CAEN
Tools for Discovery

Electronic Instrumentation

User Manual UM5833

A1702/DT5702

32-channel Silicon Photomultipliers Readout Front-End Board

Rev. 2 - July 20th, 2018

Purpose of this Manual

This document contains the technical description of the front-end electronic board (FEB) A1702/DT5702 for the readout of signals from an array of 32 Silicon Photomultipliers (SiPM) and the hardware/software operation.

Change Document Record

Date	Revision	Changes
January 10th, 2017	00	Initial release
May 10th, 2018	01	Added firmware description and ordering option. Revised chapter **Board Structure and Operation.** Extended chapter **Test Software.**
July 20th, 2018	02	Revised chapter **Power Requirements.**

Symbols, abbreviated terms and notation

ADC	Analog-to-digital converter
ASIC	Application Specific Integrated Circuit
GUI	Graphical User Interface
SiPM or MPPC	Silicon Photo-Multiplier
FEB	Front End Boards
PPS	Pulse Per Second
SPI	Serial Peripheral Interface
TDC	Time-to-digital converted

Reference Documents

[RD1] A.Aloisio, et al., *FPGA Implementation of a High- resolution Time-to-Digital Converter*, 2007 IEEE Nuclear Science Symposium Conference Record, N15-137.

[RD2] M.Auger et al., *Multi-channel front-end board for SiPM readout*, Journal of Instrumentation, Volume 11, 2016

[RD3] WeeROC Citiroc1A datasheet (https://www.weeroc.com/en/products/citiroc-1a)

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

Index

Purpose of this Manual 2
Change Document Record 2
Symbols, abbreviated terms and notation 2
Reference Documents 2
Index 3
List of Figures 3
List of Tables 4
1 Introduction 6
2 Technical Specifications 7
3 Power Requirements 8
4 General Functionality 9
32-channel SiPMs readout Front-End Board 9
5 Board Structure and Operation 10
General structure and main components 10
Triggering logic 10
Board firmware 11
Bias generator and analog signal readout 12
Time stamp generator 13
Event buffer and back-end Ethernet interface 14
Connectors and jumpers 16
Back-end communication and FEBDTP v3.0 data transmission protocol 18
6 Test Software 24
Prerequisites 24
Compiling and installation 24
Usage 24
Software interface 25
DAQ FEB controls 26
Display Area 27
Statistics bar 32
DAQ using DT5702 33
7 Annexes 36
CITIROC Configuration files 36
8 Technical Support 40

List of Figures

Figure 1.1: General view of CAEN DT5702 6
Figure 3.1: AC/DC power supply provided with the DT5702 module. Only the power supply adapter connector is provided with A1702. 8
Figure 4.1: General view of the CAEN A1702 Front-End Board 9
Figure 5.1: General block-scheme of the FEB 10
Figure 5.2: Block-scheme of the triggering circuit for even-odd adjacent channels coincidence 11
Figure 5.3: Timing diagram of the triggering circuit. Ch0 & Ch1 coincidence logic is shown. The second event on Ch0 (red) is in coincidence with the event on Ch1 (blue) and triggers a readout cycle. 11
Figure 5.4: Block-scheme of the analog signal processing circuit 12
Figure 5.5: Timing diagram of the analog signal processing circuit 13
Figure 5.6: Typical performance of analog signal processing circuit (with Hamamatsu S12825-050P-MPPC). The blue spectrum is a dark count measurement with a threshold of 0.5 p.e., while the red spectrum is obtained triggering on some other channel. 13

Figure 5.7: Block-scheme of time stamp generation circuit 14
Figure 5.8: Block-scheme of the oscillator control loop for the time stamp generation circuit 14
Figure 5.9: Block-scheme of the back-end data transmission and control interface 15
Figure 5.10: the 8-bits MAC address switch array. The configuration shown in the picture [01000000] corresponds to MAC[5] = 0X40. 15
Figure 5.11: Scheme of the FEBs daisy chain and connection to the host computer. Note that the MAC[5] byte is set differently for each board. 16
Figure 5.12: External connectors 16
Figure 5.13: Pinout of the X1 connector 17
Figure 5.14: Jumpers and bias regulator 17
Figure 5.15: Auxiliary connections and CPU reset button 17
Figure 5.16: Pinout of the auxiliary connector X5 17
Figure 5.17: JTAG pin headers for firmware programming 18
Figure 5.18: Pinout of JTAG pin headers JT1 (on the left) and JT2 (on the right) 18
Figure 6.1: screenshot of the Ubuntu terminal at the DAQ software startup. The MAC address and the firmware release are highlighted in red and yellow respectively. 25
Figure 6.2: main page of the DAQ software GUI. Its main parts are highlighted: the DAQ controls menu in blue, the display area in red and the statistics bar in green. 26
Figure 6.3: "Configuration" tab in the GUI to control several bits of the CITIROC Slow Control register 28
Figure 6.4: the "All histos" tab during acquisition. Here the FPGA firmware rel FLX7.003 was used and only channels 0,1,2,3 where enabled to trigger in coincidence. Non-triggering channels are displaying their characteristic noise peak. See DAQ using DT5702 for more details. 29
Figure 6.5: the "*One channel*" tab showing the histogram of channel 1 during a DAQ. Here typical peaks generated by a SiPM are shown. 30
Figure 6.6: the "Timing data" tab when a PPS timer is connected to T0 LEMO input. 30
Figure 6.7: the "Channel profile" tab when channels 1 and 29 are acquiring events in coincidence using “nfoldcoinc” FPGA software. 31
Figure 6.8: the "Event rate” tab during DAQ, while changing the DAC1 threshold. The event rate is plotted as a function of the Poll Nr. In the bottom statistics bar it is possible to read the composition of the last transmitted poll (113 events acquired in 0.08 s). 31
Figure 6.9: the "Event display" tab showing the rainfall-style plot of the event number recorded by each channel. 32
Figure 6.10: the statistics bar at the bottom of the GUI 32
Figure 6.11: the statistics bar during different DAQs. The top bar is a DAQ performed with optimal parameters: the trigger rate is sufficiently low and a small amount of events is lost (only 2%). The bottom bar is the result of a bad set of DAQ parameters: the trigger rate is too high and many events are lost (41.9%). 33
Figure 6.12: a possible setup to perform DAQ using a CAEN DT5702 33
Figure 6.13: DAQ of two SiPMs signals in coincidence. Triggering channels 12 and 13 are not showing any noise peak, while the other channels are clearly showing their pedestals. 34
Figure 6.14: DAQ of two SiPMs signals in “OR32” mode, using the FPGA firmware rel FLX7.003. With respect to Figure 6.13, channel 12 is showing its intrinsic low energy noise peaks. Non-triggering channels are also showing their noise peak. Note also that, w.r.t. Figure 6.13, the DAC1 threshold has been increased to avoid trigger rate massive increase. 34
Figure 6.15: signal from a SiPM illuminated by a CAEN SP5601 LED driver. A typical SiPM peak is visible. 35

List of Tables

Table 1.1: Available ordering options 6
Table 2.1: Technical specifications of CAEN A1702/DT5702 7
Table 5.1: FEBDTP V3.0 datagram structure and example. The total length of the datagram is in the range 64 to 1500 bytes. 18
Table 5.2: Command to read ETHERNET switch control/status register. REG is the register address to read. The answer of the FEB is contained in the FEB-OK-SR command payload. 19
Table 5.3: Command to write ETHERNET switch control/status register. REG is the register address to be written. The content to be written is contained in the FEB-WR-SR command payload. 19
Table 5.4: Command to read ETHERNET switch control/status, 256 registers at once. The answer of the FEB is contained in the FEB-OK-SR command payload. 19
Table 5.5: Command to write ETHERNET switch control/status, 256 registers at once. The content to be written is contained in the FEB-WR-SRFF command payload. 19

Table 5.6: Command to broadcast the MAC of the host to all FEBs and to collect MACs of all FEBs into the host's client DB. It returns the firmware version string in FEB_OK data field. If RR RR >0, it sets the VCXO correction value to RR RR ...20
Table 5.7: Acquisition control command. 20
Table 5.8: Command to turn SiPMs HV bias ON 20
Table 5.9: Command to turn SiPMs HV bias OFF 20
Table 5.10: This command requests current trigger rate from the FEB, returned as float in the first 4 bytes of Data field of FEB_OK reply 21
Table 5.11: Command to read CITIROC Slow Control register (SR) 21
Table 5.12: Command to write and latch CITIROC Slow Control register (SR) 21
Table 5.13: Command to read the event buffer. The FEB_DATA_CDR message is repeated till the FEB buffer is empty. The FEB_EOF_CDR command terminates the data buffer transmission 21
Table 5.14: Command to read the CITIROC Probe register (PMR) 21
Table 5.15: Command to write and latch in the CITIROC Probe register (PMR) 22
Table 5.16: Command to read the FPGA input flexible logic register (FIL). The 9 bytes control string is formed by 4-bytes mask1, 4-bytes mask2 and a majority byte. 22
Table 5.17: Command to write and latch in the FPGA input logic register (FIL) 22
Table 5.18: Command to read SPI Flash Interface content. Bytes A0-A2 define the starting address in PROM address space. Bytes N0-N1 define the number of requested blocks. The CC CC field contains the CRC code of the block. 22
Table 5.19: Command to write SPI Flash Interface content. Options are specified in RR RR field. 23
Table 6.1: latest firmware revision name. 25
Table 7.1: description of CITIROC probe register bit stream as reported in *"CITIROC_PROBEbitstream.txt"* 36

1 Introduction

The Multi-channel Front-End Board A1702/DT5702 is a custom design developed by the Albert Einstein Center for Fundamental Physics of the University of Bern to **perform charge and timing measurements with SiPMs.**

Given the increasing use of SiPMs in physics experiments, this solution can become a valid approach for a variety of applications thanks to its **flexibility, compact form factor and channel density**. The board provides adjustable bias voltage and is able to process and digitize the analog signals

The analog input signal is processed by a WeeROC (*) CITIROC 1A ASIC, providing a charge amplifier with configurable gain, a fast shaper with peaking time of 15 ns for **trigger formation** and a slow shaper with configurable peaking time in the range of 12.5 ns to 87.5 ns for **amplitude measurements**.

The triggers are routed to a XILINX Spartan-6 FPGA, where the event triggering logic is realized. The height of analog signals are measured by a Track-and-Hold (T&H) circuit and multiplexed to a single analog output. This output is routed to an ARM micro-controller chip, which operates as an ADC. Thanks to this process the board is able to measure the **energy spectrum** on each channel and obtain a **cumulative charge image** using SiPMs arrays or matrices.

The board communicates with the host computer through Ethernet protocol.

CAEN distributes this product thanks to a License Agreement with the University of Bern valid worldwide.

Two different models are available by ordering options (see **Table 1.1**).

Board Models	Description	Product Code
A1702	32-channel SiPM readout Front-End Board	WA1702XAAAAA
DT5702	32-channel SiPM readout Front-End Board BOXED	WDT5702XAAAA

Table 1.1: Available ordering options

Figure 1.1: General view of CAEN DT5702

(*) https://www.weeroc.com

2 Technical Specifications

GENERAL	**Form Factor** 218x16x62.5 mm^3 (WxHxD) A1702 230x20x80 mm^3 (WxHxD) DT5702	
POWER CONSUMPTION	0.550 A @ 5V	
SiPM INPUT	32 channels (72-pin SiPM connector)	
INPUT POLARITY	Positive	
BIAS VOLTAGE	Common bias: 20 V to 90 V - single-channel fine adjustment +0.5 V to 4.5 V	
DIGITAL CONVERSION	80 Ms/s , 12 bits ADC	
CHARGE AMPLIFIER	Configurable gain and dynamic range of 1 to 2500 p.e. (@SiPM gain of 10^6)	
SHAPER	- Fast shaping: peaking time of 15 ns - Slow shaping: configurable peaking time of 12.5 ns to 87.5 ns	
DISCRIMINATION	From 0 to 50 SiPM photoelectrons - 10-bit DAC common threshold - 4-bits DAC for single-channel fine adjustment	
TDC	250 MHz clock frequency	
TIME STAMP	**Resolution** up to 1 ns	
DIGITAL I/O	**TIN** (LEMO) Validation input **T0** and **T1** (LEMO) Reference inputs for timing 3.3V LVCMOS, High impedance	**TOUT** (LEMO) Output pulse 3.3 LVCMOS
TRIGGER	**Trigger source** - *Internal trigger:* even-odd adjacent channels coincidence, individual channel trigger - Trigger validation signal on TIN connector	
MEMORY BUFFER	up to 1024 events	
ASIC	WeeROC CITIROC1A	
COMMUNICATION INTERFACE	100 Mbps Ethernet links	
MULTI BOARDS CONNECTION	Daisy-chain of up to 256 units into one network interface	
FIRMWARE	- even-odd adjacent channels coincidence or channels OR32 trigger	
SOFTWARE	DAQ software is a CERN ROOT script	

Table 2.1: Technical specifications of CAEN A1702/DT5702

3 Power Requirements

The CAEN A1702/DT5702 standalone module is powered by an external AC/DC stabilized power supply. The AC/DC power supply is provided with the DT5702 board and included in the delivered kit. Only the power supply adapter connector is provided with the A1702.

Input: 100-240 V AC, 47-63 Hz; Output: 5.0 V, 3.3 A. The typical power consumption is 0.8 A (@ +5 V).

Note.: Using a different power supply source, like battery or linear type, it is recommended the source to provide +5 V and, at least, 1.5 A.

Figure 3.1: AC/DC power supply provided with the DT5702 module. Only the power supply adapter connector is provided with A1702.

CAUTION: to manage the product, consult the operating instructions provided.

A POTENTIAL RISK EXISTS IF THE OPERATING INSTRUCTIONS ARE NOT FOLLOWED!

CAEN provides the specific document "Precautions for Handling, Storage and Installation" available in the documentation tab of the product web page that the user is mandatory to read before to operate with CAEN equipment.

4 General Functionality

32-channel SiPMs readout Front-End Board

The A1702/DT5702 is designed to perform **charge and timing measurements with SiPMs and SiPM arrays**. It is suitable to readout up to 32 SiPMs signals, using a coincidence trigger or a single-channel trigger. The main functionalities of the board are reported below:

1. Provides bias voltage in the range of 20-90 V, individually adjustable for each channel;
2. Amplification and shaping of the SiPM output pulse;
3. Discrimination of shaped signal at configurable level from 0 to 50 photo-electrons;
4. Energy and time measurements;
5. Provides basic coincidence of signals from each pair of adjacent even-odd channels;
6. Multiple boards event validation using an external signal;
7. LEMO I/O for time reference and control signals;
8. Data buffering;
9. Efficient back-end communication based on Ethernet standard;
10. Daisy chain of up to 256 boards into one network interface;

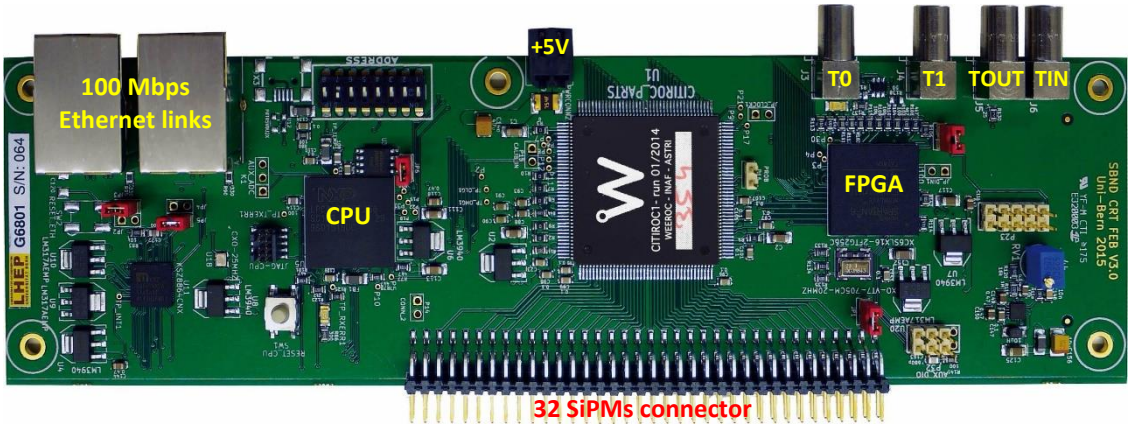


Figure 4.1: General view of the CAEN A1702 Front-End Board

5 Board Structure and Operation

General structure and main components

The board is designed to acquire 32 analog signals coming from an array of 32 SiPMs. A general view of the board is shown in **Figure 4.1.** At one side, a 72-pin connector is located; the other side hosts a 5V power connector, two RJ45 network jacks for Ethernet communication and four LEMO connectors for time reference and control signals.

A general block scheme of the FEB is shown in **Figure 5.1**. Each analog input signal is processed by a Weeroc CITIROC ASIC. For each channel, the chip provides charge amplifier with configurable gain, fast shaping with the peaking time of 15 ns and slow shaping with configurable peaking time in the range of 12.5 ns to 87.5 ns. Each signal from the fast shaper is discriminated at configurable level and the ASIC produces the correspondent digital signals (C0-C31) for event triggering.

These 32 signals are routed to *XILINX Spartan-6 FPGA* chip, where the event triggering logic is realized. Moreover, the FPGA produces event timestamps.

The analog signal heights are sampled in the ASIC Sample-and-Hold (S/H) circuit and multiplexed to a single analog output. This output is routed to the ADC (part of *NXP LPC4370 ARM micro-controller* chip).

The micro-controller also provides the common high voltage bias for each of the 32 MPPCs.

The communication interface is a 3-port Ethernet switch, which allows connection to a host computer and daisy chain of multiple boards (up to 256 FEBs).

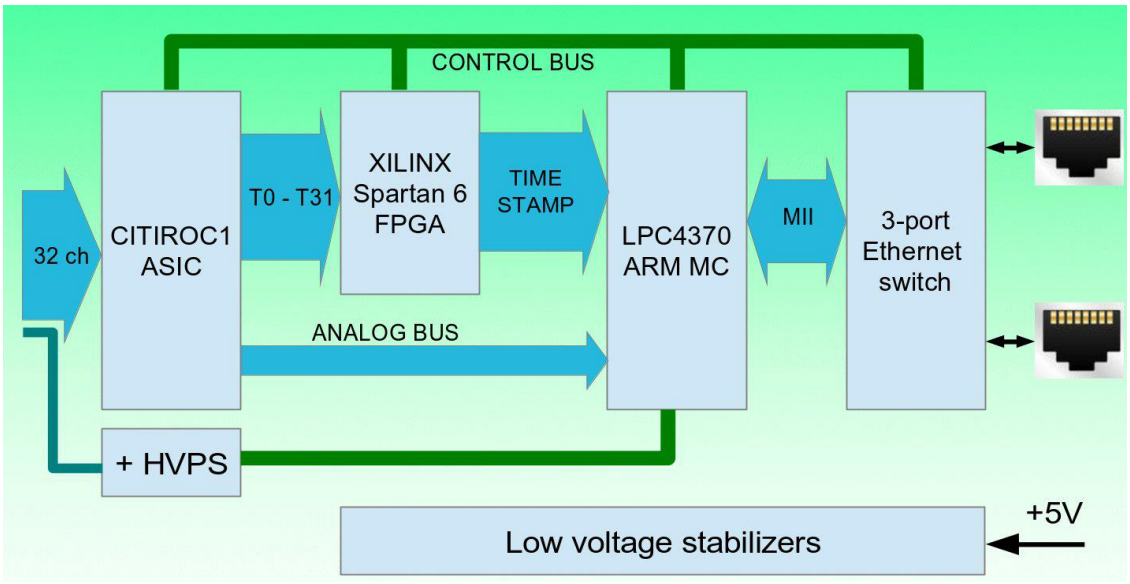


Figure 5.1: General block-scheme of the FEB

Triggering logic

In **Figure 5.2,** a block-scheme of the trigger formation circuit is shown.

Within the ASIC, for each of the 32 channels, a charge amplifier with configurable gain and dynamic range of 1 to 2500 p.e. (at SiPM gain of 10^6) is followed by a fast RC-CR shaper with peaking time of 15 ns, providing trigger signals.
The 32 digital trigger signals (C0 to C31, 3.3V LVCMOS) are routed to the FPGA where they are combined with AND logic to form coincidence signals of even-odd adjacent channels (C0&C1, C2&C3, … C30&C31). These signals, together with each individual channel trigger (C0-C31), are fed to the trigger selector where the *"primary event" trigger* (general board trigger) is formed with a combination of these signals (see below). **Whenever a *"primary event"* trigger is issued, all 32 channels are digitized and read out and the time stamp for the event is generated**. During digitization the board does not accept triggers, but counts missed ones, if any. The number of missed triggers is recorded.
Depending on the firmware, **the board trigger is issued by even-odd adjacent channels coincidence or by single channels trigger** (see paragraph **Board firmware** for more details).

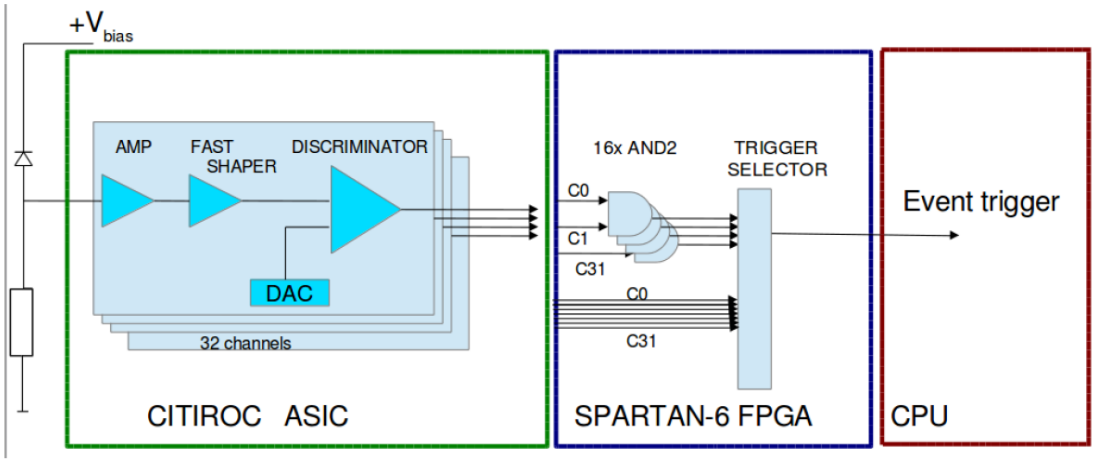


Figure 5.2: Block-scheme of the triggering circuit for even-odd adjacent channels coincidence.

The timing diagram of the triggering circuit is shown in **Figure 5.3**.
In case of coincidence trigger, the coincidence time window varies from 0 to 30 ns depending on the amplitude of the input pulse w.r.t the discriminator threshold. The "primary event trigger", after a delay of 50 ns, leads to the generation of the "track and hold" signal to memorize instantaneous signal heights of all 32 channels. The "track and hold" signal, that also defines the readout window, is kept active for 150 ns.

The "track and hold" signal is output to the "TOUT" LEMO connector of the board to allow the user to perform trigger validation between multiple boards. When a FEB gets its internal trigger, it waits for 150 ns for the presence of a high level at "TIN" LEMO connector, sourced by the "TOUT" signal of its master FEB. If no signal is received by "TIN", the "track and hold" signal is reset by the FPGA and the event is discarded. Such functionality can be disabled by shortcutting "TOUT" signal to "TIN" input at the individual FEB.

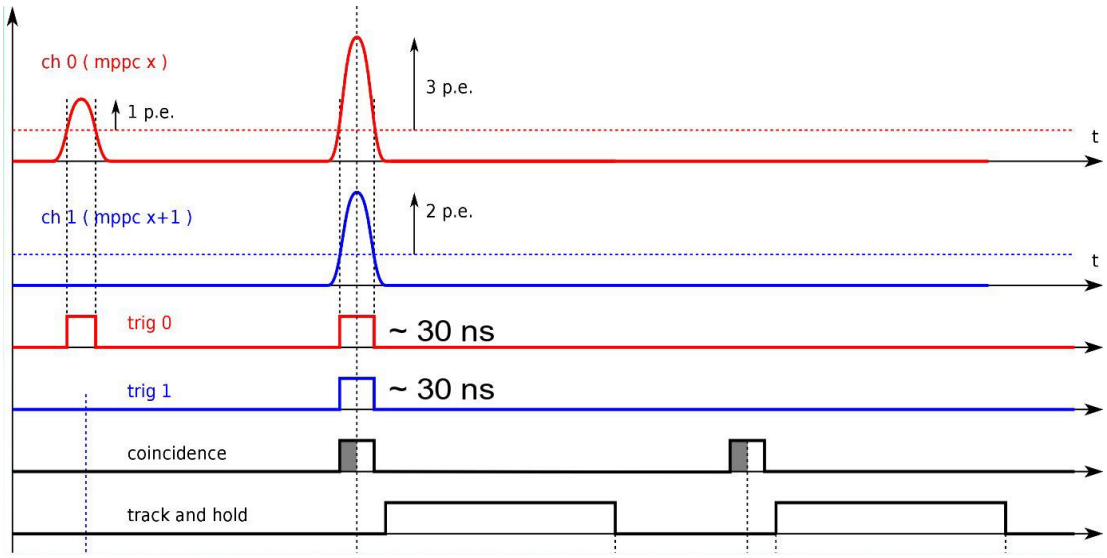


Figure 5.3: Timing diagram of the triggering circuit. Ch0 & Ch1 coincidence logic is shown. The second event on Ch0 (red) is in coincidence with the event on Ch1 (blue) and triggers a readout cycle.

Board firmware

The triggering logic of CAEN A1702/DT5702 board is controlled at firmware level. In particular, it allows the user to readout SiPM signals triggering on

- **adjacent even-odd channels coincidence :** the *"primary event"* trigger is the logic OR of the active pairs coincidence (e.g. C0&C1 OR C2&C3 OR C8&C9).
- **individual channels trigger**. the *"primary event"* trigger is the logic OR of the active channels (e.g. C0 OR C1 OR C30).

Depending on the firmware version the readout triggering logic can be slightly different (see table below and **Software interface** paragraph for instructions on how to get your firmware release)

Firmware release	Features
FLX7.003	Selectable even-odd adjacent channels coincidence and channels OR32
IAP7.007	Even-odd adjacent channels coincidence

Bias generator and analog signal readout

The block-scheme of the analog signal processing circuit is shown in **Figure 5.4**.
The common bias voltage for MPPCs is generated by a switching stabilized power supply circuit, operating at 100 MHz. The voltage can be adjusted from 20V to 90V by the trimmer resistor (see section **Connectors and jumpers** for details). This voltage is common for all 32 SiPMs connected to the FEB inputs and can be read with a voltmeter at connector X2 on the board (see section Connectors and jumpers). The power supply can be enabled or disabled by a dedicated signal, generated by the on-board CPU.
The individual bias voltage adjustment is performed by 8-bits DACs within the CITIROC. The DAC's positive output levels are supplied to the signal lines as DC-offset, therefore increasing this voltage reduces effective bias for individual SiPMs. The full DAC range is +0.5 to +4.5 V.
Amplified charge pulse from SiPMs are shaped by a slow RC-CR shaper with configurable peaking time (12.5 ns to 87.5 ns). This time is adjusted in such a way that the "track and hold" signal, delayed by 50 ns w.r.t. event, has its rise flank at the flat top of the peak of the shaper output pulse, minimizing the noise due to time jitter. The amplitudes of the 32 shaper outputs are latched at the hold circuit and routed to an analog multiplexer. When the CPU receives the trigger interrupt, it initiates readout cycle.

Note: only the high gain line of the Citiroc1A is used for charge measurements. The low gain line is present but not implemented in the board structure.

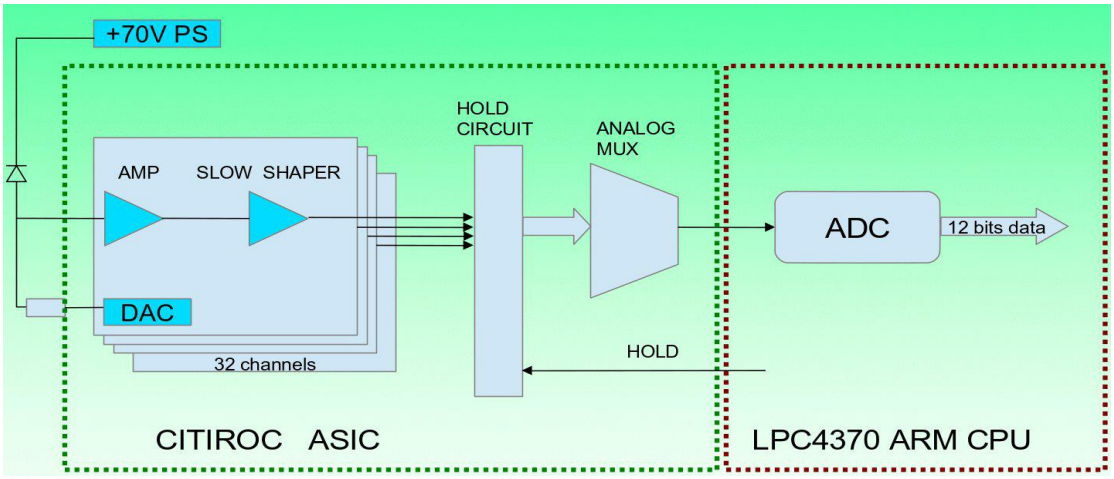


Figure 5.4: Block-scheme of the analog signal processing circuit

The timing diagram is shown in **Figure 5.5**. The solid coloured lines represent the shaper outputs, the dotted lines the outputs of S/H circuits. The event illustrated in **Figure 5.5** is triggered by coincidence of two neighbouring channels (x and x+1). The CPU controls multiplexing of all 32 outputs via a single line, which is routed to 12-bits ADC input. In **Figure 5.5**, only 8 channels multiplexing is shown for simplicity. Once all 32 channels are digitized and stored in the event buffer, the CPU sends the reset signal to FPGA and completes the readout. Each of the 32 charge amplifiers can be individually enabled or disabled by CITIROC configuration bit stream.
The performance of the analog readout circuit is illustrated in **Figure 5.6**, where the amplitude spectrum for dark counts with a discrimination threshold of 0.5 p.e. is shown in blue. The red spectrum shows the location of pedestal when triggered by some other channel. This means that pedestals will always be visible for non-triggering channels.

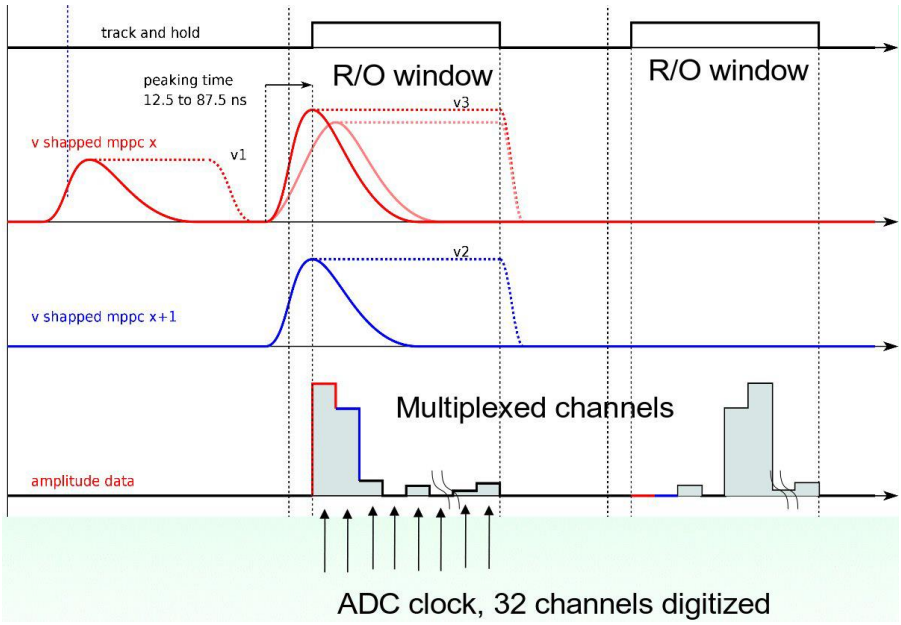


Figure 5.5: Timing diagram of the analog signal processing circuit

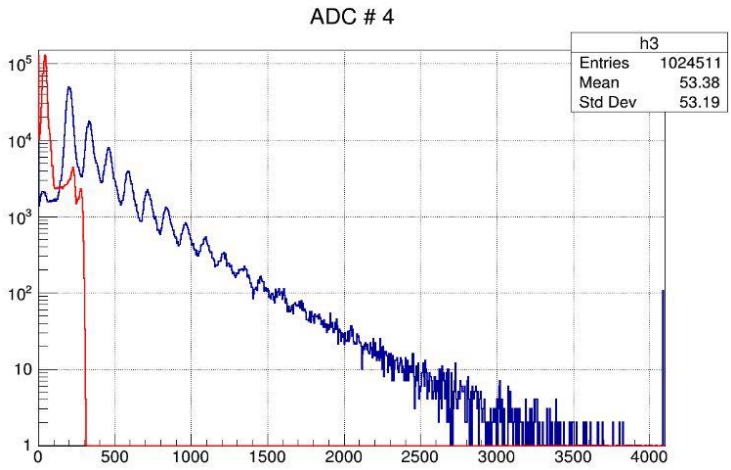


Figure 5.6: Typical performance of analog signal processing circuit (with Hamamatsu S12825-050P-MPPC). The blue spectrum is a dark count measurement with a threshold of 0.5 p.e., while the red spectrum is obtained triggering on some other channel.

Time stamp generator

The Time-to-Digit Converter (TDC) of the time stamp generator is composed of the coarse counter, working at the clock frequency of 250 MHz, and the delay-chain interpolator, improving accuracy down to 1 ns. The 20 MHz temperature-compensated voltage-controlled crystal oscillator (VCXO) is used as a source for the reference clock of the FPGA and timing circuit. The feedback voltage for VCXO is generated by the 10-bits DAC under control of the on-board CPU (see **Figure 5.8**). The solution is based on the approach published in [RD1].

The time stamp is defined as the interval of time between the event of interest and the input reference pulse. For each event, the FEB is capable of recording ***two independent time stamps*** w.r.t positive flank on "T0" and "T1" LEMO inputs (see **Figure 5.7**).

Each time stamp is a 32-bits word, having time information in 30 Least Significant Bits (LSB) represented in Gray code. Two Most Significant Bits (MSB) are used for flagging special events. Two special events are foreseen. The first is the arrival of the reference signal at either "T0" or "T1" inputs. For such an event, the time passed since the previous reference event is recorded, and the timing circuit is reset to zero. This allows to measure the period between reference signals and, in case the real reference signal period is highly stable and accurate, the measured period allows deriving deviation of the internal on-board oscillator frequency from nominal value. If this deviation is known, it can be applied offline to scale all time stamps between two reference pulses to recover accuracy. An eventual frequency correction can be derived on-board basing on the measured periods between reference pulses, supplied to "T0" input from an external high-stability GPS-disciplined pulse-per-second (PPS) generator. Refer to [RD2] for more details on the performance of the oscillator control loop.

The second special event happens in the absence of the reference pulse for more than 1074 ms, which leads to overflow of the coarse counter. This situation is flagged and can be used offline to invalidate the time stamp until the presence of the reference pulse is restored.

Together with two 32-bits words from the two time stamp generators, an additional word is present in time stamp data. This word contains the number of extra primary event triggers, occurred during the readout cycle of one event. These events will be missed by the FEB, and knowledge of their number allows the user to measure the current acquisition inefficiency.
Once the time stamp for the event is latched in the internal FPGA register, the CPU initiates the transmission of the data via a SPI interface.

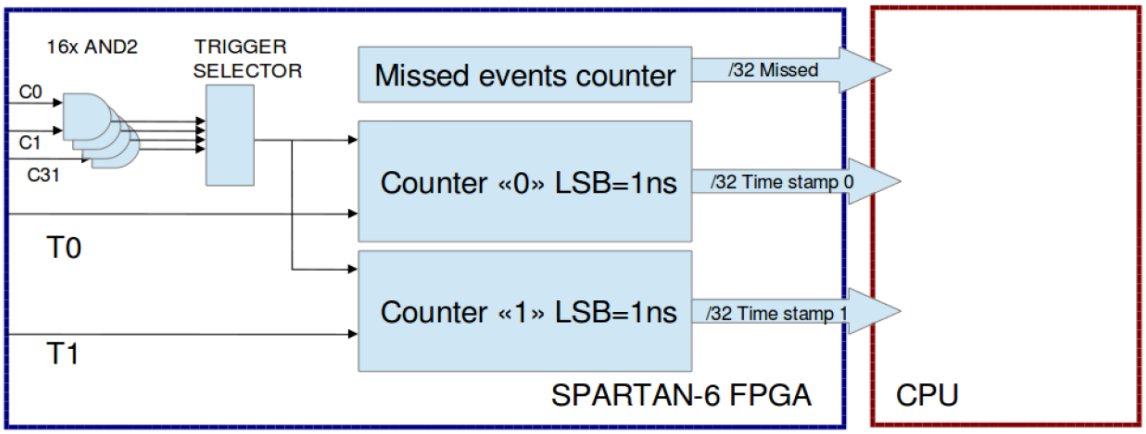


Figure 5.7: Block-scheme of time stamp generation circuit

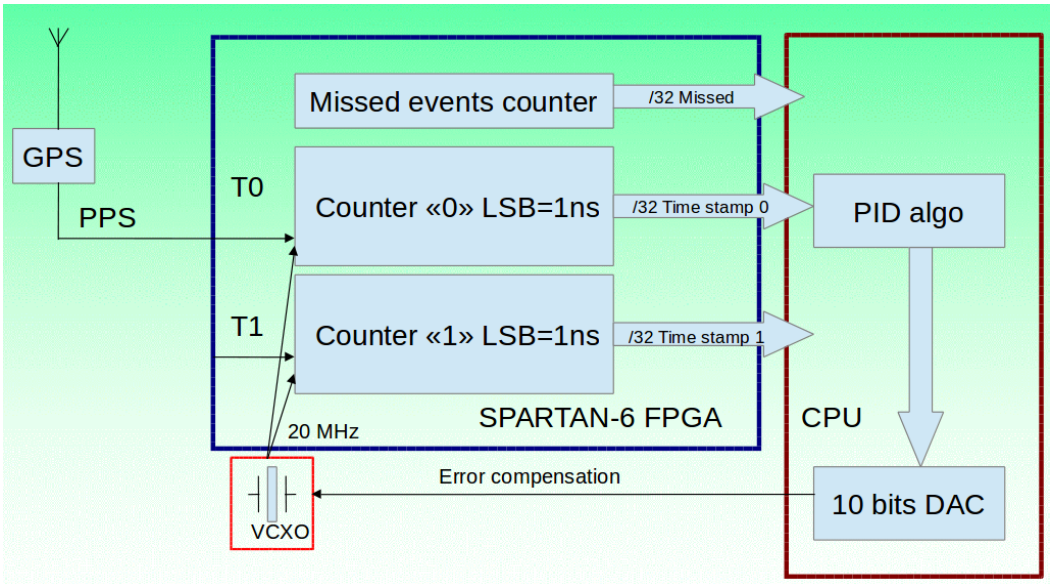


Figure 5.8: Block-scheme of the oscillator control loop for the time stamp generation circuit

Event buffer and back-end Ethernet interface

Once the CPU completes the digitization of 32 analog channels and transmission of the time stamp from the FPGA, it combines this data into an event and stores it in the internal ring buffer, with the capacity of 1024 events. Each event is 76 bytes long . The on-board micro-controller contains three CPU cores, each working at 160 MHz clock frequency. One core is taking care of filling the ring event buffer, while the second one is emptying it sending data out via an on-board Ethernet switch, when it is requested to do so by the host PC. If the incoming data rate exceeds the capacity of the back-end interface, the events are overwritten in the ring buffer. This situation is detected by the CPU and the number of overwritten (and therefore lost) events is stored, to be transmitted to host PC. This number, together with the "Missed event" counter in FPGA, contributes to the acquisition inefficiency of the FEB.
The structure of the event in buffer is shown below:

```
typedef struct{
  uint32_t flags;
  uint32_t T0;
  uint32_t T1;
  uint16_t adc[32];
}
```

The "flags" field contains in its lowest two bytes the number of overwritten event in the buffer, at the moment of the acquisition of the current event. If no events are lost in the buffer, it is equal to zero. In the two high bytes, the content of missed event in the FPGA is contained. In order to obtain the total lost events number, the user application should sum up numbers for missed counter for each event, and add the number of overwritten events from the last event in the buffer.
The “T0” and “T1” fields contain the timestamps measured by “counter 0” and “counter 1” respectively.
The “adc[32]” array contains the ADC data of all 32 channels.

The second CPU core communicates with the host computer via three-port Ethernet switch IC. The switch to the CPU interface (port 3) is 25 MHz 4-bits MII bus with throughput of 100 Mbps. The other two ports (1 and 2) of the switch have Physical Layer (PHY) transmitters and they are connected to two RJ45 Ethernet jacks. The switch is configured to forward ports 1 and 2 in both directions, to accept control commands on these ports, and to forward these commands via port 3 to the on-board CPU (see **Figure 5.9**). The event data from the CPU is forwarded only to that port from which the FEB has received data transmission request.

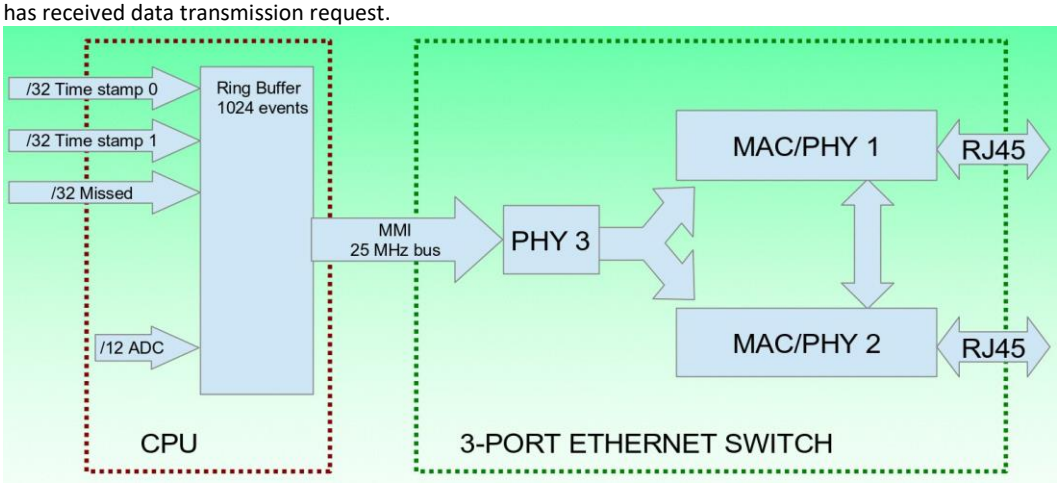


Figure 5.9: Block-scheme of the back-end data transmission and control interface

Both PHY ports of the switch have MDI/MDIX Auto Cross functionality, therefore the user does not need to care about the structure of the connecting Ethernet cables (straight or cross-wired). The switch detects the type of the cable at the connection instance, and sets its configuration accordingly.
The back-end host interface is realized on the base of 100 Mbit Ethernet. The on-board Ethernet switch allows to route data stream from one RJ45 jack to the other without delay, and add data flow from the FEB to this stream. This allows to daisy chain multiple boards with inexpensive CAT5 copper cables and read out this chain with one host computer.

The MAC address of each board is set by firmware to 00:60:37:12:34:XX, where last XX byte (MAC[5], also named “mac5” in the software interface) can be set from the hardware 8-bits switch array (see **Figure 5.10**). The value set at the switch is read only once at the CPU reset.

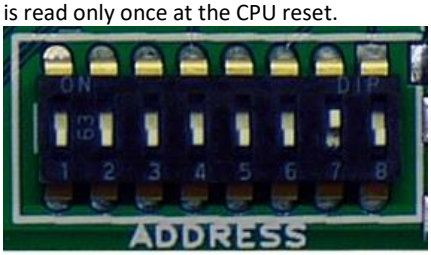


Figure 5.10: the 8-bits MAC address switch array. The configuration shown in the picture [01000000] corresponds to MAC[5] = 0X40.

The board accept the packets with the destination MAC address matching its own address. It also accepts Multicast packets with XX byte set to 0xFF. The board will ignore any other packets.

In **Figure 5.11,** an example of communication scheme between several FEBs and a host PC is shown. FEBs are daisy-chained and connected to a single Ethernet port of the host PC. Although such configuration can be a part of a more extended network, it is strongly recommended to have it isolated and to connect FEBs daisy chain to a dedicated Ethernet port, to maximize performance and the usage of the link throughput.

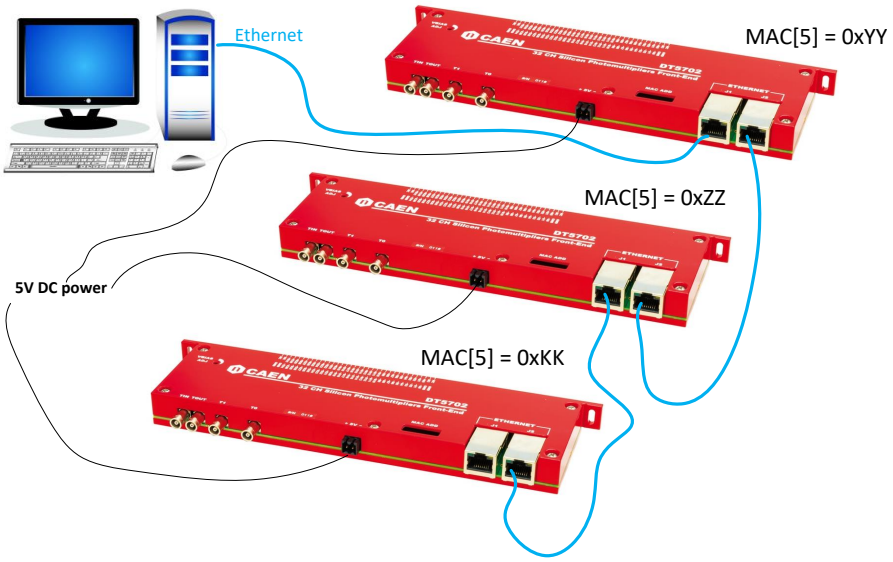


Figure 5.11: Scheme of the FEBs daisy chain and connection to the host computer. Note that the MAC[5] byte is set differently for each board.

Connectors and jumpers

External connectors are shown in **Figure 5.12**.

- **J1** and **J2** are RJ45 Ethernet sockets for connection by CAT5 twisted pair copper cable. They are equivalent to each other in terms of functionality.
- **MAC[5]** switch block is used to set the last XX byte of the board MAC address: 00:60:37:12:34:XX.
- **Power socket** feeds +5V DC power to the board. The range of the supply is from 4.5V to 5.5V. The current consumption varies depending on the event rate up to a maximum of 510 mA.
- **J3** and **J4** LEMO coaxial sockets are inputs for two timing reference pulses T0 and T1. The input logic standard is 3.3V LVCMOS. By default, they are not terminated; the termination resistors (R137 and R138) can be mounted to their corresponding footprints at the top side of the board between these connectors.
- **J5** and **J6** LEMO coaxial sockets are "TOUT" output pulse and "TIN" validation input respectively. "TOUT" is 3.3V LVCMOS logic signal, fed via 25 Ohms serial resistor. "TIN" input can be configured to 3.3V or 1.2V LVCMOS logic by mounting one of the shortcut resistors at the backside of the board (R131 and R132). As default, R131 is mounted for 1.2V LVCMOS. This input has a weak pull-up resistor on-board to allow operation without external signal supplied.
- **X1** is a 72-pins connector for connecting 32 SiPMs to the board. The pinout of this connector is shown in **Figure 5.13** as the view from the bottom of **Figure 5.12**. "NC" stands for Not Connected, "GND" pins are connected to common GND plane of the board, 32 "+B" and "-S" signals are positive bias and signal lines respectively. "+B" lines must be connected to cathodes of the SiPMs and "-S" line to anodes.

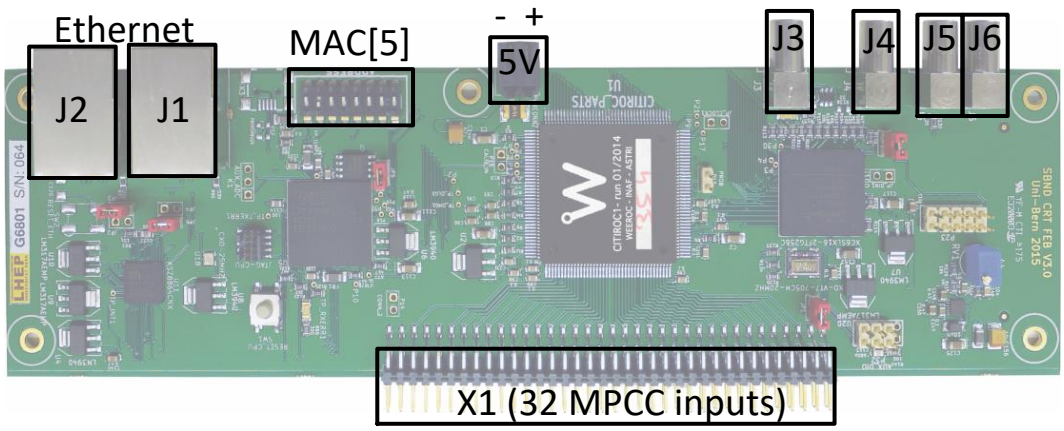


Figure 5.12: External connectors

GND	GND	-S0	-S1	-S2	-S3	-S4	-S5	-S6	-S7	-S8	-S9	-S10	-S11	-S12	-S13	-S14	-S15	-S16	-S17	-S18	-S19	-S20	-S21	-S22	-S23	-S24	-S25	-S26	-S27	-S28	-S29	-S30	-S31	GND	GND
NC	NC	+B0	+B1	+B2	+B3	+B4	+B5	+B6	+B7	+B8	+B9	+B10	+B11	+B12	+B13	+B14	+B15	+B16	+B17	+B18	+B19	+B20	+B21	+B22	+B23	+B24	+B25	+B26	+B27	+B28	+B29	+B30	+B31	NC	NC

Figure 5.13: Pinout of the X1 connector

The board hosts several jumpers for correct functionality (see **Figure 5.14**):

- **JP1** connects the output of the bias power supply to the +B lines of the X1 MPPC connector.
- **RV1** is a multi-turn trimmer, which allows adjustment of the positive bias voltage in the range of 20V to 90V.
- **JP2** to **JP4** define the start-up configuration and the control interface of the Ethernet switch.
- **JP3** and **JP5** must be closed for correct operation under control of the on-board CPU.
- **JP6** sets the booting mode of the LPC4370 micro-controller. Closed JP6 allows booting from the on-board ROM with pre-programmed firmware.
- **JP-CS1** sets the booting mode of the FPGA. Closed JP-CS1 configures the FPGA to load configuration file from its pre-programmed ROM.

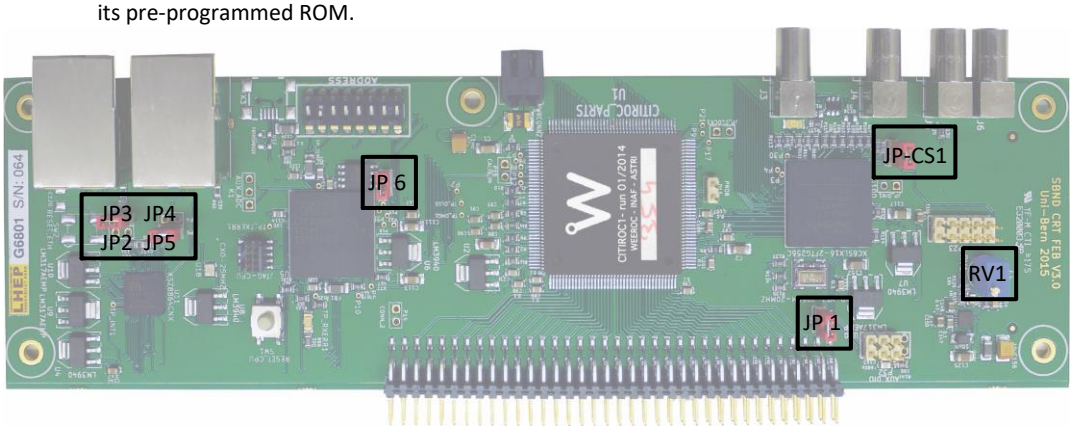


Figure 5.14: Jumpers and bias regulator

The CPU reset button **B1** and auxiliary connectors are located on the top side of the board (see **Figure 5.15**):

- **X2** connector allows monitoring the common bias voltage on MPPCs with external voltmeter.
- **X3** connector exposes PROBE outputs (analog and digital) of the CITIROC chip. By setting corresponding bits in the configuration bit stream, several internal ASIC signals can be multiplexed to this output and observed with an oscilloscope (refer to the Weeroc CITIROC datasheet for more details). The bit stream to set PROBE outputs is contained in the *"CITIROC_PROBEbitstream.txt"* file.
- **X4** connector allows to send an external calibration pulse to the CITIROC analog calibration input (refer to the Weeroc CITIROC datasheet).
- **X5** connector hosts 6 pins connected to 4 spare FPGA I/O pins, GND plane and +3.3V output of on-board stabilizer, that supplies power to the FPGA (see **Figure 5.16**).
- **XA** 3-pin connector hosts two spare ADC inputs and the ground (middle pin).

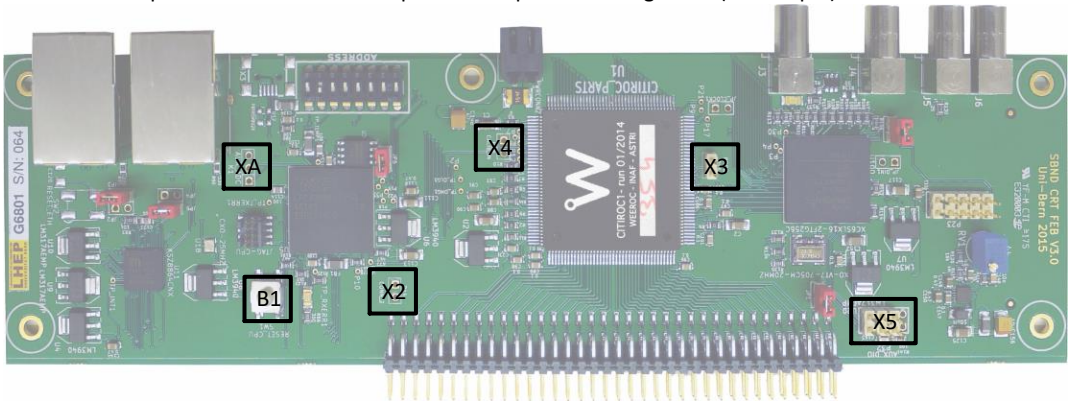


Figure 5.15: Auxiliary connections and CPU reset button

+3.3V	IO_L2P_0	IO_L2N_0
GND	IO_L6N_0	IO_L5N_0

Figure 5.16: Pinout of the auxiliary connector X5

Two 10-pin connectors **JT1** and **JT2** are used to program the board firmware via JTAG protocol (see **Figure 5.17**):

- **JT1** connector (JTAG-CPU on the PCB serigraphy) is used to upload program to LPC4370 micro-controller or its dedicated FLASH PROM. *Programming of the micro-controller can be done from the LPCxpresso development tool via supported JTAG programmers*.
- **JT2** connector (P23 on the PCB serigraphy) provides similar functionality for the XILINX Spartan-6 FPGA. In particular, this connector can be used to upgrade the FPGA firmware. Pinouts of the connectors is shown in **Figure 5.18.**

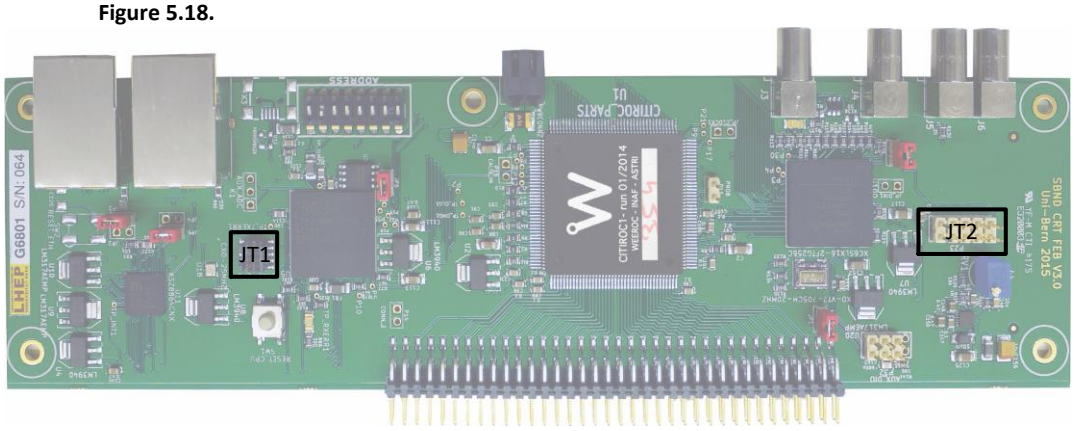


Figure 5.17: JTAG pin headers for firmware programming

+3.3V	TMS
GND	TCK
GND	TDO
NC	TDI
GND	RESET

GND	NC	GND	GND	+3.3V
NC	TDI	TDO	TCK	TMS

Figure 5.18: Pinout of JTAG pin headers JT1 (on the left) and JT2 (on the right)

Back-end communication and FEBDTP v3.0 data transmission protocol

The CAEN A1702/DT5702 FEB uses proprietary data transmission protocol (called *FEBDTP*) based on L2/MAC Ethernet abstraction layer, with minimized overhead, fast, and deterministic data flow. The data transmission from each board starts only after the board is interrogated by the host PC. The board then transmits its data buffer (up to 1024 events) within limited time slot. Once transmission is over, the host PC interrogates the next board in the chain. Such logic excludes packet collisions on the line and allows having deterministic data transfer without random delays, as in case of standard Ethernet.

General structure of the FEBDTP datagram is shown in **Table 5.1** First two fields define destination and source MAC addresses. ID field (2 bytes) is fixed and it represents a signature of FEBDTP protocol. Command field defines the functionality of the datagram. The 2-bytes register field defines a sub-address within the FEB, to which the command is related. For instance, this can be the index of the configuration register of the Ethernet switch IC. This field is followed by the variable size data payload. The size of this latter field is limited by a minimum and a maximum allowed datagram length (46 to 1482 bytes).

Communication is always initiated by the host PC by addressing a command datagram to an individual FEB (full destination MAC address), or by addressing datagram to all FEBs in a chain (Multicast datagram with last byte of MAC address set to 0xFF). The addressed FEB replies to the received command with acknowledgement datagram, informing the host computer about success or failure of the requested operation. The register and payload fields of the reply may bring to the host computer the requested data. One FEB sends a single reply datagram in response to a single command datagram received from the host computer. The only exception to this rule is the event buffer data request command FEB-RD-CDR.

Field	Destination MAC	Source MAC	ID	Command	Register	Data payload
Length, bytes	6	6	2	2	2	46 to 1482
Bytes (HEX)	00:11:22:33:44:55	00:60:37:12:34:85	08:01	01:00	00:00	xx:xx:xx

Table 5.1: FEBDTP V3.0 datagram structure and example. The total length of the datagram is in the range 64 to 1500 bytes.

This command triggers the FEB to initiate transmission of its internal event buffer content to the host computer. The whole transmission may consist of up to 54 full-length datagrams, send by the FEB to the host, one by one, with minimum pause between them. The transmission end is marked by a special datagram.

The event data payload must have a size in the range 46 to 1482 bytes. Therefore, the maximum number of events that fit into the datagram is 19, each 76-bytes long. This results in a 1462-bytes long datagram. As mentioned above, the full buffer is transmitted by 54 datagrams.

A detailed description of FEBDTP V3.0 commands is given in the following tables. The "xx" notation means a byte with no usage. The first line describes the command that the HOST sends to the FEB. The following lines are alternative answers that the HOST expects from the FEB, within configurable timeout.

The following commands are used by the host computer to *control the ETHERNET switch* of the board:

- FEB-RD-SR
- FEB-WR-SR
- FEB-RD-SRFF
- FEB-WR-SRFF

Description of each command is given in the correspondent table caption.

Direction	Command mnemonic	Command bytes	Register	Payload	Datagram Length
HOST → FEB	FEB-RD-SR	00 01	REG	xx.xx.xx	64 bytes
FEB → HOST	FEB-OK-SR	00 00	xx	DD.xx.xx	64 bytes
FEB → HOST	FEB-ERR-SR	00 FF	00 00	xx xx xx	64 bytes

Table 5.2: Command to read ETHERNET switch control/status register. REG is the register address to read. The answer of the FEB is contained in the FEB-OK-SR command payload.

Direction	Command mnemonic	Command bytes	Register	Payload	Datagram Length
HOST → FEB	FEB-WR-SR	00 02	REG	DD.xx.xx	64 bytes
FEB → HOST	FEB-OK-SR	00 00	xx	DD.xx.xx	64 bytes
FEB → HOST	FEB-ERR-SR	00 FF	00 00	xx xx xx	64 bytes

Table 5.3: Command to write ETHERNET switch control/status register. REG is the register address to be written. The content to be written is contained in the FEB-WR-SR command payload.

Direction	Command mnemonic	Command bytes	Register	Payload	Datagram Length
HOST → FEB	FEB-RD-SRFF	00 03	xx	xx.xx.xx	64 bytes
FEB → HOST	FEB-OK-SR	00 00	xx	256 bytes	274 bytes
FEB → HOST	FEB-ERR-SR	00 FF	00 00	xx xx xx	64 bytes

Table 5.4: Command to read ETHERNET switch control/status, 256 registers at once. The answer of the FEB is contained in the FEB-OK-SR command payload.

Direction	Command mnemonic	Command bytes	Register	Payload	Datagram Length
HOST → FEB	FEB-WR-SRFF	00 04	xx	256 bytes	274 bytes
FEB → HOST	FEB-OK-SR	00 00	xx	256 bytes	274 bytes
FEB → HOST	FEB-ERR-SR	00 FF	00 00	xx xx xx	64 bytes

Table 5.5: Command to write ETHERNET switch control/status, 256 registers at once. The content to be written is contained in the FEB-WR-SRFF command payload.

The following commands of the FEBDTP protocol are used by the host computer as *board configuration functions and data acquisition functions*:

- FEB-SET-RECV
- FEB-GEN-INIT
- FEB-GEN-HVON
- FEB-GEN-HVOF
- FEB-GET-RATE
- FEB-RD-SCR
- FEB-WR-SCR
- FEB-RD-CDR
- FEB-RD-PMR
- FEB-WR-PMR
- FEB-RD-FIL
- FEB-WR-FIL

Each command is described in the correspondent table caption.

Direction	Command mnemonic	Command bytes	Register	Payload	Datagram Length
HOST → FEB	FEB-SET-RECV	01 01	RR RR	6 bytes MAC	64 bytes
FEB → HOST	FEB-OK	01 00	00 00	6 bytes	64 bytes
FEB → HOST	FEB-ERR	01 FF	00 00	00 00 00	64 bytes

Table 5.6: Command to broadcast the MAC of the host to all FEBs and to collect MACs of all FEBs into the host's client DB. It returns the firmware version string in FEB_OK data field. If RR RR >0, it sets the VCXO correction value to RR RR

Direction	Command mnemonic	Command bytes	Register	Payload	Datagram Length
HOST → FEB	FEB-GEN-INIT	01 02	RR RR	xx xx xx	64 bytes
FEB → HOST	FEB-OK	01 00	00 00	xx xx xx	64 bytes
FEB → HOST	FEB-ERR	01 FF	00 00	00 00 00	64 bytes

Table 5.7: Acquisition control command.
RR RR = 00 00 disables acquisition.
RR RR = 01 01 resets data buffer.
RR RR = 02 02 enables acquisition.
RR RR = FF FF resets CPU with WatchDog.

Direction	Command mnemonic	Command bytes	Register	Payload	Datagram Length
HOST → FEB	FEB-GEN-HVON	01 03	00 00	xx xx xx	64 bytes
FEB → HOST	FEB-OK	01 00	00 00	xx xx xx	64 bytes
FEB → HOST	FEB-ERR	01 FF	00 00	00 00 00	64 bytes

Table 5.8: Command to turn SiPMs HV bias ON

Direction	Command mnemonic	Command bytes	Register	Payload	Datagram Length
HOST → FEB	FEB-GEN-HVOF	01 04	00 00	xx xx xx	64 bytes
FEB → HOST	FEB-OK	01 00	00 00	xx xx xx	64 bytes
FEB → HOST	FEB-ERR	01 FF	00 00	00 00 00	64 bytes

Table 5.9: Command to turn SiPMs HV bias OFF

Direction	Command mnemonic	Command bytes	Register	Payload	Datagram Length
HOST → FEB	FEB-GET-RATE	01 05	00.00	xx xx xx	64 bytes
FEB → HOST	FEB-OK	01 00	00 00	DD DD DD DD	64 bytes
FEB → HOST	FEB-ERR	01 FF	00 00	00 00 00	64 bytes

Table 5.10: This command requests current trigger rate from the FEB, returned as float in the first 4 bytes of Data field of FEB_OK reply

Direction	Command mnemonic	Command bytes	Register	Payload	Datagram Length
HOST → FEB	FEB-RD-SCR	02 01	6 bytes register	xx xx xx	64 bytes
FEB → HOST	FEB-OK-SCR	02 00	00 00	143 bytes Control string	161 bytes
FEB → HOST	FEB-ERR-SCR	02 FF	00 00	xx xx xx	64 bytes

Table 5.11: Command to read CITIROC Slow Control register (SR)

Direction	Command mnemonic	Command bytes	Register	Payload	Datagram Length
HOST → FEB	FEB-WR-SCR	02 02	6 bytes register	143 bytes Control string	161 bytes
FEB → HOST	FEB-OK-SCR	02 00	00 00	143 bytes Control string	161 bytes
FEB → HOST	FEB-ERR-SCR	02 FF	00 00	xx xx xx	64 bytes

Table 5.12: Command to write and latch CITIROC Slow Control register (SR)

Direction	Command mnemonic	Command bytes	Register	Payload	Datagram Length
HOST → FEB	FEB-RD-CDR	03 01	6 bytes register	xx xx xx	64 bytes
FEB → HOST	FEB-DATA-CDR	03 00	00 00	Event data buffer	64 to 1482 bytes
FEB → HOST	FEB-ERR-CDR	03 FF	00 00	xx xx xx	64 bytes
FEB → HOST	FEB-EOF-CDR	03 03	00 00	xx xx xx	64 bytes

Table 5.13: Command to read the event buffer. The FEB_DATA_CDR message is repeated till the FEB buffer is empty. The FEB_EOF_CDR command terminates the data buffer transmission

Direction	Command mnemonic	Command bytes	Register	Payload	Datagram Length
HOST → FEB	FEB-RD-PMR	04 01	6 bytes register	xx xx xx	64 bytes
FEB → HOST	FEB-OK-PMR	04 00	00 00	28 bytes Control string	64 bytes
FEB → HOST	FEB-ERR-PMR	04 FF	00 00	xx xx xx	64 bytes

Table 5.14: Command to read the CITIROC Probe register (PMR)

Direction	Command mnemonic	Command bytes	Register	Payload	Datagram Length
HOST → FEB	FEB-WR-PMR	04 02	6 bytes register	28 bytes Control string	64 bytes
FEB → HOST	FEB-OK-PMR	04 00	00 00	28 bytes Control string	64 bytes
FEB → HOST	FEB-ERR-PMR	04 FF	00 00	xx xx xx	64 bytes

Table 5.15: Command to write and latch in the CITIROC Probe register (PMR)

Direction	Command mnemonic	Command bytes	Register	Payload	Datagram Length
HOST → FEB	FEB-RD-FIL	06 01	6 bytes register	xx xx xx	64 bytes
FEB → HOST	FEB-OK-FIL	06 00	00 00	9 bytes Control string	64 bytes
FEB → HOST	FEB-ERR-FIL	06 FF	00 00	xx xx xx	64 bytes

Table 5.16: Command to read the FPGA input flexible logic register (FIL). The 9 bytes control string is formed by 4-bytes mask1, 4-bytes mask2 and a majority byte.

Direction	Command mnemonic	Command bytes	Register	Payload	Datagram Length
HOST → FEB	FEB-WR-FIL	06 02	6 bytes register	9 bytes Control string	64 bytes
FEB → HOST	FEB-OK-FIL	06 00	00 00	9 bytes Control string	64 bytes
FEB → HOST	FEB-ERR-FIL	06 FF	00 00	xx xx xx	64 bytes

Table 5.17: Command to write and latch in the FPGA input logic register (FIL)

The following commands are dedicated to the *Firmware In-Application programming*. In particular, two functions are available to read and write into the SPI Flash Interface:

- FEB-RD-FW
- FEB-WR-FW

When FEB receives FEB-RD-FW command, it starts to reply to the host with FEB-DATA-FW datagrams, containing in the payload the 1024 bytes block of the firmware PROM. Bytes The first three bytes of FEB-RD-FW payload define the starting address in PROM address space. The subsequent two bytes define the number of blocks requested (max. 64 per transaction). The register field of FEB-DATA-FW contains the CRC code of the transmitted block. The process continues until all requested blocks are transferred. The transfer is accomplished by FEB-EOF-FW, containing the CRC code of the completely transmitted data in its "register" field. The FEB replies with FEB-ERR-FW in case of CRC mismatch, otherwise FEB-OK-FW is returned.

Direction	Command mnemonic	Command bytes	Register	Payload	Datagram Length
HOST → FEB	FEB-RD-FW	05 01	00 00	A0 A1 A2 N0 N1. (+42 bytes xx)	64 bytes
FEB → HOST	FEB-DATA-FW	05 04	CC CC	1024 bytes Data block	64 to 1482 bytes
FEB → HOST	FEB-ERR-FW	05 FF	00 00	xx xx xx	64 bytes
FEB → HOST	FEB-EOF-FW	05 03	CC CC	xx xx xx	64 bytes
FEB → HOST	FEB-OK-FW	05 00	00 00	xx xx xx	64 bytes

Table 5.18: Command to read SPI Flash Interface content. Bytes A0-A2 define the starting address in PROM address space. Bytes N0-N1 define the number of requested blocks. The CC CC field contains the CRC code of the block.

A command to write a new firmware in the SPI Flash is also available. The FEB-WR-FW "register" field contains the writing options (see **Table 5.19**). The first three bytes of FEB-WR-FW payload define the starting address in PROM address space. The subsequent two bytes define the number of blocks requested (max. 64 per transaction). When FEB receives FEB-WR-FW command, it starts to reply to the host with FEB-DATA-FW datagrams, containing in the payload the 1024 bytes block of the firmware PROM bytes. The register field of FEB-DATA-FW contains the CRC code of the transmitted block. The process continues until all requested blocks are transferred. The transfer is accomplished by FEB-EOF-FW, containing the CRC code of the whole transmitted data in its "register" field. The FEB replies with FEB-ERR-FW in case of CRC mismatch, otherwise FEB-OK-FW is returned.

Direction	Command mnemonic	Command bytes	Register	Payload	Datagram Length
HOST → FEB	FEB-WR-FW	05 02	RR RR	A0 A1 A2 N0 N1. (+42 bytes xx)	64 bytes
FEB → HOST	FEB-DATA-FW	05 04	CC CC	1024 bytes Data block	64 to 1482 bytes
FEB → HOST	FEB-ERR-FW	05 FF	00 00	xx xx xx	64 bytes
FEB → HOST	FEB-EOF-FW	05 03	CC CC	xx xx xx	64 bytes
FEB → HOST	FEB-OK-FW	05 00	00 00	xx xx xx	64 bytes

Table 5.19: Command to write SPI Flash Interface content. Options are specified in RR RR field.
RR RR= 00 00 writes N0 N1 blocks to the flash and returns to main thread.
RR RR = 01 01 copies all N0 N1 blocks from A0 A1 A2 to 00 00 00 and performs CPU reset. The new firmware must become effective Bytes A0-A2 define the starting address in PROM address space. Bytes N0-N1 define the number of requested blocks. The CC CC field contains the CRC code of the block.

6 Test Software

Prerequisites

In this chapter, we describe the interface and usage of a software for data acquisition with CAEN A1702/DT5702.
A simple data acquisition program *"FEBDAQMULT"* with GUI is available on the CAEN website. This software can be used to test the performances of the FEB and the source code (available within the software package) may serve as a template for more dedicated experiment-optimized DAQ software.

The program is a CERN ROOT script, that uses the compiled library *FEBDTP.so*, containing API for communication with the board via FEBDTP protocol. The software is tested on a Linux OS (Ubuntu 16.04 – 64bit) machine with CERN ROOT version 6.10/02 installed on it from precompiled binary file, available on the CERN ROOT website. Please follow installation instructions for ROOT package from CERN web site.

Once ROOT is installed on your computer, you need to make ROOT executable from any directory, since the software "compile" file uses ROOT commands. To do this, check where is your local ROOT executable (for example /usr/local/bin/cern/root/bin/root).

From terminal, open the */.bashrc* file.

gedit ~/.bashrc

Write the following lines at the end of the file, according to your ROOT local directory:

source /usr/local/bin/cern/root/bin/thisroot.sh
alias root =/usr/local/bin/cern/root/bin/root

At this level, you are ready to compile and install the DAQ demo software.

Compiling and installation

Follow the steps below to install the DAQ demo software:

- Download the DAQ software package from the DT5702/A1702 page on CAEN web site.
- Unpack it to the desired directory. From terminal run the command

  tar -xvzf DAQ.tar.gz -C /path/to/target/directory

- Within the target directory, run

  ./compile

  The *FEBDTP.so* library should compile.

Now you are ready to run the DAQ software.

Usage

Connect 5V DC power to the board or a chain of boards (up to 256 units into one network interface). Connect an Ethernet port of the board (or the chain of boards) to a dedicated Ethernet port of your HOST PC (see **Figure 5.11**). *When using a single board it is unimportant which of the two Ethernet port of the board is used.*

Note: when connecting boards in daisy chain, a different MAC[5] byte must be set at the address physical switch of each board.

Check the name of your Ethernet port by running the on line command
ifconfig

```
yuri@yuri-VirtualBox:~$ ifconfig
enp0s3    Link encap:Ethernet  HWaddr 08:00:27:f1:d8:3b
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:32248 errors:0 dropped:38 overruns:0 frame:0
          TX packets:683 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3587112 (3.5 MB)  TX bytes:286191 (286.1 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:11700 errors:0 dropped:0 overruns:0 frame:0
          TX packets:11700 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:885391 (885.3 KB)  TX bytes:885391 (885.3 KB)
```

The name of the available Ethernet ports is listed. Check the name of the port to which the FEB is connected.

To execute the "FEBDAQMULT" program the user needs privileged access to Ethernet interface, the simplest way to get it is to run ROOT as superuser with *sudo* command. Assuming your interface is "enp0s3" as shown in the screenshot above, run the ROOT as superuser, giving as the argument the name of the DAQ script “FEBDAQMULT.C”:

sudo /usr/local/bin/cern/root/bin/root -l ‘FEBDAQMULT.C(“enp0s3”)’

If you are not giving any Ethernet port name, the script will assume by default “eth1”.

Software interface

At the start up, the script scans the MAC address space for connected FEBs, stores this table and opens the GUI.

In the following example (see **Figure 6.1**), only a FEB was connected and the table contains only one client. The MAC address of the FEB is read (in the following example it is 00:60:37:12:34:55, where the last byte is set by the board MAC[5] switch) together with the firmware revision (in this case FEB_rev3_FLX7.003). In **Table 6.1** are listed the latest firmware revision of the board.

Firmware release	Features
FLX7.003	Selectable even-odd adjacent channels coincidence and channels OR32
IAP7.007	Even-odd adjacent channels coincidence

Table 6.1: latest firmware revision name.

After the boards address table has been stored, the script performs the upload of the CITIROC configuration bit stream. The bit streams are stored in two files, which are present in the working directory:

- *"CITIROC_PROBEbitstream.txt"* contains the bit stream for the configuration of the CITIROC Probe Multiplexer. The user can set in this file the ASIC internal signals to be multiplexed at the analog probe X3 of the FEB. An example of this file is given in **Annexes.**
- *“CITIROC_SC_PROFILE1.txt"* contains the bit stream for the CITIROC Slow Control configuration. An example of this file is given in **Annexes.**

For description of each bit please refer to the CITIROC ASIC datasheet.

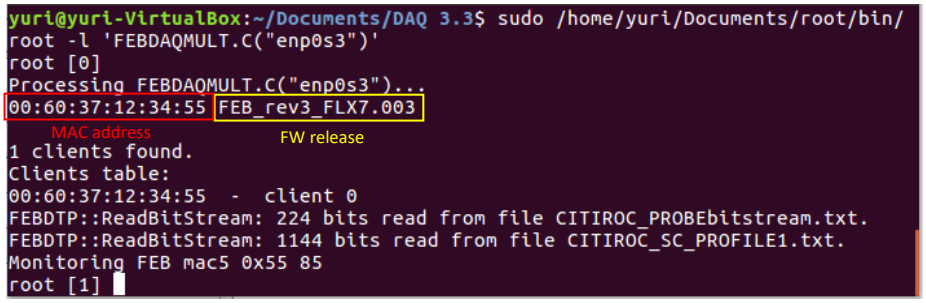


Figure 6.1: screenshot of the Ubuntu terminal at the DAQ software startup. The MAC address and the firmware release are highlighted in red and yellow respectively.

After the MAC address of FEBs has been read, the GUI opens.
The main page of the GUI contains the DAQ controls on the left, the tab panel on the right and the statistics bar at the bottom (see **Figure 6.2**).

The **DAQ FEB controls** menu contains the main DAQ configuration commands and parameters.

The **display area** visualizes the page selected by the tabs in the top bar. Clicking tabs, the user can toggle between the following display and control pages:

- "Configuration": CITIROC configuration page.
- "All histos": 32-channels histograms summary display.
- "One channel": single channel histogram display.
- "Timing data": timer data display.
- "Channel profile": 32-channels event profile. The integrated amplitude (expressed in ADC) of the events recorded by each channel is shown.
- "Event rate": trigger rate display.
- "Event display": rainfall-style event display.

The **statistics bar,** at the bottom of the GUI, contains information about the data acquisition (for example, the trigger rate and the lost events)

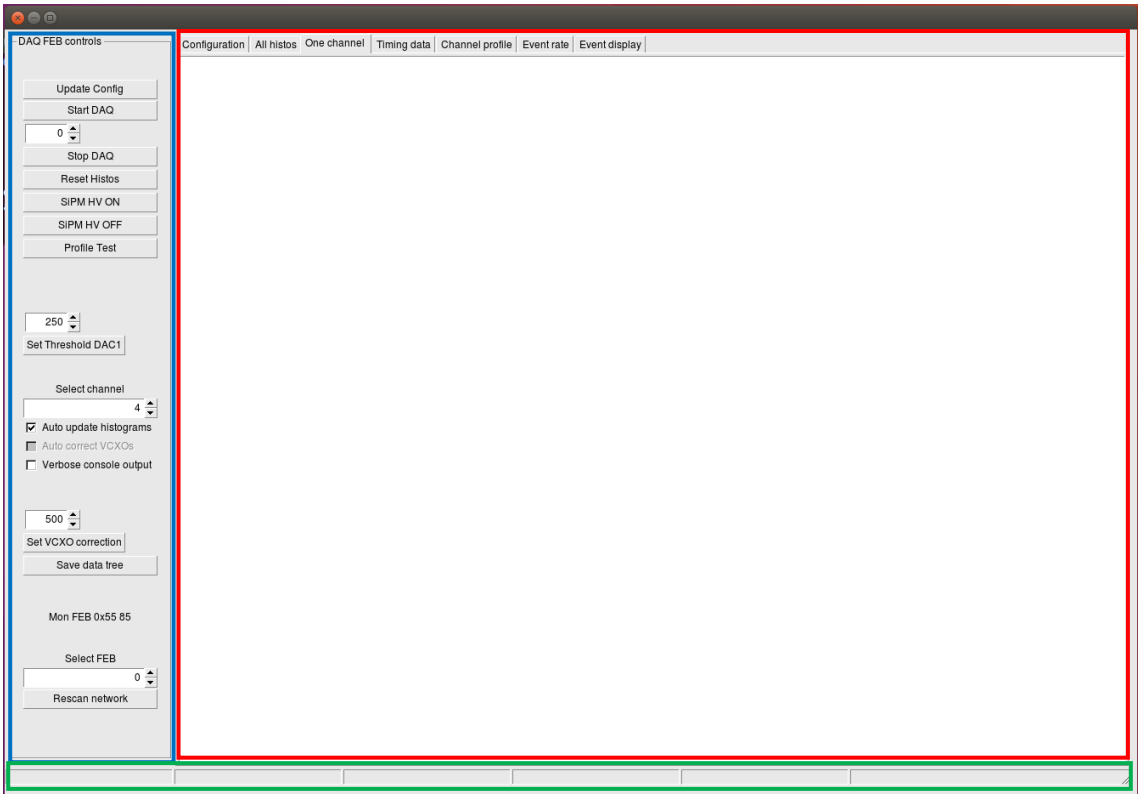


Figure 6.2: main page of the DAQ software GUI. Its main parts are highlighted: the DAQ controls menu in blue, the display area in red and the statistics bar in green.

DAQ FEB controls

The DAQ FEB controls menu provides the possibility to send commands to the board and to configure several parameters of the data acquisition. In the following a description of available functions is given.

- ***"Update Config"*** button allows to re-read the configuration files *"CITIROC_PROBEbitstream.txt"* and *"CITIROC_SC_PROFILE1.txt"* and to re-upload the bit streams to all FEBs at any time. Once the button has been pressed, eventual changes to DAQ parameters applied via the GUI will be lost.
- ***"Start DAQ"*** button performs reset of the FEBs with the "FEB-GEN-INIT" FEBDTP command and starts the acquisition process. If the check button ***"Auto update histograms"*** is checked, the histograms are updated after each data packet received from selected FEB. The FEB to be monitored can be selected at the bottom of the control panel.
- Below the "Start DAQ" button, the numerical entry field defines **time in seconds** for which DAQ should run. Setting it to 0 makes it run continuously until stopped by "Stop DAQ" button.
- ***"Stop DAQ"*** button stops the acquisition cycle.
- ***"Reset Histos"*** button resets all histograms and the data tree generated by the program.

- ***"SiPM HV ON"*** and ***"SiPM HV OFF"*** toggle activation of the SiPM bias power supplies on all connected boards.
- ***"Profile test"*** button initiates the following sequence: re-scan the network to update client list, configure connected FEBs uploading the CITIROC configuration files (changes applied in the GUI are lost), turn SiPM bias ON, start DAQ for the time period given above, turn bias OFF.
- ***"Set Threshold DAC1"*** button sets the value of the CITIROC discriminator threshold according to the corresponding numerical entry field (above the button). This threshold is common for all channels (see **CITIROC Configuration files** for individual channel threshold adjustment). Typically, the value for the threshold at 0.5 p.e would be 210 to 215. Setting this value to 200 and below will likely cause unstable behaviour of CITIROC ASIC, which can be diagnosed by quick rising of the trigger rate, displayed in the left bottom corner of the GUI window. Raising this rate above 30 kHz will lead to massive event losses and, sometimes, corrupted data.
- The "***Select channel"*** numerical entry allows choosing the channel histogram to be displayed in the "One channel" tab.
- The ***"Verbose Console Output"***, if checked, allows the user to display on the terminal messages related to the commands send to the FEBs and data packets read by the host computer.
- The correction value for VCXO oscillator can be overridden manually via ***"Set VCXO correction"*** button and the related entry field. The ***"Auto-correct VCXOs"*** check box is deactivated for FEB V3.0 since the control loop is always active on-board. However, the loop is operative only when a stable PPS pulse is supplied to "T1" LEMO inputs of the FEB (refer to **Time stamp generator**). The correction is calculated basing on 20 consecutive PPS period measurements, therefore the VCXO runs with default settings for the first 20 seconds from the DAQ start up. Currently measured periods for both timers are indicated in the status string at the bottom of the GUI window *("PPS period"* and *"SPILL trig period"*).
- ***"Save data tree"*** button allows saving the ROOT TTree object into a file, named by default "mppc.root". The file is created in the working directory and can be overwritten. The data tree contains one entry per event and has the following structure:

  ```
  tr->Branch("mac5",&mac5,"mac5/b");
  tr->Branch("chg",chg,"chg[32]/s");
  tr->Branch("ts0",&ts0,"ts0/i");
  tr->Branch("ts1",&ts1,"ts1/i");
  tr->Branch("ts0_ref",&ts0_ref,"ts0_ref/i");
  tr->Branch("ts1_ref",&ts1_ref,"ts1_ref/i");
  ```

  The "mac5" variable contains the higher byte of the FEB mac address from which the event is received (i.e. the byte set by the MAC[5] switch of the board).

  The "chg[32]" array contains ADC data for all 32 channels read from the given FEB.

  "ts0" and "ts1" represent the time stamps generated by the two internal timers.

  "ts0_ref" and "ts1_ref" contain the values of the recorded period between the latest received reference pulse and the previous one, on T0 and T1 LEMO respectively. This value can be used for correction of time stamps for a possible residual drift of the on-board VCXO. This allows to improve absolute accuracy of time stamps to better than 1 ns on 1 s interval, when using the "coincidence" firmware.

  The data saved in the *"mppc.root" file can be displayed as histograms* running in the terminal the ROOT command

  ```
  new TBrowser()
  ```

  and selecting the desired file. A histogram for each variable in the data tree is displayed.
- ***"Select FEB"*** allows selecting the desired client saved in the FEB address table. The FEB that is monitored by the software is indicated in the string ***"Mon FEB 0xAA BB"***, where AA is the value of the higher byte of its MAC address while BB is the correspondent decimal value.
- ***"Rescan Network"*** button allows to rebuild the clients table and to re-upload the CITIROC configuration files for all connected FEBs.

Display Area

The Display Area visualizes the correspondent page selected in the tab bar above. There are 6 display pages and a configuration page available.

In the ***"Configuration"*** tab (see **Figure 6.3**), the user has the possibility to directly control and modify several bits of the CITIROC Slow Control Register interactively. Changes are transmitted to all FEBs right after clicking on any check- or radio-button. Modified configuration is not saved to file and is lost after exiting the program, or after clicking *"Update Config"*, *"Profile Test"* or *"Rescan Network"*.

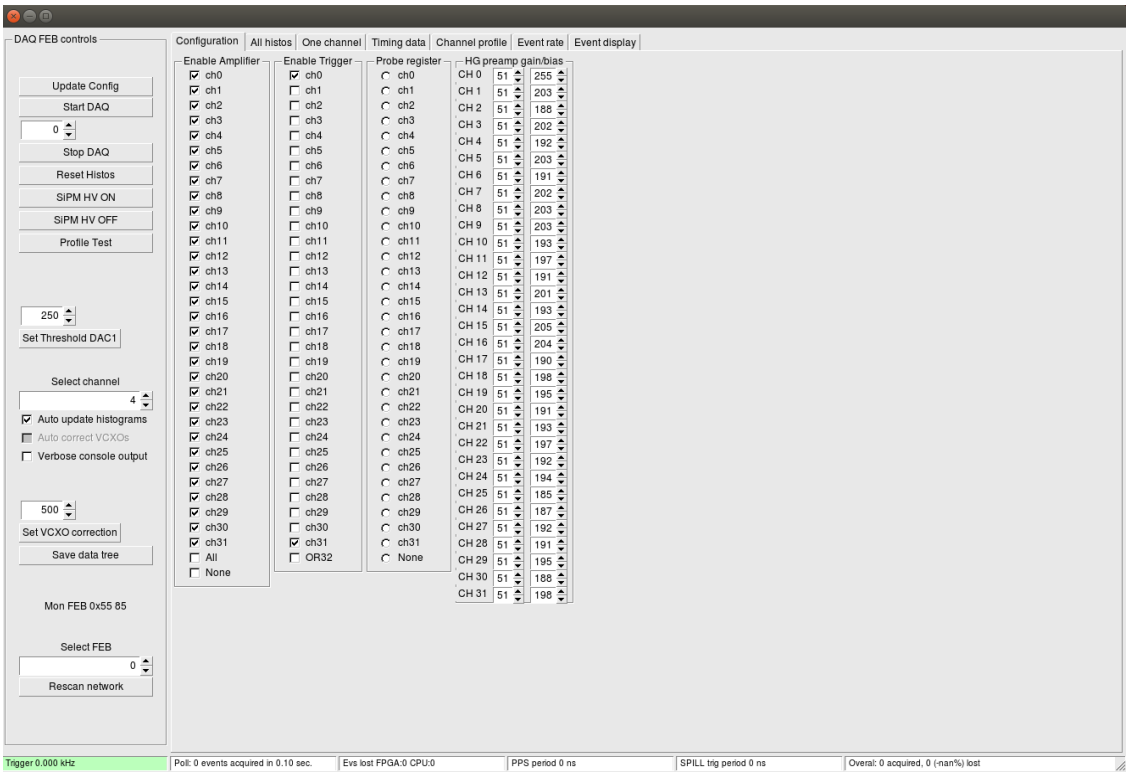


Figure 6.3: "Configuration" tab in the GUI to control several bits of the CITIROC Slow Control register

- The ***"Enable Amplifier"*** column allows enabling/disabling the amplifier for each CITIROC channel independently. The option "All" and "None" are also available. The amplifier must be active for those channels for which DAQ of a signal is wanted.

- The ***"Enable Trigger"*** column allows enabling/disabling the channels on which the system will trigger. When more channels are selected, the board performs coincidence of active channels according to the behavior described in **Board firmware** paragraph. Using the firmware rel. FLX7.003, if "OR32" is checked, the software performs the logic OR of the enabled channels individual triggers.

- The ***"Probe register"*** column allows selecting a channel of the CITIROC for which its "high-gain slow shaper" trace will be output at the X3 connector. Only a channel is selectable. The other CITIROC internal traces to be output at X3, can be set directly in the *"CITIROC_PROBEbitstream.txt"* file.

Note: this probe system is usually used for debugging. In typical ASIC operation, it is advised not to set any probe output, in order to avoid analog performances degradation**.**

- The ***"HG preamp gain"*** column allows to set the High Gain preamplification factor for all CITIROC channels.

- The ***"bias"*** column allows to set the SiPMs bias fine regulation using the internal 8-bits DAC. The numerical entry sets the decimal DAC value for each channel. Therefore extreme values are:
  255 = 11111111 = DAC maximum output voltage
  0 =00000000 = DAC minimum output voltage.
  The DAC's positive output levels are supplied to the signal lines as DC-offset, therefore increasing this voltage reduces effective bias for individual SiPMs.
  The full DAC range is +0.5 to +4.5 V, however a smaller DAC range can be set in the *"CITIROC_SC_PROFILE1.txt".* The correspondent bit description, as reported in the configuration file, is given below for reference:

  8-bit input DAC Voltage Reference (1=external 4.5V, 0=internal 2.5V)

The ***"All histos"*** tab displays the histograms of the 32 channels of the monitored FEB (see **Figure 6.4**). By default the Y-scale is set to log-scale for all plots. If the *"Auto update histograms"* option is checked, all histograms update during DAQ.

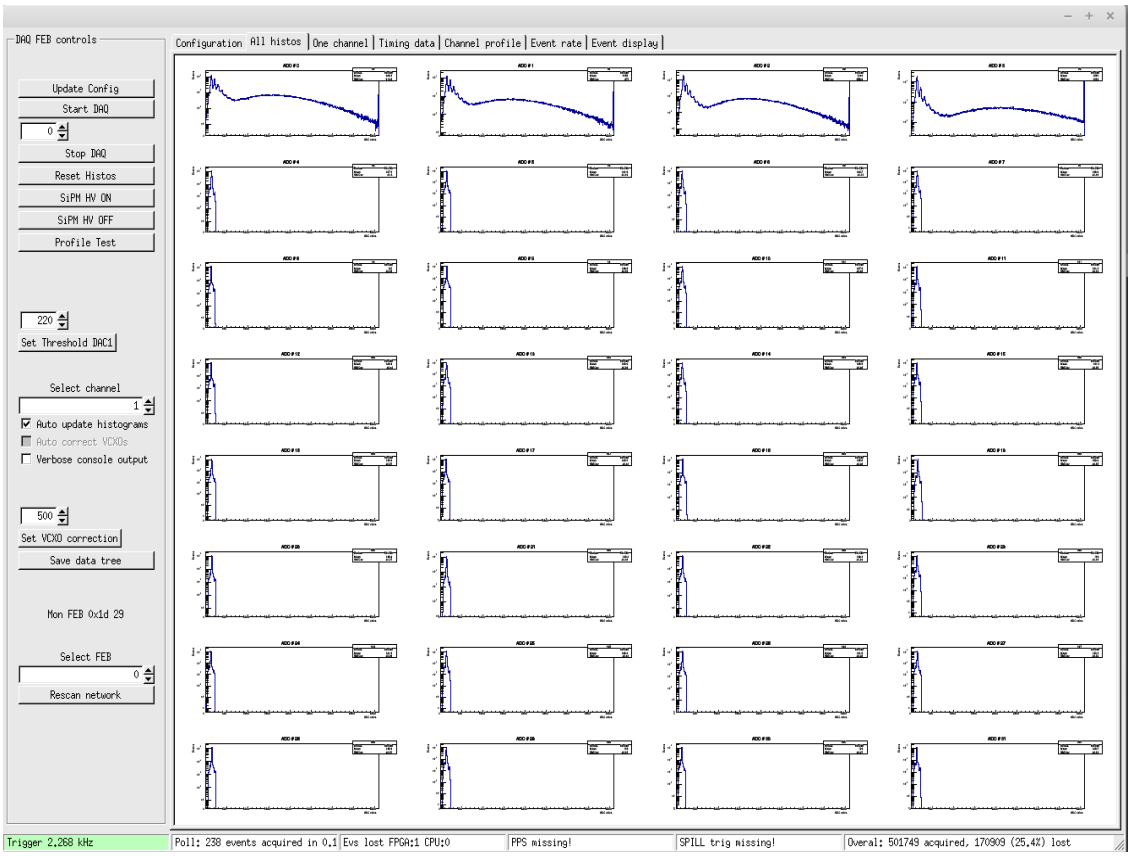


Figure 6.4: the "All histos" tab during acquisition. Here the FPGA firmware rel FLX7.003 was used and only channels 0,1,2,3 where enabled to trigger in coincidence. Non-triggering channels are displaying their characteristic noise peak. See DAQ using DT5702 for more details.

The ***"One channel"*** tab displays the histogram of one channel of the monitored FEB (see **Figure 6.5**Figure 6.4). The channel histogram to be displayed is set by the *"Select channel"* FEB control. By default, the Y-scale is set to linear-scale. If the *"Auto update histograms"* option is checked, the histogram updates during DAQ. The monitored channel is clearly indicated in the histogram title and some other information (total number of entries, mean and standard deviation of ADC values) are shown in top right part of the display area.

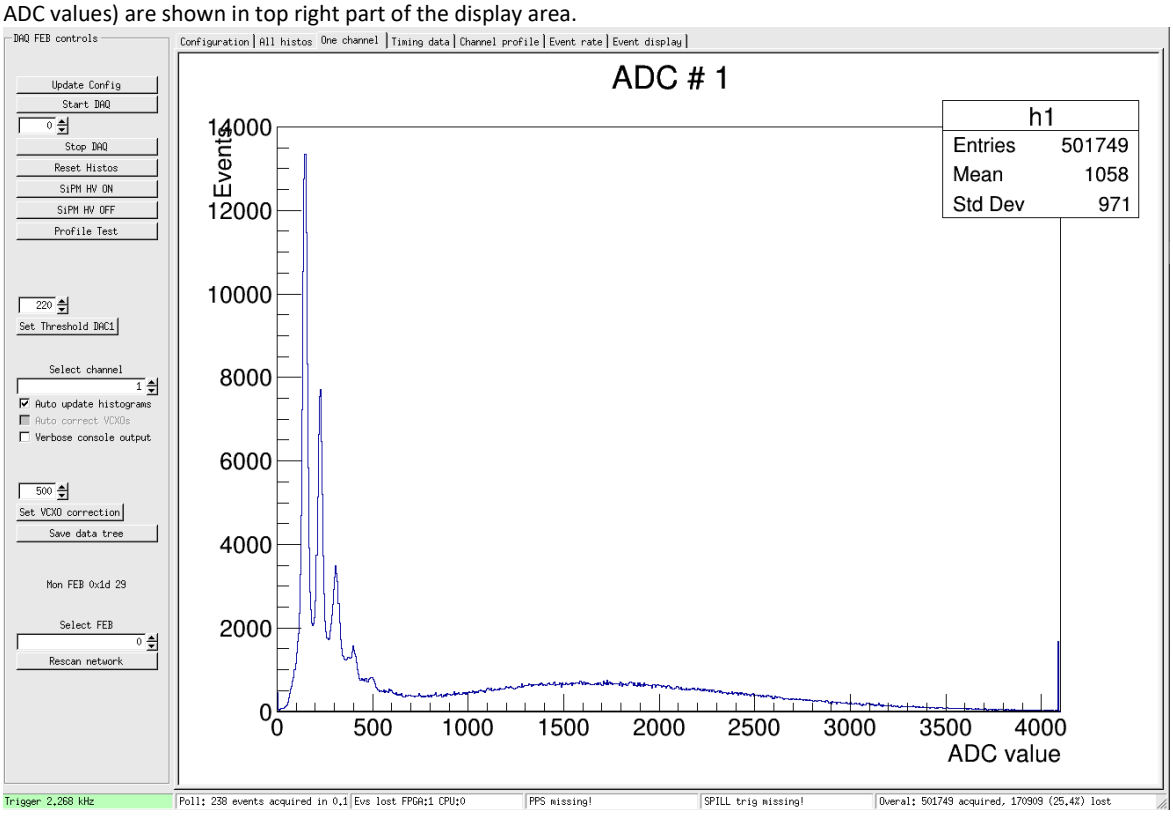


Figure 6.5: the *"One channel"* tab showing the histogram of channel 1 during a DAQ. Here typical peaks generated by a SiPM are shown.

The ***"Timing data"*** tab displays two plots related to the board timing (see **Figure 6.6**). In case a PPS timer is connected to the T0 LEMO input, the top plot shows the stability of the period of the PPS signal. The plotted parameter is its period deviation from 1s (expressed in ns). The calculation is performed basing on 20 consecutive PPS period measurements, therefore the first 20 events are not significant.
The bottom plot shows the frequency of event acquisition, basing on the TS1 timer. In particular, it is possible to extract the period of TS1 and the trigger rate frequency from this plot.

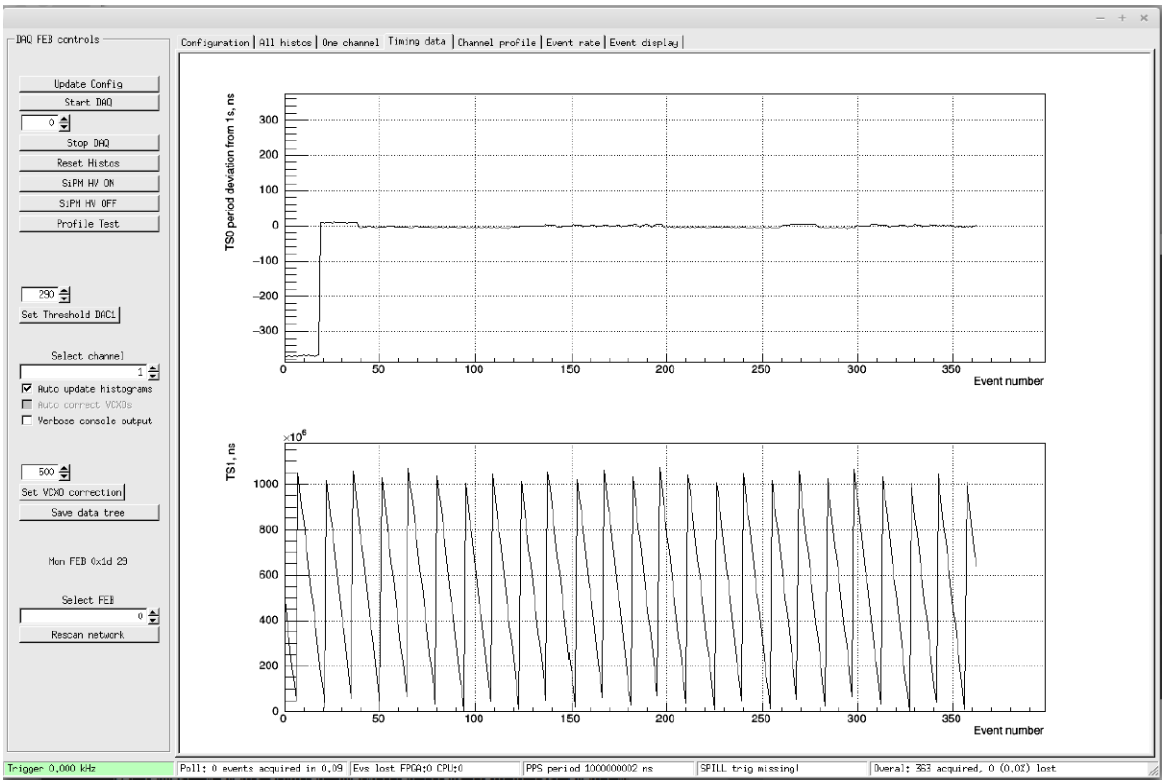


Figure 6.6: the "Timing data" tab when a PPS timer is connected to T0 LEMO input.

The ***"Channel profile"*** tab displays the 32-channels event profile (see **Figure 6.7**). The integrated amplitude (expressed in ADC) of the events recorded by each channel is shown. In the top right part of the display area some additional information are shown, like the total number of recorded events.

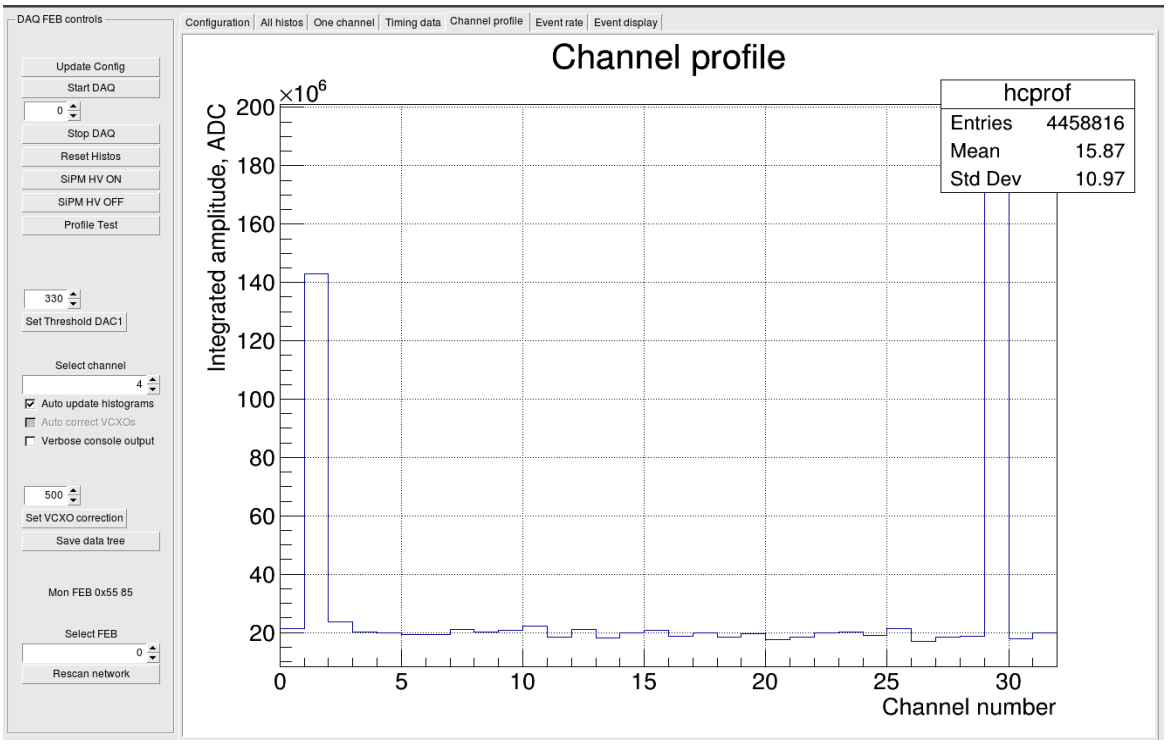


Figure 6.7: the "Channel profile" tab when channels 1 and 29 are acquiring events in coincidence using “nfoldcoinc” FPGA software.

The ***“Event rate”*** tab (see **Figure 6.8**) shows the plot of the event rate value (expressed in kHz) as a function of the “Poll Number” (Poll Nr.). A “Poll” is a package of events used to sample the event rate. In this way the number of events acquired by the FEB in a short period time (usually less than 0.1 s) is measured and the event rate is calculated. For each poll, the trigger rate is calculated and plotted. During DAQ, information about the last transmitted poll are also visible in the bottom statistics bar.

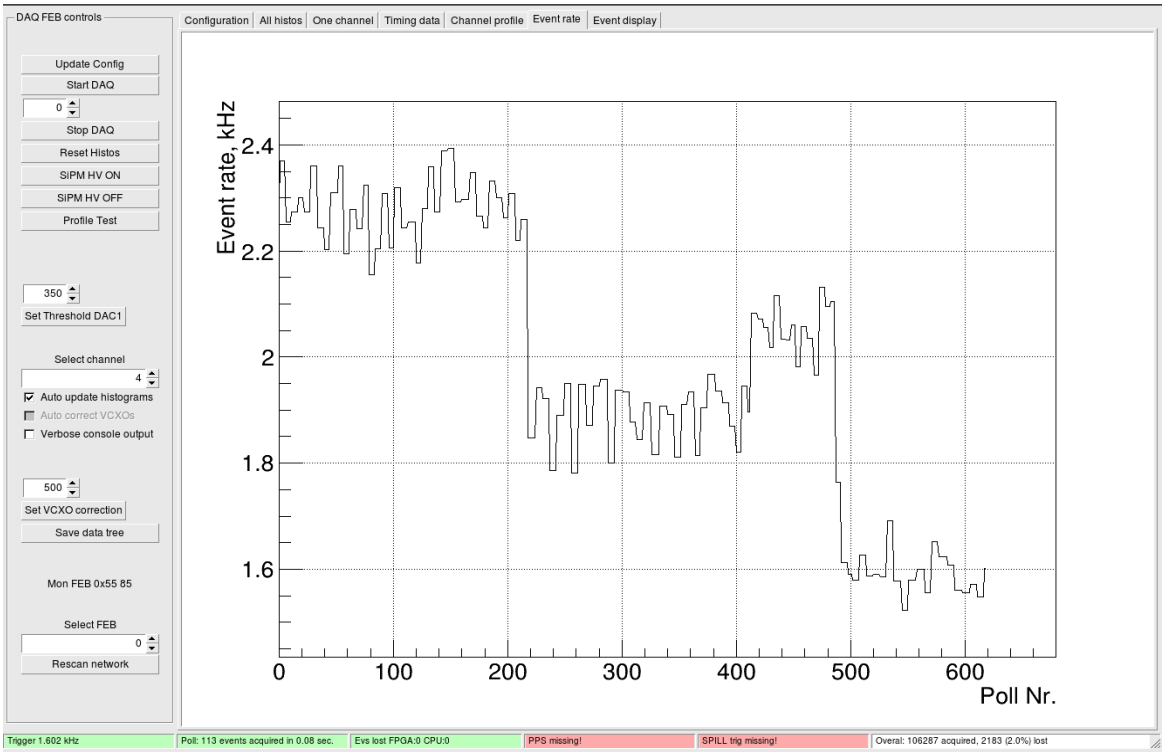


Figure 6.8: the "Event rate” tab during DAQ, while changing the DAC1 threshold. The event rate is plotted as a function of the Poll Nr. In the bottom statistics bar it is possible to read the composition of the last transmitted poll (113 events acquired in 0.08 s).

The ***“Event display”*** tab (see **Figure 6.9**) shows the rainfall-style plot of the number of events recorded by each channel. The different colours stand for the ADC values of the recorded events (the same ADC values of the X-axis of histograms). Some additional information are shown in the top right box.

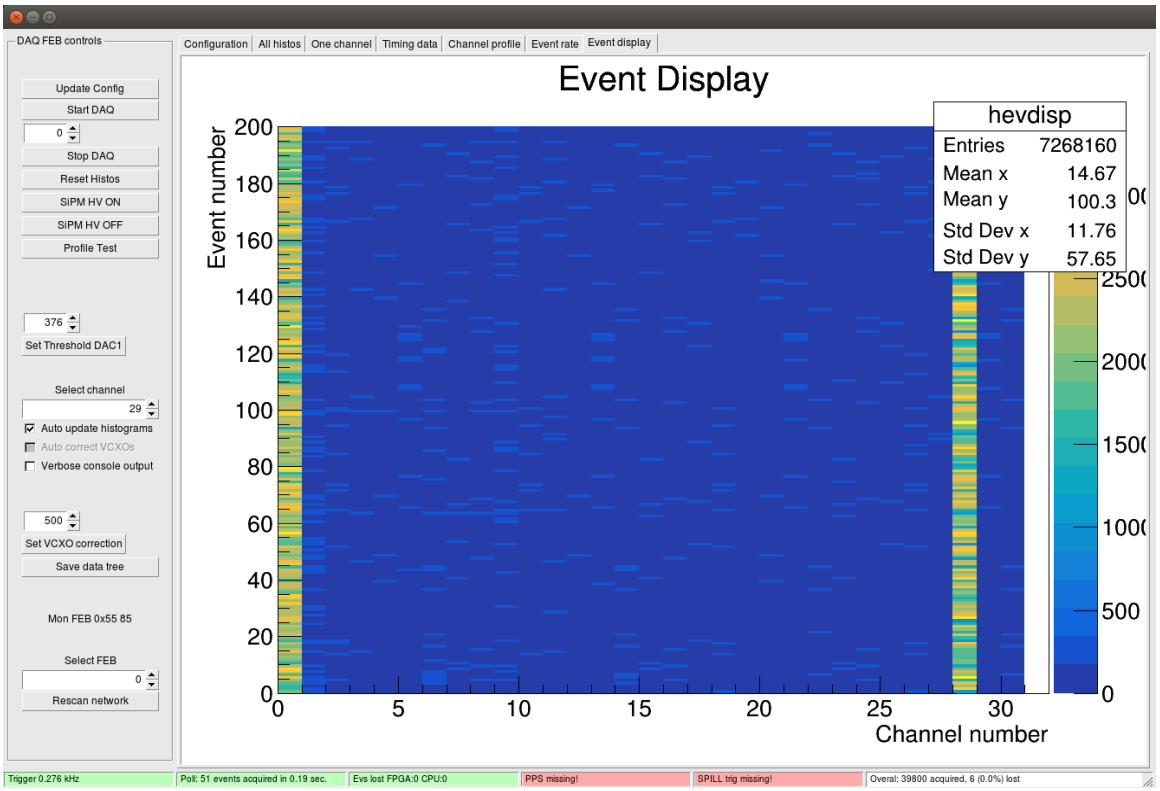


Figure 6.9: the "Event display" tab showing the rainfall-style plot of the event number recorded by each channel.

Statistics bar

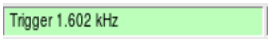
The **statistics bar**, at the bottom of the GUI, is composed of 6 fields and reports the main statistical information about DAQ (see **Figure 6.10**). From left to right, it is possible to read the following data:

- the trigger rate value (expressed in kHz);
- information about the last transmitted Poll;
- the instantaneous number of events lost, in the FPGA and in the CPU;
- the PPS period (if connected to "T0");
- the TS1 timer period (if connected to "T1"), called "SPILL trig";
- the total number of acquired and lost events (these latter also expressed in percentage).

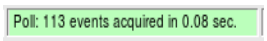
The statistics bar values update automatically during DAQ.

Figure 6.10: the statistics bar at the bottom of the GUI

We now give a brief description of each field of the bar.



The first field reports the **trigger rate** in kHz. This value should always be well below 30 kHz to avoid CITIROC unstable behaviour. A change in this field colour warns the user of a high trigger rate value, changing from green (below 3.5 kHz), to light red (up to 8-9 kHz), to red (above 9 kHz). We suggest performing DAQ with a "green" trigger rate.



The second field reports information about the last transmitted **poll** for trigger rate sampling. The poll is a package of events transmitted to the host computer as samples to calculate the event (or trigger) rate. In particular, the number of events in the poll and the acquisition time is indicated in the statistics area. The maximum number of events in a poll is 1024, i.e. the FEB buffer size. This last situation realize in presence of high trigger rates.

The third field reports, for each transmitted poll, the number of events lost in the FPGA or in the CPU. Events are lost in the FPGA due to pile-up (a trigger occurring during the readout of another event) and they are counted and flagged as described in **Time stamp generator**. Events are lost in the CPU if the incoming data rate exceeds the transmission capacity of the Ethernet interface towards the host computer, as described in **Event buffer and back-end Ethernet interface.** When the number of lost events becomes high, the field is colored in red.

In the the **PPS** field, the measured period of the PPS timer connected to “T0” can be read. If no signals are sent to “T0” the field reports “PPS missing!” and it is light-red colored.

In the **SPILL trig** field, the measured period of the TS1 timer connected to “T1” can be read. The name “SPILL trig” comes from the possible use of the “T1” input signal as an experiment-related timing reference. It could be, for example, an accelerator beam spill pulse. If no signals are sent to “T1”, the field reports “SPILL trig missing!” and it is light-red colored.

The last field contains a summary of the DAQ. The total amount of acquired and lost events is reported, as well as the percentage of lost events w.r.t. total events.

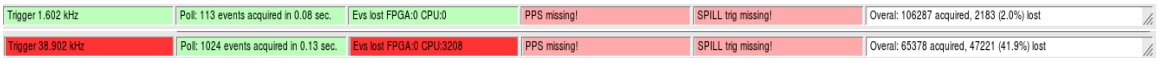


Figure 6.11: the statistics bar during different DAQs. The top bar is a DAQ performed with optimal parameters: the trigger rate is sufficiently low and a small amount of events is lost (only 2%). The bottom bar is the result of a bad set of DAQ parameters: the trigger rate is too high and many events are lost (41.9%).

DAQ using DT5702

We performed some test measurements with a CAEN DT5702, acquiring in coincidence the signals of two Hamamatsu SiPMs. We used the following setup:

- two ***CAEN SP5601 LED driver*** to illuminate SiPMs;
- two Hamamatsu SiPMs hosted in two ***CAEN SP5650C sensor holders***. The SP5650C provides a FC fiber connector and a PCB where the SiPM is soldered;
- the SiPMs were connected to the LED drivers through a 40 cm Optical Fiber;

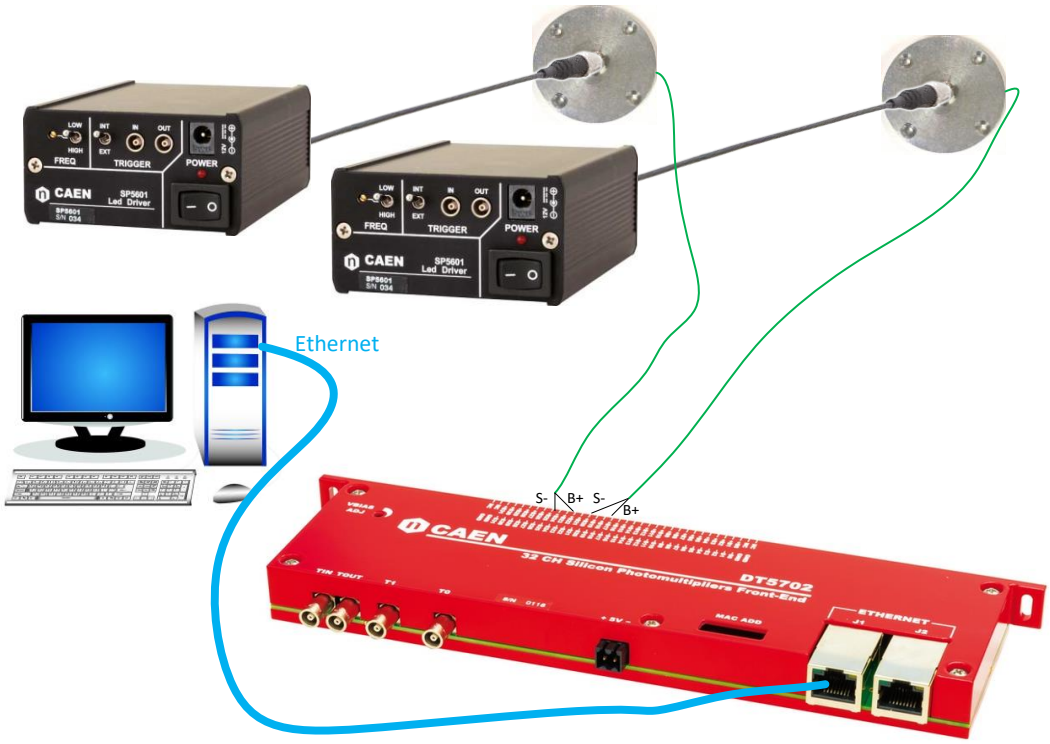


Figure 6.12: a possible setup to perform DAQ using a CAEN DT5702

The test measurements were performed with a DT5702 equipped with the FPGA firmware rel. FLX7.003. We acquired signals from two SiPMs, both in coincidence and in single channel trigger mode.

Note: it is a peculiar characteristic of the board to show an intrinsic noise peak below 300 ADC units, very similar to a pedestal due to SiPMs dark noise. However, this noise has nothing to do with SiPMs but it is rather a noise due to electronics. For this reason, also disconnected or non-triggering channels will show this noise.

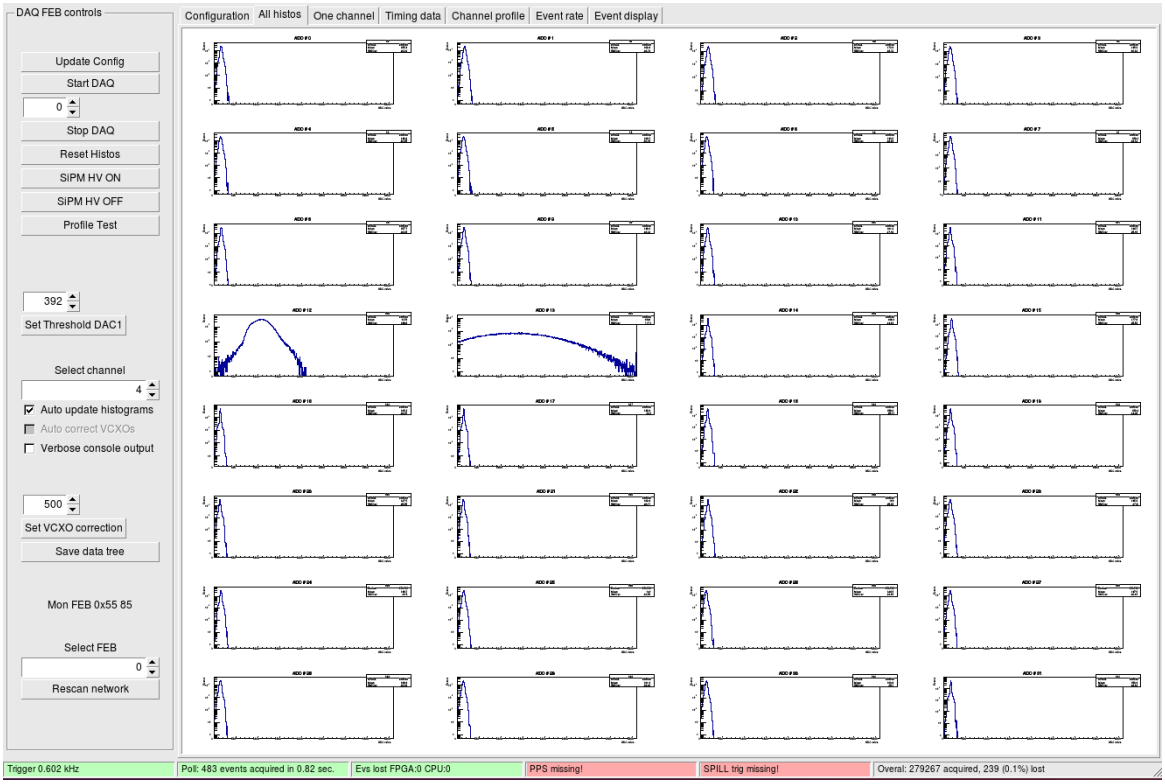


Figure 6.13: DAQ of two SiPMs signals in coincidence. Triggering channels 12 and 13 are not showing any noise peak, while the other channels are clearly showing their pedestals.

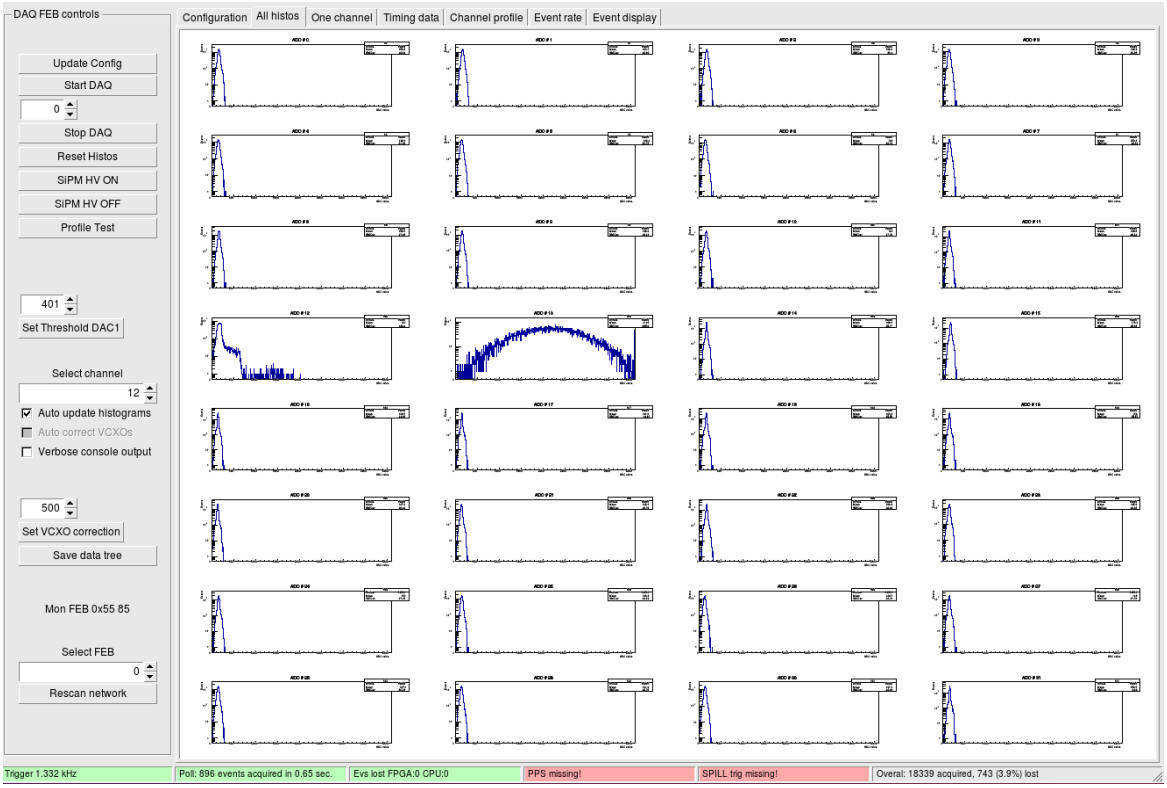


Figure 6.14: DAQ of two SiPMs signals in "OR32" mode, using the FPGA firmware rel FLX7.003. With respect to Figure 6.13, channel 12 is showing its intrinsic low energy noise peaks. Non-triggering channels are also showing their noise peak. Note also that, w.r.t. Figure 6.13, the DAC1 threshold has been increased to avoid trigger rate massive increase.

With the same setup, we acquired the signal from a single SiPM illuminated by the LED pulser. A typical SiPM peak has been obtained, as shown in **Figure 6.15**.

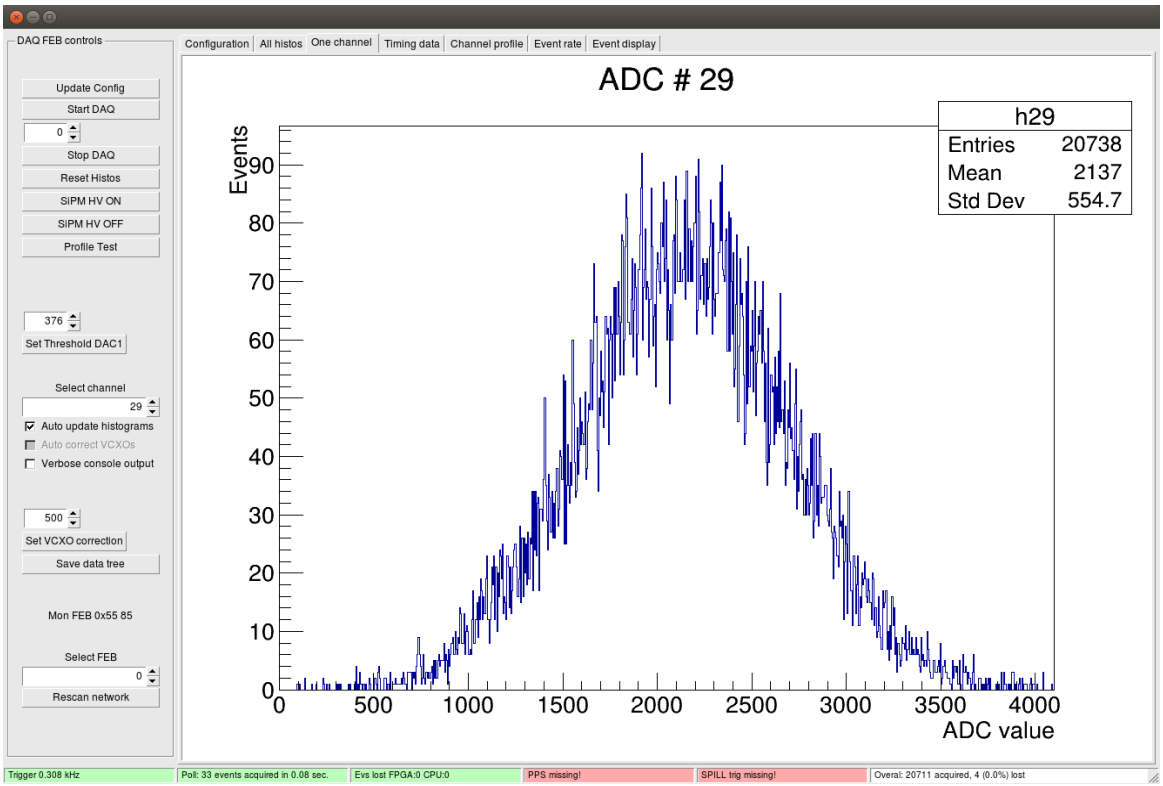


Figure 6.15: signal from a SiPM illuminated by a CAEN SP5601 LED driver. A typical SiPM peak is visible.

7 Annexes

CITIROC Configuration files

In this paragraph, the default configuration of the CITIROC bit stream files is reported and described.

The ***"CITIROC_PROBEbitstream.txt"*** contains the bit stream to configure the internal CITIROC Probe register, in order to monitor ASIC's internal signals. Analog preamplifiers and shapers signal of the 32 CITIROC channels can be monitored. Moreover, the peak detector status can be monitored (see Weeroc CITIROC Datasheet for more details). The analog signals are multiplexed and output at the X3 FEB connector.
The default file is given below with a brief description of the bit stream. Each line corresponds to a different signal to be monitored and has 32 bits, one for each CITIROC channel. A signal of a certain channel is outputted to X3 when the correspondent bit is set to 1.

00000000000000000000000000000000 ' Out_fs From channel 0 to 31 Analog
00000000000000000000000000000000 ' Out_ssh_LG From channel 0 to 31 Analog
00000000000000000000000000000000 ' PeakSensing_modeb_LG From channel 0 to 31 Digital
00000000000000000000000000000000 ' Out_ssh_HG From channel 0 to 31 Analog
00000000000000000000000000000000 ' PeakSensing_modeb_HG 32 From channel 0 to 31 Digital
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ' Out_PA_HG/Out_PA_LG 64 From channel 0 to 15 Analog
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ' Out_PA_HG/Out_PA_LG 64 From channel 16 to 31 Analog

Name	Bits	Description
Out_fs	[0:31]	Fast shaper analog output probe
Out_ssh_LG	[0:31]	Low gain slow shaper analog output probe
PeakSensing_modeb_LG	[0:31]	Low gain peak detector digital output probe
Out_ssh_LG	[0:31]	High gain slow shaper analog output probe
PeakSensing_modeb_HG	[0:31]	High gain peak detector status digital output probe
PeakSensing_modeb_HG	[0:31]	Low gain peak detector status digital output probe
Out_PA_HG/Out_PA_LG	[0:31]	High/Low gain preamplifier analog output probe (channels 0 to 15) Ex.: 00 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ch1 low gain preamplifier output enabled
Out_PA_HG/Out_PA_LG	[0:31]	High/Low gain preamplifier analog output probe (channels 16 to 31) Ex.: 00 00 00 00 00 00 00 00 00 00 00 00 00 10 00 ch30 high gain preamplifier output enabled

Table 7.1: description of CITIROC probe register bit stream as reported in *"CITIROC_PROBEbitstream.txt"*

The ***"CITIROC_SCPROFILE1.txt"*** contains the bit stream to configure the CITIROC Slow Control registers (see the Weeroc CITIROC datasheet for more details). The default file is given below together with a brief description of the bit stream in each line. For more details, refer to Weeroc CITIROC datasheet.

0000 ' Ch0 4-bit DAC_t ([0..3])' //4-bit DAC to adjust the individual "time" trigger threshold
0000 ' Ch1 4-bit DAC_t ([0..3])'
0000 ' Ch2 4-bit DAC_t ([0..3])'
0000 ' Ch3 4-bit DAC_t ([0..3])'
0000 ' Ch4 4-bit DAC_t ([0..3])'
0000 ' Ch5 4-bit DAC_t ([0..3])'
0000 ' Ch6 4-bit DAC_t ([0..3])'
0000 ' Ch7 4-bit DAC_t ([0..3])'
0000 ' Ch8 4-bit DAC_t ([0..3])'
0000 ' Ch9 4-bit DAC_t ([0..3])'
0000 ' Ch10 4-bit DAC_t ([0..3])'
0000 ' Ch11 4-bit DAC_t ([0..3])'
0000 ' Ch12 4-bit DAC_t ([0..3])'
0000 ' Ch13 4-bit DAC_t ([0..3])'

0000 ' Ch14 4-bit DAC_t ([0..3])'
0000 ' Ch15 4-bit DAC_t ([0..3])'
0000 ' Ch16 4-bit DAC_t ([0..3])'
0000 ' Ch17 4-bit DAC_t ([0..3])'
0000 ' Ch18 4-bit DAC_t ([0..3])'
0000 ' Ch19 4-bit DAC_t ([0..3])'
0000 ' Ch20 4-bit DAC_t ([0..3])'
0000 ' Ch21 4-bit DAC_t ([0..3])'
0000 ' Ch22 4-bit DAC_t ([0..3])'
0000 ' Ch23 4-bit DAC_t ([0..3])'
0000 ' Ch24 4-bit DAC_t ([0..3])'
0000 ' Ch25 4-bit DAC_t ([0..3])'
0000 ' Ch26 4-bit DAC_t ([0..3])'
0000 ' Ch27 4-bit DAC_t ([0..3])'
0000 ' Ch28 4-bit DAC_t ([0..3])'
0000 ' Ch29 4-bit DAC_t ([0..3])'
0000 ' Ch30 4-bit DAC_t ([0..3])'
0000 ' Ch31 4-bit DAC_t ([0..3])'
0000 ' Ch0 4-bit DAC ([0..3])' //4-bit DAC to adjust individual “charge” trigger threshold
0000 ' Ch1 4-bit DAC ([0..3])'
0000 ' Ch2 4-bit DAC ([0..3])'
0000 ' Ch3 4-bit DAC ([0..3])'
0000 ' Ch4 4-bit DAC ([0..3])'
0000 ' Ch5 4-bit DAC ([0..3])'
0000 ' Ch6 4-bit DAC ([0..3])'
0000 ' Ch7 4-bit DAC ([0..3])'
0000 ' Ch8 4-bit DAC ([0..3])'
0000 ' Ch9 4-bit DAC ([0..3])'
0000 ' Ch10 4-bit DAC ([0..3])'
0000 ' Ch11 4-bit DAC ([0..3])'
0000 ' Ch12 4-bit DAC ([0..3])'
0000 ' Ch13 4-bit DAC ([0..3])'
0000 ' Ch14 4-bit DAC ([0..3])'
0000 ' Ch15 4-bit DAC ([0..3])'
0000 ' Ch16 4-bit DAC ([0..3])'
0000 ' Ch17 4-bit DAC ([0..3])'
0000 ' Ch18 4-bit DAC ([0..3])'
0000 ' Ch19 4-bit DAC ([0..3])'
0000 ' Ch20 4-bit DAC ([0..3])'
0000 ' Ch21 4-bit DAC ([0..3])'
0000 ' Ch22 4-bit DAC ([0..3])'
0000 ' Ch23 4-bit DAC ([0..3])'
0000 ' Ch24 4-bit DAC ([0..3])'
0000 ' Ch25 4-bit DAC ([0..3])'
0000 ' Ch26 4-bit DAC ([0..3])'
0000 ' Ch27 4-bit DAC ([0..3])'
0000 ' Ch28 4-bit DAC ([0..3])'
0000 ' Ch29 4-bit DAC ([0..3])'
0000 ' Ch30 4-bit DAC ([0..3])'
0000 ' Ch31 4-bit DAC ([0..3])'
1 'Enable discriminator'
1 ' Disable trigger discriminator power pulsing mode (force ON)'
0 ' Select latched (RS : 1) or direct output (trigger : 0)'
1 'Enable Discriminator Two'
1 ' Disable trigger discriminator power pulsing mode (force ON)'
1 ' EN_4b_dac'
1 'PP: 4b_dac'
1 ' EN_4b_dac_t'
1 'PP: 4b_dac_t'
10000000000000000000000000000001 'Allows to Mask Discriminator (channel 0 to 31) [active low]'
1 'Disable High Gain Track & Hold power pulsing mode (force ON)'
1 'Enable High Gain Track & Hold'
1 'Disable Low Gain Track & Hold power pulsing mode (force ON)'
1 'Enable Low Gain Track & Hold'
0 ' SCA bias (1 = weak bias, 0 = high bias 5MHz ReadOut Speed)'

0 'PP: HG Pdet'
0 ' EN_HG_Pdet'
0 'PP: LG Pdet'
0 ' EN_LG_Pdet'
1 ' Sel SCA or PeakD HG'
1 ' Sel SCA or PeakD LG'
1 ' Bypass Peak Sensing Cell'
0 ' Sel Trig Ext PSC'
1 ' Disable fast shaper follower power pulsing mode (force ON)'
1 'Enable fast shaper'
1 ' Disable fast shaper power pulsing mode (force ON)'
1 ' Disable low gain slow shaper power pulsing mode (force ON)'
1 'Enable Low Gain Slow Shaper'
110 ' Low gain shaper time constant commands (0…2) [active low] 100'
1 ' Disable high gain slow shaper power pulsing mode (force ON)'
1 'Enable high gain Slow Shaper'
110 ' High gain shaper time constant commands (0…2) [active low] 100'
0 ' Low Gain PreAmp bias (1 = weak bias, 0 = normal bias)'
1 'Disable High Gain preamp power pulsing mode (force ON)'
1 'Enable High Gain preamp'
1 'Disable Low Gain preamp power pulsing mode (force ON)'
1 'Enable Low Gain preamp'
0 ' Select LG PA to send to Fast Shaper'
1 'Enable 32 input 8-bit DACs'
1 '8-bit input DAC Voltage Reference (1 = external 4,5V , 0 = internal 2,5V)'
11000110 1 ' Input 8-bit DAC Data channel 0 – (DAC7…DAC0 + DAC ON), higher-higher bias' //8-bit DAC to adjust HV bias
11001011 1 ' Input 8-bit DAC Data channel 1 – (DAC7…DAC0 + DAC ON), higher-higher bias'
10111100 1 ' Input 8-bit DAC Data channel 2 – (DAC7…DAC0 + DAC ON), higher-higher bias'
11001010 1 ' Input 8-bit DAC Data channel 3 – (DAC7…DAC0 + DAC ON), higher-higher bias'
11000000 1 ' Input 8-bit DAC Data channel 4 – (DAC7…DAC0 + DAC ON), higher-higher bias'
11001011 1 ' Input 8-bit DAC Data channel 5 – (DAC7…DAC0 + DAC ON), higher-higher bias'
10111111 1 ' Input 8-bit DAC Data channel 6 – (DAC7…DAC0 + DAC ON), higher-higher bias'
11001010 1 ' Input 8-bit DAC Data channel 7 – (DAC7…DAC0 + DAC ON), higher-higher bias'
11001011 1 ' Input 8-bit DAC Data channel 8 – (DAC7…DAC0 + DAC ON), higher-higher bias'
11001011 1 ' Input 8-bit DAC Data channel 9 – (DAC7…DAC0 + DAC ON), higher-higher bias'
11000001 1 ' Input 8-bit DAC Data channel 10 – (DAC7…DAC0 + DAC ON), higher-higher bias'
11000101 1 ' Input 8-bit DAC Data channel 11 – (DAC7…DAC0 + DAC ON), higher-higher bias'
10111111 1 ' Input 8-bit DAC Data channel 12 – (DAC7…DAC0 + DAC ON), higher-higher bias'
11001001 1 ' Input 8-bit DAC Data channel 13 – (DAC7…DAC0 + DAC ON), higher-higher bias'
11000001 1 ' Input 8-bit DAC Data channel 14 – (DAC7…DAC0 + DAC ON), higher-higher bias'
11001101 1 ' Input 8-bit DAC Data channel 15 – (DAC7…DAC0 + DAC ON), higher-higher bias'
11001100 1 ' Input 8-bit DAC Data channel 16 – (DAC7…DAC0 + DAC ON), higher-higher bias'
10111110 1 ' Input 8-bit DAC Data channel 17 – (DAC7…DAC0 + DAC ON), higher-higher bias'
11000110 1 ' Input 8-bit DAC Data channel 18 – (DAC7…DAC0 + DAC ON), higher-higher bias'
11000011 1 ' Input 8-bit DAC Data channel 19 – (DAC7…DAC0 + DAC ON), higher-higher bias'
10111111 1 ' Input 8-bit DAC Data channel 20 – (DAC7…DAC0 + DAC ON), higher-higher bias'
11000001 1 ' Input 8-bit DAC Data channel 21 – (DAC7…DAC0 + DAC ON), higher-higher bias'
11000101 1 ' Input 8-bit DAC Data channel 22 – (DAC7…DAC0 + DAC ON), higher-higher bias'
11000000 1 ' Input 8-bit DAC Data channel 23 – (DAC7…DAC0 + DAC ON), higher-higher bias'
11000010 1 ' Input 8-bit DAC Data channel 24 – (DAC7…DAC0 + DAC ON), higher-higher bias'
10111001 1 ' Input 8-bit DAC Data channel 25 – (DAC7…DAC0 + DAC ON), higher-higher bias'
10111011 1 ' Input 8-bit DAC Data channel 26 – (DAC7…DAC0 + DAC ON), higher-higher bias'
11000000 1 ' Input 8-bit DAC Data channel 27 – (DAC7…DAC0 + DAC ON), higher-higher bias'
10111111 1 ' Input 8-bit DAC Data channel 28 – (DAC7…DAC0 + DAC ON), higher-higher bias'
11000011 1 ' Input 8-bit DAC Data channel 29 – (DAC7…DAC0 + DAC ON), higher-higher bias'
10111100 1 ' Input 8-bit DAC Data channel 30 – (DAC7…DAC0 + DAC ON), higher-higher bias'
11000110 1 ' Input 8-bit DAC Data channel 31 – (DAC7…DAC0 + DAC ON), higher-higher bias'
110011 101111 000 ' Ch0 PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch1 PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch2 PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch3 PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch4 PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch5 PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch6 PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch7 PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'

110011 101111 000 ' Ch8   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch9   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch10   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch11   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch12   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch13   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch14   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch15   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch16   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch17   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch18   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch19   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch20   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch21   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch22   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch23   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch24   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch25   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch26   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch27   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch28   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch29   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch30   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
110011 101111 000 ' Ch31   PreAmp config (HG gain[5..0], LG gain [5..0], CtestHG, CtestLG, PA disabled)'
1 ' Disable Temperature Sensor power pulsing mode (force ON)'
1 'Enable Temperature Sensor'
1 'Disable BandGap power pulsing mode (force ON)'
1 'Enable BandGap'
1 'Enable DAC1'
1 ' Disable DAC1 power pulsing mode (force ON)'
1 'Enable DAC2'
1 ' Disable DAC2 power pulsing mode (force ON)'
00 1111 1010 ' 10-bit DAC1 (MSB-LSB): 00 1100 0000 for 0.5 p.e.'
00 1111 1010 ' 10-bit DAC2 (MSB-LSB): 00 1100 0000 for 0.5 p.e.'
1 'Enable High Gain OTA' -- start byte 2
1 ' Disable High Gain OTA power pulsing mode (force ON)'
1 'Enable Low Gain OTA'
1 ' Disable Low Gain OTA power pulsing mode (force ON)'
1 'Enable Probe OTA'
1 ' Disable Probe OTA power pulsing mode (force ON)'
0 ' Otaq test bit'
0 'Enable Val_Evt receiver'
0 ' Disable Val_Evt receiver power pulsing mode (force ON)'
0 'Enable Raz_Chn receiver'
0 ' Disable Raz Chn receiver power pulsing mode (force ON)'
0 'Enable digital multiplexed output (hit mux out)'
0 ' Enable digital OR32 output'
0 ' Enable digital OR32 Open Collector output'
0 ' Trigger Polarity'
0 ' Enable digital OR32_T Open Collector output'
1 'Enable 32 channels triggers outputs'

8 Technical Support

CAEN experts can provide technical support 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
Phone +49 (0)212 254 4077
Fax +49 (0)212 25 44079
Mobile +49 (0)151 16 548 484
info@caen-de.com
www.caen-de.com

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

UM5833 - A1702/DT5702 - User Manual rev. 2 - July 20th, 2018
00000-00-A1702-MUTX