CAEN

Tools for Discovery

Electronic Instrumentation

CAEN HV Wrapper Library

Rev. 10 - 13 November 2013

Purpose of this User Manual

This User's Manual contains the full description of the **CAEN HV Wrapper Library**.

Change Document Record

Date	Revision	Changes
3 October 2012	6	Event mode, subscribe parameters
5 November 2012	7	Updated CAENHV_GetChParamProp
23 May 2013	8	Updated CAENHV_InitSystem
21 June 2013	9	N568E Support
13 November 2013	10	VME8x00 and DT55xx support

Symbols, abbreviated terms and notation

T.B.D.

Reference Document

SY1527 User's Manual
SY4527 User's Manual
V6533 User's Manual
N1470 User's Manual
N568E User's Manual
DT55xx User's Manual
VME8200 User's Manual

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

1. Introduction

This document describes the CAEN HV Wrapper library and the functions it implements.

CAEN HV Wrapper is a set of ANSI C functions which allows to control CAEN devices. It contains a generic software interface independent by the Power Supply models and by the communication path used to exchange data with them (at present, CAENET via A303A/A1303, USB, CONET Optical Link or TCP/IP).

At the moment of writing this document describing Rel. 5.0, CAEN HV Wrapper is available in the following formats:

Win32 DLL (CAEN provides the CAENHVWrapper.lib stub for Microsoft Visual C++ 6.0 and later)

Linux dynamic library

CAEN HV Wrapper is logically located between an application like ActiveHV or OPC server and the lower layer software libraries[1], as shown in the scheme below:

			OPC Server					Active HV		
Communication Support Interface										
DT55xx	N/NDT14xx	N568E	V65xx	VME8x00	SYx527	SY527	SY127	SY403	N470/N570	N568B/LC
USB, CONET	USB VCP TCP/IP		CAENComm	TCP/IP				HSCAENET Lib		
			USB/CONET					CAENET (A303/A1303)		

The user of the library must initialize the Power Supply to which to connect by using the proper function; then the software will return a handle.

Once the Communication Support Interface understands that the given Power Supply is a SYx527, it calls the specific functions of the SYx527 Interface which, on his side, uses the standard socket interface to control the P.S.

[1] ActiveHV, OPC server and HSCAENETLib are described in other documents, please refer to CAEN Web site (**www.caen.it**/computing) for more info

2. Communication Support Interface

The exported functions are declared in **CAENHVWrapper.h**.

Description of the functions

```
CAENHVRESULT CAENHV_InitSystem(
CAENHV_SYSTEM_TYPE_t  system,            // In
int                         LinkType,     // In
void                        *Arg,         // In
const char                  *UserName,    // In
const char                  *Passwd,      // In
int                         *handle       // Out
);
```

Parameters	Description
system	The type of the system to connect with SY1527 0 SY2527 1 SY4527 2 SY5527 3 N568 B/LC 4 V65XX 5 N1470 6 VME CRATE 7 N568E 8 DT55XX 9
LinkType	0 = TCP/IP 2 = HS CAENET 3 = USB 2.0 4 = CONET Optical Link 5 = USB VCP
Arg	If LinkType is 0, points to: SYx527 a char IP; N568E to IP_lbusaddress If LinkType is 2, points to a char of the type "A303_IOAddr_CrNum" or "A1303_Id_CrNum" If LinkType is 3 or 4, it is: "LinkNum_ConetNode_VMEBaseAddress"; LinkNum: When using OpticalLink, it is the optical link number to be used. When using USB, it is the USB device number to be used. ConetNode: For OpticalLink, it identifies which device in the daisy-chain is addressed. For USB, it must be 0. VMEBaseAddress: The VME base address; eight figures, last four must be all 0; for example EEFF0000, with EEFF Base address set via rotary switches.[2] If LinkType is 5, it is: "commport_baudrate_commdata_commstop_commparity_lbusaddress"; With Windows: commport = COM0, COM1…; with Linux: commport = ttyS0, ttyS1, ttyUSB0, ttyUSB1…
UserName	A string containing the User's Name; has meaning only for SYX527
Password	A string containing the User's Password; has meaning only for SYX527
handle	Handle returned by the CAENHV_InitSystem function

This is the first function with parameter System to call, and it must be called for all the HV power supplies the user wants to control; if linkType is 2, it executes a CAENET 0 command to see which type of high voltage system is connected to the given CrNum. The Arg parameter, in this case, is formed by three parts: the name of the board (A303 or A1303), the IO port address in the A303 case or an identifier starting from 0 for the A1303 selection (multiple A1303 boards can be used in the same PC) and the crate number of the system in the chain.

If linkType is 0, it executes a login command (SYx527 is assumed) and, if it works well, it executes the command which returns the system model name to see which type of high voltage system is connected.

If linkType is 3 or 4, the VME Power Supply Boards are accessed via CAEN VME USB Bridge or Optical Link Bridge respectively and the DT55xx desktop power supplies are accessed via USB2.0 or CONET respectively; in order to do this, the CAENComm library shall be installed. If LinkType is 5, the device is accessed via USB Virtual Com Port.

It then inserts a new entry into the table of correspondences between the systemName and some useful parameters, like the handle (if SYx527), the model name, …

[2] See CAENComm library documentation

```
CAENHVRESULT CAENHV_DeinitSystem(
int       handle       // In
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function

This is the last function with parameter `SystemName` to call, and it must be called for all the HV power supplies the user wants to control.

```
CAENHVRESULT CAENHV_GetChName(
int                          handle,            // In
unsigned short               slot,              // In
unsigned short               ChNum,              // In
const unsigned short         *ChList,            // In
char                 (*ChNameList)[MAX_CH_NAME]  // Out
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
Slot	The slot; in case of SYx527, the MSByte indicates the crate in the cluster
ChNum	Number of channels in the list
ChList	List of channels
ChNameList	List of returned channels names.

```
CAENHVRESULT CAENHV_SetChName(
int                    handle,            // In
unsigned short          slot,             // In
unsigned short          ChNum,            // In
const unsigned short   *ChList,           // In
const char             *ChName            // In
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
Slot	The slot; in case of SYX527, the MSByte indicates the crate in the cluster
ChNum	Number of channels in the list
ChList	List of channels
ChName	New name of the channels

```
CAENHVRESULT CAENHV_GetChParamInfo(
int                    handle,            // In
unsigned short         slot,              // In
unsigned short         Ch,                // In
char                   **ParNameList       // Out
int                    *ParNumber          // Out
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
Slot	The slot; in case of SYX527, the MSByte indicates the crate in the cluster
Ch	The channel
ParNameList	List of the names of the parameters of channel Ch; the list is ended by the NUL string; memory pointed by ParNameList must be deallocated by the user
`ParNumber`	Number of the parameters in the list

As an example, in this document we show the list returned for the **A1832** board. For the list relative to the other boards, please refer to their user's manual.

Parameter Name	Description
V0Set	Set V0 voltage limit
I0Set	Set I0 current limit
V1Set	Set V1 voltage limit
I1Set	Set I1 current limit
Rup	Set ramp-up rate
RDWn	Set ramp-down rate
Trip	Set trip time
SVMax	Set software voltage limit
Vmon	Voltage monitor
Imon	Current monitor
Status	Channel status
Pw	Power ON/OFF
Pon	Power ON options
PDwn	Power down options
TripInt	Internal trip connections
TripExt	External trip connections

```
CAENHVRESULT CAENHV_GetChParamProp(
int                   handle,           // In
unsigned short        slot,             // In
unsigned short        Ch,               // In
const char            *ParName,         // In
const char           *PropName,         // In
void                 *retval            // Out
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
Slot	The slot; in case of SYX527, the MSByte indicates the crate in the cluster
Ch	The channel
ParName	The name of the parameter whose property we want to know; possible value: "Vmon"
PropName	The name of the property whose value we want to know; possible value: "Maxval"
Retval	The value of the property

This function permits to know a property of a given parameter.
For every parameter two properties are available:
the property called "Type" which can assume the following values (of type unsigned long): PARAM_TYPE_NUMERIC, PARAM_TYPE_ONOFF, PARAM_TYPE_CHSTATUS, PARAM_TYPE_STRING, PARAM_TYPE_ENUM, PARAM_TYPE_BINARY and PARAM_TYPE_BDSTATUS.
the property called "Mode" which can assume the following 3 values (of type unsigned long): PARAM_MODE_RDONLY, PARAM_MODE_WRONLY, PARAM_MODE_RDWR.

Depending on the values above, other properties exist following the relations shown in the next table:

Type = PARAM_TYPE_NUMERIC, **Value** = float

Property	Property Type	Description
Minval	Float	Minimum numeric value
Maxval	Float	Maximum numeric value
Unit	Unsigned short	Index to this list of Engineering Units: PARAM_UN_NONE, PARAM_UN_AMPERE, PARAM_UN_VOLT, PARAM_UN_WATT, PARAM_UN_CELSIUS, PARAM_UN_HERTZ, PARAM_UN_BAR, PARAM_UN_VPS, PARAM_UN_SECOND, PARAM_UN_RPM, PARAM_UN_COUNT
Exp	Short	+3 (Kilo), +6 (Mega), -3 (milli), -6 (micro)
Decimal	Unsigned short	Number of decimal figures

Type = PARAM_TYPE_ONOFF, **Value** = unsigned (0, 1)

Property	Property Type	Description
Onstate	Char *	String indicating the Onstate, i.e. “On” or “Enabled”
Offstate	Char *	String indicating the Offstate, i.e. “Off” or “Disabled”

Type = PARAM_TYPE_CHSTATUS, **Value** = the following bitfield

Bit 0	Channel is on
Bit 1	Channel is ramping up
Bit 2	Channel is ramping down
Bit 3	Channel is in overcurrent
Bit 4	Channel is in overvoltage
Bit 5	Channel is in undervoltage
Bit 6	Channel is in external trip
Bit 7	Channel is in max V
Bit 8	Channel is in external disable
Bit 9	Channel is in internal trip
Bit 10	Channel is in calibration error
Bit 11	Channel is unplugged
Bit 12	reserved forced to 0
Bit 13	Channel is in OverVoltage Protection
Bit 14	Channel is in Power Fail
Bit 15	Channel is in Temperature Error
Bit 16…31	Reserved, forced to 0

No Properties available

Type = PARAM_TYPE_BINARY, **Value** = integer

Check on board manual the meaning of the bit mask

Type = PARAM_TYPE_STRING, **Value** = char*

Type = PARAM_TYPE_ENUM, **Value** = unsigned short

Property	Property Type	Description
Minval	Float	Minimum numeric value
Maxval	Float	Maximum numeric value
Enum	Float*	Array of finite values ; dimension is equal to difference between Maxval and Minval

Type = PARAM_TYPE_BDSTATUS

Bit 0	Board is in power-fail status
Bit 1	Board has a firmware checksum error
Bit 2	Board has a calibration error on HV
Bit 3	Board has a calibration error on temperature

Bit 4	Board is in under-temperature status
Bit 5	Board is in over-temperature status
Bit 6…31	Reserved, forced to 0

No Properties available

```
CAENHVRESULT CAENHV_GetChParam(
(int                   handle,          // In
unsigned short         slot,            // In
const char            *ParName,         // In
unsigned short         ChNum,           // In
const unsigned short *ChList,           // In
void                  *ParValList       // Out
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
Slot	The slot; in case of SYX527, the MSByte indicates the crate in the cluster
ParName	Name of the parameter
ChNum	Number of channels in the list
ChList	List of channels
ParValList	List of returned parameters values

As an example, in this document we show the parameters which the user can specify for the **A1832** board. For the other boards, please refer to their user's manual.

Parameter Name	Type pointed by ParValList
V0Set	Float
I0Set	Float
V1Set	Float
I1Set	Float
Rup	Float
RDWn	Float
Trip	Float
SVMax	Float
Vmon	Float
Imon	Float
Status	Unsigned (Bitfield)
Pw	Unsigned (Boolean)
Pon	Unsigned (Boolean)
PDwn	Unsigned (Boolean)
TripInt	Unsigned
TripExt	Unsigned

```
CAENHVRESULT CAENHV_SetChParam(
int                    handle,          // In
unsigned short          slot,           // In
const char             *ParName,        // In
unsigned short          ChNum,          // In
const unsigned short   *ChList,         // In
void                   *ParValue        // In
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
Slot	The slot; in case of SYX527, the MSByte indicates the crate in the cluster

ParName	Name of the parameter
ChNum	Number of channels in the list
ChList	List of channels
ParValue	New parameter value

As an example, in this document we show the parameters which the user can specify for the **A1832** board. For the other boards, please refer to their user's manual.

Parameter Name	**Type pointed by ParValList**
V0Set	Float
I0Set	Float
V1Set	Float
I1Set	Float
Rup	Float
RDWn	Float
Trip	Float
SVMax	Float
Pw	Unsigned (Boolean)
Pon	Unsigned (Boolean)
PDwn	Unsigned (Boolean)
TripInt	Unsigned
TripExt	Unsigned

```
CAENHVRESULT CAENHV_TestBdPresence
(
int             handle,             // In
unsigned short  slot,               // In
short           *NrOfCh,            // Out
char            **Model,            // Out
char            **Description,      // Out
unsigned short  *SerNum,            // Out
unsigned char   *FmwRelMin,         // Out
unsigned char   *FmwRelMax          // Out
);
```

Parameters	**Description**
SystemName	A string like "Systemx"
Slot	The slot; in case of SYX527, the MSByte indicates the crate in the cluster
NrOfCh	Number of channels in the board
Model	Model of the board, i.e. "A1734"; NULL if board not present
Description	Description of the board, i.e. "12 channels …"
SerNum	Board Serial Number
FmwRelMin	LSByte of firmware release: 0 if rel. 1.0
FmwRelMax	MSByte of firmware release: 1 if rel. 1.0

```
CAENHVRESULT CAENHV_GetBdParamInfo(

int                     handle,      // In
ushort                  slotNum,     // In
const ushort            *slotList,   // Out
const char              *ParName,    // Out
void                    *ParValList  // Out
   );
```

Parameters	**Description**
handle	Handle returned by the CAENHV_InitSystem function
Slot	The slot; in case of SYX527, the MSByte indicates the crate in

	the cluster
ParNameList	List of the names of the parameters of the board; memory pointed by ParNameList must be deallocated by the user

As an example, in this document we show the list returned for the **A1832** board. For the list relative to the other boards, please refer to their user's manual.

Parameter Name	Description
BdStatus	Board status
HVMax	Hardware voltage limit
Temp	Board temperature

```
CAENHVRESULT CAENHV_GetBdParamProp(
int                    handle,           // In
unsigned short         slot,             // In
const char              *ParName,        // In
const char            *PropName,         // In
void                  *retval            // Out
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
Slot	The slot; in case of SYX527, the MSByte indicates the crate in the cluster
ParName	The name of the parameter whose property we want to know; possible value: "Hvmax"
PropName	The name of the property whose value we want to know; possible value: "MaxVal"
Retval	The value of the property

This function permits to know a property of a given parameter.
For every parameter two properties are available:
the property called "Type" which can assume the following values (of type unsigned long): PARAM_TYPE_NUMERIC, PARAM_TYPE_ONOFF, PARAM_TYPE_CHSTATUS, PARAM_TYPE_STRING, PARAM_TYPE_ENUM, PARAM_TYPE_BINARY and PARAM_TYPE_BDSTATUS.
the property called "Mode" which can assume the following 3 values (of type unsigned long): PARAM_MODE_RDONLY, PARAM_MODE_WRONLY, PARAM_MODE_RDWR.
Depending on the values above, other properties exist following the relations shown in the next table:

Type = PARAM_TYPE_NUMERIC, **Value** = float

Property	Property Type	Description
Minval	Float	Minimum numeric value
Maxval	Float	Maximum numeric value
Unit	Unsigned short	Index to this list of Engineering Units: PARAM_UN_NONE, PARAM_UN_AMPERE, PARAM_UN_VOLT, PARAM_UN_WATT, PARAM_UN_CELSIUS, PARAM_UN_HERTZ, PARAM_UN_BAR, PARAM_UN_VPS, PARAM_UN_SECOND, PARAM_UN_RPM, PARAM_UN_COUNT
Exp	Short	+3 (Kilo), +6 (Mega), -3 (milli), -6 (micro)

Type = PARAM_TYPE_ONOFF, **Value** = unsigned (0, 1)

Property	Property Type	Description
Onstate	Char *	String indicating the Onstate, i.e. "On" or "Enabled"
Offstate	Char *	String indicating the Offstate, i.e. "Off" or "Disabled"

Type = PARAM_TYPE_CHSTATUS, **Value** = the following bitfield

Bit 0	Channel is on
Bit 1	Channel is ramping up

Bit 2	Channel is ramping down
Bit 3	Channel is in overcurrent
Bit 4	Channel is in overvoltage
Bit 5	Channel is in undervoltage
Bit 6	Channel is in external trip
Bit 7	Channel is in max V
Bit 8	Channel is in external disable
Bit 9	Channel is in internal trip
Bit 10	Channel is in calibration error
Bit 11	Channel is unplugged
Bit 12…31	Reserved, forced to 0

No Properties available

Type = PARAM_TYPE_BINARY, **Value** = integer

Check on board manual the meaning of the bit mask

Type = PARAM_TYPE_STRING, **Value** = char*

Type = PARAM_TYPE_ENUM, **Value** = unsigned short

Property	Property Type	Description
Minval	Float	Minimum numeric value
Maxval	Float	Maximum numeric value
Enum	Float*	Array of finite values ; dimension is equal to difference between Maxval and Minval

Type = PARAM_TYPE_BDSTATUS

Bit 0	Board is in power-fail status
Bit 1	Board has a firmware checksum error
Bit 2	Board has a calibration error on HV
Bit 3	Board has a calibration error on temperature
Bit 4	Board is in under-temperature status
Bit 5	Board is in over-temperature status
Bit 6…31	Reserved, forced to 0

No Properties available

```
CAENHVRESULT CAENHV_GetBdParam(
int                    handle,           // In
unsigned short         slotNum,          // In
const unsigned short *slotList,          // In
const char            *ParName,          // In
void                  *ParValList        // Out
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
SlotNum	The number of slots
SlotList	The list of slots; in case of SYX527, the MSByte indicates the crate in the cluster
ParName	Name of the parameter
ParValList	Returned parameters values

As an example, in this document we show the parameters which the user can specify for the **A1832** board. For the other boards, please refer to their user's manual.

Parameter Name	Type pointed by ParValList
BdStatus	Unsigned (Bitfield)
HVMax	Float
Temp	Float

```
CAENHVRESULT CAENHV_SetBdParam(
int                      handle,            // In
unsigned short           slotNum,           // In
const unsigned short *slotList,             // In
const char              *ParName,           // In
void                    *ParValue           // In
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
SlotNum	The number of slots
SlotList	The list of slots; in case of SYX527, the MSByte indicates the crate in the cluster
ParName	Name of the parameter
ParValue	New parameter value

```
CAENHVRESULT CAENHV_GetCrateMap(
int                      handle,      // In
unsigned short       *NrOfSlot,       // Out
unsigned short     **NrOfChList,      // Out
char               **ModelList,       // Out
char               **DescriptionList, // Out
unsigned short     **SerNumList,      // Out
unsigned char      **FmwRelMinList,   // Out
unsigned char      **FmwRelMaxList    // Out
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
NrOfSlot	How many slots
NrOfChlList	Number of channels; memory pointed by NrOfChList must be deallocated by the user
ModelList	Model of the board, i.e. “A1734”; Empty string if board not present; memory pointed by ModelList must be deallocated by the user
DescriptionList	Description of the board, i.e. “12 channels …”; memory pointed by DescriptionList must be deallocated by the user
SerNumList	Board Serial Number; memory pointed by SerNumList must be deallocated by the user
FmwRelMinList	LSByte of firmware release: 0 if rel. 1.0; memory pointed by FmwRelMinList must be deallocated by the user
FmwRelMaxList	MSByte of firmware release: 1 if rel. 1.0; memory pointed by FmwRelMaxList must be deallocated by the user

```
CAENHVRESULT CAENHV_GetExecCommList(
int                      handle,            // In
unsigned short           *NumComm           // Out
char                     **CommNameList     // Out
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
NumComm	Number of commands in the list
CommNameList	List of the possible commands to send to the system; memory pointed by CommNameList must be deallocated by the user

In the following table we show the list returned for the SYX527 Power Supply Systems:

Command Name	Description
Kill	Kill all channels
ClearAlarm	Clear Alarm
EnMsg	To be implemented
DisMsg	To be implemented
Format	To be implemented
RS232CmdOff	To be implemented

```
CAENHVRESULT CAENHV_ExecComm(
int                     handle,      // In
const char             *CommName     // In
);
```

Parameters	Description
SystemName	A string like “Systemx”
CommName	Name of the command: one from the previous list

```
CAENHVRESULT CAENHV_GetSysPropList(
int                     handle,            // In
unsigned short          *NumProp           // Out
char                    **PropNameList     // Out
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
NumProp	Number of properties in the list
PropNameList	List of the properties of one system; memory pointed by PropNameList must be deallocated by the user

In the following table we show the list returned for the SYx527 Power Supply Systems:

SY1527/2527		SY4527/5527	
Property Name	**Description**	**Property Name**	**Description**
Sessions	List Users connected to the system	Sessions	List Users connected to the system
ModelName	System name	ModelName	System name
SwRelease	System firmware release	SwRelease	System firmware release
GenSignCfg	GEN signal configuration	GenSignCfg	GEN signal configuration
FrontPanIn	System input status	FrontPanIn	System input status
FrontPanOut	System output status	FrontPanOu	System output status
ResFlagCfg	Reset flags configuration	ResFlagCfg	Reset flags configuration
ResFlag	To be implemented	ResFlag	To be implemented
HvPwSM	Power supply modules status	HvPwSM	Power supply modules status
FanStat	Fan status	HVFanStat	Fan status
ClkFreq	Clock frequency	ClkFreq	Clock frequency
HVClkConf	Clock configuration	HVClkConf	Clock configuration
IPAddr	System IP address	IPAddr	System IP address
IPNetMsk	System IP net mask	IPNetMsk	System IP net mask
IPGw	System IP gateway	IPGw	System IP gateway
RS232Par	RS232 parameters	SymbolicName	System symbolic name
CnetCrNum	CAENET crate number	PWCurrent	Power section current status
SymbolicName	System symbolic name	FrontPanOutLvl	I/O signals level
		CmdQueueStatus	Command queue status
		CPULoad	Status of CPU load
		MemoryStatus	Status of CPU memory
		HVFanSpeed	HV section fan speed
		PWFanStat	Power section Fan status
		PWVoltage	Power section voltage status

```
CAENHVRESULT CAENHV_GetSysPropInfo(
int                     handle,            // In
const char                *PropName,          // In
unsigned                   *PropMode,          // Out
unsigned                   *PropType           // Out
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
PropName	Name of the property whose value we want to know
PropMode	Mode of the property
PropType	Type of the property

In the following table we show the Mode and the Type of the properties of SYx527 Power Supply Systems:

SY1527/2527			**SY4527/5527**		
Property Name	**Property Mode**	**Property Type**	**Property Name**	**Property Mode**	**Property Type**
Sessions	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR	Sessions	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR
ModelName	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR	ModelName	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR
SwRelease	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR	SwRelease	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR
GenSignCfg	SYSPROP_MODE_RW	SYSPROP_TYPE_UINT2	GenSignCfg	SYSPROP_MODE_RDWR	SYSPROP_TYPE_UINT2
FrontPanIn	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_UINT2	FrontPanIn	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_UINT2
FrontPanOut	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_UINT2	FrontPanOut	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_UINT2
ResFlagCfg	SYSPROP_MODE_RW	SYSPROP_TYPE_UINT2	ResFlagCfg	SYSPROP_MODE_RDWR	SYSPROP_TYPE_UINT2
ResFlag	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_UINT2	ResFlag	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_UINT2
HvPwSM	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR	HvPwSM	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR
FanStat	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR	HVFanStat	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR
ClkFreq	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_INT2	ClkFreq	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_INT2
HVClkConf	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR	HVClkConf	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR
IPAddr	SYSPROP_MODE_RW	SYSPROP_TYPE_STR	IPAddr	SYSPROP_MODE_RDWR	SYSPROP_TYPE_STR
IPNetMsk	SYSPROP_MODE_RW	SYSPROP_TYPE_STR	IPNetMsk	SYSPROP_MODE_RDWR	SYSPROP_TYPE_STR
IPGw	SYSPROP_MODE_RW	SYSPROP_TYPE_STR	IPGw	SYSPROP_MODE_RDWR	SYSPROP_TYPE_STR
RS232Par	SYSPROP_MODE_RW	SYSPROP_TYPE_STR	RS232Par	SYSPROP_MODE_RDWR	SYSPROP_TYPE_STR
CnetCrNum	SYSPROP_MODE_RW	SYSPROP_TYPE_UINT2	FrontPanOutLevel	SYSPROP_MODE_RDWR	SYSPROP_TYPE_UINT2
SymbolicName	SYSPROP_MODE_RW	SYSPROP_TYPE_STR	SymbolicName	SYSPROP_MODE_RDWR	SYSPROP_TYPE_STR
			CommandQStatus	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_UINT2
			CPULoad	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR
			MemoryStatus	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR
			HVFanSpeed	SYSPROP_MODE_RDWR	SYSPROP_TYPE_UINT2
			PWFanStat	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR
			DummyReg	SYSPROP_MODE_RDWR	SYSPROP_TYPE_UINT2
			CMDExecMode	SYSPROP_MODE_RDWR	SYSPROP_TYPE_UINT2

```
CAENHVRESULT CAENHV_GetSysProp(
int                        handle,             // In
const char                *PropName,          // In
void                       *Result             // Out
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
PropName	Name of the property whose value we want to know
Result	Value of the property

```
CAENHVRESULT CAENHV_SetSysProp(
int                        handle,             // In
const char                *PropName,          // In
void                       *Set                // In
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
PropName	Name of the property whose value we want to set
Set	New Value of the property

The following functions:

CAENHV_SubscribeSystemParams
CAENHV_SubscribeBoardParams
CAENHV_SubscribeChannelParams
CAENHV_UnSubscribeSystemParams
CAENHV_UnSubscribeBoardParams
CAENHV_UnSubscribeChannelParams

allow to manage the event mode (see § 3): the user can add a list of system, board and channel items that through the "subscribe" functions, that return value codes as soon as their value is changed; items names must be separated with column ":". If the user wants to remove one parameter from event mode, than the "unsubscribe" functions have to be used.

```
CAENHVRESULT CAENHV_SubscribeSystemParams(
int             handle,                    // In
short           Port,                      // In
const char      *paramNameList,            // In
unsigned int    paramNum ,                 // In
char            *listOfResultCodes         // Out
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
Port	TCP/IP port of TCP server created for the event mode; see §3
paramNameList	List of system parameters
paramNum	Number of system parameters
listOfResultCodes	Returned values codes

```
CAENHVRESULT CAENHV_SubscribeBoardParams(
int                         handle,                    // In
short                       Port,                      // In
const unsigned short        slotIndex,                 // In
const char                  *paramNameList,            // In
unsigned int                paramNum ,                 // In
char                        *listOfResultCodes         // Out
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
Port	TCP/IP port of TCP server created for the event mode; see §3
slotIndex	Board slot
paramNameList	List of board parameters
paramNum	Number of board parameters
listOfResultCodes	Returned values codes

```
CAENHVRESULT CAENHV_SubscribeChannelParams(
int                         handle,             // In
short                       Port,               // In
const unsigned short        slotIndex,          // In
const unsigned short        chanIndex,          // In
const char                  *paramNameList,     // In
unsigned int                 paramNum ,          // In
char                         *listOfResultCodes  // Out
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
Port	TCP/IP port of TCP server created for the event mode; see §3
slotIndex	Board slot
chanIndex	Channel number
paramNameList	List of channel parameters
paramNum	Number of board parameters
listOfResultCodes	Returned values codes

```
CAENHVRESULT CAENHV_UnSubscribeSystemParams(
int handle,                  // In
short Port,                  // In
const char *paramNameList,   // In
unsigned int paramNum ,      // In
char *listOfResultCodes      // Out
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
Port	TCP/IP port of TCP server created for the event mode; see §3
paramNameList	List of system parameters
paramNum	Number of system parameters
listOfResultCodes	Returned values codes

```
CAENHVRESULT CAENHV_UnSubscribeBoardParams(
int handle,                         // In
short Port,                         // In
const unsigned short slotIndex,     // In
const char *paramNameList,          // In
unsigned int paramNum ,             // In
char *listOfResultCodes             // Out
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
Port	TCP/IP port of TCP server created for the event mode; see §3
slotIndex	Board slot
paramNameList	List of board parameters
paramNum	Number of board parameters
listOfResultCodes	Returned values codes

```
CAENHVRESULT CAENHV_UnSubscribeChannelParams(
int                     handle,                          // In
short                   Port,                            // In
const unsigned short    slotIndex,                       // In
const unsigned short    chanIndex,                       // In
const char              *paramNameList,                  // In
unsigned int            paramNum ,                       // In
char                    *listOfResultCodes               // Out
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
Port	TCP/IP port of TCP server created for the event

	mode; see §3
slotIndex	Board slot
chanIndex	Channel number
paramNameList	List of channel parameters
paramNum	Number of board parameters
listOfResultCodes	Returned values codes

The following funcitons:

CAENHV_GetEventData

allows to receive data from the mainframe through the socket created by the TCP connection

CAENHV_FreeEventData

Deallocates the memory for the data received from the mainframe (allocated within the library).

```
CAENHVRESULT CAENHV_GetEventData(
int                             sck,              // In
CAENHV_SYSTEMSTATUS_t           *SysStatus,       // Out
CAENHVEVENT_TYPE_t              **EventData,      // Out
unsigned int                    *DataNumber       // Out
);
```

Parameters	Description
sck	Socket
SysStatus	Connection status
EventData	Changed items
DataNumber	Number of items

```
CAENHVRESULT CAENHV_FreeEventData(
CAENHVEVENT_TYPE_t      **ListOfItemsData         // In
);
```

Parameters	Description
ListOfItemsData	List of items received

Property	Property Type	Description

IDValue_t	union	char StringValue[1024];
		float FloatValue;
		int IntValue

CAENHV_ID_TYPE_t	enum	PARAMETER = 0,
		ALARM = 1,
		KEEPALIVE = 2

CAENHVEVENT_TYPE	struct	char Type;
		char ItemID[64];
		char Lvalue[4];
		char Tvalue[256];

CAENHVEVENT_TYPE_t	struct	int SystemHandle;
		long BoardIndex;
		long ChannelIndex;
		char ItemID[20];

CAENHV_SYSTEM_TYPE_t	enum	SY1527 = 0,
		SY2527 = 1,
		SY4527 = 2,
		SY5527 = 3,
		V65XX = 4,

CAENHV_EVT_STATUS_t	enum	SYNC = 0,
		ASYNC = 1,
		UNSYNC = 2,
		NOTAVAIL = 3

CAENHV_SYSTEMSTATUS_t	struct	CAENHV_EVT_STATUS_t System;
		CAENHV_EVT_STATUS_t Board[16];

Possible values of CAENHVRESULT

Value	**Description**
0x0	No error
0x1	Operating system error
0x2	Writing error
0x3	Reading error
0x4	Time out error
0x5	Command Front End application is down
0x6	Communication with system not yet connected by a Login command
0x7	Execute Command not yet implemented
0x8	Get Property not yet implemented
0x9	Set Property not yet implemented
0xa	Communication with RS232 not yet implemented
0xb	User memory not sufficient
0xc	Value out of range
0xd	Property not yet implemented
0xe	Property not found
0xf	Command not found
0x10	Not a Property
0x11	Not a reading Property
0x12	Not a writing Property
0x13	Not a Command
0x14	configuration change
0x15	Parameter’s Property not found
0x16	Parameter not found
0x17	No data present
0x18	Device already open
0x19	To Many devices opened
0x1A	Function Parameter not valid
0x1B	Function not available for the connected device
0x1C	SOCKET ERROR
0x1D	COMMUNICATION ERROR
0x1E	NOT YET IMPLEMENTED
0x1000+1	CONNECTED
0x1000+2	NOTCONNECTED
0x1000+3	OS
0x1000+4	LOG IN FAILED
0x1000+5	LOG OUT FAILED
0x1000+6	LINK NOT SUPPORTED

Note: negative error values are errors coming from the Power Supply.

WARNING!

The following functions are deprecated:

The user of this library must define a string label (HV P.S. Name) for every HV power supply to control.
The string is inserted in a table like that below:

HV P.S. Name	Connection Type	Parameters
System0	CAENET	A303 IOAddr, Crate #n
System1	CAENET	A1303 Id, Crate #m
System2	TCP/IP	IP #a
System3	TCP/IP	IP #b
System4	USB	Link #x, board #y, VME base address
System5	CONET	Link #w, board #z, VME base address

If the string identifies a CAENET controllable Power Supply, the CAEN HV Wrapper must call the procedures in the relevant interface which prepares the correct CAENET packet to pass to HSCAENETLib

```
CAENHVRESULT CAENHVInitSystem(
const char      *SystemName,        // In
int             LinkType,           // In
void            *Arg,               // In
const char      *UserName,          // In
const char      *Password           // In
);
```

Parameters	Description
SystemName	A string like “Systemx”
LinkType	0 = TCP/IP 1 = RS232 2 = HS CAENET 3 = USB 2.0 4 = CONET Optical Link
Arg	Points to a char of the type “A303_IOAddr_CrNum” or “A1303_Id_CrNum” when linkType is 2; points to a char IP when linkType is 0. If linkType is 3 or 4, then Arg is in the form x_y_ba, where x is link number, y is bdnumber and ba is baseaddress (HEX)[3]
UserName	A string containing the User’s Name; has meaning only for SYX527
Password	A string containing the User’s Password; has meaning only for SYX527

This is the first function with parameter `SystemName` to call, and it must be called for all the HV power supplies the user wants to control; if `linkType` is 2, it executes a CAENET 0 command to see which type of high voltage system is connected to the given CrNum. The Arg parameter, in this case, is formed by three parts: the name of the board (A303 or A1303), the IO port address in the A303 case or an identifier starting from 0 for the A1303 selection (multiple A1303 boards can be used in the same PC) and the crate number of the system in the chain.
If `linkType` is 0, it executes a login command (SY1527 or SY2527 is assumed).
If linkType is 3 or 4, the VME Power Supply Boards are accessed via CAEN VME USB Bridge or Optical Link Bridge respectively; in order to do this, the CAENComm library shall be installed.

```
CAENHVRESULT CAENHVDeinitSystem(
const char   *SystemName             // In
);
```

Parameters	Description
SystemName	A string like “Systemx”

This is the last function with parameter `SystemName` to call, and it must be called for all the HV power supplies the user wants to control.

[3] See CAENComm library documentation

```
char            *CAENHVGetError(
const char      *SystemName         // In
);
```

Parameters	Description
SystemName	A string like “Systemx”

This function returns a string describing the last error occurred during communication with system “Systemx”

```
char   *CAENHVLibSwRel();
```

Returns	Description
SoftwareRel	The Release of CAEN HV Wrapper, in the form “2.7-1.4” where the first 2 digits are the CAEN HV Wrapper version while the second 2 digits are the HSCAENETLib version.

```
CAENHVRESULT CAENHVGetChName(
const char            *SystemName,                // In
unsigned short         slot,                      // In
unsigned short         ChNum,                      // In
const unsigned short  *ChList,                     // In
char                 (*ChNameList)[MAX_CH_NAME]  // Out
);
```

Parameters	Description
SystemName	A string like “Systemx”
Slot	The slot; in case of SYX527, the MSByte indicates the crate in the cluster
ChNum	Number of channels in the list
ChList	List of channels
ChNameList	List of returned channels names.

```
CAENHVRESULT CAENHVSetChName(
const char             *SystemName,       // In
unsigned short          slot,             // In
unsigned short          ChNum,            // In
const unsigned short   *ChList,           // In
const char             *ChName            // In
);
```

Parameters	Description
SystemName	A string like “Systemx”
Slot	The slot; in case of SYX527, the MSByte indicates the crate in the cluster
ChNum	Number of channels in the list
ChList	List of channels
ChName	New name of the channels

```
CAENHVRESULT CAENHVGetChParamInfo(
const char            *SystemName,       // In
unsigned short        slot,              // In
unsigned short        Ch,                // In
char                  **ParNameList      // Out
);
```

Parameters	Description
SystemName	A string like “Systemx”
Slot	The slot; in case of SYX527, the MSByte indicates the crate in the cluster
Ch	The channel
ParNameList	List of the names of the parameters of channel Ch; the list is ended by the NUL string; memory pointed by ParNameList must be deallocated by the user

As an example, in this document we show the list returned for the **A1832** board. For the list relative to the other boards, please refer to their user’s manual.

Parameter Name	Description
V0Set	Set V0 voltage limit
I0Set	Set I0 current limit
V1Set	Set V1 voltage limit
I1Set	Set I1 current limit
Rup	Set ramp-up rate
RDWn	Set ramp-down rate
Trip	Set trip time
SVMax	Set software voltage limit
Vmon	Voltage monitor
Imon	Current monitor
Status	Channel status
Pw	Power ON/OFF
Pon	Power ON options
PDwn	Power down options
TripInt	Internal trip connections
TripExt	External trip connections

```
CAENHVRESULT CAENHVGetChParamProp(
const char              *SystemName,        // In
unsigned short          slot,               // In
unsigned short          Ch,                 // In
const char              *ParName,           // In
const char              *PropName,          // In
void                    *retval             // Out
);
```

Parameters	Description
SystemName	A string like “Systemx”
Slot	The slot; in case of SYX527, the MSByte indicates the crate in the cluster
Ch	The channel
ParName	The name of the parameter whose property we want to know; possible value: “Vmon”
PropName	The name of the property whose value we want to know; possible value: “Maxval”
Retval	The value of the property

This function permits to know a property of a given parameter.
For every parameter two properties are available:
the property called “Type” which can assume the following 4 values (of type unsigned long): PARAM_TYPE_NUMERIC, PARAM_TYPE_ONOFF, PARAM_TYPE_CHSTATUS and PARAM_TYPE_BDSTATUS.
the property called “Mode” which can assume the following 3 values (of type unsigned long): PARAM_MODE_RDONLY, PARAM_MODE_WRONLY, PARAM_MODE_RDWR.

Depending on the values above, other properties exist following the relations shown in the next table:

Type = PARAM_TYPE_NUMERIC, **Value** = float

Property	Property Type	Description
Minval	Float	Minimum numeric value
Maxval	Float	Maximum numeric value
Unit	Unsigned short	Index to this list of Engineering Units: PARAM_UN_NONE, PARAM_UN_AMPERE, PARAM_UN_VOLT, PARAM_UN_WATT, PARAM_UN_CELSIUS, PARAM_UN_HERTZ, PARAM_UN_BAR, PARAM_UN_VPS, PARAM_UN_SECOND, PARAM_UN_RPM, PARAM_UN_COUNT
Exp	Short	+3 (Kilo), +6 (Mega), -3 (milli), -6 (micro)

Type = PARAM_TYPE_ONOFF, **Value** = unsigned (0, 1)

Property	Property Type	Description
Onstate	Char *	String indicating the Onstate, i.e. “On” or “Enabled”
Offstate	Char *	String indicating the Offstate, i.e. “Off” or “Disabled”

Type = PARAM_TYPE_CHSTATUS, **Value** = the following bitfield

Bit 0	Channel is on
Bit 1	Channel is ramping up
Bit 2	Channel is ramping down
Bit 3	Channel is in overcurrent
Bit 4	Channel is in overvoltage
Bit 5	Channel is in undervoltage
Bit 6	Channel is in external trip
Bit 7	Channel is in max V
Bit 8	Channel is in external disable
Bit 9	Channel is in internal trip
Bit 10	Channel is in calibration error
Bit 11	Channel is unplugged
Bit 12	Channel is under current
Bit 13...31	Reserved, forced to 0

No Properties available

Type = PARAM_TYPE_BDSTATUS

Bit 0	Board is in power-fail status
Bit 1	Board has a firmware checksum error
Bit 2	Board has a calibration error on HV
Bit 3	Board has a calibration error on temperature
Bit 4	Board is in under-temperature status
Bit 5	Board is in over-temperature status
Bit 6...31	Reserved, forced to 0

No Properties available

```
CAENHVRESULT CAENHVGetChParam(
const char          *SystemName,        // In
unsigned short       slot,              // In
const char           *ParName,           // In
unsigned short        ChNum,             // In
const unsigned short *ChList,            // In
void                 *ParValList         // Out
);
```

Parameters	Description
SystemName	A string like “Systemx”
Slot	The slot; in case of SYX527, the MSByte indicates the crate in the cluster
ParName	Name of the parameter
ChNum	Number of channels in the list
ChList	List of channels
ParValList	List of returned parameters values

As an example, in this document we show the parameters which the user can specify for the **A1832** board. For the other boards, please refer to their user’s manual.

Parameter Name	Type pointed by ParValList
V0Set	Float
I0Set	Float
V1Set	Float
I1Set	Float
Rup	Float
RDWn	Float
Trip	Float
SVMax	Float
Vmon	Float
Imon	Float
Status	Unsigned (Bitfield)
Pw	Unsigned (Boolean)
Pon	Unsigned (Boolean)
PDwn	Unsigned (Boolean)
TripInt	Unsigned
TripExt	Unsigned

```
CAENHVRESULT CAENHVSetChParam(
const char            *SystemName,       // In
unsigned short         slot,             // In
const char            *ParName,          // In
unsigned short         ChNum,            // In
const unsigned short  *ChList,           // In
void                  *ParValue          // In
);
```

Parameters	Description
SystemName	A string like “Systemx”
Slot	The slot; in case of SYX527, the MSByte indicates the crate in the cluster
ParName	Name of the parameter
ChNum	Number of channels in the list
ChList	List of channels
ParValue	New parameter value

As an example, in this document we show the parameters which the user can specify for the **A1832** board. For the other boards, please refer to their user’s manual.

Parameter Name	Type pointed by ParValList
V0Set	Float
I0Set	Float
V1Set	Float
I1Set	Float
Rup	Float
RDWn	Float
Trip	Float
SVMax	Float
Pw	Unsigned (Boolean)
Pon	Unsigned (Boolean)
PDwn	Unsigned (Boolean)
TripInt	Unsigned
TripExt	Unsigned

```
CAENHVRESULT CAENHVTestBdPresence(
const char      *SystemName,        // In
unsigned short  slot,               // In
short           *NrOfCh,            // Out
char            *Model,             // Out
char            *Description,       // Out
unsigned short  *SerNum,            // Out
unsigned char   *FmwRelMin,         // Out
unsigned char   *FmwRelMax          // Out
);
```

Parameters	Description
SystemName	A string like “Systemx”
Slot	The slot; in case of SYX527, the MSByte indicates the crate in the cluster
NrOfCh	Number of channels in the board
Model	Model of the board, i.e. “A1734”; NULL if board not present
Description	Description of the board, i.e. “12 channels …”
SerNum	Board Serial Number
FmwRelMin	LSByte of firmware release: 0 if rel. 1.0
FmwRelMax	MSByte of firmware release: 1 if rel. 1.0

```
CAENHVRESULT CAENHVGetBdParamInfo(
const char             *SystemName,        // In
unsigned short          slot,              // In
char                   **ParNameList       // Out
);
```

Parameters	Description
SystemName	A string like “Systemx”
Slot	The slot; in case of SYX527, the MSByte indicates the crate in the cluster
ParNameList	List of the names of the parameters of the board; memory pointed by ParNameList must be deallocated by the user

As an example, in this document we show the list returned for the **A1832** board. For the list relative to the other boards, please refer to their user’s manual.

Parameter Name	Description
BdStatus	Board status
HVMax	Hardware voltage limit
Temp	Board temperature

```
CAENHVRESULT CAENHVGetBdParamProp(
const char              *SystemName, // In
unsigned short          slot,              // In
const char              *ParName,          // In
const char              *PropName,         // In
void                    *retval            // Out
);
```

Parameters	Description
SystemName	A string like “Systemx”
Slot	The slot; in case of SYX527, the MSByte indicates the crate in the cluster
ParName	The name of the parameter whose property we want to know; possible value: “Hvmax”
PropName	The name of the property whose value we want to know; possible value: “MaxVal”
Retval	The value of the property

This function permits to know a property of a given parameter.
For every parameter two properties are available:
the property called “Type” which can assume the following 4 values (of type unsigned long): PARAM_TYPE_NUMERIC, PARAM_TYPE_ONOFF, PARAM_TYPE_CHSTATUS and PARAM_TYPE_BDSTATUS.
the property called “Mode” which can assume the following 3 values (of type unsigned long): PARAM_MODE_RDONLY, PARAM_MODE_WRONLY, PARAM_MODE_RDWR.
Depending on the values above, other properties exist following the relations shown in the next table:

Type = PARAM_TYPE_NUMERIC, **Value** = float

Property	Property Type	Description
Minval	Float	Minimum numeric value
Maxval	Float	Maximum numeric value
Unit	Unsigned short	Index to this list of Engineering Units: PARAM_UN_NONE, PARAM_UN_AMPERE, PARAM_UN_VOLT, PARAM_UN_WATT, PARAM_UN_CELSIUS, PARAM_UN_HERTZ, PARAM_UN_BAR, PARAM_UN_VPS, PARAM_UN_SECOND, PARAM_UN_RPM, PARAM_UN_COUNT
Exp	Short	+3 (Kilo), +6 (Mega), -3 (milli), -6 (micro)

Type = PARAM_TYPE_ONOFF, **Value** = unsigned (0, 1)

Property	Property Type	Description
Onstate	Char *	String indicating the Onstate, i.e. “On” or “Enabled”
Offstate	Char *	String indicating the Offstate, i.e. “Off” or “Disabled”

Type = PARAM_TYPE_CHSTATUS, **Value** = the following bitfield

Bit 0	Channel is on
Bit 1	Channel is ramping up
Bit 2	Channel is ramping down
Bit 3	Channel is in overcurrent
Bit 4	Channel is in overvoltage
Bit 5	Channel is in undervoltage
Bit 6	Channel is in external trip
Bit 7	Channel is in max V

Bit 8	Channel is in external disable
Bit 9	Channel is in internal trip
Bit 10	Channel is in calibration error
Bit 12	Channel is under current
Bit 13…31	Reserved, forced to 0

No Properties available

Type = PARAM_TYPE_BDSTATUS

Bit 0	Board is in power-fail status
Bit 1	Board has a firmware checksum error
Bit 2	Board has a calibration error on HV
Bit 3	Board has a calibration error on temperature
Bit 4	Board is in under-temperature status
Bit 5	Board is in over-temperature status
Bit 6…31	Reserved, forced to 0

No Properties available

```
CAENHVRESULT CAENHVGetBdParam(
const char            *SystemName,         // In
unsigned short         slotNum,            // In
const unsigned short *slotList,            // In
const char            *ParName,            // In
void                  *ParValList          // Out
);
```

Parameters	Description
SystemName	A string like "Systemx"
SlotNum	The number of slots
SlotList	The list of slots; in case of SYX527, the MSByte indicates the crate in the cluster
ParName	Name of the parameter
ParValList	Returned parameters values

As an example, in this document we show the parameters which the user can specify for the **A1832** board. For the other boards, please refer to their user's manual.

Parameter Name	Type pointed by ParValList
BdStatus	Unsigned (Bitfield)
HVMax	Float
Temp	Float

```
CAENHVRESULT CAENHVSetBdParam(
const char              *SystemName,       // In
unsigned short           slotNum,          // In
const unsigned short    *slotList,         // In
const char              *ParName,          // In
void                     *ParValue         // In
);
```

Parameters	Description
SystemName	A string like "Systemx"
SlotNum	The number of slots
SlotList	The list of slots; in case of SYX527, the MSByte indicates the crate in the cluster
ParName	Name of the parameter
ParValue	New parameter value

```
CAENHVRESULT CAENHVGetGrpComp(
const char              *SystemName,        // In
unsigned short          group,              // In
unsigned short          *NrOfCh,            // Out
unsigned long           **ChList            // Out
);
```

Parameters	Description
SystemName	A string like “Systemx”
Group	The group
NrOfCh	How many channels
ChList	Which channels (slot, chinslot). Memory pointed by ChList must be deallocated by the user.

Note: this function is not implemented yet.

```
CAENHVRESULT CAENHVAddChToGrp(
const char                    *SystemName,        // In
unsigned short                group,              // In
unsigned short                NrOfCh,             // In
const unsigned long           *ChList             // In
);
```

Parameters	Description
SystemName	A string like “Systemx”
Group	The group
NrOfCh	How many channels
ChList	Which channels (slot, chinslot)

Note: this function is not implemented yet.

```
CAENHVRESULT CAENHVRemChToGrp(
const char                    *SystemName,        // In
unsigned short                group,              // In
unsigned short                NrOfCh,             // In
const unsigned long           *ChList             // In
);
```

Parameters	Description
SystemName	A string like “Systemx”
Group	The group
NrOfCh	How many channels
ChList	Which channels (slot, chinslot)

Note: this function is not implemented yet.

```
CAENHVRESULT CAENHVGetGrpParam(
const char              *SystemName,        // In
unsigned short           Group,             // In
unsigned short           NrOfPar,           // In
const unsigned char    **ParNameList,       // In
void                    *ParValList         // Out
);
```

Parameters	Description
SystemName	A string like “Systemx”
Group	The group
NrOfPar	How many parameters
ParNameList	Which Parameters
ParValList	List of returned parameters values

Note: this function is not implemented yet.

```
CAENHVRESULT CAENHVSetGrpParam(
const char            *SystemName,        // In
unsigned short         Group,             // In
const unsigned char   *ParName,           // In
void                  *ParVal             // In
);
```

Parameters	Description
SystemName	A string like “Systemx”
Group	The group
ParName	Which Parameter
ParVal	New parameter value

Note: this function is not implemented yet.

```
CAENHVRESULT CAENHVGetCrateMap(
const char            *SystemName,        // In
unsigned short        *NrOfSlot,          // Out
unsigned short        **NrOfChList,       // Out
char                  **ModelList,        // Out
char                  **DescriptionList,  // Out
unsigned short        **SerNumList,       // Out
unsigned char         **FmwRelMinList,    // Out
unsigned char         **FmwRelMaxList     // Out
);
```

Parameters	Description
SystemName	A string like “Systemx”
NrOfSlot	How many slots
NrOfChlList	Number of channels; memory pointed by NrOfChList must be deallocated by the user
ModelList	Model of the board, i.e. “A1734”; Empty string if board not present; memory pointed by ModelList must be deallocated by the user
DescriptionList	Description of the board, i.e. “12 channels …”; memory pointed by DescriptionList must be deallocated by the user
SerNumList	Board Serial Number; memory pointed by SerNumList must be deallocated by the user
FmwRelMinList	LSByte of firmware release: 0 if rel. 1.0; memory pointed by FmwRelMinList must be deallocated by the user
FmwRelMaxList	MSByte of firmware release: 1 if rel. 1.0; memory pointed by FmwRelMaxList must be deallocated by the user

```
CAENHVRESULT CAENHVGetExecCommList(
const char              *SystemName,        // In
unsigned short          *NumComm            // Out
char                    **CommNameList      // Out
);
```

Parameters	Description
SystemName	A string like “Systemx”
NumComm	Number of commands in the list
CommNameList	List of the possible commands to send to the system; memory pointed by CommNameList must be deallocated by the user

In the following table we show the list returned for the SYX527 Power Supply Systems:

Command Name	Description
Kill	Kill all channels
ClearAlarm	Clear Alarm
EnMsg	To be implemented
DisMsg	To be implemented
Format	To be implemented
RS232CmdOff	To be implemented

```
CAENHVRESULT CAENHVExecComm(
const char              *SystemName,        // In
const char              *CommName           // In
);
```

Parameters	Description
SystemName	A string like “Systemx”
CommName	Name of the command: one from the previous list

```
CAENHVRESULT CAENHVGetSysPropList(
const char              *SystemName,        // In
unsigned short          *NumProp            // Out
char                    **PropNameList      // Out
);
```

Parameters	Description
SystemName	A string like “Systemx”
NumProp	Number of properties in the list
PropNameList	List of the properties of one system; memory pointed by PropNameList must be de allocated by the user

In the following table we show the list returned for the SY1527-2527 Power Supply Systems:

Property Name	Description
Sessions	List Users connected to the system
ModelName	System name
SwRelease	System firmware release
GenSignCfg	GEN signal configuration
FrontPanIn	System input status
FrontPanOut	System output status
ResFlagCfg	Reset flags configuration
ResFlag	To be implemented
HvPwSM	Power supply modules status
FanStat	Fan status
ClkFreq	Clock frequency
HVClkConf	Clock configuration
IPAddr	System IP address
IPNetMsk	System IP net mask
IPGw	System IP gateway
RS232Par	RS232 parameters
CnetCrNum	CAENET crate number
SymbolicName	System symbolic name

```
CAENHVRESULT CAENHVGetSysPropInfo(
const char              *SystemName,        // In
const char             *PropName,           // In
unsigned                *PropMode,          // Out
unsigned                *PropType           // Out
);
```

Parameters	Description
SystemName	A string like “Systemx”
PropName	Name of the property whose value we want to know
PropMode	Mode of the property
PropType	Type of the property

In the following table we show the Mode and the Type of the properties of SY1527-2527 Power Supply Systems:

Property Name	**Property Mode**	**Property Type**
Sessions	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR
ModelName	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR
SwRelease	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR
GenSignCfg	SYSPROP_MODE_RW	SYSPROP_TYPE_UINT2
FrontPanIn	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_UINT2
FrontPanOut	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_UINT2
ResFlagCfg	SYSPROP_MODE_RW	SYSPROP_TYPE_UINT2
ResFlag	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_UINT2
HvPwSM	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR
FanStat	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR
ClkFreq	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_INT2
HVClkConf	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR
IPAddr	SYSPROP_MODE_RW	SYSPROP_TYPE_STR
IPNetMsk	SYSPROP_MODE_RW	SYSPROP_TYPE_STR
IPGw	SYSPROP_MODE_RW	SYSPROP_TYPE_STR
RS232Par	SYSPROP_MODE_RW	SYSPROP_TYPE_STR
CnetCrNum	SYSPROP_MODE_RW	SYSPROP_TYPE_UINT2
SymbolicName	SYSPROP_MODE_RW	SYSPROP_TYPE_STR

```
CAENHVRESULT CAENHVGetSysProp(
const char              *SystemName,        // In
const char             *PropName,           // In
void                    *Result          // Out
);
```

Parameters	**Description**
SystemName	A string like “Systemx”
PropName	Name of the property whose value we want to know
Result	Value of the property

```
CAENHVRESULT CAENHVSetSysProp(
const char              *SystemName,        // In
const char             *PropName,           // In
void                    *Set                // In
);
```

Parameters	**Description**
SystemName	A string like “Systemx”
PropName	Name of the property whose value we want to set
Set	New Value of the property

```
CAENHVRESULT CAENHVCaenetComm (
const char         *SystemName,       // In
unsigned short     Crate,             // In
unsigned short     Code,              // In
unsigned short     NrWCode,           // In
unsigned short     *Wcode,            // In
short              *Result,           // Out
unsigned short     *NrOfData,         // Out
unsigned short     **Data             // Out
                    );
```

Parameters	Description
SystemName	A string like “Systemx”
Crate	System's crate number to send commands
Code	Code of command
NrWCode	nr. Of additional word code
Wcode	additional word code
Result	caenet error code
NrOfData	nr. Of data
Data	response to caenet code (without caenet error code). Memory pointed by Data must be deallocated by the user

The following functions:

CAENHV_Subscribe
CAENHV_UnSubscribe

allow to manage the event mode (see §3) in a single command: the user can add a list of system, board and channel parameters that through the “subscribe” function that return value codes as soon as their value is changed; the difference is that instead of the items list, a list of strings must be passed, with the following syntax:

System item: PowerSupplyName.Itemname
Board item: PowerSupplyName.BoardXX.itemname
Channel item: PowerSupplyName.BoardXX.ChanYYY.Itemname

Strings must be separated with column “:”
If the user wants to remove one parameter from event mode, than the “unsubscribe” function have to be used.

```
CAENHVRESULT CAENHV_Subscribe (
int             handle,                // In
 short          Port,                  // In
ushort          NrOfItems,             // In
const char      *ListOfItems,          // In
char            *ListofResultCodes     // Out
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
Port	TCP/IP port of TCP server created for the event mode; see §3
NrOfItems	Number of passed items
ListOfItems	List of passed items
ListofResultCodes	Returned values codes

```
CAENHVRESULT CAENHV_UnSubscribe(
int             handle,                // In
short           Port,                  // In
ushort          NrOfItems,             // In
const char      *ListOfItems,          // In
char            *ListofResultCodes     // Out
);
```

Parameters	Description
handle	Handle returned by the CAENHV_InitSystem function
Port	TCP/IP port of TCP server created for the event mode; see §3
NrOfItems	Number of passed items
ListOfItems	List of removed items
ListofResultCodes	Returned values codes

3. Event Mode

The Event Mode can be used alternately (or in conjunction) to the polling mode for retrieving data from SY4527/5527.
In Event Mode the system will send to the connected software the data, whenever the latter have undergone a change, or send (periodically) a keep-alive message in the case in which there have been no changes.
To use the Event Mode, it is necessary to create within the used software a TCP server, which will wait for the arrival of connections on a port chosen by the user; the same port that is passed as the second parameter to the functions:
CAENHV_SubscribeSystemParams

CAENHV_SubscribeBoardParams

CAENHV_SubscribeChannelParams

CAENHV_UnSubscribeSystemParams

CAENHV_UnSubscribeBoardParams

CAENHV_UnSubscribeChannelParams

The connection is established from the system to the PC where the software runs, on the return from the first successful subscription, therefore it is necessary to check that no firewall blocks incoming connections on that port.
Within the body of the function that manages the connected client then will be necessary to make a loop in which the function CAENHV_GetEventData is called, in order to retrieve data from the created socket.

This is an example of code of client management:

```
void* ClientHandling(void *arg)
{
    // socket descriptor
    int sock=(int)(*arg);
    //! waiting power supply for data loop
    while(1) {
            unsigned int itmCnt;
            CAENHVEVENT_TYPE_t *recvItem=NULL;
            CAENHV_SYSTEMSTATUS_t stat;
            int result=CAENHV_GetEventData(sock,&stat,&recvItem,&itmCnt);
            if (result!=CAENHV_OK) {
                    //! we assume we lost connection with power supply
                    //! we can exit thread;
            }

            for(unsigned int k=0;k<itmCnt;k++) {
                    switch (recvItem[k].Type)
                    {
                    case EVENTTYPE_PARAMETER:
                            {
                                    // handle parameter update
                            }
                            break;

                    case EVENTTYPE_ALARM:
                            {
                                    // handle alert
                            }
                            break;

                    case EVENTTYPE_KEEPALIVE:
```

```
                                        {
                                                // handle keepalive
                                        }
                                        break;
                                }
                        }
                        if(recvItem)
                                CAENHV_FreeEventData(&recvItem);
                }

                return;
        }
```

4. SY127 and SY527 Interface

The implementation of these interfaces doesn’t impact on the definition of the procedures of CAEN HV Wrapper (the pubic side must be independent by the Power Supply model), so it is not necessary to describe them here.

5. Support

Our Software Support Group is available for questions, support and any other software related issue concerning CAEN Power Supplies. Moreover, a newsletter on CAEN Software issues (CAEN SOFTWARE NEWS) will be periodically sent via e-mail to all subscribers to our mailing list. For software support and subscription to the free newsletter send an e-mail to **support.computing@caen.it**.
Don’t forget to visit our Web site: **http://www.caen.it/** for the latest news.

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.	CAEN GmbH	CAEN Technologies, Inc.
Via Vetraia, 11	Klingenstraße 108	1140 Bay Street - Suite 2 C
55049 Viareggio	D-42651 Solingen - Germany	Staten Island, NY 10305
Italy	Phone +49 (0)212 254 4077	USA
Tel. +39.0584.388.398	Fax +49 (0)212 25 44079	Tel. +1.718.981.0401
Fax +39.0584.388.959	Mobile +49 (0)151 16 548 484	Fax +1.718.556.9185
info@caen.it	info@caen-de.com	info@caentechnologies.com
www.caen.it	www.caen-de.com	www.caentechnologies.com
	CAEN GmbH	

CAEN Tools for Discovery **Electronic Instrumentation**

Manual - CAEN HV Wrapper Library rev. 10 - 13 November 2013