rexroth
A Bosch Company

BOSCH REXROTH
IOT GATEWAY 2.X
USER MANUAL

Ver. 1.0E

Description

This publication presents the configuration methods and instructions for running of the Bosch Rexroth IoT Gateway software

DCEE/SLF-IND4-PL
I4.0@boschrexroth.pl

IMPORTANT! READ!

Copyrights

Liability

The specified data is intended for product description purposes only and shall not be deemed to be a guaranteed characteristic unless expressly stipulated in the contract. All rights are reserved with respect to the content of this documentation and the availability of the product

The reserved names and trademarks appearing in the publication are registered trademarks or trademarks of their owners. All proprietary names, registered trademarks and trademarks belonging to third parties have been used in this publication by Bosch Rexroth Sp. z o. o. for informational and identification purposes only.

Table of Contents

Preface: What is IoT Gateway from Bosch Rexroth?

Digitalization has entered the production environment with Industry 4.0. Its central element is the factory of the future, where the shop floor is linked to the IT infrastructure. Bringing existing resources onto a connected network is essential for digitalization, and the new IoT Gateway from Bosch Rexroth delivers that connectivity seamlessly.

Connectivity plays a central role in the factory of the future. Connecting machines, materials, products, and workers is a prerequisite for increasing productivity and for new services and business models.

The aim of the I4.0 Connectivity research project was to develop a solution that also enables existing machines to be easily readied for Industry 4.0, without any engineering effort. Devised as a retrofit kit, it includes sensors that can easily be fitted to existing machines, a networking gateway and analysis software for making good use of the acquired data.

A key to success was pooling the broad scope of competencies at Bosch in order to achieve this goal. The focus of Bosch Research was on the development of software for the gateway. From colleagues at Bosch sensor engineering to automation engineering experts from Bosch Rexroth and IT specialists from Bosch Software Innovations, all partners were involved in the implementation of this overall concept. In addition, users from Bosch plants provided practical application scenarios as input and enabled the technologies to be evaluated in a real manufacturing facility.

The collaborative effort gave rise to a gateway based on modern IT technologies and an open architecture, developed in an agile process using the SCRUM method. We put together a joint team with our partner Bosch Rexroth very early on in the development phase, enabling a smooth transfer of findings into the development of the product and a rapid market launch.

The new IoT Gateway, presented to the public for the first time in 2016, is an important building block for networking machines and helps prepare factories to meet future requirements. For I4.0 research, the data obtained in this way opens up new possibilities and enables new insights into interrelationships in production.

Bosch developed the IoT Gateway to connect existing production infrastructure over networks built on open standards. The device makes it possible to bring current production elements along for the ride in an Industry 4.0 world.

More info:

Catalog page
Video: Configure in 5 min

Brochures/Flyers:

IoT Gateway software
StarterKit PPM + IoT Gateway
Device Portal

Dedicated WWW pages:

Main WWW page of the product
Bosch Software Innovations

Contact:

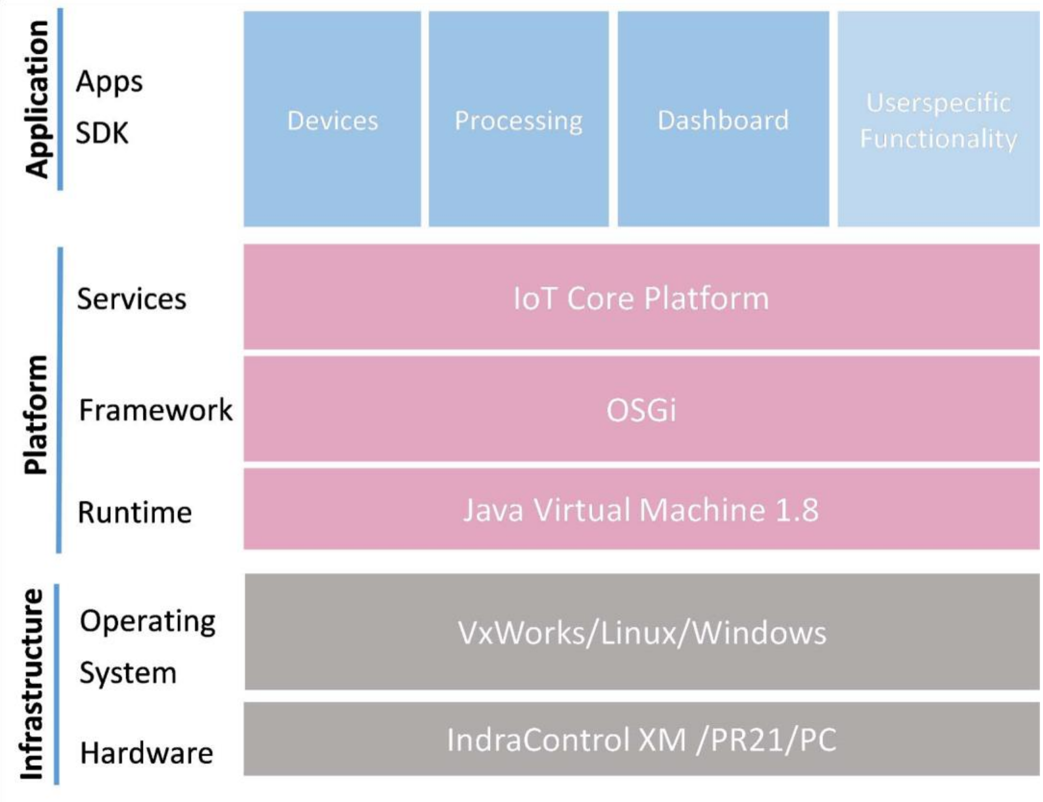
Bosch Rexroth Sp. z o. o.
ul. Jutrzenki 102/104
02-230 Warszawa
POLAND
email: i4.0@boschrexroth.pl

1. Overview

The IoT Gateway connects the industrial machine environment with the enterprise IT-world and can provide machine- and process data in a simple way.

The Java based IoT Gateway software is a modular system and uses management mechanisms and services provided by OSGi Framework.

Based on the IoT Core Framework, the IoT Gateway provides services to connect field devices, like sensors or industrial control systems as well as functions to process data and to provide this data to superior services and systems.



Running on the control system IndraControl XM there is a high performance combination of data acquisition with logic and motion processing in real time and large connectivity available. Via a SDK it is possible to create custom functions.

1.1 Definitions

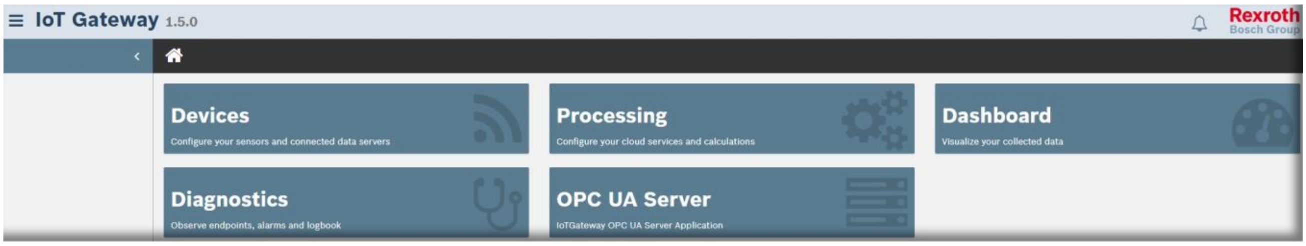
The following table shows definitions used in the manual:

Endpoint	**The Endpoint is the central location of a value. Every Endpoint has a unique name and reference to its value source. A unit and description can be added optionally**
Endpoint value	The current value of an endpoint
Device	A Device is a data source that can be accessed by Endpoints. According to the device type several parameters and settings can be configured

Process	A Process is the data drain where Endpoint values are provided to. A process can e.g. be a cloud-service or a calculation. The result of a calculation, also represented by an Endpoint can again serve as an input of another processing.
App	Different functions of the IoT Gateway are available as an App (Application). An App is provided by one or more bundles.
Bundle	A bundle is an independent module. The bundle-configurator (see 2.1.4 Bundle-Configurator) allows loading, deleting and enabling Apps

2. Usage

The web user interface provides access to administrate the IoT Gateway from any location.


≡ IoT Gateway 1.5.0
Rexroth
Bosch Group
Devices
Configure your sensors and connected data servers
Processing
Configure your cloud services and calculations
Dashboard
Visualize your collected data
Diagnostics
Observe endpoints, alarms and logbook
OPC UA Server
IoTGateway OPC UA Server Application

The modular structure allows adding and removing functionality on demand. Particular functions are displayed as tiles.

2.1 Menu ≡

2.1.1 Help

Here detailed information on the different functions can be found.

Notice: Function specific help can be provided by a click on the info icon ℹ of the selected function.

2.1.2 License Agreement

This section contains the license information of the software.

2.1.3 Open Source Software

This section contains a list of open source software components used by the IoT Gateway.

2.1.4 Bundle-Configurator

The Bundle-Configurator can enable and disable Apps. To increase performance and speeding up the boot process unused apps should be disabled.

Notice: The Bundle-Configurator is not available for ProSystMBS.

2.1.5 Settings

In the settings tab, a proxy server can be configured and a web console can be enabled for analysis purposes. Finally, the authentication settings can be changed here.

2.1.6 Logout

Logout to leave the edit-mode of the IoT Gateway. After that, it is not possible to operate the software until another login.

2.2 Home-Button 🏠

Opens the start page with an overview of the installed apps.

2.3 Functions (Apps)

The installed apps can be accessed from the start page (home button). The following chapters describe the functionality of all the apps delivered by Bosch Rexroth. Via the Bundle-Configurator (see 2.1.4 Bundle-Configurator) it is possible to install not listed, custom apps. A list of all apps provided by Bosch Rexroth can be found in the product catalogue (http://www.boschrexroth.com/gateway).

3. General

3.1 Setting up and Launching the IoT Gateway

The IoT Gateway can be configured in different steps, depending on the type of provision. The following procedure is useful:

3.1.1 Setting up the IoT Gateway

3.1.1.1 XM platform

For mechanical and electrical installation take a look into the manual of the hardware device, e.g. IndraControl XM21, XM22

- Network settings (see chapter 12.5.2 in above mentioned document)
- S20-bus (see chapter 10.3 in above mentioned document)

3.1.1.2 Docker Installation

1. Requirement: Docker runtime environment has to be installed
2. Import of a Docker package: `docker load < iiot-1.6.0.tar.gz`
3. Starting Docker: `docker run -p 8888:8888 -v ~/data:/home --name mbs brc/iiot:1.6.0`
   Explanation of the command:

`docker run`	creates a runnable docker container from a docker file
`-p 8888:888`	port mapping, host(8888) : container(8888)
`-v ~/data:/home`	directory mapping host(~/data):container(/home)
`--name mbs`	identification of a container through a name, here: “mbs”
`brc/iiot:1.6.0`	image to choose

4. List Docker containers: `docker ps`
5. Stop Docker container: `docker stop mbs`
6. Delete Docker container: `docker rm mbs`
7. Delete Docker image: `docker rmi <image id>`

3.1.1.3 Snap Installation on PR21

3.1.1.3.1 Network Configuration

In delivery status the standard configuration for the ports XF5, XF6 is:

XF5	DHCP
XF6	IP address: 192.168.0.1 Net mask: 255.255.255.0 Gateway: localhost (127.0.0.1)

Edit the network configuration:

To change the network configuration access to the console of the PR21 device is necessary.

- Connect via SSH, e.g. using Putty: https://putty.org/ , or directly using the HDMI port of the device and a hardware keyboard
- Log in (user: boschrexroth, password: boschrexroth), Notice: there is no visual output while typing the password

- After having logged in successfully you are in the home directory of user “boschrexroth”
- If you prefer to change the keyboard settings from US to other region do it by typing

  ```
  sudo loadkeys <regional code>
  ```

  where <regional code> is `de` e.g. for Germany
- Open the file `*/etc/netplan/00-snapd-config.yaml` with the vi editor (see function description https://www.washington.edu/computing/unix/vi.html)

```
network:
     version: 2
     ethernets:
       eth0:
         addresses: []
         dhcp4: true
       eth1:
         addresses: [192.168.0.1/24]
         nameservers:
         addresses: [1.1.1.1]
```

The file contains the configuration of the Ethernet ports *XF5* (eth0) and *XF6* (eth1) in section *ethernets*:

Entry	Description
eth0	Name of the interface („eth0"), XF5
addresses: [ ]	Space for ipv4 addresses, no entry since DHCP is configured
dhcp4: true	DHCP for IP v4 is active
eth1	Name of the interface („eth1"), XF6
addresses [192.168.0.1/24]:	IP address 192.168.0.1 Subnet mask 255.255.255.0
name servers:	Set DNS servers and search domains, for manual address configuration
addresses: [1.1.1.1]	Addresses to the interface in addition to those received through e.g. DHCP

* The subnet mask has to be entered as the number of relevant set bits in each byte

```
255.255.255.0
11111111.11111111.1111111.00000000 (24 bits set)
```

- Edit settings and save the modified file on the PR21 in the mentioned folder

- To activate the new settings reboot the device or type the following command:
  ```
  sudo netplan apply
  ```

- The new network settings are active now. You can check these by typing:
  ```
  ifconfig
  ```
  in the console of the device.

Further information on the network configuration can be found at https://netplan.io .

3.1.1.3.2 Installation of the Snap

An FTP tool (e.g. WinSCP https://winscp.net/eng/download.php) is required to install the Snap on the device.

- Open an SFTP connection and copy the snap file in the home directory of the user „boschrexroth" /home/boschrexroth/
- Log in via SSH connection (User: boschrexroth, password: boschrexroth)

Notice: there is no visual output while typing the password

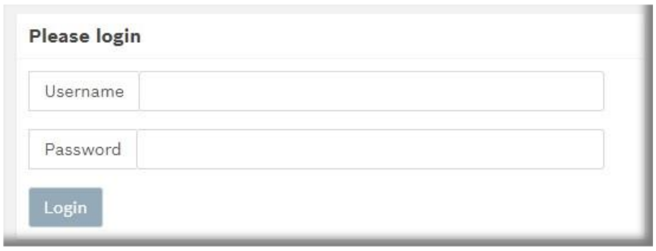
- After having logged in successfully you are in the home directory of user “boschrexroth”
- Now install the Snap typing the following command:
  ```
  sudo snap install <name of snap file> --devmode
  ```
- After a confirmation with ENTER you have to type in user and password again to start the installation.
- The installation is now complete. The service is started automatically after installation.

In the following table further useful commands can be found.

command	function
`snap list`	List of all installed snaps
`sudo systemctl status snap.iiot.iiot.service`	Output status of the currently installed snaps
`sudo snap revert iiot`	Return to a specific snap installation
`sudo snap remove iiot`	Uninstall snap
`sudo snap stop iiot`	Stop snap
`sudo snap restart iiot`	Restart snap

3.1.2 Launching the IoT Gateway and First Login

1. Open the web interface via the address `https://<IP Host>:8888`



2. Login with Username: admin, Password: admin
3. Confirm the EULA and change the initial authentication settings.
4. Configuration of data source (Devices)
5. Add Devices
6. Add Endpoints
7. Configuration of data drains (Processing)
    1. Link the Endpoints (Processing/Calculation)
    2. Provide Endpoints and Cloud Services (Processing/Cloud Services)
8. Display selected Endpoints (Dashboard).

3.2. Sampling and Properties of Endpoints

3.2.1 Properties of Endpoints

An Endpoint describes a data drain of a scalable value (Endpoint value). Besides the value, a timestamp and the quality of the value are stored.

There are three conditions of Endpoint quality:

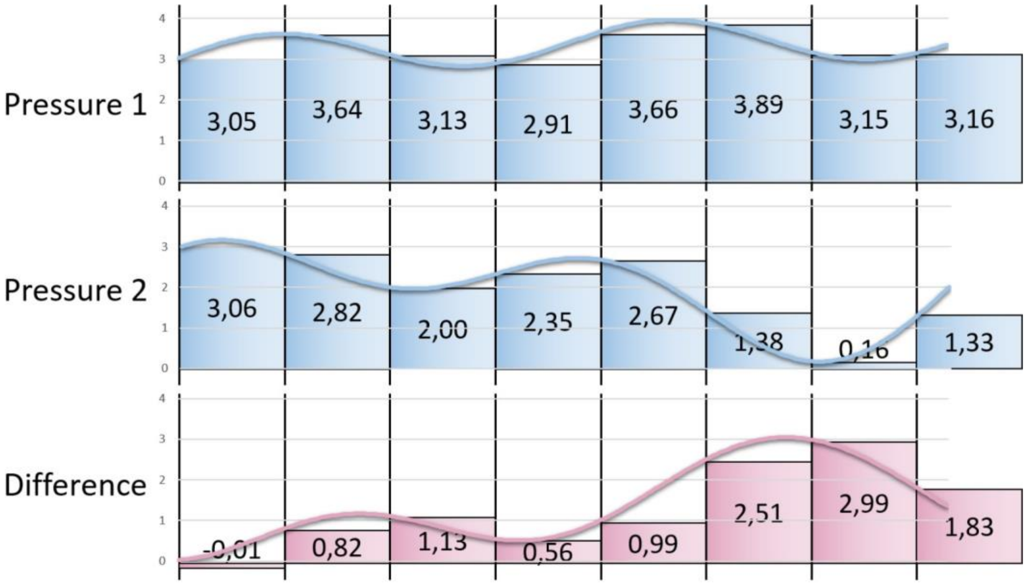
GOOD	The Endpoint value has been collected, processed and published successfully
BAD	The Endpoint value could not be collected (no communication). The quality was GOOD in a former point in time. In this condition there is no processing and no publishing of the Endpoint value
UNCERTAIN	The Endpoint value could not be collected yet. In this condition, there is no processing and no publishing of the Endpoint value to the Processing unit. However, the OPC-UA server of the IoT Gateway contains the Endpoint value

3.2.2 Sampling of Endpoints

It is necessary to bind sample rates to Endpoints and processing units to avoid systematic errors while measurement and data processing. This chapter gives a simple introduction in the technique of measurement.

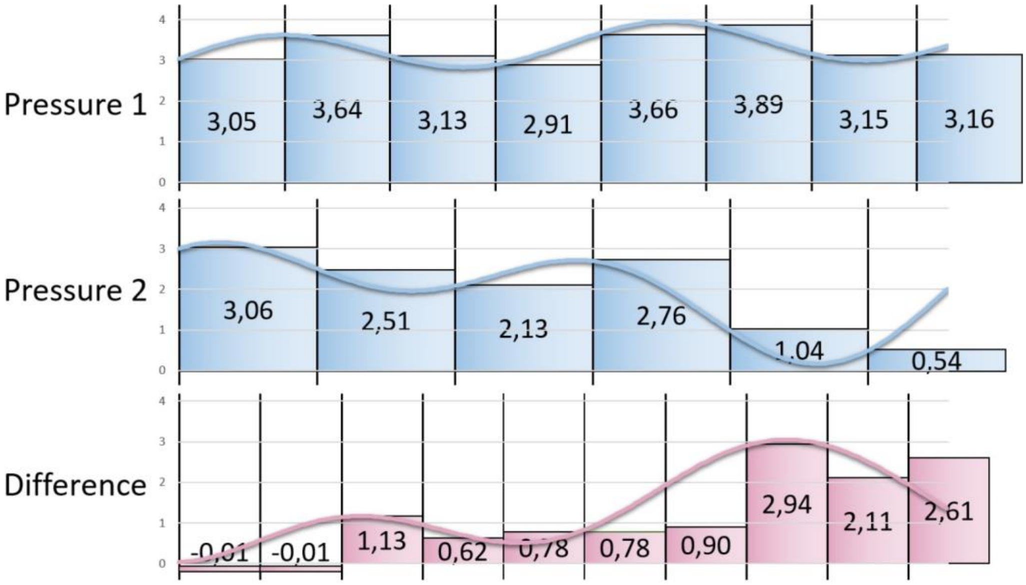
The following graphs show the signal sequence of pressure 1, pressure 2 and the difference between pressure 1 and pressure 2. Different scenarios showing the measured values and differences for different sampling times. The color gradient from dark blue to light blue illustrates the “aging” of the measured values.

Scenario 1: All Endpoints and the processing use the same sampling rate. After collecting the measured values the calculation (difference) was executed.



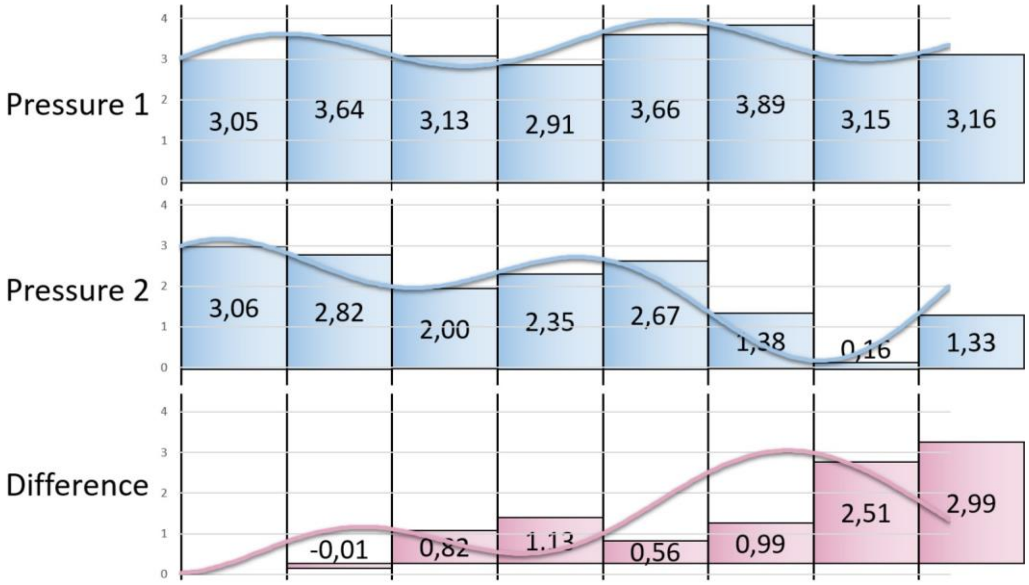
Result: The real and the calculated difference are the same at all times.

Scenario 2: Different sampling rates of the Endpoints and the calculation:



Result: The calculated difference differs (at the time of the calculation of the difference) compared with the real difference!

Scenario 3: All Endpoints and the calculation use the same sample rate. The calculation of the difference was executed before the collection of the measure values.



Result: The calculated difference differs (at the time of the calculation of the difference) compared to the real difference!

To avoid the bad behavior shown above, consider the following notes, for your selection of the sample rates:

Consider the Nyquist–Shannon sampling theorem. To reconstruct the exact curve from the sampled values, it is necessary to use at least the double frequency of the maximum frequency of the signal.

Do not set the sampling rate too high. That can decrease the performance of the system and gain no benefit.

If you connect several Endpoint values, the different sampling rates should nearly the same or integer multiples of each other.

Take care that the calculation is always after the collection of the measured values. *Notice:* The calculation inside a processing is always top down.

4. Devices – Data Sources

4.1 Beckhoff Device

4.1.1 Introduction to Beckhoff ADS

4.1.1.1 ADS Devices, Message Router, ADS Protocol and ADS Interface

On Beckhoff machines (PC based controls, HMIs, …) there are several server applications running (PLC runtime, NC runtime, …) - these applications are called ‘ADS Devices’ or just ‘Devices’. ADS devices are communicating to other ADS devices - also other machines - by sending messages (data, commands, …) to a local router application (Message Router). The local Message Router sends the messages directly to the destination device if it is on the same machine otherwise he forwards the messages to the Message Router of the remote machine. This transport layer is called ADS Protocol. The interface between ADS devices and the Message Router is called ADS Interface.

4.1.1.2 ADS Protocol and TCP/IP

The ADS protocol is based on TCP/IP. The Message Router of a machine is a TCP/IP server listening on a TCP/IP port. The default port number is 48898.

4.1.1.3 AMS NetId and ADS Port

The Message Router of each machine is identified within the ADS network by its AMS NetId. This id is often a combination of the IP address of the machine and the postfix ‘.1.1’ but it can also be any other combination. Each device within a machine has a separate ADS port. So a device on a machine in the ADS network is identified by the AMS NetId of the machines Message Router and ADS port of the device.

4.1.2 Steps to Configure the Beckhoff ADS as a Device

4.1.2.1 Find out the AMS Net Id of the Message Router of a Machine

Because a machine can have several IP addresses you have to explore the AMS NetId on the target machine.

- TwinCAT2: Start the System Manager, select SYSTEM Configuration - Routing NetId Management: Target NetId
- TwinCAT3: Select the sys tray icon ‘TwinCAT Config Mode’ and start the AMS Router dialog with ‘Router - Change AMS NetId’

4.1.2.2 Find out the ADS Port of a Device

The IoT Gateway normally reads variables from the PLC runtime device. To get the right ADS port of this devices do following steps:

- TwinCAT2: Start the System Manager, select PLC Configuration - IEC1131 - Port (801 is the default port for the PLC runtime device).
- TwinCAT3: Open your PLC project, select the PLC folder, select the PLC device, start context menu item ‘Change ADS Port…’ (851 is the default port for the PLC runtime device).

4.1.2.3 Enter a Route to Allow Access from the IoT Gateway to a Device

The IoT Gateway software communicates with a Beckhoff devices via the ADS layer. Therefore a route has to exist so that the Message Router “knows” how to connect them. The ‘Add Route Dialog’ is started differently in TwinCAT2 and 3:

- TwinCAT2: Start the System Manager, select SYSTEM Configuration - Route Settings - Current Routes
- TwinCAT3: Select the sys tray icon 'TwinCAT Config Mode' and start the AMS Router dialog with 'Router - Edit Routes'.

When the dialog appears press the 'Add…' and enter following parameters:

- 'Route Name': Enter the name of the route here e.g.: IoT Gateway
- 'AmsNetId': Enter the IP address of the IoT Gateway followed by .1.1 e.g.: 192.168.1.254.1.1
- 'Address Info': Enter the IP address of the IoT Gateway e.g.: 192.168.1.254
- Check option 'IP Address'
- Check option 'Target Route - Static'
- Check option 'Remote Route - None'
- Clock button 'Add Route' and then 'Close'

4.2 MLC Data Acquisition Device

4.2.1 Introduction

Together with a MLC the IoT Gateway is able to sample values high frequently, equidistant and gapless.

Here some key features:

- Minimal sampling interval is 1ms
- Maximal number of endpoints is 46
- The data width of the PLC variables to be sampled is restricted to four bytes. Larger datatypes E.g. STRING or LREAL variables are NOT supported.
- You can choose whether the value of an endpoint (with in the IoT Gateway) is a simple datatype or an array (=default). An array can be handled much more efficiently.

4.2.2 Technical Information

MLC Data Acquisition is done by task sharing:

1. With the IoT Gateway user interface the sampling is configured. Here the device (the MLC) must be defined and for each PLC variable which should to be sampled an endpoint has to be created and configured. If the user saves the device configuration the list of variables is sent to the MLC.
2. The MLC samples the variables e.g. in a sampling interval of 1ms, stores the sampled values in a buffer and sends the buffer to the IoT Gateway if it is filled.
3. The IoT Gateway receives the buffer, updates and processes the endpoint values.

4.2.3 Commissioning the MLC

4.2.3.1 Install Firmware and PLC Application

Do following steps:

- Connect the interfaces XF1 and XF2 of the MLC with a short Ethernet cable (simulate a closed Sercos bus).
- Connect the interface XF5 of the MLC (engineering interface) with the LAN which is used by your IoT Gateway and your Engineering PC to access the MLC.
- Start IndraWorks - at least version 14V18 is needed.
- Open the project MLC-Trending-14V18.xiwp.

- Right click the Node IndraMotionMlc1 and select the menu item Properties.
- Select the tab page Hardware/communication and check/change the IP address in the group box Ethernet communication.
- To check the availability of the MLC press start a connection test by pressing the button Execute.
- Check the result in the field Connection test.
- If the connection test succeeded close the dialog with OK.
- If it has failed please check the connection (IP address wrong, Ethernet cable, etc.)
- Right click the Node IndraMotionMlc1 and select the menu item Firmware management: The control information will be retrieved.
- If the firmware version of the control is older than the firmware version of your IndraWorks installation start the download.
- Close the firmware management dialog.
- Right click the node IndraMotionMlc1/Logic/Application and select the context menu item Login.
- The PLC application will be downloaded - after the download process to PLC is in STOP
- Right click the node IndraMotionMlc1/Logic/Application again and select Start

IMPORTANT

Do NOT rename the node Application!

4.2.3.2 Change Sampling Interval

The default sampling interval is 1ms. If you don't need to sample the value in 1 kHz you should increase the sampling interval:

1. Open the project MLC-Trending-14V18.xiwp with IndraWorks.
2. Double click the node IndraMotionMLc1/Logic/Application/Task Configuration/MlcTrending
3. Change the value of Interval from t#1ms e.g. to t#10ms -
   Note: An interval less than 1 ms is NOT allowed!
4. Save the project
5. Right click the node IndraMotionMlc1/Logic/Application and select menu item Login
6. The PLC is stopped and the changes are downloaded
7. Right click the node IndraMotionMlc1/Logic/Application and select menu item Start

4.2.3.3 Switch the Motion Runtime System to Phase BB

Equidistant sampling is only possible in Sercos phase BB (or P2).

The motion runtime is switched to phase BB in these steps:

1. Open the project MLC-Trending-14V18.xiwp with IndraWorks.
2. Right click the node IndraMotionMLc1/Logic/Application and select menu item Login
3. Right click the node IndraMotionMLc1 and select menu item Switch online
4. Right click the node IndraMotionMLc1 and select menu item Synchronize/Download motion configuration from PC to the control
5. Right click the node IndraMotionMLc1 and select menu item Control mode/BB

The PLC runtime system is in RUN mode now, the motion runtime system is in phase BB and the MLC trending function is ready to sample variables.

4.2.4 Configure MLC Data Acquisition on the IoT Gateway

Now you have to create and configure one or more MLC Data Acquisition devices in the IoT Gateway.

4.2.4.1 Create a MLC Data Acquisition Device in the IoT Gateway

- Select ‘Create a new Device’
- Select ‘MLC Data Acquisition’
- A dialog ‘Create Device: MLC Data Acquisition’ appears, set following parameters:
- **Name**: Enter a unique name for your device
- **Description**: Optional: Describe your device here
- **IP Address [:Port] of this IoT Gateway**: Enter the IP address of your IoT Gateway. For the first MLC Data Acquisition device you need not enter a port number, it’s automatically set to 2012.
- **IP Address of the MLC**: Enter the IP address of your MLC here.
- **Username**: If you changed the access rights in the MLC enter your configured username - otherwise keep this entry empty.
- **Password**: If you changed the access rights in the MLC enter your configured password - otherwise keep this entry empty.
- Click ‘Create’ to save your settings.

Continue with chapter ‘4.2.4.3 Add, Change or Delete Endpoints’.

4.2.4.2 Change or Delete a MLC Data Acquisition Device

- Select ‘Devices’ from the home page
- Select ‘Change your Devices’
- Select your MLC Data Acquisition device
- A dialog appears
- You can change the parameters (see above)
- Click ‘Save’ to save your settings.

If you click Delete and confirm the question the device and all its endpoints are deleted.

4.2.4.3 Add, Change or Delete Endpoints

- Select ‘Devices’ from the home page
- Select ‘Change your Devices’
- Select your MLC Data Acquisition device
- The list of all endpoints appears
- If there are no endpoints click Add Endpoint

In the Endpoint dialog you can set following parameters:

- **Name**: Enter a unique name here
- **Description**: Optional: Describe your endpoint here
- **Unit**: Enter the unit of your endpoint
- **Symbol Path**: Enter the symbol path of your PLC variable here.
- **Type**: Select the datatype of your PLC variable
- **Single Value Notification**: Check this option if your endpoint value should be a single value not an array
- **Value Change Trigger**: Select the event when a change trigger is fired.

If you need more information to a parameter click the (i) icon at the end of the line.

Click Save to save the changes.

To delete an endpoint click the Waste Bin icon and confirm the question.

4.3 TCP/UDP Data Acquisition (DAQ) Device

4.3.1 Introduction

For DAQ the high frequent sampling of values is delegated to the so called DAQ device. This device can be a PLC or any other device which is able to sample and send telegrams via TCP or UDP. It stores the sampled values into one of two buffers. If this buffer is full the sampled values will be stored into the second buffer and the first buffer will be sent to the IoT Gateway. There these values are stored into so called DAQ arrays. This format is much more compact than simple data types.

The key features are:

- Minimal sampling interval: 1ms
- Maximal sampling interval: 1000ms (1s)
- Maximal number of values per TCP/UDP Data Acquisition instance on the IoT Gateway: 250 Multiple instances are possible.
- The data width of the values is restricted to four bytes. Larger datatypes E.g. STRING or LREAL variables are NOT supported.

4.3.2 Technical Information

The telegram sent to the IoT Gateway has a header and a value buffer. The header has to contain following information (here the default allocation):

- Message ID, incremented by 1: n, n+1, n+2,n+3,.. (byte offset 0, data type int32)
- Length of the telegram (byte offset 4, data type int32)
- Number of records: At each sample time all values are stored into one record (byte offset 8, data type int16)
- Number of values per record (byte offset 10, data type int16)
- Sampling time in ms, min=1, max=1000 (byte offset 12, data type int16)
- Length of the header (here 24) (byte offset 14, data type int16)
- Trigger: 0 values are invalid, <>0 values are valid (byte offset 16, data type int16)
- byte 18..23 are reserved

This allocation can be changed in the device configuration on the IoT Gateway (see below).

4.3.3 Configure TCP/UDP Data Acquisition on the IoT Gateway

4.3.3.1 Create a TCP/UDP Data Acquisition Device in the IoT Gateway

- Select ‘Devices’ from the home page
- Select ‘Create a new Device’
- Select ‘TCP/UDP Data Acquisition’
- A dialog ‘Create Device: TCP/UDP Data Acquisition’ appears, set following parameters:
- **Name**: Enter a unique name for your device
- **Description**: Optional: Describe your device here
- **Port**: Enter the port where your IoT Gateway should listen for telegrams.
- **UDP**: Check this option if your DAQ device sends via UDP - BUT: We recommend TCP
- **Endianness**: Select the kind of byte ordering your DAQ device will sent the values.
- **Header Offset ‘Message Id’**: The byte offset of the Message Id

- **Header Offset 'Message Length'**: The byte offset of the Message Length
- **Header Offset 'Number of Records'**: The byte offset of the Number of Records
- **Header Offset 'Number of Values'**: The byte offset of the Number of Values
- **Header Offset 'Sampling Interval'**: The byte offset of the Sampling Interval
- **Header Offset 'Header Length'**: The byte offset of the Header Length
- **Header Offset 'Trigger'**: The byte offset of the Trigger
- **Auto generate Endpoints**: If this option is set and there are no endpoints configured endpoints are created automatically when the first telegram arrives. So don't hit the button 'Add Endpoint'.
- **Auto generated Endpoints Type**: Here you can select the data type of the automatically created endpoints.
- Click 'Create' to save your settings.

Continue with chapter '4.2.4.3 Add, Change or Delete Endpoints'.

4.3.3.2 Change or Delete a TCP/UDP Data Acquisition Device

- Select 'Devices' from the home page
- Select 'Change your Devices'
- Select your TCP/UDP Data Acquisition device
- A dialog appears
- You can change the parameters (see above)
- Click 'Save' to save your settings.

If you click Delete and confirm the question the device and all its endpoints are deleted.

4.3.3.3 Add, Change or Delete Endpoints

- Select 'Devices' from the home page
- Select 'Change your Devices'
- Select your TCP/UDP Data Acquisition device
- The list of all endpoints appears

Hint: Don't click the button 'Add Endpoint' if you have selected the option 'Auto generate Endpoints'

In the Endpoint dialog you can set following parameters:

- **Name**: Enter a unique name here
- **Description**: Optional: Describe your endpoint here
- **Unit**: Enter the unit of your endpoint
- **Type**: Select the datatype of your DAQ value
- **Value Number**: Number of the value within the telegram (1…).
- **Bit Position**: If Type is Bit: Number of the bit within the 32 bit value (0..31).
- **Bipolar Signal: +-10V, 20mA or +-5V**: If Type is S20-AI: Check this option if your signal is bipolar (+-10V, +-5V, 20mA).
- **Measuring Lower Bound**: Enter the Measuring Lower Bound value.
- **Measuring Upper Bound**: Enter the Measuring Upper Bound value.
- **Value Change Trigger**: Select the event when a change trigger is fired.

If you need more information to a parameter click the (i) icon at the end of the line.

Click Save to save the changes.

To delete an endpoint click the Waste Bin icon and confirm the question.

4.4 XM Logic Device

With a XM Logic Device the variables configured within an XLC/MLC 2G controller (as configured within the symbol configuration) can be stored in Endpoints. The local controller (IoT Gateway on XM platform) or a controller accessible via TCP/IP can be used.

Every Endpoint can be scanned using a configurable sampling interval between 100ms and 1 hour. To browse the accessible variables it is recommended to use the Bosch Rexroth UA-Testclient.

Hint: The Function XM Logic Device is not included in the provision type Docker image.

4.5 MQTT Device

Subscribes for Topics of a **MQTT Broker** and exposes the messages as Endpoint values

4.5.1 Introduction to MQTT

We recommend to read the http://www.hivemq.com/mqtt-essentials/ to get a short overview over the feature set of **MQTT** communication including an explanation of some parameters used in this device.

4.6 OPC DA

Provides an OPC DA client to cyclically read values of an OPC DA server and store these values in Endpoints. Servers of version 1.x and 2.x are supported.

OPC DA stands for OLE for Process Control (Data Access) and is the specification to transfer values via OPC. OPC DA is the first OPC specification and is based in the COM/DCOM Model from Microsoft. Every OPC DA Client can sample on an update rate from 1 to 64 seconds or 1 hour. All Endpoints will be updated within this rate. To browse the accessible variables of an OPC DA server it is recommended to use the Rexroth OPC DA client.

4.7 OPC UA

Provides an OPC UA client that can acyclically read values of an OPC UA Server and store them into Endpoints.

OPC Unified Architecture (shortage: OPC UA), is an industrial M2M communication protocol representing an extension of the OPC Data Communication Protocol. As the latest of all OPC specifications of the OPC Foundation it differs from its predecessors, mainly throughout the ability to not only transport machine data, but to describe these semantically. Since the platform independency (runs on every operating system) and its improved security concepts it is future proof for integration into existing systems.

Every Endpoint is updated with a configurable Sampling-Interval from 100ms to 1 hour.

To browse the accessible variables of an OPC UA server the Rexroth OPC UA client can be used.

4.8 Allen Bradley ControlLogix Device

4.8.1 Introduction

The ControlLogix device represents a TCP/IP based Ethernet IP client to connect controllers of the *Allen Bradley ControlLogix* family using the Common Industrial Protocol (CIP). CIP is a media independent Top level protocol using a producer-consumer communication model, and is a strictly object oriented at the upper layers. Each CIP object has attributes (data), services (commands), connections and behavior (relationship between attribute values and services).

For Further Information on Ethernet IP and CIP please visit https://www.odva.org/.

4.8.2 Supported Data Types

In a ControlLogix controller each item to read is called a *Tag*. Following the Rockwell specification the ControlLogix device is capable of using the Read Tag Service to read data of the following types

Data type	ControlLogix Type	Size
BOOL	0x0nC1	1 Byte
SINT	0x00C2	1 Byte
INT	0x00C3	2 Byte
DINT	0x00C4	4 Byte
REAL	0x00CA	4 Byte
STRING	0x0FCE	8 Byte

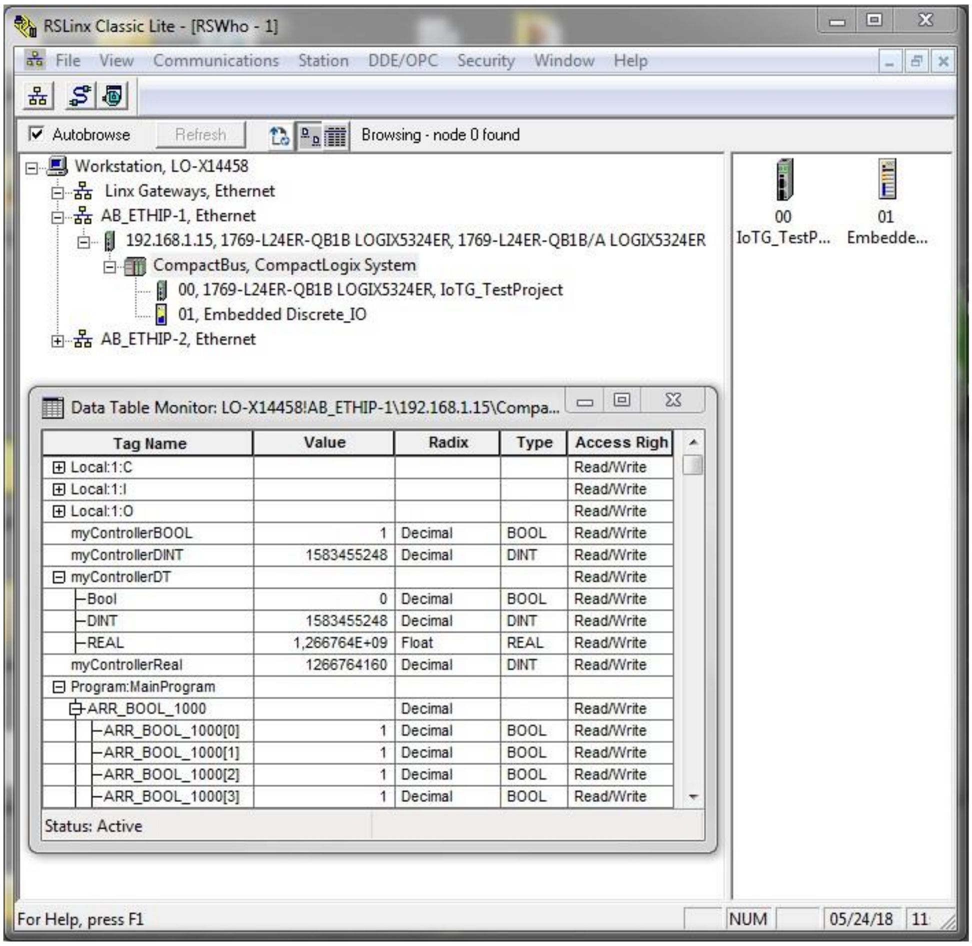
4.8.3 Sampling

Each tag is addressed as single read. The Sampling interval defines the time interval in which the configured tag shall be read.

Following Sampling intervals are available: 100ms, 200ms, 500ms, 1s, 2s, 5s, 10s, 1 min, 5 min, 30 min, and 1h

4.8.4 Usage

Tags are addressed as string following the semantic in the Data Table Monitor of *Rockwell RS Linx Classic Lite* Software. It browses the variables on a controller and can be downloaded from https://www.rockwellautomation.com.



Workstation, LO-X14458
Linx Gateways, Ethernet
AB_ETHIP-1, Ethernet
192.168.1.15, 1769-L24ER-QB1B LOGIX5324ER, 1769-L24ER-QB1B/A LOGIX5324ER
CompactBus, CompactLogix System
00, 1769-L24ER-QB1B LOGIX5324ER, IoTG_TestProject
01, Embedded Discrete_IO
AB_ETHIP-2, Ethernet

Data Table Monitor: LO-X14458!AB_ETHIP-1\192.168.1.15\Compa...

Tag Name	Value	Radix	Type	Access Righ
⊞ Local:1:C				Read/Write
⊞ Local:1:I				Read/Write
⊞ Local:1:O				Read/Write
myControllerBOOL	1	Decimal	BOOL	Read/Write
myControllerDINT	1583455248	Decimal	DINT	Read/Write
⊟ myControllerDT				Read/Write
├Bool	0	Decimal	BOOL	Read/Write
├DINT	1583455248	Decimal	DINT	Read/Write
├REAL	1,266764E+09	Float	REAL	Read/Write
myControllerReal	1266764160	Decimal	DINT	Read/Write
⊟ Program:MainProgram				
⊟ARR_BOOL_1000		Decimal		Read/Write
├ARR_BOOL_1000[0]	1	Decimal	BOOL	Read/Write
├ARR_BOOL_1000[1]	1	Decimal	BOOL	Read/Write
├ARR_BOOL_1000[2]	1	Decimal	BOOL	Read/Write
├ARR_BOOL_1000[3]	1	Decimal	BOOL	Read/Write

Status: Active

For Help, press F1

As shown each value can be addressed using the following scheme:

```
LogicalUnitType:LogicaUnitSubtype:LogicalUnitName.ParentItem[IndexIfArray].NameOfTheChildItem
```

Examples:

`MyLocalBoolean` Reads the global variable named “MyLocalBoolean”

`MyArray[0]` Reads the first element of an global array named “MyArray”

`MyStructure.MyIntegerMember` Reads the member named “MyIntegerMember” of the global structure named “MyStructure”

`MyArrayOfStructure[2].MyRealMember` Reads the member named “MyRealMember” of the 2. Element of the global array of structure named “MyArrayOfStructure”

`Program:MyProgram.MyBoolean` Reads the Variable named “MyBoolean” declared in a Program named “MyProgram”

`Local:1:C` Reads local Input module 1 named “C”

4.9 Option Module S20 Analog Input

This device option provides an analog Input module, which can read values of a local S20 Analog-Input module and store them into Endpoints.

The Device is available for the IndraControl XM controller or PR21 IPC with the S20-ETH-BK bus coupler only.

Supported Modules

Product Key	Description	Material number
S20-AI-4-I	IndraControl S20-analog-input module – 4 DC Inputs, incl. bus socket module	R911173249
S20-AI-4-U	IndraControl S20-analog-input module – 4 voltage inputs, incl. bus socket module)	R911173256

4.10 Option Module S20 Digital Input

This device option provides a digital input module that can read values from a local S20 Digital-Input module and store them into Endpoints.

The Device is available for the IndraControl XM controller or PR21 IPC with the S20-ETH-BK bus coupler only.

Supported Modules

Product Key	Description	Material Number
S20-DI-16/1	IndraControl S20-Digital-input module, 16 inputs, incl. bus socket module	R911172543

4.11 S20-ETH-BK Device

4.11.1 Introduction

With the IoT Gateway you can read analog and digital values from the input modules of one or more S20-ETH-BK devices very easily:

- Once created as a device within the IoT Gateway the module configuration of the S20-ETH-BK is scanned and all endpoints are created automatically by the IoT Gateway software.
- Your job is to adjust the name and the signal and measuring bounds.
- If the plug-and-play mode is enabled you can add and/or remove modules online.

4.11.2 Commissioning Your S20-ETH-BK

4.11.2.1 S20-ETH-BK Data Sheet

The S20-ETH-BK data sheet contains important information about commissioning and maintenance. It can be downloaded from here:

Download document: R911372205_02 “Data Sheet S20-ETH-BK”

4.11.2.2 Basic Steps

The most important steps are:

- Plug your input modules[1] (max. 63) into the local bus of the S20-ETH-BK

[1] Following module types are supported:

```
S20-DI-16/1
S20-DI-32/1
```

- Connect the input clamps of your modules with the sensors
- Connect your S20-ETH-BK and your modules to 24V+ power supply
- Choose the IP settings for your S20-ETH-BK with the switches S1 and S2 (see S20-ETH-BK data sheet) [2]
- Connect your S20-ETH-BK to your Ethernet network (XF7 and/or XF8)
- The S20-ETH-BK supports web based management. So start a web browser (regard your internet settings) and enter the URL e.g. http://192.168.1.124

Attention: The initial password is: `private`

IMPORTANT

The input modules have to run with their default settings - see parentheses above. One prefers sensors with 0..20 mA signal because they have a better EMC than 0..10 V.

4.11.2.3 Plug-and-Play Mode

Per default on the S20-ETH-BK the plug-and-play mode is enabled. During commissioning this provides an ease of use for adding or removing physical S20 modules because the according endpoints are automatically created or deleted.

After commissioning we recommend disabling the plug-and-play mode. The online configuration is not scanned permanently - this increases the performance. So you can choose a smaller sampling interval for reading the input values and updating the endpoints.

4.12 S5 Device

4.12.1 Introduction

Reads values of a Siemens S5 controller and stores them into Endpoints. Values of a Datablock-, M-Memorizer, Input-, Output-, Counter- or Timer- field can be read.

4.12.2 Specification

1. S5-Controllers are connected via the PG-SS interface. To accomplish this the S5-LAN adapter of the company *Traeger* is used: https://www.traeger.de/industrial-ethernet/s5-lan.html.
   This device converts Ethernet to 20mA-TTY physically (serial, 9600Bd) and from protocol side RFC1006 (S7-Protocol) to AS511 (S5-Protokoll).
2. Thus all S5-Device types are supported.
3. The following Data areas can be accessed: DB, M-Memorizer, Inputs, Outputs, Timer, Counter
4. The following datatypes are supported: BOOL (KM), BYTE (KY), WORD (KH), DWORD (DH), INT (KF), DINT (DF), REAL (KG), S5TIME (KT), S5COUNTER (KZ), CHAR (KC), BYTE[], STRING (KS)
5. While using a serial interface the sampling rate of the variables is limited. Only one variable can be read within a single read request (typically 200ms reading type BYTE, 600ms reading type BYTE[]).

```
S20-DI-64/1
S20-DI-16/1-HS
S20-DI-16/4
S20-AI-4-I (Filter 30Hz, Mean-Value 16-sample, Signal Range 0..20 mA)
S20-AI-4-U (Filter 30Hz, Mean-Value 16-sample, Signal Range 0..10 V)
S20-AI-8 (0..10 V)
```

[2] Be sure that the IP address does not change - we recommend a static IP address.

4.13 S7 Device

4.13.1 Introduction

Reads values of a Siemens S7 control and stores them into Endpoints. Values of datablocks-, memorizer-, periphery inputs and outputs and also timer and counter can be read.

4.13.1 Specification

1. Supported device-types: S7 of the PLC series 300/400 and 1200/1500
2. The S7 is connected to the system via Ethernet. If there is no Ethernet port, an Ethernet/MPI adapter is required. e. g.:
   S7-LAN of the company *Traeger*: http://www.process-informatik.de/produkte/s7-lan&mt=1
   S7++ of *IBH-Softech*: https://www.ibhsoftec.com/IBH-Link-S7-PP
3. The protocol RFC1006 (ISO on TCP) is used.
4. There are some special characteristics of the 1200/1500 types:
   - GET/PUT has to be enabled
   - DBs, which are stored in a memory optimized format on the S7 control, need to be converted. **In this case the native PLC project need to be present. A transfer of datablocks which are not memory optimized can affect the automation process.**
5. The following data types are supported: BOOL, BYTE, WORD, DWORD, DINT, REAL, CHAR, BYTE, STRING (on S7 limited to 254 chars).

4.13.2 S7-1200/1500 Specific Settings

To connect to a S7-1200 or -1500 type following settings must be done:

PLC settings:

- Protection: Select ‘Full access (no protection)’
- Connection mechanism: Check ‘Permit access with GET/PUT communication …’

Datablocks (DBs):

- Only **global** DBs can be accessed.
- Attributes of the DB: **Uncheck ‘Optimized block access’**

4.14 S/IP Device

Hint: S/IP Device support S/IP ReadOnlyData Service in single read access.

4.15 XM System Device

With this device, system specific data for analyzing purpose like CPU-Load can be read and stored as Endpoints.

Hint: This device is available on the IndraControl XM platform only.

4.16 XDK

Before configuring the IoT Gateway for XDK connectivity, please complete the steps in the XDK workbench. More details about programming the XDK sensor platform please find under http://xdk.io

4.16.1 Create an XDK Device

- Go to IoT Gateway homepage
- Click on **Devices**

- Click on the icon Create a new Device
- Select XDK Device
- A dialog with heading **Create Device: XDK Device** will appear
- **Device Name:** Unique name of the XDK Device which sends the sensor data
- **Device Address:** IP address of the XDK device
- **Device Port Number:** Port number in which the XDK Device sends data
  Please note that the number should be the same as the port number in XDK Workbench
- Click **Create** to store your parameters

4.16.2 Change/Delete of the XDK Device

- Select **Devices** from the home page
- Expand the icon **Change your Devices**
- Select the device you want to configure or delete
- Enter/change the parameter(s)
- Click **save** to save your changes.
- Select **delete** for deleting your configuration.

4.16.3 Receive Data from Sensors

- Select **Devices** from the home page
- Expand the icon **Change your Devices**
- Select your saved device
- In the icon **XDK Endpoint** click the + symbol
- **Name:** Name of the sensor for example: temperature, pressure, gyroscope measured by XDK
- **Endpoint Id:** ID of the sensor data
- **Symbol name:** Choose the sensor data of your interest and press **Save**

  Hint:
  - **To receive further sensor, click + symbol and repeat the above steps.**
  - **To edit sensor data configuration, click ... symbol.**
  - **To remove a sensor data, click - symbol and press save.**

4.16.4 Steps to Be Followed in the XDK Workbench

- Download project:
  http://xdk.bosch-connectivity.com/xdk_docs/html/dir_af4b04ffbf4fbfcc9675780947b8b97b.html
- Open the file **SendAccelDataOverUdpAndBle** in the XDK workbench.
- Go to source > **SendAccelDataOverUdpAndBle.h**
- Check on the left side for ? symbol yellow in color followed by the sentence
  Hint: Please provide WLAN related configuration, with valid SSID & WPA key and server IP address where packets are to be sent in the macros
- In the next line, type the WLAN SSID within double Quotes.
- In the following line, type the WLAN password
- In the next line, type the IP address of the IoT Gateway in hexadecimal format
- Finally, type the port number on which the Gateway would listen and save.
- Now flash the file **SendAccelDataOverUdpAndBle**
- If the build is successful, the dynamic IP address of the XDK Device is displayed (an example IP address could be 192.168.1.210).
- The data collection interval or sampling interval is set to 100ms – could be changed in project.

5. Functions

5.1 Average

Supports various averages over a rolling window of buffered values. Useful for smoothing of raw data.

- **Arithmetic Mean** - is the sum of a collection of numbers divided by the count of numbers in the window.
- **Quadratic Mean (Root Mean Square)** - is the square root of the mean of the squares in the window. Also known as the Effective Value (RMS).
- **Geometric Mean** - is defined as the nth root of the product of n numbers in the window. Higher values are weighted less.

5.2 Math Calculator

Supports a simple math calculator for basic operations as addition, subtraction, multiplication, division.

5.3 Dead Band

Supports a *relative* Dead Band (Neutral Zone, Hysteresis) in *percent* or as an *absolute value*.

- **Percent:** The output value does not change, until the input value exceeds a *relative* threshold in *percent*.
- **Absolute:** The output value does not change, until the input value exceeds a *relative* threshold as an *absolute value*.

5.4 Factor

Supports a linear factor computation: **y= a*x + b**

5.5 Formatter

The formatter function implements a mustache template system that allows you to generate a dynamic user defined string (e.g. JSON or XML) including your configured input values.

The formatter allows you to embed Endpoint values into a freely definable protocol. The resulting string is again stored as an Endpoint and can be provided to superior services.

The Formatter uses the template system Mustache.

The following example creates an array over all selected inputs and stores the appropriate value.

```
[{{#inputs}} # Start for-loop over all selected inputs with leading ‘[‘`
  {{name}}; # Write name of endpoint with trailing ‘;’
    {{#values}} # Start for-loop over all values from actual input
    {{value}}; # Write value of endpoint with trailing ‘;’
  {{/values}} # End for loop over values
{{/inputs}}] # End for loop over inputs with trailing ‘]’
```

Example:

If the following Endpoints have been selected

Endpoint	Endpoint value
Temperature	24.2
Pressure1	1100
Pressure2	1080
Difference	20

the following string is generated:

```
[Temperature; 24,2; Pressure1; 1100; Pressure2; 1080; Difference; 20;]
```

5.6 Introduction to Mustache

The captured values are stored and transferred internally to the mustache engine regarding following manner:

5.6.1 Internal Hash

```
{
inputs: [
        name: EndpointName,
        EndpointName: true,
        lastInput: false/true,
        values: [
            value: 0.1234,
            time: 2018-04-01T12:34:56.789Z,
            lastValue: false/true,
            valueAsBool: false/true;
            ...
         ],
         ...
    ],
    EndpointName.value: 0.1234,
    EndpointName.time: 2018-04-01T12:34:56.789Z
    ...
}
```

On this internal mustache hash it is evident, that only the keys *inputs* and *values* are *arrays*. The right side of the remaining key value pairs are either scalar Boolean (*EndpointName*, *lastInput*, *lastValue*, *valueAsBool*) or individual values like *name*, *value* or *time*.

When the key is an array, the mustache syntax {{#...}} ... {{/...}} defines a for loop over each item in the array.

If the key is a Boolean scalar like the keys *EndpointName*, *lastInput*, *lastValue* ans *valueAsBool*, then the mustache syntax {{#…}} … {{/…}} defines an if statement.

The left keys are *name*, *value*, *time*, *EndpointName.value* and *EndpointName.time* and can be replaced by his corresponding value over the mustache tag {{…}}.

On the position of the keys in the internal hash it is evident, that the keys *EndpointName.value* and *EndpointName.time* are in the root of the internal hash and can therefore be replaced without further tags directly with {{EndpointName.value}} and {{EndpointName.time}}.

Beside this two keys, only the key *inputs* is in the root of the internal hash. In the array key *inputs* are in turn the keys *name*, *EndpointName*, *lastInput* and *values*. In the array key *values* are in turn the keys *value*, *time*, *lastValue* and *valueAsBool*.

With subsequent available mustache syntax it is possible to structure your individual string like JSON or XML.

5.6.2 FOR Statement

Take care that the *values* statement has to be embedded into the *inputs* statement

```
{{#inputs}}...{{/inputs}}

{{#values}}...{{/values}}
```

5.6.3 IF statement

```
{{#EndpointName}}...{{/EndpointName}}

{{#lastInput}}...{{/lastInput}}

{{#lastValue}}...{{/lastValue}}

{{#valueAsBool}}...{{/valueAsBool}}
```

5.6.4 IF NOT statement

```
{{^EndpointName}}...{{/EndpointName}}

{{^lastInput}}...{{/lastInput}}

{{^lastValue}}...{{/lastValue}}

{{^valueAsBool}}...{{/valueAsBool}}
```

5.6.5 INSERT

```
{{name}}

{{value}}

{{time}}
```

5.6.6 Simple Scalar INSERT

```
{{EndpointName.value}}

{{EndpointName.time}}
```

5.6.7 Example

This example generates a JSON with all available configured inputs and values. Let’s assume we have selected two inputs with the names `random` and `sawtooth`. The input `random` is scanned with 500 ms and the input `sawtooth` is scanned with 1 s. If the formatter processing is called every second, we expect an array of two `random` values and one `sawtooth` value every second. So the internal mustache hash will look like subsequent example.

5.6.7.1 Internal Hash

```
{
   inputs: [
          name: random,
          random: true,
          lastInput: false,
          values: [
              value: 0.1234,
              time: 2018-04-01T12:34:56.789Z,
              lastValue: false,
              valueAsBool: false;
              value: 1.2345,
              time: 2018-04-01T12:34:57.289Z,
              lastValue: true,
              valueAsBool: false;
          ],
          name: sawtooth,
          sawtooth: true,
          lastInput: true,
          values: [
              value: 7.543,
              time: 2018-04-01T12:34:56.789Z,
              lastValue: true,
              valueAsBool: false;
          ],
     ],
     random.value: 1.2345,
     random.time: 2018-04-01T12:34:57.289Z,
     sawtooth.value: 7.543,
     sawtooth.time: 2018-04-01T12:34:56.789Z,
}
```

To generate a valid JSON-Array with a correspondent mustache syntax, we have to start and end our JSON-String with brackets. After the leading brackets we start, and before the terminating brackets we limit the loop over all inputs with the mustache syntax *{{#inputs}} … {{/inputs}}*.

```
[
        {{#inputs}}
        ...
        {{/inputs}}
]
```

In this loop we insert at first the name of the inputs *"name":"{{name}}"* and after this the values itself. Because the values must not be a scalar, we limit the values with brackets and define then a second loop over all values with {{#values}} … {{/values}}.

```
[
        {{#inputs}}
            {
               "name":"{{name}}",
               "values":[
                   {{#values}}
                        ...
                   {{/values}}
               ]
            }{{^lastInput}},{{/lastInput}}
        {{/inputs}}
]
```

With the mustache syntax *{{^lastInput}},{{/lastInput}}* we can achieve, that we separate the single inputs with a comma and didn't make a comma after the last input in the array. At least we inserting the key/value pairs for the single values and its corresponding time, and separate the single entries an additional time with the *if not* syntax *{{^lastValue}},{{/lastValue}}*.

```
[
        {{#inputs}}
            {
                "name":"{{name}}",
                "values":[
                    {{#values}}
                         {"value":{{value}},
                         "time":"{{time}}"
                         }{{^lastValue}},{{/lastValue}}
                    {{/values}}
                ]
            }{{^lastInput}},{{/lastInput}}
        {{/inputs}}
]
```

With this individual mustache template, and above mentioned configuration, we will get subsequent generic JSON-Array with actual endpoint values.

```
[{
    "name": "Random1",
    "values": [{
         "value": 0.1234,
         "time": "2018-04-01T12:34:56.789Z"
     }, {
         "value": 1.2345,
         "time": "2018-04-01T12:34:57.289Z"
     }]
}, {
     "name": "Sawtooth1",
     "values": [{
          "value": 7.543,
          "time": "2018-04-01T12:34:56.789Z"
     }]
}]
```

5.7 Limiter

Provides a simple limiter (value cut). If the input value is lower than the lower bound, the output value equals the lower bound.

5.8 Nelson Rules Monitor

Provides a Nelson Rules monitor for value out-of-control detection. If the selected rule is broken, the output value equals **true**, else **false**.

5.8.1 Nelson Rules Explanation

“Nelson rules are a method in process control of determining if some measured variable is out of control (unpredictable versus consistent). Rules, for detecting out-of-control or non-random conditions were first postulated by Walter A. Shewhart in the 1920s. The Nelson rules were first published in the October 1984 issue of the Journal of Quality Technology in an article by Lloyd S Nelson.”

Source: https://en.wikipedia.org/wiki/Nelson_rules

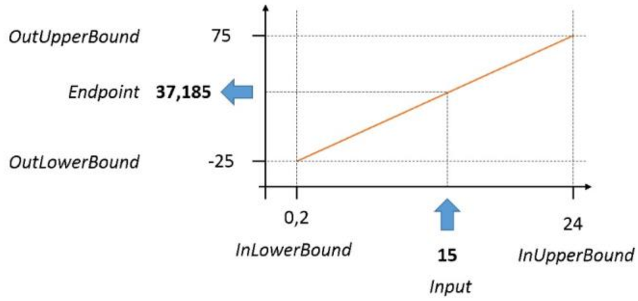
Rules:

- Rule 1: One point is more than 3 standard deviations from the mean.
- Rule 2: Nine (or more) points in a row are on the same side of the mean.
- Rule 3: Six (or more) points in a row are continually increasing (final or decreasing).
- Rule 4: Fourteen (or more) points in a row alternate in direction, increasing then decreasing.
- Rule 5: Two (or three) out of three points in a row are more than 2 standard deviations from the mean in the same direction.

- Rule 6: Four (or five) out of five points in a row are more than 1 standard deviation from the mean in the same direction Rule 7: Fifteen points in a row are all within 1 standard deviation of the mean on either side of the mean.
- Rule 8: Eight points in a row exist, but none within 1 standard deviation of the mean, and the points are in both directions from the mean.

5.9 Range Converter

Provides a linear conversion from a given input value range to another output value range using the two-point form of the line equation: **y = (y2 - y1) / (x2 - x1) * (x - x1) + y1**



5.10 Threshold Monitor

Provides a threshold monitor for value limit detection. If the input value is lower than the lower bound or higher than higher bound, the Boolean output value equals **true**, else **false**. Provides Optional warning if value is outside given range

5.11 Timer

Triggers a timed signal depending on the value of a certain Endpoint.

5.11.1 Inputs and Outputs

5.11.1.1 Inputs

5.11.1.1.1 Input Endpoint

The Input Endpoint defines the value on which a timed signal shall be triggered.

Input values can be of a numeric type only. Every value will be interpreted as Boolean value in the following manner:

```
Greater than 0: True

Lower equal  0: False
```

Other types of values result in an error message. Every action is triggered as soon as a rising or a falling edge on the *Input Endpoint* has been detected, depending on the configured type (ref. 5.11.1.1.3).

5.11.1.1.2 Delay

The time delay can be configured here. The time delay has to be defined in [ms].

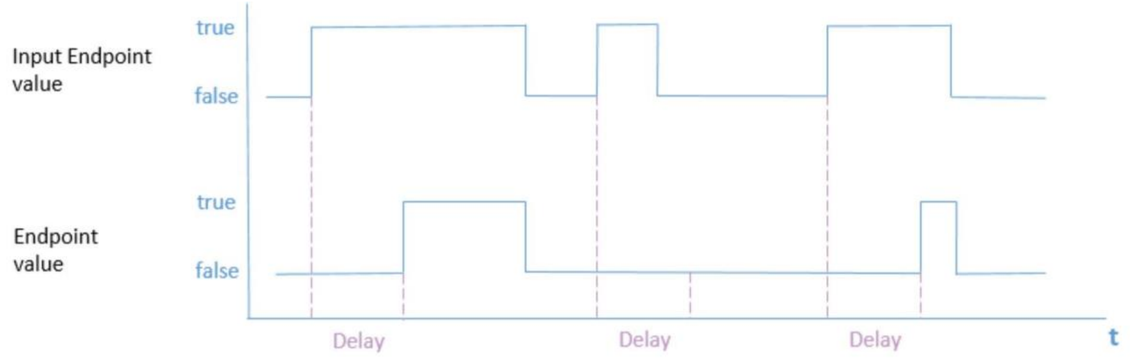
5.11.1.1.3 Type

The type of delay - There are two types of delay possible:

- On delay
- Off Delay

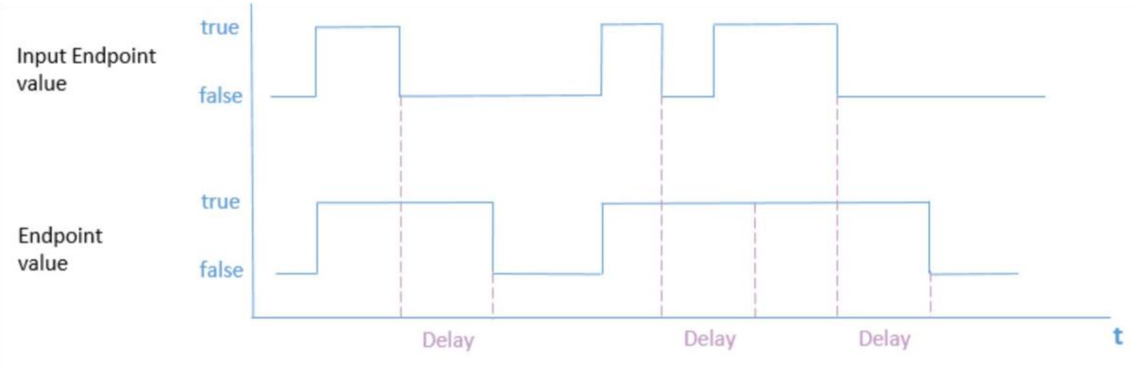
See next pages for explanation.

5.11.1.1.3.1 On Delay



The value of the output *Endpoint* is set to true Delay [ms] time after a rising edge on the *Input Endpoint* has been detected. It is set to false as soon as a falling edge on the *Input Endpoint* has been detected. In case the period between a rising edge and a falling edge on the *Input Endpoint* is shorter than Delay [ms] time the value of the output *Endpoint* does not change.

5.11.1.1.3.2 Off Delay



The value of the output *Endpoint* is set to true as soon as a rising edge has been detected on the *Input Endpoint*. It is set to false Delay [ms] time after a falling Edge has been detected on the *Input Endpoint*. In case the period between a falling edge and the following rising edge on the *Input Endpoint* is shorter than the configured Delay [ms] time the value of the output *Endpoint* does not change.

5.11.1.1.4 Invert Output

Inverts the resulting value of the output *Endpoint*.

5.11.1.2 Outputs

5.11.1.2.1 Endpoint

The output *Endpoint* represents the resulting value of a timer calculation.

6. Applications

6.1 OPC UA Server Application

6.1.1 What Is the OPC UA?

OPC Unified Architecture (UA) is the new generation of OPC. Interoperability and Standardization are what OPC is all about. While conventional OPC solved the device interoperability problem at the Control level, the demand for that same level of standardization was required for the Enterprise layer. Classic OPC is based on the Microsoft DCOM which can introduce vulnerabilities at these layers. Urgency to find simplicity, maximum interoperability, and security drove the OPC foundation into creating a unified method of data communication for parts of the existing OPC specifications for DA, HDA, A&E, and Security.

OPC Unified Architecture extends the highly successful OPC communication protocol, enabling data acquisition and information modeling and communication between the plant floor and the enterprise reliably and securely.

The key features and benefits for OPC UA are:

- Platform neutrality to run on any operating system
- Future-ready and Legacy-friendly
- Easy configuration and maintenance
- Service-based technology
- Increased visibility
- Broader scope of connectivity
- Higher Performance

6.1.2 IoT Gateway OPC UA Server

The integrated **IoT Gateway UA Server** exposes all available process data (Endpoints) from all configured sources (called Producers or Devices) into its address space for easy UA Client access and many more.

6.1.3 Server Endpoints

Connect to the OPC UA Server using any UA Client (e.g. the Bosch Rexroth UA TestClient for testing purpose available from Bosch Rexroth Engineering Network - free registration)

6.1.3.1 Secure Endpoints (Recommended)

[Basic128Rsa15, SignAndEncrypt]

```
opc.tcp://{IP}:9999
```

[Basic256, SignAndEncrypt]

```
opc.tcp://{IP}:9999
```

[Basic256Sha256, SignAndEncrypt]

```
opc.tcp://{IP}:9999
```

IMPORTANT

After trying to establish the first secure UA connection, usually the UA Server rejects your certificate if not trusted (e.g. if a self-signed certificate provided). The client receives the bad connection response “BadSecurityChecksFailed”.

You have to tell the OPC UA Server, that your Client can be trusted anyway:

1. Open the IoT Gateway Webpage, navigate to the Certificate Management
2. Trust the rejected certificate manually after your personal validation (certificate can be downloaded for review)
3. Restart your UA Client Application or retry to connect

6.1.3.2 Unsecure Endpoints

Note: This option is DISABLED by default!
[None]

```
opc.tcp://{IP}:9999
```

IMPORTANT

Unsecure Endpoints are disabled by default. We strongly encourage to use any of the available Secure Endpoints in your industrial environment (described before). Enabling **Unsecure Endpoints** is for your own risk and potentially able to damage your system or make it vulnerable. It also enables unencrypted communication, which offers easy sniffing of your process data.

By enabling **Unsecure Connections**, you understand and agree that issues may be encountered without prior warning that may impact your device.

Please notice that:

- **Bosch Rexroth does not give any warranties, whether express or implied, as to the usability of Unsecure Connections.**
- **Bosch Rexroth will not be liable for loss or damage of any kind.**

Open the **OPC UA Server Application** configuration page and enable the option **Unsecure Connections**.

6.1.4 Security

6.1.4.1 Certificates

The Server certificate is **SELF-SIGNED**, so every client has to trust it by code.

6.1.4.2 Authorization / Authentication

The Server will accept anonymous clients, because of no built-in authentication and authorization.

6.2 Endpoint Options

6.2.1 Endpoint Statistics

If enabled, each exposed Endpoint is also exposing commonly used Statistic nodes as children using a moving Statistic Window over most recent number of n values `(Statistic Window Size)`. You can control the update of the Statistics by setting the `SamplingInterval` of the Endpoint in your IoT Gateway configuration.

6.2.1.1 Statistic Values

- Minimum
- Maximum
- Mean (Moving Average)
- Quadratic Mean (Effective Value, RMS, Root Mean Square; Bigger values are weighted more)

- Geometric Mean (Bigger values are weighted less)

The Endpoint value Statistics is disabled by default. Open the **OPC UA Server Application** `https://{IP}:8888/system/console/configMgr`, and enable Statistics and set the Statistic Window Size according to your needs.

IMPORTANT

The Statistic values will be calculated after each value change, which may lead to more CPU load and memory consumption.

6.2.2 Endpoint History

If history was enabled, each exposed Endpoint is collecting his history up to a capacity of n (`History Capacity`) values. The value History can be read using any UA historyRead* service call from any OPC UA Client. You can control the update of the History by setting the `SamplingInterval` of the Endpoint in your IoT Gateway configuration.

Estimated History capacity for a given sampling interval

```
History Depth [h] = SamplingInterval [ms] / 3600
```

Desired sampling interval for a given History capacity

```
SamplingInterval [ms] = History Depth [h] * 3600
```

The Endpoint value History is disabled by default.

Open the OPC UA Server Application `https://{IP}:8888/system/console/configMgr`, enable and set the History Capacity according to your needs.

IMPORTANT

The History may lead to more memory consumption.

6.3 IndraControl XM

6.3.1 Creating a Customized IndraWorks PLC Application Running in the IoT Gateway

There are a lot of reasons to run a customized PLC application on your IoT Gateway (e.g. special IO configuration or control logic, data sampling etc.).

6.3.2 Restore Template Archive

To create such a PLC application please do following steps:

1. Start IndraWorks Engineering and restore the archive IoT_Gateway_Template.zip on the OEM partition of your IoT Gateway.
2. Right click the device “IoT_Gateway” and select menu item “Properties…”
3. Select the tab page “Hardware/communication” and change the parameter “IP address”.
4. Before you close the dialog with OK please execute the connection test and verify that the control is reachable.
5. Now you can change the application.

6.3.3 Important Hints When Adapting your PLC Application

- Don’t change the task configuration “PlcTask”. If you need a cyclic interval lesser than 50ms please use an XM22 PLC.

- For analog input one prefers S20-AI-4-I modules because a 0..20 mA signal has more electromagnetic tolerance than a 0..10 V signal.
- For digital input one prefers S20-DI-16/1 modules because you can map them very easily to the S20 global variables.

6.3.4 I/O Mapping

If you want to access the values of digital and analog input modules using the IoT Gateway S20 device types please map them to the S20 global variables with this schema:

- The number of socket place is irrelevant - use the relative number of the module per type. Count from left to right beginning with 1.
- Map the **first** AI-4-I module to Application.AI_4_**I** [**1**], the second to Application.AI_4_**I**[**2**] and so on.
- Map the **first** AI-4-U module to Application.AI_4_**U** [**1**], the second to Application.AI_4_**U**[**2**] and so on.
- Map the **first** DI-16/1 module to Application.DI_16_1 [**1**], the second to Application.DI_16_1[**2**]] and so on.

Hint: Enable those sample IO modules which you need - they are mapped as first module of the according type. For more information and how to use an AI8 module see POU PlcProg.

6.3.5 NTP - Time Synchronization with Network Time Protocol

6.3.5.1 What is Network Time Protocol?

Network Time Protocol (NTP) is a protocol designed to synchronize the clocks of computers over a network.

A client computer sends a NTP request to a NTP server. The answer of the server contains the exact date/time information. There are a lot of public NTP servers in the internet. Furthermore private networks can have also NTP servers - often DNS servers provide this function also.

6.3.5.2 NTP Client Bundle

You can use NTP to synchronize the clock of the XM2 control with a time server. Therefore the IoT Gateway bundle `IndraControl XM (com.boschrexroth.iiot.app.xm)` is provided. It contains an NTP client function, which is disabled by default.

6.3.5.3 Configure the NTP client

1. Open the IndraControl XM application. Under NTP Settings you can change following parameters:
   - **Enable**: Enables/disables the function
   - **Interval (s)**: Within this interval requests are sent to the server.
   - **NTP Server IP Address**: Enter up to four valid IP addresses of the NTP servers
     Notice: not the name/url!
2. Proof/change the parameters, check the ‘Enable’ option and press ‘Save’ to apply the changes.
3. Now the NTP client function synchronizes the time with the specified time server(s).
4. Errors will be shown in the IoT Gateways Web UI under Diagnostics/Alarms.

7. Processing – data receivers

7.1. Amazon Cloud (AWS Cloud) Processing

7.1.1 Description

Publishes data to the **Amazon Cloud (AWS Cloud)** using MQTT

7.2 Microsoft Azure Processing

7.2.1 Description

Sends the selected values of endpoints (data sources) to the cloud service **Microsoft Azure**.

7.2.2 Specification

- After data collection, the data will send in chosen publishing intervals.
- Formats: various standard-formats (Raw, JSON,…) are selectable. Through usage of a formatter, any desired formats are possible.
- Security: HTTPS

IMPORTANT

If you're behind a corporate proxy, enter your proxy setup on page **Settings/Proxy**

7.3 MongoDB Processing

7.3.1 Description

Sends the values of the selected endpoints to a **MongoDB** instance.

7.3.2 Specification

- After the data has been collected, it will be sent in the selected publishing interval.
- Protocol: TCP.
- Security: Secure Connection (SSL) available.
- Two modes to write data into the database are available:
  - Mode IOT Schema: Collects all value Endpoint changes and writes them to the generated database collection `endpoint_<name>`. Recommended if default database structure is used.
  - Mode DAS Schema: Collects all value Endpoint changes and writes them to the generated database collection `values` (default collection name).

7.3.3 Mode IOT Schema

For **each** Endpoint a collection `endpoint_<name>` is created automatically. Generated collection columns:

```
ObjectId | Value | Unit | Timestamp | DeviceId
```

- ObjectId: unique number which identifies the row.
- Value: collected Endpoint value.
- Timestamp: timestamp of the appearance of the Endpoint value change.
- DeviceId: unique identifier of the IoT Gateway hardware to differ same Endpoint names on different IoT Gateways.

7.3.4 Mode DAS Schema

All selected Endpoints will be stored in a single collection (default name “values”) Every entry has the following attributes:

```
ObjectId | DateTime | <endpoint_value>
```

• ObjectId:	unique number which identifies the row.
• DateTime:	timestamp of the appearance of the Endpoint value change.
• endpoint_value:	the value of the Endpoint.

7.3.5 Create a MongoDB Processing

- Select ‘Processing’ from the home page.
- Select ‘Create a new Processing’
- Press button ‘Add cloud service’
- Select ‘MongoDB’

A dialog will appear, enter following parameters:

- **Name:** The name of the MongoDB processing.
- **Description:** The description of the MongoDB processing.
- **Address:** The URL/IP address of the MongoDB Server.
- **Port:** The port of the MongoDB Server (default 27017).
- **Schema:**
    - IOT Schema: Database and collections for every Endpoint are created automatically.
    - DAS Schema: Database and the value collection are created automatically.
- **Database:** The database of the MongoDB Server.
- **Collection:** If the DAS schema has been selected, the default collection name (values) can be modified here.
- **User:** The user for authentication.
- **Password:** The password for authentication.
- **Checkbox: “Use user and password authentication”** If unchecked, the credentials will not be used.
- **Checkbox: “Use Secure Connection”** If checked, the connection will secured.
- **Buffer Size:** The maximum number of stored values for each Endpoint.
- **Inputs:** Select the Endpoints you want to post to the MongoDB processing.

Click ‘Save’ to store your configuration.

7.3.6 Configure an Existing MongoDB Processing

- Select ‘Processing’ from the home page.
- Select ‘Change your Processing’
- Expand the service you want to configure.
- Enter/change the parameter(s).

Click ‘Save’ to store your changes.

7.3.7 Delete an Existing MongoDB Processing

- Select ‘Processing’ from the home page.
- Select ‘Change your Processing’

Click 'Delete' and confirm the deletion of the MongoDB processing.

7.8 MQTT Processing

7.8.1 Description

Sends the selected values of Endpoints (data sources) to a **MQTT Broker**.

7.8.2 Specification

- After data collection, the data will send in chosen publishing intervals.
- Format: various standard-formats (Raw, JSON,…) are selectable. Through usage of a formatter, any desired formats are possible.
- Security: no security (standard) or SSL

7.8.3 Introduction in MQTT

We recommend to read the http://www.hivemq.com/mqtt-essentials/ to get a short overview of the features set of **MQTT** communication including explanation of some parameters used in this processing.

IMPORTANT

If you're behind a corporate proxy, enter your proxy setup on page **Settings/Proxy**.

7.9 MySQL Processing

7.9.1 Description

Sends the selected values of Endpoints (data sources) to the database service MySQL Database.

7.9.2 Specification

- After data collection, the data will send in chosen publishing intervals.
- Protocol: TCP
- Security: There are no security options active.
- Two modes to write data into the database are implemented:
  - Mode IoT Schema: Collects all value Endpoint changes and writes it to the generated database table `endpoint_<name>`. Recommended if default database structure is used.
  - Mode SQL Statement: The user can submit self-created INSERT statements.

If the IoT Schema is chosen, the given database and all required tables are generated automatically.

If the Mode SQL Statement is chosen, it is necessary to set a formatter ahead. The formatter must return the INSERT Statement, which is the endpoint input of MySQL.

7.9.3 Mode IoT Schema

For each Endpoint a table `endpoint_<name>` is created automatically. Generated Table Columns:

```
id | value | unit | timestamp | macaddress
```

- id: unique number which identifies the row
- value: collected endpoint value
- timestamp: timestamp of the appearance of the Endpoint value
- macaddress: MAC address of the IoT Gateway to differ same Endpoint names on different IoT Gateways

7.9.4 Mode SQL Statement

In this mode the user can send INSERT statements directly to the MySQL database. This is helpful for inserting data into an existing table. Use the formatter to create the INSERT statements.

IMPORTANT

Attention: Write only. Use only INSERT statements not to block the entire processing.

Hints:

- Semicolon at the end of a statement is optional, but recommended
- It is possible to write several SQL statements in a row
- Use `STR_TO_DATE('{{time}}','%Y-%m-%dT%H:%i:%s.%fZ')` for saving UTC time in timestamp
- `Use {{^lastValue}},{{/lastValue}}` for buffering with one endpoint
- Use `{{^lastValue}},{{/lastValue}}{{^lastInput}}{{#lastValue}},{{/lastValue}}{{/lastInput}}` for buffering with multiple endpoints

You can use the following example to create a table with structure: value | unit | timestamp. All columns with datatype VARCHAR:

```
CREATE TABLE `mytable` (

  `id` INT(11) NOT NULL AUTO_INCREMENT,

  `value` VARCHAR(200) NOT NULL,

  `unit` VARCHAR(10) NULL DEFAULT NULL,

  `timestamp` VARCHAR(100) NULL DEFAULT NULL,

PRIMARY KEY (`id`)

);
```

The following Formatter examples assumes the table structure: `value | unit | timestamp`.

For each endpoint value an `INSERT` statement. All columns with datatype `VARCHAR`:

```
{{#inputs}}

  {{#values}}

INSERT INTO mytable (value, unit, timestamp) VALUES ('{{value}}','myunit','{{time}}');

  {{/values}}

{{/inputs}}
```

One `INSERT` statement for several endpoint values. All columns with datatype `VARCHAR`:

```
INSERT INTO mytable (value, unit, timestamp) VALUES

{{#inputs}}

  {{#values}}

('{{value}}','myunit','{{time}}'){{^lastValue}},{{/lastValue}}{{^lastInput}}{{#lastValue}},{{/la
stValue}}{{/lastInput}}

  {{/values}}

{{/inputs}}
```

One `INSERT` statement for several endpoint values and column `timestamp` in datatype `TIMESTAMP(3)`:

```
INSERT INTO mytable (value, unit, timestamp) VALUES

{{#inputs}}

{{#values}}

('{{value}}','myunit',STR_TO_DATE('{{time}}','%Y-%m-
%dT%H:%i:%s.%fZ')){{^lastValue}},{{/lastValue}}{{^lastInput}}{{#lastValue}},{{/lastValue}}

{{/values}}

{{/inputs}}
```

7.9.10 How to Add a New User for the MySQL Server

```
CREATE USER 'newuser'@'%' IDENTIFIED BY 'password';
```

%: wildcard for all IPs

7.9.11 How to Give Rights to the New User

```
GRANT ALL PRIVILEGES ON * . * TO 'newuser'@'%';
```

- `ALL PRIVILEGES`: all Statements are allowed to set up (`SELECT, INSERT, UPDATE` etc.)
- Left asterisks: which database (* = all)
- Right asterisks: which tables (* = all)

7.9.12 Configuration of the Firewall

It is eventually necessary to configure the firewall.

Example for MS Windows operating system:

- Open the advanced settings of the Windows firewall and create a new inbound rule:
  - Rule Type: Port
  - Protocol and Ports: TCP, specific local port: 3306 (default)
  - Action: Allow the connection
  - Profile: Check Domain, Private or Public, depending on your network configuration
  - Name: Enter a name for the rule, e.g. IoT Gateway MySQL

7.10 Bosch Online Diagnostics Network (ODiN) Processing

7.10.1 Description

Sends the selected values of Endpoints (data sources) to the cloud service Bosch Online Diagnostics Network (ODIN).

7.10.2 Specification

- After data collection, the data will send in chosen publishing intervals.
- Format: ODIN format
- Security: no security

IMPORTANT

If you're behind a corporate proxy, enter your proxy setup on page **Settings/Proxy**

7.10.3 Create an ODiN Processing

- Select ‘Processing’ from the home page.
- Select ‘Create a new Processing’
- Press button ‘Add cloud service’
- Select ‘Bosch Online Diagnostics Network (ODiN)’

A dialog will appear, enter following parameters:

- **Name:** The Name of the Processing.
- **Description:** The Description of the Processing.
- **Server URL:** The URL address of the ODiN Server
- **Protocol:** The Bosch ODiN Protocol.
- **Import ID:** The inventory Import ID (username) of the Bosch ODiN application
- **Password:** The password of the Bosch ODiN account
- **Endpoint Mapping:** Indicates the Endpoint Mapping
- **Endpoint Description:** Map the Endpoint’s description to the corresponding channel
- **Endpoint Name:** Map the Endpoint’s name to the corresponding channel
- **Model Mapping Description:** Use the Device Model Mapping Description
- **Round Decimals:** Round decimal values to the closest long value (ties rounding to positive infinity).
- **Buffer Size:** The maximum number of stored values for each Endpoint.
- **Inputs:** Check the Endpoints you want to post to the Processing.
- Click ‘Create’ to store your changes.

7.10.4 Configure an existing ODiN Processing

- Select ‘Processing’ from the home page.
- Select ‘Change your Processing’
- Expand the service you want to configure.
- Enter/change the parameter(s).
- Click ‘Save’ to store your changes.

7.10.5 Delete an existing ODiN Processing

- Select ‘Processing’ from the home page.
- Select ‘Change your Processing’
- Click ‘Delete’ and confirm the deletion of the Processing.

7.11 Oracle IoT Cloud Service Processing

7.11.1 Description

Sends the selected values of endpoints (data sources) to the cloud service **Oracle IoT Cloud Service**.

7.11.2 Specification

- After data collection, the data will send in chosen publishing intervals.
- Format: Oracle IoT Cloud Service format
- Security: HTTPS

IMPORTANT

If you’re behind a corporate proxy, enter your proxy setup on page **Settings/Proxy**

7.12 Bosch Production Performance Manager - Machine Message

7.12.1 Description

Sends the selected values of Endpoints (data sources) to the cloud service **NEXEED Production Performance Manager** in Machine Message format.

7.12.1 Specification

- After data collection, depending on processing configuration, the data will send cyclical or event triggered.
- Protocol: PPM – Machine Message REST API
- Security:
  - PPM on premises: none
  - PPM in Cloud: OpenVPN
- Fringe conditions:
  - The chosen Endpoints needs to return values, which match the Machine Message format.
- Ensure while changing the key value “code”, that the optional values (Title, Severity, …) of the data source are updated too.

The message corresponds to the Machine Message Payload of the PPMP Specification.

IMPORTANT

If you’re using the server hostname instead of its IP address, you have to add the hostname and its corresponding IP address to the `hosts` file to allow proper name resolution for the IoT Gateway

- Login to your IoT Gateway using any SFTP Client (e.g. WinSCP)
- Edit file `/OEM/ProjectDataProtected/hosts`
- Save the file
- Reboot

7.13 Bosch Production Performance Manager - Measurement

7.13.1 Description

Sends the selected values of Endpoints (data sources) to the cloud service **NEXEED Production Performance Manager** in Measurement format.

7.13.2 Specification

- After data collection, depending on processing configuration, the data will send cyclical or event triggered. Protocol: PPM – Measurement REST API
- Security:
  - PPM on premises: none
  - PPM in Cloud: OpenVPN
- Fringe conditions:
  - Endpoints will recorded, if it is possible to cast their values in a number format (double, float, int, Boolean, …)

Notice: The Endpoint values (sampling interval < publishing interval) will recorded within transmitting cycle and published in the protocol as array at publishing interval.

The message corresponds to the Measurement Payload of the PPMP Specification.

IMPORTANT

If you're using the server hostname instead of its IP address, you have to add the hostname and its corresponding IP address to the `hosts` file to allow proper name resolution for the IoT Gateway

- Login to your IoT Gateway using any SFTP Client (e.g. WinSCP)
- Edit file `/OEM/ProjectDataProtected/hosts`
- Save the file
- Reboot

7.14 Bosch Production Performance Manager - Process Message

7.14.1 Description

Sends the selected values of endpoints (data sources) to the cloud service **NEXEED Production Performance Manager** in Process Message format.

7.14.2 Specification

- After data collection, depending on processing configuration, the data will send cyclical or event triggered.
- Protocol: PPM - Process Message REST API
- Security:
  - PPM on premises: none
  - PPM in Cloud: OpenVPN
- Fringe conditions:
  - The chosen Endpoints need to return values, which match the Process Message format.
  - Endpoints will recorded, if it is possible to cast their values in a number format (double, float, int, boolean, …)

Notice: It is required to set specific Endpoint values in the Endpoint configuration to trigger the start and end of a recording.

The message corresponds to the Process Message Payload of the PPMP Specification.

IMPORTANT

If you're using the server hostname instead of its IP address, you have to add the hostname and its corresponding IP address to the `hosts` file to allow proper name resolution for the IoT Gateway

- Login to your IoT Gateway using any SFTP Client (e.g. WinSCP)
- Edit file `/OEM/ProjectDataProtected/hosts`
- Save the file
- Reboot

7.15 REST Processing

7.15.1 Description

Publishes REST requests to a **WEB Server**

IMPORTANT

If you’re behind a corporate proxy, enter your proxy setup on page **Settings/Proxy**

7.15.2 Description of the REST Client Options

The present **REST client** provides subsequent HTTP request Methods:

POST:	Used to create a new or update of an existing resource
PUT:	Used to update or replace of an existing resource
PATCH:	Used to update or modify of an existing resource
DELETE:	Used to remove a resource

and Content-Types (the MIME type of the body):

- Application/json
- Application/xml
- Application/javascript
- text/plain
- text/xml
- text/html

7.16 TCP/IP Processing

7.16.1 Description

Publishes to a **TCP Server**

IMPORTANT

If you’re behind a corporate proxy, enter your proxy setup on page **Settings/Proxy**

8. Other

8.1 Enhancements of Functionality

The IoT Gateway is a modular framework based on Java and OSGI. The framework can be enlarged case specific. To ensure that, an SDK is provided. The SDK provides all necessary interfaces to create and run custom bundles. Various examples can be provided as well. A detailed documentation of the SDK may be available from the Bosch Rexroth local partner.

8.2 Ordering Information

R911xxxxxx	IoT Gateway Software options package for PPM

8.3 Additional source of information

https://www.boschrexroth.com/gateway	Information pages for IoT Gateway. The download area provides tools for the IoT Gateway.
https://www.boschrexroth.com/network	Bosch Rexroth Engineering network community with plenty of information and files to download, users forum in between for IoT Gateway.
Bosch Rexroth Media Directory	The media area is a central location for various documentation for Bosch Rexroth products in between IoT Gateway.

8.4 Industrial Security Recommendations

The following recommendations show a package of measures to achieve best possible barriers against unintentional or malicious modification or potential vulnerability attacks of low and medium motivation, against the **IoT Gateway** itself and provided `END-TO-END` connections in an industrial production environment.

8.4.1 General Information on Proper Use

The **IoT Gateway** is to be run as an intermediate server providing an entry point from the internet to enterprise or service provider core networks (**Edge Device**). It is strongly recommended to solely use the **IoT Gateway** within a secured, closed network or over a secure VPN internet connection.

IMPORTANT:

- We recommend to follow the instructions in the Bosch Rexroth IndraControl Security Manual
- We recommend to follow the instructions in the VDMA Industrial 4.0 Security Guideline

Bosch Rexroth does not assume any liability for risks resulting from malicious application code.

8.4.2 Use Secure Networking Hardware

Use secure networking hardware such as an industrial firewall (e.g. Phoenix Contact MGUARD VPN Router) protecting the **IoT Gateway** to prevent access from unauthorized individuals.

8.4.3 Web Console

Navigate to the Web Console by opening any supported Web Browser (e.g. Google Chrome, Mozilla Firefox) and navigate to `https://[IP_OF_YOUR_IOT_GATEWAY]:8888/system/console`

8.4.4 Self Signed Certificate

The certificate of the **IoT Gateway** is currently self-signed and therefore your web browser displays a security warning upon the initial connection and navigation to a website in the web root. Please verify and add the untrusted certificate to your exceptions.

8.4.5 Login Authentication

On first access of the built-in Web Console `http://{YOUR_IOT_GATEWAY_IP_HERE}` you have to provide a **strong** password.

Please login with the `Default` user:

```
user: admin
```

```
password: admin
```

IMPORANT:

Your provided credentials (username, password) must differ from credentials of the `Default` user

If you forgot your credentials, please contact any **IoT Gateway** service personal to reset.

8.4.6 Password Change on PR21 Hardware

It is strongly recommended to change the initial password on a PR21 hardware to ensure access control

- Connect via SSH, e.g. using Putty: https://putty.org/, or directly via the HDMI port of the device
- Log in (user: boschrexroth, password: boschrexroth), Note: No visual output available when entering the password
- Following the login you are now in the home directory of user “boschrexroth”

Be aware to remember your new password. After a change the initial password cannot longer be used

- Enter the following command
  `passwd`
- You are now prompted to enter the initial password (“boschrexroth”).
- *Note: There is no* shown while entering the password*
- After having entered the old password you are prompted for a new password.
- After entering and repeating the new password the setting is changed permanently.

8.4.7 Secure File Transfer Protocol (SFTP) Authentication

The **IoT Gateway** provides a full SFTP file the built-in filesystem access, which has to protected from unauthorized read and write access. Please change the SFTP credentials (username, password) to other than the `Default` user.

NOTE: This feature is currently not supported and will be available in future releases.

8.4.8 Secure OPC UA Server Communication

The **IoT Gateway** comes with an built-in OPC UA Server to be connected by any external OPC UA Client. The OPC UA Server does only expose secure server endpoints by default. Despite of that, it’s possible to enable unsecure endpoints for testing- or commissioning purposes.

Disable this feature on production mode and use **Only Secure OPC UA Communication**.

Please read the OPC UA Server Application documentation for further information.

8.4.9 Secure END-TO-END Connections (Message Security and Data Integrity)

The **IoT Gateway** is a proxy which facilitates establishing a connection from any cloud service to industrial field devices on the production floor. `END-TO-END` defines the connection route between the cloud service and the connected field device this **IoT Gateway** is connecting to. It can be divided into two main parts:

- **Route A**: The route between the cloud service and the **IoT Gateway** using any communication (e.g. HTTPS, AMQP, MQTT, others, …)
- **Route B**: The route between the **IoT Gateway** and the connected field devices using the native protocols (e.g. OPC UA, OPC DA, TCP/IP, others, …)

IMPORTANT

Always ensure the best possible data encryption of the entire `END-TO-END` connection

8.4.9.1 Route A: Cloud Service to IoT Gateway

Always use a secure protocol to any connected target like cloud services etc. (e.g. HTTPS, AMQP_S, MQTT SSL, others, …)

8.4.9.2 Route B: IoT Gateway to the Field Device

If the device is providing a secure connection or protocol, we recommend to enable in the individual device configuration.

8.4.10 OPC UA Device

Connections to OPC UA Servers using the OPC UA Device

8.4.10.1 Message Security

- Specify one of the available `SecurityModes` as connect parameter of the connect method other than `None`

If specified, **IoT Gateway** manages the created certificate for you.

8.4.10.2 User Authentication

- If the connected OPC UA Server supports user authentication, specify the credentials (username, password) in the device configuration

8.4.11 Logic Device / System Device (MLPI, OCI)

Connections to external MLC-/IoT-Gateway Devices

NOTE: This feature is currently not supported and will be available in future releases.

8.4.11.1 Message Security

- Enable `Secure Connection` (currently not supported, available in future releases)

8.4.11.2 User Authentication and Authorization

- Enable the built-in authentication of your connected `MLPI Server` by configuring the existing MLPI user management (`accounts.xml`)
- If you are using the MLPI user management, specify the credentials (username, password) in the device configuration

Please see the available MLPI documentation for details.

8.4.12 Security Issues of Third Party Components (Open Source Software)

Even though **Bosch Rexroth** strives to provide the most secure versions of used `Open Source Software` available, we recommend to check the shipped versions against known vulnerabilities when using the **IoT Gateway**

A list of used third party components can be found below and is also available from `Open Source Software` option in the **IoT Gateway** help.

8.4.13 Open Source Software (OSS)

The IoT Gateway uses the following open source software, which are covered by individual licenses, available in the following links.

8.4.13.1 Used Open Source Components

Apache Commons Codec

Version	1.11.0
Copyright	Copyright (c) 2002 Kevin Atkinson (kevina@gnu.org) Copyright (c) 2008 Alexander Beider & Stephen P. Morse. Copyright 2002-2016 The Apache Software Foundation
License	**The Apache Software License Version 2.0**
Notice	**commons-codec.txt**

Apache Commons IO

Version	2.6.0
Copyright	Copyright 2002-2016 The Apache Software Foundation
License	**The Apache Software License Version 2.0**
Notice	**commons-io.txt**

Apache Commons Lang

Version	3.7.0
Copyright	Copyright 2001-2016 The Apache Software Foundation
License	**The Apache Software License Version 2.0**
Notice	**commons-lang3.txt**

Apache Commons Math

Version	3.6.1

Copyright Copyright 2001-2016 The Apache Software Foundation
Copyright 2010-2012 CS Systèmes d'Information

License The Apache Software License Version 2.0

Notice commons-math3.txt

Apache Felix Http API

Version 3.0.0

Copyright Copyright 2002-2016 The Apache Software Foundation

License The Apache Software License Version 2.0

Apache Felix Http Jetty

Version 4.0.6

Copyright Copyright 2002-2016 The Apache Software Foundation

License The Apache Software License Version 2.0

Apache Felix Log Service

Version 1.0.1

Copyright Copyright 2002-2016 The Apache Software Foundation

License The Apache Software License Version 2.0

Notice org.apache.felix.log.txt

Apache Felix Main

Version 5.6.1

Copyright Copyright 2002-2016 The Apache Software Foundation

License The Apache Software License Version 2.0

Notice org.apache.felix.main.txt

Apache Felix Servlet API

Version 1.1.2

Copyright Copyright 2002-2016 The Apache Software Foundation

License The Apache Software License Version 2.0

Apache Felix Web Console Service Component Runtime/Declarative Services Plugin

Version 2.0.8

Copyright Copyright 2002-2016 The Apache Software Foundation

License **The Apache Software License Version 2.0**

Apache Felix Web Management Console (All In One)

Version 4.3.8.all

Copyright Copyright 2002-2016 The Apache Software Foundation

License **The Apache Software License Version 2.0**

Apache Log4j

Version 1.2.17

Copyright Copyright 2010 The Apache Software Foundation

License **The Apache Software License Version 2.0**

Notice **log4j.txt**

Apache Qpid Proton-J

Version 0.25.0

Copyright Copyright 2012-2017 The Apache Software Foundation

License **The Apache Software License Version 2.0**

Notice **proton-j.txt**

asm

Version 5.0.3

Copyright Copyright (c) 2000-2011 INRIA, France Telecom

License **Berkeley Software Distribution (BSD)**

asm-analysis

Version 5.0.3

Copyright Copyright (c) 2000-2011 INRIA, France Telecom

License **Berkeley Software Distribution (BSD)**

asm-commons

Version 5.0.3

Copyright Copyright (c) 2000-2011 INRIA, France Telecom

License **Berkeley Software Distribution (BSD)**

asm-tree

Version 5.0.3

Copyright Copyright (c) 2000-2011 INRIA, France Telecom

License **Berkeley Software Distribution (BSD)**

asm-util

Version 5.0.3

Copyright Copyright (c) 2000-2011 INRIA, France Telecom

License **Berkeley Software Distribution (BSD)**

AWS IoT Device SDK for Java

Version 1.2.1

Copyright Copyright 2010-2016 Amazon.com, Inc. or its affiliates. All Rights Reserved.

License **The Apache Software License Version 2.0**

Notice **aws-iot-device-sdk-java.txt**

Bouncy Castle PKIX, CMS, EAC, TSP, PKCS, OCSP, CMP, and CRMF APIs

Version 1.60.0

Copyright Copyright (c) 2000 - 2016 The Legion of the Bouncy Castle Inc. (https://www.bouncycastle.org)

License **Bouncy Castle Licence**

Bouncy Castle Provider

Version 1.60.0

Copyright Copyright (c) 2000 - 2016 The Legion of the Bouncy Castle Inc. (https://www.bouncycastle.org)

License **Bouncy Castle Licence**

Bouncy Castle S/MIME API

Version 1.53.0

Copyright Copyright (c) 2000 - 2017 The Legion of the Bouncy Castle Inc. (https://www.bouncycastle.org)

License Bouncy Castle Licence

classmate

Version 1.4.0

Copyright Copyright (c) 2018 Tatu Saloranta

License The Apache Software License Version 2.0

com.eclipsesource.jaxrs.publisher

Version 5.3.2

Copyright Copyright 2011,2015 Holger Staudacher

License Eclipse Public License, version 1.0

Commons FileUpload

Version 1.3.3.prosyst

Copyright 2002-2019 The Apache Software Foundation.

License The Apache Software License Version 2.0

Dependencies for Iot Hub Java SDK

Version 0.8.0

Copyright Copyright (c) Microsoft Corporation

License MIT License

derby

Version 10.11.1.1

Copyright Copyright 2004-2018 The Apache Software Foundation

License Apache License 2.0

Notice derby.txt

Drizzle-JDBC

Version 1.4.0

Copyright Copyright (c) 2009-2013, Marcus Eriksson

License Simplified BSD

Eclipse Paho Java Client

Version 1.2.0

Copyright Copyright (c) 1999, 2014 IBM Corp.

License Eclipse Public License 1.0

Notice org.eclipse.paho.client.mqttv3.txt

EPICSTools etherip

Version 1.3.0

Copyright Copyright (c) 2012 Oak Ridge National Laboratory

License Eclipse Public License 1.0

exp4j - Mathematical expression evaluator for the Java

Version 0.4.8

Copyright Copyright 2016 Frank Asseg

License The Apache Software License Version 2.0

Gson

Version 2.8.5

Copyright Copyright 2008 Google Inc.

License The Apache Software License Version 2.0

Notice gson.txt

Guava: Google Core Libraries for Java

Version 15.0.0

Copyright Copyright (c) 2012 The Guava Authors

License The Apache Software License Version 2.0

hibernate-validator

Version 6.0.13.Final

Copyright Copyright (c) 2019 Adam Stawicki, Ahmed Al Hafoudh, Alaa Nassef, Andrey Derevyanko, Andrey Rodionov, Asutosh Pandya, Benson Margulies, Brent Douglas, Carlos Vara, Carlo de Wolf, Chris Beckey, Christian Ivan, Dag Hovland, Davide D'Alto, Davide Marchignoli, Denis Tiago, Doug Lea, Emmanuel Bernard, Efthymis Sarbanis, Federico, Federico Mancini, Gavin King, George Gastaldi, Gerhard Petracek, Guillaume Husta, Guillaume Smet, Gunnar Morling, Hardy Ferentschik, Henno Vermeulen, Hillmer Chona, Jan-Willem Willebrands, Jason T. Greene, Jiri Bilek, Julien Furgerot, Julien May, Juraci Krohling, Justin Nauman, Kathryn Killebrew, Kazuki Shimizu, Kevin Pollet, Khalid Alqinyah, Lee KyoungIl, Leonardo Loch Zanivan, Lucas Pouzac, Lukas Niemeier, Mark Hobson, Marko Bekhta, Matthias Kurz, Mert Çalişkan, Michal Fotyga, Nicola Ferraro, Nicolas François, Paolo Perrotta, Pete Muir, Rob Dickinson, Sanne Grinovero, Sebastian Bayerl, Shahram Goodarzi, Shane Bryzak, Shelly McGowan, Sjaak Derksen, Steve Ebersole, Strong Liu, Tadhg Pearson, Takashi Aoe, Tomaz Cerar, Tommy Johansen, Victor Rezende dos Santos, Willi Schönborn, Xavier Sosnovsky, Yanming Zhou, Yoann Rodière

License The Apache Software License Version 2.0

IoT Hub Java Device Client

Version 1.15.0

Copyright Copyright (c) Microsoft Corporation

License MIT License

jackson-annotations

Version 2.9.7

Copyright Copyright(c) 2014 FasterXML, LLC

License The Apache Software License Version 2.0

Jackson-core

Version 2.9.7

Copyright Copyright(c) 2014 FasterXML, LLC

License The Apache Software License Version 2.0

jackson-databind

Version 2.9.7

Copyright Copyright(c) 2014 FasterXML, LLC

License The Apache Software License Version 2.0

Java Servlet API

Version 3.1.0

Copyright Copyright (c) 1989, 1991 Free Software Foundation, Inc. 59 Temple Place

License CDDL-1.0

Javassist

Version 3.18.2

Copyright Copyright 2006 Envoi Solutions LLC

License The Apache Software License Version 2.0

JAX-RS API

Version 2.0.1

Copyright Copyright (c) 2011-2017 Oracle and/or its affiliates. All rights reserved.

License CDDL-1.1

jboss-logging

Version 3.3.2.Final

Copyright Copyright ©2017 Red Hat, Inc.

License The Apache Software License Version 2.0

Jersey Min (Rebundled)

Version 2.25.1

Copyright Copyright 2011,2015 Holger Staudacher

License Eclipse Public License 1.0

jettison

Version 1.3.8

Copyright Copyright 2006 Envoi Solutions LLC

License The Apache Software License Version 2.0

jffi

Version 1.2.15

Copyright Copyright (c) Wayne Meissner

License The Apache Software License Version 2.0

jnr-constants

Version 0.9.9

Copyright Copyright (c) Wayne Meissner
Copyright (c) Charles Oliver Nutter

License The Apache Software License Version 2.0

jnr-enxio

Version 0.17.0

Copyright Copyright (c) Wayne Meissner

License The Apache Software License Version 2.0

jnr-ffi

Version 2.1.4

Copyright Copyright (c) Wayne Meissner
Copyright (c) Charles Oliver Nutter

License The Apache Software License Version 2.0

jnr-posix

Version 3.0.45

Copyright Copyright (c) Thomas E Enebo
Copyright (c) Wayne Meissner
Copyright (c) Charles Oliver Nutter

License Eclipse Public License 1.0

jnr-unixsocket

Version 0.19.0

Copyright Copyright (c) Thomas E Enebo
Copyright (c) Fritz Elfert

License The Apache Software License Version 2.0

jnr-x86asm

Version 1.0.2

Copyright Copyright (c) 2010 Wayne Meissner
Copyright (c) 2008-2009, Petr Kobalicek kobalicek.petr@gmail.com

License MIT License

JSON In Java

Version 20131018.0.0.prosyst5

Copyright Copyright (c) 2002 JSON.org

License The JSON License

JSR 374 (JSON Processing) Default Provider

Version 1.0.4

Copyright Copyright (c) 1989, 1991 Free Software Foundation, Inc.
Copyright (c) 2011-2017 Oracle and/or its affiliates. All rights reserved.

License CDDL-1.0

Knopflerfish Device Manager

Version 4.0.1

Copyright Copyright (c) 2000-2005 INRIA, France Telecom
Copyright (c) OSGi Alliance (2000, 2014). All Rights Reserved.
Copyright 2003-2013 The Knopflerfish Project

License Knopflerfish License

Notice device.txt

kotlin-stdlib

Version 1.2.71

Copyright Kotlin Foundation

License The Apache Software License Version 2.0

kotlin-stdlib-common

Version 1.2.71

Copyright Kotlin Foundation

License The Apache Software License Version 2.0

log4j-api

Version 2.11.1

Copyright Copyright 1999-2017 Apache Software Foundation

License The Apache Software License Version 2.0

Notice log4j-api.txt

log4j-core

Version 2.11.1

Copyright Copyright 1999-2017 Apache Software Foundation

License The Apache Software License Version 2.0

Notice log4j-core.txt

Logback Classic Module

Version 1.2.3

Copyright Copyright (c) 1999-2015, QOS.ch. All rights reserved.

License Eclipse Public License 1.0

Notice logback-classic.txt

Logback Core Module

Version 1.2.3

Copyright Copyright (c) 1999-2015, QOS.ch. All rights reserved.

License Eclipse Public License 1.0

Notice logback-core.txt

Microsoft Azure Storage Client SDK

Version 2.2.0

Copyright Copyright (c) 2015 Microsoft Corporation

License The Apache Software License Version 2.0

MLPI for Java

Version 1.20.0

Copyright Copyright (c) 2014 Bosch Rexroth. All rights reserved.

License MLPI License

Moka7

Version 1.0.2

Copyright Copyright (c) 2014 Davide Nardella

License GNU Lesser General Public License 2.1

mongo-java-driver

Version 3.8.1

Copyright Copyright 2008-present MongoDB, Inc.

License The Apache Software License Version 2.0

Mustache.java

Version 0.9.4

Copyright Copyright 2010 RightTime, Inc.

License The Apache Software License Version 2.0

Notice compiler.txt

net.tammon:sip

Version 0.4.2

Copyright Copyright (c) 2017. tammon (Tammo Schwindt)

License MIT License

OPC Foundation UA JAVA

Version 1.3.346

Copyright Copyright (c) 1996-2015, OPC Foundation. All rights reserved.

License GNU Lesser General Public License 2.1

openSCADA JCIFS

Version 1.2.25.v20140625

Copyright Copyright (c) 2007 Free Software Foundation, Inc. http://fsf.org/

License GNU Lesser General Public License 3.0

openSCADA j-Interop Core

Version 2.1.8.201708110742

Copyright Copyright (c) 2007 Free Software Foundation, Inc. http://fsf.org/

License **GNU Lesser General Public License 3.0**

openSCADA j-Interop Dependencies

Version 1.5.0.201708100918

Copyright Copyright (c) 2007 Free Software Foundation, Inc. http://fsf.org/

License **The Guile License**

openSCADA Utgard DCOM Mappings

Version 1.5.0.201708100918

Copyright Copyright (c) 2007 Free Software Foundation, Inc. http://fsf.org/

License **GNU Lesser General Public License 3.0**

openSCADA Utgard Library

Version 1.5.0.201708100918

Copyright Copyright (c) 2007 Free Software Foundation, Inc. http://fsf.org/

License **GNU Lesser General Public License 3.0**

Oracle IoT Cloud Service Device Client Library

Version 18.2.5

Copyright Copyright (c) 2016, Oracle and/or its affiliates. All rights reserved.

License **MIT License (MIT) and the Universal Permissive License (UPL)**

org.everit.osgi.bundles.org.scream3r.jssc

Version 2.8.0

Copyright Copyright (c) 2010-2014 Alexey Sokolov (scream3r)

License **GNU Lesser General Public License Version 3**

Source [snap] scp <USER>@<ip-address>:/snap/iiot/current/app/gpl-sources/java-simple-serial-connector-2.8.0.zip ./

Source [docker] /iotgateway/app/gpl-sources/java-simple-serial-connector-2.8.0.zip

org.jetbrains.annotations

Version 13.0.0

Copyright 2000–2019 JetBrains s.r.o.

License **The Apache Software License Version 2.0**

org.osgi.core

Version 6.0.0

Copyright Copyright (c) OSGi Alliance (2000, 2015). All Rights Reserved.

License **The Apache Software License Version 2.0**

OSGi Companion Code for org.osgi.service.device

Version 1.1.0

Copyright Copyright (c) OSGi Alliance (2000, 2015). All Rights Reserved.

License **The Apache Software License Version 2.0**

OSGi Companion Code for org.osgi.service.log

Version 1.3.0

Copyright Copyright (c) OSGi Alliance (2000, 2015). All Rights Reserved.

License **The Apache Software License Version 2.0**

OSGi Compendium Release 6

Version 5.0.0.201305092017

Copyright Copyright (c) OSGi Alliance (2000, 2015). All Rights Reserved.

License **The Apache Software License Version 2.0**

OSGi JAX-RS Gson Provider

Version 2.3.0.201506221200

Copyright Copyright 2011,2015 Holger Staudacher

License **Eclipse Public License 1.0**

Notice **provider-gson.txt**

PrivateKeyReader

Version Unspecified

Copyright Copyright 2016 Amazon.com, Inc. or its affiliates. All Rights Reserved.

License The Apache Software License Version 2.0

Notice privatekeyreader.txt

Provisioning Security Provider

Version 1.2.0

Copyright Copyright (c) Microsoft. All rights reserved.

License MIT License

Notice security-provider.txt

Sharp7

Version 1.0.11

Copyright Copyright (c) 2018 Federico Barresi

License MIT License

SLF4J API Module

Version 1.7.25

Copyright Copyright (c) 2004-2017 QOS.ch

License MIT License

SLF4J Simple Binding

Version 1.7.25

Copyright Copyright (c) 2004-2017 QOS.ch

License MIT License

validation-api

Version 2.0.1.Final

Copyright Copyright © Red Hat, Inc.

License The Apache Software License Version 2.0

XStream Core

Version 1.4.10

Copyright Copyright (c) 2003-2006, Joe Walnes
Copyright (c) 2006-2015 XStream Committers
All rights reserved.

License BSD style

Zip4j

Version 1.3.2

Copyright Copyright 2013 Srikanth Reddy Lingala

License The Apache Software License Version 2.0

8.4.13.2 Used Open Source Files

AdminLTE

Version 2.3.3

Copyright Copyright (c) 2014-2016 Abdullah Almsaeed

License MIT License

angular-base64

Version 2.0.5

Copyright Copyright (c) 2010 Nick Galbreath

License MIT License

AngularChartJS

Version 1.1.0

Copyright Copyright (c) Jerome Touffe-Blin

License BSD 2-clause “Simplified” License

angular-cookies

Version 1.6.9

Copyright Copyright (c) 2010-2015 Google, Inc. http://angularjs.org

License MIT License

AngularJS

Version 1.6.9

Copyright Copyright (c) 2010-2016 Google, Inc.

License MIT License

AngularJS directives specific to Bootstrap

Version 2.5.0

Copyright Copyright (c) 2012-2017 the AngularUI Team, https://github.com/organizations/angular-ui/teams/291112

License MIT License

angular-messages

Version 1.6.9

Copyright Copyright (c) 2010-2018 Google, Inc. http://angularjs.org

License MIT License

angular-resource

Version 1.6.9

Copyright Copyright (c) 2010-2015 Google, Inc. http://angularjs.org

License MIT License

angular-route

Version 1.6.9

Copyright Copyright (c) 2010-2015 Google, Inc. http://angularjs.org

License MIT License

angular-sanitize

Version 1.6.9

Copyright Copyright (c) 2010-2015 Google, Inc. http://angularjs.org

License MIT License

Bootstrap

Version 3.3.6

Copyright Copyright (c) 2011-2016 Twitter, Inc.
Copyright (c) 2011-2016 The Bootstrap Authors

License MIT License

chart.js

Version 2.7.2

Copyright Copyright (c) 2018 Chart.js Contributors

License MIT License

es6-shim

Version 0.34.2

Copyright Copyright (c) 2013-2016 Paul Miller (http://paulmillr.com) and contributors

License MIT License

Font Awesome

Version 4.4.0

Copyright Copyright (c) 2016 Dave Gandy

License MIT License

Notice font-awesome.txt

jQuery

Version 2.2.4

Copyright Copyright jQuery Foundation and other contributors

License MIT License

LdapEncoder

Version Unspecified

Copyright Copyright 2005-2010 Adam Skogman, Mattias Hellborg Arthursson

License The Apache Software License Version 2.0

moment

Version 2.21.0

Copyright Copyright (c) JS Foundation and other contributors

License MIT License

Select2

Version	4.0.5
Copyright	Copyright (c) 2012-2017 Kevin Brown, Igor Vaynberg, and Select2 contributors
License	**MIT License**

8.4.13.3 Used Open Source Snippets

TrustAllCerts

Version	Unspecified
Copyright	Copyright (c) 2010 Pascal Thivent (stackoverflow user)
License	**Creative Commons Attribution Share Alike 2.5**

8.4.13.4 Used Open Source Tools

dmidecode

Version	3.1
Copyright	Copyright (c) 2003-2005 Jean Delvare jdelvare@suse.de
License	**GNU General Public License version 2**
Notice	**dmidecode.txt**
Source [snap]	scp <USER>@<ip-address>:/snap/iiot/current/app/gpl-sources/dmidecode/dmidecode-3.1.tar.gz ./
Source [docker]	/iotgateway/app/gpl-sources/dmidecode-3.1.tar.gz

openJDK

Version	8
Copyright	OpenJDK is a trademark of Oracle America, Inc.
License	**GNU General Public License, version 2, with the Classpath Exception**
Notice	**openjdk.txt**
Source [snap]	scp <USER>@<ip-address>:/snap/iiot/current/app/gpl-sources/openjdk/openjdk-8-8u191-b12/src.zip ./
Source [docker]	/iotgateway/app/gpl-sources/openjdk/openjdk-8-8u191-b12/src.zip

OpenVPN

Version	2.4.6
Copyright	Copyright (c) 2002-2018 OpenVPN Inc. This program is free software
License	[GNU General Public License version 2]()
Source [snap]	scp <USER>@<ip-address>:/snap/iiot/current/app/gpl-sources/openvpn-2.4.6.zip ./
Source [docker]	/iotgateway/app/gpl-sources/openvpn-2.4.6.zip
Hint	**Only uses with security bundles**

strongSwan

Version	5.7.2
Copyright	Copyright (c) 1999-2013 Henry Spencer, D. Hugh Redelmeier, Michael Richardson, Ken Bantoft, Stephen J. Bevan, JuanJo Ciarlante, Thomas Egerer, Heiko Hund, Mathieu Lafon, Stephane Laroche, Kai Martius, Stephan Scholz, Tuomo Soini, Herbert Xu., Martin Berner, Marco Bertossa, David Buechi, Ueli Galizzi, Christoph Gysin, Andreas Hess, Patric Lichtsteiner, Michael Meier, Andreas Schleiss, Ariane Seiler, Mario Strasser, Lukas Suter, Roger Wegmann, Simon Zwahlen, ZHW Zuercher Hochschule Winterthur (Switzerland)., Philip Boetschi, Tobias Brunner, Christoph Buehler, Reto Buerki, Sansar Choinyambuu, Adrian Doerig, Andreas Eigenmann, Giuliano Grassi, Reto Guadagnini, Fabian Hartmann, Noah Heusser, Jan Hutter, Thomas Kallenberg, Patrick Loetscher, Daniel Roethlisberger, Adrian-Ken Rueegsegger, Ralf Sager, Joel Stillhart, Daniel Wydler, Andreas Steffen, HSR Hochschule fuer Technik Rapperswil (Switzerland). Martin Willi (revosec AG), Clavister (Sweden).
License	[GNU General Public License version 2]()
Source [snap]	scp <USER>@<ip-address>:/snap/iiot/current/app/gpl-sources/strongswan-5.7.2.zip ./
Source [docker]	/iotgateway/app/gpl-sources/strongswan-5.7.2.zip
Hint	**Only uses with security bundles**

8.4.13.5 Written Offer for Source Code

This product contains software components that are licensed by the holder of the rights under GNU General Public License (GPL), GNU Lesser General Public License (LGPL) or any other Open Source Software license, which requires that source code be made available.

The source code of these software components is not already delivered together with this product. You can obtain the source code for these software components on a physical medium (CD or DVD) by submitting a written request to our open source office address listed below or by sending an email to mailto:open.source@boschrexroth.dealito:. When sending such a request, please name the relevant product and the date of purchase of the same.

Bosch Rexroth AG
Open Source Office
Zum Eisengießer 1
97816 Lohr am Main
Deutschland

We may charge you a fee (up to max. 20,- €) to cover the cost of physical media and processing. You may send your request (i) within three (3) years from the date you received the product that included the binary that is the subject of your request or (ii) in the case of code licensed under the GPL v3 for as long as Bosch Rexroth offers spare parts or customer support for that product.

8.4.14.1 Check built-in WebServer (Jetty) for Known Security Issues

The **IoT Gateway** contains the built-in WebServer component Jetty. Check the shipped version against known vulnerabilities when using the **IoT Gateway:**

http://www.eclipse.org/jetty/documentation/9.4.x/security-reports.html

9. FAQ

Q: Which version of the Java VM is required to run the IoT Gateway?

A: At least 64 bits Java environment in version 1.8 or above is required.

Q: Which bundles are currently available?

A: Your application support can provide you a list of all available bundles.

Q: How can I change the character set on PR21 if the character set of the keyboard is wrong?

A: The default character set of the keyboard settings on PR21 is English (US). To change the key board settings e.g. to German, execute the following command:

```
sudo loadkeys de
```

Q: How can I set the date and time on PR21?

A: Check the current system time with:

```
sudo timedatectl status
```

To change the system time execute:

```
sudo timdatectl set-time <HH:MM:SS>
```

or

```
sudo timdatectl set-time <YYYY-MM-TT>
```

or

```
sudo timedatect set-time <’YYY-MM-TT HH:MM:SS’>
```

Q: How can I setup a NTP client on PR21?

A:

1. Enter the NTP server address in the configuration file `/etc/systemd/timesyncd.conf`:

```
[Time]

NTP=<NTP server address>
```

2. Optional: Disable the NTP synchronization via:

```
sudo timedatectl set-ntp 0
```

3. Check the current system time with:

```
sudo timedatectl
```