

X20BC008T

Data sheet
1.42 (August 2025)



Publishing information

B&R Industrial Automation GmbH

B&R Strasse 1

5142 Eggelsberg

Austria

Telephone: +43 7748 6586-0

Fax: +43 7748 6586-26

office@br-automation.com

Disclaimer

All information in this document is current as of its creation. The contents of this document are subject to change without notice. B&R Industrial Automation GmbH assumes unlimited liability in particular for technical or editorial errors in this document only (i) in the event of gross negligence or (ii) for culpably inflicted personal injury. Beyond that, liability is excluded to the extent permitted by law. Liability in cases in which the law stipulates mandatory unlimited liability (such as product liability) remains unaffected. Liability for indirect damage, consequential damage, business interruption, loss of profit or loss of information and data is excluded, in particular for damage that is directly or indirectly attributable to the delivery, performance and use of this material.

B&R Industrial Automation GmbH notes that the software and hardware designations and brand names of the respective companies used in this document are subject to general trademark, brand or patent protection.

Hardware and software from third-party suppliers referenced in this document is subject exclusively to the respective terms of use of these third-party providers. B&R Industrial Automation GmbH assumes no liability in this regard. Any recommendations made by B&R Industrial Automation GmbH are not contractual content, but merely non-binding information for which no liability is assumed. When using hardware and software from third-party suppliers, the relevant user documentation of these third-party suppliers must additionally be consulted and, in particular, the safety guidelines and technical specifications contained therein must be observed. The compatibility of the products from B&R Industrial Automation GmbH described in this document with hardware and software from third-party suppliers is not contractual content unless this has been separately agreed in individual cases; in this respect, warranty for such compatibility is excluded in any case, and it is the sole responsibility of the customer to verify this compatibility in advance.

1533498333614-1.42

1 General information

1.1 Other applicable documents

For additional and supplementary information, see the following documents.

Other applicable documents

Document name	Title
MAX20	X20 System user's manual

Additional documentation

Document name	Title
Cybersecurity	For cyber-secure operation, the guidelines from Cybersecurity / Defense in Depth for B&R products must be taken into account.

1.2 Order data


Order number	Short description	Figure
	Bus controllers	
X20BC008T	X20 bus controller, 1 OPC UA FX Ethernet interface, integrated 2-port switch, 2x RJ45, order bus base, power supply module and terminal block separately!	
	Required accessories	
	System modules for bus controllers	
X20BB80X	X20 bus controller base for X20BC008T and X20 power supply module, X20 end cover plates (left and right) X20AC0SL1/X20AC0SR1 included	
X20PS9400	X20 power supply module, for bus controller and internal I/O power supply X2X Link power supply	
X20PS9402	X20 power supply module, for bus controller and internal I/O power supply, X2X Link supply, supply not galvanically isolated	
	Terminal blocks	
X20TB12	X20 terminal block, 12-pin, 24 VDC keyed	

Table 1: X20BC008T - Order data

1.3 Module description

This bus controller provides OPC UA FX functions. This allows any OPC UA clients access to read or write data from I/O modules connected to the bus controller.

- Communication technology: OPC UA Field eXchange (FX)
- I/O configuration via OPC UA FX
- 400 µs minimum cycle time
- Integrated switch for daisy-chaining
- 2x 1 Gbit/s full-duplex mode
- OPC UA diagnostics and module diagnostics at runtime via OPC UA clients

2 Technical description

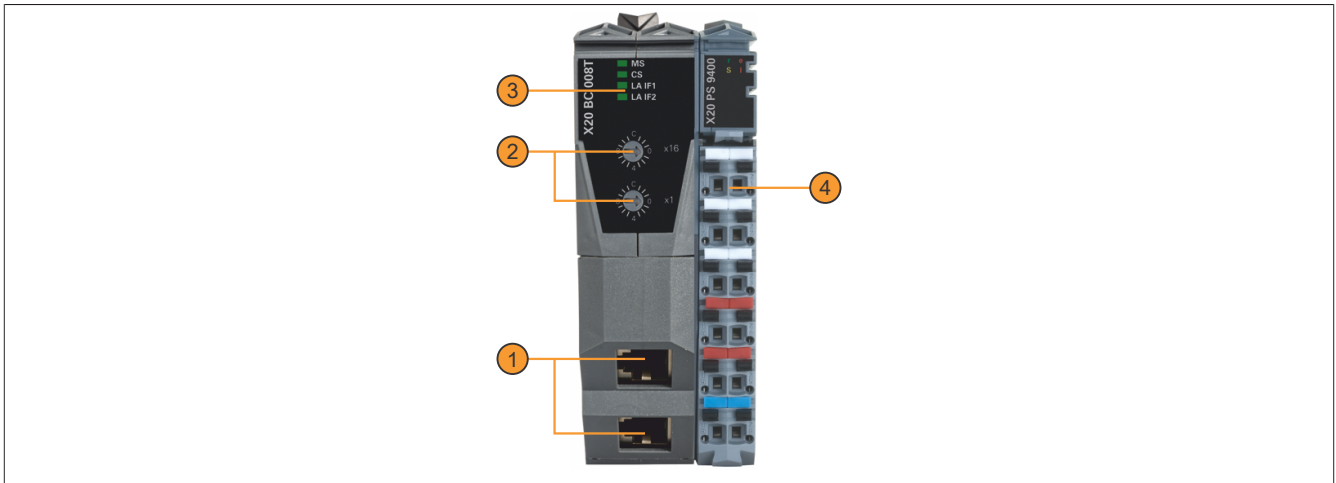
2.1 Technical data

Order number	X20BC008T
Short description	
Bus controller	OPC UA FX
General information	
B&R ID code	0xF629
Status indicators	Module status, bus function
Diagnostics	
Module status	Yes, using LED status indicator and software
Bus function	Yes, using LED status indicator and software
Power consumption	
Bus	3.5 W
Additional power dissipation caused by actuators (resistive) [W]	-
Certifications	
CE	Yes
UKCA	Yes
UL	cULus E115267 Industrial control equipment
Interfaces	
Fieldbus	OPC UA Field eXchange (FX)
Variant	2x shielded RJ45 (switch)
Line length	Max. 100 m between 2 stations (segment length)
Transfer rate	1 Gbit/s
Transfer	
Physical layer	100BASE-TX/1000BASE-T
Half-duplex	No
Full-duplex	Yes
Autonegotiation	Yes
Auto-MDI/MDIX	Yes
Min. cycle time ¹⁾	
Fieldbus	400 µs
X2X Link	400 µs
Synchronization between bus systems possible	Yes
Electrical properties	
Electrical isolation	OPC UA FX isolated from bus and I/O
Operating conditions	
Mounting orientation	
Horizontal	Yes
Vertical	Yes
Installation elevation above sea level	
0 to 2000 m	No limitation
>2000 m	Reduction of ambient temperature by 0.5°C per 100 m
Degree of protection per EN 60529	IP20
Ambient conditions	
Temperature	
Operation	
Horizontal mounting orientation	-25 to 45°C
Vertical mounting orientation	-25 to 40°C
Derating	-
Storage	-40 to 85°C
Transport	-40 to 85°C
Relative humidity	
Operation	5 to 95%, non-condensing
Storage	5 to 95%, non-condensing
Transport	5 to 95%, non-condensing
Mechanical properties	
Note	Order 1x terminal block X20TB12 separately. Order 1x power supply module X20PS9400 or X20PS9402 separately. Order 1x bus base X20BB80X separately.
Pitch ²⁾	37.5 ^{+0.2} mm

Table 2: X20BC008T - Technical data

- 1) The minimum cycle time specifies how far the bus cycle can be reduced without communication errors occurring.
- 2) Pitch is based on the width of bus base X20BB80X. In addition, power supply module X20PS9400 or X20PS9402 is always required for the bus controller.

2.2 Operating and connection elements




1	OPC UA FX connection with 2x RJ45 for simple wiring	2	Number switches
3	LED status indicators	4	Terminal block for bus controller and I/O supply

2.2.1 LED status indicators

The following table lists the LED status indicators available on the bus controller. Exact blink times are specified in the timing diagram in the next section.

Immediately after switching on, the LEDs flash red. This is not an error message.

Figure	LED	Color	Status	Description
	MS ¹⁾	-	Off	No power supply to module or mode RESET ²⁾
		Green	2 pulses	Firmware update
			On	Module OK
		Red	1 pulse	Mode RESET: Restart
			2 pulses	Mode RESET: Deletes the configuration
			3 pulses	Mode RESET: Deletes the safety configuration
			4 pulses	Mode RESET: Resets to the factory settings
		Green + Red	On	Error state
			On	Mode RESET: Confirmation of the deletion procedure
	CS ³⁾	Green	1 pulse	Waiting for IP configuration
			2 pulses	Waiting for PTP synchronization
			3 pulses	Waiting for NTP synchronization
			On	Network OK
		Red	1 pulse	IP configuration timeout ⁴⁾
			2 pulses	PTP synchronization timeout ⁵⁾
			3 pulses	NTP synchronization timeout
			4 pulses	PTP status error ⁶⁾
	LA IFx	Green	On	IP address conflict
			Off	No link to remote station
			Flickering	The link to the remote station is established. The LED blinks if Ethernet activity is taking place.

1) Module status "MS": This LED is a green/red dual LED.

2) See 2.2.2 "Number switches".

3) LAN status "LS": This LED is a green/red dual LED.

The LED changes from the green pulsed state to the red pulsed state if the current "Waiting for" status is present for longer than 15 s. When the status changes, this time is reset.

4) The bus controller has not yet been assigned an IP address.

5) The bus controller is not yet synchronized via PTP. Possible causes:

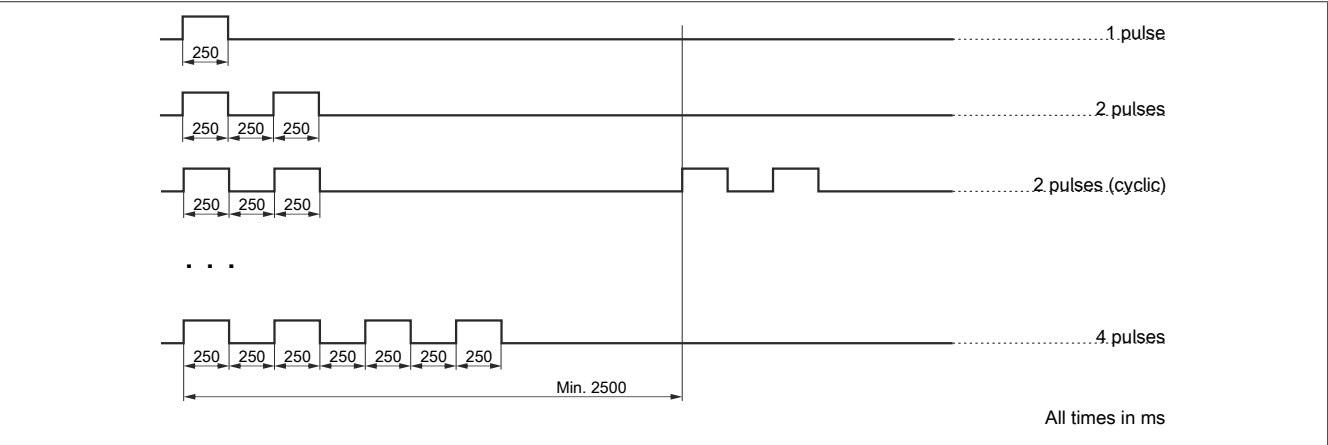
- No connection to a PTP grandmaster
- The synchronization offset to the PTP grandmaster is outside the specification ($\text{abs}(\text{OffsetFromMaster}) > \text{SyncOffsetNs}$).
- PTP configuration error

6) Possible causes:

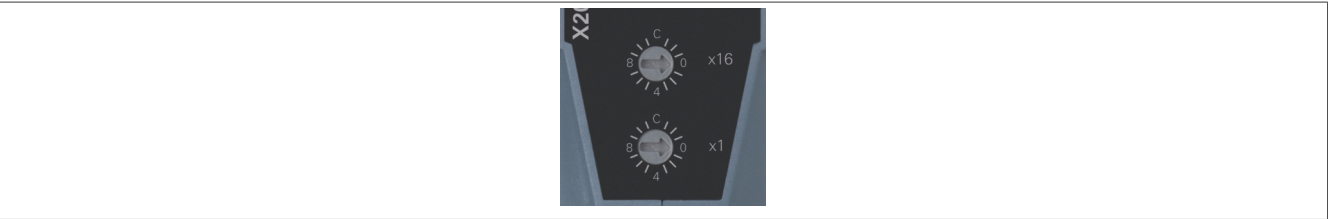
- The bus controller was configured as a PTP grandmaster ($\text{Priority1} < 128$), but it is a PTP slave.
- The bus controller was configured as a PTP slave ($\text{SlaveOnly} = \text{true}$), but is a PTP grandmaster.

Technical description

LED status indicators - Blink times



2.2.2 Number switches




Only reset mode can be enabled using the two number switches.

Switch position	Description
0x00 - 0xFE	Reset mode not active
0xFF	Reset mode active

Reset during startup

Indication of startup: LED "MS" is not yet permanently lit up green or red.



Information:
Setting the reset mode is not permitted during startup.

Reset during operation


During operation, the triggered function depends on the setting duration of the reset mode.

Function	Setting duration	LED status indicator ¹⁾	Confirmation
Setting a temporary IP address ²⁾	1 s	LED "MS": Off	-
Restart	5 s	LED "MS": 1 pulse after 5 seconds	-
Deleting a configuration	10 s	LED "MS": 2 pulses after 10 seconds	If the number switches are pressed again within 5 s, the action is performed and the bus controller is then restarted.
Deleting a safety configuration	15 s	LED "MS": 3 pulses after 15 seconds	
Resetting to the factory settings	20 s	LED "MS": 4 pulses after 20 seconds	

1) See 2.2.1 "LED status indicators".
2) Temporary IP address 192.168.1.1, see 2.3 "Setting an IP address".

"Reset mode - Set temporary IP address" example

- 1) Set the number switch to 0xFF.
- 2) After 1 s, LED "MS" goes out.
- 3) Set the number switch not equal to 0xFF within 5 s before LED "MS" blinks red with 1 pulse (mode: restart).



Information:
The temporary IP address is only set for the current boot procedure and no longer present after the device is restarted. It enables an initial connection to the device to configure a static IP address.

"Reset mode - Restart" example

- 1) Set the number switch to 0xFF.
- 2) After 1 s, LED "MS" goes out (mode: temporary IP address); after 5 s, LED "MS" blinks red with 1 pulse (mode: restart).
- 3) Set the number switch not equal to 0xFF within 5 s before LED "MS" blinks red with 2 pulses (mode: delete configuration).
- 4) The device is restarted.

"Reset mode - Delete configuration" example

- 1) Set the number switch to 0xFF.
- 2) After 1 s, LED "MS" goes out (mode: temporary IP address); after 5 s, LED "MS" LED blinks red with 1 pulse, and after 10 s it blinks red with 2 pulses (mode: delete configuration).
- 3) Set the number switch not equal to 0xFF within 5 s before LED "MS" blinks red with 3 pulses (mode: delete safety configuration).
- 4) LED "MS" lights up green and red for 5 s (mode: confirmation of the delete procedure). Within this time, turn number switch to 0xFF again to confirm and then turn again to not equal to 0xFF.
The configuration is not deleted if there is no confirmation.
- 5) The configuration is deleted, and the device is restarted.

Deleting a configuration

The following settings will be deleted:

- Network configuration
- X2X configuration
- Time synchronization configuration
- TSN configuration

Deleting a safety configuration

The following settings will be deleted:

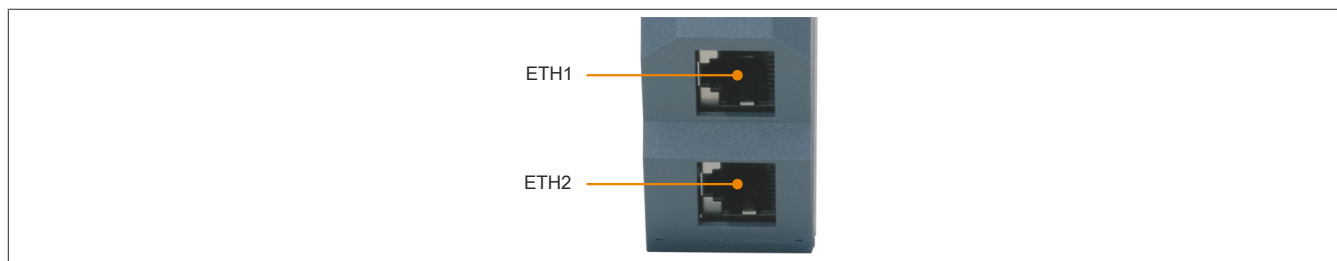
- Users / passwords
- OPC UA certificates
- Netconf certificates / SSH keys

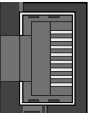
Resetting to the factory settings

Corresponds to "Deleting a configuration" and "Deleting a safety configuration".

2.2.3 Ethernet interface

For information about wiring X20 modules with an Ethernet interface, see section "Mechanical and electrical configuration - Wiring guidelines for X20 modules with Ethernet cables" in the X20 user's manual.



Interface	Pinout		
	Pin	Ethernet	
 Shielded RJ45 port	1	D1+	Data 1+
	2	D1-	Data 1-
	3	D2+	Data 2+
	4	D3+	Data 3+
	5	D3-	Data 3-
	6	D2-	Data 2-
	7	D4+	Data 4+
	8	D4-	Data 4-

2.3 Setting an IP address

Depending on the type of application used, an IP address can be assigned to the bus controller in various ways.

- Automatic assignment via DHCP server
By default, the bus controller is configured for automatic IP address assignment via a DHCP server. In machine networks with a B&R controller, the DHCP server function is provided by Automation Runtime. However, PCs or laptops with desktop operating systems such as Windows or Linux usually do not offer a DHCP server.
- Setting the temporary IP address (192.168.1.1) using the number switch. (See section [2.2.2 "Number switches"](#).)
- Configuration via OPC UA server (see section [3.2 "Establishing a connection"](#))

3 Getting started

The bus controller is delivered with factory settings. No device function or any security settings are configured in this case. For secure commissioning, the bus controller is operated in a secure environment.

Examples of secure environments:

- Connection to a network separate from the company network.
- Direct connection to the PC used for configuration.

After the security configuration has been completed, the bus controller can also be securely operated in an unsecured environment.

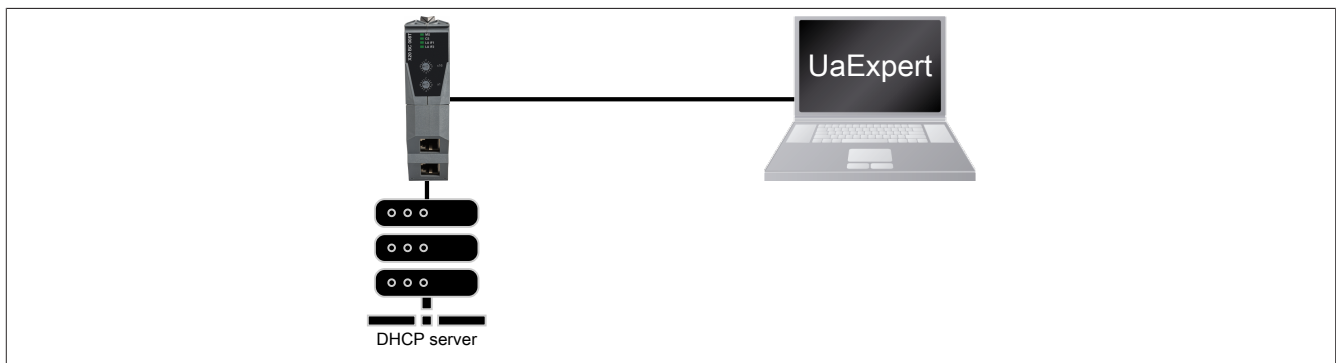
3.1 Preparation

In the examples in this documentation, the OPC UA client software "UaExpert" is used for the configuration. Other similar tools however, can also be used for the configuration.

Recommended minimum version:

- UaExpert version 1.6 or later
Download: <https://www.unified-automation.com>

To begin, the following structure can be used for initial configuration. This consists of a PC with UaExpert software, a directly connected bus controller and a DHCP server. The DHCP server can also be part of the PC.



3.2 Establishing a connection



Information:

To avoid problems when establishing a connection, see also section 5.6 "Integration in the IT network".

In the factory setting, a DHCP client is started on the bus controller and a hostname is generated depending on the product ID and MAC address. A DHCP server available on the network can therefore assign an IP address to the bus controller. In addition, multicast DNS (mDNS) is enabled on the bus controller.

The following network settings are applied by the DHCP server in the factory setting:

- IP address
- Subnet mask
- Gateway
- Hostname
- Domain
- DNS server
- NTP server

To change the factory settings (see section 3.5 "General network settings via OPC UA"), one of the following factory mechanisms must first be used for the initial connection.

3.2.1 Establishing a connection via the hostname

3.2.1.1 Determining the hostname

In order to establish a connection, the hostname of the bus controller must be known. In the factory settings, this is generated from the product ID and the MAC address of the bus controller and has the following format:

x20bc008t-[MAC address]



Information:

After the hostname is changed, the default hostname composed of product identifier and bus controller MAC address is no longer valid.

Example

For a bus controller with MAC address 00:60:65:00:22:01, the hostname is as follows:

x20bc008t-006065002201

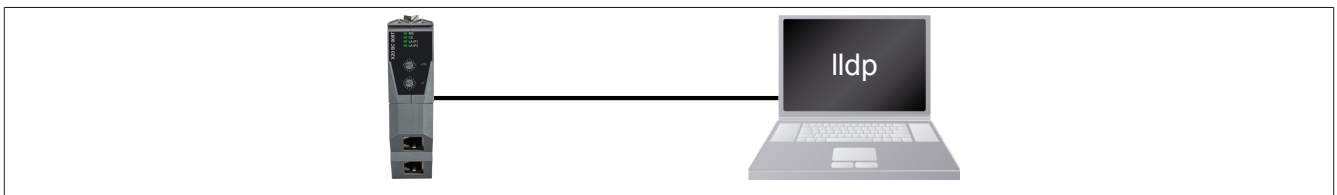
The following possibilities are available to determine the hostname:

3.2.1.1.1 Determining the hostname via housing label

The bus controller MAC address is printed on the housing along with the MAC addresses of the ports.

3.2.1.1.2 Determining the hostname via LLDP and direct connection

Alternatively, the hostname can be determined via a network connection using LLDP. The bus controller publishes the MAC address of the endpoint in the network to direct neighbor devices via the "Link Layer Discovery Protocol" (LLDP) using the name "ChassisID". This can be determined from a PC with Linux and direct device connection using LLDP, for example:

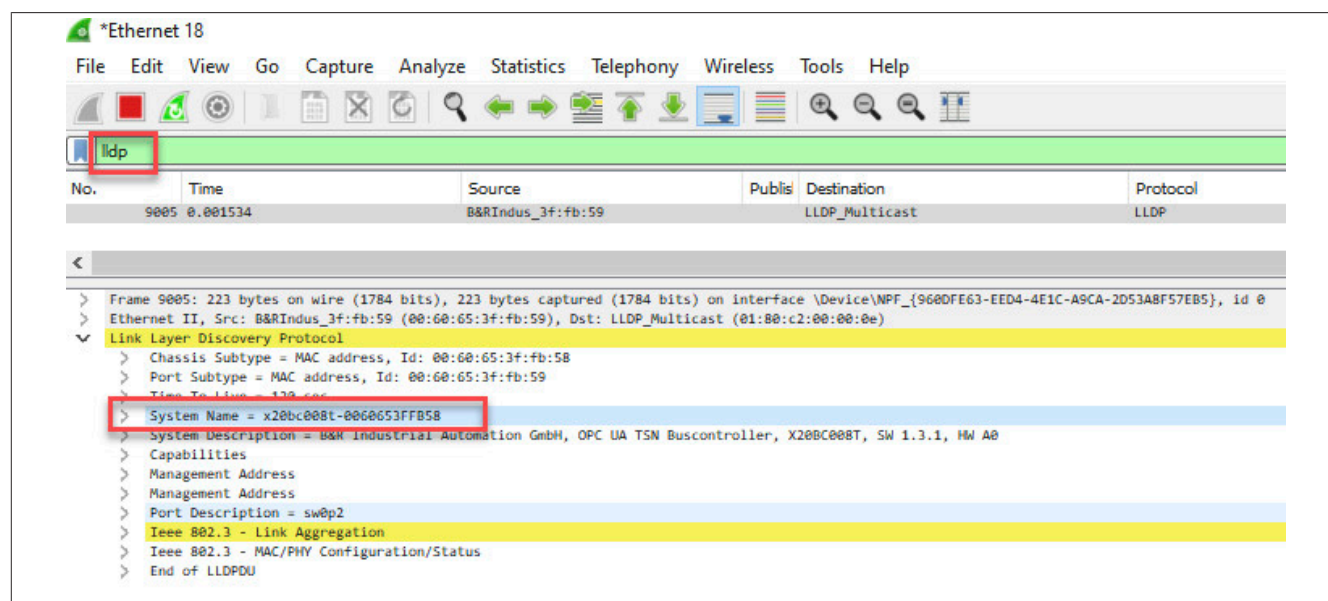


Example

```
$ lldpctl
-----
Interface:      enx9cebe8ae5553, via: LLDP, RID: 42, Time: 0 day, 01:03:00
Chassis:
  ChassisID:    mac 00:60:65:00:22:01
  SysName:      x20bc008t-006065002201.home
  SysDescr:     B&R Industrial Automation GmbH, 802.1Q OPC UA FX buscontroller, X20BC008T,
                SW 1.0.0, HW C0
  MgmtIP:       192.168.0.128
  MgmtIP:       2a02:810d:6e3f:e9a0:260:65ff:fe00:2201
  Capability:   Bridge, on
  Capability:   Router, off
  Capability:   Wlan, off
  Capability:   Station, off
Port:
  PortID:       mac 00:60:65:00:22:03
  PortDescr:    sw0p3
  PMD autoneg:  supported: yes, enabled: yes
  Adv:          100Base-TX, HD: no, FD: yes
  Adv:          1000Base-T, HD: no, FD: yes
  MAU oper type: 100BaseTXFD - 2 pair category 5 UTP, full duplex mode
-----
```

3.2.1.1.3 Determining the hostname with LLDP and connecting directly using Wireshark

If LLDP is not available on the PC, a network capture can also be performed, e.g. using Wireshark. These contain the hostname. To do this, apply the "lldp" filter in Wireshark and read out the "System name" in the details of the LLDP datagram → This corresponds to the hostname.

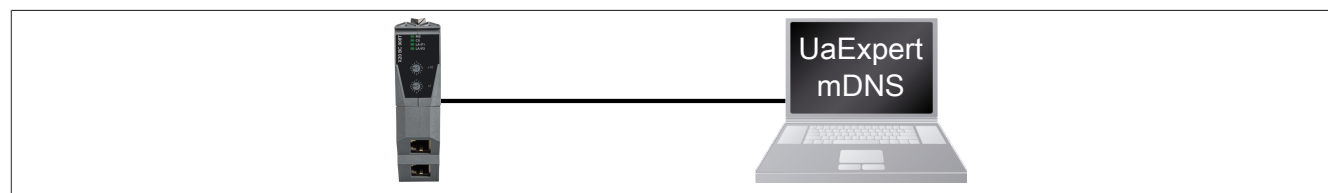


3.2.1.2 Resolving the hostname

After the hostname has been determined, it must be resolved into an IP address by the network infrastructure. There are the following possibilities to do this:

3.2.1.2.1 Resolving the hostname via mDNS

When the hostname is known, the bus controller can be addressed from the PC using this name. In this case, the connection is made via the hostname and mDNS domain ".local". The IP address does not need to be known with this possibility.



The following "Endpoint URL" can be used in UaExpert to establish the connection (see 3.4 "Creating the first user"):

opc.tcp://<Product ID>-<MAC address>.local:4840

Or for this example:

opc.tcp://x20bc008t-006065002201.local:4840



Information:

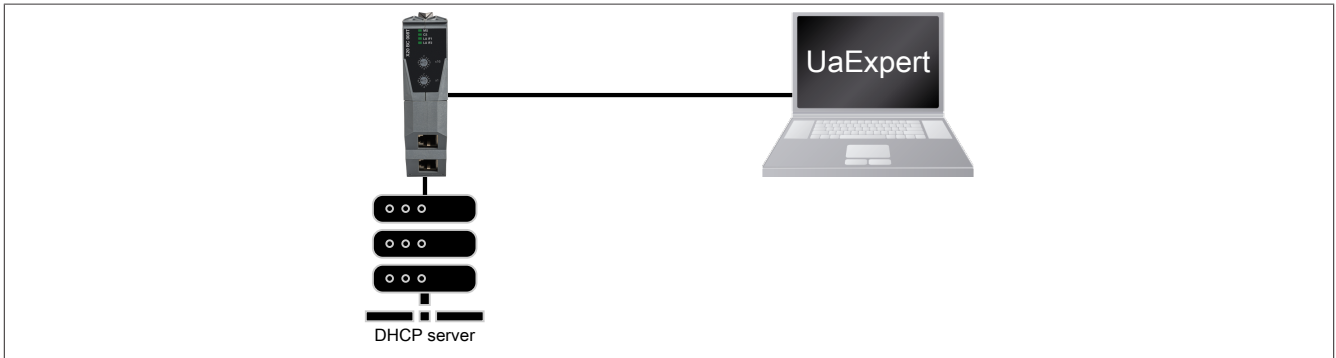
The OPC UA server on the bus controller expects incoming connections on port 4840.

Getting started

3.2.1.2.2 Resolving the hostname via DNS

In large networks with many stations or if a DHCP/DNS infrastructure is available and used, it is possible to disable mDNS via the OPC UA information model.

The connection is made via the hostname since a DHCP/DNS infrastructure exists. The IP address does not need to be known with this possibility.



The following "Endpoint URL" can be used in UaExpert to establish the connection (see [3.4 "Creating the first user"](#)):

```
opc.tcp://<Product ID>-<MAC address>:4840
```

Or for this example:

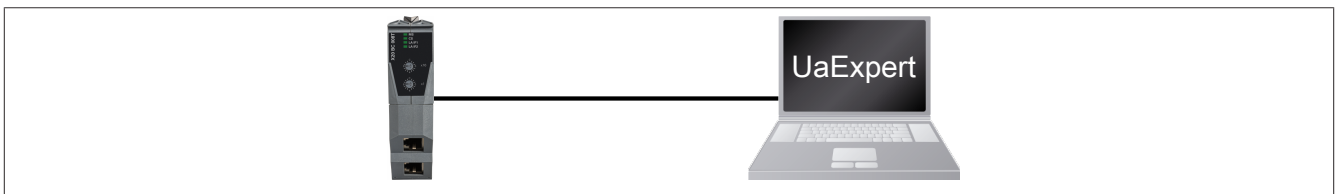
```
opc.tcp://x20bc008t-006065002201:4840
```

3.2.2 Establishing a connection with IP address

The connection can be established using a static or dynamic IP address depending on the existing infrastructure.

3.2.2.1 Static IP address

A DHCP server is not required for this method. The [number switch](#) is used to set the IPv4 address to value "192.168.1.1" for the current boot procedure.



The following "Endpoint URL" can be used in UaExpert to establish the connection (see [3.4 "Creating the first user"](#)):

```
opc.tcp://192.168.1.1:4840
```

If the IPv4 address is already configured and known, the reset procedure is not necessary. In this case, the "endpoint URL" in UaExpert is as follows:

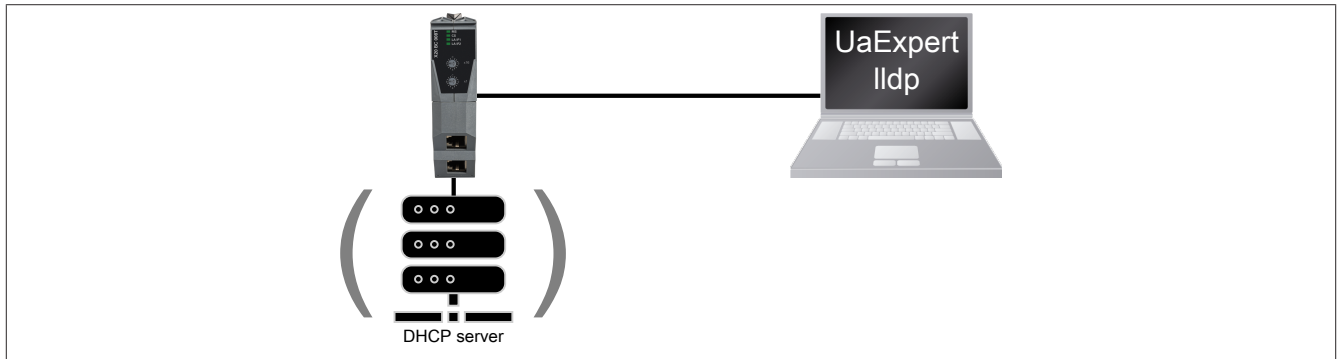
```
opc.tcp://<Known IP address>:4840
```

3.2.2.2 Dynamic or unknown IP address

There are several ways to assign an IP address to a bus controller:

- Via the DHCP server
- If the bus controller is not assigned an IP address via DHCP, the bus controller automatically generates a random IPv4 Link-Local (IPv4LL) address.

This assigned IPv4 address can be determined via LLDP (see section [Determining the hostname with LLDP and direct connection](#)) using the name "MgmtIP".

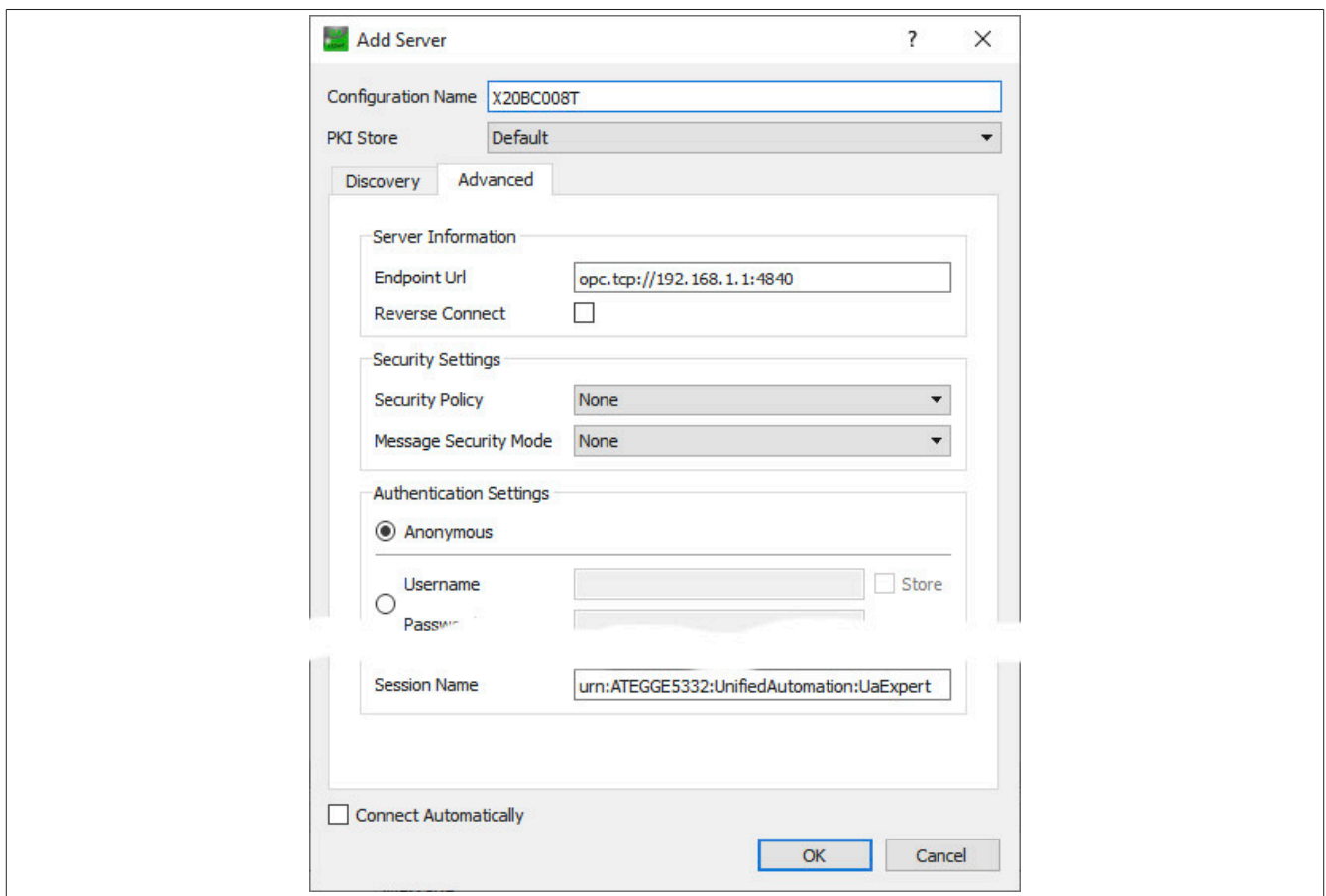


The following "Endpoint URL" can be used in UaExpert to establish the connection (see [3.4 "Creating the first user"](#)):

`opc.tcp://<Determined IP address>:4840`

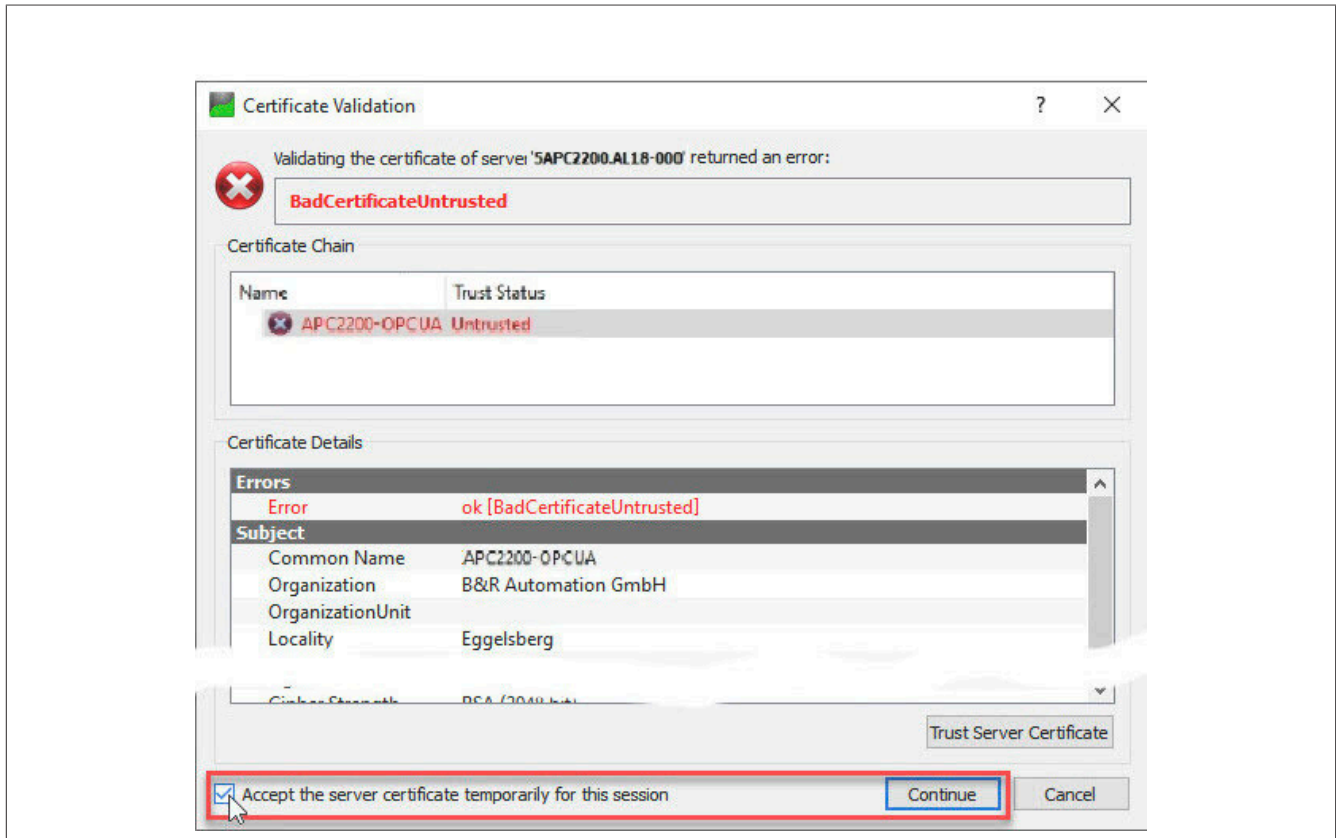
3.3 Connecting to an OPC UA client

Setting `Anonymous` must be used to establish the initial OPC UA connection. Since sensitive data will be transferred, an appropriate security policy such as `Basic256SHA256` should also be selected.



Getting started

The Bus Controller is not yet initially integrated into a public key infrastructure (PKI) and therefore only has a self-signed certificate. This certificate is correct but the client cannot verify its origin and therefore issues a warning message. In a trusted environment, however, it is safe to accept this certificate.



The certificate is accepted by selecting "Accept the server certificate temporarily for this session" and clicking on Continue.



Information:

In an untrusted environment, accepting such a self-signed certificate presents a risk. An attacker can read and corrupt the data traffic during a "man-in-the-middle" attack despite encryption.

3.4 Creating the first user



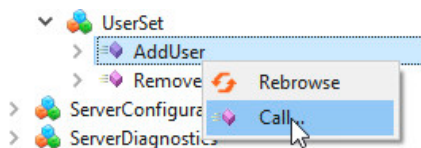
Information:

A user must be created; otherwise, no further configuration can be performed.

Creating a user

The bus controller is currently still in commissioning mode and only permits the anonymous client to call a few methods. These include creating the first user, setting the password and assignment to role SecurityAdmin.

- The first step is to create the user who is responsible for further configuration. This is done by calling method Root/Objects/Server/ServerCapabilities/UserSet/AddUser. The user dialog box is displayed by clicking on Call.



Input Arguments			
Name	Value	DataType	Description
UserName	admin		String

Output Arguments			
Name	Value	DataType	Description
UserNodeId	0	Numeric	NodeId

Result	

A successful call is displayed under "Result", and the node ID of the created user is returned.

Input Arguments			
Name	Value	DataType	Description
UserName	admin		String

Output Arguments			
Name	Value	DataType	Description
UserNodeId	1	String	User.admin@UserObject

Result	
Succeeded	

Getting started

Assigning a password

- The name of the newly created user is displayed in the information model. Method Root/Objects/Server/ServerCapabilities/UserSet/<NAME>/SetPassword is called to configure the password. By clicking on Call..., the password dialog box is displayed.

UserSet

>

AddUser

>

RemoveUser

>

admin

>

AddSshKey

>

DisablePassword

>

HasPassword

>

RemoveSshKey

>

Roles

>

SetPassword

>

SshKeys

>

UserName

ServerConfiguration

Rebrowse

Call...

Call SetPassword on admin

Input Arguments		
Name	Value	DataType Description
Password	secret	String

Result

Call

Close



Information:

The password is transferred from the client to the bus controller in encrypted form. To prevent unauthorized access to the bus controller, it is important to ensure that the password cannot be seen by unauthorized persons when it is entered.

Assigning role SecurityAdmin

- As a next step, the user must be assigned the permission required for further configuration as "SecurityAdmin". Method Root/Objects/Server/ServerCapabilities/RoleSet/SecurityAdmin/AddIdentity is called for this.

RoleSet

>

Anonymous

>

AuthenticatedUser

>

ConfigureAdmin

>

Engineer

>

Observer

>

Operator

>

SecurityAdmin

>

AddIdentity

>

Identities

>

RemoveIdentity

>

Supervisor

Call AddIdentity on SecurityAdmin

Input Arguments		
Name	Value	DataType Description
Rule	Click '...' to display value	IdentityMappingRuleType

Result

Call

Close

After clicking on "...", entry "1 (UserName)" can be selected as CriteriaType. The username is specified as "Criteria".

Edit Value

Name	Value
IdentityMappingRuleType	
CriteriaType	1 (UserName)
Criteria	admin

Write

Cancel

Clicking on Write applies the data and closes the dialog box. Dialog box SecurityAdmin is then closed by clicking on Call and the user is logged in as SecurityAdmin.

Displaying the role assignment

• Multiple roles can be assigned to a user, or multiple users can perform the same role. These can be viewed via properties Root/Objects/Server/ServerCapabilities/RoleSet and ... /ServerCapabilities/UserSet.

Example

Output of all users that are logged in as SecurityAdmin.

<ul style="list-style-type: none"> RoleSet <ul style="list-style-type: none"> Anonymous AuthenticatedUser ConfigureAdmin Engineer Observer Operator SecurityAdmin <ul style="list-style-type: none"> AddIdentity Identities RemoveIdentity Supervisor ServerProfileArray SoftwareCertificates 	<ul style="list-style-type: none"> Value <ul style="list-style-type: none"> SourceTimestamp 03-Mar-21 16:01:29.198 SourcePicoSeconds 0 ServerTimestamp 03-Mar-21 16:01:29.198 ServerPicoSeconds 0 StatusCode Good (0x00000000) Value IdentityMappingRuleType Array[1] <ul style="list-style-type: none"> [0] IdentityMappingRuleType <ul style="list-style-type: none"> CriteriaType 1 (UserName) Criteria admin
--	---

Example

Output of all roles that are assigned to the user with name "admin".

<ul style="list-style-type: none"> UserSet <ul style="list-style-type: none"> AddUser RemoveUser admin <ul style="list-style-type: none"> AddSshKey DisablePassword Password RemoveSshKey Roles SetPassword SshKeys 	<ul style="list-style-type: none"> Value <ul style="list-style-type: none"> SourceTimestamp 03-Mar-21 16:06:27.937 SourcePicoSeconds 0 ServerTimestamp 03-Mar-21 16:06:27.937 ServerPicoSeconds 0 StatusCode Good (0x00000000) Value String Array[1] <ul style="list-style-type: none"> [0] SecurityAdmin
---	---

Further assignments

The possibilities of the client logged in anonymously are limited to creating the first user, setting the password and assigning role SecurityAdmin.

• To make further assignments and settings, the connection to the bus controller must be disconnected and a new session authenticated with username and password must be started.

Security Settings

Security Policy

Basic256Sha256

Message Security Mode

Sign & Encrypt

Authentication Settings

Anonymous

☐

Username

admin

☒ Store

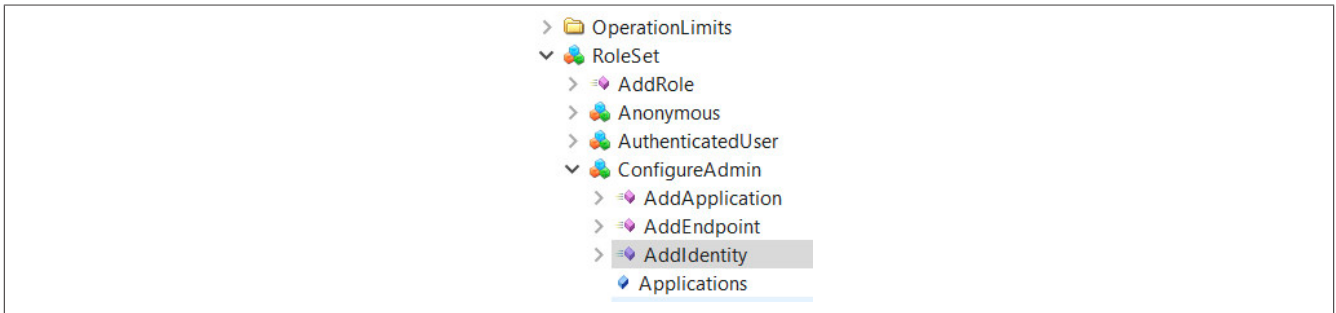
☒

Password

•••••

Getting started

- To make additional settings, the user must also be assigned role `ConfigureAdmin`. Method `Root/Objects/Server/ServerCapabilities/RoleSet/ConfigureAdmin/AddIdentity` is called for this and the name is assigned as described in [Assigning role SecurityAdmin](#).



3.5 General network settings via OPC UA

It is possible to make a valid network configuration via OPC UA.

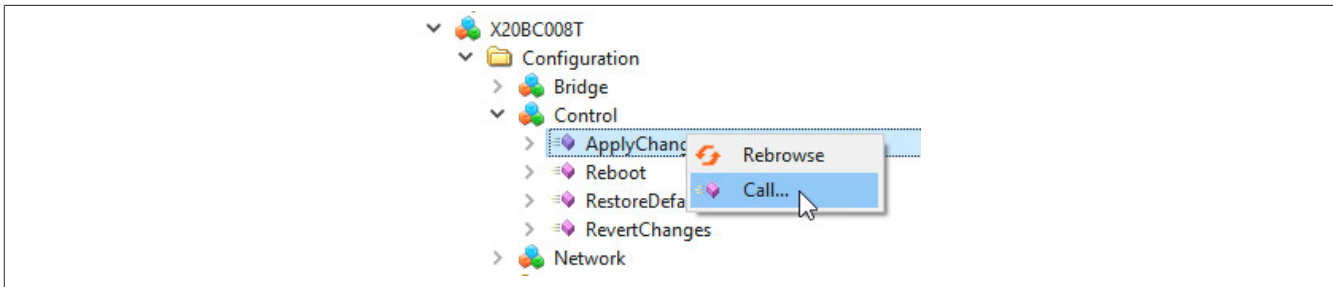
- For this, the various parameters for the network configuration are called under `Root/Objects/DeviceSet/X20BC008T/Configuration/Network` and written to accordingly.

The screenshot shows a tree view of OPC UA objects. The 'DeviceSet' object is expanded, showing its children: 'DeviceFeatures', 'X20BC008T', 'Configuration', 'Bridge', 'Control', 'Network', 'EnableDHCP', 'EnableMulticastDNS', 'Gateway', 'Hostname', 'IP-Address' (highlighted), 'Netmask', 'PrimaryDNS', and 'SecondaryDNS'. The 'Network' object is also expanded, showing its children: 'EnableDHCP', 'EnableMulticastDNS', 'Gateway', 'Hostname', 'IP-Address', 'Netmask', 'PrimaryDNS', and 'SecondaryDNS'.

Parameter	Value
SourceTimestamp	01.01.1970 02:23:28.219
SourcePicoSeconds	0
ServerTimestamp	01.01.1970 02:23:28.219
ServerPicoSeconds	0
StatusCode	Good (0x00000000)
Value	192.168.1.1
DataType	String
NamespaceIndex	0
IdentifierType	Numeric
Identifier	12 [String]

Node name	Description
EnableDHCP	Enables or disables the DHCP client functionality. - If an IP assignment by a DHCP server is missing, the bus controller is assigned a random link local address from range 169.254.0.0/16. IPv4LL (RFC3927). - If the DHCP client is enabled, parameters Gateway, IP address, Netmask as well as Primary DNS and Secondary DNS are obtained from the DHCP server.
Gateway	Configures the default gateway IP address. - If parameter EnableDHCP is set, a gateway address can additionally be obtained from the DHCP server. - If parameter Gateway is set, the manual configuration is used and the address obtained from the DHCP server is ignored.
Hostname	Configures the hostname.
IP address	Configures a static IP address. - If parameter EnableDHCP is set, the parameter is ignored and the IP address transmitted by the DHCP servers is used.
Primary DNS Secondary DNS	Configures a primary or secondary DNS server. - If parameter EnableDHCP is set, additional addresses for DNS servers can be obtained from the DHCP server. - If at least 1 DNS server is set manually, the manual configuration is used and the addresses obtained from the DHCP server are ignored.
Netmask	Sets the subnet mask. - If parameter EnableDHCP is set, this parameter is ignored and the subnet mask obtained from the DHCP server is used.
EnableMulticastDNS	Enables or disables multicast DNS (mDNS). - mDNS is enabled in the factory setting in order to be able to address the bus controller via the hostname even if there is no network infrastructure.

- Method `Root/Objects/DeviceSet/X20BC008T/Configuration/Control/ApplyChanges` must be called to save the new configuration data.



Information:

The new network configuration is only applied when the bus controller is restarted.

3.6 Time synchronization

Information about the current time is required in order for the bus controller to operate. This is mainly needed to process digital certificates correctly and to correctly set the timestamps of OPC UA values.

The following description shows how to configure "WallClock" to enable synchronization over the Network Time Protocol (NTP). For the required parameters, see `Root/Objects/DeviceSet/X20BC008T/Configuration/TimeSynchronization/WallClock`.

- To use NTP for time synchronization, the protocol for synchronization must be set to NTP via parameter `.../WallClock/TimeSyncProtocol`.


A screenshot of the configuration tree for 'X20BC008T'. The tree structure is: X20BC008T > Configuration > Bridge > Control > Network > TimeSynchronization > WallClock > NTP. The 'NTP' folder is expanded, showing several parameters: 'TimeServer01', 'TimeServer02', 'TimeServer03', 'TimeServer04', 'PTP', 'DomainNumber', 'Priority1', 'Priority2', 'SlaveOnly', 'SyncOffsetNs', 'TimeSyncProtocol', and 'WorkingClock'. The 'TimeSyncProtocol' parameter is selected and highlighted with a blue box.

UserKolePermissions	KolePermission type Array[UI]
AccessRestrictions	BadAttributeIdInvalid (0x80350000)
Value	
SourceTimestamp	01.01.1970 02:30:43.695
SourcePicoseconds	0
ServerTimestamp	01.01.1970 02:30:43.695
ServerPicoseconds	0
StatusCode	Good (0x00000000)
Value	0 (NONE)
DataType	0 (NONE)
NamespaceIndex	1 (PTP)
IdentifierType	2 (NTP)
Identifier	3012
ValueRank	-1 (Scalar)
AccessRestrictions	BadAttributeIdInvalid (0x80350000)

- The next configuration step depends on the type of network in which the bus controller is located.
 - If time servers are obtained in the network via DHCP, no time server must be set. The time servers obtained by the DHCP server are used instead.
 - If no time server is obtained in the network via DHCP, at least 1 time server must be configured in subobject NTP. For this, the hostname or the IP address must be entered in attribute Value of node TimeServer0x.

Getting started

- Method Root/Objects/DeviceSet/X20BC008T/Configuration/Control/ApplyChanges must be called to save the new configuration data.



Information:

The new network configuration is only applied when the bus controller is restarted.

3.7 Restart and reset

A restart can be triggered via method Root/Objects/DeviceSet/X20BC008T/Configuration/Control/Reboot. Configurations previously saved using method ApplyChanges are applied and used when the bus controller is started up.

If the network configuration was changed, the bus controller is only accessible under the new settings after a restart. In case of connection problems, the connection settings in UaExpert must therefore be adjusted according to the new configuration.

3.8 Updating the self-signed certificate

The bus controller has a method in the information model that can be used to easily generate a new self-signed certificate, which contains the required application-specific information.

For the update, method UpdateSelfSignedCertificate must be called by clicking on Call under Root/Objects/Server/ServerConfiguration (1).

3

ServerConfiguration

>

ApplyChanges

>

AuthorizationServices

>

CertificateGroups

>

CreateSigningRequest

>

GetRejectedList

>

KeyCredentialConfiguration

>

MaxTrustListSize

>

MulticastDnsEnabled

>

ServerCapabilities

>

SupportedPrivateKeyFormats

>

UpdateCertificate

1

UpdateSelfSignedCertificate

>

ServerDiagnostics

>

ServerRedundancy

>

ServerStatus

Rebrowse

CN...

Call UpdateSelfSignedCertificate on ServerConfiguration

Input Arguments			
Name	Value	DataType	Description
ExpiryDate	2030-01-01T00:00:00.000Z	Date	
Subject	2trial Automation GmbH/CN=X20BC008T-	String	
DNS		String	If empty then the configured hostname is used
IncludeIPAddress	<input checked="" type="checkbox"/>	Boolean	If set then the IP-Address is included into the certificate
IPAddress		String	If empty and a static IP-Address is configured then the configured IP is used

Output Arguments			
Name	Value	DataType	Description
ApplyChangesRequired	<input type="checkbox"/>	Boolean	

Result

CallClose

The desired values are entered in the method dialog box (2). The method has the following arguments:

Argument	Description																								
Input arguments																									
ExpiryDate	Expiration date until which the certificate is valid. Information: The input is only evaluated to the exact day.																								
Subject	Sequence of X.509 name-value pairs separated by character "/". The following names are provided: <table><tr><th>Name</th><th>Complete name</th><th>Description</th></tr><tr><td>CN</td><td>CommonName</td><td>Name of the product or equivalent information</td></tr><tr><td>O</td><td>Organization</td><td>Name of the organization that operates the bus controller.</td></tr><tr><td>OU</td><td>Organization unit</td><td>Organizational unit</td></tr><tr><td>DC</td><td>Domain component</td><td>Domain of the organization</td></tr><tr><td>L</td><td>Locality</td><td>Town or city</td></tr><tr><td>S</td><td>State</td><td>State</td></tr><tr><td>C</td><td>Country</td><td>2-character country code</td></tr></table> Information: Specification of values /CN and /O is mandatory. Example "/O=B&R Industrial Automation GmbH/CN=X20BC008T-OPCUA/DC=X20BC008T/DC=ma- chine/DC=customer/DC=com"	Name	Complete name	Description	CN	CommonName	Name of the product or equivalent information	O	Organization	Name of the organization that operates the bus controller.	OU	Organization unit	Organizational unit	DC	Domain component	Domain of the organization	L	Locality	Town or city	S	State	State	C	Country	2-character country code
Name	Complete name	Description																							
CN	CommonName	Name of the product or equivalent information																							
O	Organization	Name of the organization that operates the bus controller.																							
OU	Organization unit	Organizational unit																							
DC	Domain component	Domain of the organization																							
L	Locality	Town or city																							
S	State	State																							
C	Country	2-character country code																							
DNS (optional)	Hostname or fully qualified domain name (FQDN) of the bus controller. If an empty string is specified for this parameter, the configured hostname of the bus controller is entered in the certificate. Note: If the complete fully qualified domain name is not specified for this parameter, an error message is returned each time a new connection is established that states that the hostname does not match. The connection is established anyway.																								
IncludeIPAddress	Specifies whether an IP address should be entered in the certificate. Entering the IP address is necessary if the IP address is statically assigned and clients access the bus controller via the IP address (for example, via URL opc.tcp://192.168.1.1:4840). If the IP address is obtained via DHCP server, it does not make sense to enter an IP address in the certificate since it is assigned dynamically and therefore does not remain constant.																								
IP address (optional)	IP address that should be entered in the certificate. If an empty string is specified here and IncludeIPAddress is set, the configured IP address is entered in the certificate.																								
Output arguments																									
ApplyChangesRequired	Indicates whether method Root/Objects/Server/ServerConfiguration/ApplyChanges can be executed to apply the changes.																								

When the certificate has been successfully created, method Root/Objects/Server/ServerConfiguration/ApplyChanges (3) must then be called in order to apply the changes.



Information:

Calling method `ApplyChanges` **disconnects all connected clients. A new connection is only possible when the new certificate is trusted.**



Information:

Since private keys may be transferred for certificate management, the call is only possible if there is an encrypted connection between the bus controller and OPC UA client.

3.9 Real time with TSN and PubSub

The TSN configuration can be used in conjunction with the PubSub configuration to establish data exchange in real time.

3.9.1 Limitations for real-time operation

- Multicast IP addresses must be used. Unicast IP addresses are only supported for best-effort traffic.
- Only 1 real-time publisher and 1 real-time subscriber are supported.

3.9.2 Requirements

- AS 6.1.1.14
- AR 6.1.1
- OPC UA FX Technology Package 6.3.0
- Bus controller firmware 1.6.0
- AS X20BC008T hardware upgrade 1.0.0.0

3.9.3 TSN configuration on the controller

In order to establish TSN communication with the bus controller, the TSN configuration must be established on the controller.

For details, see "Communication → OPC UA FX → PubSub real-time communication between two TSN-capable network subscribers" in Automation Help. The following steps can be followed in this order:

- 1) Preparing the project
- 2) Configuring the TSN switch
- 3) Create certificate and SSL configuration configuration
- 4) Configuring PTP time synchronization
- 5) Configuring the system timer and task classes
- 6) Creating a publisher and subscriber program
- 7) Configuring the OPC UA client/server and default view
- 8) Generating the publisher configuration
- 9) [3.9.4 "PubSub configuration on the bus controller"](#)
- 10) Creating a subscriber configuration
- 11) Compiling the project and checking PubSub communication

Notes regarding step 2 "Configuring a TSN switch"

It is important to note that the bus controller must be added to the hardware tree parallel to the switch (see image) and its integrated switch must be configured.

The bus controller has an internal 3-port switch, with one port connected to the internal endpoint. The other two ports are routed externally and can be used for daisy chaining. The internal port is configured automatically, which means that only the two external ports are configurable in the configuration.

When configuring the real-time windows for the bus controller, it is important to note that no real-time window is permitted to be configured on the switch at the time when the bus controller is transmitting real-time frames.

The transmission time of the real-time frames is determined by PublishingOffset in the PubSub configuration and is 0 in the default configuration. To ensure a sufficient time reserve, a window with offset 0 μ s and a size of 30 μ s should therefore be placed at the beginning of the cycle. A real-time window can then be configured in which the buffered frames are exchanged between the switches.

The size of the real-time window depends on the amount of data that is transferred via the switch. Around 12.2 μ s can be assumed for a full frame with 1500 bytes. The real-time frames to be transferred are usually much smaller, however.

A window can then be set up for the remaining traffic. For stable operation, a small window is required at the end of the cycle that is open for all performance classes.

For additional information, see "Communication → OPC UA FX → TSN network configuration → TSN network configuration in Automation Studio → Detailed information about the TSN network configuration".

The screenshot displays the Automation Studio configuration environment. On the left, the 'Physical View' shows a hierarchical tree of components. The component 'X20BC008T' is highlighted with a red box. The main window on the right, titled 'X20BC008T [Configuration]', shows the configuration settings for this device. The settings are organized into a tree structure with columns for Name, Value, Unit, and Description.

Name	Value	Unit	Description
X20BC008T			
Connection settings			
Authentication type	username & password		
User name	admin		
Password	*****		
Address settings			
Initial connection settings			
Automatic address detection	off		
Connection mode	IP address		
IP address	192.168.1.50		
Device configuration			
Set static IP address	off		
Set hostname	off		
TSN Switch settings			
Persistent configuration	no		
Port 1	manual configuration		
VLAN IDs	1100		
Real time windows			
Cycle time	400.000	µs	
Real time window 1			
Offset	30.080	µs	
Duration	70.080	µs	
Priority Code Points	6		
Real time window 2			
Offset	100.160	µs	
Duration	280.000	µs	
Priority Code Points	0-5,7		
Real time window 3			
Offset	380.160	µs	
Duration	19.840	µs	
Priority Code Points	0-7		
Real time window 4			
Offset	0	µs	
Duration	0	µs	
Priority Code Points	6		
Port 2	use Port 1 configuration		

Notes regarding step 8 "Creating a publisher configuration"

In this step, it is important to note that the VLAN is enabled and the same VLAN ID has been used as in the switch configurations. In addition, "isochronous" operating mode must be used so that PubSub handling in Automation Studio takes place in real time.

Notes regarding 10 "Creating a subscriber configuration"

Before this step can be carried out, the publisher configuration must be built on the bus controller.

3.9.4 PubSub configuration example

Configuring the publisher



Information:

Separate "Connections" should be created for the publisher and subscriber configuration to avoid combining ReaderGroups and WriterGroups within one "Connection". Sharing would severely limit the flexibility of the connection parameters.

Requirements

The following conditions are required to make a PubSub configuration with UaExpert:

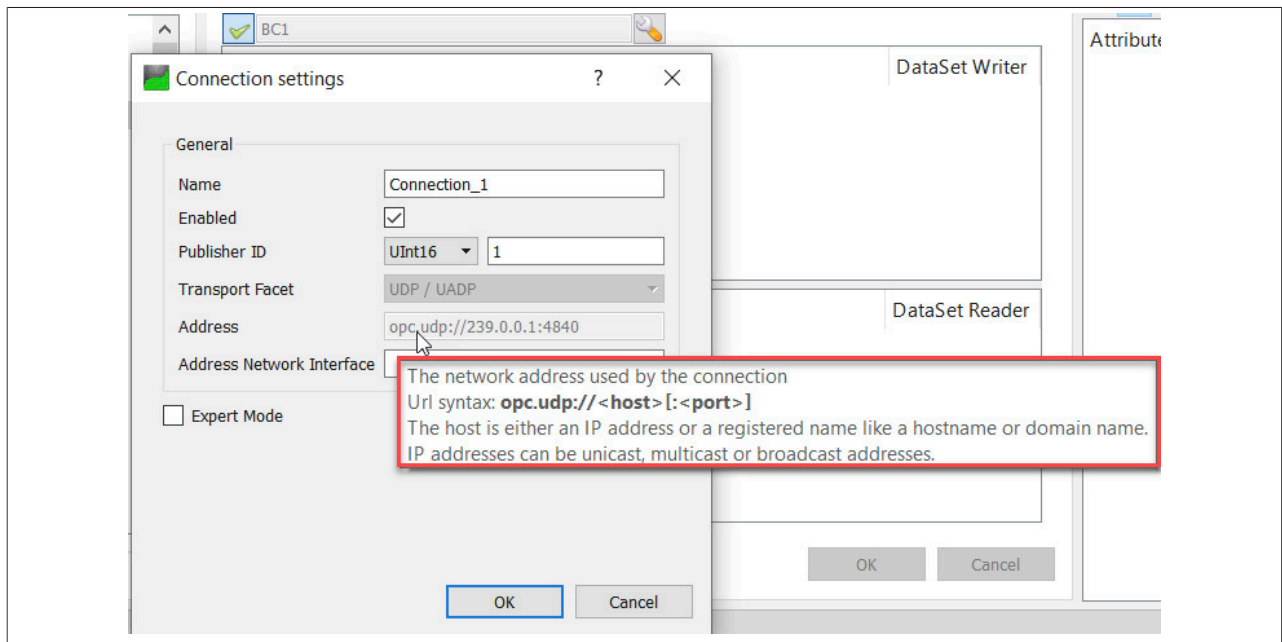
- Installation of UaExpert version 1.7.1 or later
- A valid license. This can be obtained from Unified Automation. Alternatively, the tool at <https://github.com/br-automation-com/RnCommTest-Windows> can also be used.
- A connection to the bus controller has been established with UaExpert.
- The modules present on the X2X Link network have already been configured (see 6.2.1 "X2X Link commissioning").
- All modules are displaying a constantly lit green LED status indicator.



Information:

For documentation of the PubSub configuration in UaExpert, see "Help - UaExpert Manual - Index - PubSub Config View".

An explanation of a configuration field in UaExpert can be displayed by placing the mouse pointer over it.

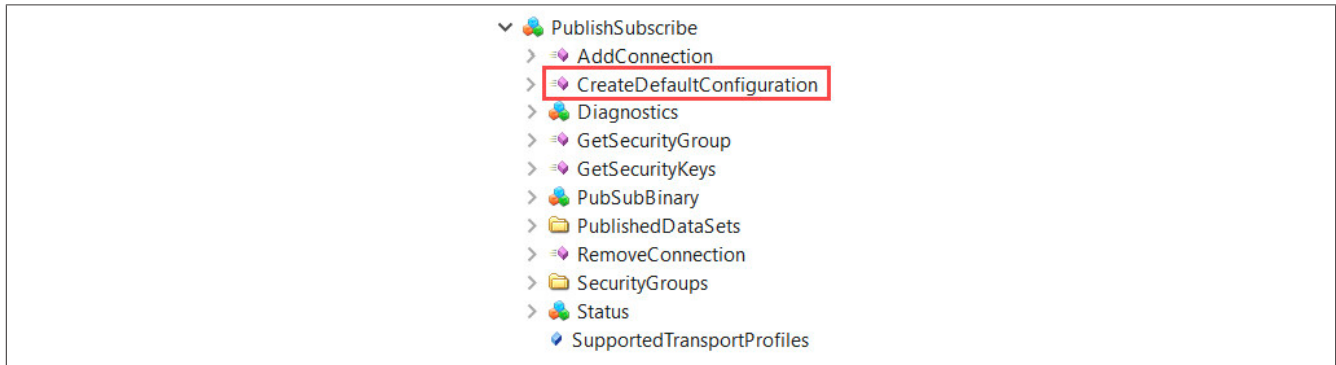


3.9.4.1 Configuring the bus controller as a publisher with UaExpert

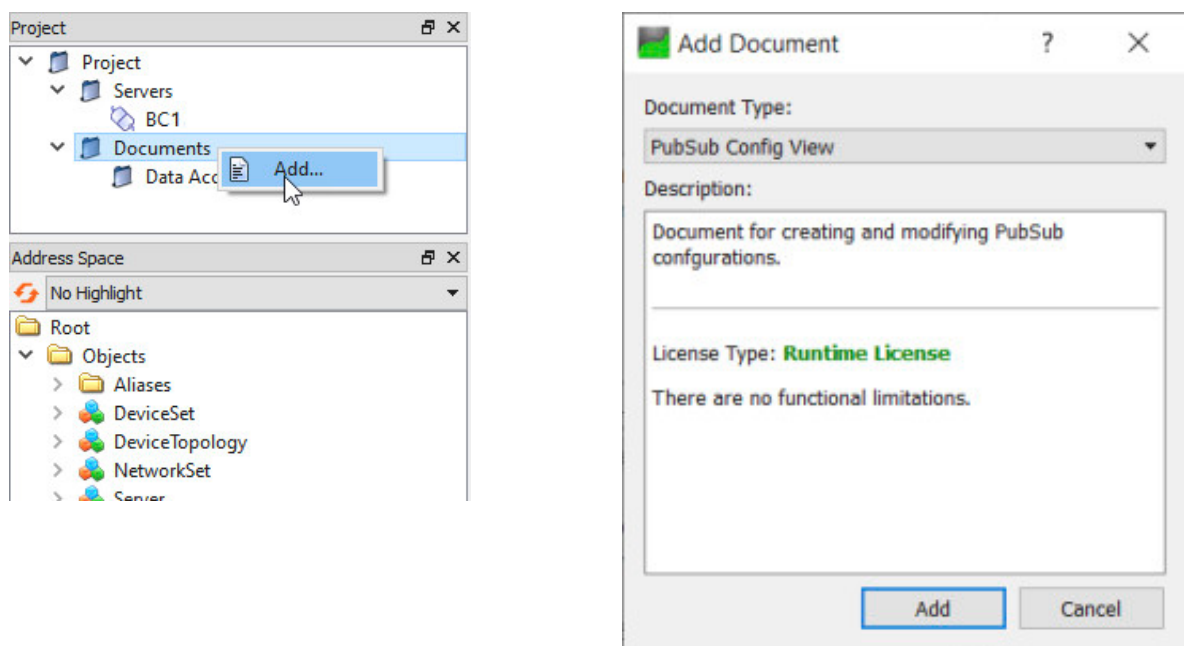
Generating and retrieving the default publisher

- Call method `CreateDefaultConfiguration` at the bus controller under `Root/Objects/Server/PublishSubscribe`.

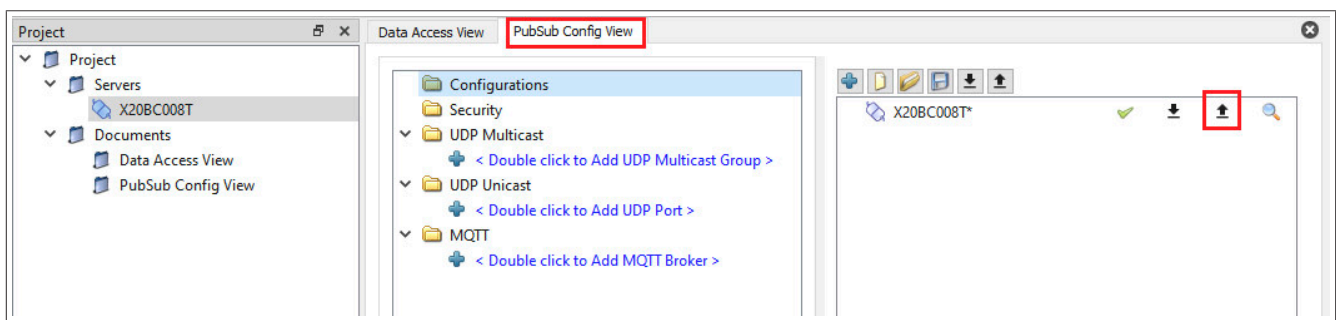
This creates a default configuration for PubSub that significantly reduces and simplifies the configuration effort. It contains valid values for all essential parameters, and all input process data of the configured I/O modules is already added to the publisher configuration.



- Click on **Add** in the shortcut menu of **Documents** in the UaExpert project window. A dialog box opens. Select document type **PubSub Config View** in this dialog box and confirm by clicking on **Add**.



- Upload the generated default publisher configuration file. This is done by selecting folder **Configurations** under **PubSub Config View** and clicking the upload arrow.



Getting started

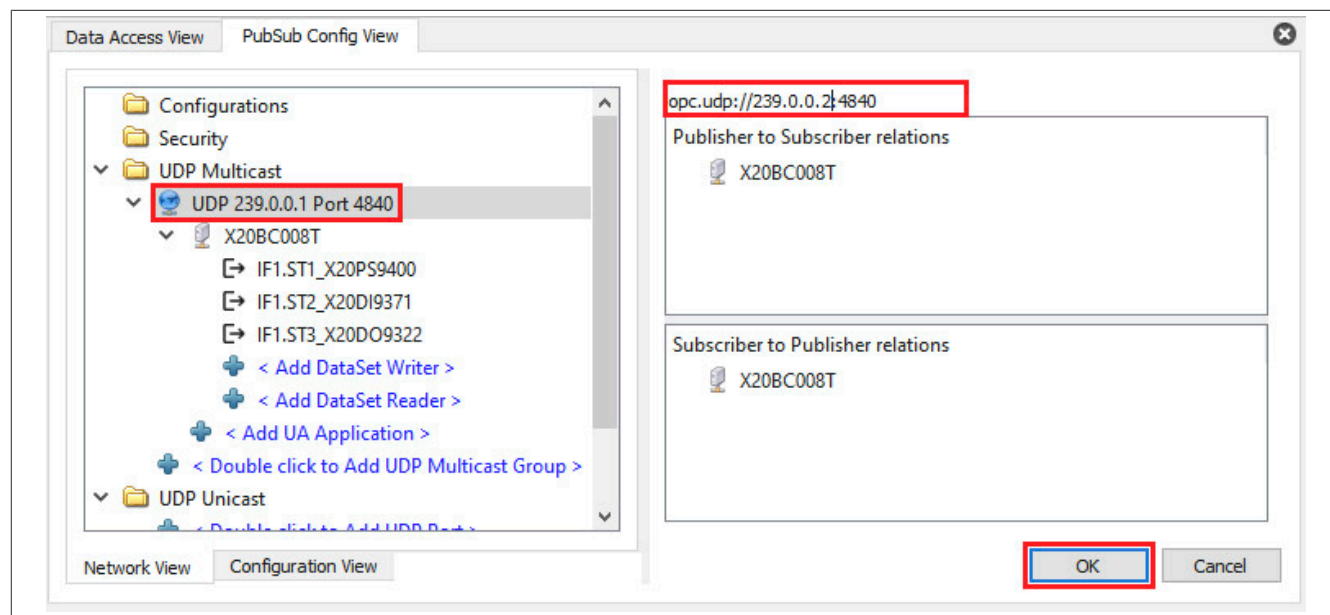
Adjusting the parameters

Only the parameters described here should be adjusted if necessary.

- Destination IP address

The destination IP address is defined during network planning and is part of a UDP multicast group in UaExpert. Publishers and their subscribers must be in the same multicast group.

opc.udp://<Destination IP address>:<Port>

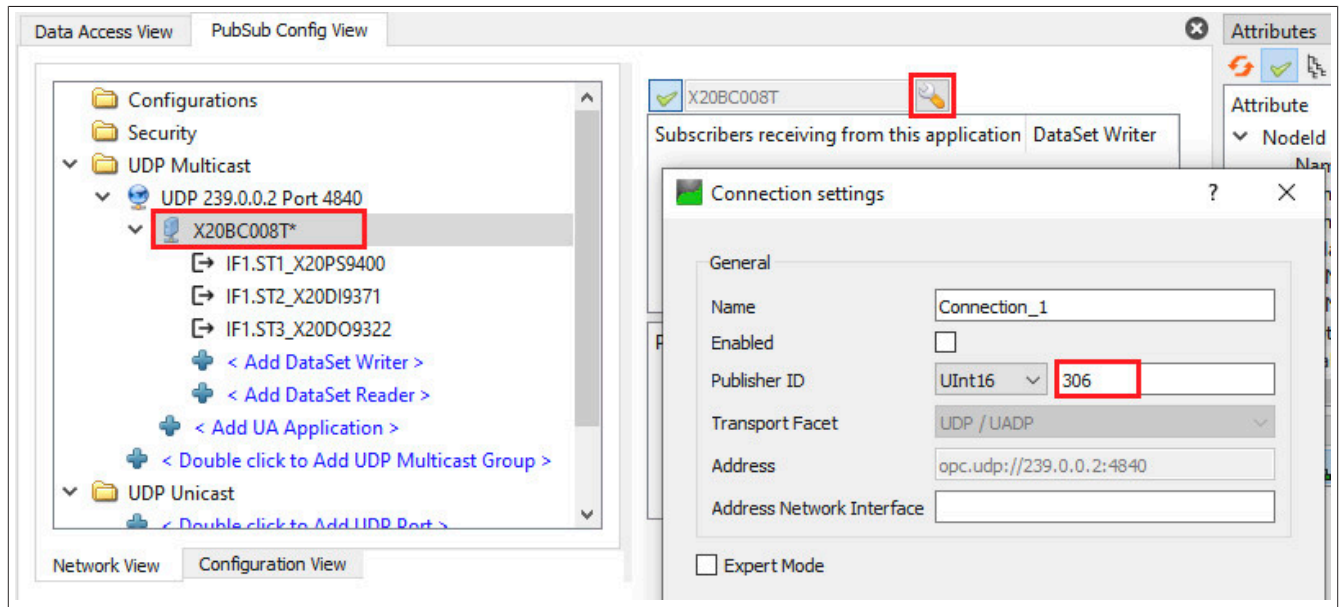


Information:

When connecting to a TSN stream, it is important to ensure that a multicast IP address is used that matches the multicast DMAC (destination MAC address).

- Publisher ID

The publisher ID must be unique in the network, i.e. it is only permitted to exist once for a publisher.



Information:

The publisher ID of the default publisher is derived from the last two positions of the bus controller IP address and thus unique in the network for small networks. It therefore does not necessarily have to be changed.



Information:

The PublisherID only supports data type UINT16.

Getting started

- Publishing interval

The publishing interval (in milliseconds) defines the period with which the DataSet is updated (i.e. sent) and is defined within a WriterGroup. This means that a change to DataSetWriter also affects other DataSetWriters in a WriterGroup. These cannot be adjusted individually.

The screenshot displays the 'PubSub Config View' of the X20BC008T configuration tool. On the left, a tree view shows the configuration hierarchy: 'Configurations' > 'Security' > 'UDP Multicast' > 'UDP 239.0.0.1 Port 4840' > 'X20BC008T*' > 'IF1.ST1_X20PS9400' > 'IF1.ST2_X20DI9371' (highlighted with a red box). Below this tree are buttons for adding Data Set Writers, Readers, UA Applications, UDP Multicast Groups, UDP Ports, and MQTT Brokers. On the right, the 'Published DataSet' dialog is open. It shows a table of fields with columns: '#', 'Field Name', 'Data Type', and 'Published Variable NodeId'. The first row is 'IF1.ST2.Channels@DigitalInput01' with a Boolean data type. Below the table, the 'DataSet Writer' tab is active. It shows a 'Group selection' dropdown set to 'WriterGroup_1 | 2ms | UADP-Periodic-Fixed'. The 'General' section contains several settings: 'Name' (WriterGroup_1), 'Enabled' (checked), 'Security Mode' (None), 'Max Network Message Size' (1400), 'Writer Group ID' (1), 'Publishing Interval' (2,00, highlighted with a red box), 'Keep Alive Time' (6,00), and 'Header Layout' (UADP-Periodic-Fixed). There is also an 'Expert Mode' checkbox.



Information:

When the default configuration is generated, the X2X cycle time is used as the publishing interval.

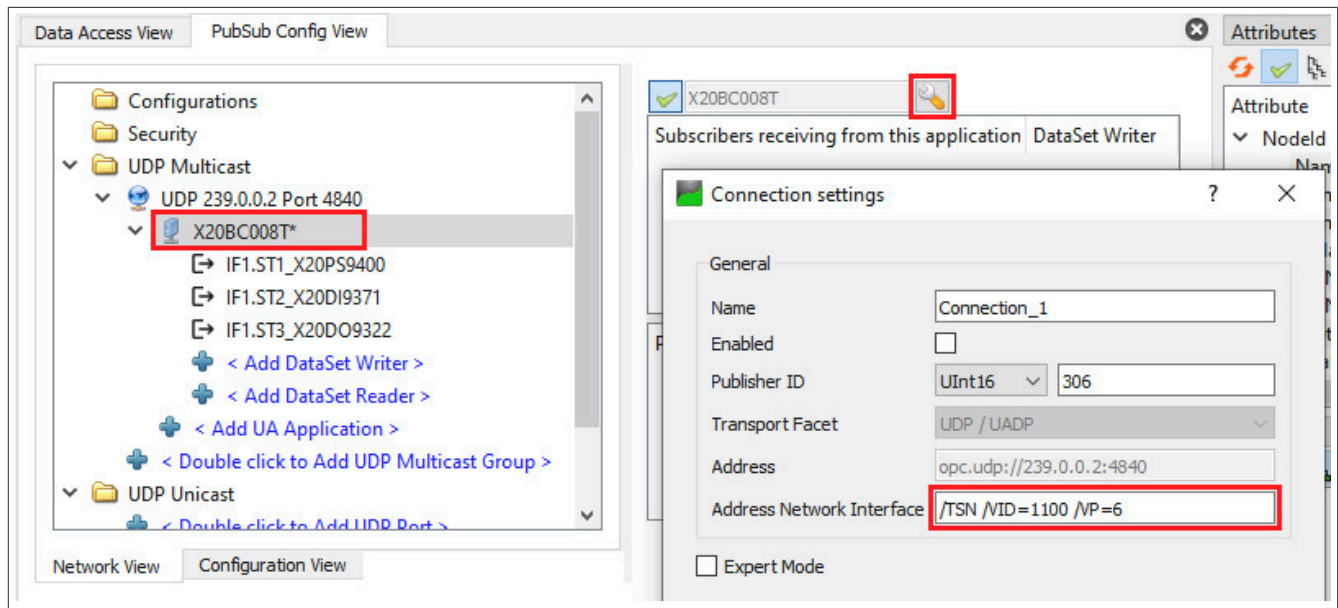
- VLAN parameters (optional)

These parameters are defined in field Address network interface in Connection settings. They are used in the publisher context to bind a publisher to a TSN stream as a talker or to tag the publisher datagram with a VLAN tag.

The following VLAN parameters can be used:

- VLAN ID: `"/VID=<VLAN ID>"`
- Priority code point (PCP): `"/VP=<PCP>"`
- Source IP address:¹⁾ `"/VIP=<Source IP address>"`

- 1) Optional parameter. If specified, this parameter is used as the source IP address for the publisher datagram. If not specified, the bus controller host IP address is used.
The subnet mask is also optional and corresponds to the value "32" if not specified. 2 variants of this parameter are possible: `"/VIP=<Source IP address>"` or `"/VIP=<Source IP address>/<Subnet mask>"`.



Information:

TSN parameters /TSN /VID and /VP must be specified for the publisher to function in real time.

Getting started

Transferring the configuration to the bus controller

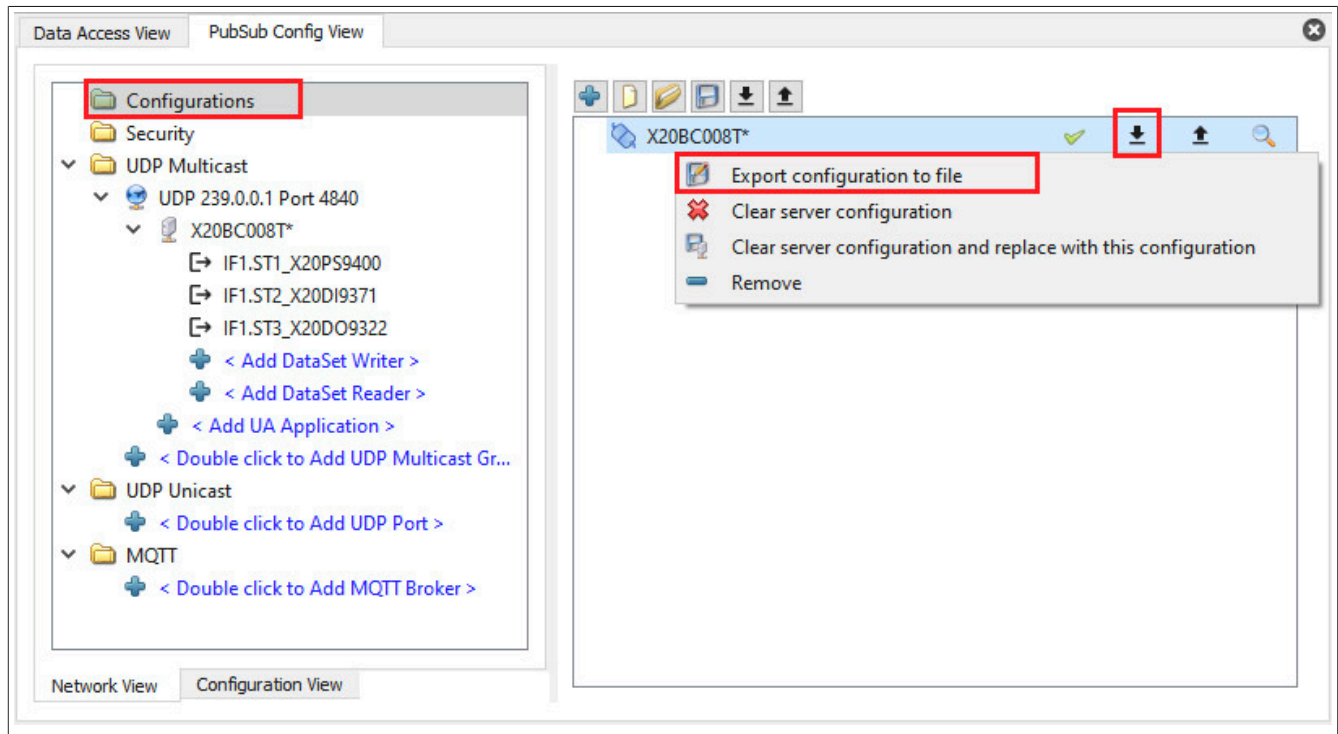
- Select folder Configurations in PubSub Config View and click on the download arrow.

In addition, a local backup copy should be saved by right-clicking on the bus controller - Export configuration to file.



Information:

The backup copy is not only used for backup, but also as a reference file for the subscriber configuration of another server and can be imported into Automation Studio, for example.

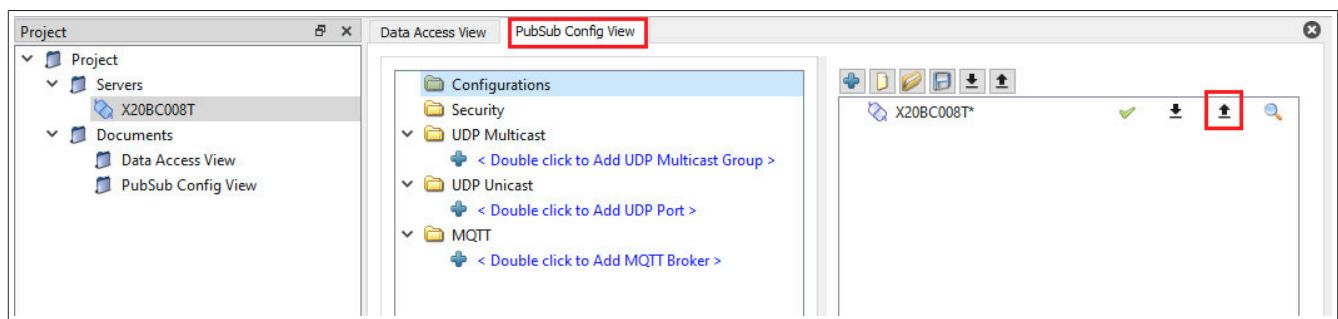


The bus controller sends publisher datagrams periodically after the configuration has been transferred.

3.9.4.2 Configuring the bus controller as a subscriber with UaExpert

Referencing a publisher for the subscriber configuration

- If a configuration already exists on the bus controller, it must be loaded beforehand; otherwise, it will be overwritten empty during the next configuration transfer. To do this, select folder Configurations in PubSub Config View and click on the upload arrow.

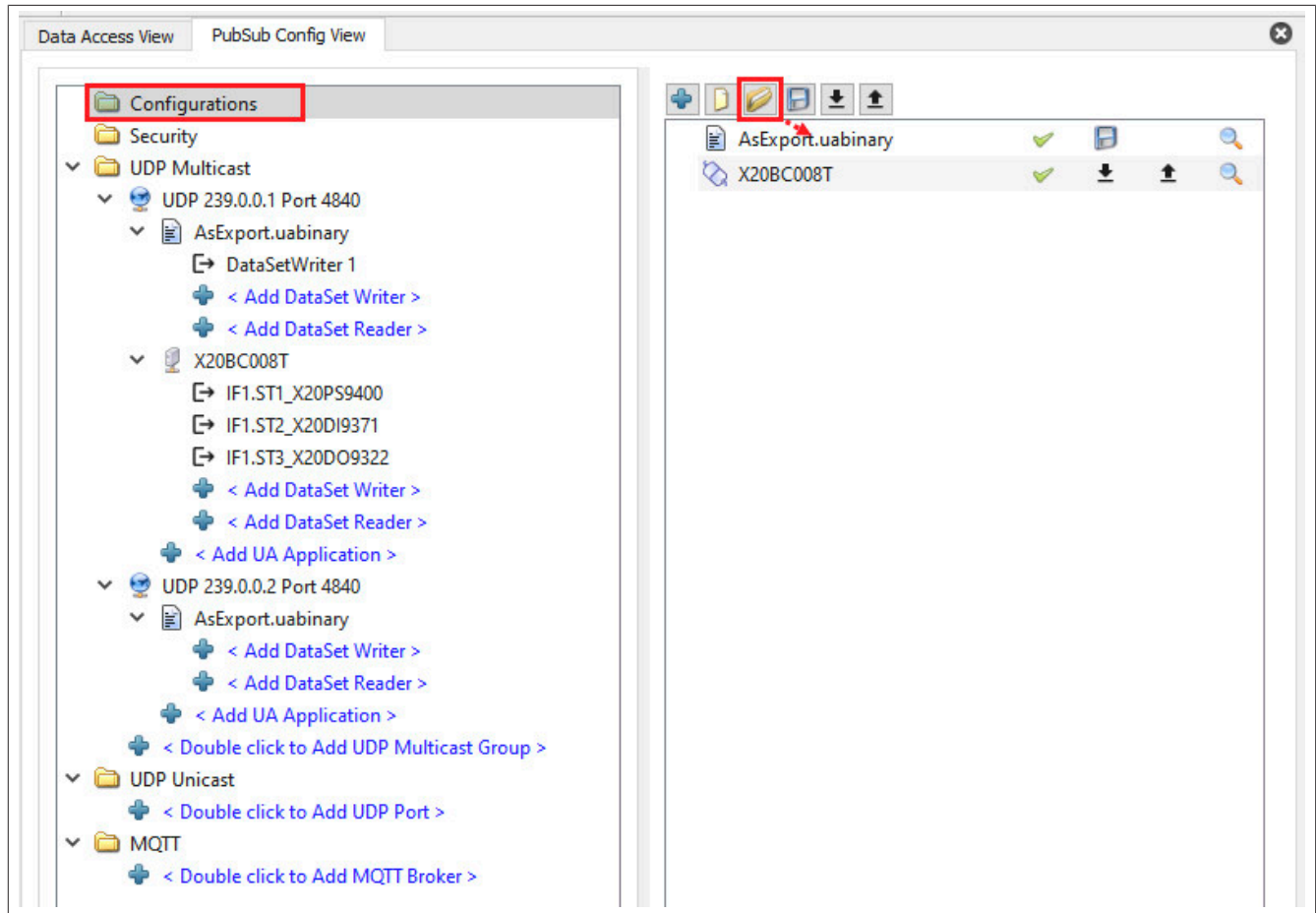


- Select folder Configurations folder to load the configuration. Select the local ".uabinary" file or load the configuration from another server by clicking on the upload arrow.



Information:

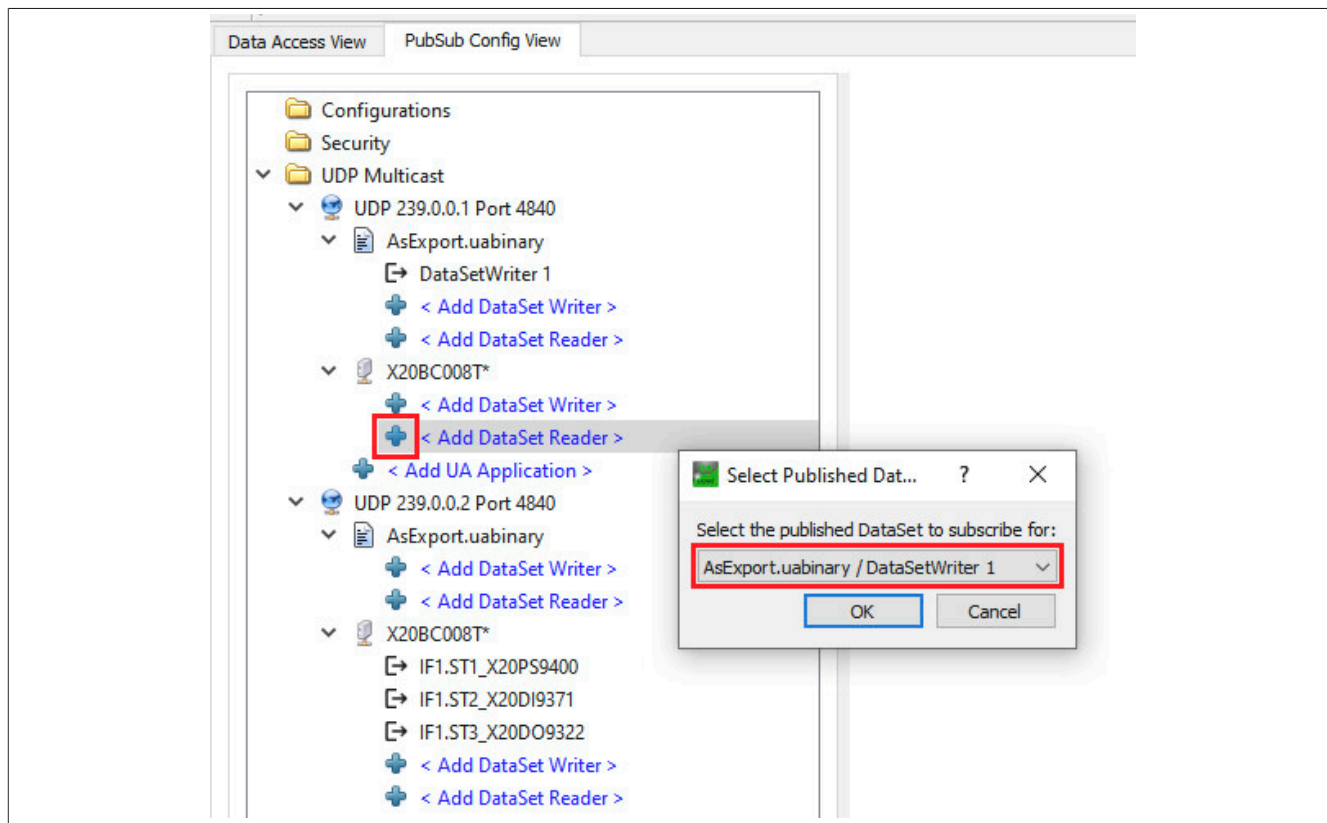
For details about creating a ".uabinary" file, see "Communication → OPC UA FX → PubSub real-time communication between two TSN-capable network subscribers → Getting started → Generating the publisher configuration" in Automation Help.



Getting started

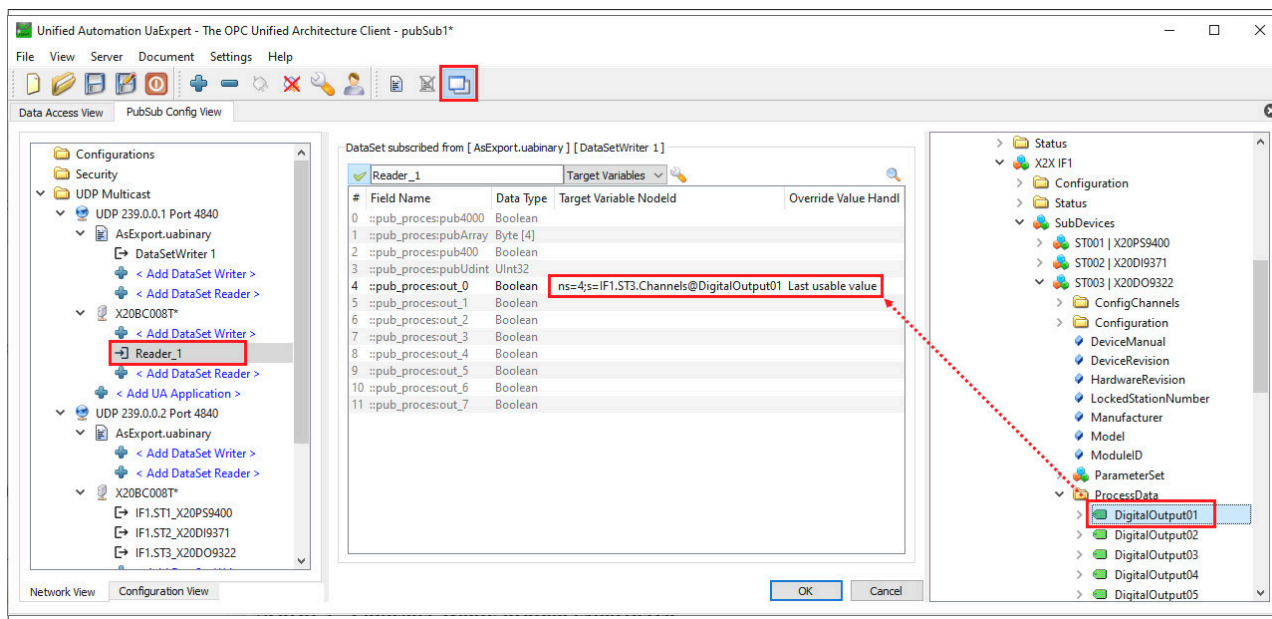
- Create a new DataSetReader for the bus controller connection by clicking "+" and select a "DataSetWriter" as a reference in dialog box "Select published DataSet".

If the publisher whose data should be imported is located in a different connection, i.e. with a different IP address, the bus controller must be added as a UA application to the respective connection.



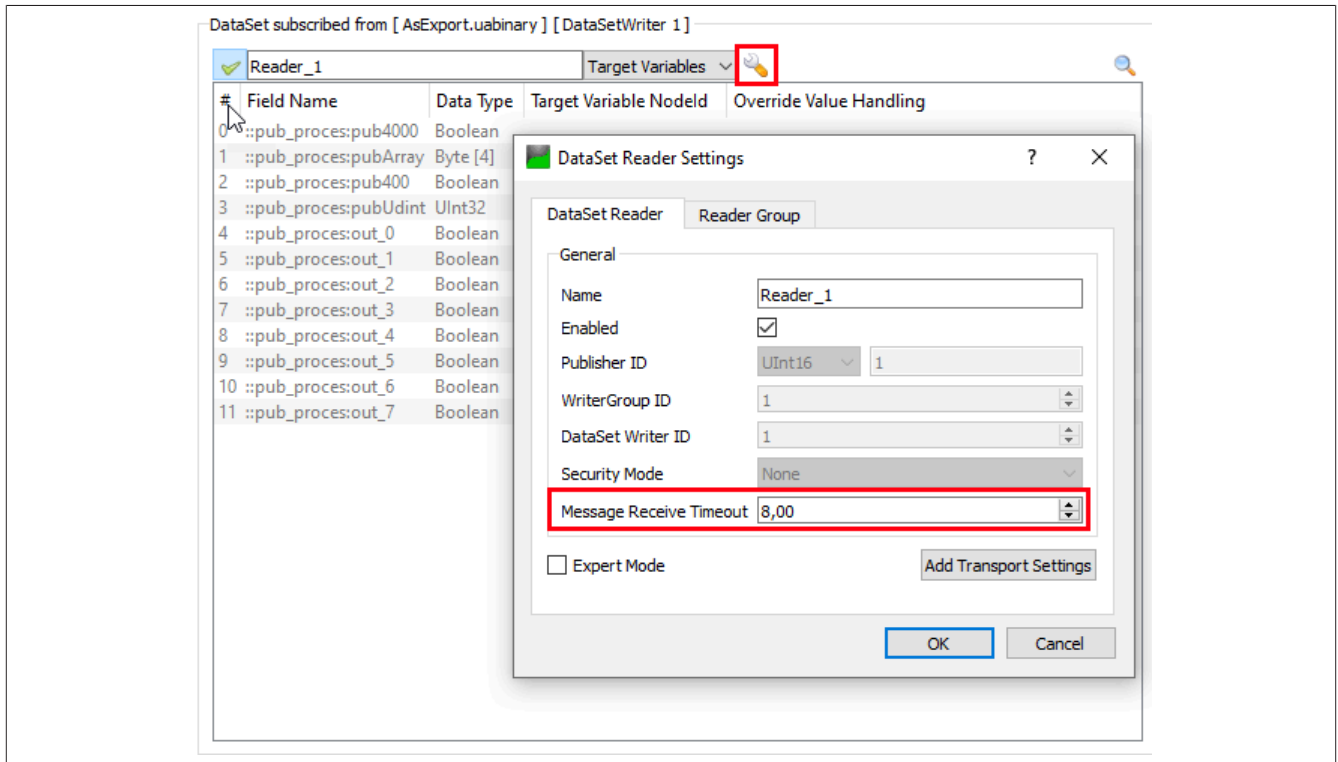
- Perform the following steps to link the module data points to publisher data:

- 1) Change the view by clicking on "Hide all dock widgets".
- 2) Select DataSetReader. This adds the "Address space view" of the associated server to the view.
- 3) Navigate to the node that should be linked to the publisher data, e.g. Root/Objects/DeviceSet/X20BC008T/X2X IF1/SubDevices/ST003/X20DO9322/ProcessData/DigitalOutput02 and drag the node to the desired target variable NodeId. Only one data point can be assigned in this example. If necessary, assign further nodes one after the other.



- Message receive timeout

The message receive timeout can be set for each DataSet Reader in the respective configuration. This is important in the event that the network connection or respective publisher is lost. All outputs are reset after the message receive timeout has expired.



Getting started

- VLAN parameters (optional)

These parameters are defined in field Address network interface in dialog box "Connection settings". They are used in the subscriber context to bind a subscriber to a TSN stream as a listener or to receive a subscriber datagram with VLAN tag.

The following VLAN parameters can be used:

- VLAN ID: `"/VID=<VLAN ID>"`
- Priority code point (PCP): `"/VP=<PCP>"`
- Source IP address: `"/VIP=<Source IP address>"`



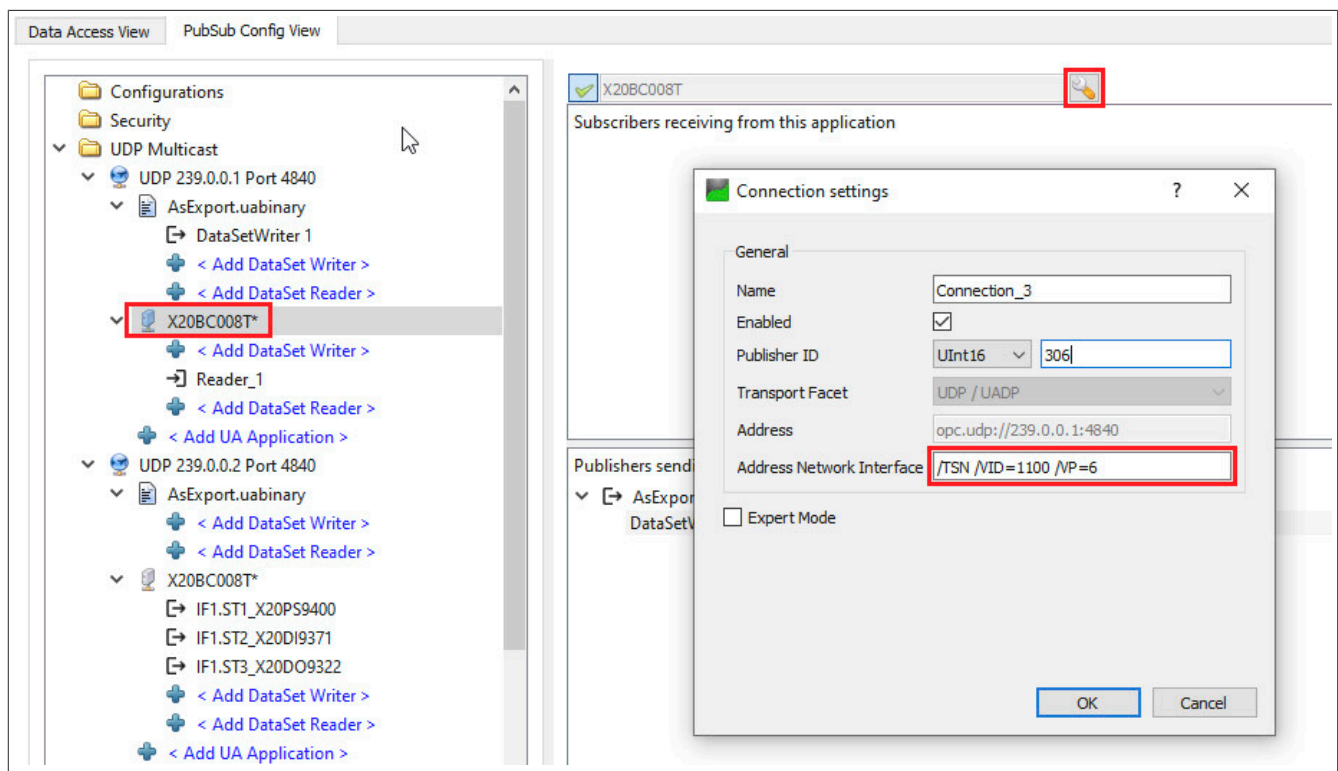
Information:

Parameters `/VP` and `/VIP` are only applied to a possibly included publisher.



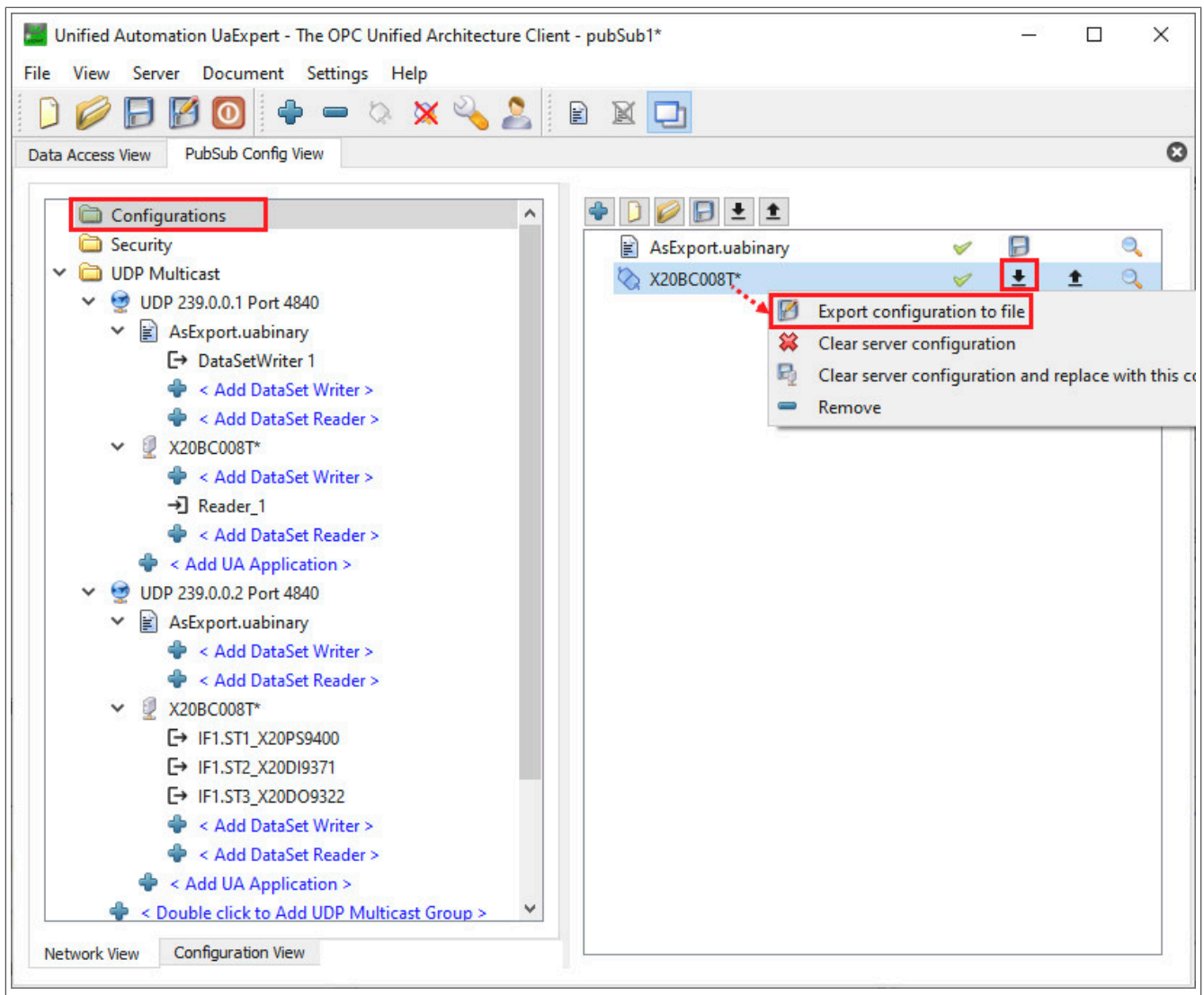
Information:

TSN parameters `/TSN /VID` and `/VP` must be specified in order to use the subscriber in real time.



Transferring the configuration to the bus controller

- Select folder Configurations in PubSub Config View and click on the download arrow.
- In addition, save a local backup copy by right-clicking on the bus controller / "Export configuration to file".



Changes to the publisher data should now also affect the linked X2X modules.

Getting started

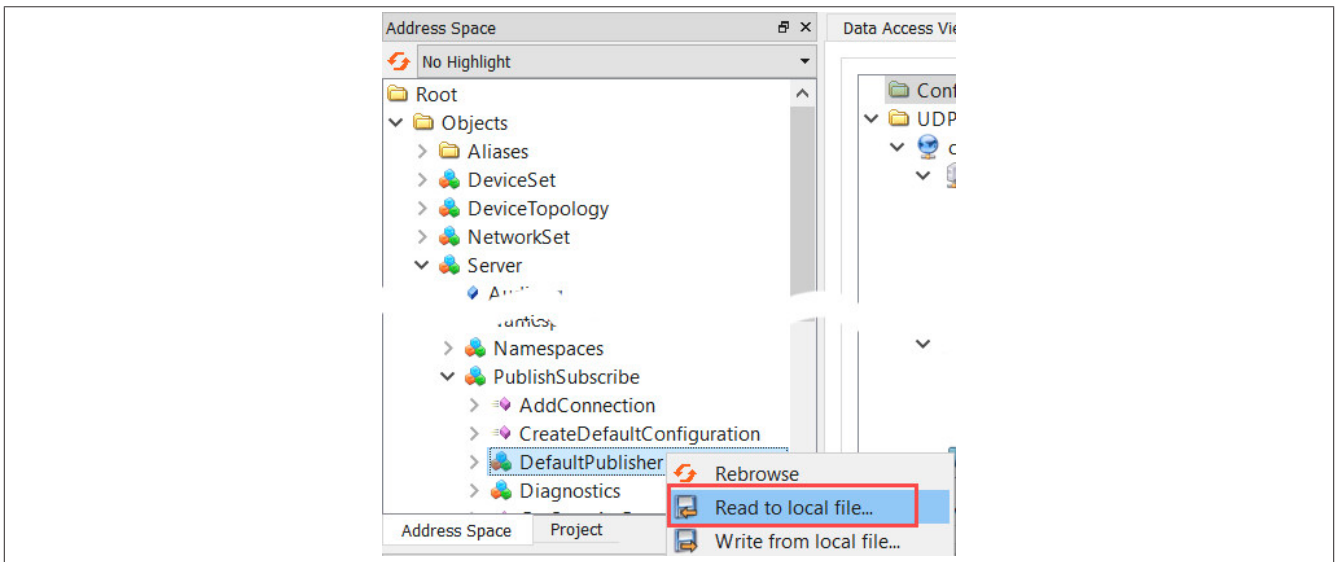
3.9.4.3 Configuring communication between the bus controller and B&R controller

3.9.4.3.1 Process variable image in Automation Studio

An online connection between the bus controller and controller as well as an already created Automation Studio project are required for this example.

To obtain an image of the module data points that can be communicated with, variables with corresponding data types can be generated for Automation Studio on the bus controller. After successfully configuring the modules and executing method `CreateDefaultConfiguration`, reading can take place via object `Root/Objects/Server/PublishSubscribe/DefaultPublisher`. This contains a suggestion for variables in Automation Studio that can be linked to Automation Studio via PubSub.

- Read object `DefaultPublisher` and save it as a ZIP file by right-clicking on Read to local file.



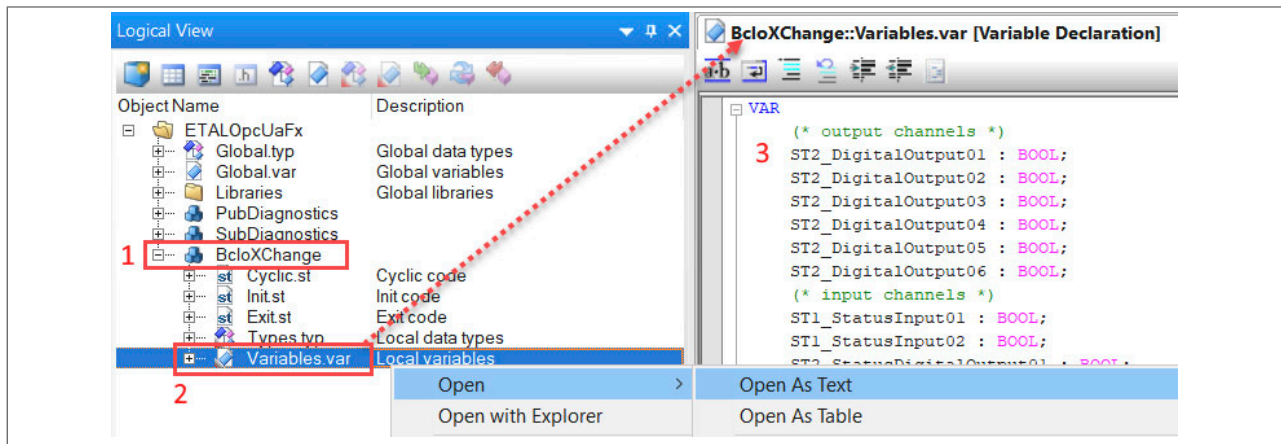
- Open the saved ZIP file and open file "default.var" it contains with a text editor.

Example of file "default.var"

```
VAR
  (* output channels *)
  ST2_DigitalOutput01 : BOOL;
  ST2_DigitalOutput02 : BOOL;
  ST2_DigitalOutput03 : BOOL;
  ST2_DigitalOutput04 : BOOL;
  ST2_DigitalOutput05 : BOOL;
  ST2_DigitalOutput06 : BOOL;
  (* input channels *)
  ST1_StatusInput01 : BOOL;
  ST1_StatusInput02 : BOOL;
  ST2_StatusDigitalOutput01 : BOOL;
  ST2_StatusDigitalOutput02 : BOOL;
  ST2_StatusDigitalOutput03 : BOOL;
  ST2_StatusDigitalOutput04 : BOOL;
  ST2_StatusDigitalOutput05 : BOOL;
  ST2_StatusDigitalOutput06 : BOOL;
  ST3_AnalogInput01 : INT;
  ST3_AnalogInput02 : INT;
  ST3_StatusInput01 : BYTE;
END_VAR
```

• Perform the following steps to add the contents of file "default.var" in Automation Studio:

- 1) Create a program in Automation Studio. (e.g. BcloXchange)
- 2) Open the variable declaration by right-clicking Open → Open as text.
- 3) Copy the content of file "default.var" opened in the text editor, paste it in the variable declaration and save it.



Information:

For details about making variables available via OPC UA, see "Communication → OPC UA FX → PubSub real-time communication between two TSN-capable network subscribers → Getting started → Configuring the OPC UA client/server and default view" in Automation Help.

Getting started

3.9.4.3.2 Bus controller transmitting to the controller

3.9.4.3.2.1 UaExpert configuration for the bus controller

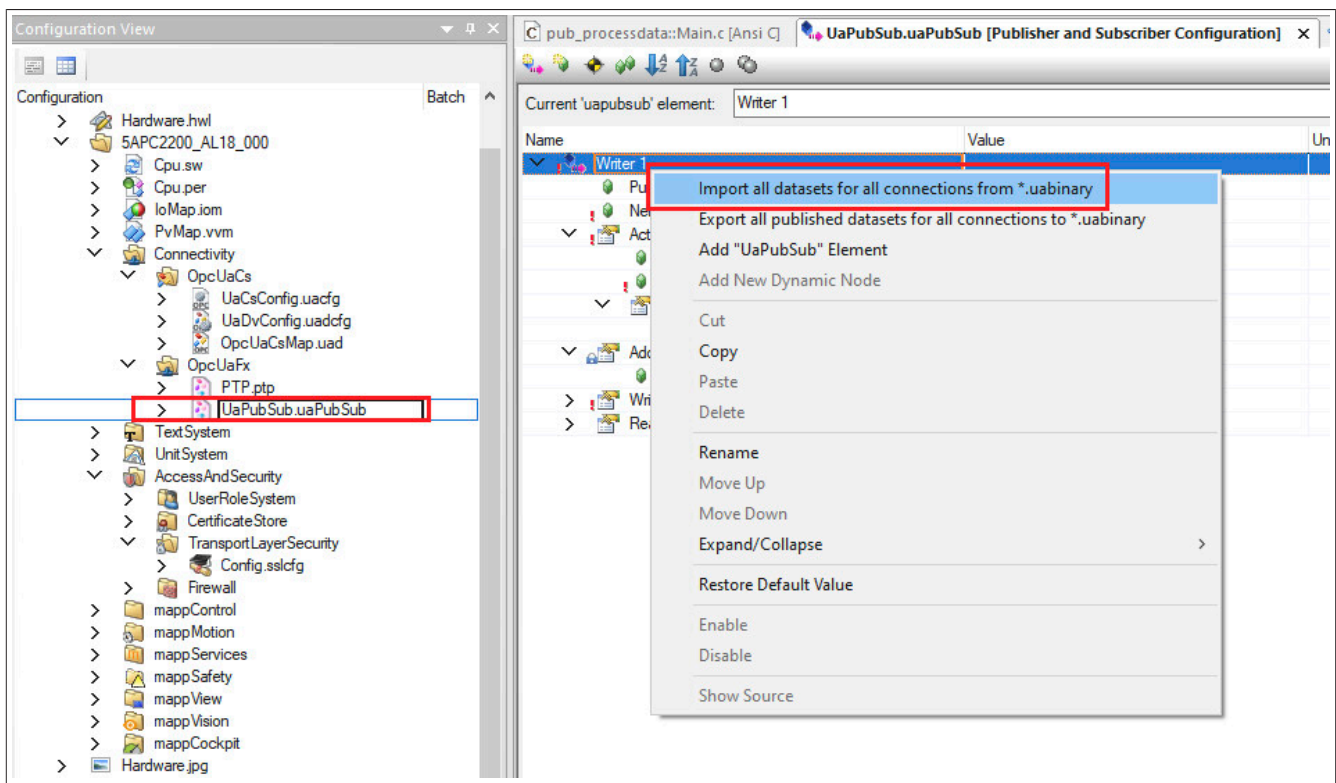
Configure the bus controller publisher and save the configuration file (.uabinary) (see [3.9.4.1 "Configuring the bus controller as a publisher with UaExpert"](#)).

3.9.4.3.2.2 Automation Studio configuration for the controller

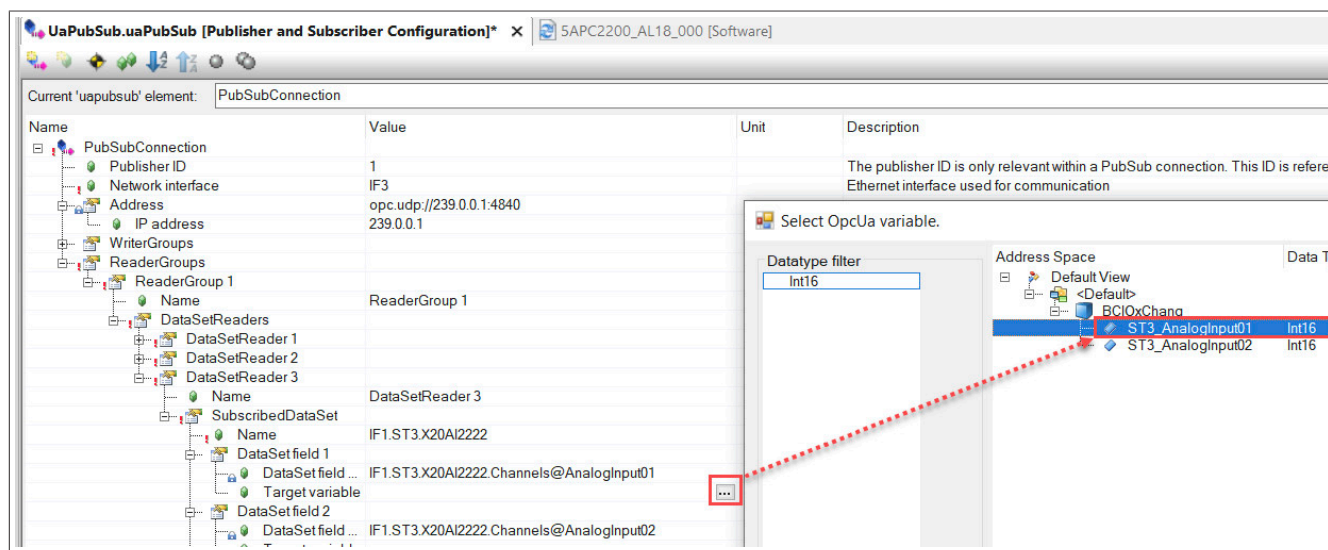
Make variables accessible via OPC UA and create the PubSub configuration (see [3.9.4.3.1 "Process variable image in Automation Studio"](#)).

Configuring the subscriber

- After right-clicking on the existing WriterGroup or a free area in the window, the saved PubSub configuration file of the bus controller can be selected using "Import all datasets for all connections *.uabinary" and imported into Automation Studio.



- After double-clicking on DataSet field, select the desired variable in the dialog box. After confirming the selection, the variable is added to PubSubConnection. Repeat this step for additional variables.

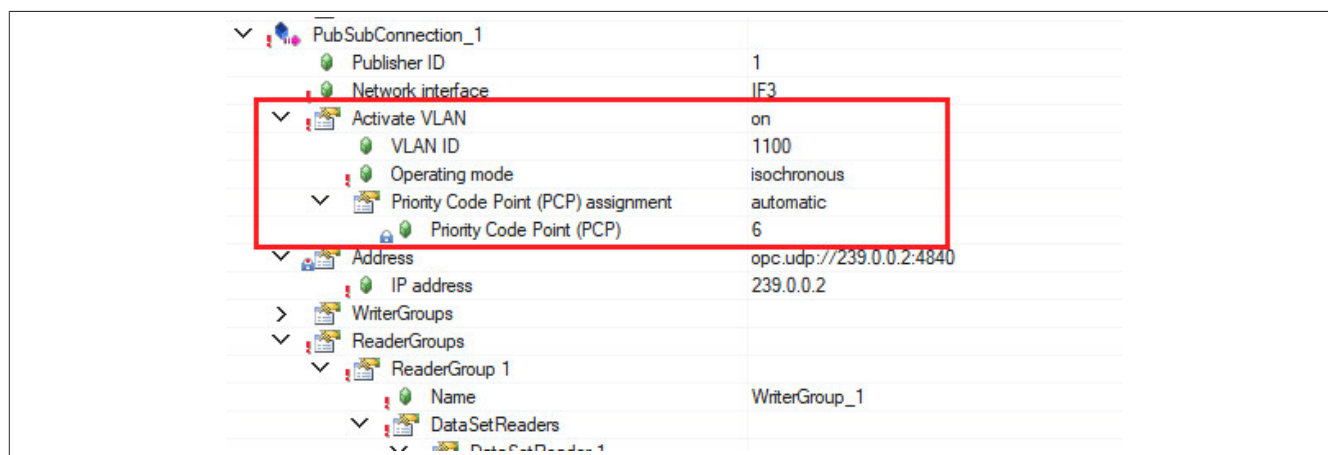


- Adjust the subscriber IP address to publisher. For the publisher IP address, see ["Adjusting the parameters" on page 26](#).

- Adjust VLAN (optional).

The VLAN parameters are not part of the imported ".uabinary" file configuration and can therefore be modified afterwards.

- 1) Select "On" for parameter Activate VLAN.
- 2) Adjust the VLAN parameters.



Information:

Parameter Priority code point **is applied only to contained publishers.**



Information:

The VLAN configuration must always be carried out if real-time communication is required.

- Save the configuration and transfer it to the controller.

Getting started

3.9.4.3.2.3 Check the communication.

Modification of the data on the bus controller should now also be visible on the corresponding variable value of the controller. The variable values can be made visible in Automation Studio under "Logical View → Task (right-click) → Open → Watch".

3.9.4.3.3 Controller transmitting to the bus controller

3.9.4.3.3.1 Automation Studio configuration for the controller

Make variables accessible via OPC UA and create the PubSub configuration (see [3.9.4.3.1 "Process variable image in Automation Studio"](#)).

For information about adding the PublisherConnection, see [OPC UA FX](#)"Communication → OPC UA FX" in Automation Help.

4 Firmware update via OPC UA

The firmware update function can be used to update the firmware of the bus controller to any version via OPC UA. During this, it is ensured that firmware capable of communication is always loaded even in the event of a power failure or interruption of the transfer.

The update mechanism is based on specification "OPC 10000-100 - UA Specification Part 100 - Devices 1.03.0" and uses option "Cached-Loading", where the firmware file is first loaded to the server and installed in a second step. As a last step, the installed firmware must be enabled.

A UaExpert client that supports the following OPC UA types is required to perform a firmware update:

- FileType
- TemporaryFileTransferType
- OptionSet

For detailed information about how to perform the firmware update with UaExpert, see section [4.1 "Performing an update"](#).

For a detailed description of the structure of the firmware update object, see [11.1.2 "Firmware update"](#).

4.1 Performing an update

All required methods and status information for the firmware update are located under Root/Objects/DeviceSet/X20BC008T/FirmwareUpdate. In addition, method "Reboot" under Root/Objects/DeviceSet/X20BC008T/Configuration/Control/Reboot is needed.

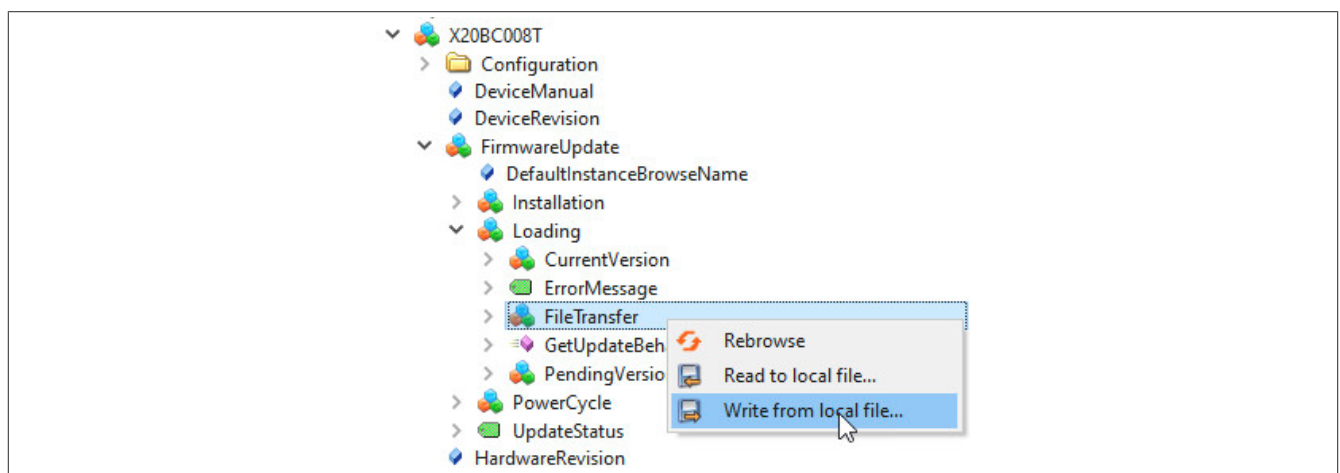
A firmware update can easily be performed with UaExpert. The following steps are required for this:

• Preparation

Download the desired firmware update file from the [B&R website \(https://www.br-automation.com\)](https://www.br-automation.com) and unpack it.

• Transfer

After a connection to the bus controller has been established, right-click on Write from local file in object Root/Objects/DeviceSet/X20BC008T/FirmwareUpdate/Loading/FileTransfer and select the unpacked firmware update file (.fw).



• The selected file is transferred to the bus controller by UaExpert. The file to be installed can be checked by calling method Root/Objects/DeviceSet/X20BC008T/FirmwareUpdate/Loading/PendingVersion/SoftwareRevision. This must match the file that has just been transferred and can be determined via the last part of the filename:

Firmware update via OPC UA

- <Order number>*V<SoftwareRevision>.zip

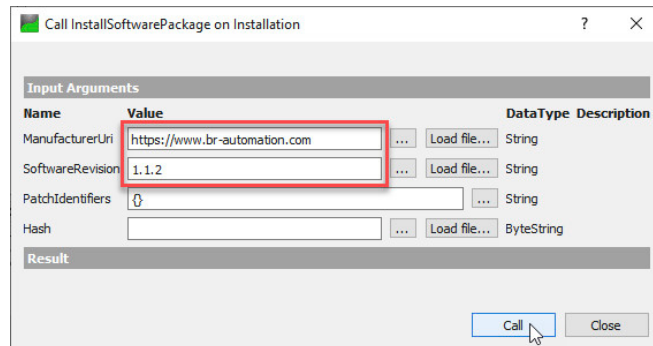
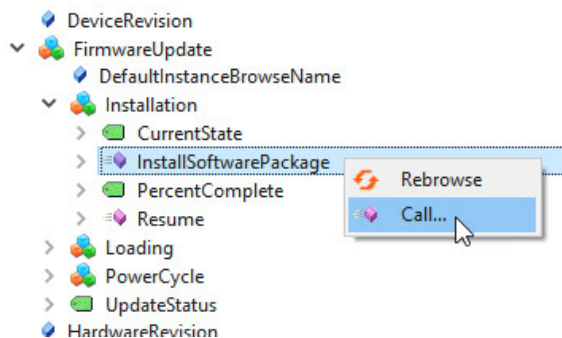
Example

X20BC008T_FIRMWARE_V1.0.0.zip → corresponds to SoftwareRevision = 1.0.0.

• Installation

Right-click on Call in object Root/Objects/DeviceSet/X20BC008T/FirmwareUpdate/Installation/InstallSoftwarePackage and enter the required parameters. These include:

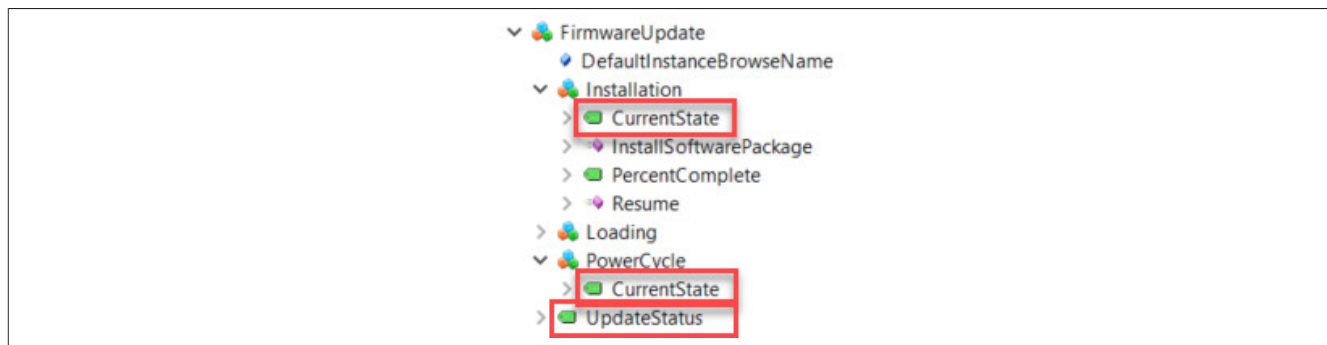
- ManufacturerURI: "https://www.br-automation.com"
- SoftwareRevision: According to the example above



Complete the installation by clicking Call and wait until the installation is completed. The status of a successful installation can be checked using the following parameters:

- Parameter Root/Objects/DeviceSet/X20BC008T/FirmwareUpdate/Installation/CurrentState shows "Installing".
- Parameter Root/Objects/DeviceSet/X20BC008T/FirmwareUpdate/PowerCycle/CurrentState shows "WaitingForPowerCycle".

Both parameters must display the described value. Alternatively, parameter Root/Objects/DeviceSet/X20BC008T/FirmwareUpdate/UpdateStatus can be evaluated. This should contain value "[INFO] Installation successful, reboot required".



Information:

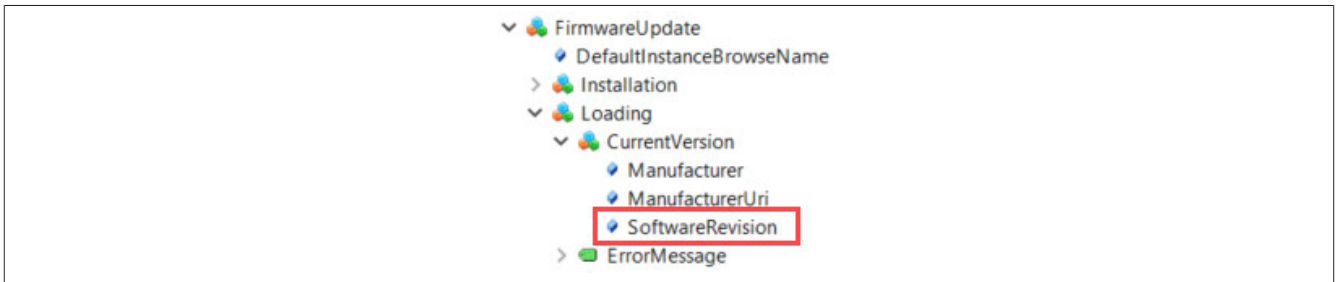
Firmware installation can take up to one minute.

The subsequent restart is only permitted to be performed when parameter Root/Objects/DeviceSet/X20BC008T/PowerCycle/CurrentState indicates status "WaitingForPowerCycle". Otherwise, the firmware update is aborted and the device reboots with the old version.

• Restart and check

Perform a restart. This can be done by calling method `Root/Objects/DeviceSet/X20BC008T/Configuration/Control/Reboot` (see 6.1.1 "Methods for bus controller configuration") or by switching the power supply off and on.

Whether the firmware updates were enabled successfully can be checked after a restart. This is done by reading node `Root/Objects/DeviceSet/X20BC008T/Loading/CurrentVersion/SoftwareRevision`. The firmware version displayed must be identical to the software revision of the firmware update ZIP file.



• Error handling

If a serious error occurs during the firmware update, it must be reset since no further firmware update is possible in the error state. This can be done in the following ways:

- Acknowledging the error using method `Root/Objects/DeviceSet/X20BC008T/FirmwareUpdate/Installation/Resume`
- Resetting the error state with a restart. This will reload the original firmware.

The failed firmware update is properly aborted and ended by either method.

5 Features/Functionality

5.1 Supported modules



Information:

The modules listed only apply to the bus controller starting with version 1.4.6. Customized or already discontinued modules were not taken into account.

The bus controller supports modules from the following groups:

- X20 modules

Modules not yet supported

- X20BR7300, X20CM4810, X20MM4455, X20DO4F49, X20DOF321, X20DIF372
- All reACTION Technology modules

- X20 coated modules

Modules not yet supported

- X20cAT2311, X20cCM4810, X20cCMR011, X20cDC2395, X20cDO4332-1, X20cMM2436

- X67 modules

Modules not yet supported

- X67DS838A.L12, X67DS838B.L12

- All 4XPxxx keypad modules
- All 80SDxxx ACOPOSmicro stepper motor modules
- ACOPOSmicro power supply module 80PS080X3.10-01
- Motor starter module SFM1-A11.1

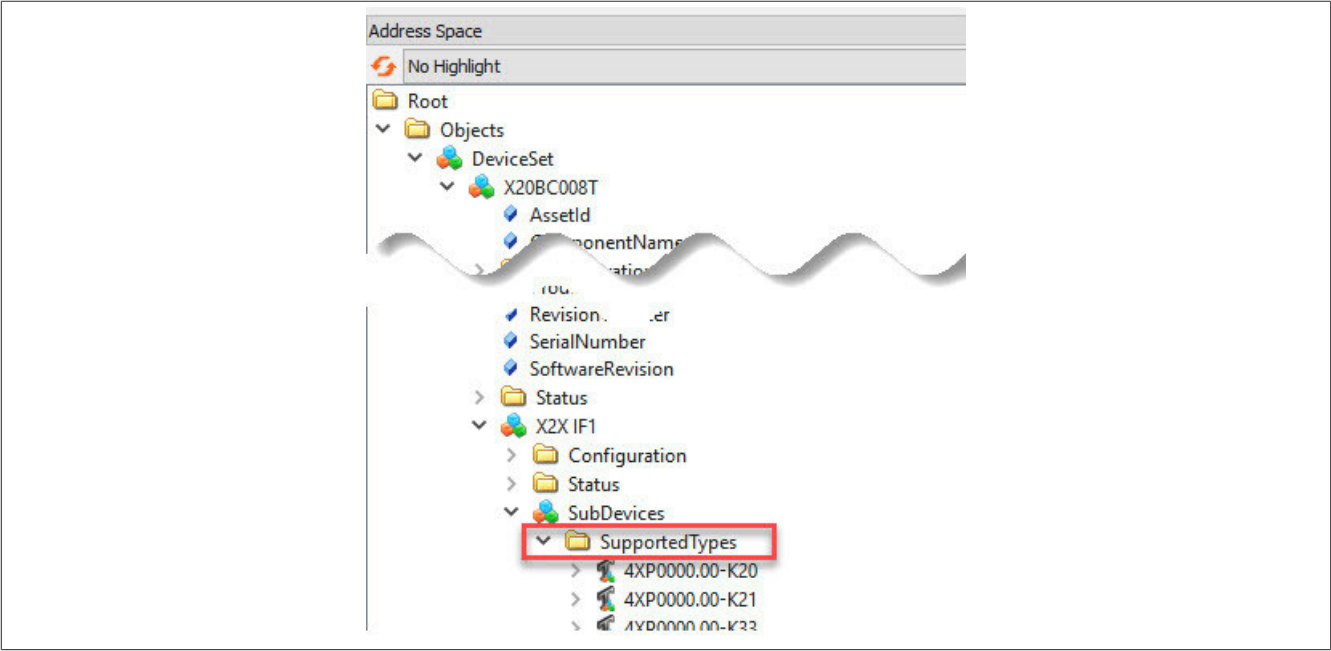


Information:

Safety modules are not supported.

Reading out the module list from the bus controller

The list of supported modules is constantly being expanded. The exact list can be read out from the bus controller under Root/Objects/DeviceSet/X20BC008T/X2X IF1/SubDevices/SupportedTypes.



5.2 Namespaces used

The following namespaces are used in the bus controller:

Index	Namespace URL	Description
0	http://opcfoundation.org/UA/	Address space for types and objects defined in the OPC UA specification
1	http://br-automation.com/OpcUa/X20BC008T/<Serial number>/	This namespace URL is the address space of the bus controller on which the OPC UA server is running. <Serial number> corresponds to the serial number of the bus controller.
2	http://opcfoundation.org/UA/DI/	Address space for types and objects defined in the OPC UA companion specification for device integration (DI).
3	http://br-automation.com/OpcUa/BrDevice	Basic information model for B&R field devices
4	http://br-automation.com/OpcUa/io-system	Information model of the bus controller

The namespaces used can also be read out from the OPC UA information model:

Root

Objects

Aliases

DeviceSet

DeviceTopology

NetworkSet

Server

Auditing

Dictionaries

EstimatedReturnTime

GetMonitoredItems

LocalTime

NamespaceArray

Namespaces

PublishSubscribe

UserRolePermissions

RolePermissionType Array[2]

AccessRestrictions

BadAttributeIdInvalid (0x80350000)

Value

SourceTimestamp

12.07.2021 15:19:05.387

SourcePicoSeconds

0

ServerTimestamp

12.07.2021 15:19:05.387

ServerPicoSeconds

0

StatusCode

Good (0x00000000)

Value

String Array[5]

[0]

http://opcfoundation.org/UA/

[1]

http://br-automation.com/OpcUa/X20BC008T/F6290168454/

[2]

http://opcfoundation.org/UA/DI/

[3]

http://br-automation.com/OpcUa/BrDevice

[4]

http://br-automation.com/OpcUa/BC/io-system/

DataType

String

NamespaceIndex

0

IdentifierType

Numeric

Identifier

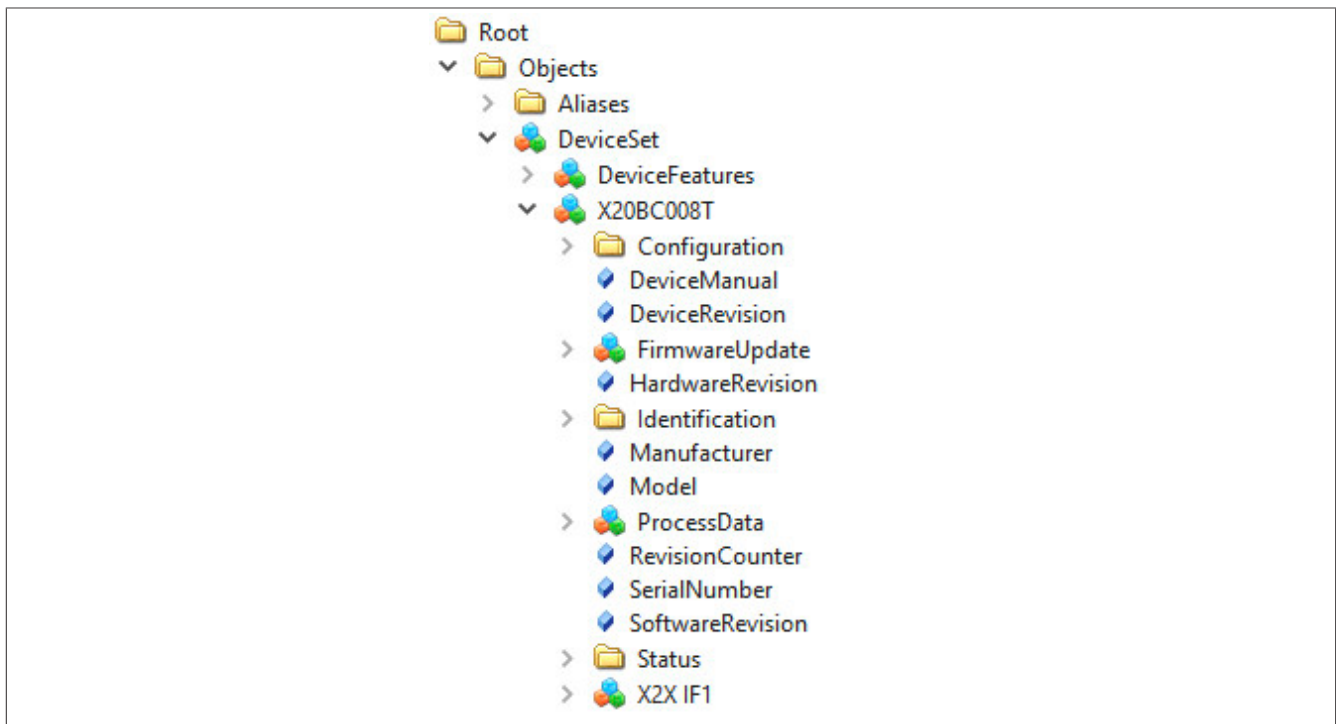
12 [String]

ValueRank

1 (OneDimension)

5.3 Device information

Node Root/Objects/DeviceSet/X20BC008T contains additional nodes that can be used to read out basic information about the bus controller:



Node name	Description
DeviceManual	URL providing additional information about the module
DeviceRevision or ProcessData/HardwareVariant	B&R hardware variant
HardwareRevision	Hardware revision of the bus controller
Manufacturer	Manufacturer of the bus controller
Model	Module ID
RevisionCounter	Reserved (always -1)
SerialNumber	Full serial number as a string
SoftwareRevision	Current software revision
Identification/ModuleID or ProcessData/ModuleID	Numeric identification number of the module
ProcessData/SerialNumber	Serial number as a 32-bit integer

5.4 Time synchronization and time domains

The bus controller has 2 independent clocks that can be synchronized with different time domains. This ensures uniform time behavior of all network devices synchronized to the same time domain. Both the time values and the frequencies of the clocks are thus coordinated with each other. This makes it possible to precisely coordinate activities on various devices and include their timestamps in an exact time sequence.

Either the Network Time Protocol (NTP) or the IEEE 802.1AS-2020 profile of the Precision Time Protocol (PTP) can be used for time synchronization on the bus controller.

The two clocks can be used as follows:

WallClock

WallClock corresponds to the classic system clock. It can be synchronized to the current UTC time via NTP or PTP and used for logging timestamps or certificate validation, for example.

WorkingClock

WorkingClock is independent of UTC time and is mainly used to send TSN Ethernet frames in a time-accurate manner. Compared to WallClock, WorkingClock ensures that there are no further jumps as time progresses after the first synchronization procedure (e.g. due to leap seconds as they occur with UTC time). To ensure that TSN Ethernet frames are sent with sufficient accuracy, the requirements on synchronization accuracy are much higher for WorkingClock. Only PTP is therefore available as a synchronization mode for WorkingClock.



Information:

It is important to note that PTP time synchronization must be enabled on the bus controller for each PTP time domain used by connected network devices. This is also necessary if the bus controller does not use the corresponding domain itself.

5.5 Network management protocols

For advanced network management, the bus controller supports additional protocols that are automatically enabled during startup.

5.5.1 Multiple Spanning Tree Protocol (MSTP)

This protocol is used to logically disconnect redundant connections on the network so that broadcast frames cannot be transmitted multiple times in a circle, which would place an unnecessary load on the network. A logical tree topology that prevents Ethernet frames from being forwarded on redundant paths is created by exchanging configuration messages. In the case of MSTP, a separate logical tree is built for all existing virtual local area networks (VLANs).

Since logical tree topologies can change dynamically on the network, they are not suitable for time-controlled communication. It is therefore necessary to define network paths for time-controlled communication explicitly via a TSN configuration. This is done by configuring forwarding rules. These defined paths are not affected by MSTP.

For detailed information about MSTP, see the IEEE 802.1Q standard. Configuration of the MSTP stack or status queries can be carried out via NETCONF. For the configuration parameters and status values, see the corresponding YANG model.

5.5.2 Link Layer Discovery Protocol (LLDP)

LLDP is used to exchange information about identity and supported functionality between neighboring network devices. This information is collected individually for each port to which an LLDP-capable device is connected.

For detailed information about LLDP, see the IEEE 802.1AB standard. The status can be queried via NETCONF using the associated YANG model. For diagnostic purposes, some of the status values are also stored in the OPC UA information model (see also [7.1 "Port status"](#)).

5.6 Integration in the IT network

When the bus controller starts up, the Multiple Spanning Tree Protocol (MSTP) stack is automatically started (see 5.5.1 "Multiple Spanning Tree Protocol (MSTP)"). When this stack is executed, configuration messages are exchanged between network devices, which are transferred using Bridge Protocol Data Unit (BPDU) frames.

This configuration data can influence the logical topology of a network, which can result in undesired network behavior, especially in the event of error or deliberate manipulation. To prevent such problems, many commercially available switches support BPDU filters, which can be applied to individual ports. These filters can either be used to discard BPDU packets or to block the port to which the originating device is connected.

When integrating the bus controller into an IT network, the specifications of IT administration must be observed to prevent disruption (e.g. automatic exclusion of the port to which the bus controller is connected). If no BPDU packets are permitted in the IT network, a BPDU filter should be set on the IT switch that serves as access to the IT network for the bus controller in order to prevent corresponding packets from being forwarded.

5.7 Integration in the TSN network

Up to software version 1.4.x, integration of the bus controller into a TSN network is only supported as an end node (endpoint), i.e. only 1 Ethernet port is permitted to be used. In particular, forwarding datagrams with a VLAN tag via both Ethernet ports is not supported. The bus controller should therefore be connected directly to a TSN-capable switch.

5.8 Device properties

The ports of the bus controller, their names and TSN capability are listed in the following table.

Name in the OPC UA information model	Internal ID (LLDP)	TSN-capable
ETH1	swOp2	Yes
ETH2	swOp3	Yes



Information:

Internal names swOp1 and swOp2 are used for the internal port or the management port of the bus controller. These have no reference to the external ports of the bus controller.

The following table shows the dimensioning of different properties of the bus controller. If not specified otherwise, the specified values apply globally to the entire bus controller.

Property	Value
Number of filtering database (FDB) entries	512
Maximum number of VLANs	64
Number of queues per port	8
Number of gate control list (GCL) entries per port	255

6 Configuration

Configuration takes place by writing values to corresponding OPC UA variable nodes. These configuration nodes are organized as hierarchical OPC UA objects, each grouping a specific functionality.

Written configuration values are not immediately applied. Changes within the respective object are saved and applied only by calling method [6.1.1.1 "ApplyChanges"](#).

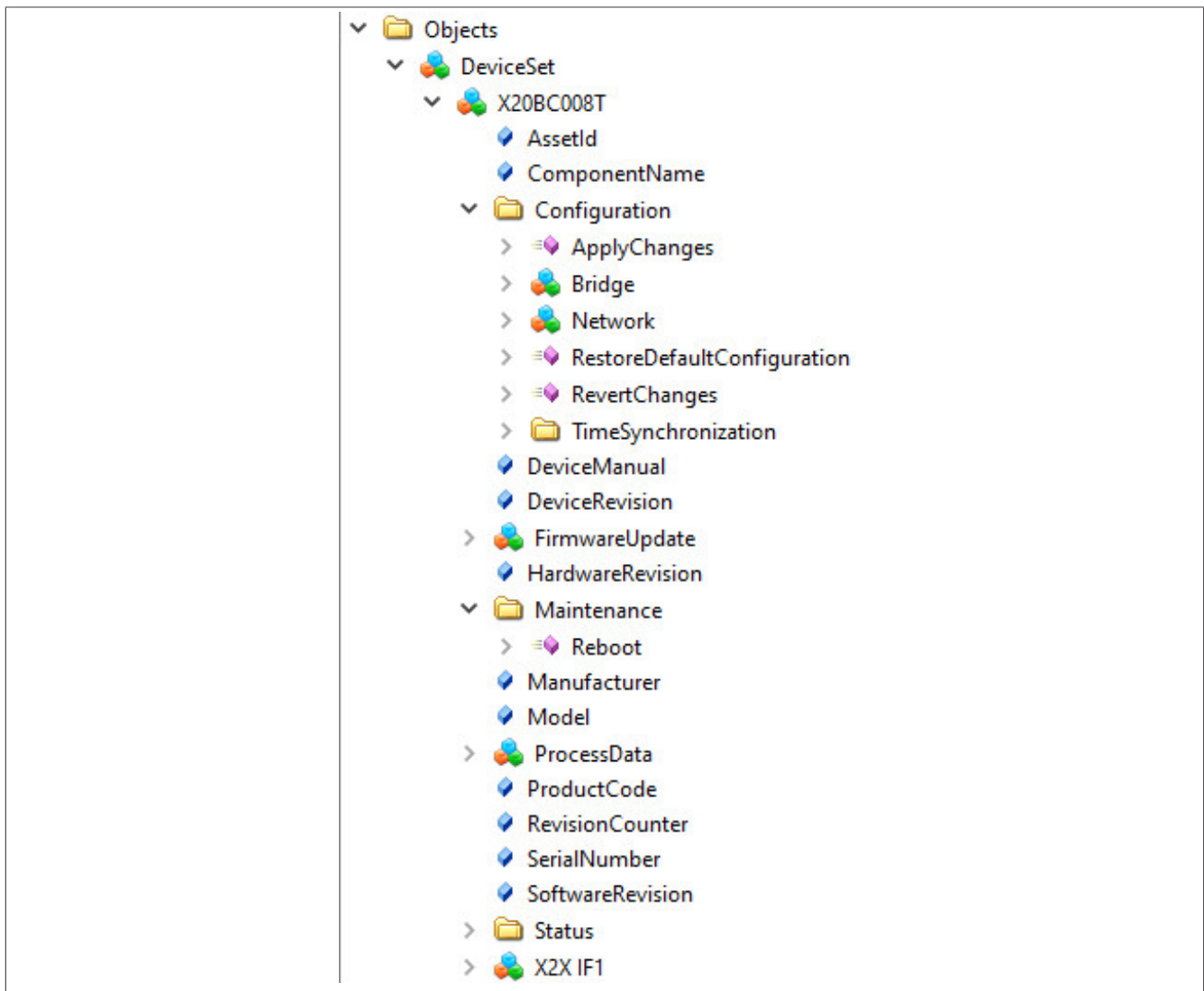


Information:

The structure of the nodes in the information model may be subject to change. It is ensured that the NodeIDs do not change as a result, however. If nodes are addressed automatically, they should therefore be addressed via the NodeIDs and not via the BrowseNames.

6.1 Bus controller object

This object contains all data and methods of the bus controller application.



Information:

Methods [6.1.1.1 "ApplyChanges"](#), [6.1.1.3 "RevertChanges"](#) and [6.1.1.2 "RestoreDefaultConfiguration"](#) refer only to the configuration values within the higher-level "Configuration" folder. Other configurations, such as the X2X configuration or user configuration, are NOT affected by this.

6.1.1 Methods for bus controller configuration

Position of the methods in the information model: Root/Objects/DeviceSet/X20BC008T/Configuration/Control

6.1.1.1 ApplyChanges

Changed values are only saved and applied by calling this method.

6.1.1.2 RestoreDefaultConfiguration

The default configuration values are restored.



Information:

Calling the method only temporarily loads the default configuration values into the nodes. To save or apply the default configuration, method `ApplyChanges` must also be called.

6.1.1.3 RevertChanges

The values previously saved with `ApplyChanges` are restored.

6.1.2 General network configuration

The configuration can be made via the OPC UA information model. For the corresponding parameters, see node Root/Objects/DeviceSet/X20BC008T/Configuration/Network in the model.

Node name	Description
EnableDHCP	Enables or disables the DHCP client functionality. - If an IP assignment by a DHCP server is missing, the bus controller is assigned a random link local address from range 169.254.0.0/16. IPv4LL (RFC3927). - If the DHCP client is enabled, parameters Gateway, IP address, Netmask as well as Primary DNS and Secondary DNS are obtained from the DHCP server.
Gateway	Configures the default gateway IP address. - If parameter EnableDHCP is set, a gateway address can additionally be obtained from the DHCP server. - If parameter Gateway is set, the manual configuration is used and the address obtained from the DHCP server is ignored.
Hostname	Configures the hostname.
IP address	Configures a static IP address. - If parameter EnableDHCP is set, the parameter is ignored and the IP address transmitted by the DHCP servers is used.
Primary DNS Secondary DNS	Configures a primary or secondary DNS server. - If parameter EnableDHCP is set, additional addresses for DNS servers can be obtained from the DHCP server. - If at least 1 DNS server is set manually, the manual configuration is used and the addresses obtained from the DHCP server are ignored.
Netmask	Sets the subnet mask. - If parameter EnableDHCP is set, this parameter is ignored and the subnet mask obtained from the DHCP server is used.
EnableMulticastDNS	Enables or disables multicast DNS (mDNS). - mDNS is enabled in the factory setting in order to be able to address the bus controller via the hostname even if there is no network infrastructure.

Method Root/Objects/DeviceSet/X20BC008T/Configuration/Control/ApplyChanges must be called to save the new configuration data.



Information:

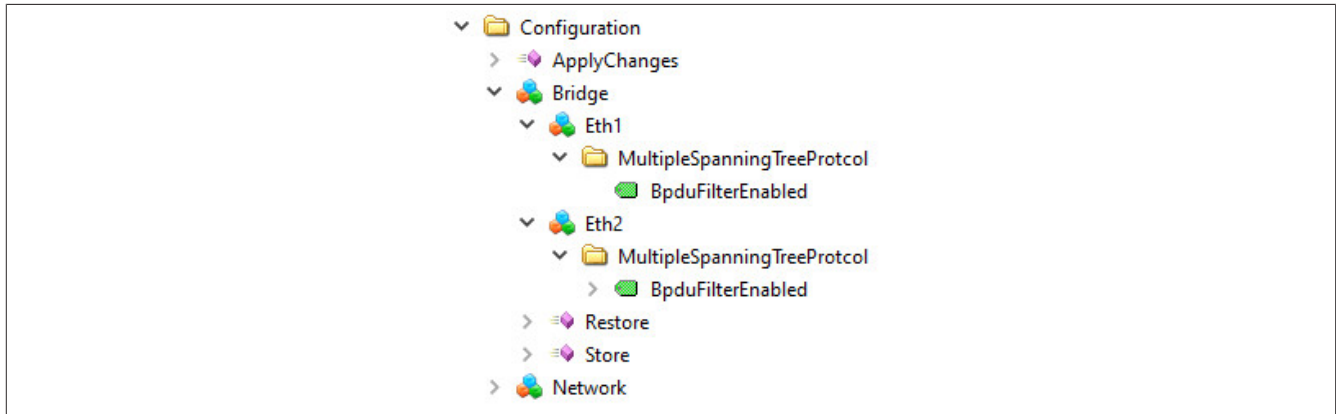
The network configuration is only applied when the bus controller is restarted.

Configuration

6.1.3 Bridge configuration

6.1.3.1 Methods for bridge configuration

2 methods are provided in the OPC UA information model to persist the current bridge configuration or to reset it to the factory settings.



6.1.3.2 BpduFilterEnabled

The [5.5.1 "Multiple Spanning Tree Protocol \(MSTP\)"](#) is active by default on the bus controller. In some networks, however, this is undesirable. The protocol for the respective port can be disabled by enabling the BPDU filter.

6.1.3.3 Restore

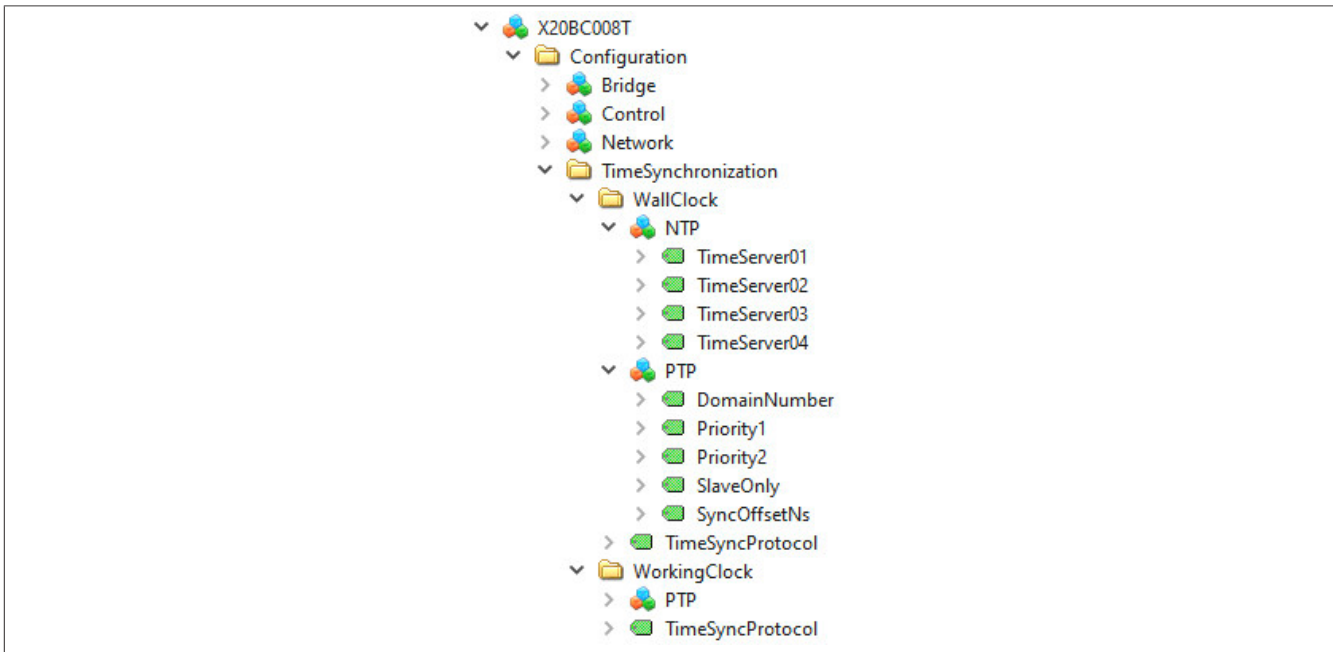
Deletes the current bridge configuration and resets it to the factory settings. The device must be restarted after this method is executed.

6.1.3.4 Store

Persists the current bridge configuration on the device.

6.1.4 Time synchronization

The protocols used for time synchronization must be configured separately for the two time domains WallClock and WorkingClock.



Position of the data in the information model: Root/Objects/DeviceSet/X20BC008T/Configuration/TimeSynchronization



Information:

The parameters for the time synchronization are only applied by calling method `ApplyChanges`.

6.1.4.1 NTP

The NTP client can be enabled for WallClock by setting configuration parameter `TimeSyncProtocol` to value "2 (NTP)". Up to 4 time servers can be specified. Optionally, it is possible that NTP servers are assigned the addresses by the DHCP server. If multiple time servers are available (either by configuration or obtained from the DHCP server), one of them is selected to be used for time synchronization. The others are available as redundancy and are used in case the currently active time server fails.

Node name	Description
TimeServer0x	URL or IP address of up to 4 time servers that can be configured manually. When configuring multiple time servers, the selection order of the time servers is not fixed.

6.1.4.2 PTP



Information:

PTP configuration via OPC UA is not yet supported in version 1.3.1.

6.1.4.3 TimeSyncProtocol

The configuration parameters for NTP or PTP are only valid if the corresponding synchronization protocol has been selected.

Node name	Description
TimeSyncProtocol	This parameter can be used to enable synchronization of the respective clock or select the synchronization protocol. Possible values: 0 No synchronization protocol selected 1 PTP protocol selected 2 NTP protocol selected

Configuration

6.1.5 Reboot

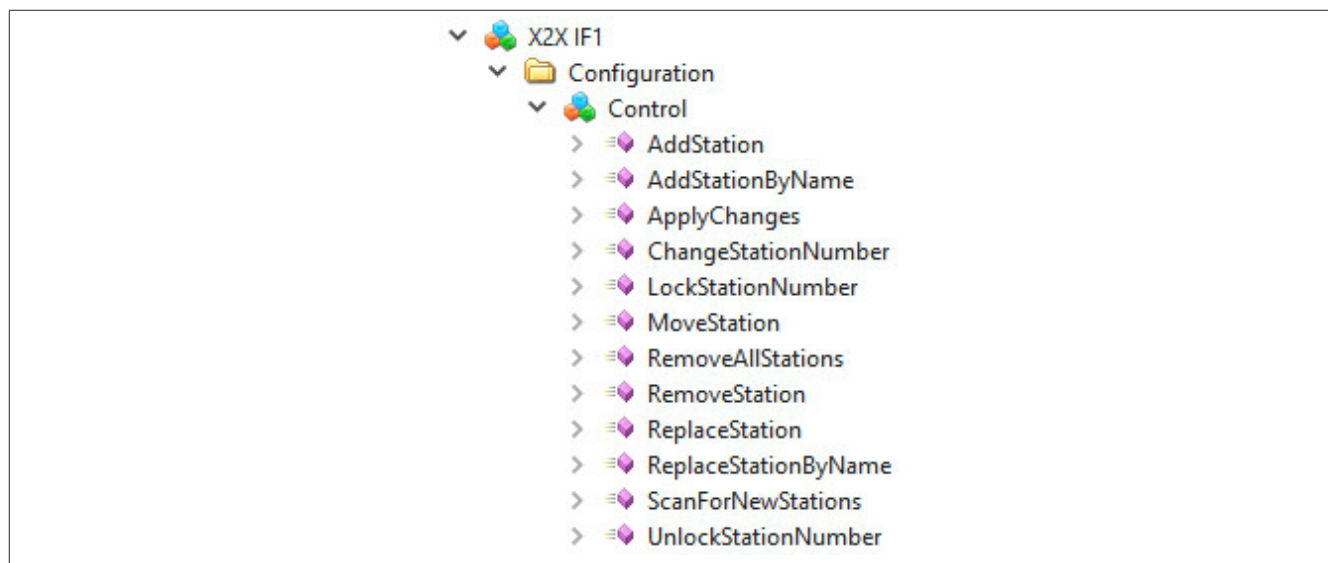
Restarts the bus controller.

6.2 X2X Link object

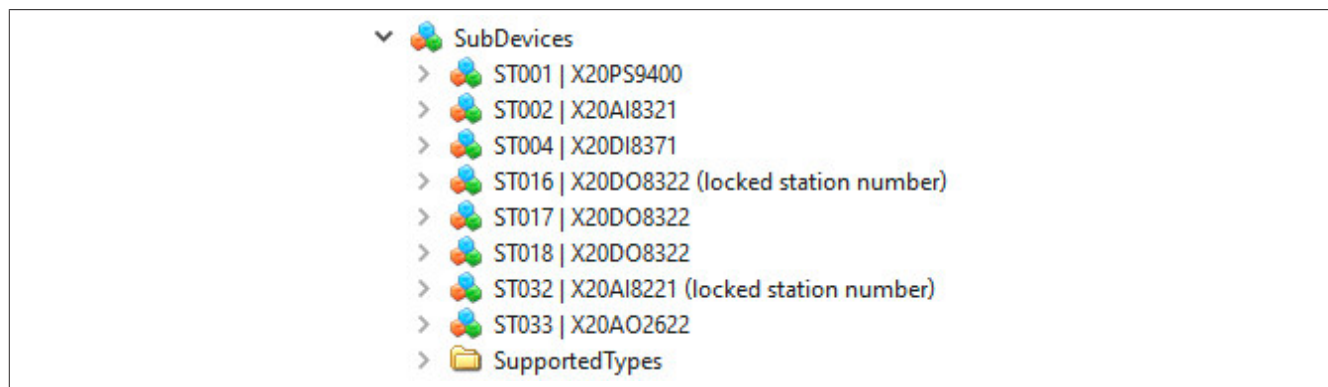
The X2X Link object is subordinate to the bus controller object (X20BC008T) and contains all data related to the X2X Link network. The available configuration methods and variables can be used to add, move or delete stations. In addition, the stations can be configured via corresponding parameters.

6.2.1 X2X Link commissioning

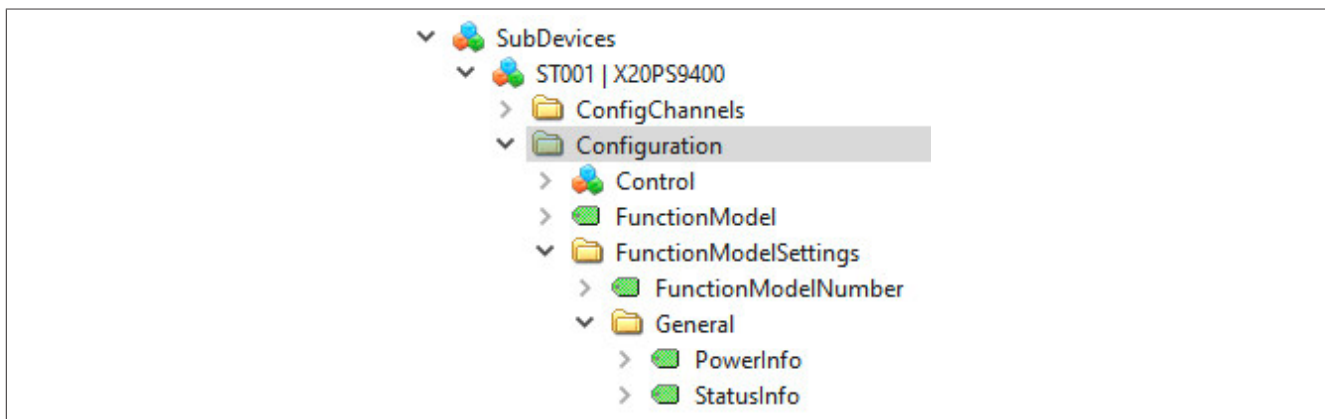
To commission a bus controller that has not yet been configured, the required modules must first be added to the OPC UA model. This is done using the methods described in section [6.2.2 "Methods for X2X Link configuration"](#):



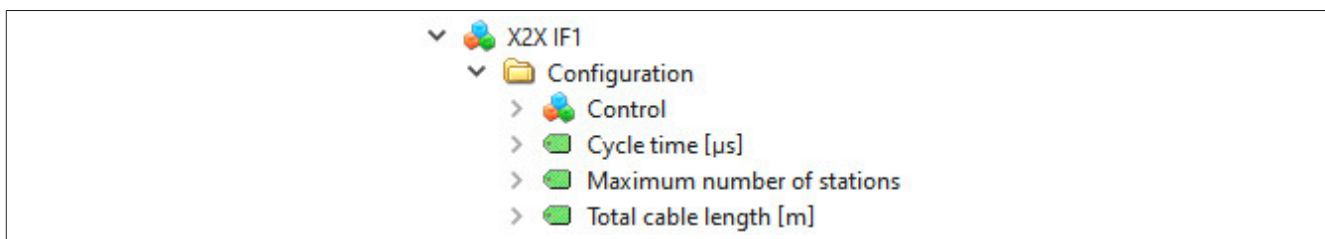
- All modules physically connected on the bus controller can be added automatically using method "ScanForNewStations". The information model must then be updated in the browser to display the stations under "SubDevices".



- Depending on requirements, the configuration of the modules now takes place in the OPC UA model. If this is not changed, the preset configuration is applied.



The global X2X Link parameters are located in folder "Configuration" below the interface node:



- The configuration is saved and applied to the X2X Link network using method [6.2.2.10 "ApplyChanges"](#). This saved configuration is loaded at every restart and automatically started.

If the I/O configuration of the modules or X2X Link parameters are subsequently changed, the change must be applied again with "ApplyChanges".



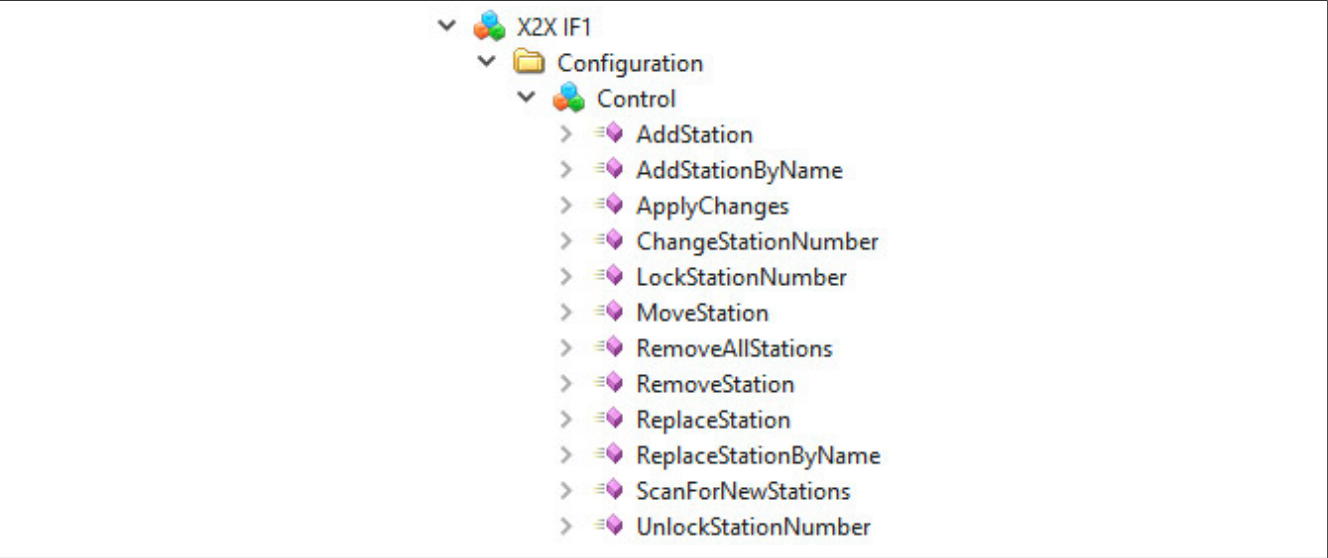
Information:

Method "ApplyChanges" in this object only refers to the X2X configuration and the configuration of the underlying I/O modules. Other configuration values, such as the network configuration, are not applied with this.

6.2.2 Methods for X2X Link configuration

The physical X2X Link interface is offered a number of methods in the OPC UA information model for controlling and configuring the X2X Link. The name (or NodeId) of the interface corresponds to the interface address as is also common in Automation Studio ("IF1").

The methods of IF1 are located in node "IF1@Control" and have NodeId "IF1.Control@<MethodName>":



Information:

Model manipulations and configuration parameters of the stations are only applied and permanently stored when **6.2.2.10 "ApplyChanges"** is called.

6.2.2.1 AddStation / AddStationByName

Adds new stations to the OPC UA information model.
Subsequent modules are not moved. An error is reported if the station number is not available.

AddStation

Input arguments	Data type	Description
Station	UInt32	Station number (0 = append after the last module)
ModuleId	X2XAvailableModules	Module ID of the module to be generated

AddStationByName

Input arguments	Data type	Description
Station	UInt32	Station number (0 = append after the last module)
ModuleName	String	Module name of the module to be generated

6.2.2.2 ChangeStationNumber

Changes the station number of a station group.

Moves all stations in a [station group](#) to a new position. This allows the configuration to be adapted to a changed node number switch, for example.

The following rules apply:

- The target position can be at any position before or after the current position. It is permitted to skip other station groups.
- The target position specified under Destination must still be available, unless it is located within the station group to be moved.
- There must be sufficient space for all subsequent stations of the station group, i.e. no modules are permitted to be located within the target area.

In the event of error, the method will not be executed.

After moving, the station number of the first module is automatically marked as locked. This means that the module will no longer be moved when other groups are moved. See [6.2.2.3 "LockStationNumber"](#).

Example

Output state	Move: Station 16 to station 17	Move: Station 16 to station 64
Input arguments	Data type	Description
Station	UInt32	Station number
Destination	UInt32	Station number of the target position

6.2.2.3 LockStationNumber

Locks a station number.

This can be used to define related station groups. A station group contains all stations from the locked station number up to but excluding the next locked station number (see 1 in the image below). Defined station groups can be moved independently of one another by calling method [6.2.2.2 "ChangeStationNumber"](#).

Example of several station groups

The lock status of a station number can be read out using property LockedStationNumber of the respective station.

Input arguments	Data type	Description
Station	UInt32	Station number

6.2.2.4 MoveStation

Moves a station in the OPC UA information model to another position.

An error is reported if the station number is not available.

Input arguments	Data type	Description
Station	UInt32	Station number
Destination	UInt32	Station number of the target position



Information:

Stations with a locked station number cannot be moved. In this case, method [6.2.2.2 "ChangeStationNumber"](#) must be used.

6.2.2.5 RemoveAllStations

Removes all stations from the OPC UA information model.

6.2.2.6 RemoveStation

Removes one station from the OPC UA information model.

Subsequent modules are not moved.

Input arguments	Data type	Description
Station	UInt32	Station number

6.2.2.7 ReplaceStation / ReplaceStationByName

Replaces a station in the OPC UA information model with another module.

No change is made if a module with the same name already exists at the station number. If the name changes, the module is replaced with the new one. Otherwise, the module is created again.

The modified module is immediately included in the OPC UA information model and can be further configured using the OPC UA client.

ReplaceStation

Input arguments	Data type	Description
Station	UInt32	Station number (0 = append after the last module)
ModuleID	X2XAvailableModules	Module ID of the module to be generated

ReplaceStationByName

Input arguments	Data type	Description
Station	UInt32	Station number (0 = append after the last module)
ModuleName	String	Module name of the module to be generated

6.2.2.8 ScanForNewStations

Scans the X2X Link network for new stations.

To scan the X2X Link network, it must be fully started up and ready beforehand. Otherwise, an error is reported.

This method checks whether new stations have been found on the X2X Link network that are not yet present in the configuration. Newly found stations are automatically added to the OPC UA information model.

Modules that are connected on bus modules with node number switches are automatically marked as locked and registered with the corresponding number.



Information:

If the node number switch of a bus module has been changed, this change must first be carried out manually with method [6.2.2.2 "ChangeStationNumber"](#). Otherwise, the stations will be added again at the new position.

6.2.2.9 UnlockStationNumber

Unlocks a station number and dissolves a station group.

For details, see [6.2.2.3 "LockStationNumber"](#).

Input arguments	Data type	Description
Station	UInt32	Station number

6.2.2.10 ApplyChanges

Applies a new or changed configuration.

Any configuration changes made are permanently saved and reloaded when the system is restarted. The available channels, which depend on the configuration of the modules, are visible in folder "ProcessData" via the OPC UA client after this step.

If the configuration was applied without errors, it is then started on the X2X Link network. If an error occurs when checking and saving the configuration, this is reported and the old configuration is retained.

6.2.3 Flatstream interface

Some modules have a Flatstream interface. This divides larger data packets (also known as frames) into smaller packets and transfers them piece by piece via the X2X Link network.

The OPC UA bus controller automatically operates this Flatstream interface, i.e. the piece-by-piece transfer of a frame. The chapter in the module documentation that describes the Flatstream communication itself can therefore be ignored.

Only the frames that are exchanged via the Flatstream interface and access to the data via OPC UA are important.



Information:

These functions are available starting with firmware version 1.4.6.

6.2.3.1 Modules with a Flatstream interface

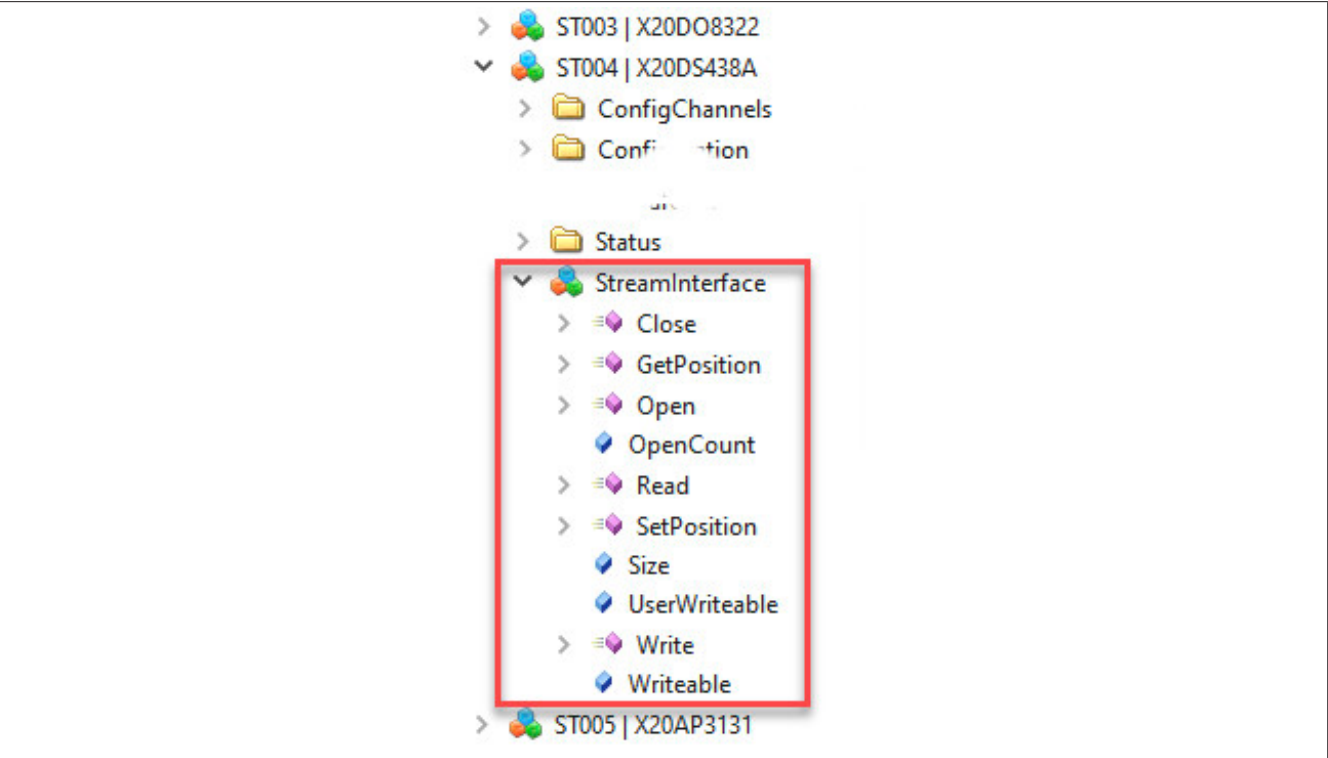
The structure of the module-specific frames is described in the respective module documentation.

Module	Usage	Name of the streaming object	Description of the stream data in the module documentation
X20CS1020 X20CS1030 X67IF1121-1	Accessing serial interface data	IF1	Serial on Flatstream
X20CS1070 X20CS2770	Accessing CAN interface data	IF1, IF2	The CAN object
X20CS1012	Reading and writing M-Bus frames	StreamInterface ¹⁾	M-Bus on Flatstream
X20CMR010 X20CMR100 X20CMR111	Accessing nonvolatile internal flash memory	UserFlashInterface ¹⁾	Internal module memory for user data
X20AO2438 X20AI2438	Reading and writing HART commands	StreamInterface ¹⁾	HART on Flatstream
X20DS438A X67DS438A	Accessing the object dictionary object of IO-Link devices	StreamInterface ¹⁾	IO-Link information for the Flatstream
X20APxxxx	Collected process data of the module	StreamInterface ¹⁾	Interface for transferring the process variable image

1) Must be enabled in the module configuration.

6.2.3.2 Accessing the Flatstream via OPC UA

A module-specific data stream is accessed in OPC UA via object "StreamInterface" of type "FileType". This object is subordinate to the module object and provides corresponding methods that can be used to read and write the streaming data.



6.2.3.2.1 Open

This method must be called initially to open the interface.

Input arguments	Data type	Description
Mode	Byte	1 = RO - Open read-only 2 = WO - Open write-only 3 = RW - Open read and write 4 to 255 = Reserved
Output arguments	Data type	Description
FileHandle	UInt32	Needed for methods 6.2.3.2.3 "Read", 6.2.3.2.4 "Write" and 6.2.3.2.2 "Close".



Information:

- The file is automatically closed if it is not accessed for more than one minute (by calling methods 6.2.3.2.3 "Read" or 6.2.3.2.4 "Write") or if the session of the client that called method Open is closed.
- The file handle is only valid within the respective OPC UA session.
- A streaming file cannot be opened multiple times simultaneously.

6.2.3.2.2 Close

Used to close the stream interface. The required resources are freed up, and the interface can be reopened by another client.

Input arguments	Data type	Description
FileHandle	UInt32	File handle that was returned by method 6.2.3.2.1 "Open".

6.2.3.2.3 Read

This method can be used to read data from the streaming interface. The following applies:

- If parameter "Length" is specified large enough, the entire frame is read out from the buffer with each call.
- If "Length" was specified as less than the length of the next frame, then a subframe with length "Length" is returned with each call.

Input arguments	Data type	Description
FileHandle	UInt32	File handle that was returned by method 6.2.3.2.1 "Open" .
Length	Int32	Maximum data length that should be returned.

Output arguments	Data type	Description
Data	ByteString	Read data. If the frame is smaller than the maximum value specified by "Length", a correspondingly shorter byte string is returned.

6.2.3.2.4 Write

This method is used to write data to the stream interface. A frame is written with each call.

Input arguments	Data type	Description
FileHandle	UInt32	File handle that was returned by method 6.2.3.2.1 "Open" .
Data	ByteString	Data that should be written.

6.2.3.2.5 GetPosition / SetPosition

These methods are not supported for streaming files.

6.2.3.2.6 Variables

Size

Shows the size of the next frame that can be read.

If the value 0 is returned, no new data is available. The variable can be monitored using "MonitoredItem" to determine when new data can be read.

OpenCount

Shows how often the file is currently open.

Only 0 or 1 since streaming files can only be opened once.

6.3 PubSub configuration

The OPC UA publisher-subscriber configuration can be performed by transferring a ".uabinary" file. This file contains all the information required for cyclic data exchange. UaExpert is equipped with an integrated editor to create and edit such a file (for a detailed example, see section [3.9.4 "PubSub configuration example"](#)).

The file can be read or written using object `Root/Objects/Server/PublishSubscribe/PubSubBinary`. The configuration is enabled as soon as a new file is written.

The file contains the following configuration data:

- Publisher configuration: Specifies which process data is sent periodically and the period with which this takes place.
- Subscriber configuration: Specifies which process data of a publisher is received and where this data is forwarded to.
- General connection configuration

Currently, only cyclic process data from X2X modules can be used in publisher and subscriber configurations. These are all nodes in the process data folder of an X2X module except for nodes `ModuleOk`, `ModuleId`, `SerialNumber` and `FirmwareVersion`.

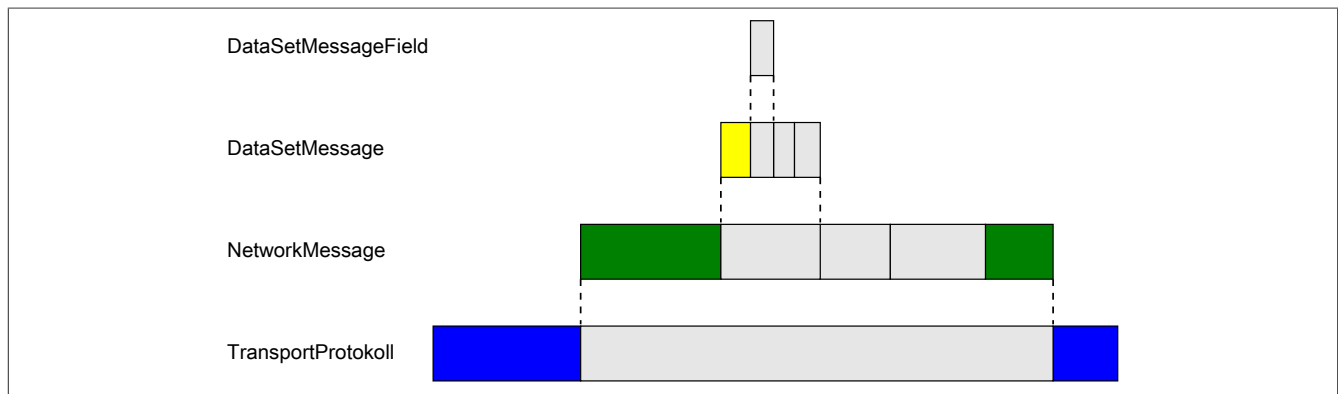
6.3.1 General information

In the PubSub communication model, communication takes place between "publishers" and "subscribers". Publishers send the messages. Subscribers receive the messages and process them further.

The X20BC008T currently only supports the "UADP Periodic Fixed" header layout and transport protocol "opc.udp":

- **UADP Periodic Fixed:** In this mode, messages are transmitted periodically at a fixed interval, similar to POWERLINK, regardless of changes in the data. The variables (data points) that are transferred within a message are defined at configuration time.
- **Opc.udp:** Data is typically transmitted in the multicast network. Receipt of the transmitted messages is not confirmed, i.e. there is no check as to whether or how many subscribers have received a message.

6.3.1.1 Structure of a PubSub message



1) The **DataSetMessageField** contains a single variable value / data point.

2) Each **DataSetMessage** consists of one or more DataSetMessageFields. Among other things, the header (highlighted in yellow) of the DataSetMessage contains a DataSetWriterId and status information:

- **DataSetWriterId:** Identifies a dataset within the NetworkMessage. The DataSetWriterId must therefore be unique within the NetworkMessage.
- **Status information:** Provides information about the validity of the contained DataSetMessageFields.



Information:

With the X20BC008T, one DataSetMessage is created per I/O module in the default configuration.

3) There are one or more DataSetMessages within a **NetworkMessage**. Among other things, the header (highlighted in green) of the NetworkMessage contains various IDs and GroupVersion:

- **WriterGroupId:** Identifies different messages from the same publisher and must therefore be unique within the publisher.
- **PublisherId:** Identifies the publisher and must therefore be unique within the network.
- **GroupVersion:** Identifies the version of a message. The version is automatically changed when a message is reconfigured.

4) The NetworkMessage is transferred via the **transport protocol**.



Information:

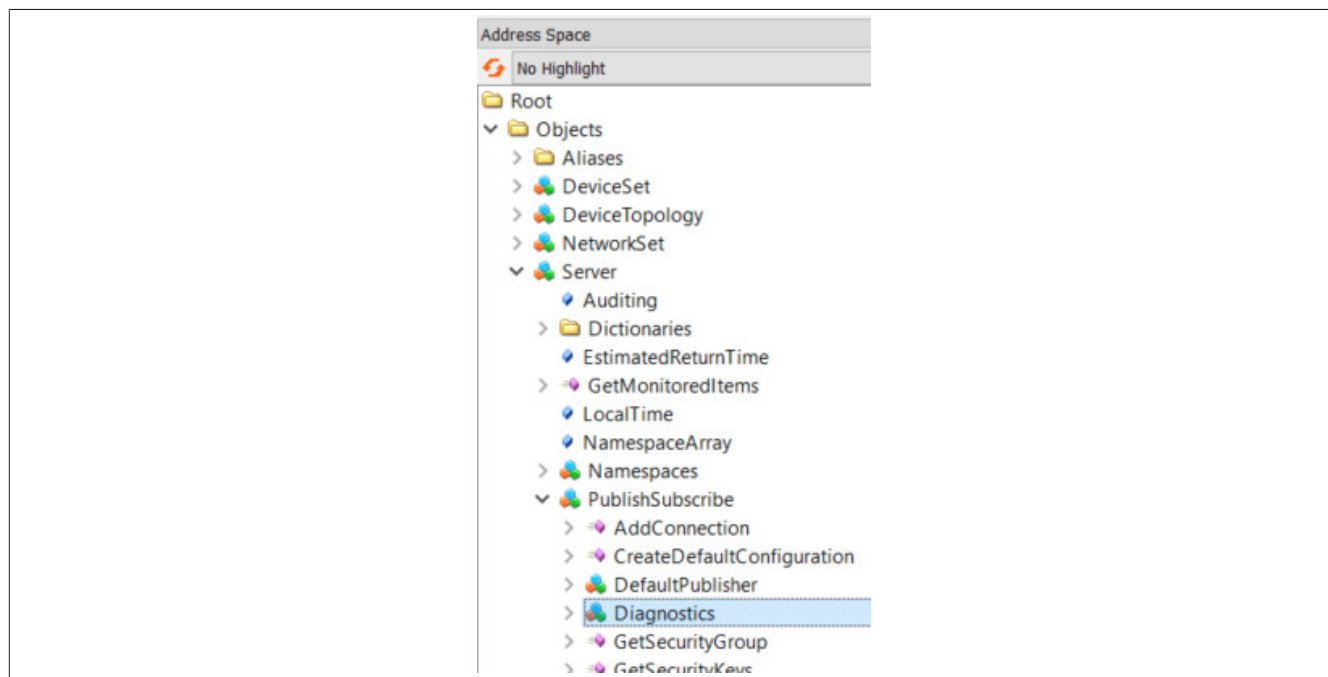
For a subscriber to accept and process messages, all 4 IDs (DataSetWriterId, Writer-GroupId, PublisherId and GroupVersion) must match in the respective reader (subscriber) configuration.

For a configuration example, see [3.9.4 "PubSub configuration example"](#).

6.4 PubSub diagnostics

A valid configuration provides diagnostic data points in the information model of the device. This can be used to diagnose a faulty PubSub configuration and evaluate the configuration status.

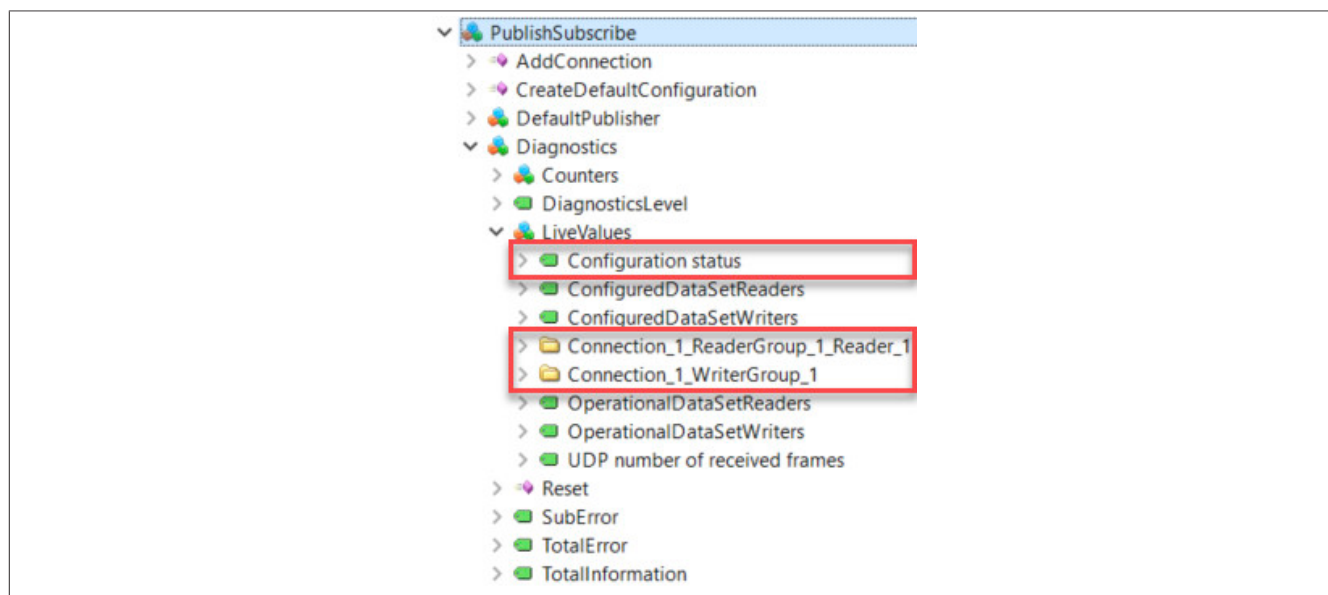
The generated diagnostic data points are located in the path Root/Objects/Server/PublishSubscribe/Diagnostics.



In object Diagnostics, only the subnodes of object LiveValues marked in red can be used for diagnostic purposes.

- Node "Configuration status" contains information about the current configuration.
- A separate folder is created for each publisher/subscriber configuration, provided that the PubSub configuration file (PubSub.uabinary) has been read successfully. The name of the folder is derived from the connection and group name of the PubSub configuration, e.g. "Connection_1_WriterGroup_1".

All other nodes or objects are not supported and do not contain any values. Object LiveValues is located in the path Root/Objects/Server/PublishSubscribe/Diagnostics/LiveValues.



Configuration

6.4.1 Configuration status

Node "Configuration status" contains information about the current configuration.

Node name	Description
Configuration status	Current configuration status 0: OK -1: Problem with the configuration of the kernel drivers -2: Problem with the interpretation of file "PubSub.uabinary" (e.g. the mapped nodes do not exist) -99: Configuration not yet loaded

6.4.2 Writer Group diagnostic node

Diagnostic node <Writer Group Name> is generated by a valid WriterGroup configuration and located in the path Root/Objects/Server/PublishSubscribe/Diagnostics/LiveValues/<Writer Group Name>.

Publishers have limited diagnostics compared to subscribers since only the sending of telegrams is monitored.

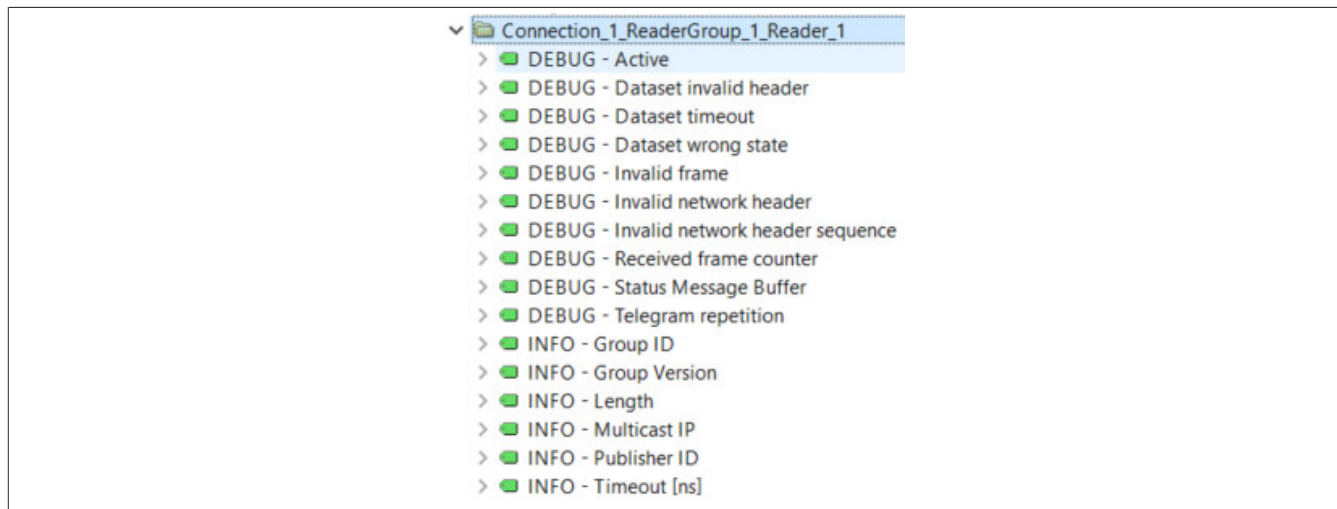


Node name	Description
DEBUG - Active	Packet sending is active / not active.
DEBUG - Status message buffer	0: No error -1: Socket could not be opened
INFO - Group ID	GroupId of the WriterGroup
INFO - Group version	GroupVersion of the WriterGroup
INFO - Interval [ns]	Transmission interval of the WriterGroup in ns
INFO - Length	UADP telegram length - except the Ethernet and IP/UDP header
INFO - Multicast IP	Multicast IP destination address of the publisher, e.g. 239.0.0.1
INFO - Publisher ID	PublisherId of the Publisher

6.4.3 Reader Group diagnostic node

Diagnostic node <Reader Group Name> is generated by a valid ReaderGroup configuration and located in the path Root/Objects/Server/PublishSubscribe/Diagnostics/LiveValues/<Reader Group Name>.

Compared to publishers, subscribers have extended diagnostics since telegrams detected as invalid are recorded in different error counters.



Node name	Description
DEBUG - Active	Packet sending is active / not active.
DEBUG - Dataset invalid header	The counter is incremented if the dataset header flags do not match the subscriber configuration.
DEBUG - Dataset timeout	The counter is incremented if a dataset fails, i.e. no new data was received before the set timeout expired.
DEBUG - Dataset wrong state	The counter is incremented if the status field of the DataSetMessage header contains a "StatusCode severity" not equal to "Good".
DEBUG - Invalid frame	The counter is incremented if the network header or group flags do not match the subscriber configuration, i.e. the expected telegram header. If no data has been received since the time of PubSub configuration, the counter also increases (receive buffer contains 0 as the telegram header).
DEBUG - Invalid network header	The counter is incremented if any of the extended flags, PublisherId, WriterGroupId or GroupVersion fields do not match the subscriber configuration.
DEBUG - Invalid network header sequence	The counter is incremented if SequenceNumber in the network header was not ascending compared to the predecessor.
DEBUG - Status message buffer	0: No error -1: Socket could not be opened
DEBUG - Telegram repetition	The counter is incremented if the same telegram was received twice in succession, i.e. the SequenceNumber of the NetworkHeader is identical. Information: With non-synchronized systems, it is likely that this error counter counts up slowly.
INFO - Group ID	Expected GroupID
INFO - Group version	Expected GroupVersion
INFO - Length	Expected UADP telegram length - except the Ethernet and IP/UDP header
INFO - Multicast IP	Multicast IP address at which the subscriber expects telegrams, e.g. 239.0.0.1
INFO - Publisher ID	Expected PublisherId
INFO - Timeout [ns]	Configured message receive timeout of the DataSet reader in ns

7 Status

The OPC UA information model of the bus controller shows status information that should be used for information or diagnostics of malfunctions. These include:

- [Port status](#)
- [Time synchronization](#)
- [Network](#)

7.1 Port status

The corresponding node names are located in the OPC UA information model under nodes `Root/Objects/DeviceSet/X20BC008T/Status/BridgePorts`. Under this path, there is an entry for each port of the bus controller named after the port itself (ETHx). The following table provides the status information available for each port.

Node name		Description
InternalName		System-internal name of the interface.
FrameStatistics/		
FcsErrorFrameCount		Number of Ethernet frames received on the respective port with a corrupt Ethernet Frame Check Sequence (FCS).
GeneralRxErrorFrameCount		Number of incoming Ethernet frames that could not be received due to internal bus controller errors.
GeneralTxErrorFrameCount		Number of Ethernet frames to be transmitted that could not be transmitted due to internal bus controller errors.
RxFrameCount		Number of Ethernet frames received successfully on the respective port.
SizeErrorFrameCount		Number of Ethernet frames received on the port that were discarded due to invalid length (< 64 byte or > max. Ethernet frame length).
TxFrameCount		Number of Ethernet frames transmitted successfully on the respective port.
LinkPartner/		
ChassisId		Name of component "Chassis" for the connected device, e.g. the MAC address.
ManagementAddress		Management address of the connected device, e.g. the IP address.
PortId		Name of component "Port" for the connected device, e.g. the internal name of the interface.
LinkProperties/		
Duplex		Duplex mode of the port. Possible values: Full Port operates in full-duplex mode Half Port operates in half-duplex mode No entry Connection is not active.
LinkStatus		Status of the connection. Possible values: UP Connection is active. DOWN Connection is not active.
Speed		Speed of the connection. Possible values: 100 MB/s Connection is working with 100 Mbit/s. 1000 MB/s Connection is working with 1000 Mbit/s. No entry Connection is not active.

7.2 Time synchronization

The state of time synchronization can be queried via the nodes in object `Root/Objects/DeviceSet/X20BC008T/Status/TimeSynchronization`. The corresponding information is available for both `WallClock` and `WorkingClock`.

Node name	Description
WallClock/NTP/	
SyncOK	Status of the NTP synchronization of WallClock. Possible values: True WallClock is synchronized with a time server. False WallClock is not synchronized with a time server.
TimeServer	URL or IP address of the time server that is used to synchronize WallClock.
WallClock/PTP/	
ClockIdentity	Unique identifier of the PTP instance of WallClock.
GrandmasterIdentity	Identity of the PTP instance that serves as a grandmaster for WallClock on the network.
OffsetFromMaster	Calculated time deviation of the local WallClock from the clock of the grandmaster in 1/65536 nanoseconds ¹⁾ .
ParentPortIdentity	Identity of the port of the neighboring device that is used to transmit PTP synchronization messages to the PTP instance of the local WallClock. The identity is represented as a byte string. If the local WallClock is the grandmaster instance, this string corresponds to <code>ClockIdentity</code> followed by 2 null bytes.
WallClock/PTP/ETHx/	
PortIdentity	Unique port identifier of Ethernet port ETHx with enabled PTP synchronization of WallClock on this port. x corresponds to the port number on the housing of the bus controller.
PortState	Status of WallClock PTP synchronization on Ethernet port ETHx . Possible values: 3 Port is disabled. 6 Port is a master for WallClock. 7 Port is passive. 9 Port is a slave for WallClock. For detailed information, see IEEE 802.1AS – 2020, table 14-7.
WorkingClock/PTP/	
ClockIdentity	Unique identifier of the PTP instance of WorkingClock.
GrandmasterIdentity	Identity of the PTP instance that serves as a grandmaster for WorkingClock on the network.
OffsetFromMaster	Calculated time deviation of the local WorkingClock from the clock of the grandmaster in 1/65536 nanoseconds ¹⁾ .
ParentPortIdentity	Identity of the port of the neighboring device that is used to transmit PTP synchronization messages to the PTP instance of the local WorkingClock. The identity is represented as a byte string. If the local WorkingClock is the grandmaster instance, this string corresponds to <code>ClockIdentity</code> followed by 2 null bytes.
WorkingClock/PTP/ETHx/	
PortIdentity	Unique port identifier of Ethernet port ETHx with enabled PTP synchronization of WorkingClock on this port. x corresponds to the port number on the housing of the bus controller.
PortState	Status of WorkingClock PTP synchronization on Ethernet port ETHx . Possible values: 3 Port is disabled. 6 Port is a master for WorkingClock. 7 Port is passive. 9 Port is a slave for WorkingClock. For detailed information, see IEEE 802.1AS – 2020, table 14-7.

1) Value 65536 = 1 nanosecond.

7.3 Network

The currently used network configuration can be read out via the nodes in object `Root/Objects/DeviceSet/X20BC008T/Status/Network`.

Node name	Description
CurrentDNS	DNS server currently being used. The string can contain multiple entries if multiple DNS servers are available.
CurrentGateway	Default gateway currently being used
CurrentHostname	Hostname currently being used
CurrentIPConfig	Current IP configuration. The string can contain multiple entries if multiple IP addresses exist (e.g. a temporary IP address added by adjusting the node number switch).

8 Cybersecurity



Information:

For cyber-secure operation of the bus controller, the guidelines from [Cybersecurity / Defense in Depth for B&R products](#) must be taken into account.

This chapter gives a brief introduction to cybersecurity. The terms are only described on a general level. General terms can therefore refer to a number of different aspects depending on the situation.

Devices are delivered with factory settings. This means that normally neither any device functionality nor any security settings are configured. To enable secure commissioning of these devices, it is important to ensure that they are only used in a trusted environment. This can be achieved, for example, by separating the machine network from the company network or by connecting the devices directly to the PC used for configuration, for example.

OPC UA over TSN enables converged IT/OT networks in which it cannot be assumed that all network stations can be trusted. This does not require an explicit attack. Even a misconfiguration of controllers outside the actual machine network can result in unintentional malfunctions.

Cybersecurity issues therefore play an important role in OPC UA over TSN and both technologies (OPC UA as well as TSN) contain all the mechanisms required for this.



Information:

The device has only limited resources and cannot process all messages that could potentially arrive via the network. A message storm on the network can cause a temporary outage. To avoid this, the device must be operated behind a firewall with rate limiting.

Security-relevant errors and notifications

Cybersecurity depends on an open error culture. Errors in device firmware that allow unauthorized access, for example, are actively tracked and managed by B&R. Critical security gaps and their corrections are collected and made available under <https://www.br-automation.com/en/service/cyber-security/>.



Information:

All errors concerning the safety of B&R equipment should be reported immediately to the website specified above.

8.1 Basic terms and information

8.1.1 Encryption

The aim of encryption is to make sensitive data unreadable for external parties. Even if an attacker has access to the data, e.g. by using tools to follow the traffic, it should be impossible for him to derive valuable information from it.

Encryption also distinguishes between whether data remains stored on a computer, device or data storage medium, or whether it is transferred over a communication medium. The basic mechanisms are similar in all cases.

The industry standard is AES encryption (Advanced Encryption Standard), which uses key lengths of 128 or 256 bits. Both OPC UA and NETCONF support this standard.

8.1.2 Integrity

Attackers do not necessarily need to decrypt secret data to cause, for example, disruptions in machine operation. It is often sufficient just to corrupt the data. This is possible even if the data is encrypted and actually unreadable.

To prevent this threat, a digital signature is added to the data. This signature is comparable to a CRC checksum (Cyclic Redundancy Check). The algorithms are explicitly designed to detect falsification by an attacker.

It is often sufficient to ensure the integrity of the data without encrypting it. This includes the following use cases:

- Diagnostics of data traffic using tools such as Wireshark:
A digital signature does not prevent diagnostics, whereas additional encryption would make the data unreadable and therefore useless for diagnostics.
- Ensuring the integrity of the device firmware:
The firmware can be read out, but attackers cannot make any changes.

The industry standard for signature algorithms is the (SHA) Secure Hash Algorithm with key lengths, for example, of 256 bits. Both OPC UA and NETCONF support these algorithms.

8.1.3 Symmetric and asymmetric keys

Algorithms such as AES are called "symmetric" since a single key is used for both encryption and decryption. If encrypted data should be exchanged between 2 devices, it must be ensured that both devices have the same key. This is not always easy to do.

Algorithms such as RSA (named after their inventors Rivest, Shamir and Adleman), however, are referred to as "asymmetric". These algorithms use 2 different keys to simplify the issue of key exchange.

- A "private" key is used to decrypt data or to create the signature.
- A "public" key is used to encrypt data or to check the authenticity of a signature.

Anyone can read the public key. Devices that want to communicate with each other exchange their public keys. Since the private key is not transmitted, it can easily be kept secret on the device.

Sequence of data transfer with asymmetric algorithms:

- Transmitter A signs the data with its private PV_A key.
- Transmitter A encrypts the signed data with the public PB_B key of receiver B.
- Receiver B decrypts the encrypted and signed data with its private PV_B key.
- Receiver B checks the authenticity of the signed data with the public PB_A key of transmitter A.

The disadvantage of asymmetric algorithms is that they require more computational effort. Since keys with a length of 2048 bits are used for RSA, they are not suitable for exchanging large amounts of data. Symmetrical algorithms, however, only use key lengths of 256 bits, which makes data transfer much easier.

In practice, both procedures are therefore used in combination. Asymmetric algorithms are used to exchange a symmetric key temporarily generated for the communication session. The actual communication afterwards takes place with symmetric algorithms.

8.1.4 Asymmetric key exchange

Even though a public key is permitted to be read and used by anyone, caution is needed when using it. Transmitter A must, for example, be sure that the public PB_B key actually belongs to the desired receiver B. If this is not the case, an attacker could interfere with the communication through a "man-in-the-middle" attack:

Transmitter A \leftrightarrow Attacker C \leftrightarrow Receiver B

- Transmitter A signs the data with its private PV_A key.
- Instead of using the public PB_B key of receiver B, transmitter A encrypts the signed data with the public PB_C key from attacker C.
- Attacker C decrypts the encrypted and signed data with its private PV_C key.
- Attacker C reads the data and possibly corrupts it.
- Attacker C signs the data with its private PV_C key.

Cybersecurity

- Attacker C encrypts the signed data with the public PB_B key from receiver B.
- Instead of using the public PB_A key of transmitter A, receiver B checks the authenticity of the signed data with the public PB_C key from attacker C.

The attacker can also read and corrupt data in the reverse communication direction.

There are 3 ways to prevent a man-in-the-middle attack:

- 1) Ensure that no attacker is able to be present when communication is established, e.g. by disconnecting the machine from the intranet and Internet. The exchanged keys will be secure, even if the machine is reconnected to the intranet and Internet.
- 2) Ensure that the received public PB_X key actually belongs to communication station X. This is possible if there is a trusted third party guaranteeing that this correlation exists.
- 3) Distributing the public keys to the respective devices in a suitable manner. This includes manual interaction by a user or administrator.

NETCONF supports the first and third option if communication protocol Secure Shell (SSH) is used. OPC UA supports the second and third option.

8.1.5 Chain of trust and authority

Transfer protocol Hypertext Transport Protocol Secure (HTTPS), which is also used on the Internet, is based on the fact that a trusted third party guarantees that the public PB_X key belongs to communication station X.

This is guaranteed in the "certificate" including additional information. The format of the certificates was standardized by the International Telecommunication Union (ITU) and follows the X.509 standard. The "guaranteeing" party is referred to as a "certificate authority" (CA).

A web browser accepts the certificate for <https://www.br-automation.com> since, for example, it can be proved that it was issued by certification authority "GlobalSign" and the web browser trusts this certification authority. The certificate for <https://www.br-automation.com> must therefore be protected against corruption, which in turn is ensured via [symmetric and asymmetric procedures](#).

While a CA can be certified by a higher CA, there are CAs that are basically trusted by web browsers and other devices and are specified. They are called "root CAs". These form the highest certificate authority on the Internet.

- Root CA R generates certificate C_R for itself that contains its public PB_R key.
- Root CA R signs certificate C_R with its private PV_R key (self-signed certificate).
- Device A (or the web browser) imports certificate C_R and can use it to check whether additional certificates have been issued by root CA R.
- Root CA R generates certificate C_A for device A that contains its public PB_A key.
- Root CA R signs certificate C_A with its private PV_R key.
- Device B (or the web browser) imports certificate C_R and can use it to check whether additional certificates have been issued by root CA R.
- Root CA R generates certificate C_B for device B that contains its public PB_B key.
- Root CA R signs certificate C_B with its private PV_R key.

As soon as device A and device B establish a communication connection, they first exchange their certificates:

- Transmitter A transmits its C_A certificate to receiver B.
- Receiver B verifies the integrity of certificate C_A using the public PB_R key taken from certificate C_R of root CA R.
- Receiver B transmits its C_B certificate to transmitter A.
- Transmitter A verifies the integrity of certificate C_B using the public PB_R key taken from certificate C_R of root CA R.
- Transmitter A signs the data with its private PV_A key.
- Transmitter A encrypts the signed data with the public PB_B key of receiver B taken from certificate C_B .
- Receiver B decrypts the encrypted and signed data with its private PV_B key.
- Receiver B checks the authenticity of the signed data with the public PB_A key of transmitter A taken from certificate C_A .

The use of a certification authority results in increased effort in the beginning. For larger systems or machines, however, this eliminates the task of distributing certificates to the individual devices.

**Information:**

In companies that have their own IT department, the necessary requirements for a Public Key Infrastructure (PKI) are usually in place.

OPC UA uses certificates in X.509 format. Even if no certificate authority is used, the public PB_x key for device X must be wrapped in certificate C_x and signed by the device with its private PV_x key.

When establishing the connection for the first time, receiver B cannot ensure whether certificate C_A actually originates from transmitter A or from a man-in-the-middle and must therefore blindly trust this certificate. This is the reason why a warning appears when connecting to a device for the first time with a program like UaExpert.

8.2 User access

Assigning access rights

The bus controller has a user rights and role system that defines the actions of a logged-in user. All users usually do not have the same rights.

It is much more common to define only one or more administrators who are permitted to make sensitive settings on the bus controller.

User identification

Access to the bus controller is usually provided via authentication. Identification takes place either with their username and password or, in the case of NETCONF, with an SSH key.

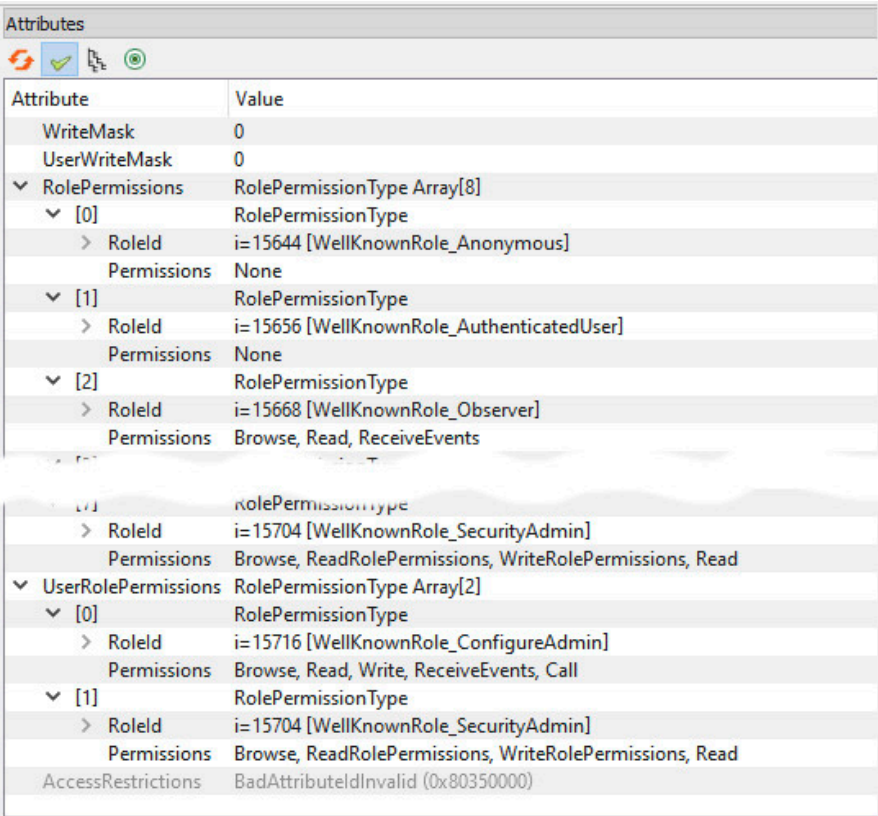
Only first access to the bus controller in configuration mode is anonymous since at this time no known users exist on the bus controller and they must be created first.

Role assignment

OPC UA provides 8 "known roles" listed under node `Root/Objects/Server/ServerCapabilities/RoleSet`. Users can be assigned one or more of these roles (see ["Assigning role SecurityAdmin"](#)).

Each node in the information model has attributes `RolePermissions` and `UserRolePermissions`. `UserRolePermissions` displays the permissions for a user's roles for this node. `SecurityAdmins` have permission to read attribute `RolePermissions`, which displays the permissions of all roles on the node.

Example of possible values of attributes `RolePermissions` and `UserRolePermissions`:



Attribute	Value
WriteMask	0
UserWriteMask	0
RolePermissions	RolePermissionType Array[8]
[0]	RolePermissionType
> RoleId	i=15644 [WellKnownRole_Anonymous]
Permissions	None
[1]	RolePermissionType
> RoleId	i=15656 [WellKnownRole_AuthenticatedUser]
Permissions	None
[2]	RolePermissionType
> RoleId	i=15668 [WellKnownRole_Observer]
Permissions	Browse, Read, ReceiveEvents
[3]	RolePermissionType
> RoleId	i=15704 [WellKnownRole_SecurityAdmin]
Permissions	Browse, ReadRolePermissions, WriteRolePermissions, Read
UserRolePermissions	RolePermissionType Array[2]
[0]	RolePermissionType
> RoleId	i=15716 [WellKnownRole_ConfigureAdmin]
Permissions	Browse, Read, Write, ReceiveEvents, Call
[1]	RolePermissionType
> RoleId	i=15704 [WellKnownRole_SecurityAdmin]
Permissions	Browse, ReadRolePermissions, WriteRolePermissions, Read
AccessRestrictions	BadAttributeIdInvalid (0x80350000)

The nodes in the information model are assigned to groups. All nodes within a group have the same permission settings. The permission settings are fixed and cannot be changed.

The following table shows the possible access rights of the roles for the nodes within the different groups:

Group	Node path	Role	Permissions					
			B ¹⁾	R ²⁾	RE ³⁾	W ⁴⁾	C ⁵⁾	RP ⁶⁾
Default	All nodes that are not subordinate in any of the other groups	Anonymous						
		AuthenticatedUser						
		Observer	✓	✓	✓			
		Operator	✓	✓	✓			
		Engineer	✓	✓	✓	✓	✓	
		Supervisor	✓	✓	✓		✓	
		ConfigureAdmin	✓	✓	✓	✓	✓	
		SecurityAdmin	✓	✓				✓

Group	Node path	Role	Permissions					
			B ¹⁾	R ²⁾	RE ³⁾	W ⁴⁾	C ⁵⁾	RP ⁶⁾
Security	Server/ServerConfiguration/* Server/ServerCapabilities/RoleSet/* Server/ServerCapabilities/UserSet/*	Anonymous						
		AuthenticatedUser						
		Observer						
		Operator						
		Engineer	✓	✓	✓			
		Supervisor	✓	✓	✓			
		ConfigureAdmin	✓	✓	✓			
		SecurityAdmin	✓	✓	✓	✓	✓	✓
Configuration	DeviceSet/X20BC008T/Configuration/* DeviceSet/X20BC008T/X2X_IF1/Configuration/* DeviceSet/X20BC008T/X2X_IF1/SubDevices/ST[x]/ Configuration/*	Anonymous						
		AuthenticatedUser						
		Observer	✓	✓	✓			
		Operator	✓	✓	✓			
		Engineer	✓	✓	✓	✓	✓	
		Supervisor	✓	✓	✓			
		ConfigureAdmin	✓	✓	✓	✓	✓	
		SecurityAdmin	✓	✓				✓
ProcessData	DeviceSet/X20BC008T/X2X_IF1/SubDevices/ST[x]/ ProcessData/*	Anonymous						
		AuthenticatedUser						
		Observer	✓	✓	✓			
		Operator	✓	✓	✓	✓	✓	
		Engineer	✓	✓	✓	✓	✓	
		Supervisor	✓	✓	✓		✓	
		ConfigureAdmin	✓	✓	✓	✓	✓	
		SecurityAdmin	✓	✓				✓
User	Server/ServerCapabilities/CurrentUser/*	Anonymous						
		AuthenticatedUser						
		Observer	✓	✓	✓	✓	✓	
		Operator	✓	✓	✓	✓	✓	
		Engineer	✓	✓	✓	✓	✓	
		Supervisor	✓	✓	✓	✓	✓	
		ConfigureAdmin	✓	✓	✓	✓	✓	
		SecurityAdmin	✓	✓	✓	✓	✓	✓
SoftwareUpdate	DeviceSet/X20BC008T/FirmwareUpdate/*	Anonymous						
		AuthenticatedUser						
		Observer						
		Operator						
		Engineer	✓	✓	✓	✓	✓	
		Supervisor	✓	✓	✓			
		ConfigureAdmin	✓	✓	✓	✓	✓	
		SecurityAdmin	✓	✓	✓	✓	✓	✓
X2XConfigChannels	DeviceSet/X20BC008T/X2X_IF1/SubDevices/ST[x]/ ConfigChannels/*	Anonymous						
		AuthenticatedUser						
		Observer	✓	✓	✓			
		Operator	✓	✓	✓			
		Engineer	✓	✓	✓			
		Supervisor	✓	✓	✓			
		ConfigureAdmin	✓	✓	✓			
		SecurityAdmin	✓	✓				✓

- 1) Browse
- 2) Read
- 3) ReceiveEvent
- 4) Write
- 5) Call
- 6) ReadRolePermissions and WriteRolePermissions

8.3 Key management for NETCONF

In order for a NETCONF client to communicate with the bus controller, SSH keys should ideally be used. Although authentication via username and password is possible, SSH keys offer a higher level of security.

First, an SSH key pair must be generated on the device on which the NETCONF client is running. In Linux or Windows with Cygwin, for example, this is done using command line tool ssh-keygen:

```
$ ssh-keygen -q -N "" -f ~/.ssh/id_rsa
```

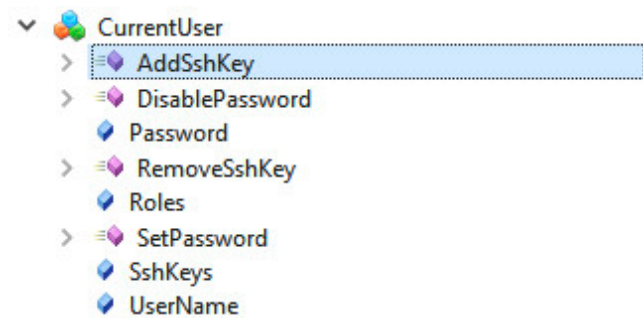
This call generates 2 files:

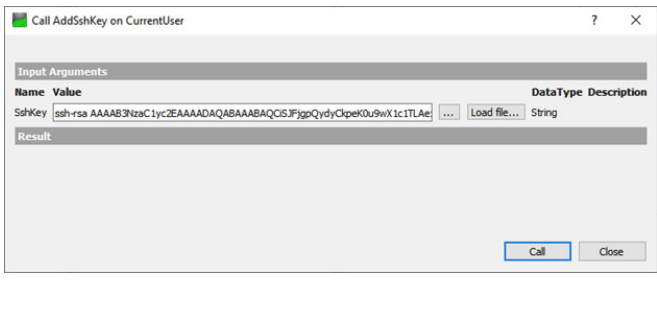
```
~/.ssh/id_rsa
~/.ssh/id_rsa.pub
```

File ~/.ssh/id_rsa contains the private key and must remain protected on the device. File ~/.ssh/id_rsa.pub contains the public key that is transferred to the bus controller. The content of this file is a single ASCII text line of the following type:

```
ssh-rsa AAAAB3NzaC1yc2EAAAAD...UmUCIxYc68QIw+OSoN admin@client
```

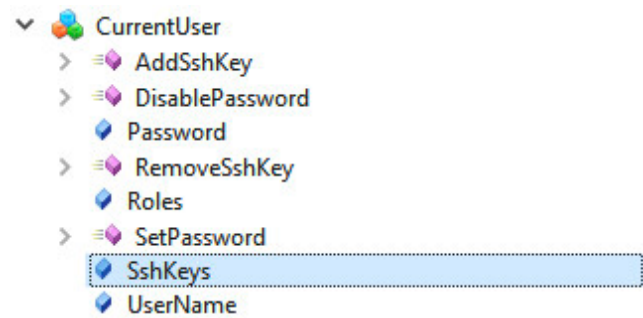
If only one user should be used for all administrative tasks, such as the previously created "admin", the key can be assigned to this user. Method Root/Objects/Server/ServerCapabilities/CurrentUser/AddSshKey must be called for this and the entire text line of the public SSH key copied into it.

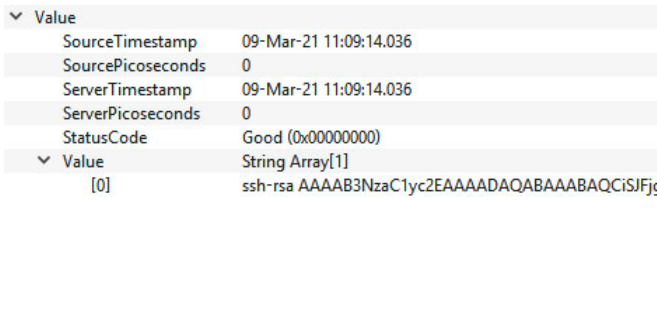




If different devices are used for bus controller management via NETCONF, a separate SSH key can be added for each of these devices. SSH keys that are no longer used can be removed from the bus controller in the same way using function Root/Objects/Server/ServerCapabilities/CurrentUser/RemoveSshKey.

The list of SSH keys is visible at the respective user:





If necessary, different users with their own roles can be defined, who are responsible for different administrative tasks, for example. In addition to a general SecurityAdmin for user and role administration, another ConfigureAdmin would be conceivable, which is responsible for the administration of the TSN functionality of the bus controller. This user would communicate with the bus controller exclusively via NETCONF. In this case, the password can be disabled and access via OPC UA denied.

8.4 Certificate management



Information:

See also [3.8 "Updating the self-signed certificate"](#).

8.4.1 Generating a certificate request

If certificates signed by a certificate authority (CA) should be used, the necessary certificate signing request (CSR) should be generated directly on the device. By generating the CSR on the device, the private key does not have to leave the device, which increases security. This is done as follows:

- Method `CreateSigningRequest` is available under `Root/Objects/Server/ServerConfiguration` to create a CSR. When the method is called, a new private key is optionally generated. If the option is not enabled, the existing key is used. The public key, information about the requester and other information are packed in a "PKCS #10 DER" encoded certificate request that is returned by the method. The byte string must be saved in a corresponding ".csr" file.
- The CSR must then be signed by a certificate authority, with additional information entered in the certificate. The result is a valid certificate.
- The signed certificate can then be installed via method `Root/Objects/Server/ServerConfiguration/UpdateCertificate`.



Information:

- The private key for the CSR remains stored on the device only until a new CSR is generated or until the device is restarted. A signed certificate can only be installed if the corresponding private key is still available.
- UaExpert makes it easy to generate and save the CSR in the GDS Push View. It is thus not required to work directly with the method.

For more details about OPC UA method `CreateSigningRequest`, see the OPC UA specification, Part 12.

8.4.2 Updating the certificate using `UpdateCertificate`

Signed certificates can be installed on the bus controller using method `Root/Objects/Server/ServerConfiguration/UpdateCertificate`. It makes no difference whether it is a certificate signed by a certificate authority or a self-signed certificate. If the certificate was not generated from a CSR generated by the bus controller, the private key must also be specified.

Method `Root/Objects/Server/ServerConfiguration/ApplyChanges` must also be called to apply the changes. All connected clients are disconnected for this. A new connection is only possible when the new certificate is trusted.



Information:

- Since a private key may be transferred when this method is called, the call is only possible if an encrypted connection exists between the bus controller and the OPC UA client.
- UaExpert makes it easy to update certificates in the GDS Push View. It is thus not required to work directly with the method.
- Certificates derived from other certificates can only be installed if all higher-level certificates were already installed (see [8.1.3 "Symmetric and asymmetric keys"](#)) so that the complete chain of trust can be checked.

For additional details about OPC UA method `UpdateCertificate`, see the OPC UA specification, Part 12.

8.4.2.1 Updating the self-signed certificate using UaExpert

UaExpert contains tools that make it easy to update certificates.

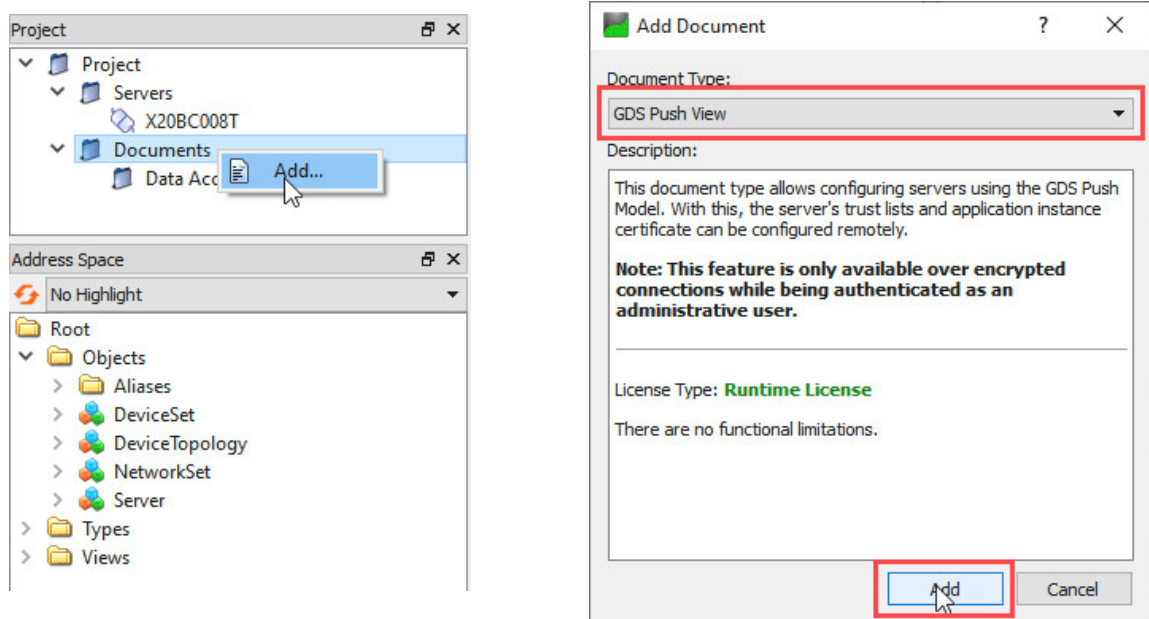


Information:

Since a private key is transferred with this method, an encrypted connection to the bus controller is required.

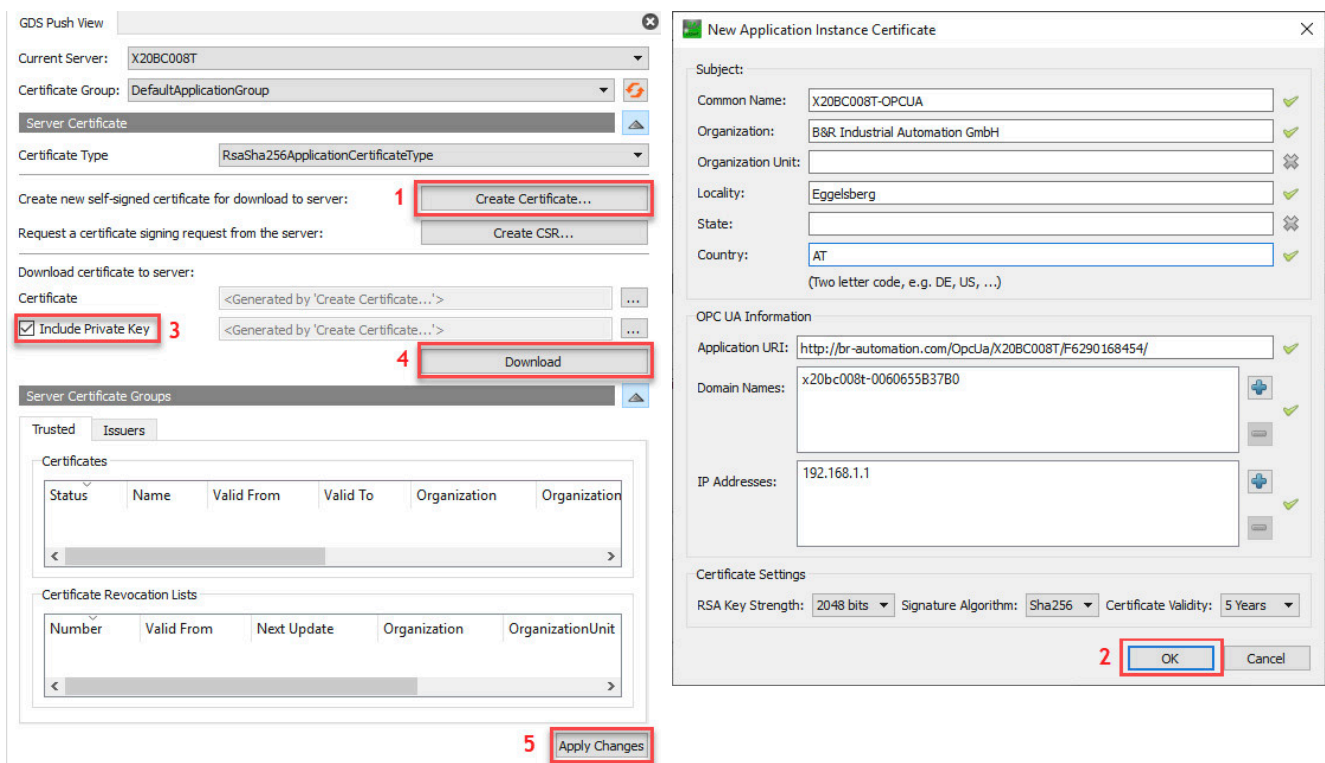
Perform the following steps in the UaExpert project window:

1. Open shortcut menus of Documents.
2. Call function Add.
3. Select document type "GDS Push View" in the following dialog box.
4. Continue with Add.



✓ A dialog box for creating and transferring the certificate is displayed.

Create and transfer certificate



1. Start creating the certificate with Create Certificate.
2. In the following dialog box, enter the data required for the certificate and confirm with OK.

**Information:**

Entering the IP address is necessary if the IP address is statically assigned and clients access the bus controller using the IP address (for example via URL `opc.tcp://192.168.1.1:4840`). If the IP address was obtained over a DHCP server, it does not make sense to enter an IP address in the certificate since this is usually assigned dynamically and may change.

3. Select option "Include private key" since the private key is transferred for the update.
 4. With Download the certificate created is transferred to the bus controller. The following query whether issuer certificates should be specified, can be confirmed with "No".
 5. Accept the new certificate with ApplyChanges.
- ✓ All connected clients are disconnected for this. A new connection is only possible when the new certificate is trusted.

9 Diagnostics

Occurring malfunctions or unexpected or undesired behavior of the bus controller can have many causes. Especially when used in larger networks with network infrastructures from different manufacturers, it can be difficult to localize possible error sources. This chapter serves as assistance during diagnostics of malfunctions and localization of causes. It describes context-related errors and shows possible causes and their solutions. These include:

- [Errors related to addressing](#)
- [Errors related to data transfer](#)
- [Errors related to time synchronization](#)
- [Errors related to cybersecurity](#)

9.1 Addressing

No.	Error characteristics	Possible cause	Solution	See
1	Cannot connect to hostname in factory state.	Unknown hostname	The bus controller is accessible by default under mDNS hostname "x20bc008t-<MAC address>.local" ¹⁾ .	- 3.2 "Establishing a connection" - 6.1.2 "General network configuration"
2	Cannot connect to IP address in factory state.	Unknown IP address	• DHCP server available in the network²⁾: Contact the responsible network administrator for the IP address assigned to the bus controller.	- 7.1 "Port status"
			• DHCP server not available in the network: Assign the static IP address 192.168.1.1 to the bus controller for operation with the node number switch (adjusting the node number switch for 1 second) and then perform a user-defined configuration via the OPC UA information model.	- 2.2.2 "Number switches" - 2.3 "Setting an IP address" - 6.1.2 "General network configuration"
			If the bus controller is directly connected to another switch that is already reachable via a hostname or an IP address, the searched IP address can be read out from the port to which the bus controller is connected. Root/Objects/DeviceSet/X20BC008T/Status/SwitchPorts/ETHx/LinkPartner/ManagementAddress	
3	In a network with a DHCP server, it is not possible to connect to the bus controller via an IP address after a static IP address has been set by configuration via the OPC UA information model.	There is an IP address conflict with another device on the network that has dynamically obtained the same address from the DHCP server.	Check whether the statically assigned IP address is located outside the range managed by the DHCP server. • Outside the range: Check all devices with static address settings for address conflicts. • Inside the range: Restore default address 192.168.1.1 and configure a static IP address that is located outside the range managed by the DHCP server.	- 2.2.2 "Number switches" - 2.3 "Setting an IP address" - 6.1.2 "General network configuration"
		During configuration via the OPC UA information model, a network mask of a subnet was assigned that cannot be accessed by the client.	Check the settings of the TCP/IP stack on the operating system of the client. • Within the same subnet: If the client is located on the same subnet, the configured network masks on the client and the bus controller must be identical. • Not within the same subnet: If the client is not located on the same subnet, the routing settings of the operating system of the client must be checked.	- 6.1.2 "General network configuration"

1) Information

- Addressing the device via the hostname mentioned above requires corresponding mDNS support on the operating system of the accessing client.
- For the MAC address to be used in the hostname of the bus controller, see the label attached to the side of the bus controller ("MAC1"), which also contains the serial number.

A user-defined hostname can be configured via the OPC UA information model.

- 2) The bus controller is delivered with an enabled DHCP client by default and an IP address must be assigned by a DHCP server in the network.

9.2 Data transfer

No.	Error characteristics	Possible cause	Solution	See
1	Complete communication failure between 2 or more devices connected to the bus controller ¹⁾ .	Ethernet autonegotiation between the bus controller and one or more connected devices failed.	<p>Check LinkStatus of the affected ports in the OPC UA information model: Root/Objects/DeviceSet/X20BC008T/Status/SwitchPorts/ETHx/LinkProperties/LinkStatus</p> <p>If this value is not set to UP, there is no link between the bus controller and the connected device at the network level.</p> <p>Checking the cable connection:</p> <ul style="list-style-type: none"> – Tightness of the connector – Max. length (100 m) of the cable not exceeded – Potential damage to the cable 	- 7.1 "Port status"
		Devices involved in the communication do not transmit Ethernet frames or faulty Ethernet frames that are detected and discarded by the bus controller.	<p>Check the error counter of the affected ports in the OPC UA information model: Root/Objects/DeviceSet/X20BC008T/Status/SwitchPorts/ETHx/FrameStatistics/</p> <ul style="list-style-type: none"> • RxFrameCount and TxFrameCount <p>If these counters have the value 0, no Ethernet frames nor faulty Ethernet frames are sent by the connected devices.</p> <p># FcsErrorFrameCount, GeneralRxErrorFrameCount, GeneralTxErrorFrameCount and SizeErrorFrameCount</p> <p>During error-free operation, these counters have the value 0. Values not equal to 0 indicate in most cases errors in network components of the connected devices or the bus controller itself.</p> <p>Check existing log outputs of the bus controller or the connected device. These can provide information about hardware problems.</p>	
		Ethernet frames that have a VLAN tag and/or multicast DMAC addresses were not configured on the bus controller.	Check the configuration for forwarding Ethernet frames with a VLAN tag and/or the multicast DMAC addresses in the configuration tool.	
2	Data is not received at a receiver connected to the bus controller at the expected time ²⁾ .	The link speed between the bus controller and the connected device does not meet the expectations (100 Mbit/s instead of 1 Gbit/s).	<p>Check nodes Speed and Duplex of the affected ports in the OPC UA information model. Root/Objects/DeviceSet/X20BC008T/Status/SwitchPorts/ETHx/LinkProperties/Speed or Duplex</p> <p>If the speed or duplex mode does not meet the expectations, check the following items:</p> <ul style="list-style-type: none"> – Does the connected device support the expected speed / duplex mode? – Was the correct cable type used? (Gigabit Ethernet requires at least Cat 5) – Checking the cable connection: <ul style="list-style-type: none"> • Tightness of the connector • Potential damage to the cable • Installation near potential sources of interference 	- 7.1 "Port status"
		The duplex mode between the bus controller and the connected device does not meet the expectations (half-duplex instead of full-duplex).		
		<p>The devices were not connected to the bus controller ports according to the system design.</p> <p>The transfer latencies do therefore not meet the expectations.</p>	<p>Check the nodes under LinkPartner of the affected ports in the OPC UA information model. Root/Objects/DeviceSet/X20BC008T/Status/SwitchPorts/ETHx/LinkPartner/</p>	

1) If messages from an OPC UA publisher are not received at an OPC UA subscriber, for example.

2) If an OPC UA subscriber detects Ethernet frames that were received late or failed.

9.3 Time synchronization

No.	Error characteristics	Possible cause	Solution	See
1	PTP time synchronization of devices connected to the bus controller is not working.	PTP time synchronization is not enabled on the bus controller.	Enable PTP time synchronization in the OPC UA information model ¹⁾ . • WallClock: Root/Objects/DeviceSet/X20BC008T/Configuration/TimeSynchronization/WallClock/TimeSyncProtocol • WorkingClock: Root/Objects/DeviceSet/X20BC008T/Configuration/TimeSynchronization/WorkingClock/TimeSyncProtocol	- 6.1.4 "Time synchronization"
		Time synchronization of an incorrect PTP domain was configured on the bus controller or connected device.	Configure an identical PTP domain on all participating devices ²⁾ . This setting can be made in the OPC UA information model on the bus controller. • WallClock: Root/Objects/DeviceSet/X20BC008T/Configuration/TimeSynchronization/WallClock/PTP/DomainNumber • WorkingClock: Root/Objects/DeviceSet/X20BC008T/Configuration/TimeSynchronization/WorkingClock/PTP/DomainNumber	- 6.1.4 "Time synchronization"
		There is no PTP grandmaster on the network.	If the bus controller should serve as a PTP grandmaster, option SlaveOnly must be disabled in the OPC UA information model. • WallClock: Root/Objects/DeviceSet/X20BC008T/Configuration/TimeSynchronization/WallClock/PTP/SlaveOnly • WorkingClock: Root/Objects/DeviceSet/X20BC008T/Configuration/TimeSynchronization/WorkingClock/PTP/SlaveOnly	- 6.1.4 "Time synchronization"
2	The bus controller was configured as a PTP grandmaster, but another network device was selected.	The priority of the PTP clock of the bus controller is too low compared to the selected network device.	Adjust setting Priority1 of the PTP clock in the OPC UA information model. The lower the value is set, the higher the priority is. • WallClock: Root/Objects/DeviceSet/X20BC008T/Configuration/TimeSynchronization/WallClock/PTP/Priority1 • WorkingClock: Root/Objects/DeviceSet/X20BC008T/Configuration/TimeSynchronization/WorkingClock/PTP/Priority1	- 6.1.4 "Time synchronization"
3	Data is not received at a receiver connected to the bus controller at the expected time ³⁾ .	The devices involved in the communication are not time synchronized according to the requirements or not time synchronized at all.	Check the correct time synchronization of the PTP domain of WorkingClock for of all network devices between the transmitter and receiver. The state of the time synchronization of the bus controller can be checked in the OPC UA information model: Root/Objects/DeviceSet/X20BC008T/Status/TimeSynchronization/WorkingClock/PTP	- 7.2 "Time synchronization"

1) Time synchronization is disabled on the bus controller by default.

2) By default, WallClock is synchronized via domain 0; WorkingClock is synchronized via domain 20.

3) If an OPC UA subscriber detects Ethernet frames that were received late or failed, for example.

9.4 Cybersecurity

No.	Error characteristics	Possible cause	Solution	See
1	Establishing a secure connection to the OPC UA server of the bus controller is rejected with information about an expired certificate.	The scope of the certificate used by the client is outside the current date or time of the bus controller ¹⁾ .	<ul style="list-style-type: none"> • Using NTP Ensure that at least one of the configured NTP servers is reachable and distributes the correct time. • Using PTP Ensure that the PTP grandmaster is active and distributes the correct time with the corresponding PTP domain (for WallClock). <p>Reset the security settings of the device if no source for WallClock is available or functional.</p> <p>A connection that is not encrypted can only be established in this state.</p>	<ul style="list-style-type: none"> - 3.5 "General network settings via OPC UA" - 5.4 "Time synchronization and time domains" - 6.1.4 "Time synchronization" - 7.1 "Port status" - 2.2.2 "Number switches"

- 1) The value of WallClock is crucial for this.
When establishing a secure connection using SSL/TLS, the certificates used are checked on both the client and the server.

10 Licenses

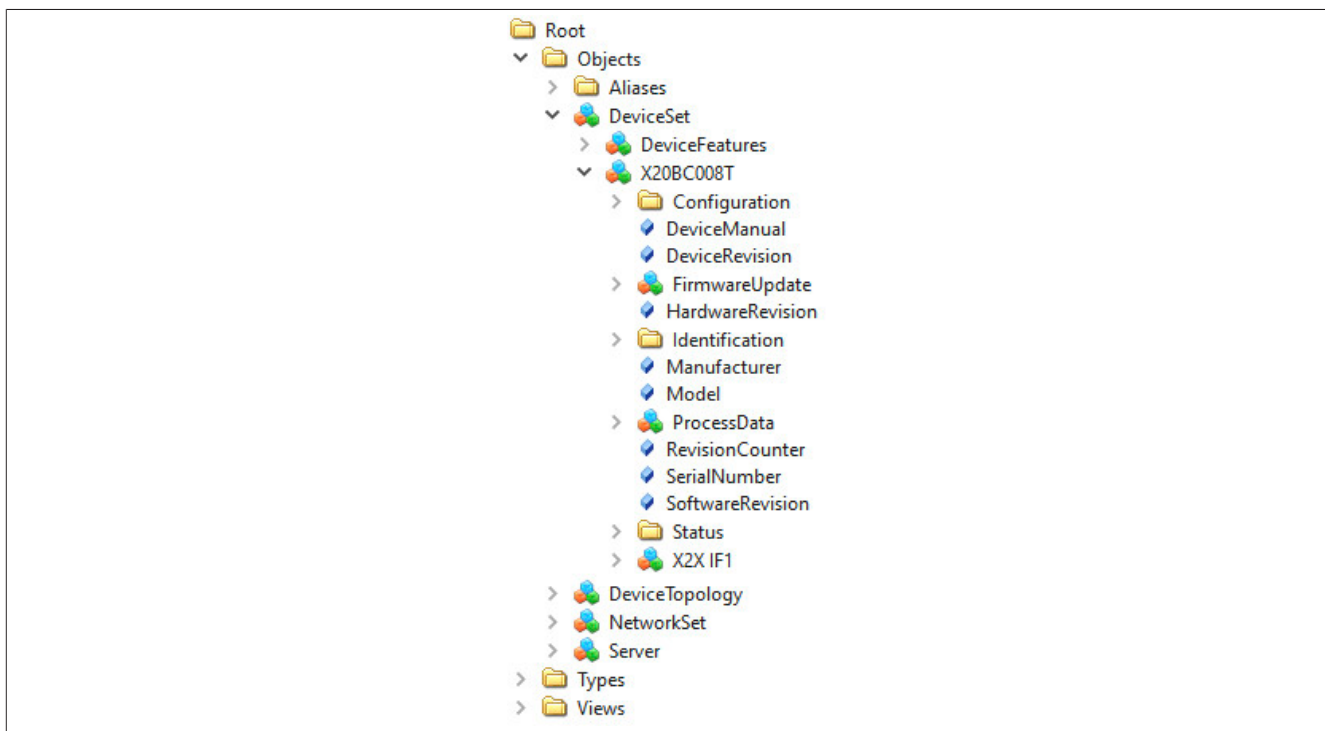
The license information can be retrieved using the firmware upgrades that can be downloaded from the B&R website (www.br-automation.com).

1. Download the firmware upgrade (ZIP file) of the module from the B&R website.
2. Unpack the firmware upgrade in a new folder.
ZIP file licenses.zip should then be available.
3. Unpack the ZIP file.
For technical reasons, the ZIP file may contain files with the same name. This should be taken into account when unpacking the ZIP file.
4. After unpacking, the license files can be viewed in folder ...\\licenses.

11 Appendix

11.1 OPC UA information model

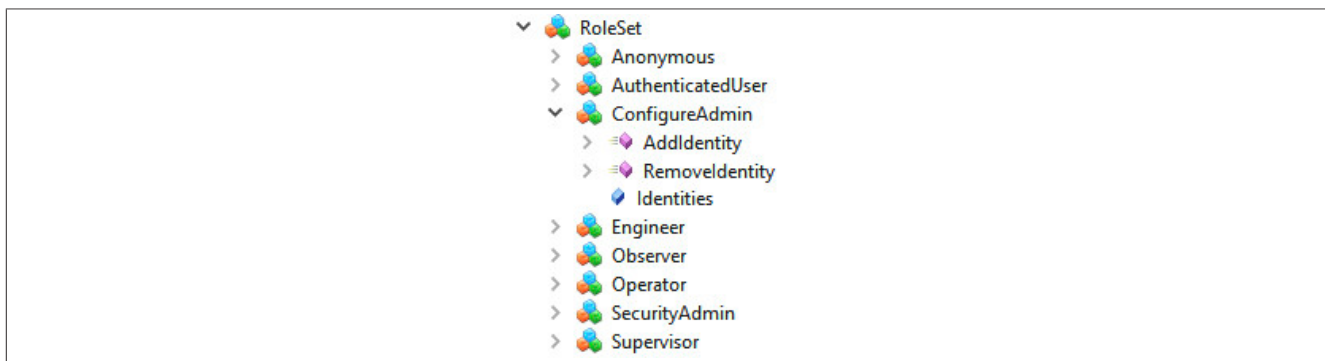
The bus controller provides access to the configuration and data of the I/O modules of the bus controller via the OPC UA information model. This information model also enables OPC UA clients to access existing data.



All nodes that are available for the bus controller are subordinated to main node Root/Objects/DeviceSet/X20BC008T via hierarchical references. These include nodes for configuration and access to I/O module process data.

11.1.1 User management

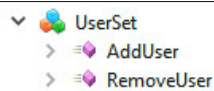
Access to the bus controller is restricted to authorized users during normal operation. Users have different rights according to the roles assigned to them. The user and role management required for this is carried out via the OPC UA information model.



The bus controller comes with predefined, standardized roles. The roles do not differ structurally in the information model, but all have the same methods and attributes. Role ConfigureAdmin, which is responsible for administration tasks, is shown here as an example.

Position of the data in the information model: Root/Objects/Server/ServerCapabilities/RoleSet

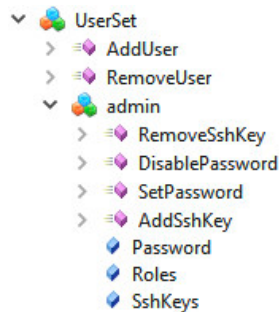
Node name	Description
AddIdentity	Adds a user to this role
RemoveIdentity	Removes a user from this role
Identities	List of all users in this role



The bus controller comes without predefined users in the factory setting. These can be freely assigned, except for a few reserved names such as "root". They will then appear below object UserSet.

Position of the data in the information model: Root/Objects/Server/ServerCapabilities/UserSet

Node name	Description
AddUser	Adds a user
RemoveUser	Removes a user



Created users do not differ structurally in the OPC UA information model, but have the same methods and attributes. Frequently used user "admin" is shown here as an example.

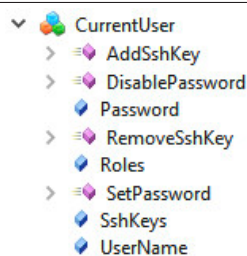
Users can authenticate themselves on the bus controller in different ways. Access via OPC UA is secured by passwords, whereas access via NETCONF is secured via SSH keys. It is permitted to enable both types at the same time.

A user can have one password at the most. It makes sense to choose a password that is complex enough. The bus controller does not check the set password, so the empty password "" is also possible.

A user can have any number of SSH keys. This is useful if access to the bus controller from different devices is desired. Unlike passwords, SSH keys are fundamentally secure and cannot be guessed. The bus controller permits the use of SSH keys only for NETCONF. The OPC UA standard does not support SSH.

Position of the data in the information model: Root/Objects/Server/ServerCapabilities/UserSet

Description	Node name
Roles	List of all roles held by the user
Password	Displays whether password authentication is active for this user
SetPassword	Sets a password
DisablePassword	Deletes the password and disables password authentication for this user
SshKeys	List of public SSH keys
AddSshKey	Adds an SSH public key
RemoveSshKey	Removes a public SSH key



Access to the general user and role administration is restricted to privileged users. However, each user has the necessary permissions to change his or her own password, for example. The current user of a session is represented separately in the information model. This dynamically mirrors the attributes and methods of a user who is also accessible via the general user management system.

Position of the data in the information model: Root/Objects/Server/ServerCapabilities/CurrentUser

Appendix

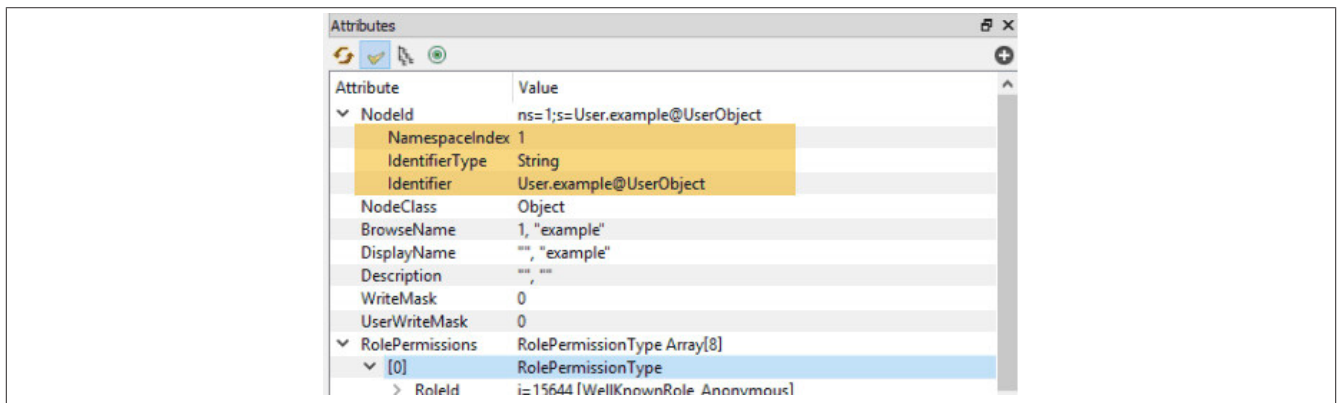
Node name	Description
Roles	List of all roles held by the user of this session
Password	Displays whether password authentication is active for this user
SetPassword	Sets a password
DisablePassword	Deletes the password and disables password authentication for this user
SshKeys	List of public SSH keys
AddSshKey	Adds an SSH public key
RemoveSshKey	Removes a public SSH key
UserName	Name of the current user of the session

11.1.1.1 Deleting a created user

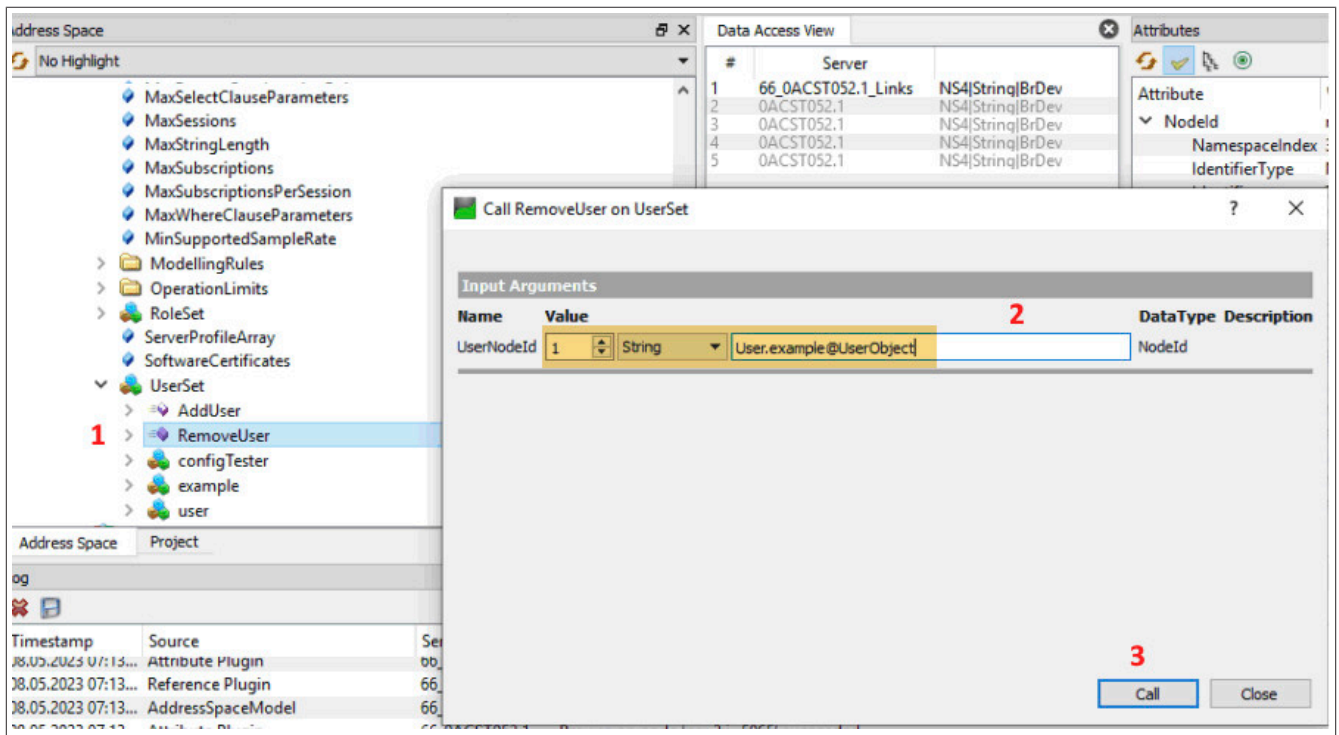
The following must be taken into account to delete users that have already been created:

- Only a user with SecurityAdmin rights can delete a user.
- Users cannot delete themselves.

User "Example" should be deleted in this example.



- 1) Method 11.1.1 "RemoveUser" must be called first.
- 2) NamespaceIndex and Identifier of the user are specified as input arguments.
- 3) Finally, the user is deleted by clicking on "Call".



11.1.2 Firmware update

The firmware update functionality is provided to the bus controller in the OPC UA information model by node `Root/Objects/DeviceSet/X20BC008T/FirmwareUpdate`. The following table describes the hierarchy of all subnodes and their meaning:

Node name	Data type	Description
DefaultInstanceBrowseName	QualifiedName	Name of the object used. Defined default name = "SoftwareUpdate"
Installation		
CurrentState	LocalizedText	Installation status readable by the user Possible values Idle Installing Error
Id	NodeId	Machine-readable installation status Possible values 271 Idle object 273 Installation object 275 Error object
InstallSoftwarePackage		Starts the installation of the previously loaded firmware package
InputArguments	String ManufacturerUri String SoftwareRevision String[] PatchIdentifiers ByteString Hash	ManufacturerUri and SoftwareRevision. Used to identify the firmware package to be installed. Arguments PatchIdentifiers and Hash have no function and must be left empty. Information: If a firmware installation that overwrites an identical, already installed version should be forced, string "force" must be assigned to argument PatchIdentifiers[0].
PercentComplete	Byte	Displays the installation progress. Information: This method is not suitable for detecting a completed installation.
Resume		Resets the installation status from "Error" back to "Idle"
Loading		
CurrentVersion		Shows properties of the active firmware
Manufacturer	LocalizedText	Manufacturer
ManufacturerUri	String	Manufacturer URI
SoftwareRevision	String	Version of the firmware package
ErrorMessage	LocalizedText	User information for file transfer - see Error messages .
FileTransfer		Provides information about the file transfer of the firmware package
ClientProcessingTimeout		Time in seconds after which the server ends the transfer if the client stops making method calls required for the transfer.
CloseAndCommit		Ends the file transfer
GenerateFileForRead		Not supported
GenerateFileForWrite		Generates a FileType instance that is used for the file transfer.
TransferState		Transfer state of the object
GetUpdateBehavior		Shows update properties of the loaded firmware
InputArguments	String ManufacturerUri String SoftwareRevision String[] PatchIdentifiers	ManufacturerUri and SoftwareRevision. Used to identify the previously loaded firmware package. Argument PatchIdentifiers has no function and must be left empty.
OutputArguments	UInt32	Describes how the bus controller can perform an update. Possible values 4 RequiresPowerCycle
PendingVersion		Shows properties of the loaded firmware ¹⁾
Manufacturer	LocalizedText	Manufacturer
ManufacturerUri	String	Manufacturer URI
SoftwareRevision	String	Version of the firmware package
PowerCycle		
CurrentState	LocalizedText	User-readable reboot status Possible values NotWaitingForPowerCycle WaitingForPowerCycle
Id	NodeId	Machine-readable reboot status Possible values 299 Object NotWaitingForPowerCycle 301 Object WaitingForPowerCycle
UpdateStatus	LocalizedText	User information and feedback for the entire update procedure, see Update status .

1) Only if the corresponding firmware update file has already been transferred to the target device and is ready for installation.

Error messages

No.	Text	Explanation
0	[ERROR] File invalid or not loaded	Displayed when node <code>Root/Objects/DeviceSet/X20BC008T/FirmwareUpdate/Loading/PendingVersion</code> is read, but the firmware package is invalid or missing.

Update status

No.	Text	Explanation
0	[ERROR] Requested version not present or file invalid	Input parameters of method <code>GetUpdateBehavior</code> or <code>InstallSoftwarePackage</code> do not match the loaded firmware package. The requested version does not exist or the firmware package is invalid.
1	[ERROR] Installation failed. See <code>FirmwareInstall.log</code> in system dump archive.	The installation of the firmware package triggered by method <code>InstallSoftwarePackage</code> was started, but failed. For additional information, see object "SystemDump" in file "FirmwareInstall.log".
2	[ERROR] Action not allowed in current state	The method call was denied since it is not permitted in the current state.
3	[INFO] Installation successful, reboot required	The bus controller requires a reboot to enable the installed firmware. This can be done by calling method <code>Root/Objects/DeviceSet/X20BC008T/Configuration/Control/Reboot</code> or by switching the power supply off and on.