CAEN HV Wrapper Library

Rev. 16 - 10 September 2019

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
29 July 2014	11	Discontinued support for older systems
15 April 2015	12	Updated CAENHV_InitSystem, Event Data Items, Event Mode description
21 June 2016	13	Updated InitSystem, GetBdParamProp, GetChParamProp
31 March 2017	14	Updated GetBdParamInfo
30 June 2017	15	Updated InitSystem
10 September 2019	16	Updated InitSystem and Introduction

Symbols, abbreviated terms and notation

T.B.D.

Reference Document

SY4527 User's Manual
V6533 User's Manual
N1470 User's Manual
N568E User's Manual
DT55xx User's Manual
VME8200 User's Manual
NDT14xx User's Manual
DT55xxE User's Manual

© CAEN SpA – 2011

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.

CAEN HV Wrapper is logically located between a higher level application and the lower layer, as shown in the scheme below:

CAEN HV Wrapper API					
USB / OPTICAL LINK		*TCP / IP*			*USB VCP / ACM*
V65xx[1]	DT55xx[2]	SYx527	VME8x00	N568E NDT14xx R14xx DT14xx N1570 DT55xxE	N14xx

---

[1] V65xx power supplies require also CAENComm library installed
[2] DT55xx power supplies require also CAENComm library installed

2. Communication Support Interface

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

Any time a connection with a power supply is performed, the first function to call is CAENHV_InitSystem (see below). Right then, it is necessary to call CAENHV_GetCrateMap, in order to allow the used Client software to create the required data structures to communicate with the device.

When using the SYx527 system, it must be noticed that it implements a 30 seconds communication timeout; therefore the used Client software must implement a keep alive loop, in order to have not the communication shut: for example, a suggestion is to call CAENHV_GetSysProp to acknowledge the SwRelease parameter, every 15 seconds.
If the communication is shut, whether due to timeout or for any other reason, prior to re-establish it, it is necessary to de-initialize the library data structures, by calling the CAENHV_DeinitSystem funtcion

Description of the functions

InitSystem

```
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	Definition	
system	The type of the system to connect with	
	SY1527	0
	SY2527	1
	SY4527	2
	SY5527	3
	V65XX	5
	N-NDT14xx R14xx DT14xx N1570	6
	VME CRATE	7
	N568E	8
	DT55XX	9
	FTK	10
	DT55XXE	11
	N1068	12
LinkType	0 = TCP/IP 3 = USB 2.0 4 = Optical Link 5 = USB VCP	
Arg	If LinkType is 0, points to: SYx527 a char IP; N568E to IP_lbusaddress 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.[3] 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 to be called any time a connection with a device is performed.

DeinitSystem

```
CAENHVRESULT CAENHV_DeinitSystem(
```

---

[3] See CAENComm library documentation

```
int        handle        // In
);
```

Parameters	Definition
handle	Handle returned by the CAENHV_InitSystem function

This is the function to be called when a connection with a device is shut.

ExecComm

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

Parameters	Definition
SystemName	A string like "Systemx"
CommName	Name of the command: one from the previous list

GetBdParam

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

Parameters	Definition
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

GetBdParamInfo

```
CAENHVRESULT CAENHV_GetBdParamInfo(

int                   handle,            // In
ushort                slot,              // In
char                  **ParNameList,     // Out
  );
```

Parameters	Definition
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 parameters of the board; memory pointed by ParNameList must be deallocated by 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	Definition

BdStatus	Board status
HVMax	Hardware voltage limit
Temp	Board temperature

GetBdParamProp

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

Parameters	Definition
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

Name	Type	Definition
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)

Name	Type	Definition
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

Name	Type	Definition
Minval	Float	Minimum numeric value
Maxval	Float	Maximum numeric value
Enum	Char**	Array of strings (string means an array of char); the number of strings is equal to difference between Maxval and Minval. The max string length is 15 bytes.

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

GetChName

```
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	Definition
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.

GetChParam

```
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	Definition

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

GetChParamInfo

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

Parameters	Definition
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

GetChParamProp

```
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	Definition

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

Name	Type	Definition
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)

Name	Type	Definition
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

Name	Type	Definition
Minval	Float	Minimum numeric value
Maxval	Float	Maximum numeric value
Enum	Char**	Array of strings (string means an array of char); the number of strings is equal to difference between Maxval and Minval. The max string length is 15 bytes.

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

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	Definition
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

GetCrateMap

```
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	Definition
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

GetExecCommList

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

Parameters	Definition
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	Definition
Kill	Kill all channels
ClearAlarm	Clear Alarm
EnMsg	To be implemented
DisMsg	To be implemented
Format	To be implemented
RS232CmdOff	To be implemented

GetSysProp

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

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

GetSysPropInfo

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

Parameters	Definition
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:

SY4527/5527		
Name	**Mode**	**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_RDWR	SYSPROP_TYPE_UINT2
FrontPanIn	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_UINT2
FrontPanOut	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_UINT2
ResFlagCfg	SYSPROP_MODE_RDWR	SYSPROP_TYPE_UINT2
ResFlag	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_UINT2
HvPwSM	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR
HVFanStat	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR
ClkFreq	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_INT2
HVClkConf	SYSPROP_MODE_RDONLY	SYSPROP_TYPE_STR
IPAddr	SYSPROP_MODE_RDWR	SYSPROP_TYPE_STR
IPNetMsk	SYSPROP_MODE_RDWR	SYSPROP_TYPE_STR
IPGw	SYSPROP_MODE_RDWR	SYSPROP_TYPE_STR
RS232Par	SYSPROP_MODE_RDWR	SYSPROP_TYPE_STR
FrontPanOutLevel	SYSPROP_MODE_RDWR	SYSPROP_TYPE_UINT2
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

GetSysPropList

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

Parameters	Definition
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

CAEN Electronic Instrumentation

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

SY4527/5527	
Name	**Definition**
Sessions	List Users connected to the system
ModelName	System name
SwRelease	System firmware release
GenSignCfg	GEN signal configuration
FrontPanIn	System input status
FrontPanOu	System output status
ResFlagCfg	Reset flags configuration
ResFlag	To be implemented
HvPwSM	Power supply modules status
HVFanStat	Fan status
ClkFreq	Clock frequency
HVClkConf	Clock configuration
IPAddr	System IP address
IPNetMsk	System IP net mask
IPGw	System IP gateway
SymbolicName	System symbolic name
PWCurrent	Power section current status
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

SetBdParam

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

Parameters	Definition
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

SetChName

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

Parameters	Definition
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

SetChParam

```
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	Definition
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

SetSysProp

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

Parameters	Definition
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

TestBdPresence

```
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	Definition
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

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.

SubscribeBoardParams

```
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	Definition
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

SubscribeChannelParams

```
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	Definition
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

SubscribeSystemParams

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

Parameters	Definition
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

UnSubscribeBoardParams

```
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	Definition
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

UnSubscribeChannelParams

```
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	Definition
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

UnSubscribeSystemParams

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

Parameters	Definition
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

Event Data Items

Name	Type	Definition			
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 = 5, N1470 = 6, V8100 = 7, N568E = 8, DT55XX = 9			
CAENHV_EVT_STATUS_t	enum	Used with SY4527 SY5527 Only!	SYNC	0	All events received correctly
			ASYNC	1	Event queue filling up; expected slowness
			UNSYNC	2	Some events dropped
			NOTAVAIL	3	The device is not SY4527/SY5527
CAENHV_SYSTEMSTATUS_t	struct	CAENHV_EVT_STATUS_t System; CAENHV_EVT_STATUS_t Board[16];			

The following funcitons:
CAENHV_FreeEventData
Deallocates the memory for the data received from the mainframe (allocated within the library).
CAENHV_GetEventData
allows to receive data from the mainframe through the socket created by the TCP connection

FreeEventData

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

Parameters	Definition
ListOfItemsData	List of items received

GetEventData

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

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

Possible values of CAENHVRESULT

Value	Definition
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	Too 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.

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.

Please remember that, before calling such functions, it is necessary to call CAENHV_GetChParamInfo, in order to allow the used Client software to create the required data structures to communicate with the device.

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 the power supply,
                        ** so we can exit thread */
                }
                /* data handling loop */
                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. 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.

Via Vetraia, 11

55049 Viareggio
Italy
Tel. +39.0584.388.398
Fax +39.0584.388.959
info@caen.it
www.caen.it

CAEN GmbH

Klingenstraße 108

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

CAEN Technologies, Inc.

1140 Bay Street - Suite 2 C

Staten Island, NY 10305
USA
Tel. +1.718.981.0401
Fax +1.718.556.9185
info@caentechnologies.com
www.caentechnologies.com

Electronic Instrumentation