Technical Information Manual

Revision n. 4
3 July 2003

CAEN HV WRAPPER

C LIBRARY
FOR CAEN PSS
CONTROL

NPO:
00100/00:1527C.MUTx/04

CAEN will repair or replace any product within the guarantee period if the Guarantor declares that the product is defective due to workmanship or materials and has not been caused by mishandling, negligence on behalf of the User, accident or any abnormal conditions or operations.

CAEN declines all responsibility for damages or injuries caused by an improper use of the Modules due to negligence on behalf of the User. It is strongly recommended to read thoroughly the CAEN User's Manual before any kind of operation.

CE

CAEN reserves the right to change partially or entirely the contents of this Manual at any time and without giving any notice.

TABLE OF CONTENTS

1. Introduction

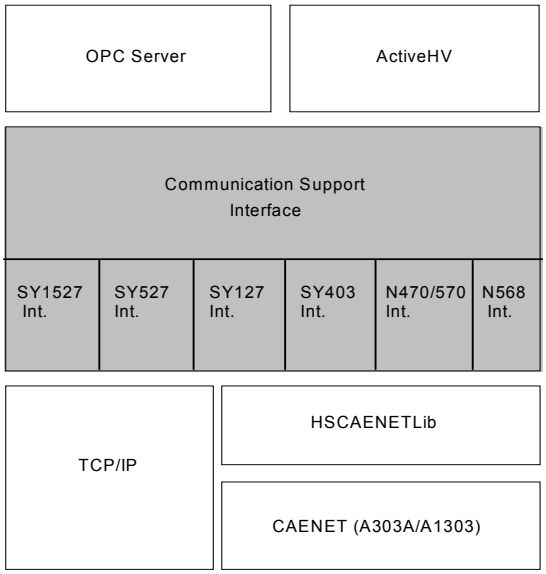
This document describes the CAEN HV Wrapper library and the functions it implements.

CAEN HV Wrapper is a set of ANSI C functions which permits an user program the control of CAEN Power Supply systems. 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 or TCP/IP).

At the moment of writing this document describing Rel. 2.7, CAEN HV Wrapper is available in the following formats:

Win32 DLL (CAEN provides the CAENHVWrapper.lib stub for Microsoft Visual C++ 6.0)
Linux dynamic library

CAEN HV Wrapper is logically located between an application like ActiveHV or OPC server and the lower layer software libraries1, as shown in the picture below:



The user of the library must identify the Power Supply to which to connect by choice of a string, like “SY1527”, “SY527”, “System0”, or any other value the user prefers.
Once the Communication Support Interface understands that the given Power Supply is a SY1527/SY2527, it calls the specific functions of the SY1527 Interface which, on his side, uses the standard socket interface to control the P.S.
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.

[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**.

The user of this library must define a string label (HV P.S. Name) for every HV power supply he/she wants 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

Description of the functions

```
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	A constant like: LINKTYPE_CAENET, or LINKTYPE_TCPIP
Arg	Points to a char of the type “A303_IOAddr_CrNum” or “A1303_Id_CrNum” when linkType is CAENET; points to a char IP when linkType is TCPIP
UserName	A string containing the User’s Name; has meaning only for SY1527/SY2527
Password	A string containing the User’s Password; has meaning only for SY1527/SY2527

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 LINKTYPE_CAENET, 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 borads can be used in the same PC) and the crate number of the system in the chain.
If `linkType` is LINKTYPE_TCPIP, it executes a login command (SY1527 or SY2527 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.
It then inserts a new entry into the table of correspondences between the `systemName` and some useful parameters, like the handle (if SY1527/2527), the model name, …

```
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.

```
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 SY1527/SY2527, 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 SY1527/SY2527, 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 SY1527/SY2527, 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 SY1527/SY2527, 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…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 SY1527/SY2527, 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 SY1527/SY2527, 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 SY1527/SY2527, 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 SY1527/SY2527, 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 SY1527/SY2527, 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 11	Channel is unplugged
Bit 12…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 SY1527/SY2527, 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 SY1527/SY2527, 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 SY1527/SY2527 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 deallocated by the user

In the following table we show the list returned for the SY1527/SY2527 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/SY2527 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

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 interface on SY1527 is down
0x6	Not present
0x7	Slot not present
0x8	Communic. Via RS232 not implemented yet
0x9	Not enough user memory
0xa	Value out of range
0xb	Command not implemented yet
0xc	Reading property not implemented yet
0xd	Writing property not implemented yet
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	SY1527 configuration change
0x15	Parameter's Property not found
0x16	Parameter not found
0x1001	Power Supply already connected
0x1002	Power Supply not connected
0x1004	Login failed
0x1005	Logout failed
0x1006	Type of connection not supported

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

3. 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.

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.