Rev. 3 - June 27th, 2024

CAEN PLU Library

C Library for DT5495 and V2495 boards

CAEN
Tools for Discovery

User Manual
UM6510

Purpose of this User Manual

This User Manual contains the full description of the CAEN PLU library (Windows/Linux).

Change Document Record

Date	Revision	Changes
July 23rd, 2018	00	Initial release
April 09th, 2019	01	Revised Sec. **Installation**. Updated Chap. **2** with error code "-13". Updated Chapp. **3**, **4** by replacing "CAEN_PLU_API CAEN_PLU_ERROR_CODE" with "CAEN_PLU_ERROR_CODE CAEN_PLU_API" in the library functions.
October 15th, 2021	02	Added library support to CAEN new Bridges V3718/V418 and the A4818 adapter. Updated Chap. **1**. Updated Function **OpenDevice (deprecated)**. Added Function **OpenDevice2** in Chap. **3**.
June 27th, 2024	03	Added Chapter **5.**

Symbols, Abbreviated Terms and Notation

CONET	Chainable Optical NETwork
OS	Operating System
PLU	Programmable Logic Unit

Reference Document

[RD1] UM1934 – CAENComm Library User & Reference Manual

[RD2] UM5175 - V2495/VX2495 User Manual

[RD3] UM6508 – DT5495 User Manual

[RD4] V1718/VX1718 Technical Information Manual

[RD5] V2718/VX2718 Technical Information Manual

[RD6] UM7685 – V3718/VX3718 User Manual

[RD7] UM8305 – V4718/VX4718 User Manual

[RD8] A3818 PCI Express Optical Link Controller Technical Information Manual

[RD9] A2818 PCI Optical Link Controller Technical Information Manual

[RD10] DS7799 – A4818 Data Sheet

https://www.caen.it/support-services/documentation-area/

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

Disclaimer

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

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

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

Index

List of Figures

List of Tables

1 Introduction

CAEN PLU library (also referred to as PLULib) has been developed to support the V2495 and DT5495 CAEN programmable logic units. This library specifically provides a set of functions to interface these boards through the direct USB and Ethernet communication interfaces, but also the VMEbus connection by using CAEN Bridges (V1718, V2718, V3718 and V41718), controllers (A2818, A3818), and A4818 adapter is supported.

The PLULib supports all the communication channels currently provided by CAEN:

- PC → USB2 → V2495 / DT5495
- PC → ETH → DT5495
- PC → ETH → V4718 → V2495
- PC → PCI/PCIe → A2818/A3818 → CONET → V2718/V3718/V4718 → VME → V2495
- PC → USB2 → V1718/V3718 → VME → V2495
- PC → USB3 → V4718 →VME →V2495
- PC → USB3 →A4818 → CONET →V2718/V3718/V4718 → VME → V2495

"CONET" (Chainable Optical NETwork) indicates the CAEN proprietary protocol for communication on Optical Link **[RD1]**.

The core functions of the CAEN PLU library (Sec. **Core Functions**) manage the connection with the target board, as well as the write and read to registers, while dedicated functions handle multiple boards management by USB link (Sec. **Enumeration Functions**), provide to initialize and program the gate-and-delay generators (Sec. **Gate and Delay Generators Functions**), permit to write and read the FLASH memory of the target board FPGAs for the firmware management (Sec. **Low-level Flash Memory Access Functions**), retrieve the board and status information (Sec. **Miscellaneous**).

System Requirements

CAEN PLU library is available for Windows® and Linux® OS. Working with this library, commonly requires the driver for the used communication link and the CAENComm library to be installed (see **Tab. 1.1**)

Links	OS	CAEN Drivers	Dependencies	Thirdy-party Software
Direct USB	Windows	DT5495/V2495 USB Driver	CAENComm Library **[RD1]**	Not needed
	Linux	Not needed		
V1718 USB	Windows	V1718 Win USB Driver		
	Linux	V1718 Linux USB Driver		
V3718 USB	Windows	V3718 Win USB Driver		
	Linux	V3718 Linux USB Driver		
V4718 USB	Windows	Not needed		
	Linux	V4718 Linux Driver		
V4718 ETH	Windows	Not needed		
	Linux			
V2718/V3718/V4718 + A2818 CONET	Windows	A2818 Win Driver		
	Linux	A2818 Linux Driver		
V2718/V3718/V4718 + A3818 CONET	Windows	A3818 Win Driver		
	Linux	A3818 Linux Driver		
Ethernet	Windows	Not needed		
	Linux			
A4818 USB	Windows	A4818 Win Driver		
	Linux	Not needed		

Tab. 1.1: Library requiremets

The drivers for the DT5495/V2495 communication interfaces are free downloadable on CAEN web site in the DT5495 and V2495 pages (**login required**). Installation instructions can be found in the User Manual of the board **[RD2] [RD3]**. In case the of using CAEN Bridges, Controllers or Adapter (V4718, V3718, V2718, V1718, A2818, A3818, A4818), refer the relevant product web page (**login required**).

Note: When using the direct USB link, V2495 and DT5495 are fully supported by Linux from kernel version 3.13 on. This means that the USB driver compatible with the unit is in the Linux system itself, and the user is not required to install any driver from CAEN (refer to **Tab. 1.1**).

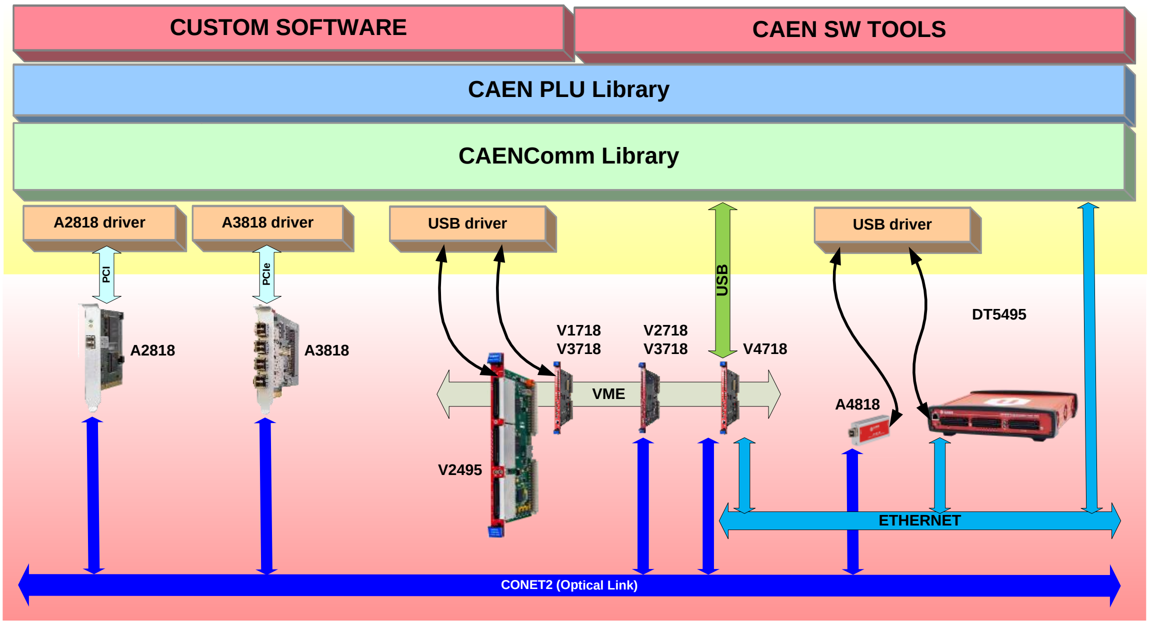


Fig. 1.1: Hardware and Software layers for Windows OS

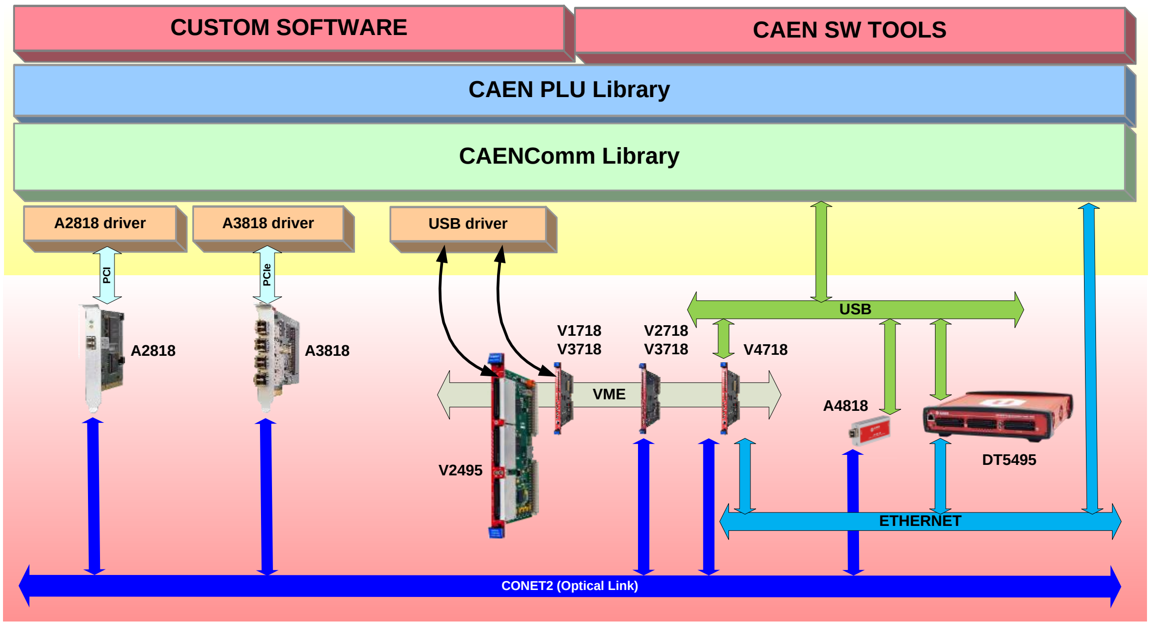


Fig. 1.2: Hardware and Software layers for Linux OS

Installation

Before installing the PLU library:

- Make sure that your hardware (Programmable Logic Unit and/or Bridge, or Controller) is properly installed **[RD4][RD5][RD6][RD7][RD8][RD9][RD10]**.
- If required, make sure that you have installed the driver compliant to your OS and to the physical communication layer being used (see Sec. **System Requirements**).
- Make sure you have installed the required CAENComm library, downloadable on CAEN website (**login required**).

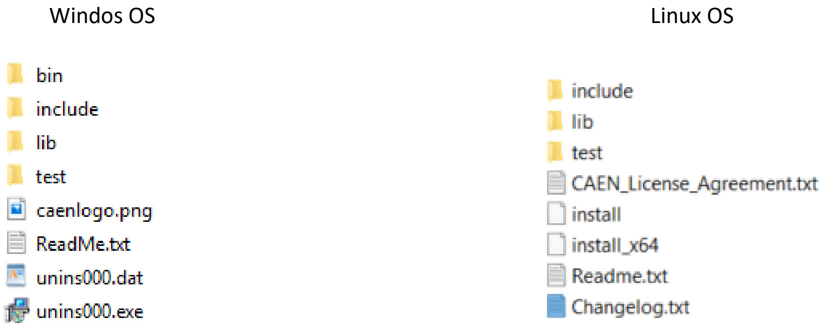
Download and unpack the CAEN PLULib installation package compliant to your OS from the DT5495 or V2495 web page (**login required**).

For Windows users:

- Launch the installer file *CAEN_PLULib-X.X-buildYYYMMDD.exe* and complete the installation wizard
- By default, find the library installed at: *C:\Program Files\CAEN\PLULibrary*.
- The main directory includes subfolders as in **Fig. 1.3**, left side.

For Linux users:

- The unpacked library includes subfolders as in **Fig. 1.3**, right side.
- Check the system requirements in the *README* file.
- Execute "*sudo sh install*" in case of 32-bit Linux or "*sudo sh install_x64*" in case of 64-bit.



Windos OS

bin
include
lib
test
caenlogo.png
ReadMe.txt
unins000.dat
unins000.exe

Linux OS

include
lib
test
CAEN_License_Agreement.txt
install
install_x64
Readme.txt
Changelog.txt

Fig. 1.3: PLULib subfolders

PLULib Test

The CAEN PLULib package includes a simple test demo:

- The Windows executable file is *CAEN_PLULib_TEST.exe*, included in the *bin* folder
- The Linux executable is *CAENPLUTest*, included in the *test* folder.
- The *test* folder contains the source C files and the Visual Studio project.

The test demo must not be intended as a readout software. The User cannot do any operation on the target board, but the demo test permits to automatically check the library functions good working.

The test demo, as is, does not support the connections through the A4818 Adapter and the V4718 Bridge.

In any case, the provided source files allow the User integrating and customizing the software basing on the PLU library functions.

The connection parameters currently implemented in the test demo are:

➔ *connection type "c"* -> the identifier of the active communication link:

- *0* = USB direct link
- *1* = Ethernet link
- *2* = USB-to-VME through the V1718 or V3718 CAEN Bridges
- *3* = CONET-to-VME through the V2718 or V3718 CAEN Bridges

➔ *Device serial number* "sn" -> the serial number or the PID of the target board to be used in the case of direct USB connection.

➔ *IP address "ip"* -> the IP address of the target board; this parameter is meaningful only in case of Ethernet connection to the DT5495 board.

➔ *VME base address "b"* -> the VME Base Address of the target board; this parameter is meaningful only in case of connection to the V2495 board through a CAEN Bridge (V1718/V2718/V3718).

Instructions for Windows users:

- Go to the *bin* folder by the Windows Command Prompt:

  C:\>cd "Program Files\CAEN\PLULibrary\bin"

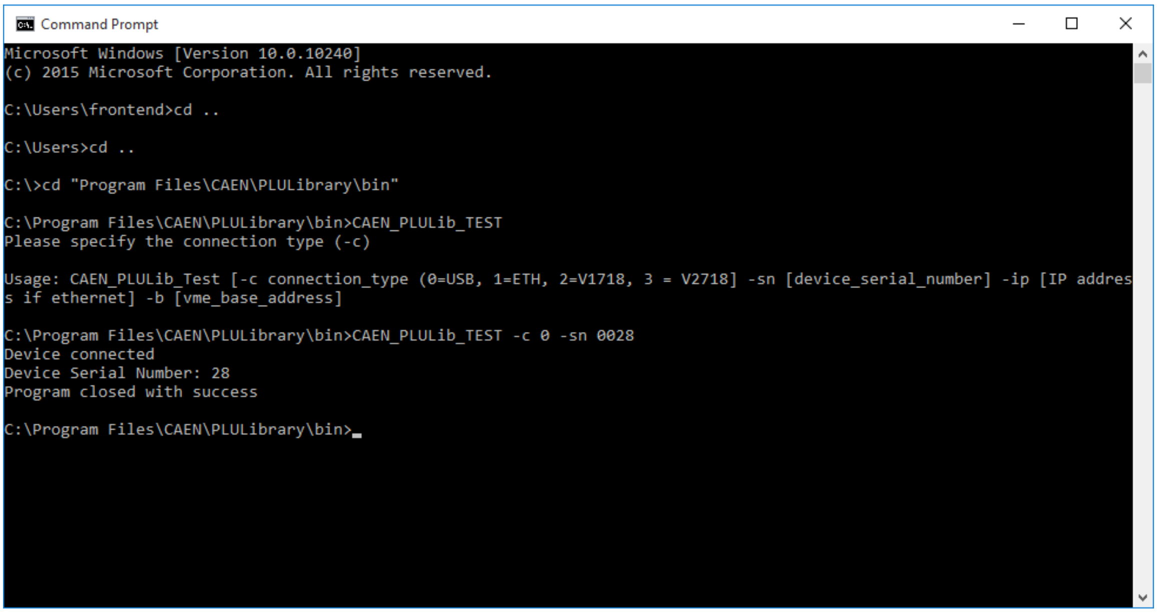
- Type *CAEN_PLULib_TEST* if you want to recall the usage syntax string:

  "Usage: CAEN_PLULib_TEST -c [connection type 0=USB, 1=ETH, 2=V1718, 3=V2718] -sn [device_serial_number] -ip [IP address if ethernet] -b [vme_base_address]"

- To run the test demo in case of direct USB connection and serial sumber 28, type:

  CAEN_PLULib_TEST -c 0 -sn 28

The demo will automatically connect to the target board, execute the test calling the library functions, normally ending with a message of process completed with success.



```
Microsoft Windows [Version 10.0.10240]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\frontend>cd ..

C:\Users>cd ..

C:\>cd "Program Files\CAEN\PLULibrary\bin"

C:\Program Files\CAEN\PLULibrary\bin>CAEN_PLULib_TEST
Please specify the connection type (-c)

Usage: CAEN_PLULib_Test [-c connection_type (0=USB, 1=ETH, 2=V1718, 3 = V2718] -sn [device_serial_number] -ip [IP addres
s if ethernet] -b [vme_base_address]

C:\Program Files\CAEN\PLULibrary\bin>CAEN_PLULib_TEST -c 0 -sn 0028
Device connected
Device Serial Number: 28
Program closed with success

C:\Program Files\CAEN\PLULibrary\bin>_
```

Instructions for Linux users:

Note: Linux USB drivers do not automatically give to the user the privileges needed to direct connect to the target board. Using the test demo in case of direct USB connection, the user must log in as root or activate the low-level user permissions.

- Go to the *test* subfolder of the library directory.
- Execute the *make* command.
- Type "sudo *./CAENPLUTest"* if you want to recall the usage syntax string:

  "usage: sudo ./CAENPLUTest -c [connection type 0=USB, 1=ETH, 2=V1718, 3=V2718] -sn [device_serial_number] -ip [IP address if ethernet] -b [vme_base_address]"

- To run the test demo in case of CONET connection by V2718 CAEN Bridge and V2495 with VME Base Address 3210000, type:

sudo ./CAENPLUTest -c 3 -b 32100000

The demo automatically connects to the target board, executes the test, and ends like in Windows case above.

2 Error Codes

Here are described the macros to define the error codes from the library functions (see **Tab. 2.1**).

Synopsis

```
enum CAEN_PLU_ERROR_CODE {
                          CAEN_PLU_OK =0,
                          CAEN_PLU_GENERIC =-1,
                          CAEN_PLU_INTERFACE =-2,
                          CAEN_PLU_FPGA =-3,
                          CAEN_PLU_TRANSFER_MAX_LENGTH =-4,
                          CAEN_PLU_NOTCONNECTED =-5,
                          CAEN_PLU_NO_DATA_AVAILABLE =-6,
                          CAEN_PLU_TOO_MANY_DEVICES_CONNECTED =-7,
                          CAEN_PLU_INVALID_HANDLE =-8,
                          CAEN_PLU_INVALID_HARDWARE =-9,
                          CAEN_PLU_INVALID_PARAMETERS = -10
                          CAEN_PLU_TERMINATED = -13
                          };
```

Error Code	Value	Description
OK	0	No errors.
GENERIC	-1	Generic (not specified) error.
INTERFACE	-2	Interface error while connecting to the target board.
FPGA	-3	FPGA internal error.
TRANSFER_MAX_LENGTH	-4	Transfer size in a read or write access exceeds 16 MB.
NOTCONNECTED	-5	Attempting to perform a read or write access while the target board is not connected.
NO_DATA_AVAILABLE	-6	No data from the target device is available for readout.
TOO_MANY_DEVICES_CONNECTED	-7	Attempting to connect to more than 100 devices.
INVALID_HANDLE	-8	Invalid handle used.
INVALID_HARDWARE	-9	Attempting to connect through an invalid hardware/interface.
INVALID_PARAMETERS	-10	At least one of the function parameters is not valid (value is not in the allowed range).
CAEN_PLU_TERMINATED	-13	Communication terminated by the device.

Tab. 2.1: Return codes table

3 Library Functions

Core Functions

OpenDevice (deprecated)

Description
This is the original function that allows establishing the connection with a device in your network. It returns a handle that can be used later to interact with the system. However, as it covers only a subset of all the possible connections, it is rather suggested using the **OpenDevice2** function.

Synopsis
```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_OpenDevice(t_ConnectionModes connection_mode,
                    char *IPAddress_or_SN_or_VMEBaseAddress,
                    int VMElink,
                    int VMEConetNode,
                    int *handle);
```

Arguments

Name	I/O	Description
connection_mode	Input	Indicates the physical communication channel: - *CAEN_PLU_CONNECT_DIRECT_USB* (direct USB connection) - *CAEN_PLU_CONNECT_DIRECT_ETH* (direct Ethernet connection) - *CAEN_PLU_CONNECT_VME_V1718* (USB-to-VME connection through V1718/V3718) - *CAEN_PLU_CONNECT_VME_V2718* (Optical-to-VME connection through V2718/V3718) See **t_ConnectionModes**.
IPAddress_or_SN_or_VMEBaseAddress	Input	Pointer to the: - IP address of the unit in case of Ethernet connection; - Serial Number of the unit in case of direct USB connection; - VME Base Address of the unit (only V2495) in case of connection through CAEN Bridges.
VMElink	Input	Link number (VME). In case of USB and Ethernet connections, this paramenter is not significant.
VMEConetNode	Input	Conet node in the Daisy chain (VME with V2718/V3718 bridges). In case of USB and Ethernet connections, or VME V1718, this paramenter is not significant.
handle	Output	Returns a handle for subsequent library accesses. Can be NULL if connection is not possible.

Return Values
CAEN_PLU_OK (0) in case of success. Negative numbers are error codes (see **Error Codes**).

Examples
Connecting to a PLU module (V2495 or DT5495) via direct USB link:

```
ret = CAEN_PLU_OpenDevice(CAEN_PLU_CONNECT_DIRECT_USB, "4", 0, 0, &handle);
```

A connection via a Ethernet is opened with:

```
ret = CAEN_PLU_OpenDevice(CAEN_PLU_CONNECT_DIRECT_ETH, "192.168.7.11", 0, 0, &handle);
```

V1718 and V3718 (USB) access can be opened with:

```
ret = CAEN_PLU_OpenDevice(CAEN_PLU_CONNECT_VME_V1718, vme_base_address, 0, 0, &handle);
```

V2718 and V3718 (CONET) bridge connection can be opened with:

```
ret = CAEN_PLU_OpenDevice(CAEN_PLU_CONNECT_VME_V2718, vme_base_address, 0, 0, &handle);
```

In both the latter cases, a VME Base Address of the PLU module **must** be specified.

OpenDevice2

Description

This function is suggested to be used rather than the **OpenDevice (deprecated)** function, as it allows to open a device in all the possible connections, including those through the V4718 CAEN VME bridge and the A4818 adapter.

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_OpenDevice2(
                     t_ConnectionModes connection_mode,
                     void* IPAddress_or_SN_or_VMELink,
                     int VMEConetNode,
                     char* VMEBaseAddress,
                     int* handle
                     );
```

Arguments

Name	I/O	Description
connection_mode	Input	Indicates the physical communication channel. It can be: - *CAEN_PLU_CONNECT_DIRECT_USB* (direct USB connection); - *CAEN_PLU_CONNECT_DIRECT_ETH* (direct Ethernet connection); - *CAEN_PLU_CONNECT_VME_V1718* (USB-to-VME connection through V1718/V3718); - *CAEN_PLU_CONNECT_VME_V2718* (Optical-to-VME connection through V2718/V3718); - *CAEN_PLU_CONNECT_VME_V4718_ETH* (Ethernet-to-VME connection through V4718); - *CAEN_PLU_CONNECT_VME_V4718_USB* (USB-to-VME connection through V4718); - *CAEN_PLU_CONNECT_VME_A4818* (USB-to-CONET connection through A4818). See **t_ConnectionModes**.
IPAddress_or_SN_or_VMELink	Input	Pointer to the: - IP address of the unit in the case of an Ethernet connection to the V4718 or direct Ethernet; - Serial Number/PID of the unit in the case of USB connection to the A4818 adapter, V4718 bridge, or direct USB; - Link number (VME) in the case of USB connection to V1718 or V3718.
VMEConetNode	Input	Conet node in the Daisy chain (VME). In case of direct USB and Ethernet connections, or USB to VME, this paramenter is not significant.
VMEBaseAddress	input	VME Base Address of the unit (only V2495) in the case of VME connection through CAEN Bridges. In the case of Direct connection, this paramenter is not significant.
handle	Output	Returns a handle for subsequent library accesses. Can be NULL if connection is not possible.

Return Values

CAEN_PLU_OK (0) in case of success. Negative numbers are error codes (see **Error Codes**).

CloseDevice

Description

This function closes the connection with the programmable logic unit. The CloseDevice function must be called before to exit the application.

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_CloseDevice(int handle);
```

Arguments

Name	I/O	Description
handle	Input	Device handler to be closed.

Return Values

0: Success. Negative numbers are error codes (see **Error Codes**).

Examples

Close the device with:

```
ret = CAEN_PLU_CloseDevice(handle);
```

WriteReg

Description

Generic write access to a register of the device.

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_WriteReg(
                  int handle,
                  uint32_t address,
                  uint32_t value
                  );
```

Arguments

Name	I/O	Description
handle	Input	Library handle (as returned by *CAEN_PLU_OpenDevice()*).
address	Input	Register address (for the VME access, this is the lower 16-bit part of the VME address bus).
value	Input	32-bit data to write at the addressed register.

Return Values

0: Success. Negative numbers are error codes (see **Error Codes**).

ReadReg

Description

Generic read access to a register of the device.

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_ReadReg(int handle,
                 uint32_t address,
                 uint32_t *value);
```

Arguments

Name	I/O	Description
handle	Input	Library handle (as returned by *CAEN_PLU_OpenDevice()*).
address	Input	Register address (for the VME access, this is the lower 16-bit part of the VME address bus).
value	Output	Pointer to the 32-bit data read at the addressed register.

Return Values

0: Success. Negative numbers are error codes (see **Error Codes**).

WriteData32

Description

This function writes 32-bit data into memory.

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_WriteData32(int handle,
                     uint32_t start_address,
                     uint32_t size,
                     uint32_t *value);
```

Arguments

Name	I/O	Description
handle	Input	Library handle (as returned by *CAEN_PLU_OpenDevice()*).
start_address	Input	Start adddress for read operation. Address automatically incremented for each new value access.
size	Input	Size of transfert in 32-bit words.
value	Output	Pointer to the values to write.

Return Values

0: Success; Negative numbers are error codes (see **Error Codes**).

WriteFIFO32

Description

This function writes 32-bit data at the same address (FIFO mode).

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_WriteFIFO32(int handle,
                     uint32_t start_address,
                     uint32_t size,
                     uint32_t *value);
```

Arguments

Name	I/O	Description
handle	Input	Library handle (as returned by *CAEN_PLU_OpenDevice()*).
start_address	Input	Start address for write operation. Address is **NOT** incremented for each new value access.
size	Input	Size of transfert in 32-bit words.
value	Output	Pointer to the values to write.

Return Values

0: Success. Negative numbers are error codes (see **Error Codes**).

ReadData32

Description

This function reads 32-bit data from memory.

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_ReadData32(int handle,
                    uint32_t start_address,
                    uint32_t size,
                    uint32_t *value,
                    uint32_t *nw);
```

Arguments

Name	I/O	Description
handle	Input	Library handle (as returned by *CAEN_PLU_OpenDevice()*).
start_address	Input	Start adddress for read operation. Address automatically incremented for each new value access.
size	Input	Size of transfer in 32-bit words.
value	Output	Pointer to read values.
nw	Output	Number of 32-bit words read. It is less or equal to *size*.

Return Values

0: Success. Negative numbers are error codes (see **Error Codes**).

ReadFIFO32

Description

Thi function reads 32-bit data from the same adddress (FIFO mode).

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_ReadFIFO32(int handle,
                    uint32_t address,
                    uint32_t size,
                    uint32_t *value,
                    uint32_t *nw);
```

Arguments

Name	I/O	Description
handle	Input	Library handle (as returned by *CAEN_PLU_OpenDevice()*).
start_address	Input	Start adddress for read operation. Address is **NOT** incremented for each new value access.
size	Input	Size of transfer in 32-bit words.
value	Output	Pointer to read values.
nw	Output	Number of 32-bit words read. It is less or equal to *size*.

Return Values

0: Success. Negative numbers are error codes (see **Error Codes**).

Enumeration Functions

USBEnumerate

Description

This function enumerates the boards connected via USB direct link.

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_USBEnumerate(tUSBDevice *pvArg1,
                      uint32_t *numDevs);
```

Arguments

Name	I/O	Description
pvArg1	Output	Pointer to USB devices enumerated.
numDevs	Output	Number of enumerated boards.

Return Values

0: Success. Negative numbers are error codes (see **Error Codes**).

USBEnumerateSerialNumber

Description

This function enumerates the boards connected via USB direct link and returns a Serial Number as a string.

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_USBEnumerateSerialNumber(unsigned int *numDevs,
                                  char *DeviceSNs,
                                  uint32_t buffersize);
```

Arguments

Name	I/O	Description
numDevs	Output	Number of enumerated devices.
deviceSNs	Output	Serial number.
buffersize	Input	String length.

Return Values

0: Success. Negative numbers are error codes (see **Error Codes**).

Gate and Delay Generators Functions

InitGateAndDelayGenerators

Description

This function performs the Gate and Delay initialization. It **MUST** be called prior to any Gate and Delay function call.

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_InitGateAndDelayGenerators(int handle);
```

Arguments

Name	I/O	Description
handle	Input	Library handle (as returned by *CAEN_PLU_OpenDevice()*).

Return Values

0: Success. Negative numbers are error codes (see **Error Codes**).

SetGateAndDelayGenerator

Description

This function enables and sets a single gate and delay generator channel.

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_SetGateAndDelayGenerator(int handle,
                                  uint32_t channel,
                                  uint32_t enable,
                                  uint32_t gate,
                                  uint32_t delay,
                                  uint32_t scale_factor);
```

Arguments

Name	I/O	Description
handle	Input	Library handle (as returned by *CAEN_PLU_OpenDevice()*).
channel	Input	Gate and Delay channel to set.
enable	Input	Channel enable.
gate	Input	Gate value in gate steps (valid range = 0-65535.)
delay	Input	Delay value in delay steps (valid range = 0-65535).
scale_factor	Input	Scale factor for delay (valid range = 0-255). 0 is the minimum gate and delay resolution (~10 ns).

Note: Gate+Delay parameters cannot exceed 65535.

Return Values

0: Success. Negative numbers are error codes (see **Error Codes**).

SetGateAndDelayGenerators

Description

This function enables and set **ALL** gate and delay generators channels with a common value.

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_SetGateAndDelayGenerators(int handle,
                                   uint32_t gate,
                                   uint32_t delay,
                                   uint32_t scale_factor);
```

Arguments

Name	I/O	Description
handle	Input	Library handle (as returned by *CAEN_PLU_OpenDevice()*).
gate	Input	Gate value in gate steps (Valid range = 0-65535).
delay	Input	Delay value in delay steps (Valid range = 0-65535).
scale_factor	Input	Scale factor for delay (Valid range = 0-255). 0 is the minimum gate and delay resolution (~10 ns).

Note: Gate+Delay parameters cannot exceed 65535.

Return Values

0: Success. Negative numbers are error codes (see **Error Codes**).

GetGateAndDelayGenerator

Description

This function gets the Gate and Delay channel parameters.

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_GetGateAndDelayGenerator(int handle,
                                  uint32_t channel,
                                  uint32_t *gate,
                                  uint32_t *delay,
                                  uint32_t *scale_factor);
```

Arguments

Name	I/O	Description
handle	Input	Library handle (as returned by *CAEN_PLU_OpenDevice()*).
channel	Input	Gate and Delay channel.
gate	Output	Gate value in gate steps.
delay	Output	Delay value in delay steps.
scale_factor	Output	Fine tune value in fine tune steps.

Return Values

0: Success. Negative numbers are error codes (see **Error Codes**).

Low-level Flash Memory Access Functions

EnableFlashAccess

Description

By this function, it is possible to enable the Flash access. It **MUST** be called prior to any Flash access function call.

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_EnableFlashAccess(int handle,
                           t_FPGA_V2495 FPGA);
```

Arguments

Name	I/O	Description
handle	Input	Library handle (as returned by *CAEN_PLU_OpenDevice()*).
FPGA	Input	The possible target FPGA: - *FPGA MAIN* (MAIN FPGA); - *FPGA USER* (USER FPGA); - *FPGA DELAY* (GATE AND DELAY FPGA). See **t_FPGA_V2495**.

Return Values

0: Success. Negative numbers are error codes (see **Error Codes**).

DisableFlashAccess

Description

By this function, it is possible to disable the Flash access. It **MUST** be called prior to any flash access function call.

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_DisableFlashAccess(int handle,
                            t_FPGA_V2495 FPGA);
```

Arguments

Name	I/O	Description
handle	Input	Library handle (as returned by *CAEN_PLU_OpenDevice()*).
FPGA	Input	The possible target FPGA: - *FPGA MAIN* (MAIN FPGA); - *FPGA USER* (USER FPGA); - *FPGA DELAY* (GATE AND DELAY FPGA). See **t_FPGA_V2495**.

Return Values

0: Success. Negative numbers are error codes (see **Error Codes**).

DeleteFlashSector

Description

This function deletes a single Flash sector.

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_DeleteFlashSector(int handle,
                           t_FPGA_V2495 FPGA,
                           uint32_t sector);
```

Arguments

Name	I/O	Description
handle	Input	Library handle (as returned by *CAEN_PLU_OpenDevice()*).
FPGA	Input	The possible target FPGA: - *FPGA MAIN* (MAIN FPGA); - *FPGA USER* (USER FPGA); - *FPGA DELAY* (GATE AND DELAY FPGA). See **t_FPGA_V2495**.
sector	Input	Flash se tor to delete (64 KB). MAIN and USER Flash (N25Q256 model) have 512x64KB sectors; GATE AND DELAY Flash (W25Q64 model) has 128x64KB sectors.

Note: Please, **BE AWARE** that some sectors are reserved for factory and user firmware. User storage area is in sectors 106-510 for MAIN Flash and sectors 458-510 for USER Flash. DELAY Flash should not be used for user data.

Return Values

0: Success. Negative numbers are error codes (see **Error Codes**).

WriteFlashData

Description

This function allows to write data into the Flash.

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_WriteFlashData(int handle,
                        t_FPGA_V2495 FPGA,
                        uint32_t address,
                        uint32_t *data,
                        uint32_t length);
```

Arguments

Name	I/O	Description
handle	Input	Library handle (as returned by *CAEN_PLU_OpenDevice()*).
FPGA	Input	The possible target FPGA: - *FPGA MAIN* (MAIN FPGA); - *FPGA USER* (USER FPGA); *FPGA DELAY* (GATE AND DELAY FPGA). See **t_FPGA_V2495**.
address	Input	Flash start address.
data	Input	Pointer to data to write into the Flash.
length	Input	Data length in 32-bit words.

Return Values

0: Success. Negative numbers are error codes (see **Error Codes**).

ReadFlashData

Description

This function allows to read data from the Flash.

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_ReadFlashData(int handle,
                       t_FPGA_V2495 FPGA,
                       uint32_t address,
                       uint32_t *data,
                       uint32_t length);
```

Arguments

Name	I/O	Description
`handle`	Input	Library handle (as returned by *CAEN_PLU_OpenDevice()*).
`FPGA`	Input	The possible target FPGA: - *FPGA MAIN* (MAIN FPGA); - *FPGA USER* (USER FPGA); *FPGA DELAY* (GATE AND DELAY FPGA). See **t_FPGA_V2495**.
`address`	Input	Flash start address.
`data`	Output	Pointer to data read from the Flash.
`length`	Input	Data length in 32-bit words.

Return Values

0: Success. Negative numbers are error codes (see **Error Codes**).

Miscellaneous

GetInfo

Description

This function retrieves the module information.

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_GetInfo(int handle,
                 tBOARDInfo *HWOPTIONS;
```

Arguments

Name	I/O	Description
handle	Input	Library handle (as returned by *CAEN_PLU_OpenDevice()*).
HWOPTIONS	Output	Pointer to a *tBOARDInfo* structure (see **tBOARDInfo)**: - *checksum*; - *checksum_length2*; - *checksum_length1*; - *checksum_length0*; - *checksum_constant2*; - *checksum_constant1*; - *checksum_constant0*; - *c_code*; - *r_code*; - *oui2*; - *oui1*; - *oui0*; - *version*; - *board2*; - *board1;* - *board0*; - *revis3*; - *revis2*; - *revis1*; - *revis0*; - *reserved[12]*; - *sernum1*; - *sernum0.*

Return Values

0: Success. Negative numbers are error codes (see **Error Codes**).

GetSerialNumber

Description

This function retrieves the module serial number stored into the Configuration ROM

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API
CAEN_PLU_GetSerialNumber(int handle,
                         char *sn,
                         uint32_t buffersize);
```

Arguments

Name	I/O	Description
handle	Input	Library handle (as returned by *CAEN_PLU_OpenDevice()*).
sn	Output	Serial number.
buffersize	Input	Serial number string length.

Return Values

0: Success. Negative numbers are error codes (see **Error Codes**).

ConnectionStatus

Description

This function gets the current connection status from the unit.

Synopsis

```
CAEN_PLU_ERROR_CODE CAEN_PLU_API_t
CAEN_PLU_ConnectionStatus(int handle,
                          int *status);
```

Arguments

Name	I/O	Description
handle	Input	Library handle (as returned by *CAEN_PLU_OpenDevice()*).
status	Output	Connection status with ID (0 = USB, 1 = ETH, 2 = V1718, 3 = V2718).

Return Values

0: Success. Negative numbers are error codes (see **Error Codes**).

4 Data Structure and Type Description

t_ConnectionModes

Description

Enumerated type for the kind of connection link.

Synopsis

```
typedef enum
{
 CAEN_PLU_CONNECT_DIRECT_USB,
 CAEN_PLU_CONNECT_DIRECT_ETH,
 CAEN_PLU_CONNECT_VME_V1718,
 CAEN_PLU_CONNECT_VME_V2718,
 CAEN_PLU_CONNECT_VME_V4718_ETH,
 CAEN_PLU_CONNECT_VME_V4718_USB,
 CAEN_PLU_CONNECT_VME_A4818
} t_ConnectionModes;
```

Fields

Value	Type	Description
CAEN_PLU_CONNECT_DIRECT_USB	enum	USB direct connection type.
CAEN_PLU_CONNECT_DIRECT_ETH	enum	Ethernet direct connection type.
CAEN_PLU_CONNECT_VME_V1718	enum	USB-to-VME connection type through the V1718 Bridge.
CAEN_PLU_CONNECT_VME_V2718	enum	CONET-to-VME connection type through the V2718 Bridge.
CAEN_PLU_CONNECT_VME_V4718_ETH	enum	Ethernet connection type to V4718 Bridge.
CAEN_PLU_CONNECT_VME_V4718_USB	enum	USB connection type tp V4718 Bridge.
CAEN_PLU_CONNECT_VME_A4818	enum	USB connection type to A4818 Adapter.

t_FPGA_V2495

Description

Enumerated type for the kind of V2495/DT5495 target FPGA.

Synopsis

```
typedef enum
{
FPGA_MAIN  = 0,
FPGA_USER  = 1,
FPGA_DELAY = 2
} t_FPGA_V2495;
```

Fields

Value	Type	Description
FPGA_MAIN	enum	MAIN FPGA.
FPGA_USER	enum	USER FPGA.
FPGA_DELAY	enum	GATE AND DELAY FPGA.

tBOARDInfo

Description

This structure defines the board generic information from the Configuration ROM.

Synopsis

```
typedef struct _tBOARDInfo
{
 uint32_t   checksum;
 uint32_t   checksum_length2;
 uint32_t   checksum_length1;
 uint32_t   checksum_length0;
 uint32_t   checksum_constant2;
 uint32_t   checksum_constant1;
 uint32_t   checksum_constant0;
 uint32_t   c_code;
 uint32_t   r_code;
 uint32_t   oui2;
 uint32_t   oui1;
 uint32_t   oui0;
 uint32_t   version;
 uint32_t   board2;
 uint32_t   board1;
 uint32_t   board0;
 uint32_t revis3;
 uint32_t revis2;
 uint32_t revis1;
 uint32_t revis0;
 uint32_t reserved[12];
 uint32_t sernum1;
 uint32_t sernum0;
} tBOARDInfo;
```

Fields

Name	Type	Description
checksum	uint32_t	Checksum value of the Configuration ROM space.
checksum_length2	uint32_t	3-byte checksum length (i.e. the number of bytes in the Configuration ROM to checksum).
checksum_length1	uint32_t	
checksum_length0	uint32_t	
checksum_constant2	uint32_t	3-byte Configuration Rom constant.
checksum_constant1	uint32_t	
checksum_constant0	uint32_t	
c_code	uint32_t	ASCII C character code (identifies this as CR space).
r_code	uint32_t	ASCII R character code (identifies this as CR space).
oui2	uint32_t	3-byte IEEE Organizationally Unique Identifier (OUI).
oui1	uint32_t	
oui0	uint32_t	
version	uint32_t	Board version information.
board2	uint32_t	3-byte board ID.
board1	uint32_t	
board0	uint32_t	
revis3	uint32_t	4-byte hardware revision.
revis2	uint32_t	
revis1	uint32_t	
revis0	uint32_t	
reserved[12]	uint32_t	*n.a.*
sernum1	uint32_t	Board Serial Number.
sernum0	uint32_t	

_tUSBDevice

Description

This structure defines the USB device descriptor.

Synopsis

```
typedef struct _tUSBDevice
{
 uint32_t   id;
 char       SN[64];
 char       DESC[64];
} tUSBDevice;
```

Fields

Name	Type	Description
id	uint32_t	Incremental number of the enumerated interface (starts from 0).
SN[64]	char	The string of the serial number of the device.
DESC[64]	char	USB device string descriptor: can be “DT5495” or “V2495”.

5 CAEN PLU Python Binding

The CAENPLU Python binding is distributed through the Pypi repository. It can be installed via the command:

pip install caen-libs

The package includes the CAENVME, the CAENComm, the CAEN PLU library, and the CAEN HV Wrapper. It requires the C library installed. The CAEN PLU module can be import in the python script as follows:

```
from caen_libs import caenplu as plu
```

An example of the usage of the Python functions may be found on the CAEN GitHub repository (https://github.com/caenspa/py-caen-libs).

6 Technical Support

CAEN makes available the technical support of its specialists for requests concerning the software and hardware. Use the support form available at the following link:

https://www.caen.it/support-services/support-form/

Scan me

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

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

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

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

UM6510 - CAEN PLU Library rev. 3 - June 27th, 2024 00106/17:PLULIB.MUT/02