Technical reference

netPROXY

Object reference

V2.0

Hilscher Gesellschaft für Systemautomation mbH
www.hilscher.com

DOC160204TR02EN | Revision 2 | English | 2018-02 | Released | Public

Table of contents

1 Introduction

1.1 About this document

This document describes the netPROXY objects.

1.2 List of revisions

Rev	Date	Version	Revisions
1	2017-11-22	V2.0	All sections created.
2	2018-02-15	V2.0	Section *System Status (0x10001050)* [▶ page 14] updated.
			Section *Generic Diagnosis (0x10006000)* [▶ page 15] added.
			Section *Interface Control (0x20000001)* [▶ page 19] added. Object 0x20000001 replaces object 0x20000000.
			Section *Interface State (0x20000002)* [▶ page 20] added.
			Section *Device Description (0x20000010)* [▶ page 21] added.

Table 1: List of revisions

2 netPROXY object ID range overview

The following table is an overview about the object ID ranges of netPROXY objects.

2.1 netPROXY object ID range overview

Object ID	Category
0x00000000	NULL
0x00000001 - 0x0FFFFFFF	netPROXY objects
0x10000000 - 0x1FFFFFFF	System objects for instance: Device ID
0x20000000 - 0x2FFFFFFF	Communication protocol objects for instance: Parameters
0x30000000 - 0x3FFFFFFF	Hilscher objects
0x40000000 - 0x4FFFFFFF	Customer-specific objects
0x50000000 - 0xFFFFFFFF	Reserved

Table 2: Global address space for Object IDs

2.2 netPROXY objects usable for the host

Note:
All other existing objects hold configuration data and are all (firmware) internal objects. The host application must neither read nor write these objects.

3 netPROXY objects (0x0xxxxxxx)

3.1 netPROXY Object List (0x00001000)

The netPROXY Object List object is a list of all existing object entities. The application can read this object to browse all available objects to get an overview about available objects.

The application has to use an interface to get access to the netPROXY objects.

The netPROXY Server creates and handles this object.

Object characteristics:

- This object is read only.
- This object only appears in the netPROXY Group 0.
- Every instance of an existing netPROXY object is listed in the netPROXY Object list.
- The object order in the list is not deterministic.
- The number of existing objects can be determined by the number of instances of this object.
- The netPROXY Object List object is included in this list.

Object

Object ID	Object name	Description	Instances
0x00001000	netPROXY Object List	A list of all existing objects.	N

Table 3: Component Configuration object

Elements

#	Element name	Data type	Data format	Description	Access
0	Object ID	UINT32	Value	netPROXY object ID	Read only
1	Group ID	UINT32	Value	Group ID this object belongs to.	Read only

Table 4: Component Configuration elements

4 System objects (0x1xxxxxxx)

4.1 Generic Device objects

4.1.1 Device Description (0x10001000)

The Device Description object stores general vendor information.

This object appears in the netPROXY Group 0.

The netX Studio Engineering Tool configures this object.

Object

Object ID	Object name	Description	Instances
0x10001000	Device Description	This object stores general vendor information.	1

Table 5: Device Description object

Elements

#	Element name	Data type	Data format	Description	Access
0	Vendor Name	char[64]	Printable ASCII string	Representation of the company name. This name is identical for all devices that a company sell. Example 1: "Hilscher Gesellschaft für Systemautomation mbH" Example 2: "Hilscher North America"	Read only
1	Vendor Address	char[128]	Printable ASCII string	The official contact address of the company, or department, which is responsible for this product Example 1: "Rheinstraße 15, 65795 Hattersheim, Germany" Example 2: "Suite 410, 901 Warrenville Road, Lisle (Chicago), IL 60532"	Read only
2	Vendor Phone	char[64]	Printable ASCII string	The official phone number of the company, or department, which is responsible to this product Example 1: "+49 (0) 6190-9907-0". Example 2: "+1 630.505.5301".	Read only
3	Vendor URL	char[64]	Printable ASCII string	The main vendor URL (Domain) of the responsible department. Example: "www.hilscher.com"	Read only

Table 6: Device Description elements

4.1.2 Device Type Label (0x10001010)

The Device Type Label object stores unique device properties, which are available after production or quality assurance.

This object appears in the netPROXY Group 0.

The netX Studio Engineering Tool configures this object.

Object

Object ID	Object name	Description	Instances
0x10001010	Device Type Label	This object stores unique device properties.	1

Table 7: Device Type Label object

Elements

#	Element name	Data type	Data format	Description	Access
0	Serial Number	char[32]	Printable ASCII string	Serial number of the device. This number must be unique for all devices of the same product.	Read only
1	Production Date	char[32]	ASCII time	Date/Time of manufacturing or shipment	Read only
2	Quality Assurance Date	char[32]	Printable time	Date/Time when quality assurance test has been performed on the device. • This test is typically performed after production. • Last test in the field (i.e. after a firmware update)	Read / write
3	Production Location	char[128]	Printable ASCII string	Address location of the final product production (finishing for shipment) Example: "Rheinstraße 15, 65795 Hattersheim, Germany"	Read only
4	MAC Address	uint8_t[6]		Unique device MAC address	Read only

Table 8: Device Type Label elements

4.1.3 Device Maintenance (0x10001020)

The Device Maintenance object contains device specific information, which can be changed by the end user. This is only informative and can be used for example for web-based management or SNMP or other bus-specific mapping.

This object appears in the netPROXY Group 0.

The netX Studio Engineering Tool configures this object.

Object

Object ID	Object name	Description	Instances
0x10001020	Device Maintenance	This object contains device specific information, which can be changed by the end user.	1

Table 9: Device Maintenance object

Elements

#	Element name	Data type	Data format	Description	Access
0	Name	char[64]	Printable ASCII string	Unique label (string) in the plant for the function of this device.	Read / write
1	Installation Location	char[32]	Printable ASCII string	Unique label (string) in the plant for the location where this device is installed.	Read / write
2	Installation Date	char[32]	ASCII Time	Date of installation or commissioning of this device, format may be defined by field bus organization.	Read / write
3	Contact Information	char[32]	Printable ASCII string	The textual identification of the contact person for this managed node, together with information on how to contact this person.	Read / write
4	Description	char[64]	Printable ASCII string	Human-readable comment field to store any individual additional information and annotation.	Read / write Read / write
5	Signature	char[128]	Printable ASCII string	-	Read / write
6	Change Count	char[32]	ASCII Decimal Number	Counter for changes to hardware or device parameters. Must be incremented only if data is really changed.	Read / write
7	Last Service Date	char[32]	ASCII Time	Date/Time of the last service, e.g. firmware update.	Read / write
8	Next Service Date	char[32]	ASCII Time	Date/Time of next scheduled service for the device, e.g. cleaning	Read / write

Table 10: Device Maintenance elements

4.1.4 Device Version Information (0x10001030)

The Device Version Information object stores all general version information of the device.

This object appears in the netPROXY Group 0.

The netX Studio Engineering Tool configures this object.

Object

Object ID	Object name	Description	Instances
0x10001030	Device Version Information	This object stores all general version information of the device.	1

Table 11: Device Version Information object

Elements

#	Element name	Data type	Data format	Description	Access
0	Product Revision	char[32]	ASCII Version	Overall revision of the device	Read / write
1	Hardware Name	char[32]	Printable ASCII string	Human readable name of the device hardware	Read / write
2	Hardware Version	char[32]	ASCII Version	Revision of the device hardware, format may be defined by field bus organization	Read / write
3	Software Name	char[32]	Printable ASCII string	Human readable name of the device firmware	Read / write
4	Software Version	char[32]	ASCII Version	Revision of the device firmware, format may be defined by field bus organization	Read / write
5	Bootloader Name	char[32]	Printable ASCII string	Human readable name of the boot loader	Read / write
6	Bootloader Version	char[32]	ASCII Version	Revision of the device boot loader, format may be defined by field bus organization	Read / write

Table 12: Device Reset Control elements

4.1.5 Device Reset Control (0x10001040)

The Device Reset Control object is used to control the main reset function of the device. Performing a reset means, that the device stops all communication activities (I/O connections etc.), releases its configuration and starts over again.

This object appears in the netPROXY Group 0.

The following steps must be performed by the application

- Inform the protocol adapter via the Common Communication - Interface Control object that the stack goes to state passive or release the configuration.
- Ensure that all write operations are executed, and all remanent data is stored
- Execute the device reset

Object Handling

1. An application (Web Server, Protocol Stack Adapter etc.) that requests a reset has to write value 1 into the Reset element.

   Additionally, it is possible to delay the reset execution by specifying a Delay time in milliseconds. The delay time must be set before or at the same time writing the Reset element.
2. The reset must be executed as soon the Host application accepts the request (Writes value 2 to the Reset element). In case of a stand-alone device, this step can be omitted.

Object

Object ID	Object name	Description	Instances
0x10001040	Device Reset Control	This object is used to control the main reset function of the device.	1

Table 13: Device Reset Control object

Elements

#	Element name	Data type	Data format	Description	Access
0	Reset	UINT8	Enumeration	0 = None 1 = Request a Reset 2 = Reset accepted	Read / write
1	Delay ms	UINT32	Value	Delay in ms when the reset should occur.	write

Table 14: Device Reset Control elements

Example sequence

The following figure shows the firmware internal sequence of a device reset.

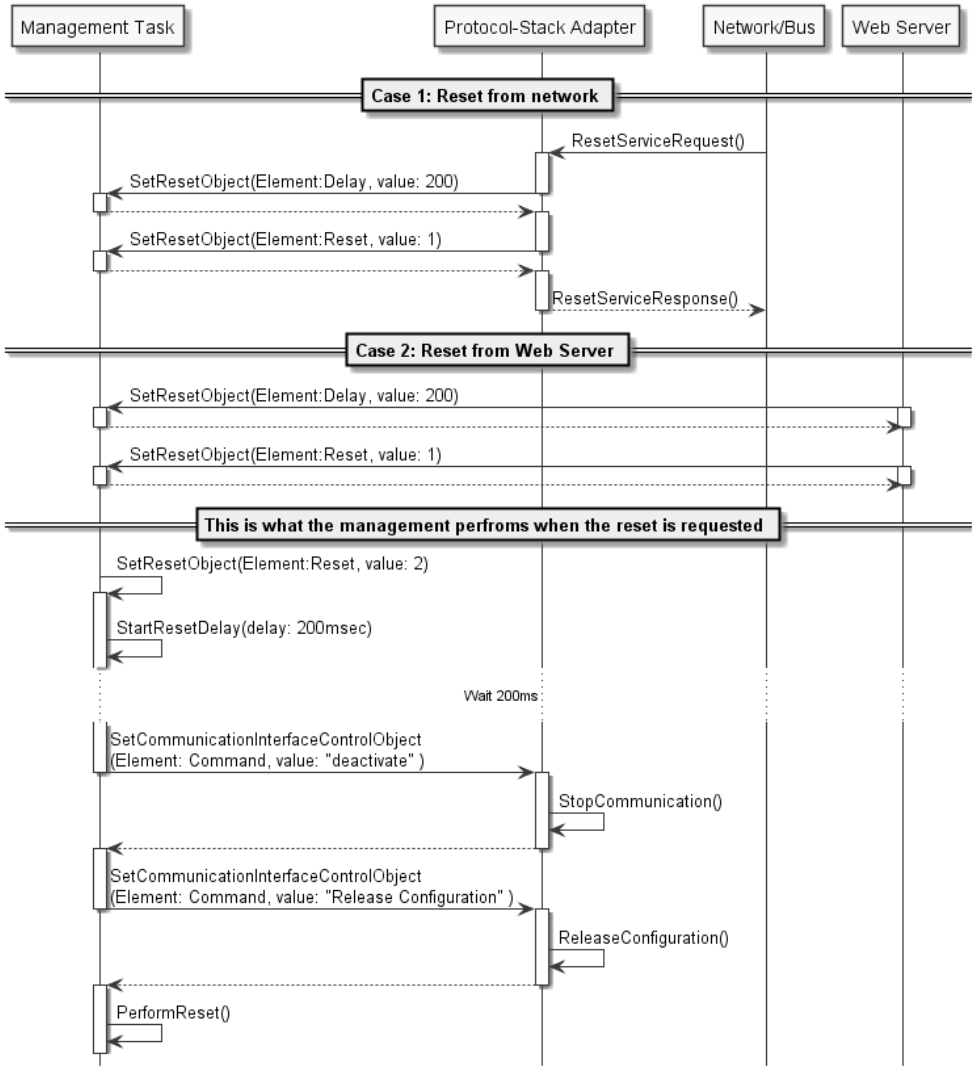

Figure 1: Device Reset Control sequence

4.1.6 System Status (0x10001050)

The System Status object contains an overall firmware status of the device. Since the package will also started during the start of netPROXY firmware, some states may not visible for application.

This object appears in the netPROXY Group 0.

Object

Object ID	Object name	Description	Instances
0x10001050	System Status	This object contains an overall firmware status of the device.	1

Table 15: System Status object

Elements

#	Element name	Data type	Data format	Description	Access
0	Status	char[32]	Printable ASCII string	Human readable description of the Status code "OK", "Error", ...	Read only
1	Status code	UINT32	Value	2000-2999 = Starting Packages 3000-3999 = Initialization & Configuration 4000-4999 = Reserved 5000 = Init done	Read only
2	Uptime	UINT32	Value	Time since the last restart in seconds	Read only
3	CPU Load	UINT32	Value	The CPU Load in %	Read only
4	Free Memory	UINT32	Value	Free memory in bytes (Heap)	Read only

Table 16: System Status elements

4.2 Diagnosis object

4.2.1 Generic Diagnosis (0x10006000)

The Generic Diagnosis object is used to set, remove, or update diagnosis entries in the communication protocol stack adapter which is configured to use one or more instances of this object. Depending on the communication protocol stack (e.g. PROFINET IO Device), a write access to an instance of this object automatically triggers or resets an alarm associated with the given diagnosis.

Each diagnosis is represented by one instance of the Generic Diagnosis object. If an application is to support simultaneous multiple diagnoses (each with a different "Error Identifier") then the application has to use multiple instances of this object.

Note:
Writing the elements "Channel Number", "Error Identifier", "Reserved" and "Error Value Format" is not allowed as long as diagnosis is active which is the case as soon as the element "Severity Level" is non-zero.

Object

Object ID	Object name	Description	Instances
0x10006000	Generic Diagnosis	This object is used to set, remove, or update diagnosis entries for the protocol.	N

Table 17: Generic Diagnosis object

Elements

#	Element name	Data type	Data format	Description	Access
0	Severity Level	UINT16	Value	This element describes the urgency of the problem reported by the diagnosis. The Severity Level can be switched from any state to another state. Default value = 0. For a value description, see below.	Read / write
1	Channel Number	UINT16	Value	This element is used to associate a diagnosis with a specific channel or I/O line. E.g.: A short circuit can thus be traced down to a specific output line. If the diagnosis is not associated with a specific I/O line, then the value 0x8000 shall be used, e.g. in case if device's power supply is overloaded. Default value = 0x8000. 0x0000 … 0x00FE: Number or ID of the channel or I/O line that is effected. 0x00FF … 0x7FFF: Reserved. 0x8000: The module or the device as a whole is affected. 0x8001 … 0xFFFF: Reserved.	Read / write

#	Element name	Data type	Data format	Description	Access
2	Error Identifier	UINT16	Enumeration	This element holds an enumeration value giving a category or a typical reason why the diagnosis has been created. The **value must be non-zero** before activating a diagnosis by setting the "Severity Level" element to a non-zero value. For a value description, see below.	Read / write
3	Reserved	UINT8	Value	Default value: 0.	-
4	Error Value Format	UINT8	Value	Defines the storage format of the error value. The value is one of the netPROXY data type codes (NPX_TYPE_xxx). Default value: 0 (NPX_TYPE_UNKNOWN). The value of this element can be used for improved interpretation / visualization of the data given in the Error Value element. If the format code is NPX_TYPE_UNKNOWN (0), then the data in the Error Value element is not used and should be disregarded. If the format code denotes a numeric value (NPX_TYPE_INTEGER, NPX_TYPE_UNSIGNED, NPX_TYPE_REAL, NPX_TYPE_ENUMERATION) or a boolean value (NPX_TYPE_BOOLEAN) or a bitfield (NPX_TYPE_BITFIELD), then the data in the Error Value element is to be interpreted as a 64-bit value stored low-order-byte first. If the format code denotes a text string (NPX_TYPE_STRING), then the given text must be NUL-terminated. For other non-numeric data, the format code NPX_TYPE_BINARY is used which denotes that the data is to be interpreted as an array of 8 raw data bytes.	Read / write
5	Error Value	BINARY[8]	Value	Buffer for the error value, i.e. the actual value that caused the problem. E.g.: The actual value measured by an A/D converter which exceeds the allowed range set for scaling. Although this element is defined as an array of 8 raw data bytes, the format code given in the Error Value Format element can be used for improved interpretation / visualization of the buffered data. E.g.: Depending on the format code, a web page may display the data buffered here as a floating point value representing the voltage actually measured by the A/D converter. Default value: 0 (all bytes)	Read / write

Table 18: Generic Diagnosis elements

Security Level - Values (Element 0)

Value	Description
0	No diagnosis No Diagnosis is reported from application. Diagnosis is inactive.
1	Information The lowest possible severity level. Typically used to announce that a service will be required in the near future so that a service date can be scheduled. The regular function of the device is not affected, though. E.g.: A belt has been in use for more than 80% of its expected life time and should be replaced soon. Diagnosis is active.
2	Warming A medium level which is used if a mechanical or electrical element experiences more stress than it has been designed for. E.g.: An output line is overloaded, but still working as expected. Or: The expected life time of a mechanical element has been exceeded and a device failure is to be expected in the near future. Diagnosis is active.
3	Error The highest possible severity level. Typically used to report that the device is in a faulty state, or does not operate within the specified conditions. E.g. A short circuit on an output line has been detected. Diagnosis is active.

Table 19: Security Level - Values (Element 0)

Error Identifier – Enumeration values (Element 2)

Value	Name	Description
0	Reserved	Initialization value in instances of the object that has never been used.
1	Short circuit	The associated I/O line reports a short circuit.
2	Line break	The associated I/O line reports a broken line or wire.
3	Ground fault	The ground connection is faulty.
4	Overvoltage	The absolute voltage is above the nominal level.
5	Undervoltage	The absolute voltage is below the nominal level.
6	Overcurrent	The current is outside the nominal range.
7	Overload	The associated output is overloaded.
8	Output disabled	The associated output is disabled.
9	Overtemperature	Overtemperature has been detected.
10	Sensor supply overload	An attached device overloads the sensor supply rail.
11	Actor supply overload	A attached device overloads the actor supply rail.
12	Fuse blown	A blown (open) fuse was detected.
13	Parameter missing	The requested status cannot be reached because of missing parameter data.
14	Parameterization fault	Incorrect parameter values have been configured (e.g. value out of range).
15	Power supply fault	Power supply fault detected.
16	Reference point lost	Reference point for (re-)positioning lost.
17	Sampling error	Process event lost / sampling error.
18	Communication error	Error in the communication subsystem.
19	Device software	A software error was detected (e.g. an exception occurred).
20	Software reset (watchdog)	A watchdog timer has not been retriggered in due time.
21	Application software	An application software error has been detected.
22	Non-volatile memory	Non-volatile memory corruption detected (e.g. caused by battery failure).
23	Data memory	Data memory corruption detected (RAM).
24	Measured value above range	The measured value is above the nominal value range.
25	Measured value below range	The measured value is below the nominal value range.
26 - 0x00FF	Reserved	-
0x0100 - 0x01FF	User-defined area	• PROFINET maps this to 0x0100 - 0x01FF • Sercos maps this to 0x0100 - 0x01FF • CANopen emergency maps this to 0xFF00 - 0xFFFF • EtherCAT 0xE000 - 0xE0FF
0x0200 - 0xFFFF	Reserved	-

Table 20: Error Identifier – Enumeration values (Element 2)

5 Communication objects (0x2xxxxxxx)

5.1 Communication objects (0x2000xxxx)

5.1.1 Interface Control (0x20000001)

The Common Communication Interface Control object is a generic object and controls the communication protocol.

The application can use command services to control the communication protocol. The new state (in the *Interface State (0x20000002)* [page 20] object) is reached as soon as the write access has been completed.

Object

Object ID	Object name	Description	Instances
0x20000001	Common Communication Interface Control	This object is a generic object and controls the communication protocol.	1

Table 21: Common Communication Interface Control object

Elements

#	Element name	Data type	Data format	Description	Access
0	Command	NPX_TYPE_UNSIGNED (size 1)	Enumeration	Command service 0 = None 1 = Load Configuration	Read / write

Table 22: Common Communication Interface Control elements

5.1.2 Interface State (0x20000002)

The Common Communication Interface State object is a generic object and keeps the actual state of the communication protocol.

The application can use the Interface Control object to control the communication protocol. The new state is reached as soon as the write access to the *Interface Control (0x20000001)* [page 19] object has been completed.

Object

Object ID	Object name	Description	Instances
0x20000002	Common Communication Interface State	This object is a generic object and keeps the actual state of the communication protocol.	1

Table 23: Common Communication Interface State object

Elements

#	Element name	Data type	Data format	Description	Access
0	State	NPX_TYPE_UNSIGNED (size 4)	Enumeration	Communication interface state 0 = Reset 1 = Ready 2 = Waiting for communication 3 = Communicating	Read only
1	Result	NPX_TYPE_UNSIGNED (size 4)		Result code of the last transition	Read only

Table 24: Common Communication Interface State elements

5.1.3 Device Description (0x20000010)

The Common Communication Device Description object collects device description information.

The netX Studio Engineering Tool configures this object.

Object

Object ID	Object name	Description	Instances
0x20000010	Common Communication Device Description	This object collects device description information.	1

Table 25: Common Communication Device Description object

Elements

#	Element name	Data type	Data format	Description	Access
0	Vendor Id	char[64]	ASCII Decimal Number	Numeric Vendor ID, which is typically assigned by fieldbus Organization.	Read only
1	Product Name	char[240]	Printable ASCII string	Human readable Product Name.	Read only
2	Product Id	char[32]	Printable ASCII string	Numeric Product ID, assigned either by fieldbus organization or vendor.	Read only
3	Product Type	char[240]	Printable ASCII string	Human readable product classification.	Read only
4	Order Id	char[32]	Printable ASCII string	Human readable product order number.	Read only
5	Profile Id	char[32]	ASCII Decimal Number	Numeric Profile ID, assigned by fieldbus organization.	Read only
6	Profile Type	char[32]	Printable ASCII string	Numeric Profile Type, assigned either by fieldbus organization or vendor.	Read only

Table 26: Common Communication Device Description elements

Mapping

#	Element name	Mapping to EtherNet/IP	Mapping to PROFINET
0	Vendor Id	Class: 1 “Identity”, Inst: 1, Attr: 1 “Vendor ID”	I&M0: MANUFACTURER_ID
1	Product Name	Class: 1 “Identity”, Inst: 1, Attr: 7 “Product Name”	-
2	Product Id	Class: 1 “Identity”, Inst: 1, Attr: 3 “Product Code”	Device ID
3	Product Type	EDS: Catalog Number	Device Type
4	Order Id	-	I&M0: ORDER_ID
5	Profile Id	Class: 1 “Identity”, Inst:1, Attr: 2 “Device Type” (means CIP device profile)	I&M0: PROFILE_ID
6	Profile Type	-	I&M0: PROFILE_SPECIFIC_TYPE

Table 27: Common Communication Device Description - Mapping

5.1.4 LED Indicators (0x20000020)

The Common Communication LED Indicators object represents the LED status of the communication protocol.

Object

Object ID	Object name	Description	Instances
0x20000020	Common Communication LED Indicator	This object represents the LED status.	N

Table 28: Common Communication LED Indicator object

Elements

#	Element name	Data type	Data format	Description	Access
0	Indicator Name	char[32]	Printable ASCII string	Name of the Indicator LED	Read only
1	Indicator Status	char[32]	Printable ASCII string	Current LED Status	Read only
2	Update Count	uint32_t	Number	The counter must be increased if the script is updated. As soon as the counter has changed, the script has restarted.	Read only
3	LED Color	uint8_t[32]	Enumeration for each octet	Each octet of this element representing the color of one script step. Step color 0 = Off 1 = Red 2 = Green 3 = Yellow 4 = Blue 5 = Magenta 6 = Cyan 7 = White	Read only
4	LED Script	uint8_t[32]	Time value for each octet	Each octet of this element representing the step duration. 0 = End of script. The next step is the first octet of the script. 1-254 = Step duration in 10 ms steps. 255 = Stop script. The script execution can be stopped until the update count is incremented.	Read only

Table 29: Common Communication LED Indicators elements

5.2 Internet Protocol V4 objects (0x20019xxx)

5.2.1 Internet Protocol V4 Configuration (0x20019000)

The Internet Protocol V4 Configuration object holds the configuration for the IP stack.

Object ID	Object name	Description	Instances
0x20019000	Internet Protocol V4 Configuration	This object holds the configuration for the IP stack.	1

Table 30: Internet Protocol V4 Configuration object

#	Element name	Data type	Data format	Description	Access
0	Config Control	UINT8	Enumeration	Configuration control. Defines how to obtain an IP address 0 = STATIC 1 = BOOTP 2 = DHCP	Read / write
1	Config Source	UINT8	Enumeration	Internal source of the configuration (Informative) 0 = Factory default 1 = Remanent data 2 = Hardware switch	Read / write
2	IP Address	UINT32	Value	IPv4 address of the device or this IP interface. A value of 0 indicates that no IP address has been configured.	Read / write
3	Subnet Mask	UINT32	Value	Subnet mask A value of 0 indicates that no subnet mask address has been configured.	Read / write
4	Gateway Address	UINT32	Value	Subnet mask IP address of the default gateway	Read / write

Table 31: Internet Protocol V4 Configuration elements

5.2.2 Internet Protocol V4 Status(0x20019001)

The Internet Protocol V4 Status object holds the current IPv4 configuration of the IP stack.

Object ID	Object name	Description	Instances
0x20019001	Internet Protocol V4 Status	This object holds the current IPv4 configuration of the IP stack.	1

Table 32: Internet Protocol V4 Status object

#	Element name	Data type	Data format	Description	Access
0	IP Address	UINT32	Value	IPv4 Address	Read only
1	Subnet Mask	UINT32	Value	Subnet mask	Read only
2	Gateway Address	UINT32	Value	Gateway Address	Read only
3	Address conflict	UINT8	Enumeration	0 = No conflict 1 = Conflicted device detected	Read only
4	Conflict device	UINT8[6]	Value	Subnet mask MAC address of conflicted device	Read only

Table 33: Internet Protocol V4 Status elements

5.2.3 Internet Protocol V4 DHCP Configuration (0x20019010)

The Internet Protocol V4 DHCP Configuration object holds information for the automatic IP configuration process.

Object ID	Object name	Description	Instances
0x20019010	Internet Protocol V4 DHCP Configuration	This object holds information for the automatic IP configuration process.	1

Table 34: Internet Protocol V4 DHCP Configuration object

#	Element name	Data type	Data format	Description	Access
0	DHCP Options	UINT8	Enumeration	Supported DHCP options. Valid only if Config Control == DHCP Bit 0: Option 81 Bit 1: Option 82	Read only
1	Name Server 1	UINT32	Value	IP address of name server 1 A value of 0 indicates that no IP address has been configured.	Read only
2	Name Server 2	UINT32	Value	IP address of name server 2 A value of 0 indicates that no IP address has been configured.	Read only
3	Domain Name	STRING[240]	Printable ASCII string	Domain name	Read only
4	Host Name	STRING[240]	Printable ASCII string	Host name	Read only

Table 35: Internet Protocol V4 DHCP Configuration elements

5.3 Ethernet objects (0x20024xxx)

5.3.1 Ethernet Status (0x20024001)

The Ethernet Status object holds the current Ethernet PHY status. Each PHY has its own instance.

Object ID	Object name	Description	Instances
0x20024001	Ethernet Status	This object holds the current Ethernet PHY status. Each PHY has its own instance.	1 per PHY

Table 36: Ethernet Status object

#	Element name	Data type	Data format	Description	Access
0	Link Mau	UINT8	Enumeration	The Mau type indicates the current active link. 0: Link down 10: 10 MBit Half Duplex Copper 11: 10 MBit Full Duplex Copper 15: 100 MBit Half Duplex Copper 16: 100 MBit Full Duplex Copper 17: 100 MBit Half Duplex Glass Fiber 18: 100 MBit Full Duplex Glass Fiber	Read only
1	MDI State	UINT8	Enumeration	Current MDI mode 0: Reserved 1: AUTO 2: MDI 3: MDI-X	Read only

Table 37: Ethernet Status elements

5.4 MQTT Client (0x2002Axxx)

5.4.1 Component Configuration (0x2002A000)

The MQTT Client Component Configuration object provides the configuration data required by an MQTT client to adapt its component identification and its general startup and run-time behavior.

The netX Studio Engineering Tool creates one instance of this object for each instance of the MQTT client component running in the firmware.

The object instance number used by the MQTT client for identifying the netPROXY object is identical with the task instance.

At startup time or during re-configuration, the MQTT client component reads this object and evaluates the Flags element in order to adapt its startup behavior.

Object

Object ID	Object name	Description	Instances
0x2002A000	MQTT Client - Component Configuration	This object provides the configuration data required by an MQTT client to adapt its component identification and its general startup and run-time behavior.	1

Table 38: MQTT Client Component Configuration object

Elements

#	Element name	Data type	Data format	Description	Access
0	Flags	UINT32	Bitfield	Flags for adapting the behavior of the MQTT client component at startup / run-time. Bit 0: "Enable": enable the client functionality (otherwise the component will not search for a configuration and will not start) (default: 1) Bit 1: "Default Info": in the Component Info object, overwrite the configured identification values with built-in default values (default: 1) Bit 2 … 31: reserved (default: 0).	Read only

Table 39: MQTT Client Component Configuration elements

5.4.2 Component Information (0x2002A001)

The MQTT Client Component Information object provides information about the MQTT client component to other components such as the web server or a diagnostics application.

The netX Studio Engineering Tool creates one instance of this object for each instance of the MQTT client component running in the firmware.

The object instance number used by the MQTT client for identifying the netPROXY object is identical with the task instance.

At startup time, the MQTT Client reads the corresponding object and may change and update its contents, depending on other settings.

If one of the text elements is configured as an empty string, then the MQTT Client fills this element with the corresponding default text built-in the component or defined by other settings.

This approach facilitates OEM branding of the component.

Object

Object ID	Object name	Description	Instances
0x2002A001	MQTT Client - Component Information	This object provides information on the MQTT client component to other components.	1

Table 40: MQTT Client Component Information object

Elements

#	Element name	Data type	Data format	Description	Access
0	Vendor Name	STRING[64]	Printable UTF-8 string	Component vendor name (e.g. "Hilscher Gesellschaft fuer Systemautomation mbH") (default: empty string. In this case, the firmware will write the built-in MQTT stack vendor name string into this element.)	Read only
1	Component Name	STRING[32]	Printable UTF-8 string	Component name (e.g. "MQTT Client") (default: empty string. In this case, the firmware will write the built-in MQTT stack name string into this element.)	Read only
2	Version	STRING[32]	Printable ASCII string	Component version (e.g. "V1.0.0.0") (default: empty string. In this case, the firmware will write the built-in MQTT stack version string into this element.)	Read only
3	Build Timestamp	STRING[32]	Printable ASCII string	Component build timestamp (e.g. "Jun 23 2015, 16:20:33") (default: empty string. In this case, the firmware will write the built-in MQTT stack timestamp into this element.)	Read only

Table 41: MQTT Client Component Information elements

5.4.3 Connection Configuration (0x2002A002)

The MQTT Client Connection Configuration object provides the configuration data required by an MQTT client for establishing and maintaining a network connection to an MQTT broker.

The netX Studio Engineering Tool creates one instance of this object with the instance number 0.

At startup time or during re-configuration, the MQTT client application reads this instance.

Object

Object ID	Object name	Description	Instances
0x2002A002	MQTT Client - Connection Configuration	This object provides the configuration data required by an MQTT client for establishing and maintaining a network connection to an MQTT broker.	1

Table 42: MQTT Client Connection Configuration object

Elements

#	Element name	Data type	Data format	Description	Access
0	Unique ID	UINT32	Value	Unique numeric identifier that can be used as a connection reference (an object instance number may change when adding / deleting objects)	Read only
1	Client ID	STRING[24]	Printable ASCII string	MQTT client ID (globally unique name string) used at connection establishment time, NUL-terminated, alphanumeric characters only	Read only
2	Broker Address	STRING[256]	Printable ASCII string	MQTT broker name (fully qualified host name, e.g. "m2m.eclipse.org") or IP address in dotted decimal notation, NUL-terminated	Read only
3	Broker Port	UINT32	Value	MQTT broker IP port number, typically 1883 (default: 1883)	Read only
4	Flags	UINT32	Bitfield	Flags for adapting the handling of an MQTT connection (cf. OASIS MQTT 3.1.1 standard). If a bit is 1, the corresponding feature is enabled.) Bit 0: "Clean Session": on connect, request the broker to discard the context of a previous session of the same client (if such a context exists) (default: 1) Bit 1: "Reliable": enable "reliable" publishing (require completion of an active publish operation before triggering another) (default: 0) Bit 2: "Will": enable the MQTT 3.1 "will and testament" feature (automatic publishing of a "will" if the broker detects the client to be missing) (default: 0) Bit 3: "Prefix Will": add the topic prefix to the given will topic (if the MQTT 3.1 will feature is enabled) (default: 1) Bits 4 … 31: reserved (default: 0)	Read only
5	Topic Prefix	STRING[256]	Printable UTF-8 string	MQTT topic prefix (if not empty, this string will be used as a prefix to the topics of MQTT actions flagged accordingly), NUL-terminated (default: empty string, in which case the firmware will use the MAC address)	Read only

#	Element name	Data type	Data format	Description	Access
6	User Name	STRING[32]	Printable UTF-8 string	MQTT 3.1 username string used for establishing connections that require a login (e.g. secure connections), NUL-terminated (default: empty string)	Read only
7	Password	STRING[32]	Binary data string	MQTT 3.1 password string used for establishing connections that require a login (e.g. secure connections), NUL-terminated (default: empty string)	Read only
8	Will Topic	STRING[256]	Printable UTF-8 string	MQTT 3.1 will topic string (prefixed by the firmware if the "Prefix Will" flag is set) (default: empty string, in which case the firmware will use the MAC address followed by "/will/" in order to create the required unique string)	Read only
9	Connect Timeout	UINT32	Value	timeout for MQTT connection establishment, given in seconds, 0 = connect timeout check disabled (default: 0)	Read only
10	Connection Idle Timeout	UINT32	Value	required idle time before closing an MQTT connection, given in seconds, 0 = connection idle check disabled (default: 0)	Read only
11	MQTT Keep Alive Interval	UINT32	Value	interval for MQTT connection keepalive handling, given in seconds, 0 = keepalive feature disabled (default: 0) Attention: This is not the socket option "KeepAlive" of the TCP stack but rather a short message generated by the MQTT stack.	Read only

Table 43: MQTT Client Connection Configuration elements

5.5 OPC UA Server (0x2002Bxxx)

5.5.1 Component Configuration (0x2002B000)

The Component Configuration object provides the configuration data for the OPC UA Server.

The netX Studio Engineering Tool configures this object.

The OPC UA Server in the firmware reads this object during startup or for re-configuration and uses these configuration data for its identification, general startup, and runtime behavior. The OPC UA Server evaluates each value of the elements and if necessary, the OPC UA Server adapts one or more configuration parameters.

Object

Object ID	Object name	Description	Instances
0x2002B000	Component Configuration	Configuration data for the OPC UA Server.	1

Table 44: Component Configuration object

Elements

#	Element name	Data type	Data format	Description	Access
0	General flags	UINT32	Bitfield	Flags for configuring the OPC UA Server for startup and runtime.	Read / write
1	Reserved	UINT32	Bitfield	Reserved	Read only
2	Server Port	UINT32	Value	OPC UA Server IP port number, typically 4840 (default: 4840) Port 0 and 1 are not supported Value range: 0, 2, 3, ... 65535 Value 0 sets the default port.	Read / write
3	Reserved	STRING[64]	Printable ASCII string	Reserved	Read only
4	Reserved	UINT32	Value	Reserved	Read only
5	Maximum allowed Nodes	UINT32	Value	Allowed maximum number of Nodes which the Server can represent in the defined Address Space. These are self defined Nodes that are configured by the netX Studio Engineering Tool as well as the the mandatory server nodes. Value range: 400 ... 2000, default 2000 Value 0 sets the default value.	Read / write
6	Maximum Sessions	UINT32	Value	Allowed maximum number of Session which the Server has to handle. The Nano profile requires only 1. Value range: 1, 2, default 2 Value 0 sets the default value.	Read / write
7	Maximum Session Lifetime in ms	UINT32	Value	Allowed maximum lifetime for a session in ms. Value range: 30000 ... 1200000 (30 s to 20 min), default 1200000 Value 0 sets the default value.	Read / write
8	Reserved	UINT32	Value	Reserved	Read only

#	Element name	Data type	Data format	Description	Access
9	Reserved	UINT32	Value	Reserved	Read only
10	Reserved	UINT32	Value	Reserved	Read only
11	Reserved	UINT32	Value	Reserved	Read only
12	Reserved	UINT32	Value	Reserved	Read only
13	Reserved	UINT32	Value	Reserved	Read only
14	Reserved	UINT32	Value	Reserved	Read only
15	Reserved	UINT32	Value	Reserved	Read only
16	OPC UA Trace Output IP	STRING[64]	Printable ASCII string	Contains the URL to a certain target PC. default: "" Only usable with a firmware that contains a debug library of the OPC UA Server.	Read / write

Table 45: Component Configuration elements

General flags - Bitfield (Element 0)

Bit	Name	Description
0	Enable	Enable the OPC UA Server 0 = OPC UA Server disabled (default) 1 = OPC UA Server enabled
1	Allow anonymous	Access without user/password combination (not recommendable) 0 = disabled 1 = enabled (default)
2	Allow user/password access	Get the server user/password 0 = disabled 1 = enabled (default)
3	Non-secure communication	The SecurityPolicy - None security. This Facet defines a SecurityPolicy used for configurations with the lowest security needs. This SecurityPolicy can affect the behavior of the CreateSession and Activate Session services. It also results in a SecureChannel which has no Channel Security. By default this SecurityPolicy should be disabled if any other SecurityPolicies are available. 0 = disabled 1 = enabled (default)
4 … 21	Reserved	-
22	Use default identification	Defines the source for the component information. 0 = use values from object 1 = use built-in information (vendor = Hilscher, name, version, date, ...) (default)
23 … 31	Reserved	-

Table 46: General flags - Bitfield (Element 0)

5.5.2 Component Information (0x2002B001)

The Component Information object provides information about the OPC UA Server. The host application or other components e.g. the web server can read this information.

The netX Studio Engineering Tool creates one instance of this object and configures this object.

The OPC UA Server reads this object at startup time, evaluates the element values, and if necessary, the OPC UA Server adapts one or more element values.

Object

Object ID	Object name	Description	Instances
0x2002B001	Component Information	Identification information of the OPC UA Server.	1

Table 47: Component Information object

Elements

#	Element name	Data type	Data format	Description	Access
0	Vendor Name	STRING[64]	Printable UTF-8 string	Corresponds to the Manufacturer Name of the OPC UA specification (e.g. "Hilscher Gesellschaft für Systemautomation mbH"). Default: empty string. In this case, the firmware will write the built-in OPC UA Manufacturer Name string into this element.	Read only
1	Component Name	STRING[32]	Printable UTF-8 string	Corresponds to the Product Name of the OPC UA specification (e.g. "OPC UA Server"). Default: empty string. In this case, the firmware will write the built-in OPC UA Product Name string into this element.	Read only
2	Software Version	STRING[32]	Printable ASCII string	Software Version (e.g. "V1.0.0.0"), OPC UA Variable 'BuildNumber' will be generated from Software Version. Example: input Software Version = "V1.2.335.0" will represent in the UA Server Object 'BuildInfo' as Software Version = "V1.2" and BuildNumber = "335.0"). Default: empty string. In this case, the firmware will write the built-in OPC UA Software Version and Build Number string into this element.	Read only
3	Build Timestamp	STRING[32]	Printable ASCII string	Corresponds to the Build Date of the OPC UA specification (e.g. " 2014-12-18T19:59:15.000Z"). Default: empty string. In this case the firmware will write the built-in OPC UA Build Date string into this element.	Read only

Table 48: Component Configuration elements

5.5.3 Action Configuration (0x2002B002)

The Action Configuration object has configuration data for the OPC UA Server. The configuration data is the list of netPROXY objects that the OPC UA Server provides to an OPC UA Client.

The netX Studio Engineering Tool creates one ore more instance of this object and configures this object.

The OPC UA Server reads this object at startup time or for re-configuration and evaluates the element values.

Object

Object ID	Object name	Description	Instances
0x2002B002	Action Configuration	Object configuration for the OPC UA Server.	N

Table 49: Action Configuration object

Elements

#	Element name	Data type	Data format	Description	Access
0	Object Reference	netPROXY Object Reference		netPROXY Object Reference (group ID, object ID, object instance, element number) whose data the OPC UA Server has to provide to an OPC UA Client. The OPC UA Server evaluates the group ID and object ID. The OPC UA Server uses the complete object.	Read only
1	Flags	UINT32	Bitfield	Configuration flags The OPC UA Server always supports read access to the Object Reference.	Read only

Table 50: Action configuration elements

Flags - Bitfield (Element 1)

Bit	Name	Description
0	Write Access	Enable/disable write access for the netPROXY object. 0 = read only (default) 1 = read/write
1 … 31	Reserved	-

Table 51: Flags - Bitfield (Element 1)

5.5.4 Component Status (0x2002B003)

The Component Status object provides status information and diagnostic information of the OPC UA Server. This netPROXY object includes the OPC UA specified "ServerDiagnosticsSummary" elements and the OPC UA Server adds further server internal informations. Unlike the OPC UA specified object the netPROXY object behavior depends NOT on the EnableFlag from the UA Variable "ServerDiagnostics".

The OPC UA Server creates one instance this object during firmware startup.

This object is read-only.

Object

Object ID	Object name	Description	Instances
0x2002B003	Component Status	Status information of the OPC UA Server.	1

Table 52: Component Status object

Elements

#	Element name	Data type	Data format	Description	Access
0	Component State	UINT32	Enumeration	Component state code, These states are identical to the OPC UA specified DataType "ServerState" from the OPC UA specification part 5 section 12.6 ServerState.	Read only
1	Server View Count Total	UINT32	Value	Number of server-created views in the server.	Read only
2	Current Session Count Total	UINT32	Value	Number of client sessions currently established in the server.	Read only
3	Cumulated Session Count Total	UINT32	Value	Cumulative number of client sessions that have been established in the server since the server was started or restarted. This includes the currentSessionCount.	Read only
4	Security Rejected Session Count Total	UINT32	Value	Number of client session establishment requests (ActivateSession and CreateSession) that were rejected due to security constraints since the server was started or restarted.	Read only
5	Rejected Session Count Total	UINT32	Value	Number of client session establishment requests (ActivateSession and CreateSession) that were rejected since the server was started or restarted. This number includes the securityRejectedSessionCount.	Read only
6	Session Timeout Count Total	UINT32	Value	Number of client sessions that were closed due to timeout since the server was started or restarted.	Read only
7	Session Abort Count Total	UINT32	Value	Number of client sessions that were closed due to errors since the server was started or restarted.	Read only
8	Publishing Interval Count Total	UINT32	Value	Number of publishing intervals currently supported in the server.	Read only
9	Current Subscription Count Total	UINT32	Value	Number of subscriptions currently established in the server.	Read only
10	Cumulated Subscription Count Total	UINT32	Value	Cumulative number of subscriptions that have been established in the server since the server was started or restarted. This includes the currentSubscriptionCount.	Read only

#	Element name	Data type	Data format	Description	Access
11	Security Rejected Requests Count Total	UINT32	Value	Number of requests that were rejected due to security constraints since the server was started or restarted. The requests include all Services defined in the OPC UA specification part 4, also requests to create sessions.	Read only
12	Rejected Requests Count Total	UINT32	Value	Number of requests that were rejected since the server was started (or restarted). The requests include all Services defined in the OPC UA specification part 4, also requests to create sessions. This number includes the securityRejectedRequestsCount.	Read only
13	Warning Count Total	UINT32	Value	Number of occurred warning during server start up and runtime	Read only
14	Last Warning	STRING[128]	Printable UTF-8 string	Contains the last occurred warning as text (NUL-terminated). Default value is "none".	Read only

Table 53: Component Status elements

Component Status – Enumeration values (Element 0)

Value	Name	Description
0	Running	The server is running normally. This is the usual state for a healthy server. All supported UA Services works as expected.
1	Failed	A fatal error has occurred within the server. The server is no longer functioning. The recovery procedure is the attempt to restart the server component. All service requests are expected to fail.
2	NoConfiguration	The server component is running but has no configuration information loaded or some configuration information do not permit the server startup. Therefore the server does no data transfer.
3	Suspended	Configuration information available. Startup server with these configuration information. Actions: Create Server and netPROXY address space, open endpoints for communication.
4	Shutdown	The server has shut down or is in the process of shutting down.
5	Test	The server runs its Unit Test Suite.
6	CommunicationFault	The server main loop was exited with an unexpected status code.
7	Unknown	This state is used only to indicate that the OPC UA Server component was started and an unexpected error happed at the beginning.

Table 54: Component Status – Enumeration values (Element 0)

6 Hilscher objects (0x3xxxxxxx)

Hilscher objects have the object ID range from 0x30000000 to 0x3FFFFFFF. These objects holds configuration data and are all internal objects.

The host application must neither read nor write these objects.

7 Customer-specific objects (0x4xxxxxxx)

Customer-specific objects have the object ID range from 0x40000000 to 0x4FFFFFFF. The netX Studio Engineering Tool is used to create and configure these customer-specific objects.

Using the netX Studio Engineering Tool these objects can be mapped to

- input or output data of the Real-Time Ethernet network
- input or output of the periphery
- data for read / write access via the Real-Time Ethernet network (acyclic communication)
- OPC UA for read / write access
- MQTT to be published or subscribed
- integrated web server to be integrated into a web page

The host application can read and write these objects.

8 Appendix

8.1 Legal notes

Copyright

Important notes

Utmost care was/is given in the preparation of the documentation at hand consisting of a user's manual, operating manual and any other document type and accompanying texts. However, errors cannot be ruled out. Therefore, we cannot assume any guarantee or legal responsibility for erroneous information or liability of any kind. You are hereby made aware that descriptions found in the user's manual, the accompanying texts and the documentation neither represent a guarantee nor any indication on proper use as stipulated in the agreement or a promised attribute. It cannot be ruled out that the user's manual, the accompanying texts and the documentation do not completely match the described attributes, standards or any other data for the delivered product. A warranty or guarantee with respect to the correctness or accuracy of the information is not assumed.

We reserve the right to modify our products and the specifications for such as well as the corresponding documentation in the form of a user's manual, operating manual and/or any other document types and accompanying texts at any time and without notice without being required to notify of said modification. Changes shall be taken into account in future manuals and do not represent an obligation of any kind, in particular there shall be no right to have delivered documents revised. The manual delivered with the product shall apply.

Under no circumstances shall Hilscher Gesellschaft für Systemautomation mbH be liable for direct, indirect, ancillary or subsequent damage, or for any loss of income, which may arise after use of the information contained herein.

Liability disclaimer

The hardware and/or software was created and tested by Hilscher Gesellschaft für Systemautomation mbH with utmost care and is made available as is. No warranty can be assumed for the performance or flawlessness of the hardware and/or software under all application conditions and scenarios and the work results achieved by the user when using the hardware and/or software. Liability for any damage that may have occurred as a result of using the hardware and/or software or the corresponding documents shall be limited to an event involving willful intent or a grossly negligent violation of a fundamental contractual obligation. However, the right to assert damages due to a violation of a fundamental contractual obligation shall be limited to contract-typical foreseeable damage.

It is hereby expressly agreed upon in particular that any use or utilization of the hardware and/or software in connection with

- Flight control systems in aviation and aerospace;
- Nuclear fusion processes in nuclear power plants;
- Medical devices used for life support and
- Vehicle control systems used in passenger transport

shall be excluded. Use of the hardware and/or software in any of the following areas is strictly prohibited:

- For military purposes or in weaponry;
- For designing, engineering, maintaining or operating nuclear systems;
- In flight safety systems, aviation and flight telecommunications systems;
- In life-support systems;
- In systems in which any malfunction in the hardware and/or software may result in physical injuries or fatalities.

You are hereby made aware that the hardware and/or software was not created for use in hazardous environments, which require fail-safe control mechanisms. Use of the hardware and/or software in this kind of environment shall be at your own risk; any liability for damage or loss due to impermissible use shall be excluded.

Warranty

Hilscher Gesellschaft für Systemautomation mbH hereby guarantees that the software shall run without errors in accordance with the requirements listed in the specifications and that there were no defects on the date of acceptance. The warranty period shall be 12 months commencing as of the date of acceptance or purchase (with express declaration or implied, by customer's conclusive behavior, e.g. putting into operation permanently).

The warranty obligation for equipment (hardware) we produce is 36 months, calculated as of the date of delivery ex works. The aforementioned provisions shall not apply if longer warranty periods are mandatory by law pursuant to Section 438 (1.2) BGB, Section 479 (1) BGB and Section 634a (1) BGB [Bürgerliches Gesetzbuch; German Civil Code] If, despite of all due care taken, the delivered product should have a defect, which already

existed at the time of the transfer of risk, it shall be at our discretion to either repair the product or to deliver a replacement product, subject to timely notification of defect.

The warranty obligation shall not apply if the notification of defect is not asserted promptly, if the purchaser or third party has tampered with the products, if the defect is the result of natural wear, was caused by unfavorable operating conditions or is due to violations against our operating regulations or against rules of good electrical engineering practice, or if our request to return the defective object is not promptly complied with.

Costs of support, maintenance, customization and product care

Please be advised that any subsequent improvement shall only be free of charge if a defect is found. Any form of technical support, maintenance and customization is not a warranty service, but instead shall be charged extra.

Additional guarantees

Although the hardware and software was developed and tested in-depth with greatest care, Hilscher Gesellschaft für Systemautomation mbH shall not assume any guarantee for the suitability thereof for any purpose that was not confirmed in writing. No guarantee can be granted whereby the hardware and software satisfies your requirements, or the use of the hardware and/or software is uninterruptable or the hardware and/or software is fault-free.

It cannot be guaranteed that patents and/or ownership privileges have not been infringed upon or violated or that the products are free from third-party influence. No additional guarantees or promises shall be made as to whether the product is market current, free from deficiency in title, or can be integrated or is usable for specific purposes, unless such guarantees or promises are required under existing law and cannot be restricted.

Confidentiality

The customer hereby expressly acknowledges that this document contains trade secrets, information protected by copyright and other patent and ownership privileges as well as any related rights of Hilscher Gesellschaft für Systemautomation mbH. The customer agrees to treat as confidential all of the information made available to customer by Hilscher Gesellschaft für Systemautomation mbH and rights, which were disclosed by Hilscher Gesellschaft für Systemautomation mbH and that were made accessible as well as the terms and conditions of this agreement itself.

The parties hereby agree to one another that the information that each party receives from the other party respectively is and shall remain the intellectual property of said other party, unless provided for otherwise in a contractual agreement.

The customer must not allow any third party to become knowledgeable of this expertise and shall only provide knowledge thereof to authorized users as appropriate and necessary. Companies associated with the customer shall not be deemed third parties. The customer must obligate authorized

users to confidentiality. The customer should only use the confidential information in connection with the performances specified in this agreement.

The customer must not use this confidential information to his own advantage or for his own purposes or rather to the advantage or for the purpose of a third party, nor must it be used for commercial purposes and this confidential information must only be used to the extent provided for in this agreement or otherwise to the extent as expressly authorized by the disclosing party in written form. The customer has the right, subject to the obligation to confidentiality, to disclose the terms and conditions of this agreement directly to his legal and financial consultants as would be required for the customer's normal business operation.

Export provisions

The delivered product (including technical data) is subject to the legal export and/or import laws as well as any associated regulations of various countries, especially such laws applicable in Germany and in the United States. The products / hardware / software must not be exported into such countries for which export is prohibited under US American export control laws and its supplementary provisions. You hereby agree to strictly follow the regulations and to yourself be responsible for observing them. You are hereby made aware that you may be required to obtain governmental approval to export, reexport or import the product.

Terms and conditions

Please read the notes about additional legal aspects on our netIOT web site under http://www.netiot.com/netiot/netiot-edge/terms-and-conditions/.

List of figures

List of tables

Contacts

HEADQUARTERS

Germany

Hilscher Gesellschaft für
Systemautomation mbH
Rheinstrasse 15
65795 Hattersheim
Phone: +49 (0) 6190 9907-0
Fax: +49 (0) 6190 9907-50
E-mail: info@hilscher.com

Support

Phone: +49 (0) 6190 9907-99
E-mail: de.support@hilscher.com

SUBSIDIARIES

China

Hilscher Systemautomation (Shanghai) Co. Ltd.
200010 Shanghai
Phone: +86 (0) 21-6355-5161
E-mail: info@hilscher.cn

Support

Phone: +86 (0) 21-6355-5161
E-mail: cn.support@hilscher.com

France

Hilscher France S.a.r.l.
69500 Bron
Phone: +33 (0) 4 72 37 98 40
E-mail: info@hilscher.fr

Support

Phone: +33 (0) 4 72 37 98 40
E-mail: fr.support@hilscher.com

India

Hilscher India Pvt. Ltd.
Pune, Delhi, Mumbai
Phone: +91 8888 750 777
E-mail: info@hilscher.in

Italy

Hilscher Italia S.r.l.
20090 Vimodrone (MI)
Phone: +39 02 25007068
E-mail: info@hilscher.it

Support

Phone: +39 02 25007068
E-mail: it.support@hilscher.com

Japan

Hilscher Japan KK
Tokyo, 160-0022
Phone: +81 (0) 3-5362-0521
E-mail: info@hilscher.jp

Support

Phone: +81 (0) 3-5362-0521
E-mail: jp.support@hilscher.com

Korea

Hilscher Korea Inc.
Seongnam, Gyeonggi, 463-400
Phone: +82 (0) 31-789-3715
E-mail: info@hilscher.kr

Switzerland

Hilscher Swiss GmbH
4500 Solothurn
Phone: +41 (0) 32 623 6633
E-mail: info@hilscher.ch

Support

Phone: +49 (0) 6190 9907-99
E-mail: ch.support@hilscher.com

USA

Hilscher North America, Inc.
Lisle, IL 60532
Phone: +1 630-505-5301
E-mail: info@hilscher.us

Support

Phone: +1 630-505-5301
E-mail: us.support@hilscher.com