Rev. 0 - May18th, 2023

Sci-Compiler for Digitizers 2.0

Quick Start Guide to write custom firmware for Digitizers 2.0

SCI-COMPILER

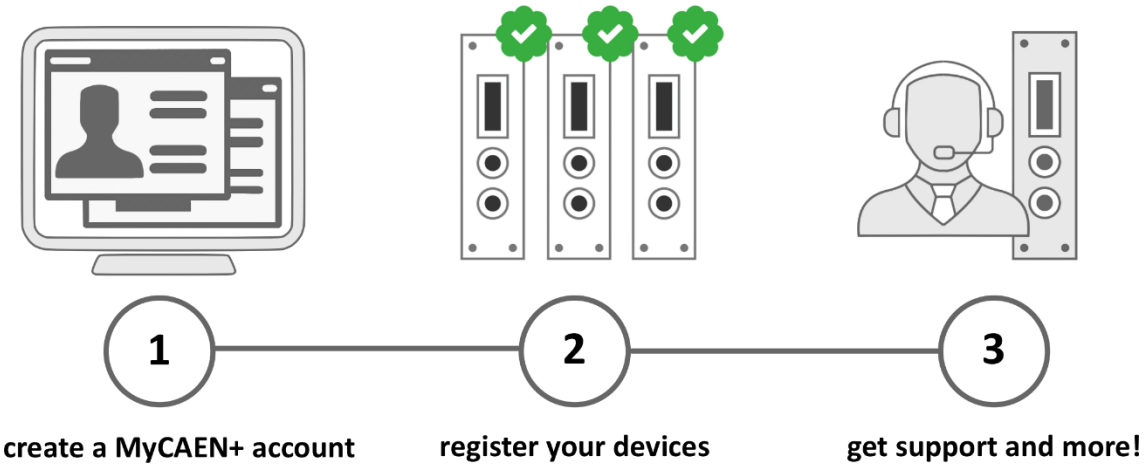
CAEN
Tools for Discovery

User Guide
GD9706

Register your device

Register your device to your **MyCAEN+** account and get access to our customer services, such as notification for new firmware or software upgrade, tracking service procedures or open a ticket for assistance. **MyCAEN+** accounts have a dedicated support service for their registered products. A set of basic information can be shared with the operator, speeding up the troubleshooting process and improving the efficiency of the support interactions.

MyCAEN+ dashboard is designed to offer you a direct access to all our after sales services. Registration is totally free, to create an account go to https://www.caen.it/become-mycaenplus-user and fill the registration form with your data.



https://www.caen.it/become-mycaenplus-user/

Purpose of this Guide

This document is the Sci-Compiler Quick Start Guide. It contains information about the compatibility of the software with CAEN programmable boards, the installation and configuration of the software, its main features and some guided examples of usage

Change Document Record

Date	Revision	Changes
May 18th, 2023	00	Initial release

Symbols, Abbreviated Terms and Notation

FPGA	Field Programmable Gate Array
OTP	One Time Password
IDE	Integrated Development Environment
GUI	Graphical User Interface
HDL	Hardware Description Language

Reference Document

[RD1] UG973 - Vivado Design Suite User Guide (available on Xilinx website)

[RD2] MNL-1065 - Intel FPGA Software Installation and Licensing (available on Altera website)

[RD3] UM5175 - V2495/VX2495 User Manual

[RD4] UM6506 – DT5550 User Manual

[RD5] 00106/03:V1718.MUTx/09 – Mod. V1718/VX1718 Manual

[RD6] 00106/03:A2818.MUTx/01 – Mod. A2818 PCI Optical Link

[RD7] 00102/08:A3818.MUTx/10 – Mod. A3818 PCI Express Optical Link

[RD8] UG908 - Vivado Design Suite User Guide, Programming and Debugging (available on Xilinx website)

[RD9] UG-USB81204 - Intel FPGA USB Download Cable User Guide

[RD10] UM6952 – R5560 User Manual

[RD11] UM7970 – DT5560SE User Manual

[RD12] UM8412 – R5560SE User Manual

[RD13] UM8717 – 2740/45 User Manual

[RD14] GD8712 – Sci-Compiler SMART kit Quick Start

[RD15] https://www.xilinx.com/products/intellectual-property/axi-amm-bridge.html#overview

[RD16] https://www.caen.it/products/caen-wavedump2/

[RD17] https://www.caen.it/products/compass/

[RD18] GD6520 – Sci-Compiler Quick Start Guide

Manufacturer contact

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

Limitation of Responsibility

If the warnings contained in this manual are not followed, CAEN will not be responsible for damage caused by improper use of the device. The manufacturer declines all responsibility for damage resulting from failure to comply with the instructions for use of the product. The equipment must be used as described in the user manual, with particular regard to the intended use, using only accessories as specified by the manufacturer. No modification or repair can be performed.

Disclaimer

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.

Made in Italy

We remark that all our boards have been designed and assembled in Italy. In a challenging environment where a competitive edge is often obtained at the cost of lower wages and declining working conditions, we proudly acknowledge that all those who participated in the production and distribution process of our devices were reasonably paid and worked in a safe environment (this is true for the boards marked "MADE IN ITALY", while we cannot guarantee for third-party manufactures).

Table of Contents

List of Figures

List of Tables

1 Introduction

Overview

The increasing use of programmable logic devices in trigger and data acquisition systems makes clear that having a general-purpose platform and technicians dedicated to the firmware development is becoming more and more important. The advantage of employing programmable logic devices with respect to standard logic modules (like NIM logic modules) is remarkable: a single programmable logic device includes the potentiality of hundreds of thousands of standard logic modules. However, the use of specific programming languages like VHDL or Verilog for the firmware development could represent a limitation in the spread of these powerful devices.

We introduce an innovative method to **simplify the firmware development** on **CAEN Open FPGA boards**. This method is based on a **block-diagram-based** software, called **Sci-Compiler**, that uses a set of high-level functionalities (blocks) to mask the real firmware coding. In this way, it is possible to obtain a VHDL firmware code starting from a rather simple block diagram structure.

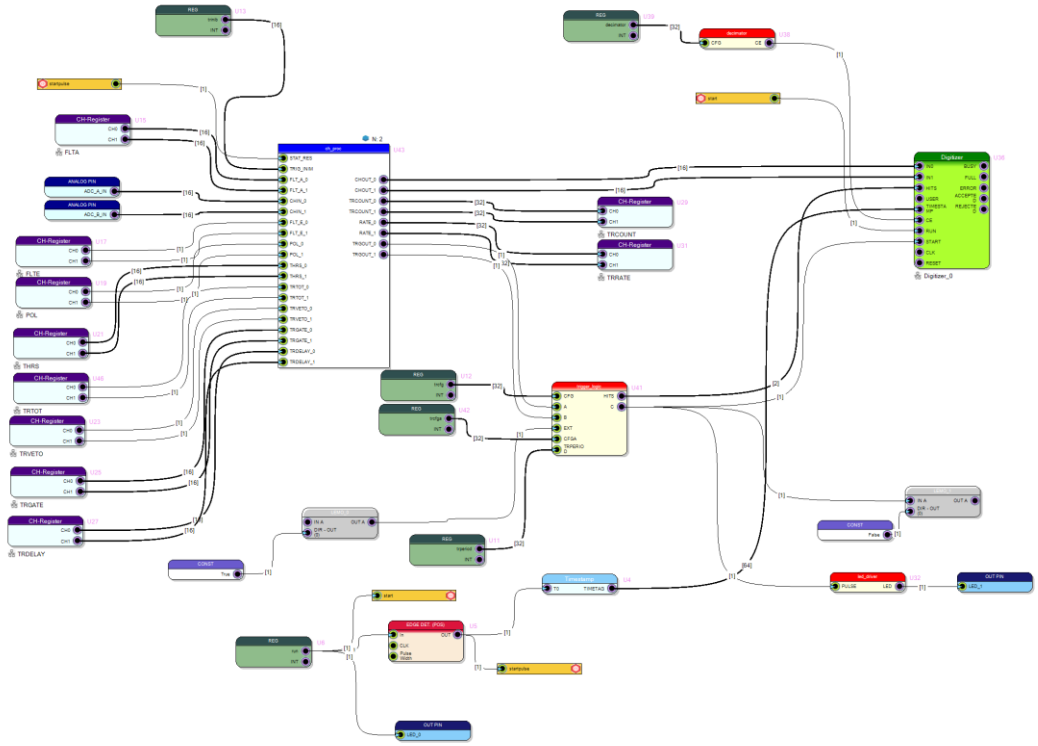


Figure 1.1: example of a block diagram in Sci-Compiler

Sci-Compiler improves and speeds-up the R&D phase. Placing and interconnecting the available blocks (e.g. oscilloscope, TDC, MCA, charge integration, etc.) on a diagram, Sci-Compiler is able to **automatically generate a VHDL piece of code** that implements the required function and deploy it to the FPGA. In this way, even a non-expert user can write his own firmware code, focusing on the functional blocks of the application to be implemented more than on the VHDL/Verilog code to achieve the goal.

Moreover, Sci-Compiler installs a **Software Development kit** (Sci-SDK) compatible with the custom firmware for any supported board. The SDK is made of drivers, libraries and example codes in C++ and Python, so that it is straightforward to build a custom DAQ software that can run on Windows or Linux.

The Sci-Compiler software allows to develop both purely **digital applications**, exploiting blocks like logic gates, scaler, state machine, and **analog processing applications**, such as custom Multichannel Analyzers, charge integration, Pulse Shape Discrimination, Waveform recording with custom trigger logic, and many others.

Sci-Compiler is able to **automatically** generate the VHDL firmware code implementing the functions requested in the block diagram. The compilation is highly accessible thanks to the optional **Remote Customization Service**, which allows you to compile the firmware even without having a full FPGA compiler software running locally on your PC, thus simplifying a lot the software setup. Using the Remote Customization Service, Sci-Compiler will take care of adding to your project the hard-coded parts that remain fixed in each firmware and will makes the final firmware file into the Remote Center and in the MyCAEN+ area as well, ready to be deployed onboard.

CAEN makes the **Remote Customization Service** and **Yearly Upgrade Plan** available, to offer a high quality software product, with the needed flexibility to meet the users' requirements.

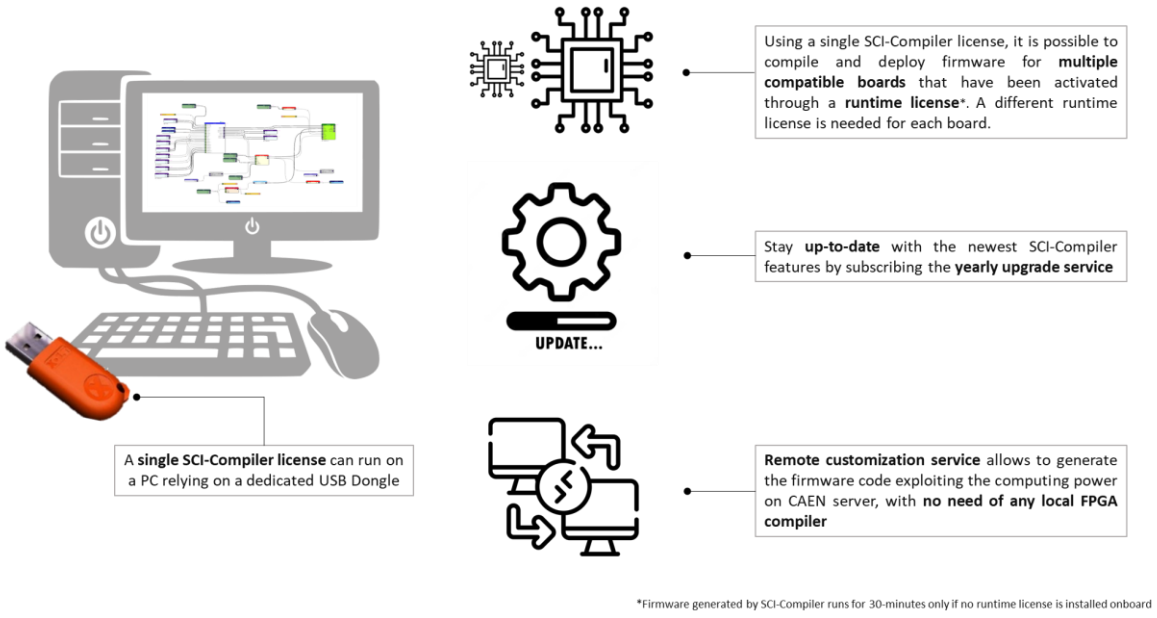
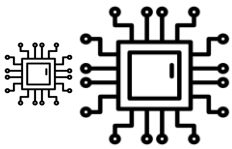
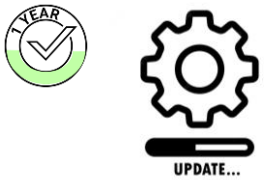


Figure 1.2: Summary of Sci-Compiler operative scenario, with description of the additional services available.

What is included in my license?

	Description
	Sci-Compiler USB Dongle is a **lifetime-working** license. Its expiration date (1 year after activation by default) refers to the possibility to access optional services (see below *Yearly Upgrade Service* and *Remote Customization Service*) E.g.: if your license expires *2024, November 30th*, it means that you can access software upgrades up to that date. If you do not renew your license, you won't be able to download latest software releases after that date. You will still be able to work lifetime with any Sci-Compiler version released before *2024, November 30th*
	The Sci-Compiler PRO USB Dongle supports compilation for any compatible hardware available at the time of purchase.
	The *Yearly Upgrade Service* gives access to the latest software releases, including new features and enhancements. **One year** of free upgrades is included by default at the time of purchase of the USB Dongle and can be renewed at the USB Dongle expiration date.

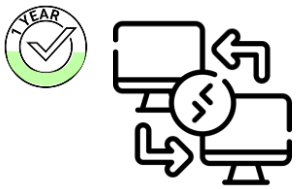
	The *Remote Customization Service* allows the user to compile the firmware even without having a full FPGA compiler software running locally on the PC, thus simplifying the software setup. **One year** of service is included by default at the time of purchase of the USB Dongle and can be renewed at the USB Dongle expiration date.

Table 1.1: description of USB Dongle factory status when you receive it.

Ordering Options

Product	Description	Product Code
Sci-Compiler PRO License	SW555 - Sci-Compiler User Firmware Generator	WSW555XAAAAA
Runtime License	Sci-Compiler runtime license for Digitizers 2.0	WSW55RUNTIME
Remote Customization Service	1-year remote customization service + upgrade	WSW55RCSXAAA
Remote Customization Service – 5-year package	5 years remote customization service + upgrade	WSW55RCSX5YA

Table 1.1: Table of ordering options

2 Requirements & Installation

System Requirements

The Sci-Compiler is compliant with Windows 10/11 OS. The software requires Microsoft .NET 4.0 or higher. If the framework is not available on your PC, it can be downloaded from Microsoft© website.

Note: Remote Customization Service is needed to compile firmware for Digitizers 2.0

To exploit the features of FPGA programming and firmware testing, the needed drivers should be installed (refer to the instructions on the correspondent User Manual).

OS	Windows Framework	Supported Boards	Local option (*)		Remote Customization Service	
			Compilation	Simulation	Compilation	Simulation
10/11	4.0 or higher	V2740 VX2740 DT2740 V2745 VX2745 DT2745	N/A	2020.2– Webpack	✓	COMING SOON

(*) Third-party software not required if using remote customization service

Table 2.1: Sci-Compiler system requirements.

Sci-Compiler License Activation

Sci-Compiler full version works upon a license and a physical USB Dongle to be plugged in the PC during software usage. The license allows to use Sci-Compiler with no time limit, while the software upgrade is limited to a certain software release which is indicated while running Sci-Compiler. All the software releases, up to the last available for your license, are downloadable from MyCAEN+ area.

Note: in order to extend your license for further upgrade, you should enable the Yearly Upgrade Service. Contact CAEN for more information

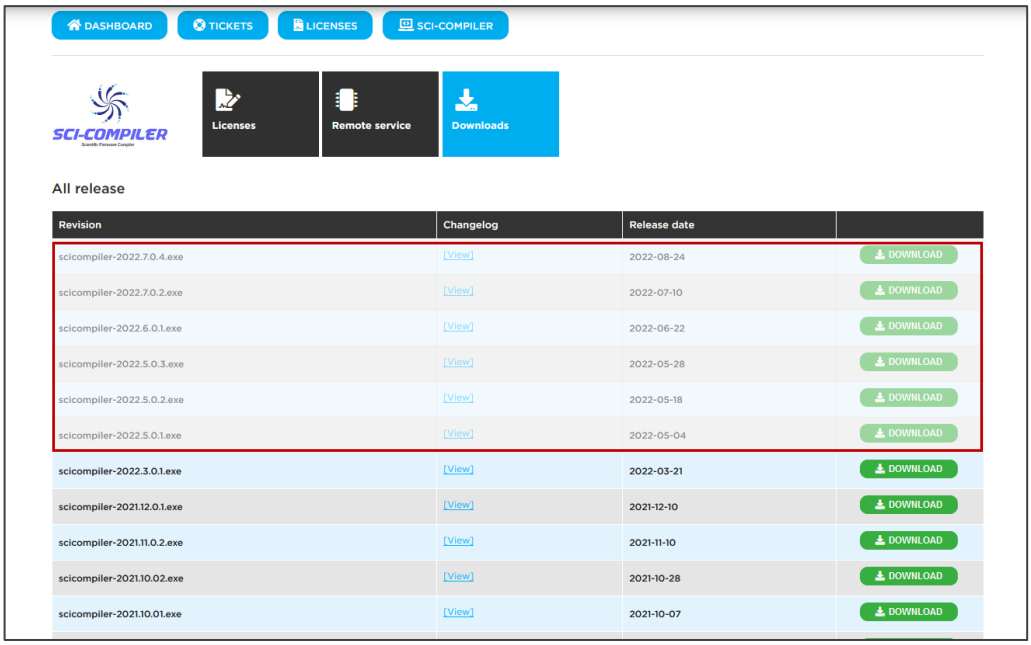
Note: a 30-days free trial version is available. Remote Customization Service is not included in the 30-days free trial. You can request your 30-days free trial registering into the MyCAEN+ area and following the instructions given in the *Sci-Compiler* tab. You can request the 30-days free trial just once.

The software license can be activated at https://www.caen.it/mycaen/Sci-compiler/ using the DONGLE SERIAL and an OTP key provided by CAEN together with the USB Dongle (see Errore. L'origine riferimento non è stata trovata.)

Refer to **[RD18]** for the exact activation step-by-step procedure.

Sci-Compiler Installation

After license first activation or import, it is possible to access the *Downloads* list, where all the available Sci-Compiler releases are listed.



Note: the *Downloads* tab shows all the available software releases but gives access only to the ones compatible with your license expiration date.

Run the latest available .exe setup file and perform the needed installation steps.

Refer to **[RD18]** for the exact installation step-by-step procedure.

Sci-Compiler Welcome Wizard

At first Sci-Compiler start-up, a Wizard will guide the user through the software configuration. You will need to login with your MyCAEN+ credentials and follow the steps suggested by the Wizard.

Note: we strongly encourage you to use the online activation, maintaining the MyCAEN+ account active. If an offline activation is needed, please refer to **[RD18]** for an alternative procedure.

Note: you should leave the Remote Customization Service active to allow compilation for Digitizers 2.0

3 Digitizers 2.0 architecture

Digitizer2.0 hosts a single, large MPSoC device from Xilinx AMD device. The MPSoC is composed by a programmable logic section (FPGA or PL) and an embedded quad core ARM processor (PS). The device and its characteristics are depicted in the following table:

Device	Xilinx Zynq UltraScale+ XCZU19EG
System Logic Cells (PL)	1,143K
Total RAM (PL)	80.4 Mb
DSP Slices	1968
Application Processing Unit (PS)	Quad-core Arm® Cortex®-A53 MPCore™

Table 3.1: general information about the SoC mounted on Digitizers 2.0

The CAEN **Open FPGA** architecture allows to write digital algorithms for online waveform data processing and to implement custom trigger and coincidence logic. The general architecture is depicted in the following figure:

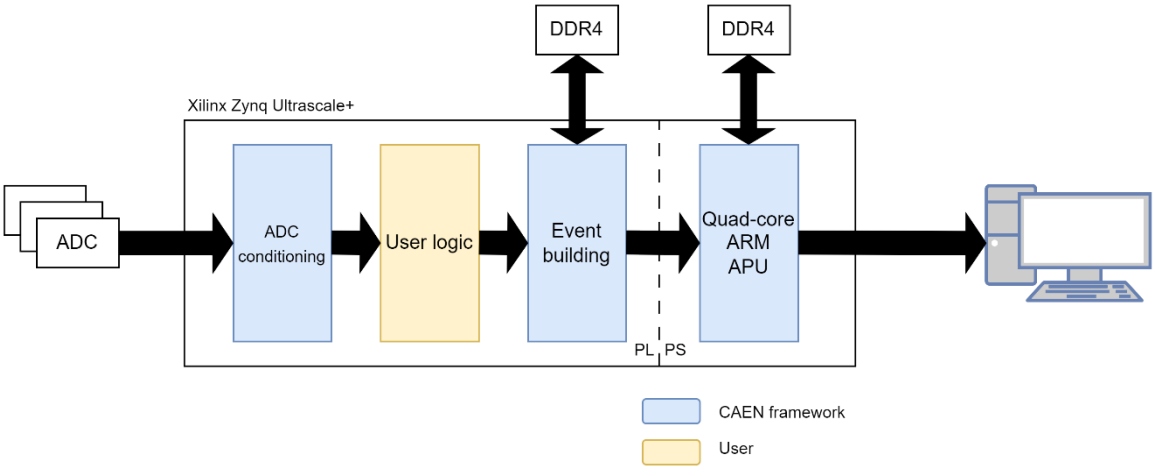


Figure 3.1: Open FPGA architecture of Digitizers 2.0

Custom logic can be integrated into the CAEN framework. Custom logic can process input data and generate triggered data (events). No access is given to some critical parts of the design, such as DDR memory or to the readout interface and protocol.

The custom logic has access to:

- ADC raw data
- Control signals from the framework (synchronization)
- Framework triggers
- Other framework status and synchronization signals

Note: Sci-Compiler is used to define the custom logic part

The CAEN framework instead, is an encrypted IP that manages:

- ADC data pre-processing for baseline calibration and adjustment
- Processing of user elaborated data
- Trigger management
- Synchronization
- Data buffering into external DDR4 memory

- Communication setup and management (USB 3.1, 1/10 Gb Ethernet)
- External I/O management

Note: Remote Customization Service is used to integrate the custom logic defined within Sci-Compiler and the CAEN framework, to ensure the correct operation of the final generated firmware (CUP file) that can be loaded onto the target board.

The Sci-Compiler-based development flow is shown in the following figure:

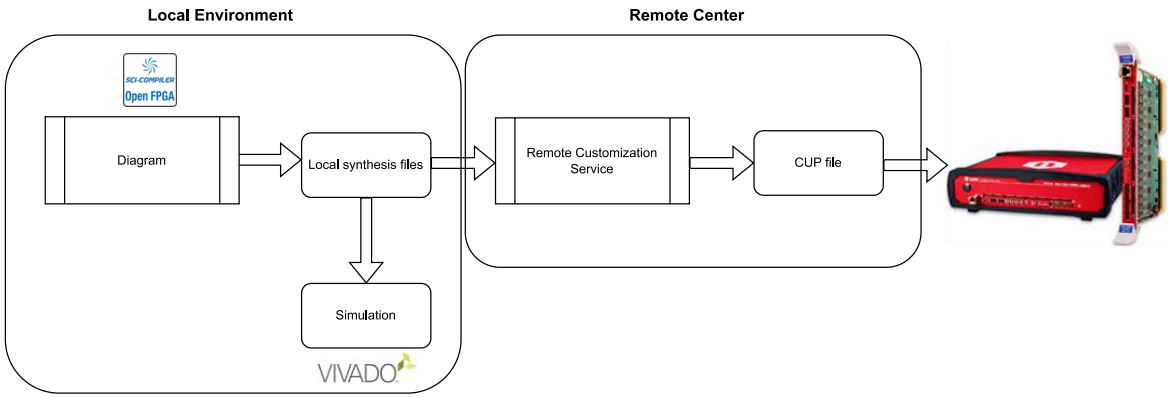


Figure 3.2: general workflow of firmware synthesis and compilation for Digitizers 2.0.

4 How to choose the right project template

Sci-Compiler provides two different templates to create a project for Digitizers 2.0:

- **Scope Mode**
- **DPP Mode**

Scope Mode is designed for raw waveform acquisition, simultaneously on the 64 channels with a common trigger.

The **DPP Mode** is designed to use algorithms to identify input pulses and extract parameters such as arrival time with sub-nanosecond resolution, pulse height or charge, pulse shape, and other relevant quantities. In this mode, the digitizer functions more like a Multi-Channel Analyzer (MCA) and provides spectra and lists rather than waveform recordings. However, the raw waveforms can still be saved for post-processing, signal inspection, and debugging. Unlike Scope mode, the DPP Mode can work with independent triggers, and channels are self-triggered without the need for an external global trigger.

Scope and DPP modes will be described in more details in the following paragraphs.

Scope Mode for Digitizers 2.0

The Scope mode allows to implement variation of a default oscilloscope acquisition mode.

The FPGA internal architecture in Scope mode is depicted in **Figure 4.1**.

In Scope mode, the CAEN framework logic implements an oscilloscope-like acquisition path (red path): all channels of the board are acquired simultaneously for a programmed number of samples around a single trigger signal. The trigger source can be selected via Sci-Compiler. The trigger source is enforced to be common to all available channels.
Since all channels are recorded at the same time, the large external memory available on-board can be fully allocated to store each event. A very large record length can be achieved. The maximum supported record length per channel is about 10M samples. A custom logic can be developed with Sci-Compiler and inserted into the overall design by implementing the yellow block in figure (User Logic).

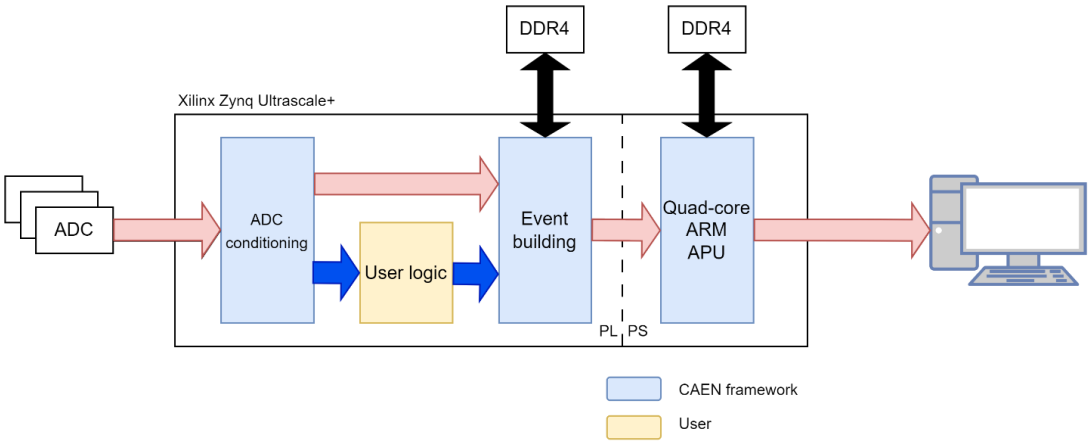


Figure 4.1: Open FPGA architecture in Scope mode.

DPP Mode for Digitizers 2.0

The DPP mode allows to implement online waveform processing with dedicated algorithms aiming to extract the parameters of interest from the input pulses, such as the pulse height, the gated integral (charge), the high-resolution timing (by sample interpolation), the pulse shape discrimination and others.

The FPGA internal architecture in DPP mode is depicted in **Figure 4.2.**

Unlike Scope mode, the output data streams can contain both raw samples and/or extracted parameters. Each channel is completely independent from the others, so one channel can produce only parameters while others the waveforms and others a mix of the two as well. The channel independence feature makes the DPP firmware a suitable solution also for the applications requiring the acquisition of raw waveforms (no need of online processing) that cannot run with the Scope firmware because of the common trigger.

Each channel can generate a local trigger and save the data into its own data queue. The board has a dedicated large external memory to store events. Since each channel contribute with its parameters and samples, the common external memory cannot be allocated for access to sparse access of every single channel, but an aggregate of channel events is collected and stored in external memory for buffering.

Unlike Scope mode, there is not a default acquisition implementation. So, an empty user logic will not produce a working implementation, in the sense that there will not be any data available for readout.

Channel events are time-ordered by the CAEN framework. Time reordering offline in software is not required.

A custom logic can be developed in Sci-Compiler and inserted into the overall design by implementing the yellow block in figure (User Logic).

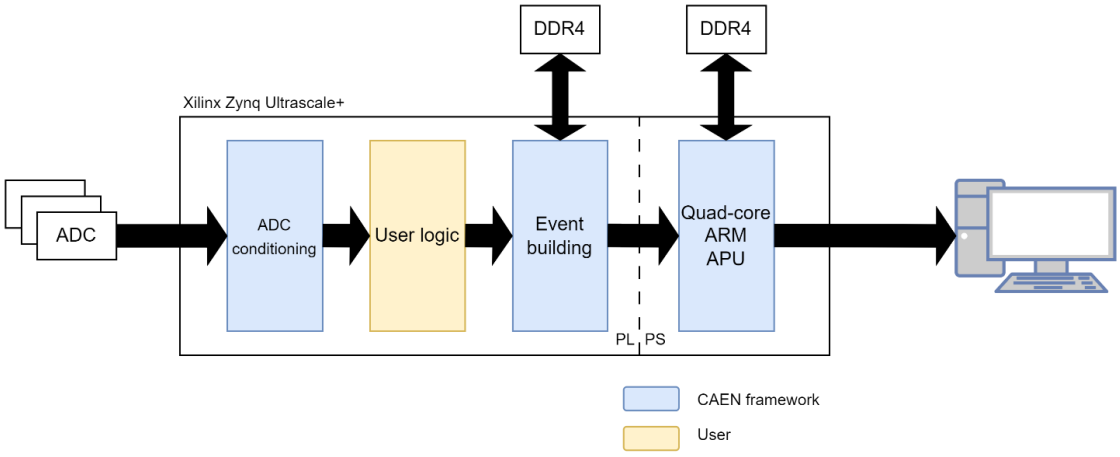


Figure 4.2: Open FPGA architecture in DPP mode.

How to choose between Scope and DPP mode project templates

Which project is right for you? Scope or DPP?

Scope Mode is mainly intended for raw waveform acquisition, simultaneously on the 64 channels with a common trigger, and **DPP Mode**, where DPP stands for Digital Pulse Processing, which is a set of algorithms aiming to detect the input pulses and extract the relevant parameters such as the arrival time with sub ns resolution, the pulse height and/or charge, the pulse shape, and other quantities of interest. In DPP mode, the digitizer looks more similar to an MCA (Multi Channel Analyzer), providing spectra and lists, rather than an oscilloscope or a waveform recorder. Nevertheless, the DPP mode keeps the possibility to save raw waveforms for post processing, signal inspection and debugging. Unlike the Scope mode, the DPP can work with independent triggers: typically, the channels are self-triggered and there is no need of an external global trigger. The DPP mode is therefore suitable for streaming readout. Although less frequent, there are cases where the DPP algorithms for the extraction of the pulse parameters are used in combination with a global trigger, whose purpose is to validate the self-triggers and build the event data packet simultaneously for all channels.

These are two main differences between Scope and DPP firmware:

- The DPP implements online waveform processing with dedicated algorithms aiming to extract the parameters of interest from the input pulses, such as the pulse height, the gated integral (charge), the high-resolution timing (by sample interpolation), the pulse shape discrimination and others. The output data can be just the extracted parameters, saved into a small data packet (typ. 8 or 16 bytes), thus making it possible to sustain an extremely high trigger rate, or the combination of parameters and raw waveforms when these are necessary for further offline analysis. It is also possible to mix events with and without attached waveforms in the same run; in particular, the User can define criteria to decide whether the pulse parameters extracted online are not

reliable and, in that case, save the full waveform and let the software to apply more complex extraction algorithms. A typical example is the case where piled-up pulses cannot be deconvoluted and it is not possible to get the correct energy calculation in a simple way.

- The DPP makes the channels independent: unlike the Scope firmware, where the acquisition trigger is common to the 64 channels, in the DPP firmware each channel can generate a local trigger and save data into the local output FIFO, independently by the other channels. The data packets coming from the channels are sorted and merged by a dedicated block in the CAEN framework, before feeding the DDR memory. The independence of the channels makes the DPP firmware a suitable solution also for the applications requiring the acquisition of raw waveforms (no need of online processing) that cannot run with the Scope firmware because of the common trigger.

The DPP mode is more flexible than the Scope mode, also in terms of user accessibility, and for this reason it is more appealing for the users that want to develop their custom firmware.

The following table summarize the difference between the two possible development kits:

Aspect	**Scope**	**DPP**
Application	Waveform processing. Offline analysis.	Algorithms for pulse processing on a channel-by-channel basis. Offline analysis possible on small waveform fragments around trigger
Triggering scheme	Common trigger for all channels	Independent trigger: each channel can trigger based on local criteria. API support for global trigger.
Triggering rate	Depends on record length.	Typically, very large. When no waveforms are included in events, events are very small (minimum size is 8 bytes)
Data flow	All channels contribute data at the same time for the same record length. Circular buffer. Events generate on global trigger.	Channels generate events independently
Waveform fragments	Maximum 10M samples	Maximum 8K samples per channel

Table 4.1: main differences between Scope and DPP projects in Sci-Compiler for Digitizers 2.0

5 Building the project in Sci-Compiler

Sci-Compiler uses block diagrams to convert the concept of a Digital Pulse Processing algorithms into a full working firmware. Typically, a project is made of three main parts:

- **Inputs**, that are blocks mapping the actual hardware input of the hardware.
- **Processing,** i.e., blocks interconnecting each other and making the Realtime Signal Processing. For instance, in **Figure 5.1,** the signal from the input feeds a *Derivative Trigger*, which in turn activates a *Trapezoidal Filter*. The flat top of the trapezoid is sampled by the *Energy Sampler* block and energy of the incoming signal is then measured.

Note: refer to the software *online Help* for the accurate description of the Inputs and Processing blocks available for each compatible board.

- **Readout**, i.e., blocks to read the results of the Processing and transfer them to the PC as data. For instance, in **Figure 5.1,** a *Spectrum* block is used to read data coming from the *Energy Sampler.* Data are read by the *Spectrum* block when the *Energy Sampler* and an external digital signal on pin IO_A_0_A of the board are in coincidence (*Logic AND*).

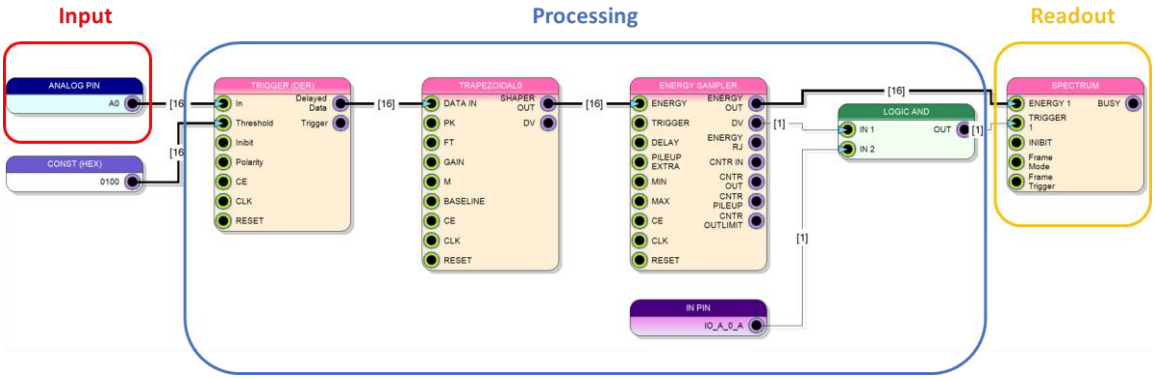


Figure 5.1: the typical structure of a Sci-Compiler project for Digital Pulse Processing.

How to choose the readout blocks

Sci-Compiler makes available different readout blocks for Digitizers 2.0. It is possible to use local bus blocks (like Oscilloscope, Spectrum, Custom packets) or special blocks taking advantage of the DMA/DDR memory available for Digitizers 2.0.

Note: to identify the Local Bus and DMA/DDR blocks clearly, they are coloured as stated below:

- Local Bus → light green
- DMA/DDR in Scope Mode → yellow
- DMA/DDR in Wave mode → dark green

Note: DMA/DDR readout block is called ***x2740 Digitizer*** and ***x2740 List*** for Scope and DPP mode respectively. Both blocks can be found under DAQ sub-menu in Sci-Compiler. The two blocks cannot be instantiated in the same project. In fact, the user has to decide at the creation of the project if using the Wave or DPP mode. Wave mode should be used for common trigger acquisitions and DDR4 is mainly used for waveform samples storage. DPP mode should be used for independent channel trigger acquisitions, where a complex Signal Processing is needed to extract a list file. In DPP mode, the DDR4 is feed with the list coming from the *CAEN List* block.

The Local Bus and DMA/DDR strategies (see Errore. L'origine riferimento non è stata trovata.) allows for a different integration with software tools:

- **Local Bus** readout (available for all compatible boards): data from the processing blocks and configuration registers are sent through a AXI Local Bus Bridge (refer to **[RD15]**) and can be directly read by the **Resource Explorer**, that is a built-in tool available in Sci-Compiler GUI (see **Figure** 5.2). The Resource Explorer gives the possibility to read and write the configuration registers (i.e. Signal Processing parameters) and shows the readout instruments like *Spectrum*, *Oscilloscope*, etc. in a GUI , so that it is possible to change the signal

processing parameters and test the results of acquisition live, with no need to write any software code. In practice, the Resource Explorer is a first-use debug interface to test the firmware and check the behaviour of the Processing algorithm specified through the block diagram.

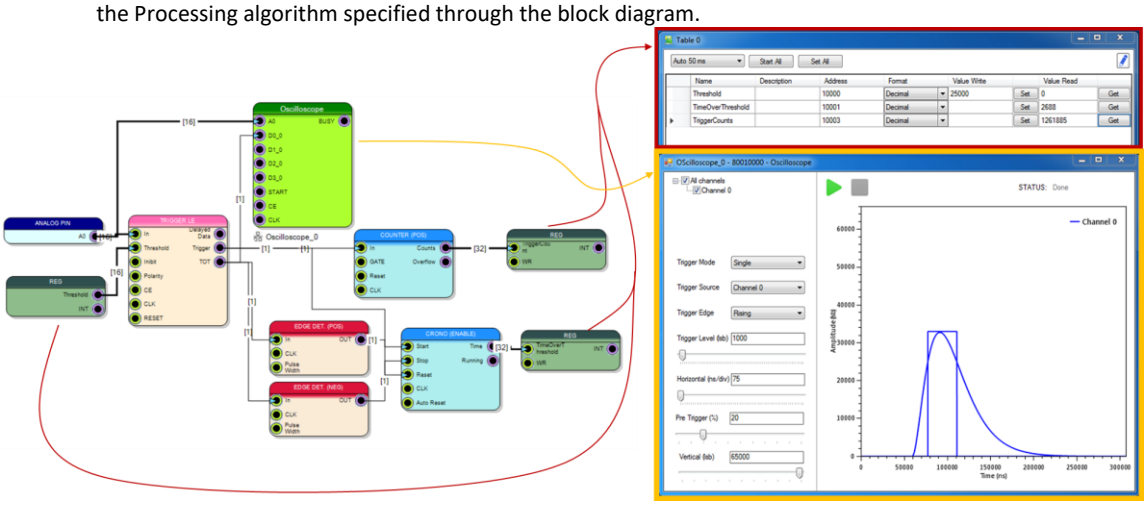


Figure 5.2: the Sci-Compiler Resource Explorer (right), showing an oscilloscope acquisition and Digital Pulse Processing parameters for a given firmware block diagram. The Read/Write registers are reported in the table, the signal acquired by the *Oscilloscope* block are shown in a user-friendly GUI.

- **DMA/DDR** readout: special readout blocks are available for boards supporting DMA/DDR architecture. In this case, the acquired events flow into a large DDR4 memory (5 GB) for buffering, waiting for a DMA that transfers the data into the DDR4 memory of the ARM processor. Finally, upon request of the DAQ software running in the host PC, the data are transferred to the computer and can be readout through the CAEN software WaveDump2 **[RD16]** and CoMPASS **[RD17]**, or alternatively using the Sci-Compiler Software Development Kit (SciSDK).

Local Bus and DMA/DDR Readout blocks can be used together in the same project. For instance, it is possible to readout data from the same *Charge Integration* block using both the *Custom Packet* (feeding the Local Bus) and the *CAEN LIST* (feeding the DMA/DDR path), as shown in **Figure 5.3**. In particular, data coming from the *Custom Packet* or from the *CAEN LIST* would be the same but in different formats. With respect to the Local Bus blocks, the Special Readout ones are faster and more performing because they use the DDR4 memory instead of the FPGA RAM.

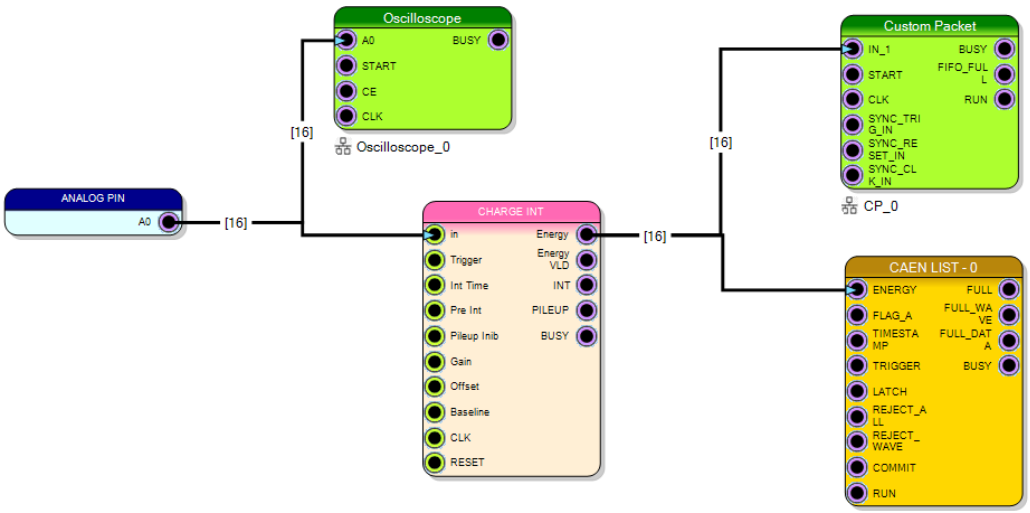


Figure 5.3: example of use of Local Bus blocks (Oscilloscope and Custom Packet) together with x2740 List in the same project.

6 Software tools

As outlined in the previous chapter, there are several ways of reading out a Digitizer 2.0 where a Sci-Compiler custom firmware has been loaded. **Figure 6.1** summarizes the possible ways of reading out the board, depending on the blocks used in the Sci-Compiler diagram. We have three main cases:

- The **Scope block** (only available for scope mode project) maintains the default output data format of the Digitizer 2.0 waveform recording firmware. Therefore, it can be directly red by the standard FELib library and associated WaveDump2 software. Alternatively, it is possible to use the SciSDK library, that is wrapping the FELib, and build a custom C/C++/Python custom software to read out the board and R/W the custom registers placed in the firmware diagram.

- The **DPP block** (only available for DPP mode project) maintains the default output data format of the Digitizer 2.0 DPP firmware. Therefore, it can be directly red by the standard FELib library and associated CoMPASS software. Alternatively, it is possible to use the SciSDK library, that is wrapping the FELib, and build a custom C/C++/Python custom software to read out the board and R/W the custom registers placed in the firmware diagram.

- The Local Bus Readout blocks, like Custom Packet, Oscilloscope, Spectrum, ….. and all the R/W custom registers, can be easily configured and read via the **Resource Explorer** available within the Sci-Compiler. Moreover the SciSDK library can be used to build a custom C/C++/Python custom software to read out the Local Bus endpoints and R/W the custom registers placed in the firmware diagram.

For instance, a Scope mode firmware diagram containing custom registers and the Scope block (DMA/DDR readout) could be easily read via WaveDump2, while configuration could be done through registers R/W in the Resource Explorer.

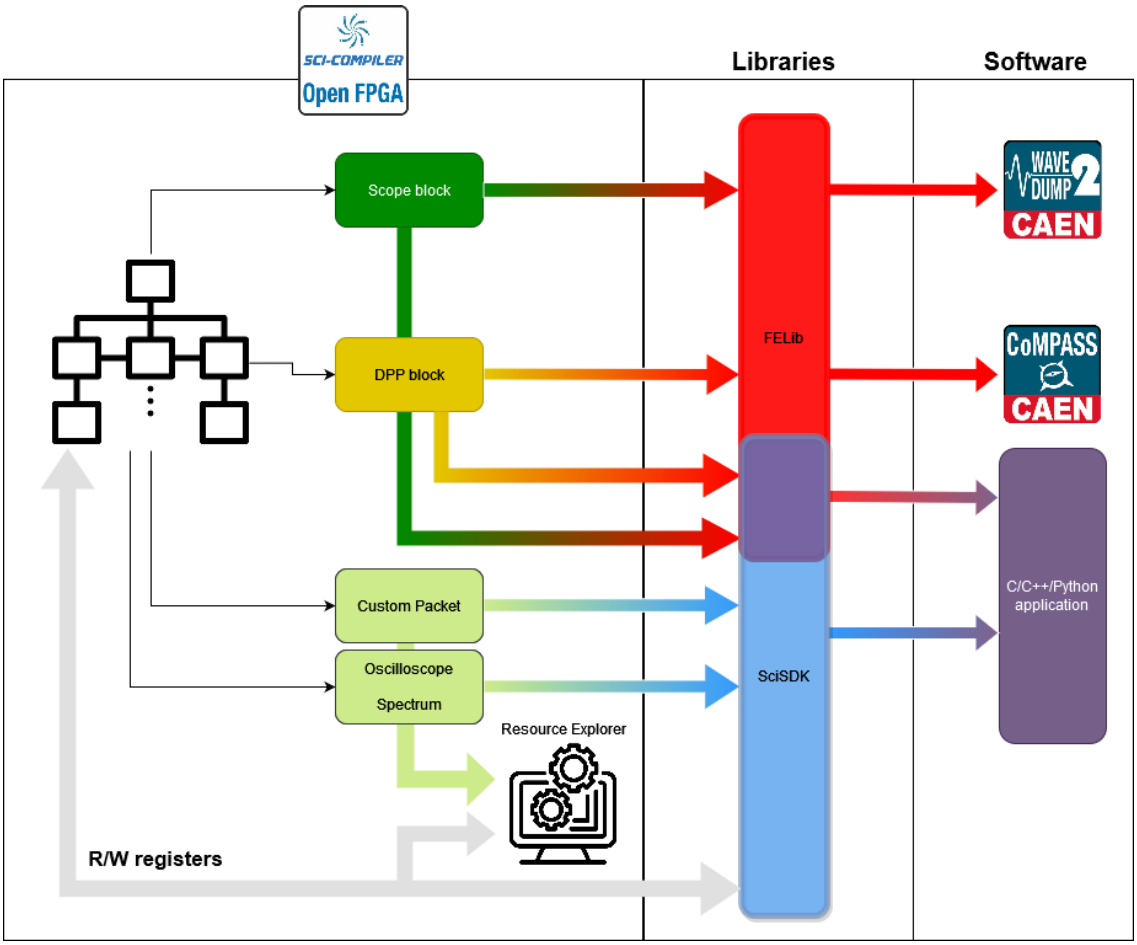


Figure 6.1: readout modes for a firmware generated in Sci-Compiler loaded on a Digitizer 2.0

WaveDump2

WaveDump2 is a C++ software developed upon Qt cross-platform application development framework. An advanced and user-friendly configuration GUI provides all the necessary tools and functionalities for managing any hardware parameter from the basic ones to the most specific ones. The settings can be conveniently stored into or loaded from a configuration file.
This software specifically supports the Scope firmware of the 274x Open FPGA Digitizers for waveform recording applications. Data acquisition from multiple boards and multi-board synchronized systems are managed through a dedicated toolbar.

Refer to https://www.caen.it/products/caen-wavedump2/ for more information.

CoMPASS

CAEN Multi-PArameter Spectroscopy Software (CoMPASS) is a DAQ software able to implement a Multiparametric Data Acquisition for Physics Applications: the detectors can be connected directly to the digitizers/MCAs inputs and the software acquires energy, timing, and PSD spectra at the same time.
CoMPASS software has been designed as a user-friendly interface to manage the acquisition with all the CAEN DPP algorithms. It allows an easy setting of the acquistion parameters and to display up to six different plots and histograms at the same time.

Refer to https://www.caen.it/products/compass/ for more information.

Note: when using CoMPASS to readout a DPP mode firmware designed with Sci-Compiler, you should always take into account that the software can decode the data packet coming from the DPP block, but it cannot know the kind of data you are writing in each available field of the packet. For instance, if you write the PSD parameter in place of the PHA one, CoMPASS will show the PSD values in the plot labelled as PHA.

Resource Explorer

Sci-Compiler offers a simple built-in tool, called *Resource Explorer,* to connect one of the supported boards and test the features of the FPGA firmware with no need to write any software code. This tool allows to manage all **Local Bus** readout blocks placed in the firmware diagram(see **Figure 5.2**), therefore it gives the possibility to read and write the configuration registers (i.e. Signal Processing parameters) and shows the readout instruments like *Spectrum* , *Oscilloscope*, etc. in a GUI , so that it is possible to change the signal processing parameters and test the results of acquisition live, with no need to write any software code. In practice, the Resource Explorer is a first-use debug interface to test the firmware and check the behaviour of the Processing algorithm specified through the block diagram.

Refer to **[RD18]** for more information

SciSDK

The SciSDK is a Software Development Kit installed together with Sci-Compiler setup and compatible with any board supported by Sci-Compiler.

Note: for specific documentation of the functions and parameters of the SciSDK, refer to the Sci-Compiler **online Help** or to **https://nuclearinstruments.github.io/SCISDK/**

Note: alternative installation methods for the SciSDK are given in the online documentation

7 Step-by-step example

In this Chapter we give two examples to be taken as reference for the procedure to create a project in Sci-Compiler and compile the firmware, either in local mode either using the Remote Customization Service. We take some specific boards as examples, but the method can be extended to all the compatible hardware.

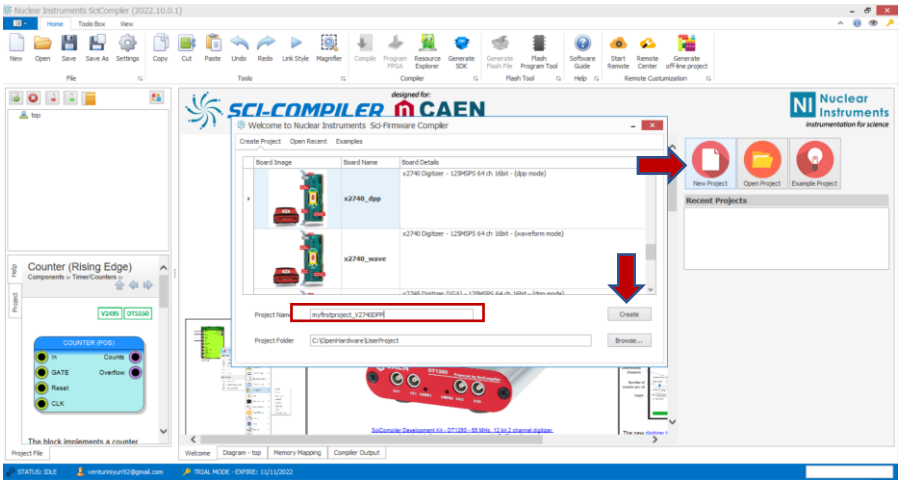
In this step by step example we give instructions to design a custom firmware for 2740/2745 Digitizers family to extract counts and charge of an analog exponential signal. **Since we are interested in quantities extracted by the pulse processing algorithm only and not in the full waveform, we create a DPP project.**

Note: in this Paragraph we give instructions basing on a DPP project V2740 board, but it can be extended to all 2740/45 Digitizer Family models (V2740, VX2740, DT2740, V2745, VX2745, DT2745). Moreover, the general concepts can be extended to DT5550x, x5560 families as well.

1. Activate your Sci-Compiler license and set up the software installation as described in Par. **Sci-Compiler License Activation**, **Sci-Compiler Installation**, **Sci-Compiler Welcome Wizard**.

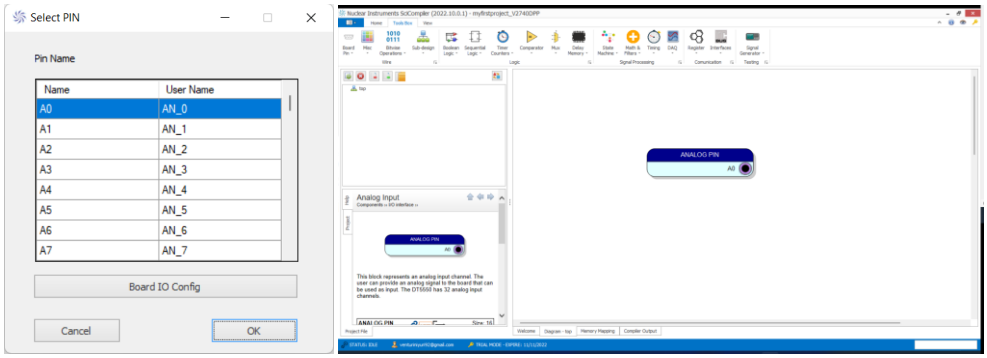
2. Plug the USB Dongle in your PC and launch Sci-Compiler

3. **Create a project** for your V2740 board: press *New Project* to open the selection menu→ Select *x2740_dpp* → Choose a name/folder for your project → Press *Create*.



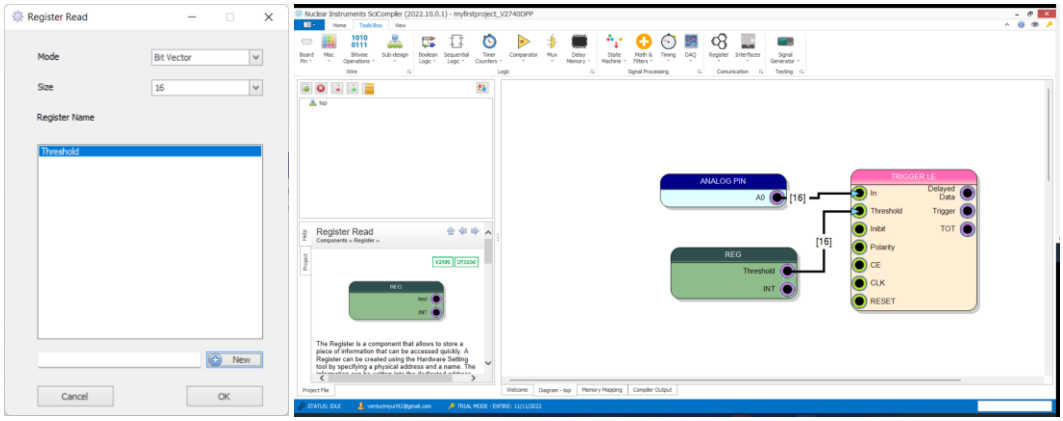
Note: In alternative to creating a project from scratch, Sci-Compiler offers ready-to-use examples that can be opened and modified to speed up the learning curve.

4. First of all, we place an analog input in the diagram. From the Tools Box bar, click Board Pin → Analog In and select, for instance, A0, corresponding to the physical CH0 of the V2740 board.

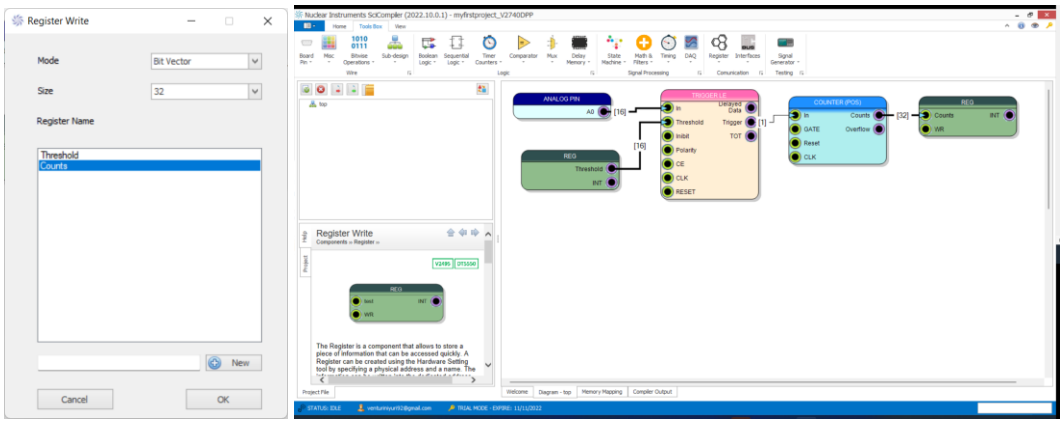


5. In order to count the input signals, we need first to discriminate them. Therefore, we should place in the diagram a Leading Edge Discriminator with a programmable threshold via a register read by the discriminator block. Place the discriminator from **DAQ→Trigger Leading Edge** and the register from **Register→ Register (read)**. In the Register window choose Bit Vector mode, size 16, in order to match the Leading Edge

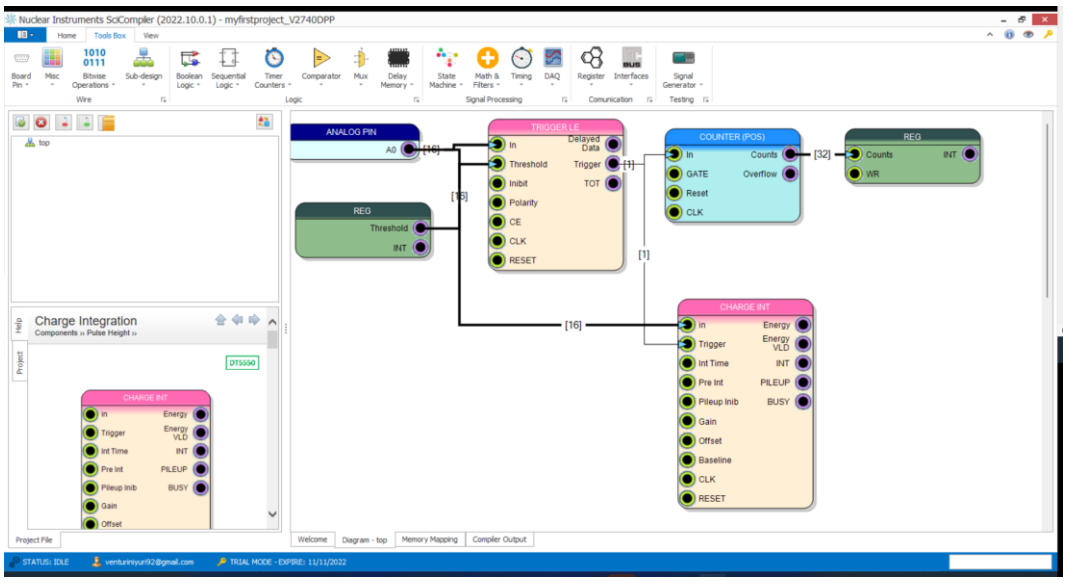
Discriminator block requirements. Give a name to the register (for instance *Threshold*), press New and then select the register name and press OK. Connect the blocks as shown in the figure below.



6. If the input signal has positive polarity, we now have to add a counter triggering on the rising edge of the signal. Place the counter block from **TimerCounter → Counter (Rising Edge)** and connect it as shown in the figure below. To store the counts number in a register written by the Counter (POS) block, add a Register Write with name *Counts,* Bit Vector mode, Size 32 and connect it to the *Counts* output of the counter block.



7. The charge of the signal can be calculated with the Charge Integration block. Place it in the diagram from **DAQ→ Charge integration** and connect it as shown in the figure below, so that the Analog In is the input of the Charge Integration block and the trigger is given by the Leading Edge Discriminator.
For the sake of simplicity, leave the the Charge Integration parameters – Integration Time, Pre-Integration, Pileup and Gain - to their default values (see Online Help of the Charge Integration block)



8. To define the Charge Integration parameters, place in the diagram a *Register→ Register File* block.
9. To transfer energy data in a packet that can be easily handled by the V2740 readout we place the proper block from **DAQ→x2740 CAEN List**. This is one of the Special Readout blocks that allows to transfer data in list mode from the board to the PC using the high-speed DMA endpoint (refer to **Scope Mode for Digitizers 2.0**

The Scope mode allows to implement variation of a default oscilloscope acquisition mode.

The FPGA internal architecture in Scope mode is depicted in **Figure 4.1.**

In Scope mode, the CAEN framework logic implements an oscilloscope-like acquisition path (red path): all channels of the board are acquired simultaneously for a programmed number of samples around a single trigger signal. The trigger source can be selected via Sci-Compiler. The trigger source is enforced to be common to all available channels.
Since all channels are recorded at the same time, the large external memory available on-board can be fully allocated to store each event. A very large record length can be achieved. The maximum supported record length per channel is about 10M samples. A custom logic can be developed with Sci-Compiler and inserted into the overall design by implementing the yellow block in figure (User Logic).

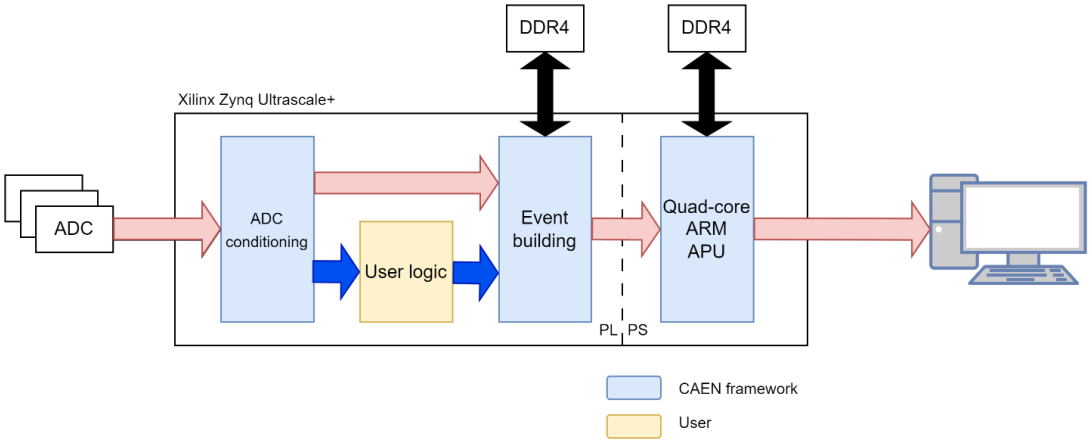


Figure 4.1: Open FPGA architecture in Scope mode.

DPP Mode for Digitizers 2.0

The DPP mode allows to implement online waveform processing with dedicated algorithms aiming to extract the parameters of interest from the input pulses, such as the pulse height, the gated integral (charge), the high-resolution timing (by sample interpolation), the pulse shape discrimination and others.

The FPGA internal architecture in DPP mode is depicted in **Figure 4.2.**

Unlike Scope mode, the output data streams can contain both raw samples and/or extracted parameters. Each channel is completely independent from the others, so one channel can produce only parameters while others the waveforms and others a mix of the two as well. The channel independence feature makes the DPP firmware a suitable solution also for the applications requiring the acquisition of raw waveforms (no need of online processing) that cannot run with the Scope firmware because of the common trigger.

Each channel can generate a local trigger and save the data into its own data queue. The board has a dedicated large external memory to store events. Since each channel contribute with its parameters and samples, the common external memory cannot be allocated for access to sparse access of every single channel, but an aggregate of channel events is collected and stored in external memory for buffering.

Unlike Scope mode, there is not a default acquisition implementation. So, an empty user logic will not produce a working implementation, in the sense that there will not be any data available for readout.

Channel events are time-ordered by the CAEN framework. Time reordering offline in software is not required.

A custom logic can be developed in Sci-Compiler and inserted into the overall design by implementing the yellow block in figure (User Logic).

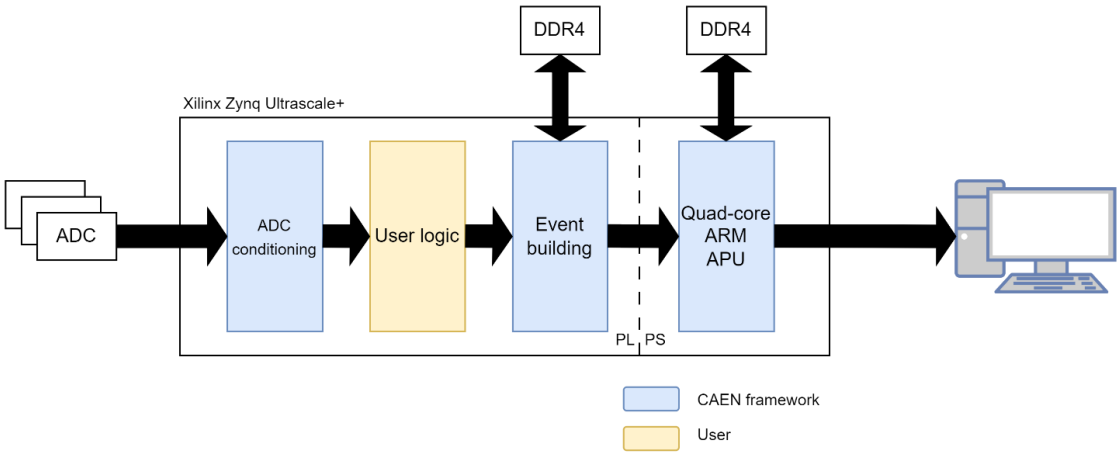


Figure 4.2: Open FPGA architecture in DPP mode.

How to choose between Scope and DPP mode project templates

Which project is right for you? Scope or DPP?

Scope Mode is mainly intended for raw waveform acquisition, simultaneously on the 64 channels with a common trigger, and **DPP Mode**, where DPP stands for Digital Pulse Processing, which is a set of algorithms aiming to detect the input pulses and extract the relevant parameters such as the arrival time with sub ns resolution, the pulse height and/or charge, the pulse shape, and other quantities of interest. In DPP mode, the digitizer looks more similar to an MCA (Multi Channel Analyzer), providing spectra and lists, rather than an oscilloscope or a waveform recorder. Nevertheless, the DPP mode keeps the possibility to save raw waveforms for post processing, signal inspection and debugging. Unlike the Scope mode, the DPP can work with independent triggers: typically, the channels are self-triggered and there is no need of an external global trigger. The DPP mode is therefore suitable for streaming readout. Although less frequent, there are cases where the DPP algorithms for the extraction of the pulse parameters are used in combination with a global trigger, whose purpose is to validate the self-triggers and build the event data packet simultaneously for all channels.

These are two main differences between Scope and DPP firmware:

- The DPP implements online waveform processing with dedicated algorithms aiming to extract the parameters of interest from the input pulses, such as the pulse height, the gated integral (charge), the high-resolution timing (by sample interpolation), the pulse shape discrimination and others. The output data can be just the extracted parameters, saved into a small data packet (typ. 8 or 16 bytes), thus making it possible to sustain an extremely high trigger rate, or the combination of parameters and raw waveforms when these are necessary for further offline analysis. It is also possible to mix events with and without attached waveforms in the same run; in particular, the User can define criteria to decide whether the pulse parameters extracted online are not reliable and, in that case, save the full waveform and let the software to apply more complex extraction algorithms. A typical example is the case where piled-up pulses cannot be deconvoluted and it is not possible to get the correct energy calculation in a simple way.

- The DPP makes the channels independent: unlike the Scope firmware, where the acquisition trigger is common to the 64 channels, in the DPP firmware each channel can generate a local trigger and save data into the local output FIFO, independently by the other channels. The data packets coming from the channels are sorted and merged by a dedicated block in the CAEN framework, before feeding the DDR memory. The independence of the channels makes the DPP firmware a suitable solution also for the applications requiring the acquisition of raw waveforms (no need of online processing) that cannot run with the Scope firmware because of the common trigger.

The DPP mode is more flexible than the Scope mode, also in terms of user accessibility, and for this reason it is more appealing for the users that want to develop their custom firmware.

The following table summarize the difference between the two possible development kits:

Aspect	**Scope**	**DPP**

Application	Waveform processing. Offline analysis.	Algorithms for pulse processing on a channel-by-channel basis. Offline analysis possible on small waveform fragments around trigger
Triggering scheme	Common trigger for all channels	Independent trigger: each channel can trigger based on local criteria. API support for global trigger.
Triggering rate	Depends on record length.	Typically, very large. When no waveforms are included in events, events are very small (minimum size is 8 bytes)
Data flow	All channels contribute data at the same time for the same record length. Circular buffer. Events generate on global trigger.	Channels generate events independently
Waveform fragments	Maximum 10M samples	Maximum 8K samples per channel

Table 4.1: main differences between Scope and DPP projects in Sci-Compiler for Digitizers 2.0

8 Building the project in Sci-Compiler

Sci-Compiler uses block diagrams to convert the concept of a Digital Pulse Processing algorithms into a full working firmware. Typically, a project is made of three main parts:

- **Inputs**, that are blocks mapping the actual hardware input of the hardware.
- **Processing,** i.e., blocks interconnecting each other and making the Realtime Signal Processing. For instance, in **Figure 5.1,** the signal from the input feeds a *Derivative Trigger*, which in turn activates a *Trapezoidal Filter*. The flat top of the trapezoid is sampled by the *Energy Sampler* block and energy of the incoming signal is then measured.

Note: refer to the software *online Help* for the accurate description of the Inputs and Processing blocks available for each compatible board.

- **Readout**, i.e., blocks to read the results of the Processing and transfer them to the PC as data. For instance, in **Figure 5.1,** a *Spectrum* block is used to read data coming from the *Energy Sampler.* Data are read by the *Spectrum* block when the *Energy Sampler* and an external digital signal on pin IO_A_0_A of the board are in coincidence (*Logic AND*).

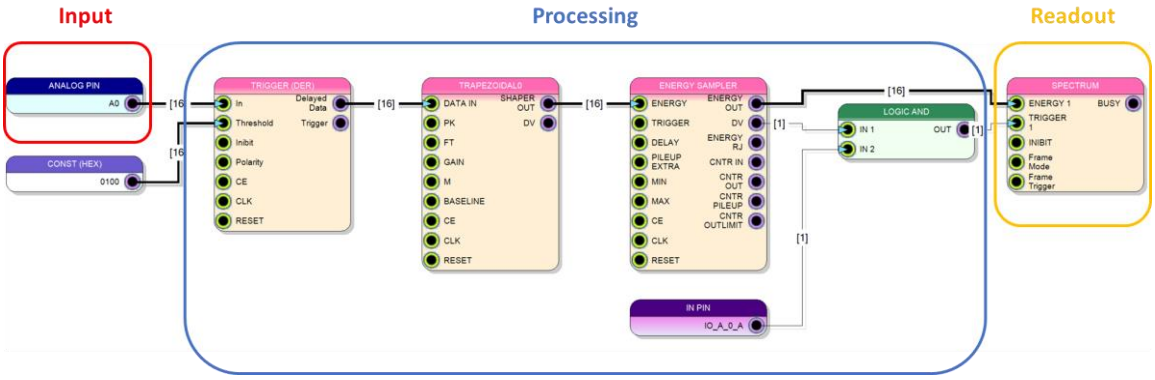
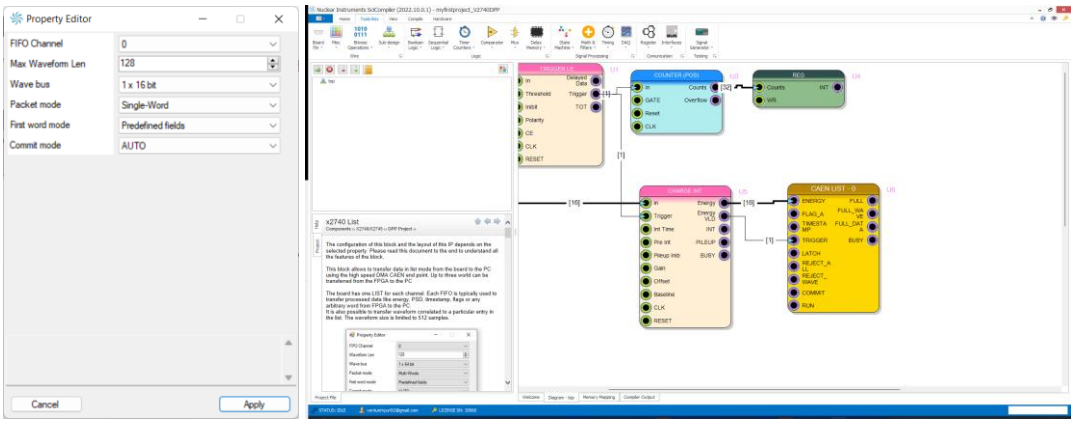


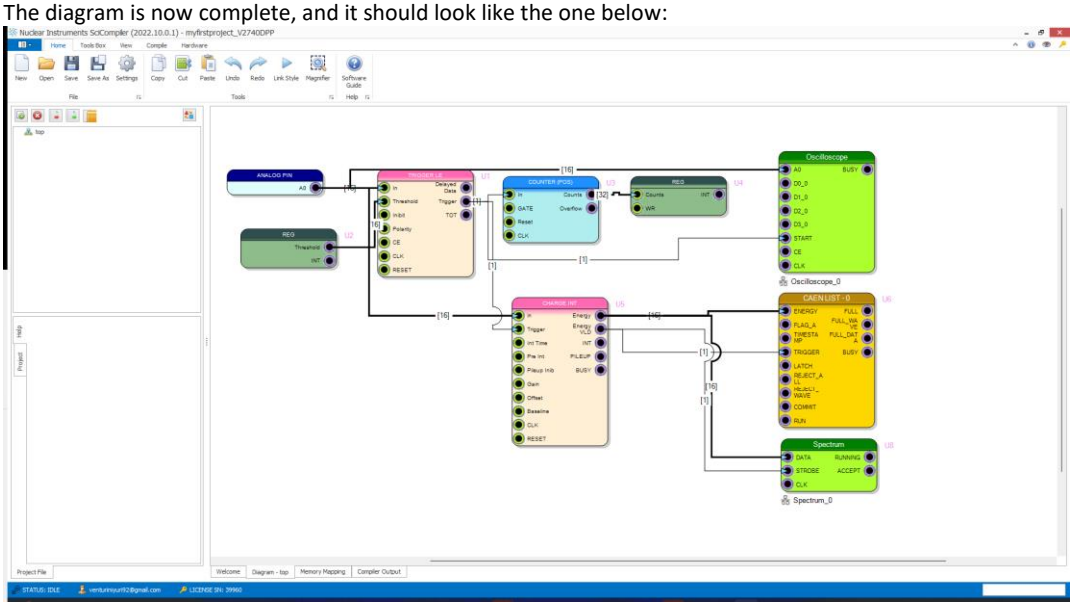
Figure 5.1: the typical structure of a Sci-Compiler project for Digital Pulse Processing.

10. for more details). The configuration details of the *CAEN list* block are available in the online help. For this purpose, we choose FIFO Channel = 0 , Packet Mode = Single Word and Predefined fields. Connect the blocks as shown in the figure below (Energy→ Energy ; EnergyVLD→Trigger)

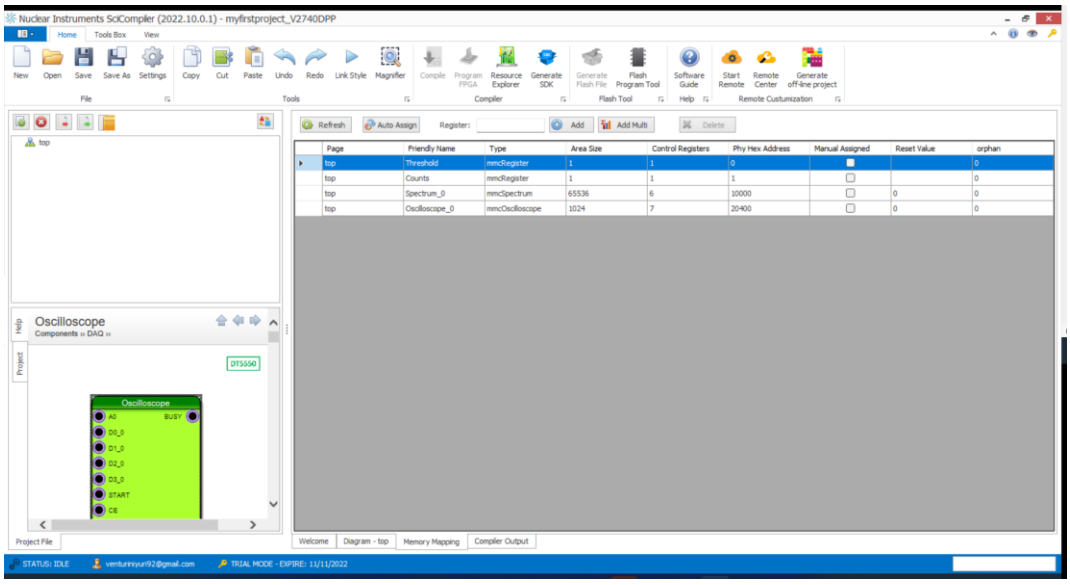


11. To easily test the firmware with the Resource Explorer before using more complete DAQ software code, we can place a **DAQ→Oscilloscope** block as debug monitor for the analog input and a **DAQ→Spectrum** block to readout the energy from the Charge Integration block. The Resource Explorer will also allow to set the threshold of the discriminator and read the number of counts detected by the counter.
Connect the Analog Pin A0 to A0 channel of the Oscilloscope, with the Oscilloscope Start given by the trigger output of the Leading Edge trigger block. Following the same concept, connect the Energy output of the Charge Integration to Data input of the Spectrum block, with the Strobe given by the Energy VLD signal coming from the Charge Integration.

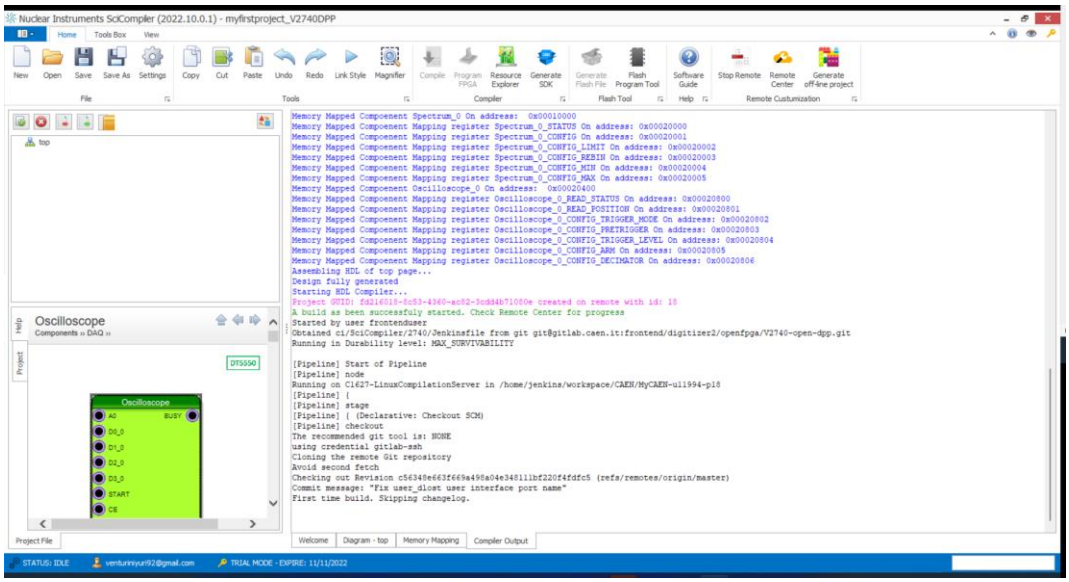
The diagram is now complete, and it should look like the one below:

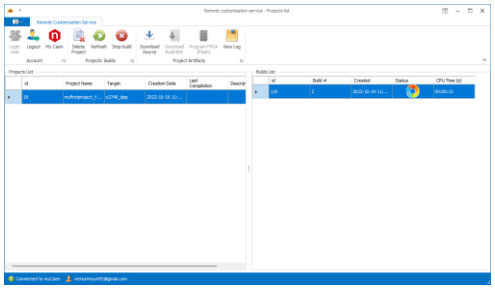


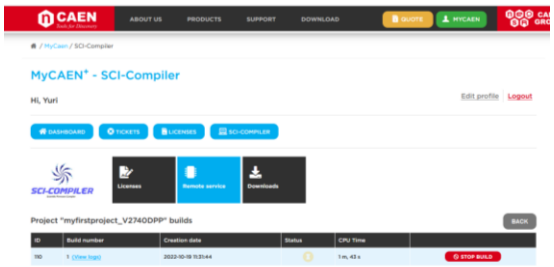
12. Before compilation, you need to set properly the Local Bus Readout blocks (registers, Oscilloscope and Spectrum). To do this, go into the *Memory Mapping* tab, where those blocks are listed, and **press *Auto Assign***. This operation assigns a Physical Hex Address to each block. A manual assignment is also possible.



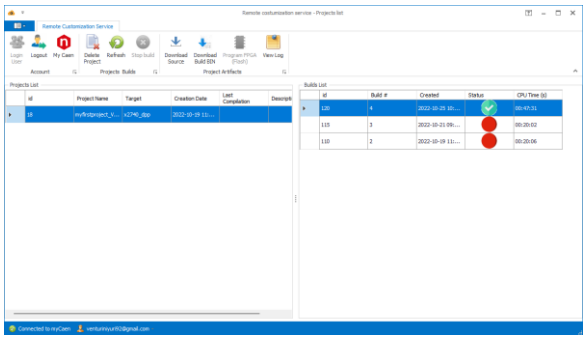
13. You are now ready to start the compilation. In particular, if you are logged in the MyCAEN+ (see Status Toolbar at the bottom of the software window), you can use the Remote Customization Service to synthetize the firmware for the V2740, with no need of installing any local Vivado compiler.
From *Compile* toolbar, press **Start Remote.** The software will automatically display the compilation log while compiling, and you can double-check the status of the compilation from the **Remote Center** or directly from your MyCAEN+ area (see figures below).

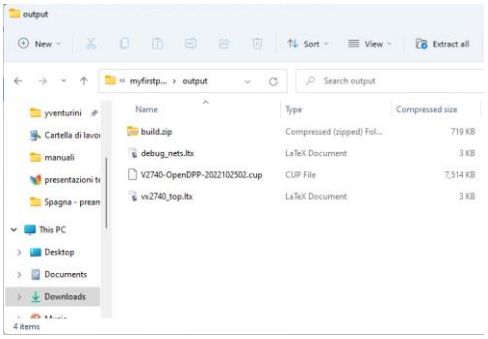




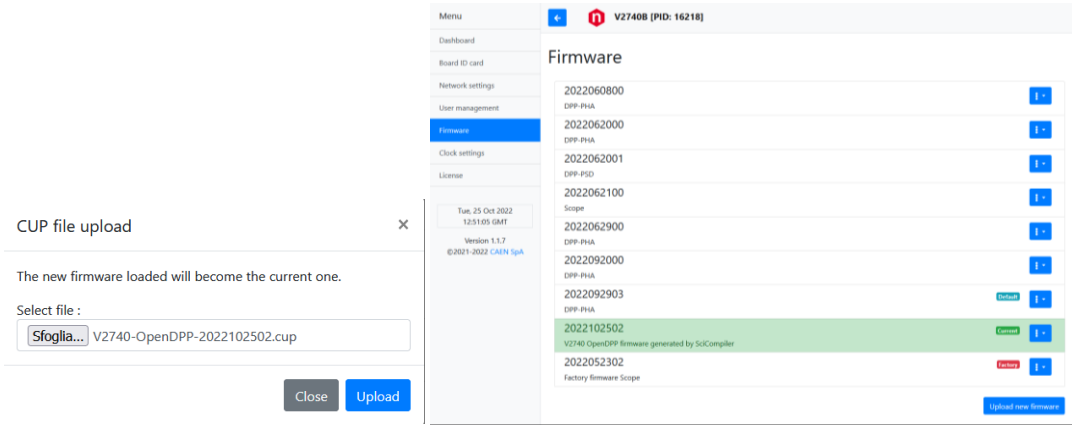


14. After compilation has been successful, the final firmware file (in the case of 2740/45 a .cup file) will be available through the Remote Center. Open the Remote Center, select the line corresponding to the successful compilation and press *Download Build BIN* to obtain the final firmware file. The Remote Center will download a .zip file containing the file to be deployed on the target hardware. In the 2740/45 case, the file is a *.cup* extension file. Extract that file from the .zip archive.



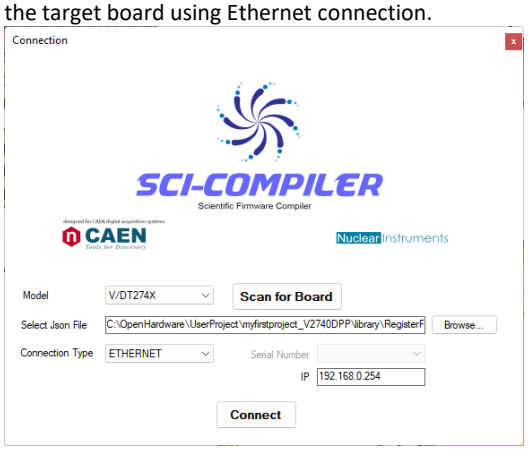


15. Load the .cup file onto the target board, in our case a V2740, following the instructions for firmware upgrade given in **[RD13] – Par.10.8**. Briefly recalling the procedure here, connect to the board Web Interface, surf into the Firmware page and *Upload new firmware.*

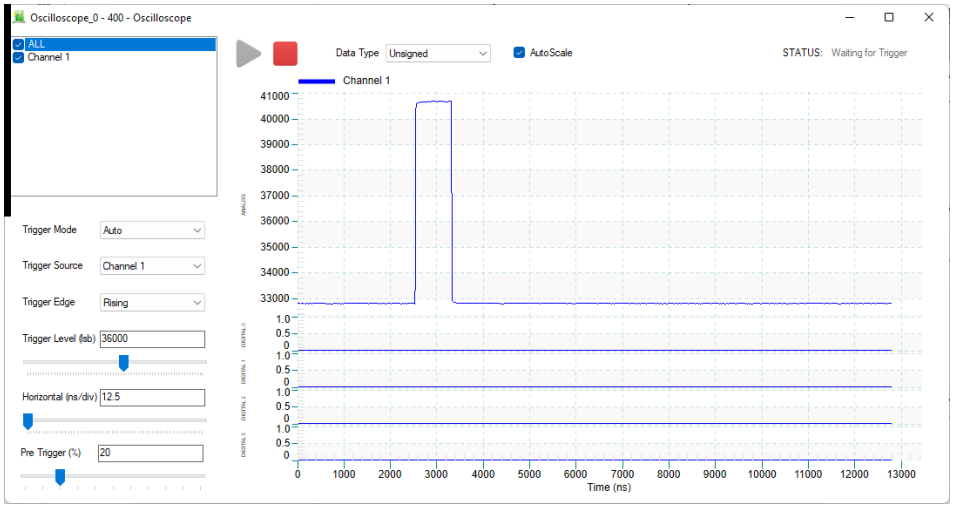


Note: the board can run the firmware generated via Sci-Compiler if a valid runtime license is installed onboard. It is possible to check the Sci-Compiler Runtime status in the *Dashboard* tab of the board Web Interface. If no runtime is installed, the firmware can run for 30-minutes, and it is possible to monitor the timebomb status from the Web Interface.

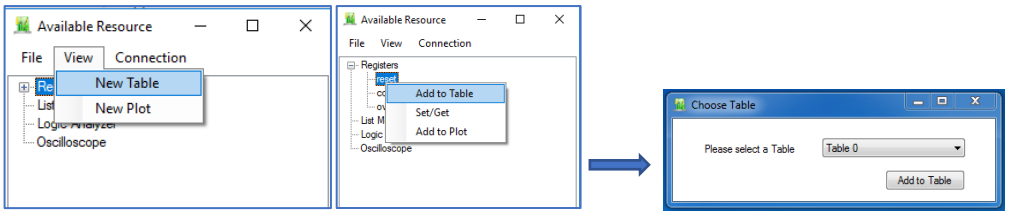
16. After upgrade, the board is running the firmware previously designed in Sci-Compiler and we can test it, for instance using the Resource Explorer tool. Open the Resource Explorer from Hardware toolbar and connect to the target board using Ethernet connection.



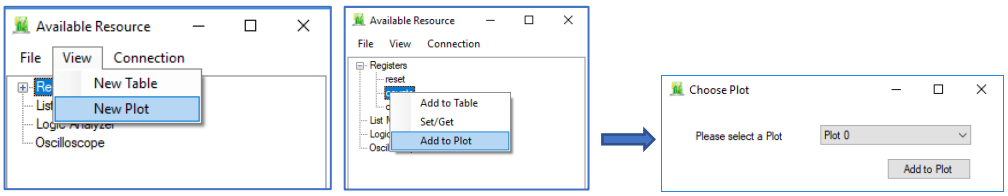
17. In the Resource Explorer, the available resources are listed (in this case Oscilloscope, Spectrum and the two registers to set the threshold and read the counts). As a first test, we want to see a signal at ch0 of the board using the Oscilloscope. Feed the V2740 with a signal (in this case a square pulse, 500mV, 1us width, 1kHz) and double-check the result from the Oscilloscope. From the Resources list right click → Oscilloscope→ View to open the oscilloscope window. Set Trigger Source = Free Running at first, to see the level of the baseline and press start. Once you have an estimate of a proper trigger level, set Trigger Source = Channel 1. In our case we set Trigger Level = 36000 to trigger correctly on the positive edge of the signal.



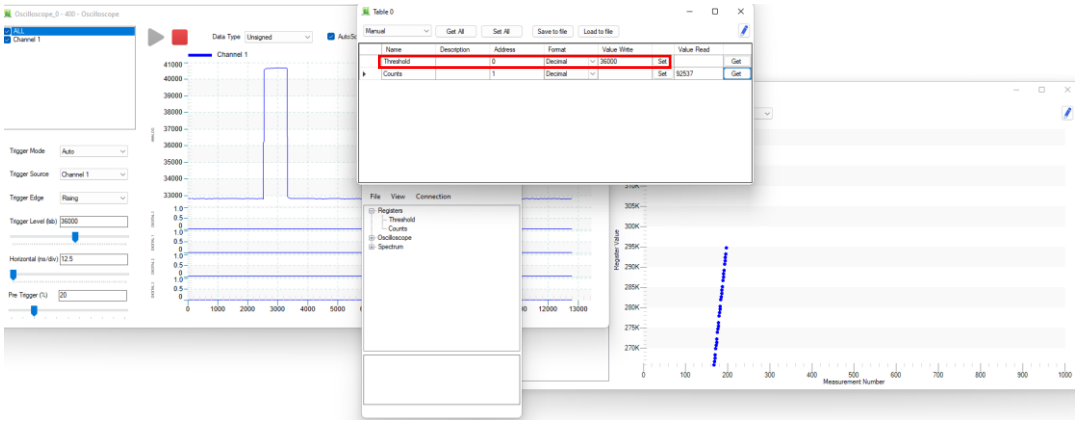
18. Now that we have found the right threshold level from the Oscilloscope instrument, we set the same threshold in the firmware threshold register. Click View→ New Table. Right click on each register name → Add to table.



At the same time, it is possible to add a plot for the counts register. Click ViewàNew Plot. Right-click on "counts" register and Add to Plot



From the table, set Threshold register =36000 and monitor the counts trend in the plot. Increasing the threshold up to 45000, it is possible to notice that the plot will stop, due to the leading edge trigger being above the signal level.



In the same way it is possible to look at the energy calculation result using the *Spectrum* resource listed among the Available Resources.

9 Technical Support

To contact CAEN specialists for requests on the software, hardware, and board return and repair, it is necessary a MyCAEN+ account on www.caen.it:

https://www.caen.it/support-services/getting-started-with-mycaen-portal/

All the instructions for use the Support platform are in the document:

A paper copy of the document is delivered with CAEN boards.

The document is downloadable for free in PDF digital format at:

https://www.caen.it/wp-content/uploads/2022/11/Safety_information_Product_support_W.pdf

Sci-Compiler for Digitizers 2.0
Quick Start Guide to write custom firmware for Digitizers 2.0

CAEN GmbH
Brunnenweg 9
64331 Weiterstadt
Germany
Tel. +49 (0)212 254 4077
Mobile +49 (0)151 16 548 484
info@caen-de.com
www.caen-de.com

CAEN Technologies, Inc.
1 Edgewater Street - Suite 101
Staten Island, NY 10305
USA
Phone: +1 (718) 981-0401
Fax: +1 (718) 556-9185
info@caentechnologies.com
www.caentechnologies.com

CAEN S.p.A.
Via Vetraia 11
55049 - Viareggio
Italy
Phone +39 0584 388 398
Fax +39 0584 388 959
info@caen.it
www.caen.it

CAENspa INDIA Private Limited
B205, BLDG42, B Wing,
Azad Nagar Sangam CHS,
Mhada Layout, Azad Nagar, Andheri (W)
Mumbai, Mumbai City,
Maharashtra, India, 400053
info@caen-india.in
www.caen-india.in