ROBOTICS

Application manual

Embedded OPC UA Server

Trace back information:
Workspace 24C version a15
Checked in 2024-10-21
Skribenta version 5.5.019

Application manual
Embedded OPC UA Server

Document ID: 3HAC085436-001
Revision: E

The information in this manual is subject to change without notice and should not be construed as a commitment by ABB. ABB assumes no responsibility for any errors that may appear in this manual.

Except as may be expressly stated anywhere in this manual, nothing herein shall be construed as any kind of guarantee or warranty by ABB for losses, damage to persons or property, fitness for a specific purpose or the like.

In no event shall ABB be liable for incidental or consequential damages arising from use of this manual and products described herein.

This manual and parts thereof must not be reproduced or copied without ABB's written permission.

Keep for future reference.

Additional copies of this manual may be obtained from ABB.

Original instructions.

Table of contents

This page is intentionally left blank

Overview of this manual

About this manual

This manual contains instructions for daily operation of the Embedded OPC UA Server in RobotWare.

Usage

This manual should be used during operation, installation and configuration of Embedded OPC UA Server.

Who should read this manual?

This manual is intended for:

- Users of the product Embedded OPC UA Server in RobotWare.

Prerequisites

The reader should.

- use the manual as an online help and
- have RobotWare 7.8 or newer installed.

References

Reference	Document ID
Technical reference manual - System parameters	*3HAC065041-001*
Technical reference manual - RAPID kernel	*3HAC050946-001*
Operating manual - RobotStudio	*3HAC032104-001*
Operating manual - Integrator's guide OmniCore	*3HAC065037-001*

Revisions

Version	Description
A	First edition.
B	Released with RW 7.10. Following are the updates: • Added the section *Maximum sessions on page 14*. • Updated the NOTE regarding RAPID persistent variables in the sections *ABB information model on page 38* and *Appendix B - ABB Robotics OPC UA proprietary information model on page 58*.
C	Released with RW 7.12. Following are the updates: • Added the section *Appendix A - Robotics companion specification on page 49*. • Updated the section *Appendix B - ABB Robotics OPC UA proprietary information model on page 58*.
D	Released with RW 7.13. Following are the updates: • Minor updates in the section *Appendix A - Robotics companion specification on page 49*.

Continues on next page

Continued

Version	Description
E	Released with RW 7.16. Following are the updates: • Updated the section *Connecting to OPC UA Server on page 16*.

Product documentation

Categories for user documentation from ABB Robotics

The user documentation from ABB Robotics is divided into a number of categories. This listing is based on the type of information in the documents, regardless of whether the products are standard or optional.

Tip

All documents can be found via myABB Business Portal, *www.abb.com/myABB*.

Product manuals

Manipulators, controllers, DressPack, and most other hardware is delivered with a **Product manual** that generally contains:

- Safety information.
- Installation and commissioning (descriptions of mechanical installation or electrical connections).
- Maintenance (descriptions of all required preventive maintenance procedures including intervals and expected life time of parts).
- Repair (descriptions of all recommended repair procedures including spare parts).
- Calibration.
- Troubleshooting.
- Decommissioning.
- Reference information (safety standards, unit conversions, screw joints, lists of tools).
- Spare parts list with corresponding figures (or references to separate spare parts lists).
- References to circuit diagrams.

Technical reference manuals

The technical reference manuals describe reference information for robotics products, for example lubrication, the RAPID language, and system parameters.

Application manuals

Specific applications (for example software or hardware options) are described in **Application manuals**. An application manual can describe one or several applications.

An application manual generally contains information about:

- The purpose of the application (what it does and when it is useful).
- What is included (for example cables, I/O boards, RAPID instructions, system parameters, software).
- How to install included or required hardware.
- How to use the application.

Continues on next page

Continued

- Examples of how to use the application.

Operating manuals

The operating manuals describe hands-on handling of the products. The manuals are aimed at those having first-hand operational contact with the product, that is production cell operators, programmers, and troubleshooters.

1 Getting started

1.1 Introduction

Overview

The Embedded OPC UA Server provides OPC UA server capabilities integrated into RobotWare. Compared to the IoT Gateway, which also provides a OPC UA server for OmniCore controllers, the Embedded OPC UA Server does not require any additional hardware.

1.2 RobotWare software requirements

Requirements

- RobotWare 7.8 and newer
- “3154-1 IoT Data Gateway” option

1.3 Product features

Overview

Embedded OPC UA Server provide OPC UA server capabilities for the Omnicore generation of robot controllers. It implements the functionality of the UA Address Space Model 1.04 Specification. It is a UA server which enables UA clients to browse the address space, create subscriptions and monitor items, and read and write data.

Certificates

Certificates are used to establish secure communication between the OPC UA Client and Embedded OPC UA Server.

Endpoints

OPC UA Clients can connect to the OPC UA Server using a URL with the following format:

`opc.tcp://HOSTNAME:PORT NUMBER/SERVERNAME`

For HOSTNAME use the WAN IP address of the controller. To get the actual server endpoint URL from the log, see *Connecting to OPC UA Server on page 16*.

User authentication

OPC UA Server supports the following user authentication modes:

- UserName: A user identified by user name and password.

Client authentication

The OPC UA Server authenticates (Identifies) OPC UA client using **Client application instance certificate**. For more details, see *Certificate management on page 24*.

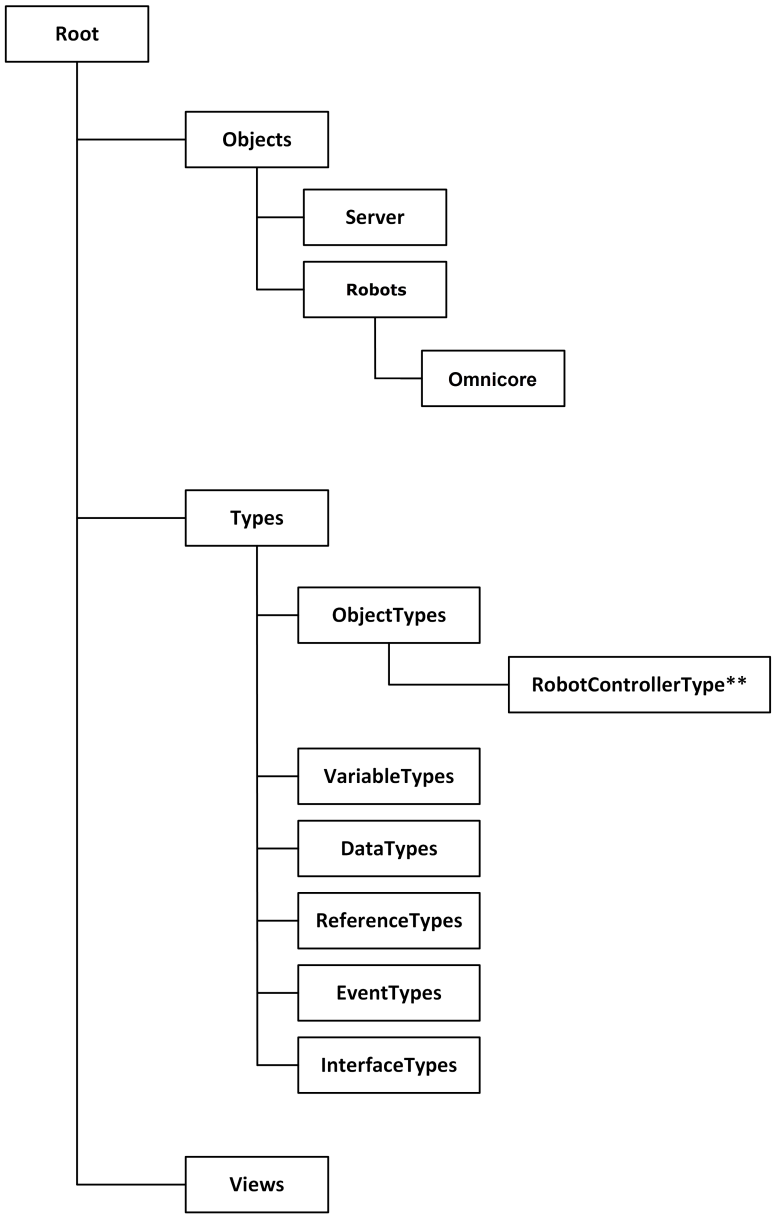
Security

OPC UA Server supports OPC UA standard security modes and policies.

Continues on next page

Address Space

The server's address space represents its contents as a set of Nodes connected by References. The address space begins with the top node **Root**, which Organizes Objects, Types and Views.



***Top level object type for an ABB Robotics Controller.**

xx2200001286

Note

For more information on address space, see section *Address space on page 38*.

Maximum sessions

The maximum sessions allowed in Embedded OPC UA Server is 10.

1.4 Cybersecurity

Overview

This product is designed to be connected to and to communicate information and data via a network interface. It is your sole responsibility to provide, and continuously ensure, a secure connection between the product and to your network or any other network (as the case may be).

You shall establish and maintain any appropriate measures (such as, but not limited to, the installation of firewalls, application of authentication measures, encryption of data, installation of anti-virus programs, etc) to protect the product, the network, its system and the interface against any kind of security breaches, unauthorized access, interference, intrusion, leakage and/or theft of data or information. ABB Ltd and its entities are not liable for damage and/or loss related to such security breaches, any unauthorized access, interference, intrusion, leakage and/or theft of data or information.

For more information, see the section Cyber security in *Operating manual - Integrator's guide OmniCore*.

1.5 Connecting to OPC UA Server

Procedure

In a secure connection, the OPC UA server and OPC UA client must trust each other to protect the data exchange between the OPC UA server and OPC UA client.

To establish a secure connection between an OPC UA server and OPC UA client, perform the following tasks:

1 Enable OPC UA Server and firewall settings.

For more details, see *Configuration on page 19*.

Make sure firewall settings allow incoming connections to the port number specified in the endpoint URL (4840 by default).

2 Configuring the OPC UA Client

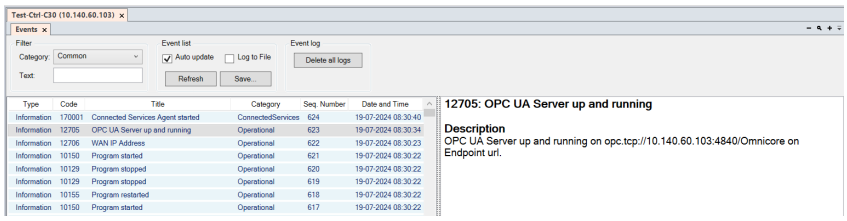
The following connection information must be configured in the OPC UA Client:

- Endpoint URL:

  OPC UA Server's endpoint URL has the format `opc.tcp://{WAN IP Address}:4840/Omnicore`

  For example, if the IP address on WAN port is 10.140.60.103 then the endpoint URL will be `opc.tcp://10.140.60.103:4840/Omnicore`

  The endpoint URL can also be seen in the event log-> common category as shown in the following figure.



xx2200001294

- Security Settings:

  Security Policy: Select one of the following OPC UA Server supported security policies:

  - Basic256Sha256
  - Aes128_Sha256_RsaOaep
  - Aes256_Sha256_RsaPss

  Message Security Mode (or) Security Policy: Select one of the following OPC UA Server supported security mode:

  - Sign
  - Sign and encrypt

  Security Message Encoding: Select **Binary** as it is supported by OPC UA Server.

- Authentication Settings:

  OPC UA Server support Username user token type (User Identity) only.

Continues on next page

User Name: A user is identified by username and password.

OPC UA Clients should provide controller user name and password as defined in RobotWare UAS are accepted by the Embedded OPC UA Server.

3 First attempt to connect to OPC UA Server.

During first attempt connection will fail. This is due to security reasons. OPC UA clients and servers may use certificates to make sure they communicate with an approved server or client, and the default security settings for the OPC UA Server is to require a trusted client certificate. In addition, firewall settings may also cause the connection to fail.

4 Trust the OPC UA Client Certificate.

When a client tries to connect for the first time, it will be rejected and its certificate will be placed in the Rejected list. This allows the administrator to review clients before allowing them to connect. For more details, see *Client certificates on page 30*.

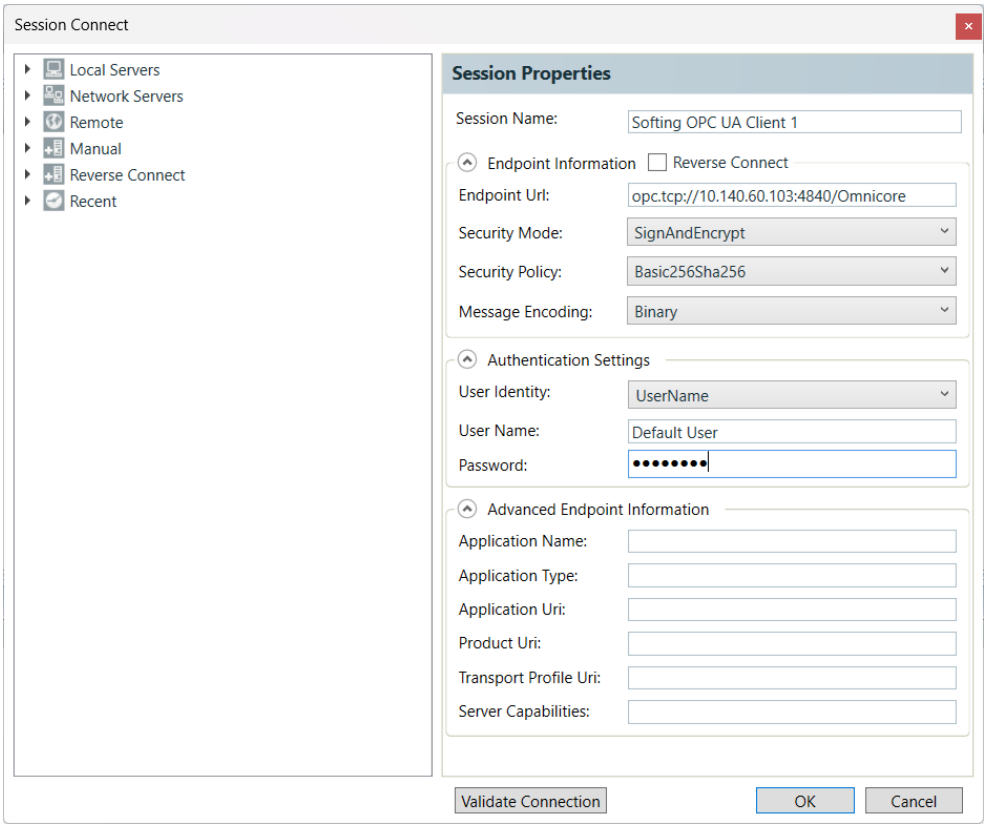
5 Trust the OPC UA Server Certificate.

OPC UA Clients should also trust the OPC UA Server certificate to establish secure communication.

6 Connect to the OPC UA Server.

The Client should be able to make successful connection with the Server.

The following image provides an example of the details in Softing's dataFEED OPC UA Client.



xx2200001295

Continues on next page

> **Note**
>
> Using management port apart from commissioning or service related activity may result in IP conflict in case of multiple controllers using management port. Hence avoid accessing OPCUA server through management port. If connection could not be established, refer *Troubleshooting on page 47*.

Virtual controller

Endpoint URL

Endpoint URL: `opc.tcp://10.140.60.103:4840/Omnicore`

Embedded OPC UA Server runs on TCP port 4880 on windows platform.

Limitation

Remote virtual controller: OPC UA client cannot connect to embedded OPC UA Server running in remote virtual controllers (VC running on different machine) .

2 Configuration

2.1 Introduction

Overview

This chapter describes how to configure the Embedded OPC UA Server using RobotStudio.

Configuring the Embedded OPC UA Server includes:

- Firewall configuration
- Configuring the OPC UA server itself
- Certificate management

> **Note**
>
> To connect RobotStudio to an OmniCore controller, follow the procedure in the **Add Controller** section in *Operating manual - RobotStudio*.

2.2 Prerequisites

Overview

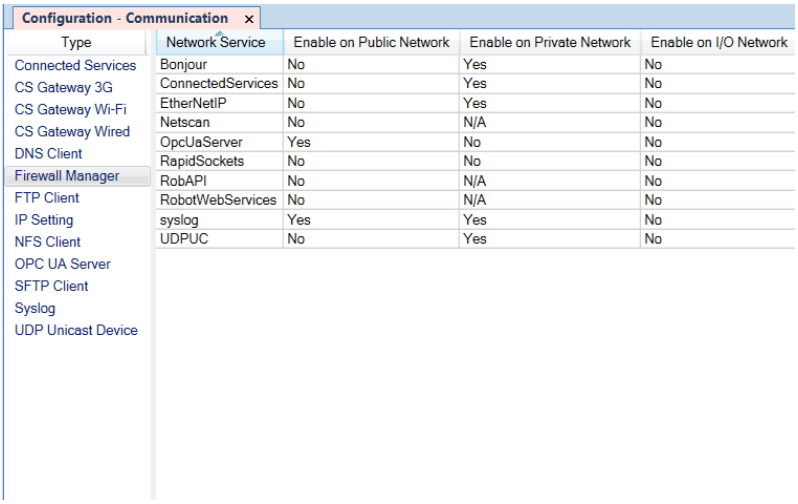
Following are the prerequisites for configuring the Embedded OPC UA Server

- RobotWare 7.8 or newer
- **3154-1 IoT Data Gateway** option

2.3 Firewall configuration

Overview

Use the Firewall Manager to enable Embedded OPC UA Server to communicate on the Public Network as shown in the following image. Although technically possible, it is not normal to use OPC UA on the Private or I/O network.



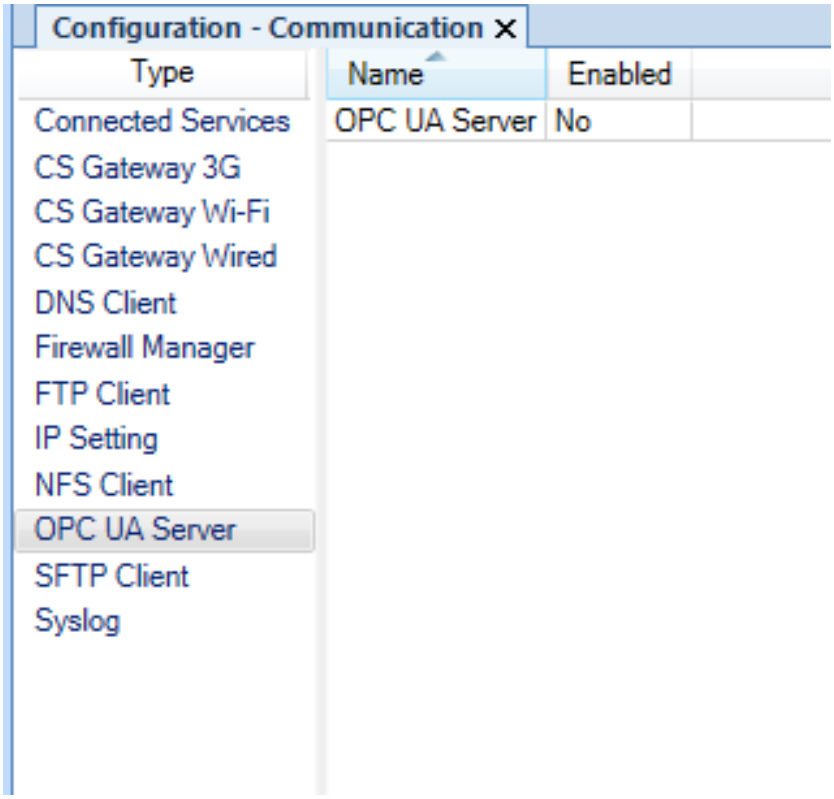
xx2200001234

2.4 Embedded OPC UA Server configuration

Procedure

Use the following procedure to configure Embedded OPC UA Server:

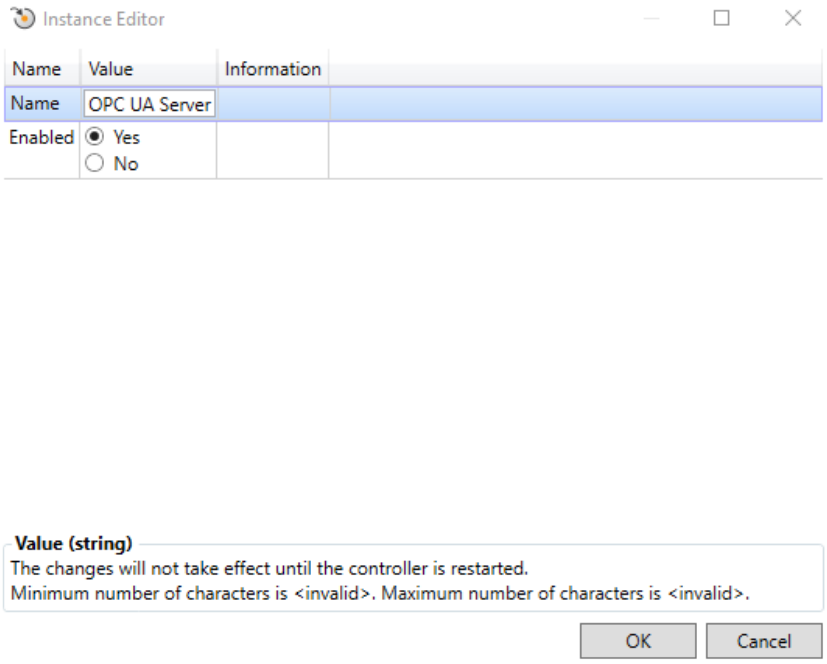
1 In the Communication configuration domain, select the **OPC UA Server** type.
The default value for OPC UA Server Enabled is **No**.



xx2200001235

Continues on next page

2 Click on the **Enabled** field and change the value of **Enabled** to **Yes**.



xx2200001236

3 Click **OK**.

The settings are saved. Restart the controller for the changes to take effect.

2.5 Certificate management

2.5.1 Overview

Introduction

OPC UA uses certificates to secure the communication between a client and a server. Both OPC UA clients and servers have their own certificates called Application Instance Certificates. For simplicity the terms “Server Certificate” and “Client Certificate” are used in this document to denote the Application Instance Certificate for a server and a client respectively.

See *OPC 10000-2 Unified Architecture Part 2 Security Model Certificate management* for a detailed description of OPC UA certificate management.

2.5.2 Server certificate

When the Embedded OPC UA Server runs for first time, it creates a certificate store(opcua_store) containing a self-signed Server Certificate. If required by local policy, an administrator can import a (CA rooted) custom certificate to replace the automatically created self-signed certificate .See the Import certificate chapter for details

2.5.3 Establishing a secure connection between client and server

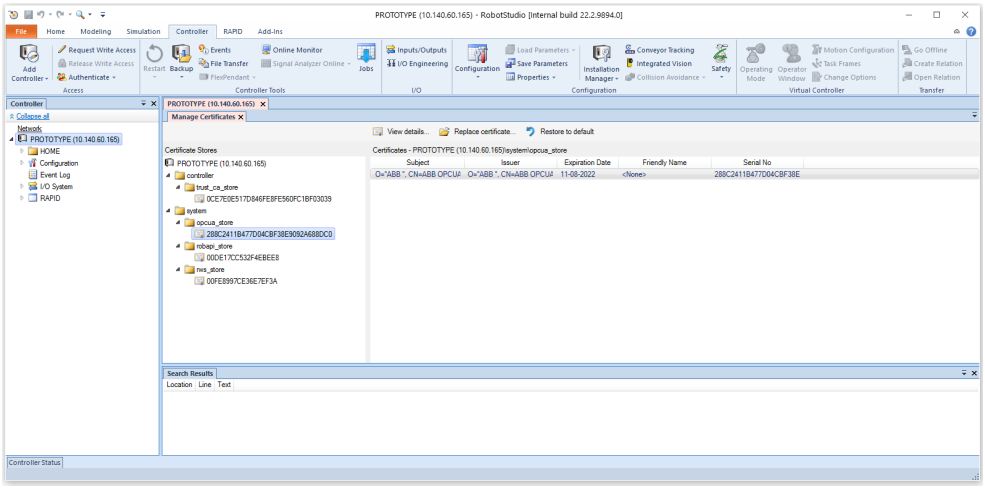
OPC UA Client initiate the connection with server and exchange certificates. Certificate validation performed both the ends.

The OPC UA server accepts the client certificate but does not initially trust it, placing it into the OPC UA server's rejected certs folder. Client certificates need to be trusted manually. Please refer Trust client certificate(s) chapter.

To complete a secure connection, a manual process to trust the client certificate must be executed on the OPC UA server. OPC UA Client also need to trust the application instance certificate.

2.5.4 Server application instance certificates

To access Server Application instance certificate, on the controller tab left panel right click on the controller, select **Properties** and click on **Manage Certificates**. The **Manage Certificate** page is displayed.



xx2200001241

IP address in the endpoint URL shall match the IP in "Subject Alternate Name" field of certificate. The OPC UA client may validate whether the endpoint URL matches the information in the certificate.

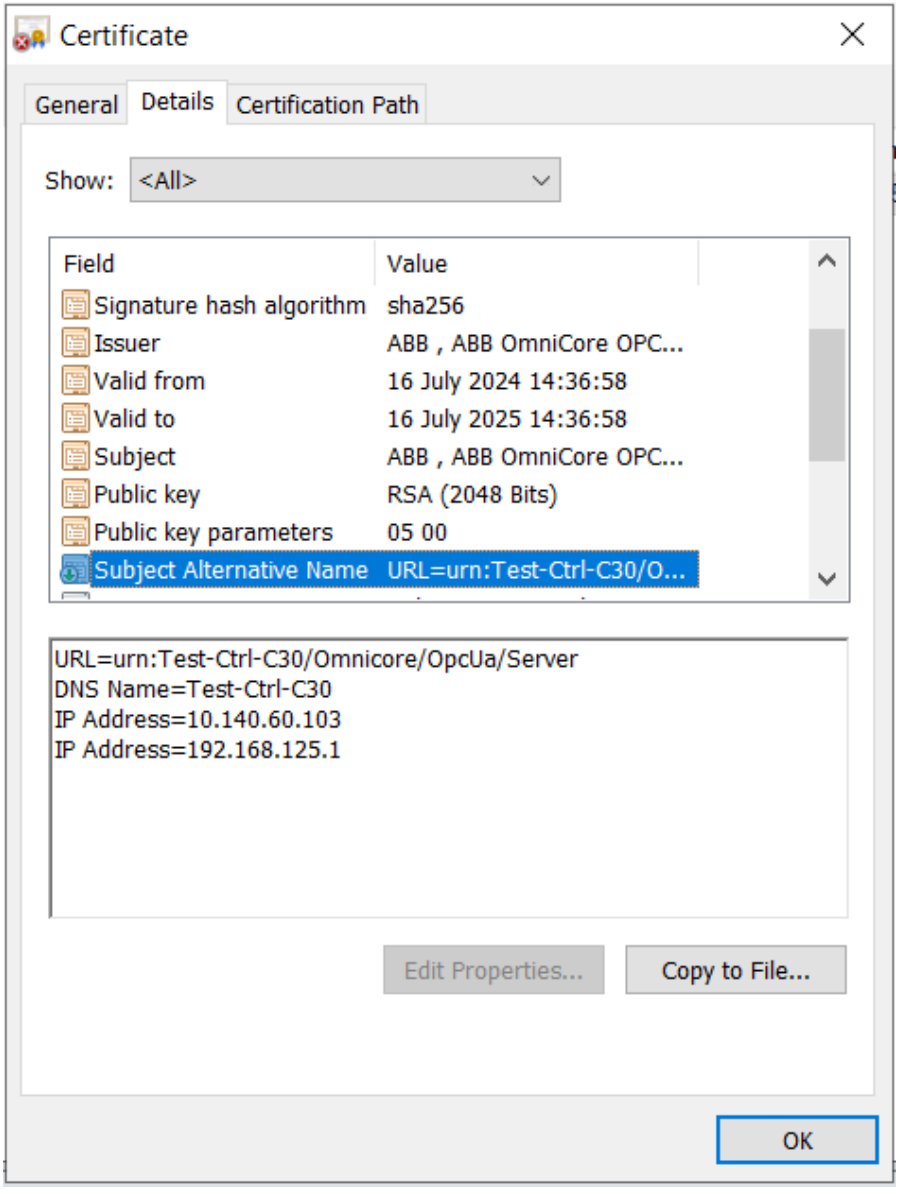
Endpoint URL syntax is `opc.tcp://{Controller WAN IP}:4840/Omnicore`. For example, `opc.tcp://10.140.60.165:4840/Omnicore`

Application instance certificate's subject alternative name should contain the following values.

- URL=`urn:vxTarget/Omnicore/OpcUa/Server`
- IP Address=`{Controller WAN IP}`

Continues on next page

2.5.4 Server application instance certificates
Continued



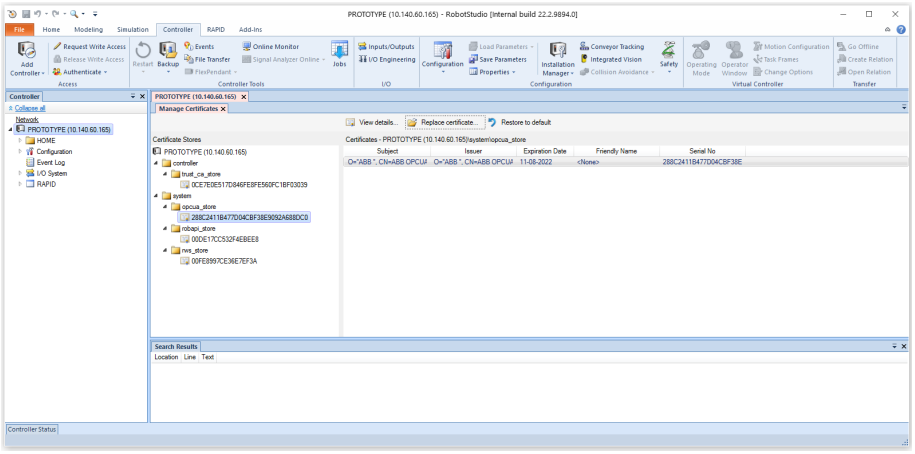
xx2200001242

Whenever controller WAN IP address changes, you can perform one of the listed action to make the OPC UA Server running:

- Update the server instance certificate in OPCUA certificate store with new controller WAN IP mentioned in "Subject Alternate Name".

Continues on next page

Click on **Replace certificate** in Manage Certificates window.



xx2200001243

- Delete the existing server instance certificate in OPCUA certificate store using the Restore to default option.
- Restart the controller. OPC UA Server generates a new self-signed certificate with new WAN IP.

2.5.5 Client certificates

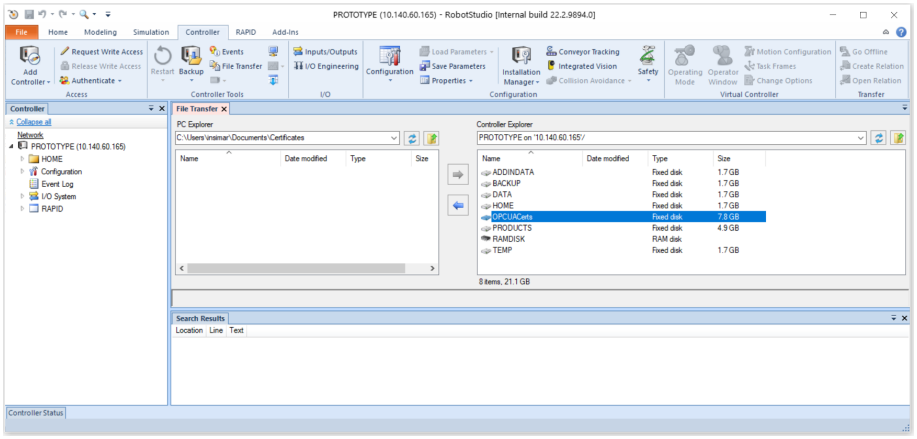
Access the OPCUACerts disk

Use the following procedure to access the OPCUACerts disk:

1. Open RobotStudio.
2. Click on Controller -> File Transfer.

   The **Controller Explorer** window is displayed.

   The **OPCUACerts** disk is available at the root.

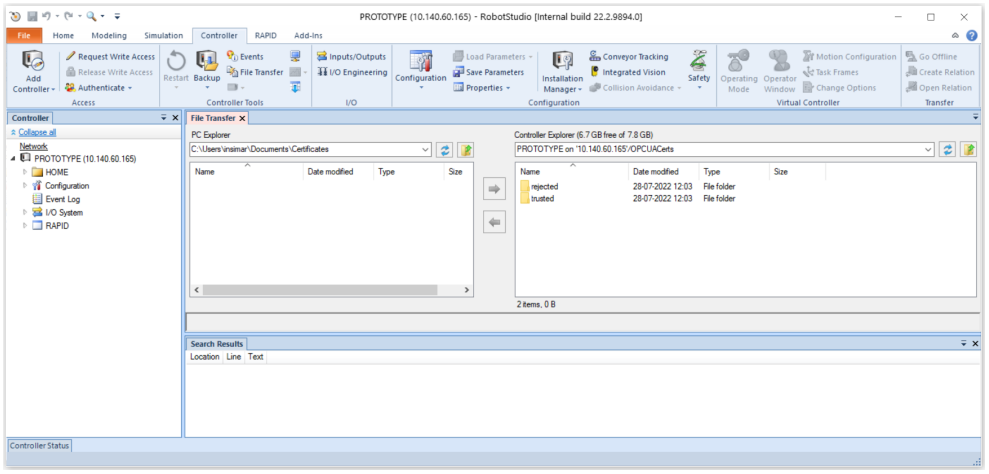


xx2200001246

> **Note**
>
> The following UAS grants are required to access the OPCUACerts disk:
>
> - System administration
> - Modify network security properties

OPCUACerts disk contains the following two folders

- rejected
- trusted



xx2200001247

Continues on next page

The following operations can be performed on client certificates:

- Trust client certificate
- Reject client certificate
- Import certificate
- Delete certificate

Trust client certificate for RC

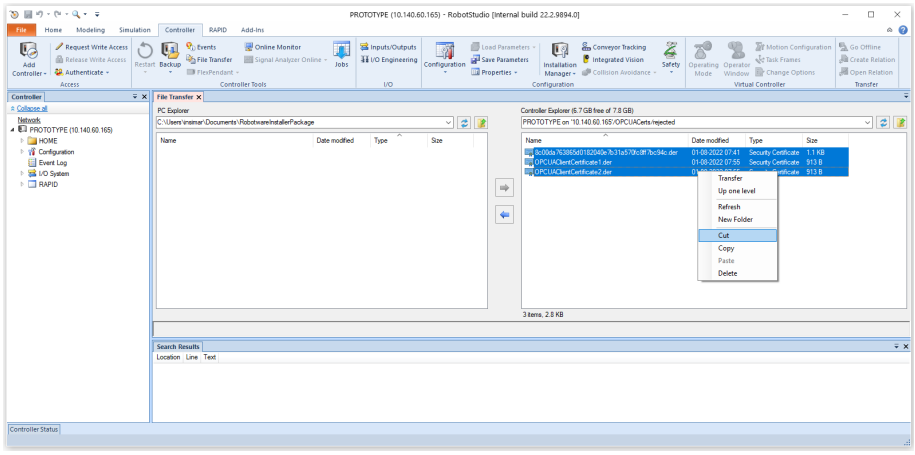
An administrator can move client certificates from the rejected folder to the trusted folder in real controller (RC).

> **Note**
>
> It is the responsibility of the administrator to review the certificates and ensure that they can be trusted.

Use the following procedure to move client certificates from the rejected folder to the trusted folder in real controller:

1. Open **RobotStudio** > **Controller** > **File Transfer**.

   The **Controller Explorer** window is displayed.
2. Navigate to the **...OPCUACerts/rejected** folder.
3. Select the required certificates, right click on it, and select **Cut**.



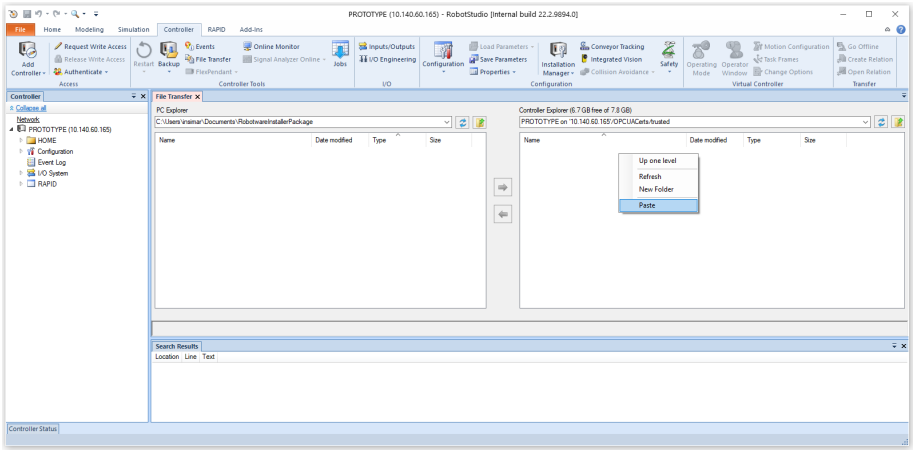
xx2200001248

4. Navigate to the **...OPCUACerts/trusted** folder.

Continues on next page

2.5.5 Client certificates
Continued

5 Right click and select **Paste**.



xx2200001249

The selected client certificates are moved from the rejected folder to the trusted folder.

Trust client certificate for VC

An administrator can move client certificates from the rejected folder to the trusted folder in virtual controller (VC).

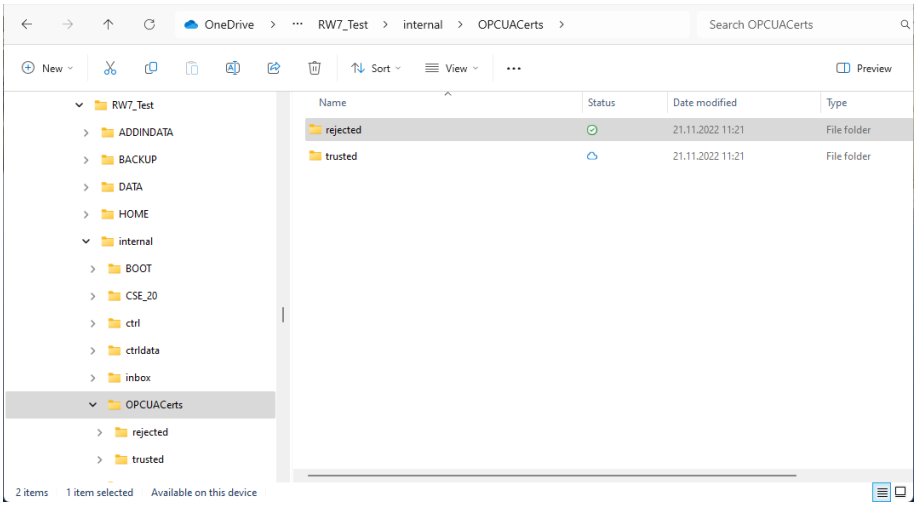
> **Note**
>
> It is the responsibility of the administrator to review the certificates and ensure that they can be trusted.

Use the following procedure to move client certificates from the rejected folder to the trusted folder in virtual controller:

1 Use the Windows File Explorer and navigate to the file system of VC.

2 Open the "...\internal\OPCUACerts" folder.

The "rejected" and "trusted" folders are displayed.



xx2400001106

Continues on next page

3 Navigate to the **...OPCUACerts/rejected** folder.
4 Select the required certificates, right click on it, and select **Cut**.
5 Navigate to the **...OPCUACerts/trusted** folder.
6 Right click and select **Paste**.

   The selected client certificates are moved from the rejected folder to the trusted folder.

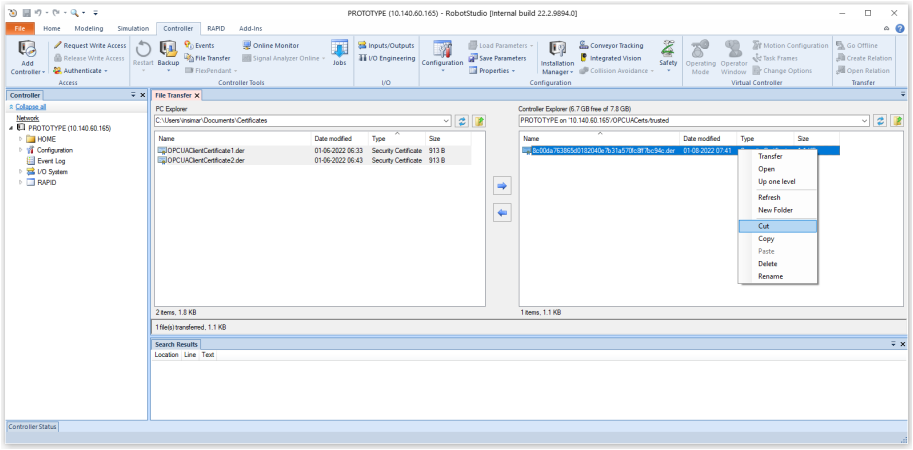
Reject client certificate for RC

Using this operation, administrator can reject the certificate(s) from the trusted list by moving the client certificate from trusted folder to rejected folder.

Use the following procedure to move client certificates from the trusted folder to rejected folder:

1 Open **RobotStudio** > **Controller** > **File Transfer**.

   The **Controller Explorer** window is displayed.
2 Navigate to the **...OPCUACerts/trusted** folder.
3 Select the required certificates, right click on it, and select **Cut**.



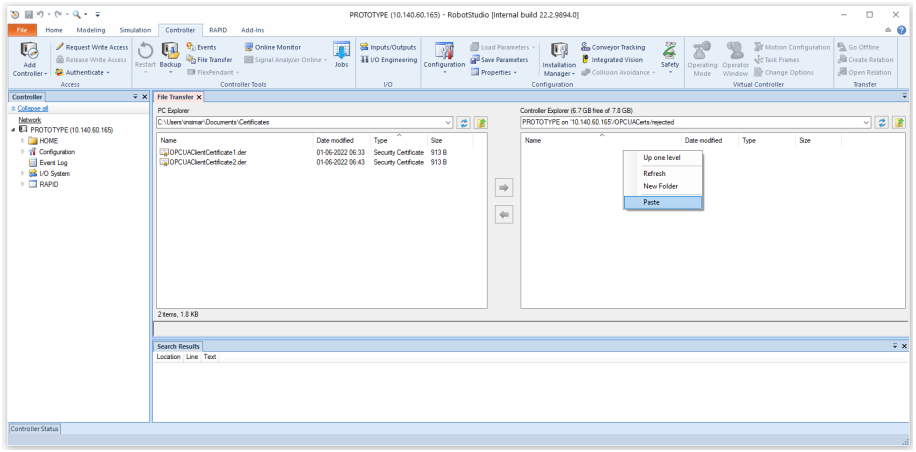
xx2200001250

4 Navigate to the **...OPCUACerts/rejected** folder.

Continues on next page

2.5.5 Client certificates
Continued

5 Right click and select **Paste**.



xx2200001251

The selected client certificates are moved from the trusted folder to the rejected folder.

Reject client certificate for VC

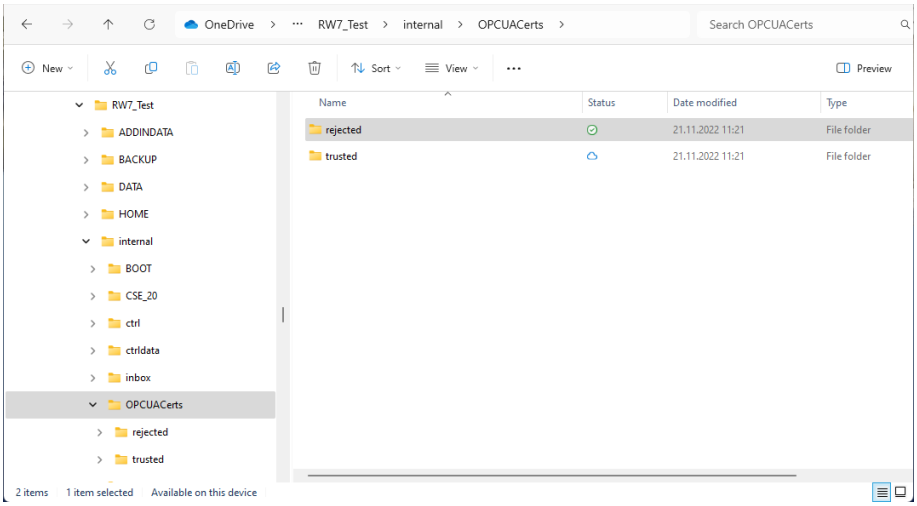
Using this operation, administrator can reject the certificate(s) from the trusted list by moving the client certificate from trusted folder to rejected folder in virtual controller.

Use the following procedure to move client certificates from the trusted folder to rejected folder:

1 Use the Windows File Explorer and navigate to the file system of VC.

2 Open the "...\internal\OPCUACerts" folder.

The "rejected" and "trusted" folders are displayed.



xx2400001106

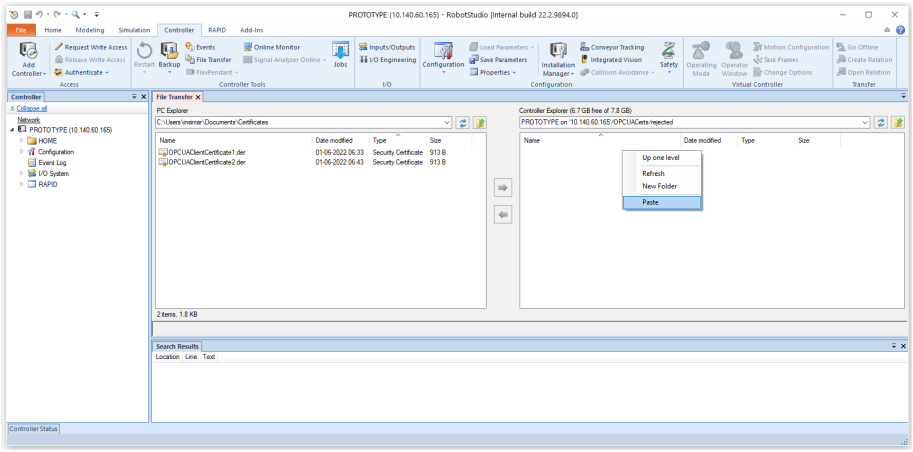
3 Navigate to the ...OPCUACerts/rejected folder.

4 Select the required certificates, right click on it, and select **Cut**.

5 Navigate to the **...OPCUACerts/rejected** folder.

Continues on next page

6 Right click and select **Paste**.



xx2200001251

The selected client certificates are moved from the trusted folder to the rejected folder.

Import certificate

Using this operation, administrator can import client certificate by transferring certificate files to the trusted folder.

> **Note**
>
> It is the responsibility of the administrator to review the certificates and ensure that they can be trusted.

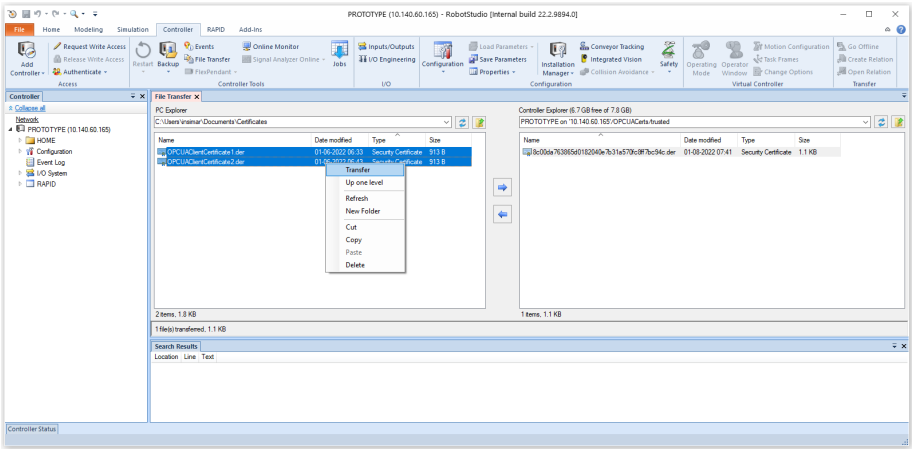
Use the following procedure to import client certificate to the trusted folder:

1 Open **RobotStudio** > **Controller** > **File Transfer**.

The **Controller Explorer** window is displayed.

2 Navigate to the **...Documents** > **Certificates** folder.

3 Select the required certificates, right click on it, and select **Transfer**.



xx2200001252

The selected certificates are moved to the **... OPCUACerts> trusted** folder.

Continues on next page

2.5.5 Client certificates
Continued

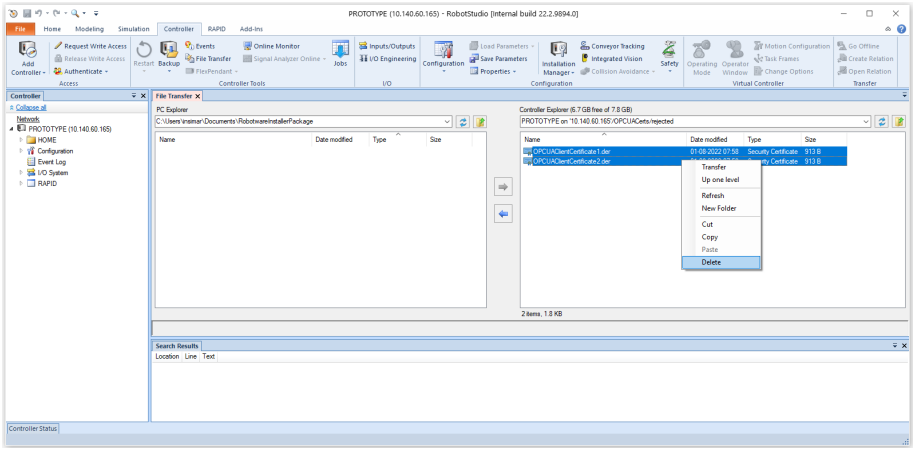
Delete certificate for RC

Using this operation, administrator can delete client certificate from the rejected and trusted certificate folders using the delete command of file transfer tool.

Use the following procedure to delete client certificate from the rejected and trusted certificate folder:

1. Open **RobotStudio** > **Controller** > **File Transfer**.

   The **Controller Explorer** window is displayed.
2. Navigate to the **... OPCUACerts> rejected** or **... OPCUACerts> trusted** folders.
3. Select the required certificates, right click on it, and select **Delete**.



xx2200001253

The selected certificates are deleted from the respective folder.

Delete certificate for VC

Using this operation, administrator can delete client certificate from the rejected and trusted certificate folders using the delete command of file transfer tool in virtual controller (VC).

Use the following procedure to delete client certificate from the rejected and trusted certificate folder:

1. Use the Windows File Explorer and navigate to the file system of VC.
2. Open the "...\internal\OPCUACerts" folder.

   The "rejected" and "trusted" folders are displayed.
3. Navigate to the **... OPCUACerts> rejected** or **... OPCUACerts> trusted** folders.
4. Select the required certificates, right click on it, and select **Delete**.

   The selected certificates are deleted from the respective folder.

3 Embedded OPC UA Server

3.1 Overview

This chapter provides detailed descriptions of the Embedded OPC UA address, space , Elog support, and security.

3.2 Address space

Introduction

The OPC UA Data Access function of the OPC UA Server is to read and write data managed by the ABB OmniCore controller.

Data items in OPC UA Server are referred by their node names.

The Embedded OPC UA Server presents various predefined nodes that provide information concerning to the robot controller's current state. In addition to these predefined nodes, the OPC UA Server presents all I/O signals and persistent RAPID variables.

ABB information model

The ABB information model is an ABB Robotics Proprietary OPC UA Information Model for robot controllers.

The tags exposed by the OPC UA Server follow the hierarchical structure of the OmniCore Controller object model.

Tags in the Controller domain of the Controller object model diagram	Unsupported tags in the Controller domain of the OmniCore controller object model diagram	Tag updated when OPC UA client application requests an update from the server
All of the RAPID and IOSYSTEM tags	CollisionDetectState	SystemClock
OperatingMode	RapidProgramFreememory	
ControllerState	RapidProgramUsedMemory	
ControllerExecutionState		
SpeedRatio		
MasterRAPID		
MasterCFG		
InterfaceState		

Note

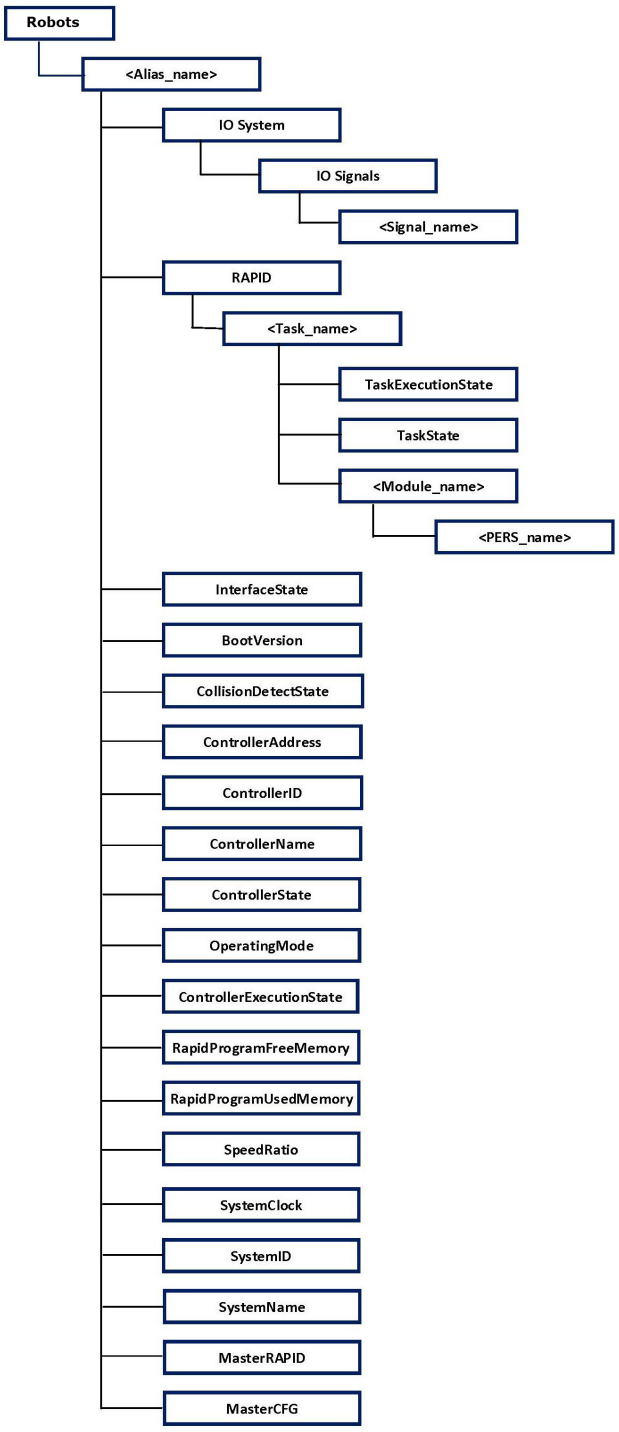
All other items are updated only when the controller restarts.

Note

Write functionality of RAPID persistent variables is not supported in the RobotWare 7.10 release of Embedded OPC UA Server.

Continues on next page

The following image shows the Objects and Variables :



xx2100000353

Note

For detailed description of the OPC UA Information Model for OmniCore robot controllers, see *Appendix B - ABB Robotics OPC UA proprietary information model on page 58*.

Continues on next page

Subscription for data changes

An Embedded OPC UA client can subscribe to multiple nodes monitored by OPC UA server which notifies the OPC UA client about the value changes.

The Embedded OPC UA server supports subscription to all variables in the address space. Notification of changes to variables like I/O signals and RAPID variables are sent to a subscribing OPC UA client based on change events from the robot controller.

RobotWare supports a maximum of 1000 subscriptions to unique resources between all consumers like Embedded OPC UA Server and Robot Web Services.

OPC UA Robotics companion specification

The OPC UA companion specification for Robotics specifies an OPC UA information model for the representation of a complete motion device system as an interface for higher-level control and evaluation systems. A motion device system consists out of one or more motion devices, which can be any existing or future robot type (e.g. industrial robots, mobile robots), kinematics or manipulator as well as their control units and other peripheral components.

The OPC UA Server supports the mandatory parts and some of the optional parts of the OPC UA companion specification for Robotics.

For more information, see *OPC 40010-1 - Robotics Part 1: Vertical integration*.

3.3 Events implementation

Overview

An event monitored item is a special type of a monitored item designed for receiving event notifications from the UA Server. For creating this kind of object, the user needs to create a monitored item for the EventNotifier attribute of an object node. The object node needs to have the SubscribeToEvents bit mask set in the EventNotifier attribute in order to allow the creation of event monitored items.

Event log event

All OPC UA event logs are OPC UA generated events. Some of the parameters included in the event structure that may have special meaning in the context are described below. See OPC UA Alarms and Events specification for the complete list.

- **Source–** The alias name of the controller that generated the event.
- **Message–** The title or brief explanation of the event.
- **Event Category –** The Event Log category.
- **Severity–** The severity of the event.

Following parameters contain event data if attribute values are requested by the client.

- **Number of Event Attributes–** The length of the event attribute array.
- **Event Attributes –** A pointer to the ABB specific event attributes as requested by the client according to the OPC specification.

Source Name	Time	Message	Severity
IN-L-BTGIS15033_OpcUa_VC_Vera	10:24:17.000 AM	Automatic mode confirmed	100
IN-L-BTGIS15033_OpcUa_VC_Vera	10:24:16.000 AM	Motors OFF state	100
IN-L-BTGIS15033_OpcUa_VC_Vera	10:24:15.000 AM	Speed adjusted	100
IN-L-BTGIS15033_OpcUa_VC_Vera	10:24:15.000 AM	Automatic mode requested	100
IN-L-BTGIS15033_OpcUa_VC_Vera	10:24:15.000 AM	Manual mode selected	100
IN-L-BTGIS15033_OpcUa_VC_Vera	10:24:15.000 AM	Safety guard stop state	100

xx2000000178

Event severity level

The OPC UA Alarms & Events Server automatically translates ABB OmniCore controller event log types to specific severity levels as shown in the table below.

Event log type	Severity
State Change	100
Warning	300
Error	600

Continues on next page

Event log attributes

In addition to the standard attributes required by the OPC UA Alarms and Events specification, the OPC UA Server Alarms and Events Server can provide the attributes defined in the table below.

Attribute ID	Attribute	Description
1	Number	Event number defined by RobotWare
2	Description	Detailed description of the event. This attribute will be empty if there is no description for a specific event log.
3	Action	Description of actions related to the event. This attribute will be empty if there is no description for a specific event log.

3.4 Security

3.4.1 Introduction

Overview

This chapter describes details about the security of Embedded OPC UA Server.

Disclaimer

The intent of this chapter is to raise awareness about security threats and to provide guidance to address them as well as to inform how ABB is working on security assurance. However, due to the high number of different security risks and complex dependencies within actual installations, this document can neither cover all possible security risks, nor guarantee the success of the presented security mechanisms.

3.4.2 Transport protocols

Data encodings

Embedded OPC UA Server supports UA Binary encoding only.

UA Binary: This message format encodes the data serialized into a byte array. UA Binary offers reduced computational cost in terms of encoding and decoding but can only be interpreted by OPC-UA compliant clients. UA Binary is more likely to be used in device level communications where processing power is limited and performance is a high priority.

Security protocols

A security protocol ensures the integrity and privacy of UA messages that are exchanged between OPC UA applications.

Embedded OPC UA Server uses UA Secure Conversation only.

Transport protocols

Embedded OPC UA Server uses UA TCP transport protocol.

3.4.3 Security configuration

Overview

When securing the communication with the OPC UA protocol, the following settings are required:

- Security policies
- User token policies

Security policies

Overview

Security policy and SecurityMode (message mode) parameters specify the security algorithms that the UA server supports.

Security policy

Selection of cryptographic algorithms. Any existing client and server which needs to interact should support this policy. Weaker security policies use outdated algorithms and should not be used. At a minimum, the Security Policy ‘Basic256Sha256’ should be chosen.

Embedded OPC UA Server supports following security policies:

- Basic256Sha256
- Aes128_Sha256_RsaOaep
- Aes256_Sha256_RsaPss

SecurityMode

The SecurityMode should be ‘Sign’ or ‘SignAndEncrypt’. This ensures that, authentication at the application level is enforced.

Embedded OPC UA Server supports only 'Sign' and 'Sign and encrypt'. 'None' security mode is disabled.

User token policies

OPC UA Applications support authentication of users by providing the necessary authentication credentials to the other entities.

Embedded OPC UA Server supports only UserName user token policies. User shall provide controller user name and password as defined in RobotWare UAS are accepted by the Embedded OPC UA Server.

This page is intentionally left blank

4 Troubleshooting

Overview

This chapter provides information about some of the possible trouble scenarios and its recommended solution.

Troubleshooting

The following table provides helpful information for troubleshooting:

Scenario	Solution
OPC UA Server is not running after a restart or reset of the controller	Go to RobotStudio -> Configuration ->Communication->OPC UA Server Set OPC UA Server option to **Yes**
An OPC UA client cannot connect to the OPC UA server. The error message “BadTimeout” or “BadCommunicationError” or Host unreachable (or similar) appears	1 Go to RobotStudio -> Configuration -> Communication -> Firewall Manager-> OpcUaServer->Enable on Public Network. and set the value on **Enable on Public Network** to **Yes**. 2 Go to RobotStudio -> Configuration ->Communication->OPC UA Server and set **OPC UA Server** option to **Yes**. 3 Any firewall between the client and server must allow TCP connection from client to server on port 4840. 4 If WAN IP of controller is modified, update server application instance certificate in opcua_store. Please refer Server Application instance certificates chapter 5 Make sure client use the correct endpoint URL.
Session Connect error: BadSecurityModeRejected Embedded OPC UA Server does not support the requested endpoint description.	Embedded OPC UA Server supports only 'Sign' and 'Sign and encrypt' modes. Embedded OPC UA Server supports following security policies: • Basic256Sha256 • Aes128_Sha256_RsaOaep • Aes256_Sha256_RsaPss Please check if the selected client is a supported security policy.
Session Connect error: BadUserAccessDenied Endpoint does not support the user identity type provided.	Embedded OPC UA Server supports only UserName user identity type(user token policy). Please check if the selected client is a supported user identity type.
An OPC UA client sees the endpoints of the server, but a connection to a secure endpoint fails. The error message BadSecurityChecksFailed appears.	Check whether the server trusts the client certificate. The required configuration steps can be found in section Trust client certificate(s).
An OPC UA client sees the endpoints of the server, but a connection to a secure endpoint fails. The error message **BadSecurityPolicyRejected** appears.	OPC UA Server rejects the client connection because of unsupported security policy provided by client. OPC UA Client should select the security polices that is supported by OPC UA Server. For more information, see *Security configuration on page 45* or *Connecting to OPC UA Server on page 16*.

Continues on next page

Continued

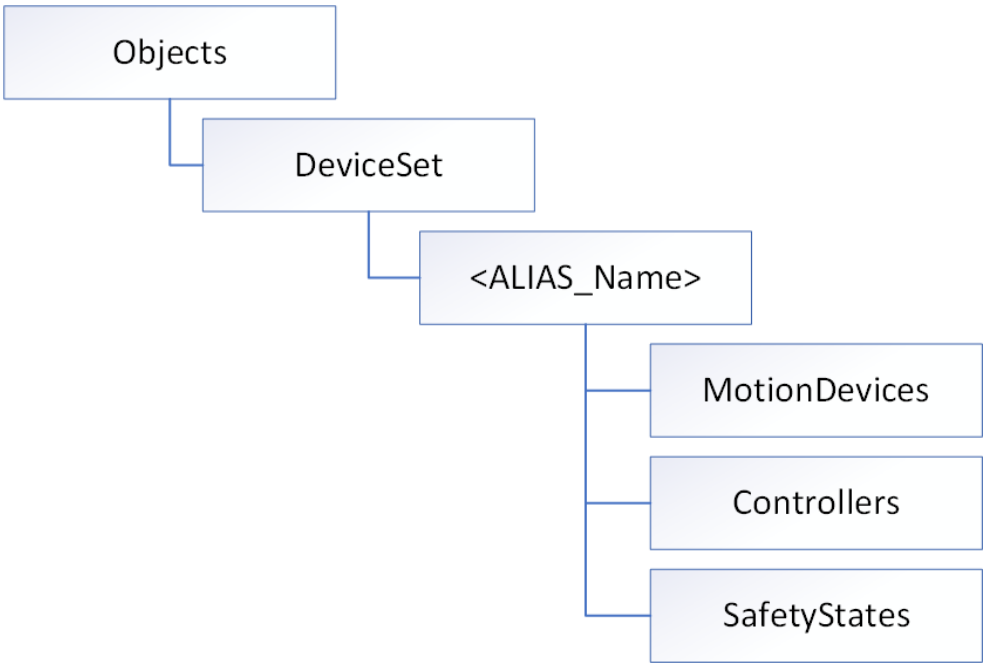
Scenario	Solution
An OPC UA client cannot connect to the OPC UA server. The error message “BadCertificateHost-NameInvalid	1 Make sure OPC UA Client uses correct endpoint URL  **Note** Some OPC UA Clients may connect with sever with wrong endpoint URL (wrong IP Address) with warning or without a warning. 2 Make sure the subject alternative name in the application instance certificate contains correct values for following. • IP Address={Controller WAN IP} For more information, see *Server application instance certificates on page 27*.
An OPC UA client cannot connect to the OPC UA server. The error message “BadCertificateUriInvalid”	1 Make sure OPC UA Client uses correct endpoint URL  **Note** Some OPC UA Clients may connect with sever with wrong endpoint URL with warning or without a warning. 2 Make sure subject alternative name in the application instance certificate contains correct values for the following: • URL=urn:vxTarget/Omnicore/OpcUa/Server For more information, see *Server application instance certificates on page 27*.
OPC UA Server performance goes down or appears to freeze.	If there are too many subscription updates due to many and frequent subscriptions to IO Signals. More than 500 subscription items which change every second can cause this behavior. Possible actions to resolve this issue are the following: • Reduce the number of subscriptions. • Reduce the frequency of changes to subscribed items.

5 Appendix

5.1 Appendix A - Robotics companion specification

Introduction

The OPC UA Server supports all mandatory and some of the optional parts of **OPC 40010-1 OPC UA for Robotics, Part 1: Vertical Integration**. This chapter describes the supported features and how they map to RobotWare.



xx2000002342

Supported features

DeviceSet

The DeviceSet is a container for all instances of ComponentType defined in OPC Unified Architecture for Devices (DI). One of the subtypes of the ComponentType is the MotionDeviceSystemType as described below.

Feature	Browse Name	Description
MotionDeviceSystem	<name>	Each instance corresponds to an ABB robot and <name> equals the Alias name given to the robot in the OPC UA server configuration.

MotionDeviceSystem

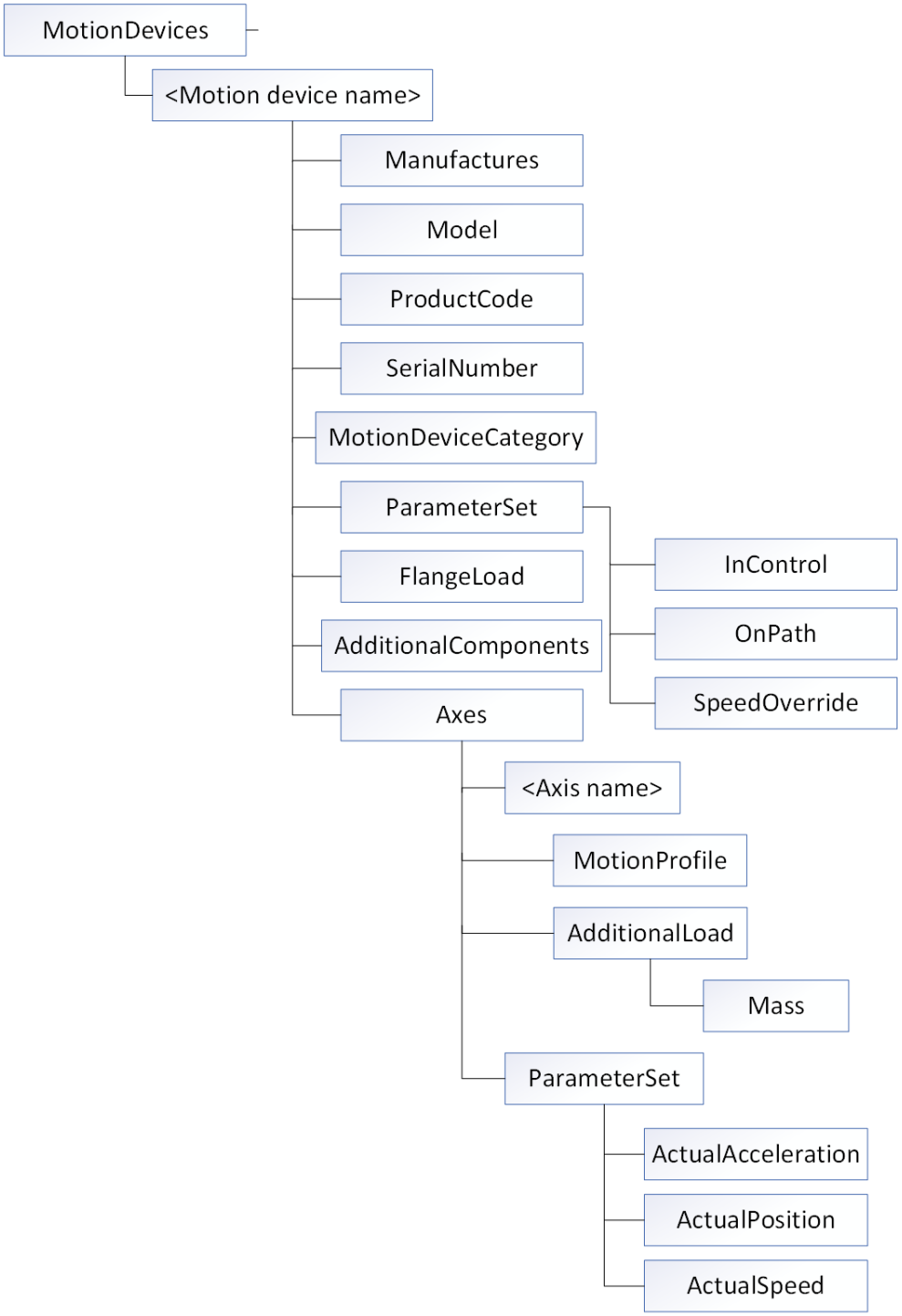
Feature	Browse Name	Description
MotionDevices	MotionDevices	A container for instances of MotionDeviceType
Controllers	Controllers	A container for instances of ControllerType
SafetyStates	SafetyStates	A container for instances of SafetyStateType

Continues on next page

5.1 Appendix A - Robotics companion specification

Continued

MotionDevices



xx2000002343

Feature	Browse Name	Description
MotionDevice	<name>	Each MotionDevice instance corresponds to a Mechanical Unit in the ABB robot. <name> is equal to the name of the Mechanical Unit.

Continues on next page

MotionDevice

Feature	Browse Name	Description
MotionDeviceCategory	MotionDeviceCategory	A categorization of the type of motion device based on ISO 8373, e.g. ARTICULATED_ROBOT
Manufacturer	Manufacturer	Name of manufacturer, i.e. “ABB”
Model	Model	Maps to the Model property of the Mechanical Unit, e.g. IRB5500_HWT
ProductCode	ProductCode	The article number for the Mechanical Unit, if available. Empty string otherwise.
SerialNumber	SerialNumber	The serial number if the Mechanical Unit, if available. Empty string otherwise.
FlangeLoad	FlangeLoad/Mass	The current Payload Mass of the Mechanical Unit.
ParameterSet	ParameterSet/OnPath	Not supported – always (null)
	ParameterSet/InControl	“true” if Motors ON, “false” otherwise
	ParameterSet/SpeedOverride	The Speed Ratio of the system 0 – 100%
Axes	Axes	A container for instances of AxisType
PowerTrains	PowerTrains	A container for instances of PowerTrainType
AdditionalComponents	AdditionalComponents	Empty folder, not in use.

Axes

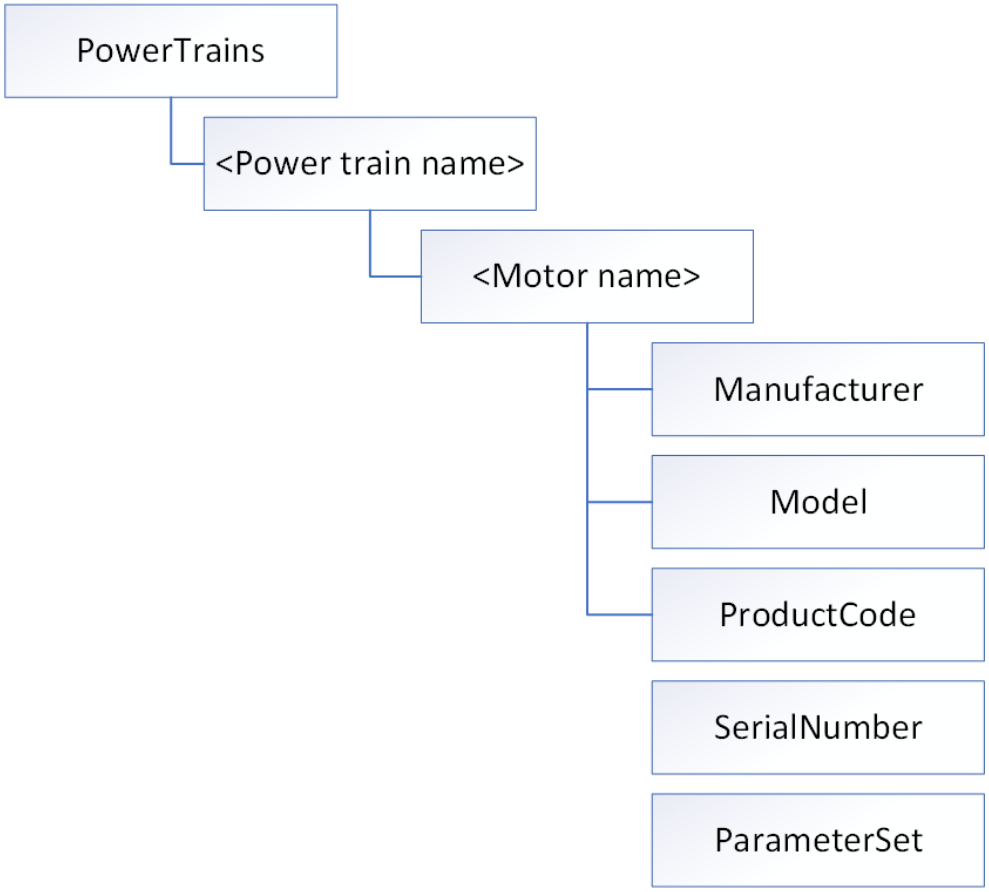
Feature	Browse Name	Description
Axis	<name>	Each instance corresponds to an axis of the Mechanical Unit. <name> is equal to the name of the axis, e.g. Rax_1 or Eax_6

Axis

Feature	Browse Name	Description
MotionProfile	MotionProfile	Property describing the type of motion for this axis, e.g. “ROTARY”.
AdditionalLoad	AdditionalLoad/Mass	Not supported – always 0.0
ParameterSet	ParameterSet/ActualPosition	Current position of axis
	ParameterSet/ActualSpeed	Not supported – always (null)
	ParameterSet/ActualAcceleration	Not supported – always (null)

Continues on next page

PowerTrains



xx2000002344

Feature	Browse Name	Description
PowerTrain	<name>	Each instance corresponds to a PowerTrain of the Mechanical Unit. <name> is equal to the joint name of the robot or the external axis of the mechanical unit that the power train drives. E.g. rob_1_1

PowerTrain

Feature	Browse Name	Description
Motor	<name>	Each instance of the MotorType corresponds to a Motor of the Axis. Normally there is one Motor per Axis. <name> is equal to the name of the Power-Train instance it belongs to.

Motor

Feature	Browse Name	Description
Manufacturer	Manufacturer	Name of manufacturer, i.e. “ABB”
Model	Model	Not supported – always (null)

Continues on next page

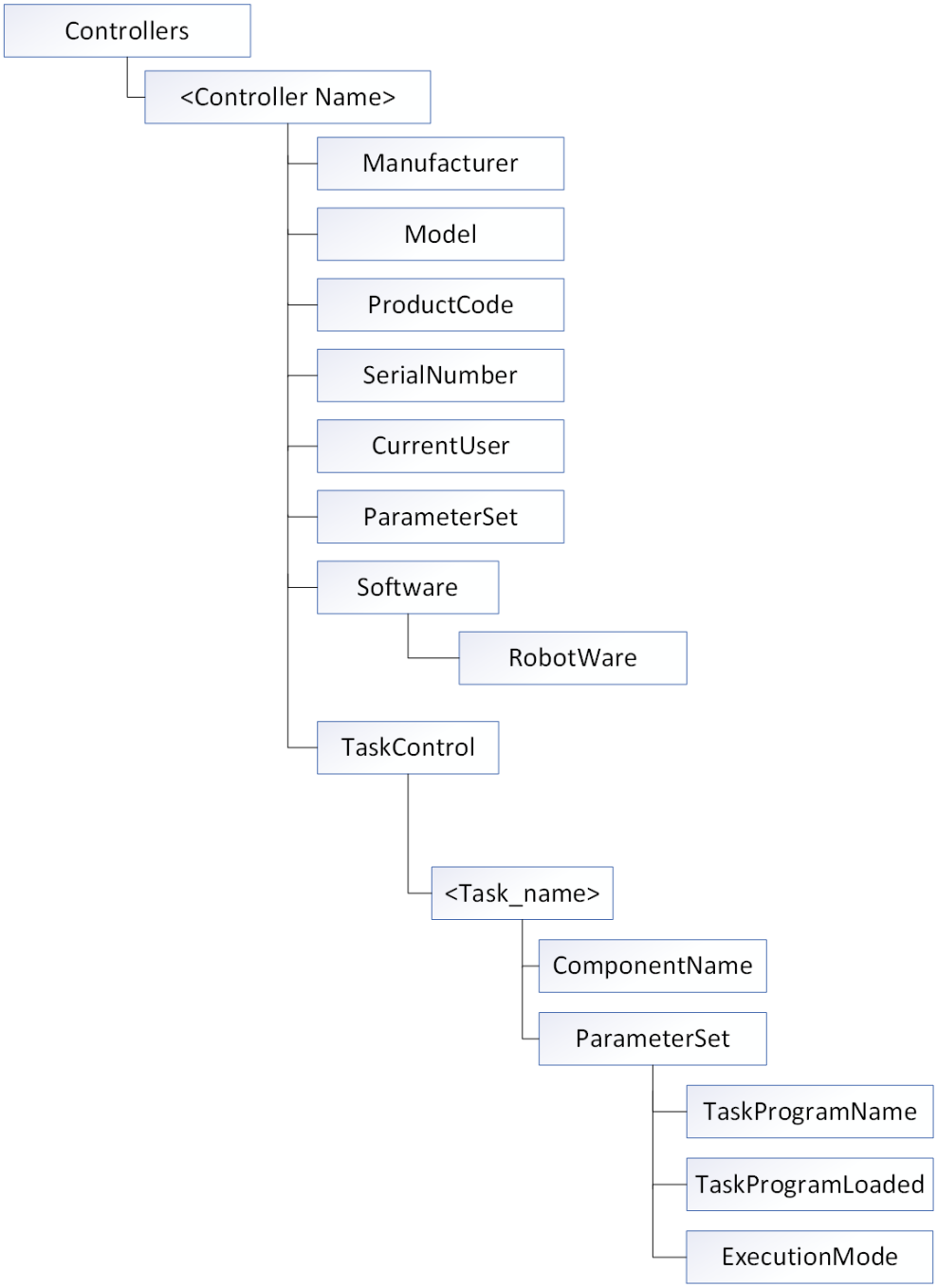
Feature	Browse Name	Description
ProductCode	ProductCode	Article number of Motor, retrieved from the “Use Motor Type” field of the Motor configuration.
SerialNumber		Not supported – always (null)
ParameterSet	ParameterSet/BrakeReleased	Not supported – always (null)
	ParameterSet/MotorTemperature	Not supported – always (null) This is a mandatory variable, but as ABB robots have only PTCs and not analog temperature sensors in the motors, there is no temperature to read.
	ParameterSet/EffectiveLoadRate	Not supported – always (null)

Continues on next page

5.1 Appendix A - Robotics companion specification

Continued

Controllers



xx2000002345

Feature	Browse Name	Description
Controller	<name>	There is always only one instance of ControllerType for ABB systems. <name> is equal to the Controller Name from the Controller Properties

Continues on next page

Controller

Feature	Browse Name	Description
Manufacturer	Manufacturer	Name of manufacturer, that is, “ABB”
Model	Model	Name of controller model, that is, “OmniCore”
ProductCode	ProductCode	Article number of controller. Not available digitally, so value is set to 'unknown'.
SerialNumber	SerialNumber	Serial number of controller. Not available digitally, so value is set to empty string.
CurrentUser	CurrentUser/Level	String containing list of grants assigned to the current user.
	CurrentUser/Name	Name of the current user. For example,“Default User”
ParameterSet	ParameterSet/TotalPowerOnTime	Provides the elapsed production time since the last SIS reset represented as an OPC UA DurationString. For details, see *https://reference.opcfoundation.org/v104/ISA-95/v100/docs/6.2.6/* A Virtual Controller will always show PT0H (that is, a zero duration)
	ParameterSet/StartUpTime	Not supported – always (null)
	ParameterSet/UpsState	Not supported – always (null)
	ParameterSet/TotalEnergyConsumption	Not supported – always (null)
	ParameterSet/CabinetFanSpeed	Not supported – always (null)
	ParameterSet/CPUFanSpeed	Not supported – always (null)
	ParameterSet/InputVoltage	Not supported – always (null)
	ParameterSet/Temperature	Not supported – always (null)
Components	Components	Empty folder, not in use.
Software	Software	A container for instances of SoftwareType
TaskControls	TaskControls	A container for instances of TaskControlType

Software

Feature	Browse Name	Description
Software	<name>	A list of software on the robot controller. For ABB controllers this list contains only one instance named RobotWare

Software: RobotWare

Feature	Browse Name	Description
Manufacturer	Manufacturer	Name of manufacturer, i.e. “ABB”
Model	Model	Name of software, typically “RobotWare”

Continues on next page

Feature	Browse Name	Description
SoftwareRevision	SoftwareRevision	Version number of software, e.g. “6.11.0.1”

TaskControls

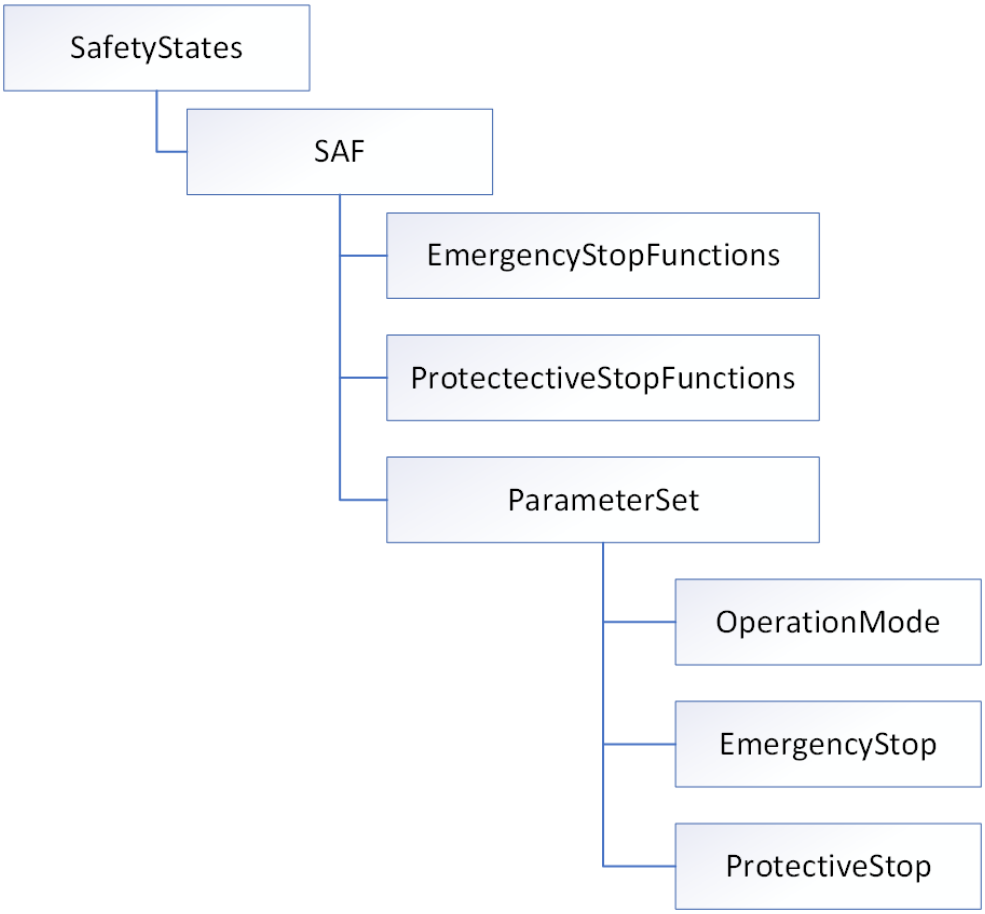
Feature	Browse Name	Description
TaskControl	<name>	Each TaskControlType instance listed corresponds to a RAPID task. <name> is equal to the name of the corresponding RAPID task.

TaskControl

Feature	Browse Name	Description
ComponentName	ComponentName	Contains the name of the corresponding RAPID task, i.e. the same as the Browse Name for the TaskControl itself.
ParameterSet	ParameterSet/TaskProgramName	
	ParameterSet/TaskProgramLoaded	True if the RAPID task has a defined execution context, i.e. that some RAPID code is loaded and the program pointer is defined.
	ParameterSet/ExecutionMode	Not supported – always (null)

Continues on next page

SafetyStates



xx2000002346

Feature	Browse Name	Description
SafetyState	<name>	For ABB controllers this list contains only one instance named “SAF”

SafetyState: SAF

Feature	Browse Name	Description
ParameterSet	ParameterSet/OperationalMode	Corresponds to the Operating Mode of the robot controller.
	ParameterSet/EmergencyStop	True if emergency stop is activated, false otherwise.
	ParameterSet/ProtectiveStop	True of one or more protective stops are activated, false otherwise.
EmergencyStopFunctions	EmergencyStopFunctions	Empty folder, not in use.
ProtectiveStopFunctions	ProtectiveStopFunctions	Empty folder, not in use.

5.2 Appendix B - ABB Robotics OPC UA proprietary information model

5.2.1 Overview

This section describes the OPC UA information model for ABB robot controllers.

5.2.2 OPC Unified Architecture for ABB Robotics Controller

ObjectType RobotControllersType

A container for Robot Controller objects.

Table 1: ObjectType RobotControllersType

Attribute	Value
BrowseName	RobotControllersType
IsAbstract	False

Subtype of FolderType of http://opcfoundation.org/UA/

Reference	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule	Access
HasComponent	Object	S_Alias_name_		RobotControllerType	OptionalPlaceholder	

S_Alias_name_: A robot controller is identified by its alias name that must be unique.

ObjectType RobotControllerType

Top level object type for an ABB Robotics Controller.

Table 2: ObjectType RobotControllerType

Attribute	Value
BrowseName	RobotControllersType
IsAbstract	False

Subtype of BaseObjectType of http://opcfoundation.org/UA/

Reference	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule	Access
HasProperty	Property	BootVersion	String	PropertyType	Mandatory	Read
HasProperty	Property	ControllerAddress	String	PropertyType	Mandatory	Read
HasProperty	Property	ControllerID	String	PropertyType	Mandatory	Read
HasProperty	Property	ControllerName	String	PropertyType	Mandatory	Read
HasProperty	Property	SystemID	Guid	PropertyType	Mandatory	Read
HasProperty	Property	SystemName	String	PropertyType	Mandatory	Read
HasComponent	Variable	CollisionDetectState	CollisionDetectStateEnum	BaseDataVariableType	Mandatory	Read
HasComponent	Variable	ControllerExecutionState	ControllerExecutionStateEnum	BaseDataVariableType	Mandatory	Read
HasComponent	Variable	ControllerState	ControllerExecutionStateEnum	BaseDataVariableType	Mandatory	Read
HasComponent	Variable	InterfaceState	InterfaceStateEnum	BaseDataVariableType	Mandatory	Read

Continues on next page

Reference	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule	Access
HasComponent	Variable	OperatingMode	OperatingModeEnum	BaseDataVariableType	Mandatory	Read
HasComponent	Variable	SpeedRatio	Int32	BaseDataVariableType	Mandatory	Read
HasComponent	Variable	SystemClock	DateTime	BaseDataVariableType	Mandatory	Read
HasComponent	Variable	RapidProgramUsedMemory	UInt32	BaseDataVariableType	Mandatory	Read
HasComponent	Variable	RapidProgramFreeMemory	UInt32	BaseDataVariableType	Mandatory	Read
HasComponent	Variable	MasterRAPID	MastershipEnum	BaseDataVariableType	Mandatory	Read
HasComponent	Variable	MasterCFG	MastershipEnum	BaseDataVariableType	Mandatory	Read
HasComponent	Object	IO_System		IOSystemType	Mandatory	
HasComponent	Object	RAPID		RAPIDType	Mandatory	

BootVersion: A read-only string that contains the value of the robot controller's RobotWare operating system version.

ControllerAddress: A read-only string that specifies either a) the IP network address of the Real Controller (RC), or b) the localhost loopvback address (127.0.0.1) for a Virtual Controller (VC) running on the PC.

ControllerID: By default, set to the serial number of the controller and is thereby a unique identifier of the controller. This is a read-only value.

ControllerName: An identification of the controller that is independent of the system or the software running on the controller. This name comes from the robot controller and may be the same as the AliasName, however while the AliasName must be unique, there is no such requirement on the ControllerName. This is a read-only value.

SystemID: A read-only GUID that contains the identifier that globally and uniquely identifies a robot controller/system combination.

SystemName: A read-only string that contains the name of the RobotWare system currently loaded. This is the name assigned by the user when creating a system in e.g. Installation Manager.

CollisionDetectState: A read-only value that contains the state of the collision detection mechanism in the robot controller. See the definition of CollissionDetectStateEnum for details.

ControllerExecutionState: A read-only value that contains the execution state (Running or Stopped) of the robot controller.

ControllerState: A read-only value that contains the state of the robot controller. See the definition of ControllerStateEnum for details.

Continues on next page

InterfaceState: A read-only value indicating the state of the communication interface to the robot controller. This state is maintained by the OPC UA server. See the definition of InterfaceStateEnum for details.

OperatingMode: A read-only value that contains the robot controller operational mode. See the definition of OperationgModeEnum for details.

SpeedRatio: A read-only value that defines the speed ratio of the robot controller in percent, range 0 - 100.

SystemClock: A read-only value that contains the robot controller's system clock value. It is only valid when the interface to the controller is operational.

RapidProgramUsedMemory: A read-only value that defines the amount of memory in bytes being used by the robot controller's RAPID program.

RapidProgramFreeMemory: A read-only value that defines the amount of memory in bytes available to the robot controller's RAPID program.

MasterRAPID: A read-only value that shows if the mastership of RAPID is held by another client. See definition of MastershipEnun for details.

MasterCFG: A read-only value that shows if the mastership of CFG is held by another client. See definition of MastershipEnun for details.

IO_System: Represents the I/O system of the controller.

RAPID: Container for all RAPID tasks in the controller.

ObjectType IOSystemType

Object type describing the the I/O system of the robot controller.

Table 3: ObjectType IOSystemType

Attribute	Value
BrowseName	IOSystemType
IsAbstract	False

Subtype of FolderType of http://opcfoundation.org/UA/

Reference	NodeClass	BrowseName	Data-Type	TypeDefinition	Modelling-Rule	Access
HasComponent	Object	IO_Signals		IOSignalsType	Mandatory	

IO_Signals: Container for all I/O signals in the controller.

ObjectType IOSignalsType

A container for I/O signals.

Table 4: ObjectType IOSignalsType

Attribute	Value
BrowseName	IOSignalsType
IsAbstract	False

Continues on next page

Subtype of FolderType of http://opcfoundation.org/UA/

Reference	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule	Access
HasComponent	Variable	S_Signal_name_	BaseDataType	DataItemType	OptionalPlaceholder	ReadWrite

S_Signal_name_: Represents an IO signal.

> **Note**
>
> Depending on the configuration of signal, it may be possible to write to a input or output signal. The 'Type of signal', 'Access level', and 'Safe level' parameters all influence whether it is possible to write to a 'signal' or not. OPC UA Client see all the signals from Robot controller's point of view, that is, O/P signal is an output from the Robot controller to some external equipment and vice versa for I/P signals.

For more information on system parameters, please refer to*Technical reference manual - System parameters*, **section I/O**.

ObjectType RAPIDType

Object type describing the RAPID sub-system of the robot controller.

Table 5: ObjectType RAPIDType

Attribute	Value
BrowseName	RAPIDType
IsAbstract	False

Subtype of FolderType of http://opcfoundation.org/UA/

Reference	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule	Access
HasComponent	Object	S_Task_name_		RAPIDTaskType	MandatoryPlaceholder	

S_Task_name_: Represents a RAPID task in the controller.

ObjectType RAPIDTaskType

Represents a RAPID task in the controller. It acts as a container for any modules loaded in the task.

Table 6: ObjectType RAPIDTaskType

Attribute	Value
BrowseName	RAPIDTaskType
IsAbstract	False

Continues on next page

Subtype of FolderType of http://opcfoundation.org/UA/

Reference	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule	Access
HasComponent	Variable	TaskExecutionState	TaskExecutionStateEnum	BaseDataVariableType	Mandatory	Read
HasComponent	Variable	TaskState	TaskExecutionStateEnum	BaseDataVariableType	Mandatory	Read
HasComponent	Object	S_Module_name_		RAPIDModuleType	OptionalPlaceholder	
HasComponent	Variable	ProgramPointer	ProgramPosition	BaseDataVariableType	Mandatory	Read

TaskExecutionState: A read-only value that contains the execution state of the RAPID task. See the definition of **TaskExecutionStateEnum** for details.

TaskState: A read-only value that contains the state of the RAPID task. See definition of TaskStateEnum for details.

S_Module_name_: Represents a RAPID module in a RAPID task.

ProgramPointer: ProgramPointer is introduced under RAPID address space to get correct module name, routine, and line number (exposed by the OPC UA server for each RAPID task).

ObjectType RAPIDModuleType

An object representing a RAPID module. It acts as a container for all persistent variables in the module.

Table 7: ObjectType RAPIDModuleType

Attribute	Value
BrowseName	RAPIDModuleType
IsAbstract	False

Subtype of FolderType of http://opcfoundation.org/UA/

Reference	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule	Access
HasComponent	Variable	S_PERS_name_	BaseDataType	DataItemType	OptionalPlaceholder	ReadWrite

S_PERS_name_: Represents a persistent (PERS) variable in a RAPID module. Clients can both read and write persistent variables. A successful write to a persistent variable requires that no other client has mastership of RAPID. See description of **MasterRAPID** variable.

CollisionDetectStateEnum Values

Defines possible states of the collision detection mechanism in the robot controller.

Table 8: CollisionDetectStateEnum Values

Continues on next page

5.2.2 OPC Unified Architecture for ABB Robotics Controller
Continued

Subtype of Enumeration of http://opcfoundation.org/UA/

Name	Value	Comment
Unknown	0	Unknown.
Initiated	1	Collision detection has been initiated.
Started	2	Collision detection has been started.
Confirmed	3	Collision detected/confirmed.
Acknowledged	4	Collision detected and acknowledged.

Continues on next page

ControllerExecutionStateEnum Values

Defines possible execution states of the robot controller.

Table 9: ControllerExecutionStateEnum Values

Subtype of Enumeration of http://opcfoundation.org/UA/

Name	Value	Comment
Unknown	0	Status is unknown.
Running	1	At least one normal RAPID task is executing or performing regain.
Stopped	2	No normal RAPID task is executing or performing regain.

ControllerStateEnum Values

Defines possible states of the robot controller.

Table 10: ControllerStateEnum Values

Subtype of Enumeration of http://opcfoundation.org/UA/

Name	Value	Comment
Init	0	Initialize state.
MotorsOff	1	Motors off state.
MotorsOn	2	Motors on state.
GuardStop	3	Guard stop state.
EmergencyStop	4	Emergency stop state.
EmergencyStopReset	5	Emergency stop reset state.
SystemFailure	6	System failure state.
Unknown	99	Unknown state.

InterfaceStateEnum Values

Defines possible states of the interface to the robot controller.

Table 11: InterfaceStateEnum Values

Subtype of Enumeration of http://opcfoundation.org/UA/

Name	Value	Comment
Disconnected	0	The interface to the robot controller is disconnected and non-functional.
Connected	1	The interface to the robot controller is connected and operational.
NoPCInterfaceOption	2	The robot controller does not have the PC Interface RobotWare option that creates the interface to the controller.
UnresolvableAlias	3	The system cannot resolve the indicated alias to a single robot controller on the network.

Continues on next page

OperatingModeEnum Values

Defines possible operational modes of the robot controller.

Table 12: OperatingModeEnum Values

Subtype of Enumeration of http://opcfoundation.org/UA/

Name	Value	Comment
Auto	0	Automatic mode (production).
Init	1	Initialize mode.
ManualReduced-Speed	2	Manual reduced speed mode.
ManualFullSpeed	3	Manual full speed mode.
AutoChange	4	A change to automatic mode has been requested.
ManualFullSpeed-Change	5	A change to manual full speed has been requested.
NotApplicable	6	Controller operating mode is not applicable in current controller state.

TaskExecutionStateEnum Values

Defines possible task execution states.

Table 13: TaskExecutionStateEnum Values

Subtype of Enumeration of http://opcfoundation.org/UA/

Name	Value	Comment
Ready	0	The task has no PCP or execution context.
Stopped	1	Task is not executing or not performing regain. PCP and execution context are defined in task.
Running	2	Task is executing or performing regain.
UnInitiated	3	The program server is not initialized. State only assumed during startup.
Unknown	4	Status is unknown.

TaskStateEnum Values

Defines possible task states.

Table 14: TaskStateEnum Values

Subtype of Enumeration of http://opcfoundation.org/UA/

Name	Value	Comment
Empty	0	No modules are loaded in the task.
Loaded	1	Modules are loaded, but not linked.
Linked	2	Modules are loaded and linked.
Initiated	3	The program server is not initialized. State only assumed during startup.

Continues on next page

MastershipEnum Values

Defines possible mastership values.

Table 15: MastershipEnum Values

Subtype of Enumeration of http://opcfoundation.org/UA/

Name	Value	Comment
NoMaster	0	No client has mastership
HeldRemote	1	A remote client has mastership.
HeldLocal	2	A local client has mastership (typically the TPU)
HeldInternal	3	The controller itself has mastership.

This page is intentionally left blank

Index

ABB AB
Robotics & Discrete Automation
S-721 68 VÄSTERÅS, Sweden
Telephone +46 10-732 50 00

ABB AS
Robotics & Discrete Automation
Nordlysvegen 7, N-4340 BRYNE, Norway
Box 265, N-4349 BRYNE, Norway
Telephone: +47 22 87 2000

ABB Engineering (Shanghai) Ltd.
Robotics & Discrete Automation
No. 4528 Kangxin Highway
PuDong New District
SHANGHAI 201319, China
Telephone: +86 21 6105 6666

ABB Inc.
Robotics & Discrete Automation
1250 Brown Road
Auburn Hills, MI 48326
USA
Telephone: +1 248 391 9000

abb.com/robotics

3HAC085436-001, Rev E, en