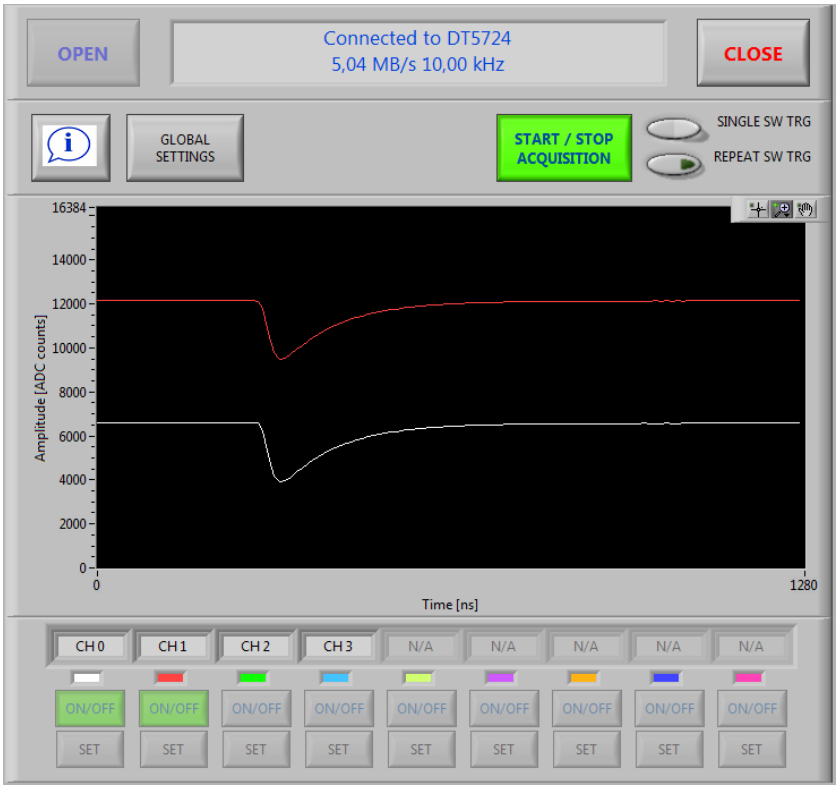
4. Press the **"Global Settings"** button to set all the common parameters of the board, as the Record Length and the Post-Trigger size.

5. For each channel it is possible to enable/disable through the **ON/OFF** button, and to set the individual settings through the **SET** button, which opens the following dialog window.



Note: set the polarity only for the first channel, than the same value will be copied for all channels.

6. Press **"Start/stop acquisition"** to enable the waveform visualization of the enabled channels.



7. It is possible to enable the Software trigger through the **"Single SW TRG"** for a single trigger, and **"Repeat SW TRG"** for a continuous software trigger.

8. **Stop** the acquisition before closing the application. Also, stop the acquisition before changing any setting.

9. Press **CLOSE** to disconnect from the device.

The demo software for firmware standard does not save any configuration nor the data; any time you connect to the device you have to program again the previous settings.

DPP-PSD / DPP-CI Demos

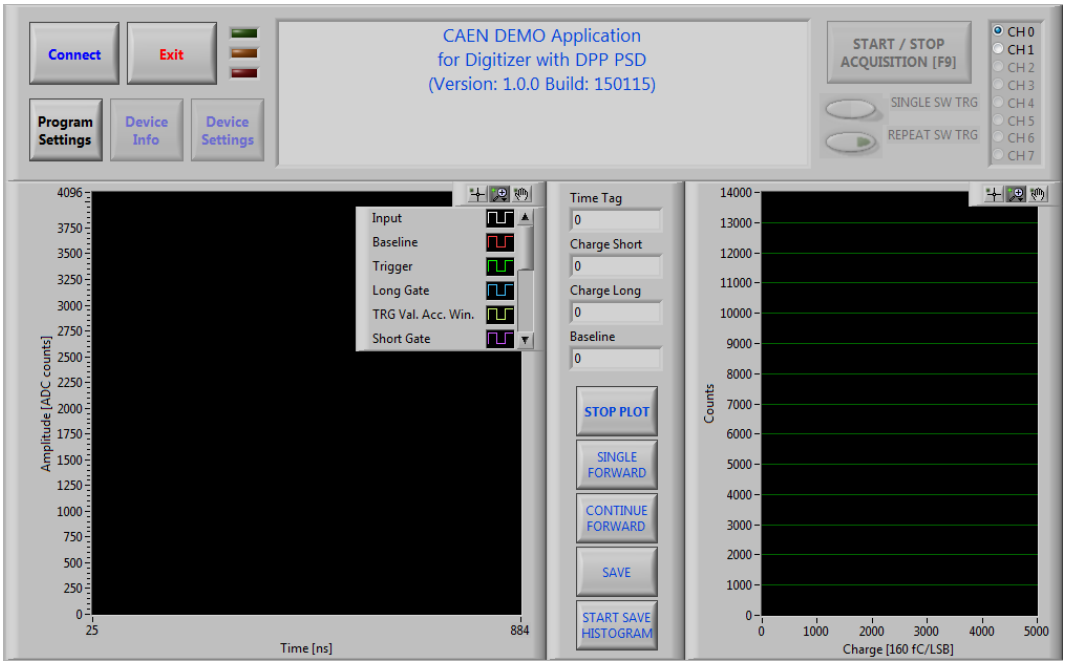
The DPP-PSD demo is compliant with x720 and x751 digitizers running DPP-PSD firmware, while the DPP-CI demo is compliant with x720 digitizers running DPP-CI firmware. The DPP-CI demo has the same functionalities of the DPP-PSD demo apart from the Short/Long Gate management, since the DPP-CI firmware only has one gate option.

The DPP-PSD and DPP-CI demos allow the user to:

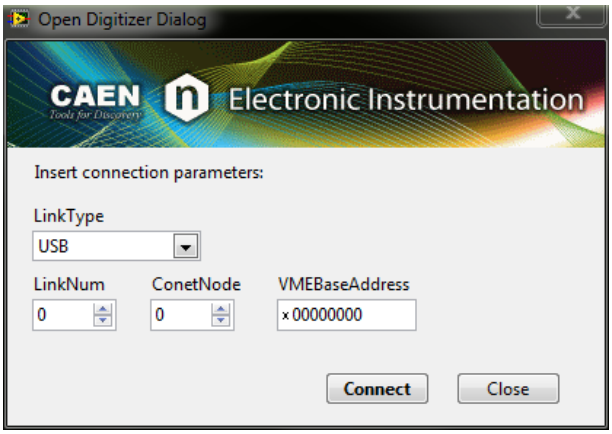
- Save and reload the configuration file;
- Visualize the waveform and histogram plots of a single selected channel (though the acquisition can start/stop for all channels simultaneously);
- Save of data events in the waveform, histogram, and list modes. All enabled channels can be saved simultaneously.

Note: the demo software does not work with DPP-PSD of x730.

The main graphical interface appears as in the following figure:

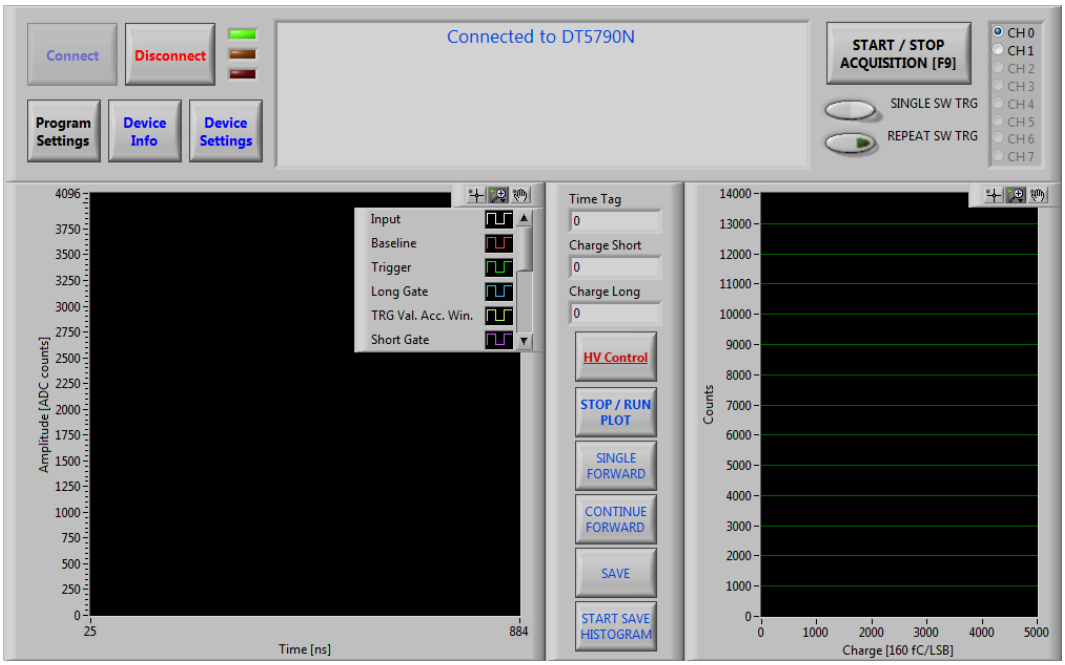


1. Press **OPEN** to connect the demo software to the digitizer. The **"Open Digitizer Dialog"** window will appear.



Set the correct connection parameters and press **"Connect".**

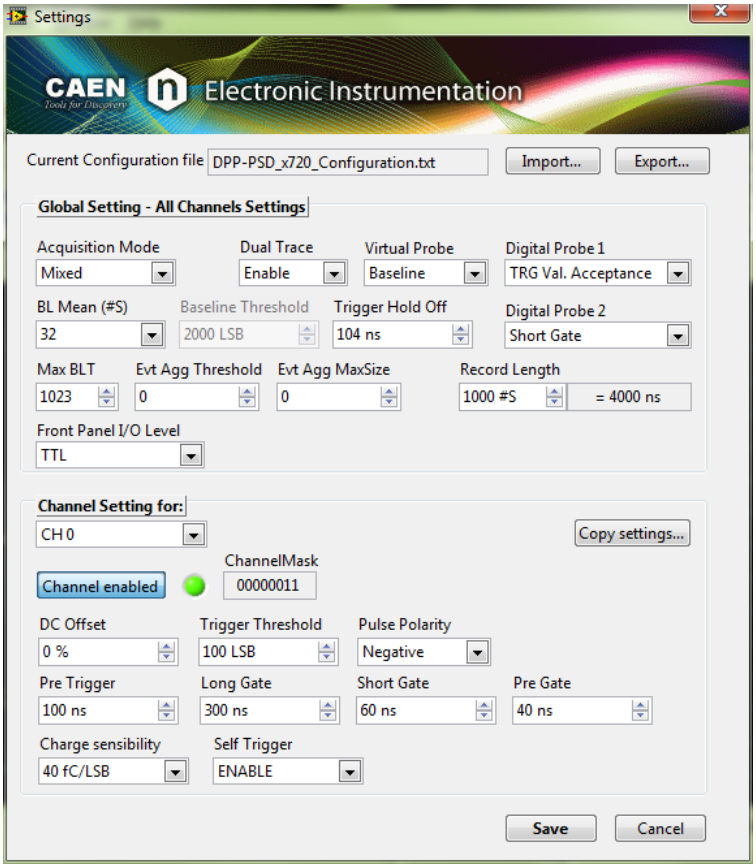
2. If the connection succeeded the following window will appear. The user can visualize only one channel at a time.



3. Press the **INFO** button to retrieve a summary of the board and firmware info

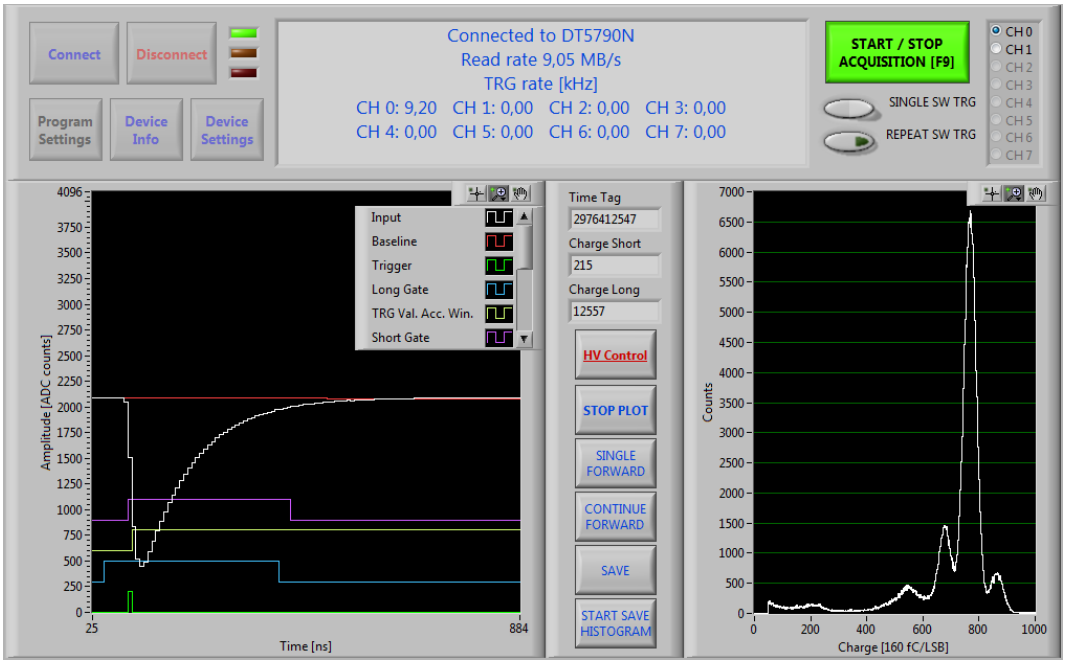
4. Press the **"Program Settings"** button to:
   a. Select the output directory for data saving;
   b. Set the number of bins of the energy histogram;
   c. Set the time for data refresh in the plots.

5. Press the **"Device Settings"** button to configure the board settings, and the individual channel settings.

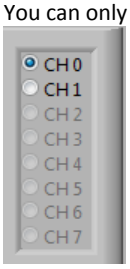


For any details about the DPP-PSD (DPP-CI) settings refer to the specific DPP-PSD User Manual **[RD10]**, DPP-CI User Manual **[RD11]**.
Once ready press **"Save"** to save and load the configuration settings.

6. Press **"Start/stop acquisition"** to start the acquisition of the enabled channels. In our example we enable the **Mixed** acquisition mode, therefore we can visualize both the waveform and the energy spectrum.



You can only visualize one channel at a time. Change the channel through the top right menu



Once you start the acquisition, the demo automatically enables the **list data** saving. To save the energy histogram press **"Start Save Histogram"** button.

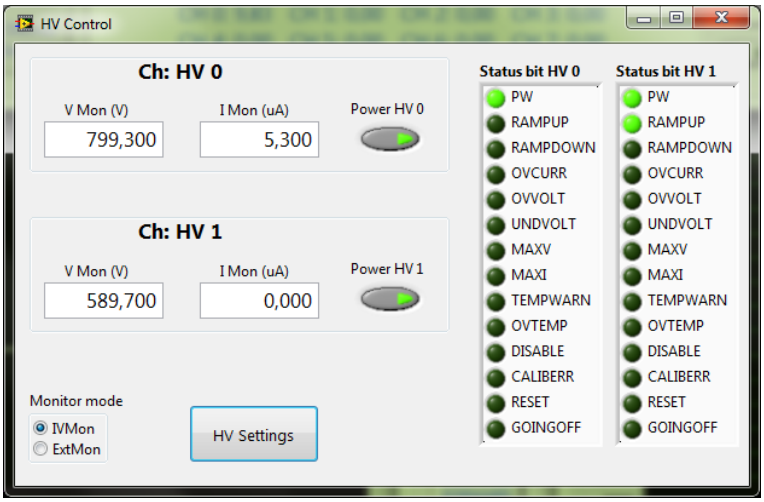
Note: the list data writing overwrites the files previously created.

7. Press **"Stop Plot"** to stop the waveform plotting. Press **"Single Forward"** for a single event visualization, **"Continue Forward"** to check consecutive plots.

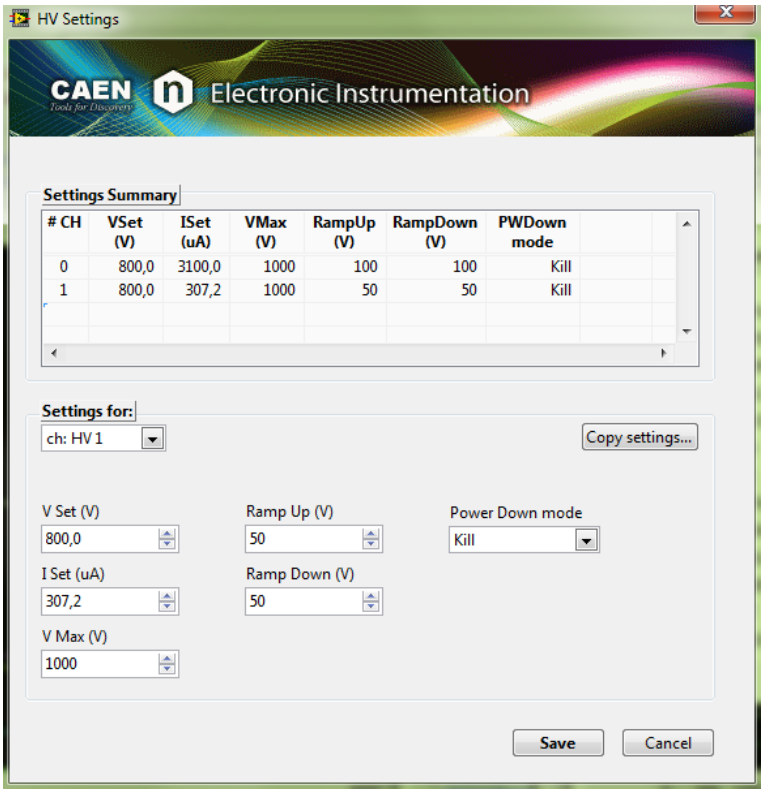
   Press **"SAVE"** to save the waveform and histogram plots in a image format.

8. It is possible to enable the Software trigger through the **"Single SW TRG"** for a single trigger, and **"Repeat SW TRG"** for a continuous software trigger.

9. In case of acquisition though DT5790 it is possible to also manage the HV channels, through the **"HV Control"** panel:



Through this panel the user can monitor the two HV channels. Press **"HV Settings"** to configure the HV channels. Once done, press **"Power HV 0/1"** to power ON/OFF the HV channels.



10. **Stop** the acquisition before closing the application. Also, stop the acquisition before chaging any setting.

11. Press **CLOSE** to disconnect from the device.

DPP-PHA Demo for 724 series, DT5780, and DT5781

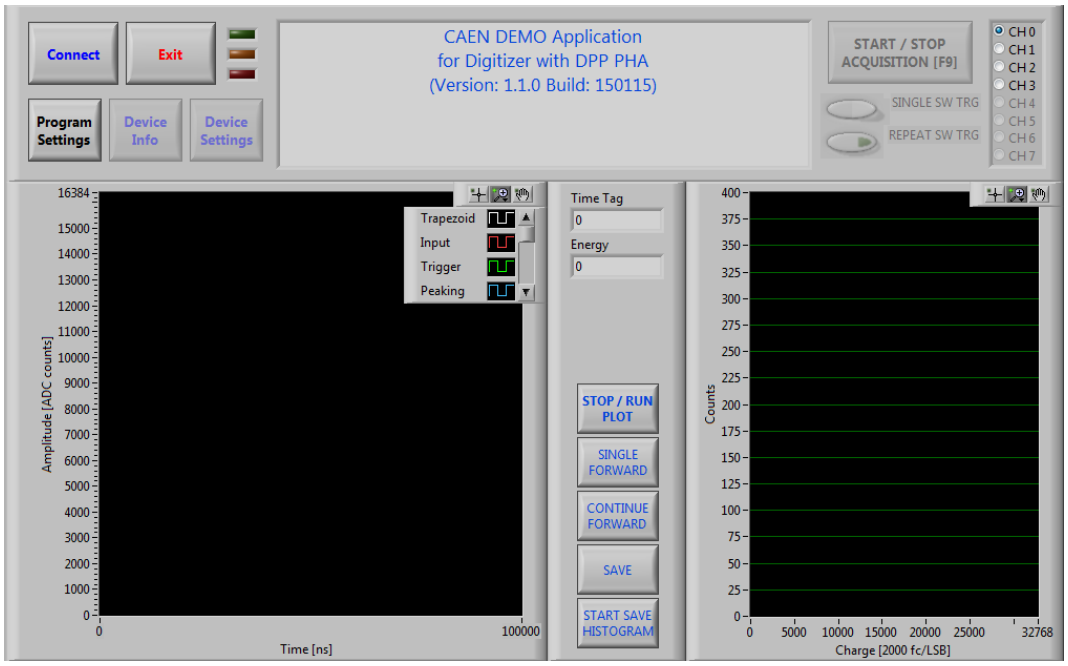
The DPP-PHA demo is compliant with x724 digitizers running DPP-PHA firmware, as well as DT5780, and DT5781 (N6781).

The DPP-PHA demo allows the user to:

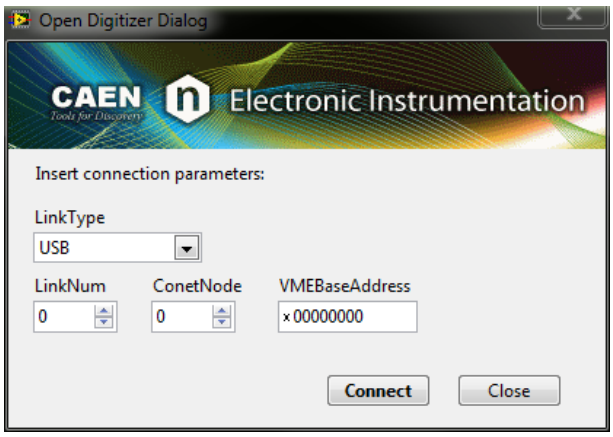
- Save and reload the configuration file;
- Visualize the waveform and histogram plots of a single selected channel (though the acquisition can start/stop all the channels simultaneously);
- Save of data events in the waveform, histogram, and list modes. All enabled channels can be saved simultaneously.

Note: the demo software does not work with DPP-PHA of x730.

The main graphical interface appears as in the following figure:

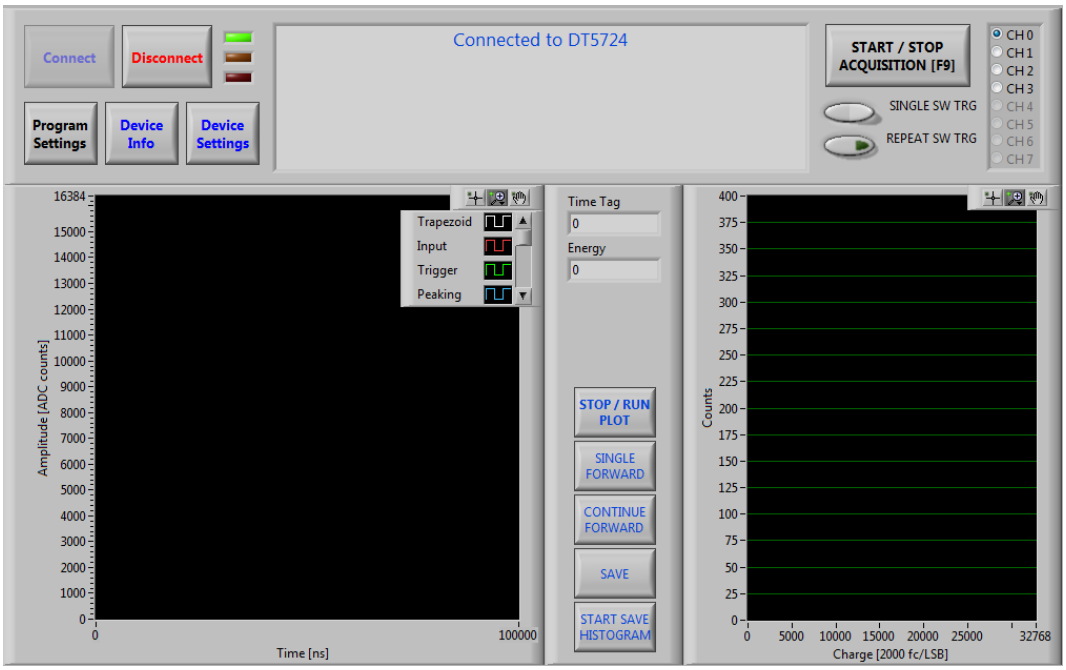


1. Press **OPEN** to connect the demo software to the digitizer. The **"Open Digitizer Dialog"** window will appear.



Set the correct connection parameters and press **"Connect".**

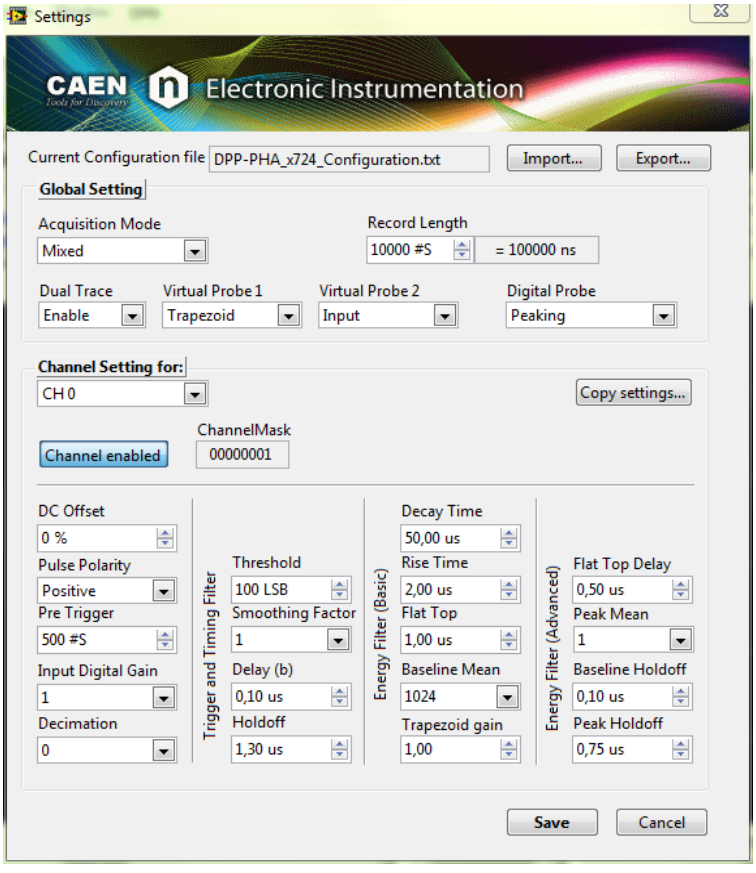
2. If the connection succeeded the following window will appear. The user can visualize only one channel at a time.



3. Press the **INFO** button to retrieve a summary of the board and firmware info

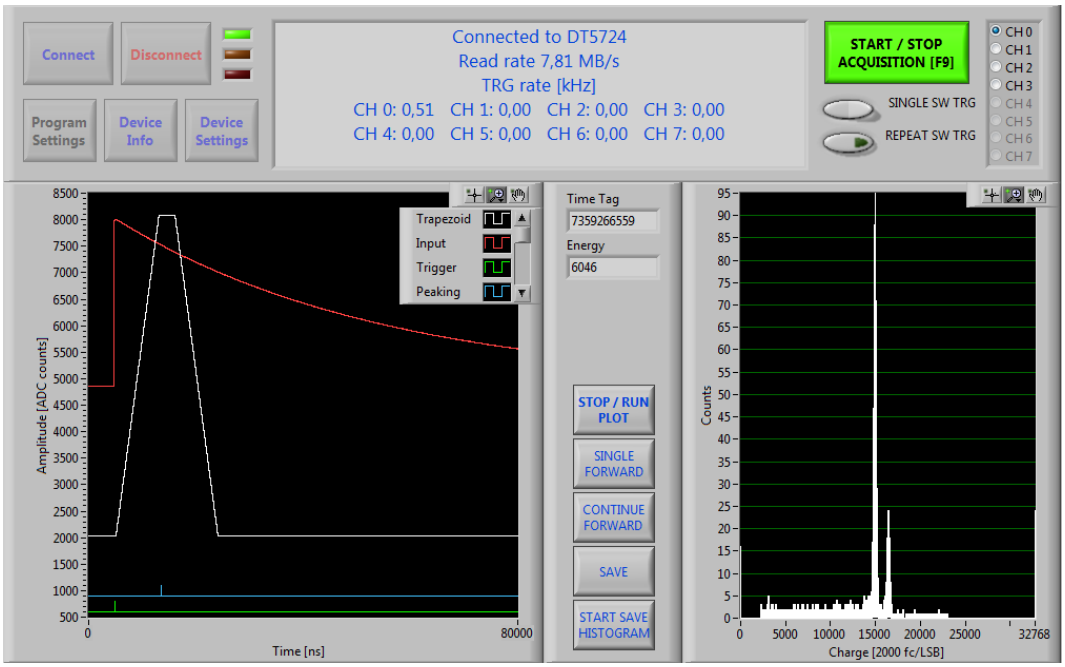
4. Press the **"Program Settings"** button to:
   a. Select the output directory for data saving;
   b. Set the number of bins of the energy histogram;
   c. Set the time for data refresh in the plots.

5. Press the **"Device Settings"** button to configure the board settings, and the individual channel settings.

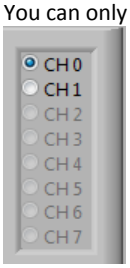


For any details about the DPP-PHA settings refer to the specific DPP-PHA User Manual **[RD12]**.
Once ready press **"Save"** to save and load the configuration settings.

6. Press **"Start/stop acquisition"** to start the acquisition of the enabled channels. In our example we enable the **Mixed** acquisition mode, therefore we can visualize both the waveform and the energy spectrum.



You can only visualize one channel at a time. Change the channel through the top right menu



Once you start the acquisition, the demo automatically enables the **list data** saving. To save the energy histogram press **"Start Save Histogram"** button.

Note: the list data writing overwrites the files previously created.

7. Press **"Stop Plot"** to stop the waveform plotting. Press **"Single Forward"** for a single event visualization, **"Continue Forward"** to check consecutive plots.

   Press **"SAVE"** to save the waveform and histogram plots in a image format.

8. It is possible to enable the Software trigger through the **"Single SW TRG"** for a single trigger, and **"Repeat SW TRG"** for a continuous software trigger.

9. In case of acquisition though DT5780 it is possible to also manage the HV channels, through the **"HV Control"** and **"HV Settings"** panels. Refer to point 9 of the **DPP-PSD / DPP-CI Demos** section.

12. **Stop** the acquisition before closing the application. Also, stop the acquisition before chaging any setting.

13. Press **CLOSE** to disconnect from the device.

8 Examples of communication settings

The examples in this chapter are intended to show how to implement through the LabVIEW's VIs the board access, the board info getting and the board closing in different hardware configuration. Examples of such connection cases are included in the CAENDigitizer LabVIEW at the destination path: *CAEN\Digitizers\LabView\Examples.*

Example No.1

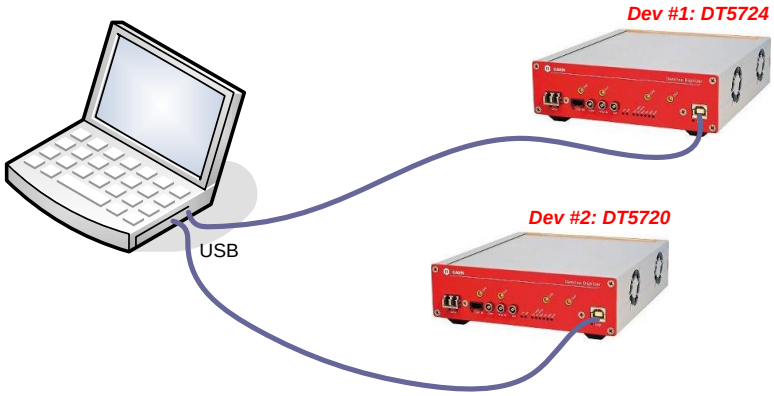
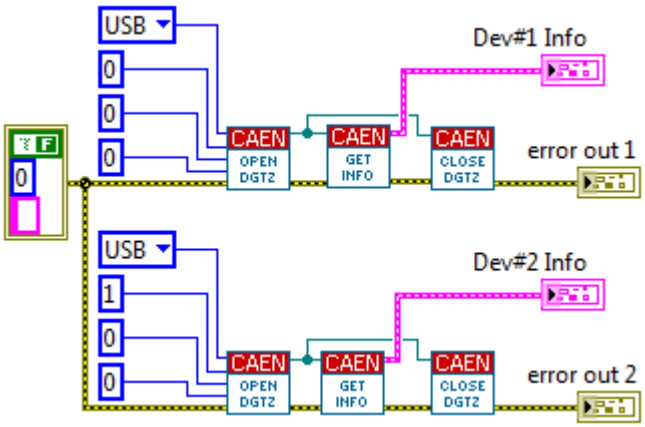


Fig. 8.1: Connection example no.1.

The host PC is connected via 2 USB ports to two desktop digitizer:

- Dev#1: DT5724 - 4 Channel 14 bit 100 MS/s Digitizer
- Dev#2: DT5720 - 4 Channel 12 bit 250 MS/s Digitizer

The computer is first connected to DT5724 then to the DT5720.



Example No.2

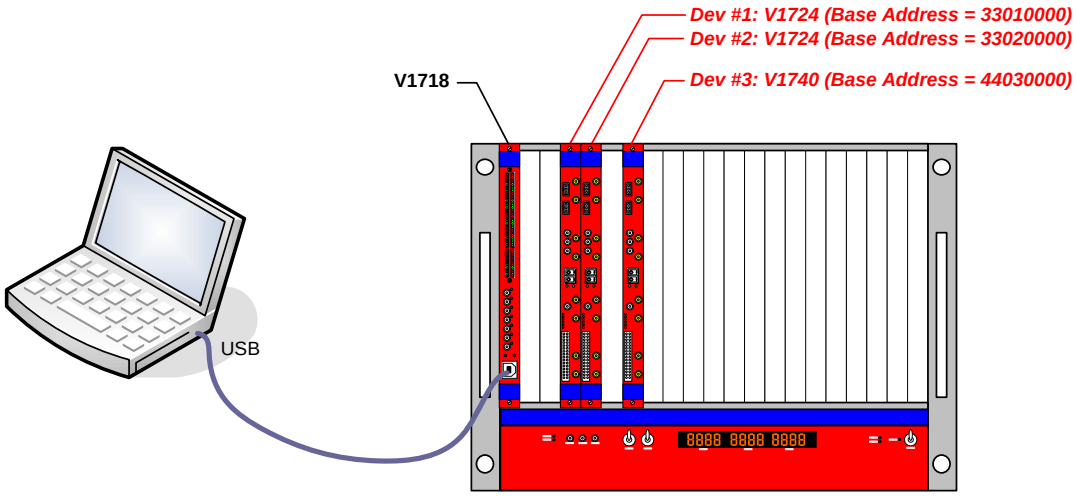
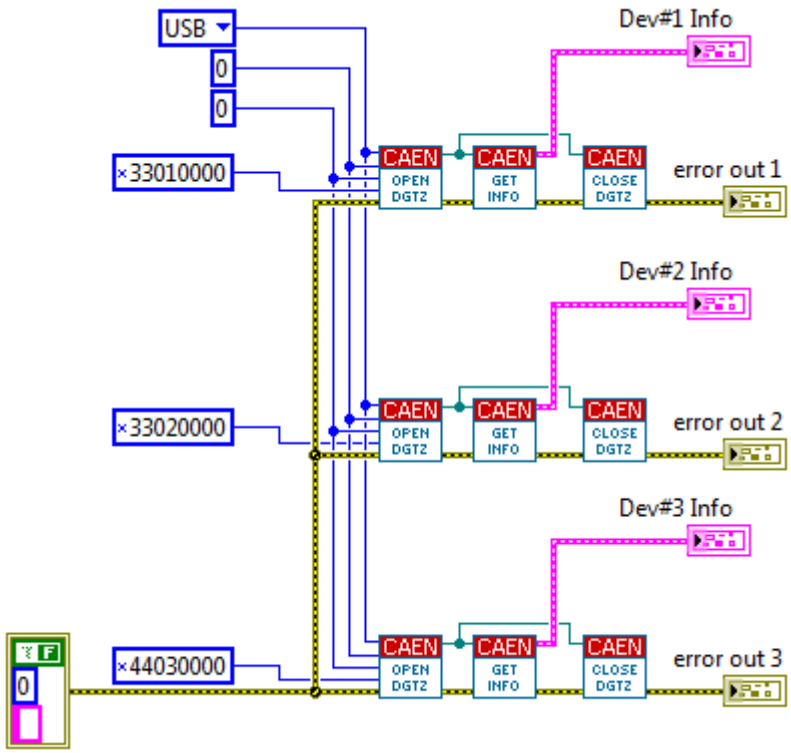


Fig. 8.2: Connection example no.2.

The host PC is connected via USB ports to one V1718 VME-USB2.0 Bridge housed in a VME crate. The crate contains also the following boards

- Dev#1: V1724 - 8 Channel 14 bit 100 MS/s Digitizer (Base address = 0x33010000)
- Dev#2: V1724 - 8 Channel 14 bit 100 MS/s Digitizer (Base address = 0x33020000)
- Dev#3: V1740 - 64 Channel 12 bit 62.5 MS/s Digitizer (Base address = 0x44030000)



Example No.3

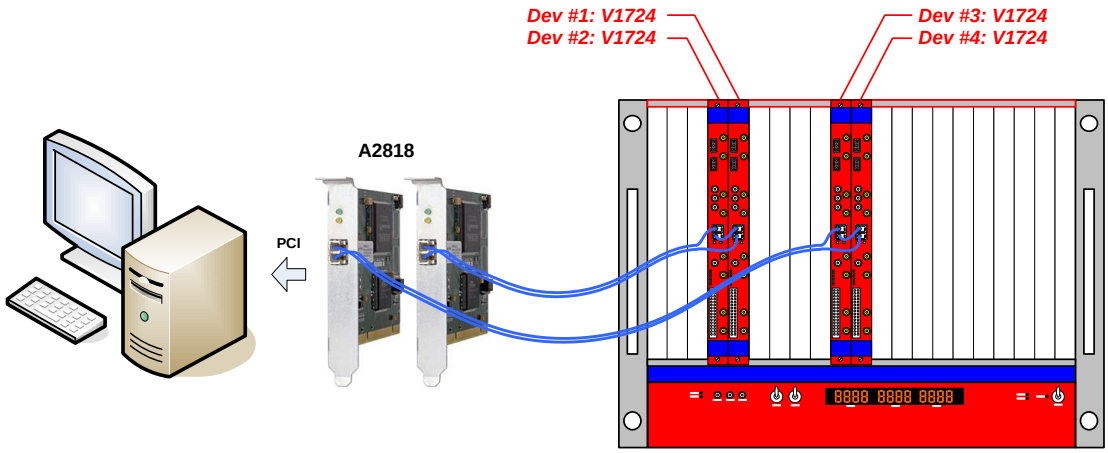
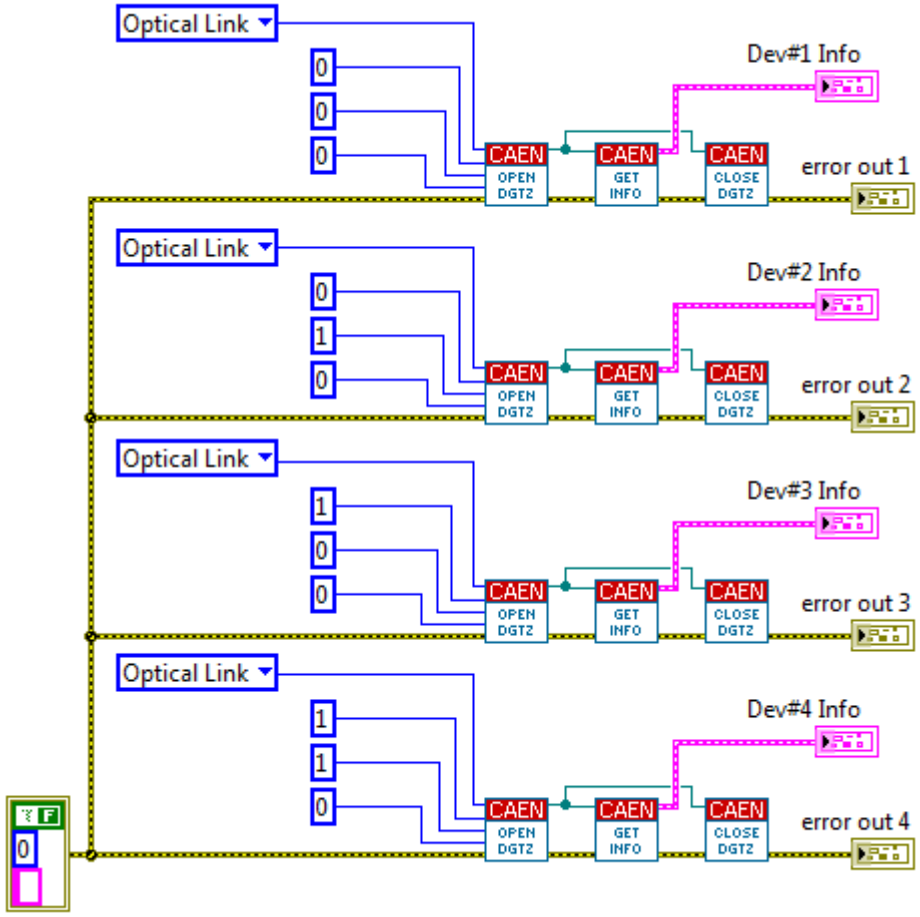


Fig. 8.3: Connection example no.3.

The host PC houses two CAEN A2818 PCI CONET Controllers; the VME crate houses the following boards:

- Two V1724 Digitizer connected in a Daisy chain between them end to the A2818 #0: Dev#1 (first in Daisy chain) and Dev#2 (second in Daisy chain)
- Two V1724 Digitizer connected in a Daisy chain between them end to the A2818 #1: Dev#3 (first in Daisy chain) and Dev#4 (second in Daisy chain)

Note: The A2818 number refers to the PCI slot and depends on the motherboard of the PC used. **It is not known which PCI card is assigned to which number a priori.** In this example we assume that the A2818 connected to Dev#1 and Dev#2, is inserted into the first PCI slot and get Link Number = 0.



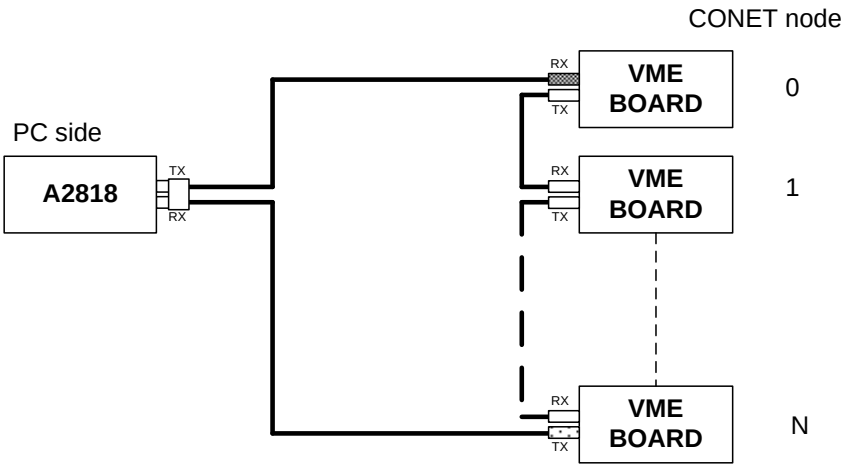


Fig. 8.4: A2818 network scheme.

Example No.4

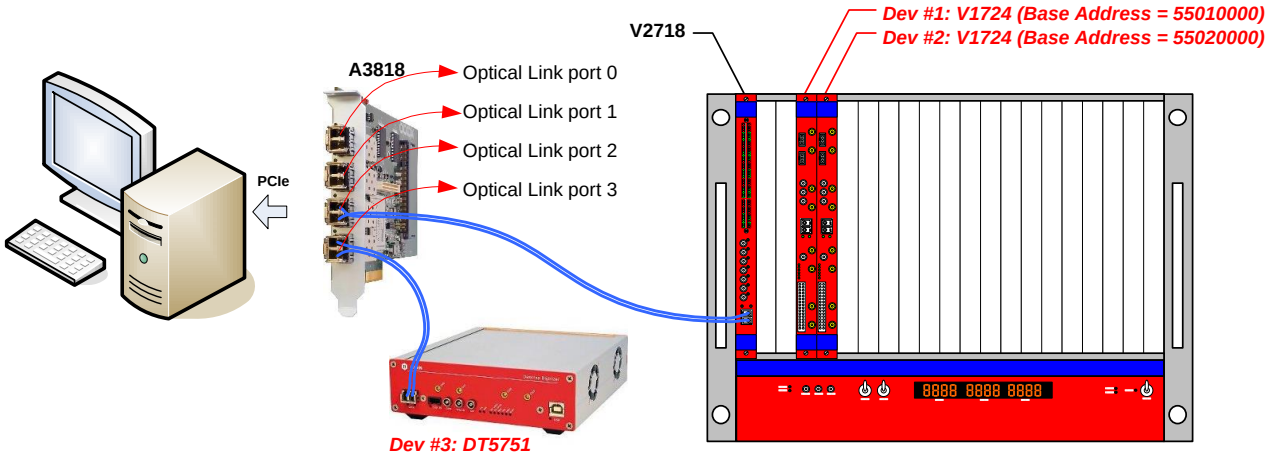
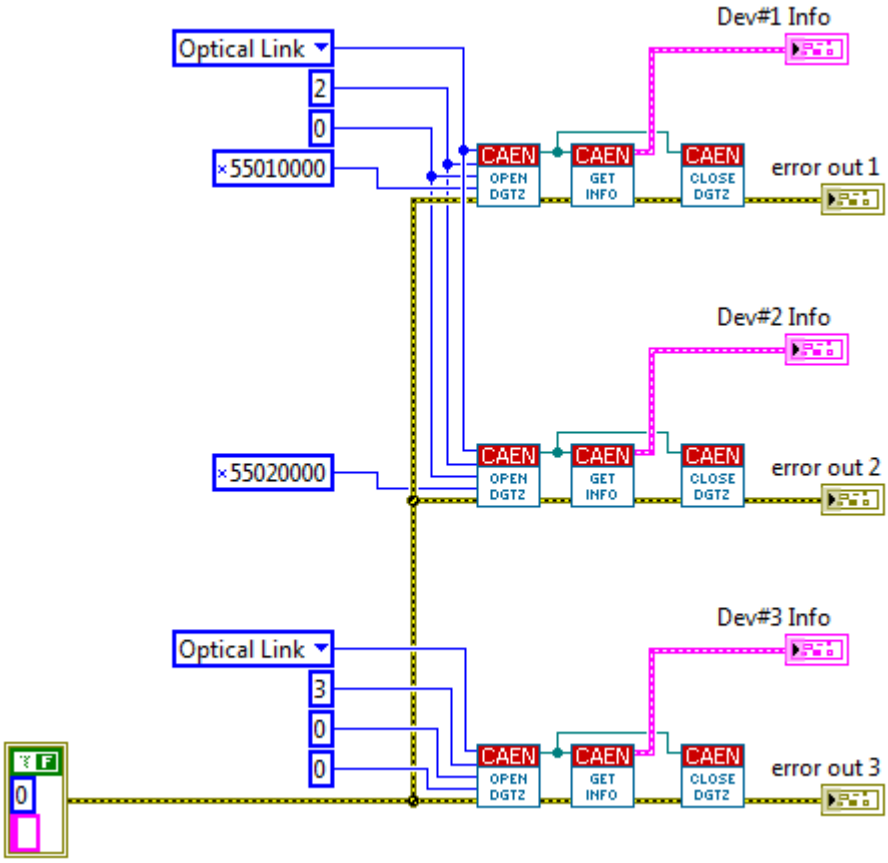


Fig. 8.5: Connection example no.4

The host PC houses one CAEN A3818C PCIe CONET Controller with 4 Optical Link;

- port#3 is connected to Dev#3 (DT5751 - 2/4 Channel 10 bit 2/1 GS/s Digitizer)
- port#2 is connected to a V2718 VME-PCI Optical Link Bridge housed in a VME crate that contains the following boards:
    - Dev#1: V1724 - 8 Channel 14 bit 100 MS/s Digitizer (Base address = 0x55010000)
    - Dev#2: V1724 - 8 Channel 14 bit 100 MS/s Digitizer (Base address = 0x55020000)



Example No.5

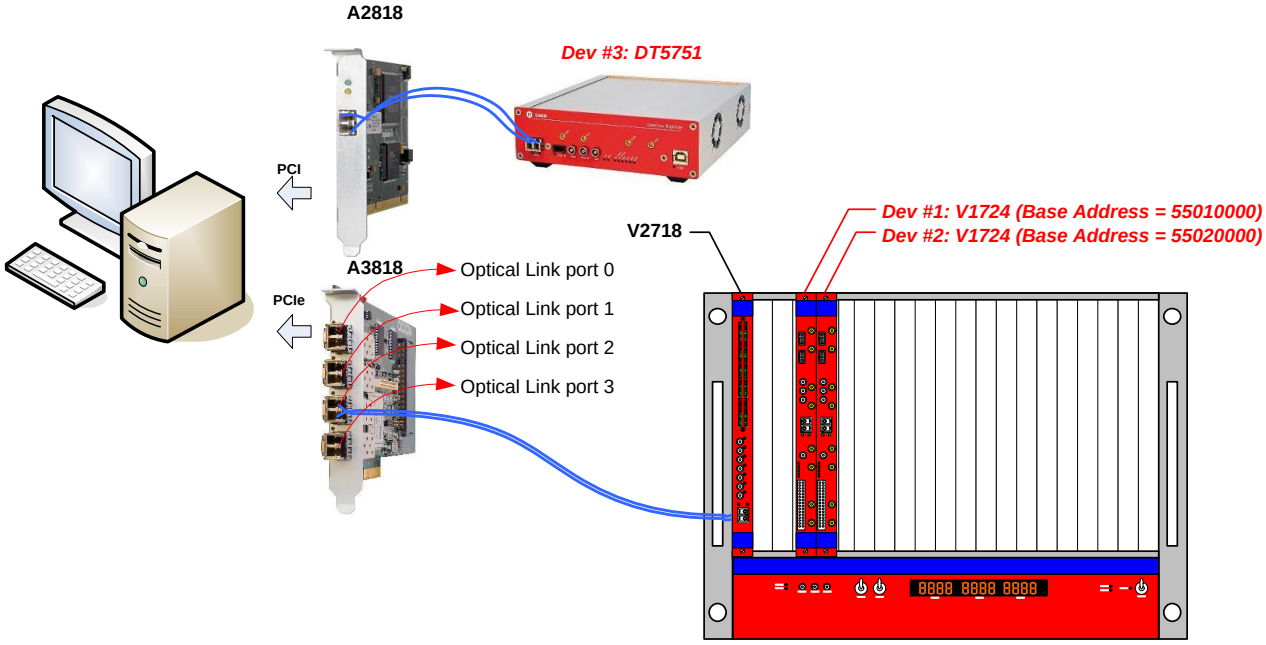
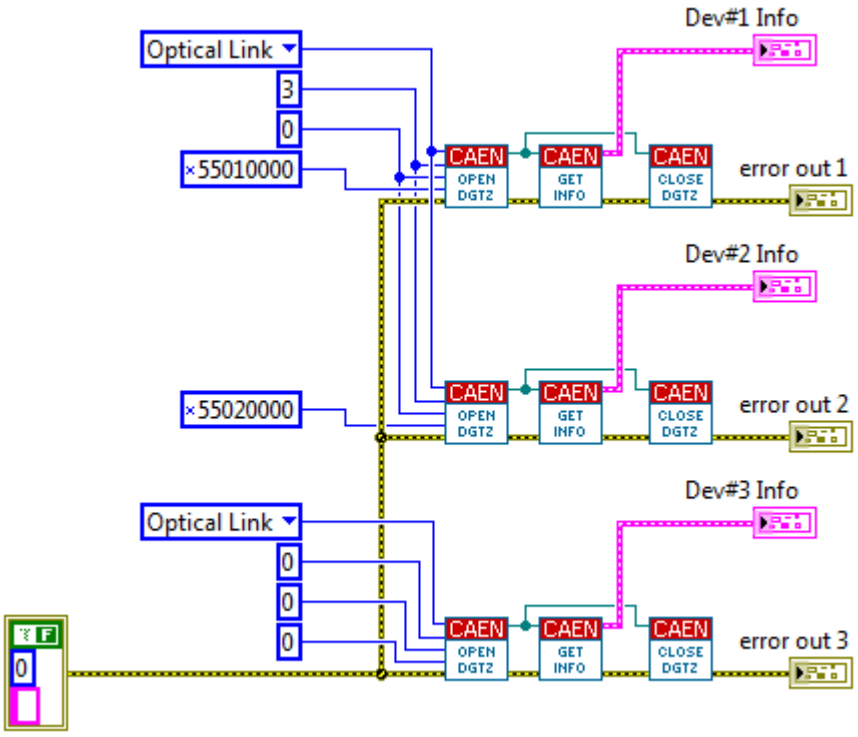


Fig. 8.6: Connection example no.5

The host PC houses

- one A2818 PCI CONET Controller connected to Dev#3 (DT5751 - 2/4 Channel 10 bit 2/1 GS/s Digitizer)
- one CAEN A3818C PCIe CONET Controller with 4 Optical Link; with port#2 connected to a V2718 VME-PCI Optical Link Bridge housed in a VME crate that contains the following boards:
    - Dev#1: V1724 - 8 Channel 14 bit 100 MS/s Digitizer (Base address = 0x55010000)
    - Dev#2: V1724 - 8 Channel 14 bit 100 MS/s Digitizer (Base address = 0x55020000)



9 Technical support

CAEN makes available the technical support of its specialists at the e-mail addresses below:

support.nuclear@caen.it
(for questions about the hardware)

support.computing@caen.it
(for questions about software and libraries)

CAEN SpA is acknowledged as the only company in the world providing a complete range of High/Low Voltage Power Supply systems and Front-End/Data Acquisition modules which meet IEEE Standards for Nuclear and Particle Physics. Extensive Research and Development capabilities have allowed CAEN SpA to play an important, long term role in this field. Our activities have always been at the forefront of technology, thanks to years of intensive collaborations with the most important Research Centres of the world. Our products appeal to a wide range of customers including engineers, scientists and technical professionals who all trust them to help achieve their goals faster and more effectively.

CAEN S.p.A.
Via Vetraia, 11
55049 Viareggio
Italy
Tel. +39.0584.388.398
Fax +39.0584.388.959
info@caen.it
www.caen.it

CAEN GmbH
Klingenstraße 108
D-42651 Solingen
Germany
Tel. +49 (0)212 254 4077
Mobile +49 (0)151 16 548 484
Fax +49 (0)212 25 44079
info@caen-de.com
www.caen-de.com
CAEN GmbH

CAEN Technologies, Inc.
1140 Bay Street - Suite 2 C
Staten Island, NY 10305
USA
Tel. +1.718.981.0401
Fax +1.718.556.9185
info@caentechnologies.com
www.caentechnologies.com

CAEN
Tools for Discovery
Electronic Instrumentation

User Manual UM2784 - CAENDigitizer LabVIEW Library User Manual rev. 1 - 15 January 2015
00118-09-DLAB-MUTX
Copyright © CAEN SpA. All rights reserved. Information in this publication supersedes all earlier versions. Specifications subject to change without notice.