Chapter II

iseg OPC Servers for iseg Multi-Channel HV systems

Document: **isegOPCServer** Version: 3.10 Date: **07. July 2004 07:53**

Table of Contents

OLE for Process Control (OPC) for the iseg Multi-Channel HV systems

The iseg OPC server as a part of OLE process control is the link between the OPC client - depends of the application - and the firmware controlled iseg Multi-Channel HV modules.

1 Introduction

The **iseg** Multi-Channel HV system is made of several devices of hardware and software components. The hardware devices are as follows:

- Multi-Channel HV power supply modules
- System crates carrying the HV modules

Each module and each crate offers a microprocessor based intelligence. The interface which controls and monitors the hardware is the CAN bus. It is following the CAN 2.0B (passive) specification. The server for the HV modules and the server for the crate must work on separate CAN buses.
The system software interface is made by an OPC server, which follows the rules defined by the OPC Foundation (OPC 2.0). Therefore the users of the system must not know the internal protocols in detail.
In order to understand the OPC interface (server namespace), the relevant details of the modules and the crates are described as follows:

2 Modules

Each modules offers up to 32 channels, made of one or two internal cards (PCB). Each internal card represents one CAN node (the most of the modules have 16 channels per card, some modules comes also with a other number of channels per card – see instruction "Placed hardware channels" of the EHQ Multi-Channel CAN operators manual). Each channel of the module offers individual properties (see below).
In addition there are properties as groups which summarize a property for all channels and which are controlled by one CAN node.

properties of one channel:

- set voltage	write / read
- current trip	write / read
- actual current	read
- actual voltage	read
- status	read

properties of a channel group (some examples):

- sum error	read
- ramp speed	read / write
- set voltage for all channels	write
- emergency cut-off	write

3 Crates

Properties of a crate (some examples):

- actual voltage of single lines	read
- temperature	read
- Power ON / OFF	read / write

The most important information of the crate is the status of the power supplies.

4 Software

' icanHVcontrol.exe' control software performs all basic monitor and control tasks for modules and crates. It provides a HMI (human machine interface) for all properties of the modules and crates using the proprietary driver of the CAN interface (PEAK).
An alternative and more general control software is based on the standardized OPC interface.

5 OPC Server *-EHQ3216Srv-* for Multi-Channel HV devices

The OPC server has been developed using the following tools:

- Softing's OPC Toolkit, Ver. 3.04
- Microsoft's Visual C++, Ver. 6.01
- PEAK System's CAN device driver

The **iseg** OPC server executable 'EHQ3216Srv.exe' has two OPC server types. The first type is 'Data Access', the second type is 'Alarms and Events'.

5.1 Configuration

First the OPC server has to be configured. It must get all information about the kind of **iseg** HV hardware connected to the CAN bus. This information is stored in the configuration file. The tool 'EHQcfg' is used to create this configuration file. It performs a scan on the CAN bus and collects information from the connected CAN nodes (modules and crates).
For further details see the configuration manual *'**iseg**OPCSetup.pdf'*.

5.2 Data Access Server and Alarm and Events Server

5.2.1 Data Access Server

The OPC (DA) server is made to work with more than one crate. Therefore each property has to be addressed in a geographical way to build a fully qualified item ID which means:

STATUS.COMPONENT	iseg OPC server components (software releases and status of CAN bus)
CANBUS.NODE.CHANNEL.PROPERTY	for channel depending properties
CANBUS.NODE.PROPERTY	for module depending or Multi-Channel properties

By the use of a special namespace file – *EHQ3216Srv.nsp* – can build user defined fully qualified item IDs. The consist is decrypted in the file "*CreateNameSpace.pdf*". The program "*BrowseIsegSrv.exe*" demonstrate how it can created a user namespace.

The **iseg** modules can be linked to the CAN bus in a passive mode or in an active/passive mode. These modes are selectable by the OPC server (caveat: the hardware is made for the active mode). Modules of active/passive mode can take control over the CAN bus and send messages to the CAN bus (e.g. if there is an overload state). Modules in passive mode send information after a request of the CAN bus only.

The properties in the OPC server are defined as items. In the simplest case, such an item is directly coupled to a read or write via CAN bus. The 'set voltage' is one example.

Some OPC items have to be built up from data read results via CAN. The 'status current limit' is one example, which is read as an unsigned integer (2 bytes). Each bit of these 2 bytes represents the status of the current limit of one channel. This bit is interpreted as Boolean. All channels result in an array with 16 elements of Boolean, the 'StatILimitBoolArray'.

There is a feature of ranking these many requests because a client can send many of them. First priority is assigned to emergency off 'Emcy', second priority to the command set voltage 'VSet'. All other requests are under normal (lowest) priority.

A background loop process can be used to update the cache of the items continuously. This process reads all data from the HV modules and fills the cache of the OPC server namespace. To implement a background loop process the "ReadSync" entry in the OPC initialising file "EHQ3216Srv.ini" have to be a different value from zero. The advantages of this mechanism are that a OPC client can access to the items very fast by cache read, the client must not consider the order while many accesses and the OPC server will read the constant items only once (e.g. "NominalV").

5.2.1.1 Items of the releases of iseg OPC server components

Status.release_EHQ3216Srv	EHQ3216Srv.EXE	readable	VT_BSTR
Status.release_isegCAN	isegcanv.DLL	readable	VT_BSTR
Status.CAN	status of CAN bus	readable	VT_BSTR

5.2.1.2 Items of data access to the module properties

Syntax:	*t* Є [a|p]	module type (active or passive)
	x Є [1..16]	number of the CAN bus
	y Є [0..63]	number of the CAN node
	zz Є [0..15]	number of the channel

can*x*.m*ty*.ch*zz*.ITrip	current trip	write-/ readable	VT_R4
can*x*.m*ty*.ch*zz*.VMeas	actual voltage	readable	VT_R4
can*x*.m*ty*.ch*zz*.IMeas	actual current	readable	VT_R4
can*x*.m*ty*.ch*zz*.NominalV	nominal voltage of channel	readable	VT_UI2
can*x*.m*ty*.ch*zz*.NominalI	nominal current of channel	readable	VT_UI2
can*x*.m*ty*.ch*zz*.VSet	set voltage	write-/ readable	VT_R4
can*x*.m*ty*.ch*zz*.ISet	set current	write-/ readable	VT_R4
can*x*.m*ty*.ch*zz*.Stat	status channel	readable	VT_UI2

can*x*.m*ty*.GeneralStat	general status	readable	VT_UI1
can*x*.m*ty*.GeneralSumError	sum error	readable	VT_BOOL
can*x*.m*ty*.GeneralStable	all channels are stable	readable	VT_BOOL
can*x*.m*ty*.GeneralSafetyLoop	safety loop is closed	readable	VT_BOOL
can*x*.m*ty*.GeneralFineAdjust	fine adjustment is on	write-/ readable	VT_BOOL
can*x*.m*ty*.StatVLimit	status voltage limit	readable	VT_UI2
can*x*.m*ty*.StatVLimitBoolArray	array status voltage limit	readable	VT_BOOL
can*x*.m*ty*.StatILimit	status current limit	readable	VT_UI2
can*x*.m*ty*.StatILimitBoolArray	array status current limit	readable	VT_BOOL
can*x*.m*ty*.StatITrip	status software current trip	readable	VT_UI2
can*x*.m*ty*.StatITripBoolArray	array status software current trip	readable	VT_BOOL
can*x*.m*ty*.StatRegulationErr	status regulation error	readable	VT_UI2
can*x*.m*ty*.StatRegulationErrBoolArray	array status regulation error	readable	VT_BOOL
can*x*.m*ty*.On	channel on=1, off=0	write-/ readable	VT_UI2
can*x*.m*ty*.OnBoolArray	array on	write-/ readable	VT_BOOL
can*x*.m*ty*.VSetAllChannels	set voltage of all channels	write able	VT_R4
can*x*.m*ty*.ITripAllChannels	set current trip of all channels	write able	VT_R4
can*x*.m*ty*.RampSpeed	ramp speed	write-/ readable	VT_R4
can*x*.m*ty*.Emcy	emergency off	write able	VT_UI2
can*x*.m*ty*.EmcyBoolArray	array emergency off	write able	VT_BOOL
can*x*.m*ty*.KillEnable	kill enable	write-/ readable	VT_UI2
can*x*.m*ty*.KillEnableBoolArray	array kill enable	write-/ readable	VT_BOOL
can*x*.m*ty*.NominalV	nominal voltage	readable	VT_UI2
can*x*.m*ty*.NominalI	nominal current	readable	VT_UI2
can*x*.m*ty*.ADCFilterFrequency	ADC filter frequency	readable	VT_UI2
canx.mty.DeviceID	device identifier	readable	VT_BSTR
canx.mty.SoftwareID	software release	readable	VT_BSTR
canx.mty.BitRate	bit rate	readable	VT_UI2
canx.mty.hardwareILimit	hardware current limit	readable	VT_R4
canx.mty.hardwareVLimit	hardware current limit	readable	VT_R4
canx.mty.Supply 24V	supply 24V	readable	VT_R4
canx.mty.Supply 15V	supply 15V	readable	VT_R4
canx.mty.Supply 5V	supply 5V	readable	VT_R4
canx.mty.Supply -15V	supply -15V	readable	VT_R4[*1]
canx.mty.Supply -5V	supply -5V	readable	VT_R4[*1]
canx.mty.BoardTemp	board temperature	readable	VT_R4
canx.mty.AllocCh	allocated hardware channels	readable	VT_UI2[*2]
canx.mty.ChnNotOK	channel don't work properly	readable	VT_UI2[*1]

canx.mty.ErrThreshold	threshold of error evaluation	write-/ readable	VT_UI2
canx.mty.ConfigRelFErr	configuration mask (of relay and regulation error)	write-/ readable	VT_UI1

[*1] for floating 8 channel HV nodes only

[*2] for floating 8 channel HV nodes only and node types 4, 5

5.2.1.3 Items to signal an alarm from the HV devices via data access

canx.mty.chzz.Alarm	alarm status	readable	VT_BOOL
canx.mty.chzz.AlarmInfo	alarm information	readable	VT_UI1

The Alarm and AlarmInfo item are implemented as event driven update inside of the OPC server and will be cleared by reading from the client, if the error as alarm trigger was cleared before (by read the corresponding status information).

5.2.1.4 Items to handle the condition related events of the alarm and event server

The bounds of data slots will set to the full range of corresponding HV module when the OPS server is started. In this case the condition related event are always inactive but via an OPC client the slots can be changed and activate a condition related events (see also condition related events below):

canx.mty.chzz.VoltageSlot.VMeasSlot	actual voltage	write-/readable	VT_R4
canx.mty.chzz.VoltageSlot.VlBoundSlot	lower voltage bound	write-/readable	VT_R4
canx.mty.chzz.VoltageSlot.VuBoundSlot	upper voltage bound	write-/readable	VT_R4
canx.mty.chzz.CurrentSlot.IMeasSlot	actual current	write-/readable	VT_R4
canx.mty.chzz.CurrentSlot.IlBoundSlot	lower current bound	write-/readable	VT_R4
canx.mty.chzz.CurrentSlot.uBoundSlot	upper current bound	write-/readable	VT_R4

5.2.1.5 Hints to the item *status channel*

can*x*.m*ty*.ch*zz*.Stat	status channel	readable	VT_UI2

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
v	c	k	e	r	o	i	x	x	x	x	x	x	x	s	t

t	current trip	t = 0	channel is ok
		t = 1	V_O shut of 0V because software current trip was exceeded
s	sum error	s = 0	channel is ok
		s = 1	detection of a sum error - consist of an OR between current and voltage limit error in time slots of 1ms
x	no information		
i	input-error	I = 0	no input-error
		I = 1	wrong message to control the module
o	switch channel to	o = 0	channel OFF
		o = 1	channel ON
r	ramping	r = 0	voltage is stable
		r = 1	voltage ramps
e	emergency cut-off	e = 0	channel works
		e = 1	cut-off V_O shut off to 0V without ramp
k	kill function	k = 0	kill function disable V_O shut off if current limit was exceeded and then V_O is ramping from 0V to V_{SET}
		k = 1	kill function enable V_O shut off permanently if current limit was exceeded
c	current limit error	c = 0	channel is ok
		c = 1	V_O shut off 0V because hardware current limit was exceeded
v	voltage limit error	v = 0	channel is ok
		v = 1	V_O shut of permanently because voltage limit was exceeded

For detection of a current or voltage limit error flag the firmware must evaluate the channel voltage at first.

5.2.1.6 Hints to the item *general status*

can*x*.m*ty*.GeneralStat general status readable VT_UI1

b7	b6	b5	b4	b3	b2	b1	b0
save	killena	vsup	avad	stbl	sloop	ramp	sum

sum	sum error flag	sum = 0	voltage limit, current limit or trip were exceeded in the module
		sum = 1	status channel flags v & c & t = 0 for all channels
ramp	ramping flag	ramp = 0	V_O is ramping at least one channel
		ramp = 1	no channel is ramping
sloop	safety loop flag	sloop = 0	safety loop is broken -V_O bas been shut off, clear this bit by reading the general status information
		sloop = 1	safety loop is closed
stbl	stable	stbl = 0	all channels are stable with programmable ADC filter frequency f_N (ADC conversion time =1 / $f_{N,}$ see 'ADC filter frequency setting', default f_N = 50 Hz)
		stbl = 1	at least one channel is ramping Vo or not yet stable after ramping (with ADC filter frequency f_N = 100 Hz)
avad	average adjust	avad=0	fine adjustment OFF for all modules with sn. 471xxx and 473xxx average of voltage and current measurement OFF for all modules with sn. 472xxx
		avad=1	fine adjustment ON for all modules with sn. 471xxx and 473xxx average of voltage and current measurement ON for all modules with sn. 472xxx
vsup	supply voltages	vsup=0	supply voltages or module temperature are out of range
		vsup=1	supply voltages and module temperature are in range
killena	kill enable	killena=0	kill function disable only at modules with sn. 473xxx
		killena=1	kill function enable only at modules with sn. 473xxx
save	save set values	save=0	no write access to EEPROM
		save=1	store all set values to EEPROM (time to save ca. 10s)

sn. serial numbers

5.2.1.7 Hints to the item *configuration of the relay and regulation error*

acanx.mty.ConfigRelFErr configuration of relay and regulation error write-/readable VT_UI1

b7	b6	b5	b4	b3	b2	b1	b0
x	x	CACO	CRErr	CSLoop	CTrip	CVErr	CILimit

CILimit discharge if the hardware current limit was exceeded at least one channel

CVErr discharge if the hardware voltage limit was exceeded at least one channel

CTrip discharge if the software current trip was exceeded at least one channel

CSLoop discharge if the safety loop was active

CRErr discharge if the regulation was out of order at least one channel of (reaction >= 1ms)

CACO discharge if all channels set to “OFF”(Group access module “Channel ON/OFF” or “Emergency cut-off”)

X not used

The relay contacts will discharge capacities connected to the output with help of an integrated load resistor (see Appendix B Operators Manual - Multi-channel High Voltage Power Supply EHQ). This item configures the conditions of how this does work.

Under the setting of one of these conditions and the corresponding error occurs following will happen:

- shut off the HV without ramp in all channels and the set voltage in all channels to 0V by software.
- close contact of discharge relay.

5.2.1.8 Hints to the item *alarm information*

scanx.mty.chzz.AlarmInfo alarm status readable VT_UI1

b7	b6	b5	b4	b3	b2	b1	b0
x	temp	vsupl	sloop	V_{Err}	C_{Limit}	S_{Err}	trip

trip	current trip	trip = 0 ⇒ no channel has tripped
		trip = 1 ⇒ software current trip at least one of the channels
S_{Err}	sum error error	S_{Err} = 0 ⇒ no channel has a sum error (see channel status)
		S_{Err} = 1 ⇒ at least one of the channels has detected a sum error
C_{Limit}	current limit	C_{Limit} = 0 ⇒ no channel has exceeded the hardware current limit
		C_{Limit} = 1 ⇒ at least one of the channels has exceeded the current limit
V_{Err}	voltage limit	V_{Err} = 0 ⇒ no channel has exceeded the voltage limit
		V_{Err} = 1 ⇒ at least one of the channels has exceeded the voltage limit
sloop	safety loop	sloop = 0 ⇒ safety loop is closed
		sloop = 1 ⇒ safety loop is broken
vsupl	voltage supplies	vsupl = 0 ⇒ supply voltages are in range
		vsupl = 1 ⇒ supply voltages are out of range

temp	module temperature	temp = 0 ⇒ module temperature <= 60°C, no action temp = 1 ⇒ module temperature > 60°C, high voltage has been switched off

5.2.1.9 Items for public groups defined by the OPC server:

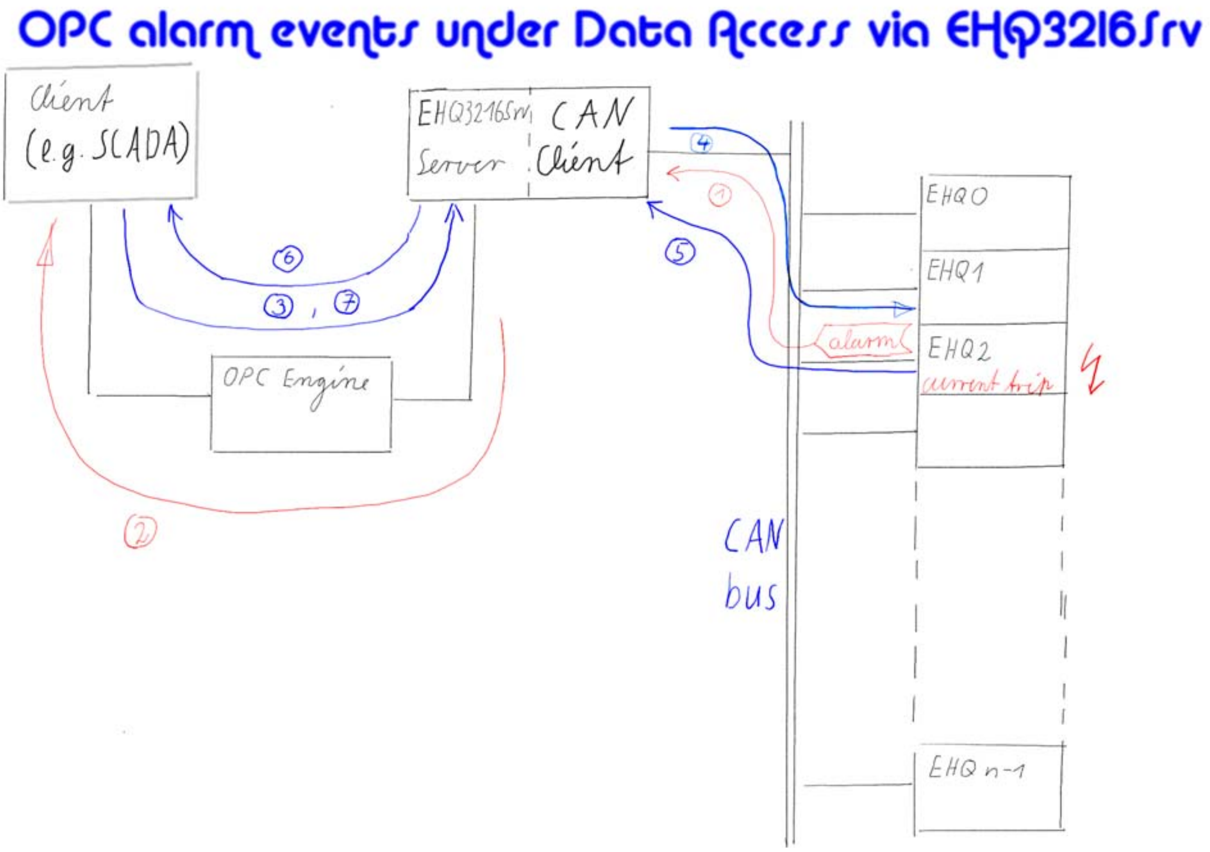
GroupDeviceID	list of all device identifiers	readable	VT_BSTR
GroupSoftwareID	list of all software identifiers	readable	VT_BSTR
GroupStatILimit	list of all statuses current limit	readable	VT_UI2
GroupStatILimitBoolArray	list of all arrays status current limit	readable	VT_BOOL
GroupStatVLimit	list of all statuses voltage limit	readable	VT_UI2
GroupStatVLimitBoolArray	list of all arrays status voltage limit	readable	VT_BOOL
GroupStatITrip	list of all statuses current trip	readable	VT_UI2
GroupStatITripBoolArray	list of all arrays status current trip	readable	VT_BOOL
GroupStatRegulationErr	list of all statuses regulation error	readable	VT_UI2
GroupStatRegulationErrBoolArray	list of all arrays status regulation error	readable	VT_BOOL
GroupSumError	list of all sum errors	readable	VT_BOOL
GroupStable	list of all stable status	readable	VT_BOOL
GroupSafetyLoop	list of all safety loop status	readable	VT_BOOL
GroupFineAdjust	list of all fine adjustment flags	readable	VT_BOOL
GroupBitRate	list of bit rates that are stored in modules	readable	VT_UI2
GroupErrThreshold	list error thresholds	readable	VT_UI2
GroupConfigRelFErr	list of bit mask for relay configuration	readable	VT_UI2
GroupAlarm	list of all alarm status information	readable	VT_BOOL

We recommend to use the following items respectively the public groups of:

StatILimit
StatILimitBoolArray
StatVLimit
StatVLimitBoolArray
StatITrip
StatITripBoolArray
StatRegulationErr

StatRegulationErrBoolArray

in OPC state *connect* only. If an error occurs it will be signalled by the item *alarm status* in connection with the check of the sum error flag from the item GeneralStat (GeneralStatSumError). These items will catch the errors when they have been read one time.



(1) A current trip happens and will generate one CAN alarm message with higher priority as the normal messages of the data transfer.

(2) The EHQ3216Srv-Server sets the item Alarm to TRUE and gives a hint of the kind of the alarm by the item AlarmInfo (both were build as a reported item in the name space).

(3, 4, 5, 6) The Client has to read which channel has tripped.

(7) Try to reset the Alarm and AlarmInfo items by read themselves. If the alarm was cleared before inside of the module the Alarm item goes to FALSE.

5.2.2 Alarms and Events server

The OPC server offers the ‘Alarms & Events’ feature built into the same executable in order to let the OPC client act quickly on a single event or an alarm.

The following alarms and events have been defined:

5.2.2.1 Simple events

canx.ErrorSafetyLoop	error status of safety loop	readable
canx.ErrorSupply	error status of supply voltages	readable
canx.ErrorSumError	error status of general sum status	readable

5.2.2.2 Tracking events

ComputerKeyboardPressed	access to local keyboard on server	readable
ComputerMouseActivity	access to local mouse on server	readable

5.2.2.3 Condition related events

If measured values are exceeding the limits of a defined slot then condition related events are generated. This applies to current and voltage.

canx.mtay.chzz.VoltageSlot	actual voltage is not between the upper and lower bound (defined in data access)
canx.mtay.chzz.CurrentSlot	actual current is not between the upper and lower bound (defined in data access)

6 OPC server for crate *ECHx38Srv*

The OPC server has been developed using the following tools:

- Softing OPC Toolkit, Ver. 3.04
- Microsoft Visual C++, Ver. 6.01
- PEAK System CAN device driver

The executable 'ECHx38Srv.exe' is an OPC Data Access server.

6.1 Configuration

The OPC server has to be configured at the beginning. It must get all information about the kind of **iseg** HV hardware connected to the CAN bus. This information is stored into the configuration file. The tool 'ECHcfg' is used to create this configuration file. It performs a scan on the CAN bus and collects information from the connected CAN nodes (modules and crates).
For further details see the configuration manual '**iseg**OPCSetup.pdf'.

6.2 Data Access Server and Alarm and Event Server

6.2.1 Data Access Server

The OPC (DA) server is made to work with more than one crate. Therefore each property has to be addressed in a geographical way to build a fully qualified item ID which means:

CANBUS.CRATE.PROPERTY

The properties in the OPC server are defined as items. In the simplest case, such an item is directly coupled to a read or write via CAN bus. The 'On' is an example.
The OPC data access methods is working via request queues. A client request is starting a read/write access to the node property via CAN bus. After this action has been finished the OPC server is sending a call back to the client. According to the above the following items are used in the iseg OPC server namespace:

6.2.1.1 Items of data access

Syntax:	*x* Є [1..16]	number of the CAN bus
	y Є [0..63]	number of the CAN node

can*x*.crate*y*.Supply 24V	crate power 24V	readable	VT_R4
can*x*.crate*y*.Supply 5V	crate power 5V	readable	VT_R4
can*x*.crate*y*.Battery	battery voltage for the UPS ca. 24V	readable	VT_R4
can*x*.crate*y*.Temp back plane	temperature on the back plane	readable	VT_R4
can*x*.crate*y*.Temp power supply	temperature on the DC/DC converter	readable	VT_R4
can*x*.crate*y*.On	status of the power	write-/readable	VT_BOOL
can*x*. Crate*y*.DeviceID	device identifier	readable	VT_BSTR
can*x*.crate*y*.SoftwareID	software release	readable	VT_BSTR
can*x*.Crate*y*.BitRate	bit rate	readable	VT_UI2
canx.cratey.AlarmFlag	alarm status	readable	VT_BOOL
canx.cratey.AlarmInformation	alarm information	readable	VT_BSTR
		EMCY supply 24V *EMCY supply 5V* *Temperature PS* *Temperature BP* *AC line power*	
canx.cratey.AlarmValue	alarm value	readable	VT_R4

An alarm status is implemented in order to signal the error message of the crate power supply. The alarm will be generated by means of EMCY ID (see manual “ECH2-38x_eng.pdf”).

6.2.2 Alarm and Events Server

The OPC server offers the ‘Alarms & Events’ feature built into the same executable in order to let the OPC client act quickly on a single event or an alarm.

The following alarms and events have been defined:

Simple events:

canx.ErrorSupply24V	error of supply 24V	readable
canx.ErrorSupply5V	error of supply 5V	readable
canx.ErrorTemperaturePS	error of temperature senor on power supply 24V-DC-PS	readable
canx.ErrorTemperatureBP	error of temperature sensor on pack plane	readable
canx.ErrorACline_power	error of AC line power	readable

Tracking events:

ComputerKeyboardPressed	access to local keyboard on server	readable
ComputerMouseActivity	access to local mouse on server	readable