# Table of contents

1 Foreword .................................................................................................................................................. 5  
1.1 Notes on the documentation............................................................................................................... 5  
1.2 Safety instructions ............................................................................................................................. 6  
2 Introduction ............................................................................................................................................. 7  
3 MP-Bus ...................................................................................................................................................... 8  
3.1 Topology ............................................................................................................................................... 8  
3.2 Actuator solutions ............................................................................................................................... 9  
4 Programming ........................................................................................................................................... 13  
4.1 POUs .................................................................................................................................................... 15  
4.1.1 Function blocks ............................................................................................................................. 16  
4.1.2 Functions ....................................................................................................................................... 48  
4.1.3 Error codes .................................................................................................................................... 50  
4.2 DUTs .................................................................................................................................................... 51  
4.2.1 Enums .......................................................................................................................................... 51  
4.2.2 Structures ..................................................................................................................................... 58  
4.3 Integration into TwinCAT ..................................................................................................................... 61  
4.3.1 KL6771 with CX5120 ..................................................................................................................... 61  
5 Appendix .................................................................................................................................................. 65  
5.1 Support and Service ............................................................................................................................ 65
1     Foreword

1.1     Notes on the documentation

This description is only intended for the use of trained specialists in control and automation engineering who are familiar with applicable national standards. It is essential that the documentation and the following notes and explanations are followed when installing and commissioning the components. It is the duty of the technical personnel to use the documentation published at the respective time of each installation and commissioning. The responsible staff must ensure that the application or use of the products described satisfy all the requirements for safety, including all the relevant laws, regulations, guidelines and standards.

Disclaimer

The documentation has been prepared with care. The products described are, however, constantly under development. We reserve the right to revise and change the documentation at any time and without prior announcement. No claims for the modification of products that have already been supplied may be made on the basis of the data, diagrams and descriptions in this documentation.

Trademarks

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® and XPlanar® are registered trademarks of and licensed by Beckhoff Automation GmbH. Other designations used in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owners.

Patent Pending

The EtherCAT Technology is covered, including but not limited to the following patent applications and patents:

EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany

Copyright

© Beckhoff Automation GmbH & Co. KG, Germany. The reproduction, distribution and utilization of this document as well as the communication of its contents to others without express authorization are prohibited. Offenders will be held liable for the payment of damages. All rights reserved in the event of the grant of a patent, utility model or design.
1.2 Safety instructions

Safety regulations

Please note the following safety instructions and explanations!
Product-specific safety instructions can be found on following pages or in the areas mounting, wiring, commissioning etc.

Exclusion of liability

All the components are supplied in particular hardware and software configurations appropriate for the application. Modifications to hardware or software configurations other than those described in the documentation are not permitted, and nullify the liability of Beckhoff Automation GmbH & Co. KG.

Personnel qualification

This description is only intended for trained specialists in control, automation and drive engineering who are familiar with the applicable national standards.

Description of symbols

In this documentation the following symbols are used with an accompanying safety instruction or note. The safety instructions must be read carefully and followed without fail!

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="DANGER" /></td>
<td>Serious risk of injury! Failure to follow the safety instructions associated with this symbol directly endangers the life and health of persons.</td>
</tr>
<tr>
<td><img src="image" alt="WARNING" /></td>
<td>Risk of injury! Failure to follow the safety instructions associated with this symbol endangers the life and health of persons.</td>
</tr>
<tr>
<td><img src="image" alt="CAUTION" /></td>
<td>Personal injuries! Failure to follow the safety instructions associated with this symbol can lead to injuries to persons.</td>
</tr>
<tr>
<td><img src="image" alt="NOTE" /></td>
<td>Damage to the environment or devices Failure to follow the instructions associated with this symbol can lead to damage to the environment or equipment.</td>
</tr>
</tbody>
</table>

Tip or pointer

This symbol indicates information that contributes to better understanding.
2 Introduction

The Tc2_MPBus library is a TwinCAT PLC library for data exchange with MP-Bus devices. This library should only be used in conjunction with a KL6771 (MP-Bus master terminal).

The user of this library requires basic knowledge of the following:

- TwinCAT XAE
- PC and network knowledge
- Structure and properties of the Beckhoff Embedded PC and its Bus Terminal system
- Technology of MP-Bus devices
- Relevant safety regulations for building technical equipment

This software library is intended for building automation system partners of Beckhoff Automation GmbH & Co. KG. The system partners operate in the field of building automation and are concerned with the installation, commissioning, expansion, maintenance and service of measurement, control and regulating systems for the technical equipment of buildings.

The Tc2_MPBus library is usable on all hardware platforms that support TwinCAT 3.1 or higher.

Hardware documentation in the Beckhoff information system:

http://infosys.beckhoff.com/content/1033/kl6771/html/bt_kl6771_title.htm
The MP-Bus (Multi-Point) is a simple sensor/actuator bus, which is used for certain sub-systems of building automation systems. The MP-Bus serves to control HVAC actuators for flaps, control valves and volumetric flow rate controllers from the Belimo Bus Solutions product range. Up to eight different devices from HVAC systems can be connected to an MP-Bus master using 3-wire technology. Additionally, a sensor can be connected to each of these eight devices; the sensor is addressed by the MP-Bus. An additional range of products with an MP-Bus connection is the FLS window ventilation system by the Belimo company (see Belimo documentation for the connection of the Belimo drives).

The MP-Bus was developed by Belimo Automation AG for connecting valves, throttle valves, air valves, fire dampers, and for window ventilation systems.

### 3.1 Topology

There are no restrictions whatsoever with regard to the topological structure of MP-Bus strands: star, ring, tree or mixed topologies are possible. The length of the entire bus strand depends on the selected cable cross-section and the type as well as the number of connected drives! The documentation from the Belimo company contains further information.
3.2 Actuator solutions
### Types

<table>
<thead>
<tr>
<th>Types</th>
<th>Unit</th>
<th>ready for extended MP-Bus®</th>
<th>Beckhoff function block</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ventilation applications</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rotary drives</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CM24-MPL-L</td>
<td>2 Nm</td>
<td>•</td>
<td>MPL_DamperLinearActuator</td>
</tr>
<tr>
<td>CM24-MPL-R</td>
<td>2 Nm</td>
<td>•</td>
<td>MPL_DamperLinearActuator</td>
</tr>
<tr>
<td>LM24A-MP</td>
<td>5 Nm</td>
<td></td>
<td>MP_DamperLinearActuator</td>
</tr>
<tr>
<td>NM24A-MP</td>
<td>10 Nm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SM24A-MP</td>
<td>20 Nm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GM24A-MP</td>
<td>40 Nm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rotary drives with emergency control function</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TF24-MFT</td>
<td>2 Nm</td>
<td></td>
<td>MP_DamperLinearActuator</td>
</tr>
<tr>
<td>LF24-MFT2</td>
<td>4 Nm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NF24A-MP</td>
<td>10 Nm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SF24A-MP</td>
<td>20 Nm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EF24A-MP</td>
<td>30 Nm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GK24A-MP</td>
<td>40 Nm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Linear drives</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LH24A-MP... 60/100/200/300</td>
<td>150 Nm</td>
<td></td>
<td>MP_DamperLinearActuator</td>
</tr>
<tr>
<td>SH24A-MP... 100/200/300</td>
<td>450 Nm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Linear drives with emergency control function</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LHK24A-MP100</td>
<td>150 Nm</td>
<td></td>
<td>MP_DamperLinearActuator</td>
</tr>
<tr>
<td>SHK24A-MP100</td>
<td>450 Nm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rotative drives</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LU24A-MP</td>
<td>3 Nm</td>
<td></td>
<td>MP_DamperLinearActuator</td>
</tr>
<tr>
<td>High-speed rotary drives</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NMC24A-MP</td>
<td>10 Nm</td>
<td></td>
<td>MP_DamperLinearActuator</td>
</tr>
<tr>
<td>SMC24A-MP</td>
<td>20 Nm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rotative drives for special applications</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NM24P-MP</td>
<td>10 Nm</td>
<td></td>
<td>MP_DamperLinearActuator</td>
</tr>
<tr>
<td>SM24P-MP</td>
<td>20 Nm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GM24G-MP-T</td>
<td>40 Nm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rotative drives with emergency control function for special applications</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NF24G-MP-L</td>
<td>10 Nm</td>
<td></td>
<td>MP_DamperLinearActuator</td>
</tr>
<tr>
<td>SF24G-MP-L</td>
<td>20 Nm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GK24G-MP</td>
<td>40 Nm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Water applications</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Drives for control ball valves / open-close control ball valves</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CQ24A-MPL</td>
<td>1 Nm</td>
<td>•</td>
<td>MPL_DamperLinearActuator</td>
</tr>
<tr>
<td>LR24A-MP</td>
<td>5 Nm</td>
<td></td>
<td>MP_DamperLinearActuator</td>
</tr>
<tr>
<td>NR24A-MP</td>
<td>10 Nm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SR24A-MP</td>
<td>20 Nm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Drives with emergency control function for control ball valves / open-close control ball valves</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRF24-MFT</td>
<td>2 Nm</td>
<td></td>
<td>MP_DamperLinearActuator</td>
</tr>
<tr>
<td>LRF24-MP</td>
<td>4 Nm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NRF24A-MP</td>
<td>10 Nm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Drives for globe valves</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Model</td>
<td>Torque (Nm)</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------------</td>
<td>--------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>LV24A-MP-TPC</td>
<td>500 N</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>NV24A-MP-TPC</td>
<td>1000 N</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>SV24A-MP-TPC</td>
<td>1500 N</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>EV24A-MP-TPC</td>
<td>2500 N</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>NVK24A-MP-TPC</td>
<td>1000 N</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>AVK24A-MP-TPC</td>
<td>2000 N</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>LVC24A-MP-TPC</td>
<td>500 N</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>NVC24A-MP-TPC</td>
<td>1000 N</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>SVC24A-MP-TPC</td>
<td>1500 N</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>NVKC24A-MP-TPC</td>
<td>1000 N</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>SR24A-MP-5</td>
<td>20 Nm</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>GR24A-MP-5/7</td>
<td>40 Nm</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>DR24A-MP-5/7</td>
<td>90 Nm</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>PRCA-BAC-S2-T</td>
<td>160 Nm</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>PRKCA-BAC-S2-T</td>
<td>160 Nm</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>SY2-24-MP-T</td>
<td>90 Nm</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>SY2-230-MP-T</td>
<td>90 Nm</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>SY3-24-MP-T</td>
<td>150 Nm</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>SY3-230-MP-T</td>
<td>150 Nm</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>SY4-24-MP-T</td>
<td>400 Nm</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>SY4-230-MP-T</td>
<td>400 Nm</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>SY5-24-MP-T</td>
<td>500 N</td>
<td>MP_DamperLinearActuator</td>
<td></td>
</tr>
<tr>
<td>BKN230-24MP for connection of BF(G)24TL-T-ST</td>
<td>11 / 18 Nm</td>
<td>MP_Smoker</td>
<td></td>
</tr>
<tr>
<td>BKN230-24-C-MP for connection of BF..24-..ST</td>
<td>4 / 9 / 11 / 18 Nm</td>
<td>MP_Smoker</td>
<td></td>
</tr>
<tr>
<td>LMV-D3-MP</td>
<td>5 Nm</td>
<td>MP_VAV</td>
<td></td>
</tr>
<tr>
<td>NMV-D3-MP</td>
<td>10 Nm</td>
<td>MP_VAV</td>
<td></td>
</tr>
<tr>
<td>SMV-D3-MP</td>
<td>20 Nm</td>
<td>MP_VAV</td>
<td></td>
</tr>
<tr>
<td>LHV-D3-MP</td>
<td>150 N</td>
<td>MP_VAV</td>
<td></td>
</tr>
<tr>
<td>CMV-100-MP</td>
<td>DN 100</td>
<td>MP_CMV</td>
<td></td>
</tr>
<tr>
<td>CMV-125-MP</td>
<td>DN 125</td>
<td>MP_CMV</td>
<td></td>
</tr>
<tr>
<td>CMV-150-MP</td>
<td>DN 150</td>
<td>MP_CMV</td>
<td></td>
</tr>
<tr>
<td>CMV-160-MP</td>
<td>DN 160</td>
<td>MP_CMV</td>
<td></td>
</tr>
</tbody>
</table>
### HVAC power units

<table>
<thead>
<tr>
<th>EPIV</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>EP..R-R6+BAC</td>
<td>DN 15-20</td>
<td>• MP_EPIV_R6_Para-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>meter</td>
</tr>
<tr>
<td>EP..R+MP</td>
<td>DN 15-20</td>
<td>MP_EPIV</td>
</tr>
<tr>
<td>P..W..E-MP</td>
<td>DN 65-150</td>
<td></td>
</tr>
</tbody>
</table>

### Belimo Energy Valve™

| EP..R+MP | DN 15-50 | • MP_EV             |
| P..W..EV-BAC | DN 65-150 | • MP_EV_Parameter   |

### Sensors

<table>
<thead>
<tr>
<th>Room sensors in the comfort zone</th>
</tr>
</thead>
<tbody>
<tr>
<td>22RT-19-1 (temp.)</td>
</tr>
<tr>
<td>22RTH-19-1 (temp., rh)</td>
</tr>
<tr>
<td>22RTM-19-1 (temp., CO₂, rh)</td>
</tr>
<tr>
<td>MS24A-R02-MPX (Temp., CO₂)</td>
</tr>
<tr>
<td>MS24A-R08-MPX (Temp., VOC, CO₂, rH)</td>
</tr>
</tbody>
</table>

1 The currently available product range can be found online at [www.belimo.eu](http://www.belimo.eu).

2 The marked devices support the extended address range. Up to 16 MP devices (addressed MP1 ... MP16) can be connected to a data line. If unmarked devices are connected to the same data line, the common address range should be limited to 8 MP devices.
## 4 Programming

### Function blocks

<table>
<thead>
<tr>
<th>Function blocks</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MP Addressing</td>
<td>This function block allows an MP-Bus slave to be addressed.</td>
</tr>
<tr>
<td>MP CMV [p.19]</td>
<td>This function block is used to control and monitor a volume flow regulator.</td>
</tr>
<tr>
<td>MP_DamperLinearActuator [p.21]</td>
<td>This function block is used to control and monitor a drive of a damper and of a globe valve.</td>
</tr>
<tr>
<td>MP EPIV [p.22]</td>
<td>This function block is used to control a pressure-independent control ball valve.</td>
</tr>
<tr>
<td>MP EPIV R6 [p.24]</td>
<td>This function block is used to control a control ball valve of series EP...R-R6+BAC.</td>
</tr>
<tr>
<td>MP EPIV R6_Parameter [p.26]</td>
<td>This function block is used to parameterize drives of series EP...R-R6+BAC.</td>
</tr>
<tr>
<td>MP EV [p.27]</td>
<td>This function block is used to control a control ball valve of series P6...W..EV-BAC.</td>
</tr>
<tr>
<td>MP EV Parameter [p.29]</td>
<td>This function block is used to parameterize drives.</td>
</tr>
<tr>
<td>MP PTH [p.31]</td>
<td>This function block is used to control and monitor a PTH sensor.</td>
</tr>
<tr>
<td>MP RoomSensor [p.33]</td>
<td>This function block is used to read room sensors.</td>
</tr>
<tr>
<td>MP RoomSensor Parameter [p.34]</td>
<td>This function block is used for the parameterization of room sensors.</td>
</tr>
<tr>
<td>MP Smoker [p.36]</td>
<td>This function block is used to control and monitor a fire damper.</td>
</tr>
<tr>
<td>MP UST 3 [p.37]</td>
<td>This function block is used to control and monitor a multi-IO module UST3.</td>
</tr>
<tr>
<td>MP VAV [p.40]</td>
<td>This function block is used to control and monitor a volume flow regulator.</td>
</tr>
<tr>
<td>MP VRU_Configuration [p.42]</td>
<td>This function block is used to configure the VAV drives VRU-D3-BAC, VRU-M1-BAC and VRU-M1R-BAC.</td>
</tr>
<tr>
<td>MP VRU_Process [p.44]</td>
<td>This function block is suitable for VAV drives VRU-D3-BAC, VRU-M1-BAC and VRU-M1R-BAC.</td>
</tr>
<tr>
<td>MP Window [p.46]</td>
<td>This function block is used to control and monitor a window ventilation system (FLS).</td>
</tr>
<tr>
<td>MPL_DamperLinearActuator [p.47]</td>
<td>This function block is used to control and monitor a drive of a damper and of a globe valve.</td>
</tr>
</tbody>
</table>

### Functions

<table>
<thead>
<tr>
<th>Function blocks</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NI1000_LuS_TO_INT [p.48]</td>
<td>This function calculates a temperature from the value of an NI1000 L&amp;S resistor.</td>
</tr>
<tr>
<td>NI1000_TO_INT [p.48]</td>
<td>This function calculates a temperature from the value of an NI1000 resistor.</td>
</tr>
<tr>
<td>NTC_TO_INT [p.49]</td>
<td>This function calculates a temperature from the value of an NTC resistor.</td>
</tr>
<tr>
<td>PT1000_TO_INT [p.49]</td>
<td>This function calculates a temperature from the value of a PT1000 resistor.</td>
</tr>
</tbody>
</table>
### Enums

<table>
<thead>
<tr>
<th>Data types</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Data_Window</code></td>
<td>Ventilation type.</td>
</tr>
<tr>
<td><code>E_MP_EPR6_UnitSel</code></td>
<td>Scaling.</td>
</tr>
<tr>
<td><code>E_MP_VRU_Application</code></td>
<td>Visualization of the application selected by the manufacturer.</td>
</tr>
<tr>
<td><code>E_MP_VRU_Command</code></td>
<td>Commands for service and test functions of the drive.</td>
</tr>
<tr>
<td><code>E_MP_VRU.OverrideControl</code></td>
<td>Setpoint override.</td>
</tr>
<tr>
<td><code>E_MP_VRU.RoomPressureCascade</code></td>
<td>Room pressure cascade control.</td>
</tr>
<tr>
<td><code>E_MP_VRU_Sensor1Type</code></td>
<td>External sensor at input S1.</td>
</tr>
<tr>
<td><code>E_MPBus_ControlMode</code></td>
<td>Control mode.</td>
</tr>
<tr>
<td><code>E_MPBus_DeltaTLimitation</code></td>
<td>Delta T (dT) limitation.</td>
</tr>
<tr>
<td><code>E_MPBus.Override</code></td>
<td>Override mode.</td>
</tr>
<tr>
<td><code>E_MPBus.Override_6wayMPIV</code></td>
<td>Override control mode.</td>
</tr>
<tr>
<td><code>MP_ERROR</code></td>
<td>Error messages.</td>
</tr>
<tr>
<td><code>UST3_EX</code></td>
<td>Voltage scaling.</td>
</tr>
<tr>
<td><code>UST3_R_SET</code></td>
<td>Resistance scaling.</td>
</tr>
</tbody>
</table>

### Structures

<table>
<thead>
<tr>
<th>Data types</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>DataKL6771</code></td>
<td>Links the send and receive blocks with the function block KL6771.</td>
</tr>
<tr>
<td><code>MP_BUS_MPX_ERROR</code></td>
<td>Error messages of the &quot;MPX&quot; sensors.</td>
</tr>
<tr>
<td><code>MP_Serial_Number</code></td>
<td>Serial number of the device.</td>
</tr>
<tr>
<td><code>St_MP_VRU.ServiceInfo</code></td>
<td>Fault and service information.</td>
</tr>
<tr>
<td><code>St_StateEV</code></td>
<td>Information on the state of the EV.</td>
</tr>
<tr>
<td><code>UST3_SET</code></td>
<td>Data structure for setting the scaling and the resistance measurement.</td>
</tr>
</tbody>
</table>
## 4.1 POUs

### Function blocks

<table>
<thead>
<tr>
<th>Function blocks</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MP Addressing [17]</td>
<td>This function block allows an MP-Bus slave to be addressed.</td>
</tr>
<tr>
<td>MP CMV [19]</td>
<td>This function block is used to control and monitor a volume flow regulator.</td>
</tr>
<tr>
<td>MP DamperLinearActuator [21]</td>
<td>This function block is used to control and monitor a drive of a damper and of a globe valve.</td>
</tr>
<tr>
<td>MP EPIV [22]</td>
<td>This function block is used to control a pressure-independent control ball valve.</td>
</tr>
<tr>
<td>MP EPIV R6 [24]</td>
<td>This function block is used to control a control ball valve of series EP...R-R6+BAC.</td>
</tr>
<tr>
<td>MP EPIV R6 Parameter [26]</td>
<td>This function block is used to parameterize drives of series EP...R-R6+BAC.</td>
</tr>
<tr>
<td>MP EV [27]</td>
<td>This function block is used to control a control ball valve of series P6...W.EV-BAC.</td>
</tr>
<tr>
<td>MP EV Parameter [29]</td>
<td>This function block is used to parameterize drives.</td>
</tr>
<tr>
<td>MP PTH [31]</td>
<td>This function block is used to control and monitor a PTH sensor.</td>
</tr>
<tr>
<td>MP RoomSensor [33]</td>
<td>This function block is used to read room sensors.</td>
</tr>
<tr>
<td>MP RoomSensor Parameter [34]</td>
<td>This function block is used for the parameterization of room sensors.</td>
</tr>
<tr>
<td>MP Smoker [36]</td>
<td>This function block is used to control and monitor a fire damper.</td>
</tr>
<tr>
<td>MP UST 3 [37]</td>
<td>This function block is used to control and monitor a multi-IO module UST3.</td>
</tr>
<tr>
<td>MP VAV [40]</td>
<td>This function block is used to control and monitor a volume flow regulator.</td>
</tr>
<tr>
<td>MP_VRU_Configuration [42]</td>
<td>This function block is used to configure the VAV drives VRU-D3-BAC, VRU-M1-BAC and VRU-M1R-BAC.</td>
</tr>
<tr>
<td>MP_VRU_Process [44]</td>
<td>This function block is suitable for VAV drives VRU-D3-BAC, VRU-M1-BAC and VRU-M1R-BAC.</td>
</tr>
<tr>
<td>MP Window [46]</td>
<td>This function block is used to control and monitor a window ventilation system (FLS).</td>
</tr>
<tr>
<td>MPL_DamperLinearActuator [47]</td>
<td>This function block is used to control and monitor a drive of a damper and of a globe valve.</td>
</tr>
</tbody>
</table>

### Functions

<table>
<thead>
<tr>
<th>Function blocks</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NI1000_LuS_TO_INT [48]</td>
<td>This function calculates a temperature from the value of an NI1000 L&amp;S resistor.</td>
</tr>
<tr>
<td>NI1000_TO_INT [48]</td>
<td>This function calculates a temperature from the value of an NI1000 resistor.</td>
</tr>
<tr>
<td>NTC_TO_INT [49]</td>
<td>This function calculates a temperature from the value of an NTC resistor.</td>
</tr>
<tr>
<td>PT1000_TO_INT [49]</td>
<td>This function calculates a temperature from the value of a PT1000 resistor.</td>
</tr>
</tbody>
</table>
4.1.1 Function blocks

<table>
<thead>
<tr>
<th>Function blocks</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MP_Addressing [17]</td>
<td>This function block allows an MP-Bus slave to be addressed.</td>
</tr>
<tr>
<td>MP_CMV [19]</td>
<td>This function block is used to control and monitor a volume flow regulator.</td>
</tr>
<tr>
<td>MP_DamperLinearActuator [21]</td>
<td>This function block is used to control and monitor a drive of a damper and of a globe valve.</td>
</tr>
<tr>
<td>MP_EPIV [22]</td>
<td>This function block is used to control a pressure-independent control ball valve.</td>
</tr>
<tr>
<td>MP_EPIV_R6 [24]</td>
<td>This function block is used to control a control ball valve of series EP...R-R6+BAC.</td>
</tr>
<tr>
<td>MP_EPIV_R6_Parameter [26]</td>
<td>This function block is used to parameterize drives of series EP...R-R6+BAC.</td>
</tr>
<tr>
<td>MP_EV [27]</td>
<td>This function block is used to control a control ball valve of series P6...W..EV-BAC.</td>
</tr>
<tr>
<td>MP_EV_Parameter [29]</td>
<td>This function block is used to parameterize drives.</td>
</tr>
<tr>
<td>MP_PTH [31]</td>
<td>This function block is used to control and monitor a PTH sensor.</td>
</tr>
<tr>
<td>MP_RoomSensor [33]</td>
<td>This function block is used to read room sensors.</td>
</tr>
<tr>
<td>MP_RoomSensor_Parameter [34]</td>
<td>This function block is used for the parameterization of room sensors.</td>
</tr>
<tr>
<td>MP_Smoker [36]</td>
<td>This function block is used to control and monitor a fire damper.</td>
</tr>
<tr>
<td>MP_UST_3 [37]</td>
<td>This function block is used to control and monitor a multi-IO module UST3.</td>
</tr>
<tr>
<td>MP_VAV [40]</td>
<td>This function block is used to control and monitor a volume flow regulator.</td>
</tr>
<tr>
<td>MP_VRU_Configuration [42]</td>
<td>This function block is used to configure the VAV drives VRU-D3-BAC, VRU-M1-BAC and VRU-M1R-BAC.</td>
</tr>
<tr>
<td>MP_VRU_Process [44]</td>
<td>This function block is suitable for VAV drives VRU-D3-BAC, VRU-M1-BAC and VRU-M1R-BAC.</td>
</tr>
<tr>
<td>MP_Window [46]</td>
<td>This function block is used to control and monitor a window ventilation system (FLS).</td>
</tr>
<tr>
<td>MPL_DamperLinearActuator [47]</td>
<td>This function block is used to control and monitor a drive of a damper and of a globe valve.</td>
</tr>
</tbody>
</table>

### 4.1.1.1 KL6771

This function block takes care of communication with the KL6771 MP-Bus master terminal. The KL6771 is configured with this block, and data exchange with the MP network is started.

Restrictions:
- The block must only be called once for each KL6771!
• It must be called in the same task as the actuator blocks!
• In the PLC project this block may only be called once per cycle!

VAR_INPUT

KL6771_IN : ARRAY [0..11] OF BYTE;

KL6771_IN: Is linked with the MP-Bus master terminal’s input process image.

VAR_OUTPUT

KL6771_OUT : ARRAY [0..11] OF BYTE;
strDataKL6771 : DataKL6771;
bError : BOOL;
ErrorCode : MP_Error;
BusLoad : INT := -1;

KL6771_OUT: Is linked with the MP-Bus master terminal’s output process image.

strDataKL6771: Data structure, which is linked with the different MP-Bus function blocks and includes the communication with the KL6771 function block (see DataKL6771 [58]).

bError: This output goes TRUE as soon as an error occurs. This error is described via the variable ErrorCode.

ErrorCode: This output outputs an error code in the event of an error (see MP_ERROR [56]). bError goes TRUE at the same time.

BusLoad: MP-Bus load in percent.

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>

4.1.1.2 MP_Address

This function block allows an MP-Bus slave to be addressed. It is also possible to use this function block to read the serial number of the slave.

MP_Address is used to transfer the address of the slave that is addressed or whose serial number is to be read. The serial number is read out on a rising edge of bReadAddr. It is output through MP_Serial_Numer_Out. A rising edge of bAddrAuto addresses a slave whose address is MP_Address. The function block waits for the time set in TMOUT for the transmission of the slave with its serial number. Transmission of the serial number is initiated differently from one slave to another. Please see the documentation for the MP-Bus device for how the serial number can be sent (in most cases there is a switch on the drive which will trigger this action when pressed). No telegrams are sent to the slaves during the time specified by TMOUT.

A rising edge at bAddrManual initiates manual addressing. This requires the serial number of the drive to be stored in MP_Serial_Number_IN. The serial number of the slave can be found on a sticker on the drive.

Sample: 00234-00016-002-031. Enter the following in the variable MP_Serial_Number_IN:
bBusy is set for as long as the function block is active. An error is indicated through bError, while the error number can be read with iErrorId.

**VAR_INPUT**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>MP_Address</td>
<td>USINT := 1;</td>
</tr>
<tr>
<td>bAddrAuto</td>
<td>BOOL;</td>
</tr>
<tr>
<td>bReadAddr</td>
<td>BOOL;</td>
</tr>
<tr>
<td>bAddrManual</td>
<td>BOOL;</td>
</tr>
<tr>
<td>MP_Serial_Number_IN</td>
<td>MP_Serial_Number;</td>
</tr>
<tr>
<td>TMOUT</td>
<td>TIME := t#30s;</td>
</tr>
<tr>
<td>strDataKL6771</td>
<td>DataKL6771;</td>
</tr>
</tbody>
</table>

**MP_Address**: The MP address that is to be used for the addressing or for reading the serial number. Valid values (1...8)

**bAddrAuto**: A rising edge starts the function block. The function block halts other MP-Bus communication, and waits until the time set through TMOUT has elapsed for an MP-Bus slave has transmitted its serial number, for instance in response to pressing a switch. The MP-Bus address that has been configured in the MP_Address variable is then transmitted to the slave.

**bReadAddr**: A rising edge starts the function block. The function block reads the serial number of the MP-Bus slave whose address is MP_Address.

**bAddrManual**: A rising edge starts the function block. The function block addresses the slave that has serial number MP_Serial_Number_IN. The slave's addresses configured with MP_Address.

**MP_Serial_Number_IN**: Serial number of the device (see MP_Serial_Number [59]).

**TMOUT**: Automatic addressing is interrupted after the time specified by TMOUT.

**strDataKL6771**: The data structure with which the KL6771() [16] function block must be linked (see DataKL6771 [58]).

**VAR_OUTPUT**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>bBusy</td>
<td>BOOL;</td>
</tr>
<tr>
<td>bError</td>
<td>BOOL;</td>
</tr>
<tr>
<td>iErrorId</td>
<td>MP_Error;</td>
</tr>
<tr>
<td>MP_Serial_Number_OUT</td>
<td>MP_Serial_Number;</td>
</tr>
</tbody>
</table>

**bBusy**: This bit is set for as long as the function block is active.

**bError**: This output goes TRUE as soon as an error occurs. This error is described via the iErrorId variable.

**iErrorId**: This output outputs an error code in the event of an error (see MP_Error [56]). bError goes TRUE at the same time.

**MP_Serial_Number_OUT**: The serial number of the addressed or queried slave (see MP_Serial_Number [59]).

**Requirements**

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>
4.1.1.3 MP_CMV

This function block is used to control and monitor a volume flow regulator.

**MP_Address** is used to specify the MP-Bus device with which the function block is to communicate. **bStart** activates communication with the MP-Bus device. **bBusy** indicates that the function block is active. If **bStart** remains TRUE, the device is addressed cyclically with a period specified by the time in **TMPolling**. The time should be set longer than 1 second. **bError** is used to indicate an error in communication with the drive. The type of the error can be read with **iErrorID**.

### VAR_INPUT

**MP_Address**: MP-Bus address of the slave.

**bStart**: A rising edge starts the function block. If this remains continuously TRUE, the function block will be activated cyclically with a period specified in **TMPolling**.

**bSet**: A rising edge writes the **MaxVol** and **MinVol** data to the drive.

**bOpen**: A rising edge opens the dampers of the drive, while a falling edge cancels the forced ventilation.

**bClose**: A rising edge closes the dampers of the drive, while a falling edge cancels the forced closure.

**bReset**: A rising edge resets the drive's error messages.

**iSensorTyp**: 0: no sensor connected, 1: digital sensor connected, 2: analog sensor connected (0..35 V), 3..6: Output of a resistor in Ohm (3...5 applies to PT1000, NI1000 and NI1000LuS; 6 applies to NTC). To convert to a temperature, use the corresponding conversion functions.

**SetPoint**: Set volume flow (0..100%).

**MaxVol**: Maximum volume flow (30..100%).

**MinVol**: Minimum volume flow (0...100%).

**strDataKL6771**: The data structure with which the **KL6771()** function block must be linked (see **DataKL6771**).
**TMpolling:** The time for which the function block should address the drive. Default 10 s, minimum time 1 s.

**VAR OUTPUT**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>bBusy</td>
<td>BOOL</td>
</tr>
<tr>
<td>bError</td>
<td>BOOL</td>
</tr>
<tr>
<td>iErrorId</td>
<td>MP_Error</td>
</tr>
<tr>
<td>ActValue</td>
<td>WORD</td>
</tr>
<tr>
<td>bMP_Sensor_Digi</td>
<td>BOOL</td>
</tr>
<tr>
<td>iParamSensor_Analog</td>
<td>INT</td>
</tr>
<tr>
<td>AirVolume_m3h</td>
<td>WORD</td>
</tr>
<tr>
<td>iNom_m3h</td>
<td>INT</td>
</tr>
<tr>
<td>act_MaxVol</td>
<td>INT</td>
</tr>
<tr>
<td>act_MinVol</td>
<td>INT</td>
</tr>
<tr>
<td>bErr_ActHunt</td>
<td>BOOL</td>
</tr>
<tr>
<td>bErr_MecTrv</td>
<td>BOOL</td>
</tr>
<tr>
<td>bErr_MecOvld</td>
<td>BOOL</td>
</tr>
<tr>
<td>rTemperature</td>
<td>LREAL</td>
</tr>
</tbody>
</table>

**bBusy:** This bit is set for as long as the function block is active.

**bError:** This output goes TRUE as soon as an error occurs. This error is described via the iErrorId variable.

**iErrorId:** This output outputs an error code in the event of an error (see MP_Error [56]). bError goes TRUE at the same time.

**ActValue:** Contains the current position of the drive (0..100%).

**bMP_Sensor_Digi:** If a digital sensor is connected, its state is indicated through this variable. iSensorTyp must be "1".

**iParamSensor_Analog:** If an analog sensor is connected, its value is indicated through this variable. iSensorTyp must be in the range "2..6".

**AirVolume_m3h:** Output of the volume flow rate in m³/h.

**iNom_m3h:** Nominal air volume flow in m³/h. This output is available from version 1.12.0. VAV is read and must be > 0. If 0, then the calculation of AirVolume_m3h is not correct.

**act_MaxVol:** Maximum set volume flow rate in %.

**act_MinVol:** Minimum set volume flow rate in %.

**bErr_ActHunt:** Drive error, "Regulating oscillation"; the drive is swinging backwards and forwards.

**bErr_MecTrv:** Drive error, "Positioning angle exceeded"; the drive has passed more than 10° beyond the adaptation position.

**bErr_MecOvld:** Drive error, "Overload"; the set position could not be reached.

**rTemperature:** Temperature in the duct in °C.

**Requirements**

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4022.14</td>
<td>Tc2_MPBus from 3.4.8.0</td>
</tr>
</tbody>
</table>
4.1.1.4 MP_DamperLinearActuator

This function block is used to control and monitor a drive of a damper and of a globe valve.

MP_Address is used to specify the MP-Bus device with which the function block is to communicate. bStart activates communication with the MP-Bus device. bBusy indicates that the function block is active. If bStart remains TRUE, the device is addressed cyclically with a period specified by the time in TMPolling. The time should be set longer than 1 second. bError is used to indicate an error in communication with the drive. The type of the error can be read with iErrorID.

A rising edge at bReset clears any pending error messages from the drive. This resets errors that affect the output variables bErr_MecOcld, bErr_ActHunt and bErrMecTrv. If the error itself is still present, the drive will set these error bits again.

SetPoint is used to adjust the position of the damper from 0...100%. The current position of the drive can be read through ActValue.

If a sensor is connected to the drive, iSensorTyp should be used to specify the sensor type. If no sensor is connected, the value "0" should be entered, or the variable should be left blank. A digital sensor should be parameterized with "1". The state of the sensor can be interrogated through bMP_Sensor_Digi. Analog sensors "2...6" are output in variable iMP_Sensor_Analog.

VAR_INPUT

MP_Address : USINT := 1;
bStart : BOOL;
bReset : BOOL;
iSensorTyp : INT;
SetPoint : USINT;
strDataKL6771 : DataKL6771;
TMpolling : TIME := t#10s;

MP_Address: MP-Bus address of the slave.

bStart: A rising edge starts the function block. If this remains continuously TRUE, the function block will be activated cyclically with a period specified in TMPolling.

bReset: A rising edge resets the drive's error messages.

iSensorTyp: 0: no sensor connected, 1: digital sensor connected, 2: analog sensor connected (0..35 V), 3..6: Output of a resistor in Ohm (3...5 applies to PT1000, NI1000 and NI1000LuS; 6 applies to NTC). To convert to a temperature, use the corresponding conversion functions.

SetPoint: 0..100 % the set damper position specified for the drive.

strDataKL6771: The data structure with which the KL6771() [16] function block must be linked (see DataKL6771 [58]).

TMpolling: The time for which the function block should address the drive. Default 10 s, minimum time 1 s.
VAR_OUTPUT

- bBusy : BOOL;  
  This bit is set for as long as the function block is active.
- ActValue : WORD;  
  Contains the current position of the drive (0..100%).
- bMP_Sensor_Digi : BOOL;  
  If a digital sensor is connected, its state is indicated through this variable. iSensorTyp must be "1".
- iMP_Sensor_Analog : INT;  
  If an analog sensor is connected, its value is indicated through this variable. iSensorTyp must be in the range "2..6".
- iErrorID : MP_Error;  
  This output outputs an error code in the event of an error (see MP_Error [56]). bError goes TRUE at the same time.
- bError : BOOL;  
  This output goes TRUE as soon as an error occurs. This error is described via the iErrorID variable.
- bErr_MecOvld : Drive error, "Overload"; the set position could not be reached.
- bErr_ActHunt : Drive error, "Regulating oscillation"; the drive is swinging backwards and forwards.
- bErr_MecTrv : Drive error, "Positioning angle exceeded"; the drive has passed more than 10° beyond the adaptation position.

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>

4.1.1.5 MP_EPIV

This function block is used to control a pressure-independent control ball valve. For more information please visit www.belimo.ch.
MP_Address is used to specify the MP-Bus device with which the function block is to communicate. bStart activates communication with the MP-Bus device. bBusy indicates that the function block is active. If bStart remains TRUE, the device is addressed cyclically with a period specified by the time in TMPolling. The time should be set longer than 1 second. bError is used to indicate an error in communication with the drive. The type of the error can be read with iErrorID.

VAR_INPUT

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MP_Address</td>
<td>USINT</td>
<td>MP-Bus address of the slave.</td>
</tr>
<tr>
<td>bStart</td>
<td>BOOL</td>
<td>A rising edge starts the function block.</td>
</tr>
<tr>
<td>bSet</td>
<td>BOOL</td>
<td>A rising edge writes the MaxVol and MinVol data to the drive.</td>
</tr>
<tr>
<td>bOpen</td>
<td>BOOL</td>
<td>A rising edge opens the dampers of the drive.</td>
</tr>
<tr>
<td>bClose</td>
<td>BOOL</td>
<td>A rising edge closes the dampers of the drive.</td>
</tr>
<tr>
<td>bReset</td>
<td>BOOL</td>
<td>A rising edge resets the drive's error messages.</td>
</tr>
<tr>
<td>iSensorTyp</td>
<td>INT</td>
<td>0: no sensor connected, 1: digital sensor connected, 2: analog sensor connected (0..35 V), 3..6: Output of a resistor in Ohm (3...5 applies to PT1000, NI1000 and NI1000LuS; 6 applies to NTC). To convert to a temperature, use the corresponding conversion functions.</td>
</tr>
<tr>
<td>SetPoint</td>
<td>USINT</td>
<td>Set volume flow (0..100 %).</td>
</tr>
<tr>
<td>MaxVol</td>
<td>WORD</td>
<td>Maximum volume flow (30..100 %).</td>
</tr>
<tr>
<td>strDataKL6771</td>
<td>DataKL6771</td>
<td>The data structure with which the KL6771() function block must be linked (see DataKL6771).</td>
</tr>
<tr>
<td>TMpolling</td>
<td>TIME</td>
<td>The time for which the function block should address the drive. Default 10 s, minimum time 1 s.</td>
</tr>
</tbody>
</table>

MP_Address: MP-Bus address of the slave.

bStart: A rising edge starts the function block. If this remains continuously TRUE, the function block will be activated cyclically with a period specified in TMPolling.

bSet: A rising edge writes the MaxVol and MinVol data to the drive.

bOpen: A rising edge opens the dampers of the drive, while a falling edge cancels the forced ventilation.

bClose: A rising edge closes the dampers of the drive, while a falling edge cancels the forced closure.

bReset: A rising edge resets the drive's error messages.

iSensorTyp: 0: no sensor connected, 1: digital sensor connected, 2: analog sensor connected (0..35 V), 3..6: Output of a resistor in Ohm (3...5 applies to PT1000, NI1000 and NI1000LuS; 6 applies to NTC). To convert to a temperature, use the corresponding conversion functions.

SetPoint: Set volume flow (0..100 %).

MaxVol: Maximum volume flow (30..100 %).

strDataKL6771: The data structure with which the KL6771() function block must be linked (see DataKL6771).

TMpolling: The time for which the function block should address the drive. Default 10 s, minimum time 1 s.

VAR_OUTPUT

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bBusy</td>
<td>BOOL</td>
<td>This bit is set for as long as the function block is active.</td>
</tr>
<tr>
<td>bError</td>
<td>BOOL</td>
<td>This output goes TRUE as soon as an error occurs. This error is described via the iErrorId variable.</td>
</tr>
<tr>
<td>iErrorId</td>
<td>MP_ERROR</td>
<td>This output outputs an error code in the event of an error (see MP_ERROR). bError goes TRUE at the same time.</td>
</tr>
<tr>
<td>ActValue</td>
<td>WORD</td>
<td>Contains the current position of the drive (0..100%).</td>
</tr>
<tr>
<td>bMP_Sensor_Digi</td>
<td>BOOL</td>
<td></td>
</tr>
<tr>
<td>bMP_Sensor_Analog</td>
<td>INT</td>
<td></td>
</tr>
<tr>
<td>Volume_lmin</td>
<td>WORD</td>
<td></td>
</tr>
<tr>
<td>iVnom_lmin</td>
<td>INT</td>
<td></td>
</tr>
<tr>
<td>act_MaxVol</td>
<td>INT</td>
<td></td>
</tr>
<tr>
<td>act_MinVol</td>
<td>INT</td>
<td></td>
</tr>
<tr>
<td>bErr_ActHunt</td>
<td>BOOL</td>
<td></td>
</tr>
<tr>
<td>bErr_MecTrv</td>
<td>BOOL</td>
<td></td>
</tr>
<tr>
<td>bErr_MecOvld</td>
<td>BOOL</td>
<td></td>
</tr>
<tr>
<td>bPositionSetByHand</td>
<td>BOOL</td>
<td></td>
</tr>
<tr>
<td>bSynchronisationActive</td>
<td>BOOL</td>
<td></td>
</tr>
</tbody>
</table>

bBusy: This bit is set for as long as the function block is active.

bError: This output goes TRUE as soon as an error occurs. This error is described via the iErrorId variable.

iErrorId: This output outputs an error code in the event of an error (see MP_ERROR). bError goes TRUE at the same time.

ActValue: Contains the current position of the drive (0..100%).
bMP_Sensor_Digi: If a digital sensor is connected, its state is indicated through this variable. iSensorTyp must be 1.

iMP_Sensor_Analog: If an analog sensor is connected, its value is indicated through this variable. iSensorTyp must be between 2 and 6.

Volume_lmin: Output of volume flow in lmin.

iVnom_lmin: Nominal air volume flow in lmin.

act_MaxVol: Maximum set volume flow rate in %.

act_MinVol: Minimum set volume flow in %.

bErr_ActHunt: Drive error "regulating oscillation": drive oscillates.

bErr_MecTrv: Drive error "Adjustment angle exceeded": drive rotation angle exceeded by more than 10° compared with adaptation.

bErr_MecOvld: Drive error "Overload": set position could not be reached.

bPositionSetByHand: The drive position was changed manually.

bSynchronisationActive: The drive is synchronizing.

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.32</td>
<td>Tc2_MPBus from 3.4.6.0</td>
</tr>
</tbody>
</table>

4.1.1.6 MP_EPIV_R6

This function block is used to control a control ball valve of series EP..R-R6+BAC.

MP_Address is used to specify the MP-Bus device with which the function block is to communicate. bStart activates communication with the MP-Bus device. bBusy indicates that the function block is active. If bStart remains TRUE, the device is addressed cyclically with a period specified by the time in TMPolling. The time should be set longer than 1 second. bError is used to indicate an error in communication with the drive. The type of the error can be read with iErrorID.

VAR_INPUT

MP_Address : USINT := 1;
bStart : BOOL;
strDataKL6771 : DataKL6771;
TMPolling : TIME := t#10s;
nOverrideControl_Write : E_MPBus_Override_6wayMPIV := MPBus_6wayMPIV_None;
rSetPoint_Write : LREAL;
**MP_Address**: MP-Bus address of the slave.

**bStart**: A rising edge starts the function block. If this remains continuously TRUE, the function block will be activated cyclically with a period specified in `TMPolling`.

**strDataKL6771**: The data structure with which the `KL6771()` function block must be linked (see `DataKL6771`).

**TMPolling**: The time for which the function block should address the drive. Default 10 s, minimum time 1 s.

**nOverrideControl_Write**: The relative set value is ignored in override control mode (see `E_MPBus_Override_6wayMPIV`). Overriding is disabled if the command is not repeated within 120 minutes.

**rSetPoint_Write**: 0…100%.

### VAR_OUTPUT

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>bBusy</td>
<td>BOOL</td>
</tr>
<tr>
<td>bError</td>
<td>BOOL</td>
</tr>
<tr>
<td>iErrorId</td>
<td>MP_Error</td>
</tr>
<tr>
<td>nOverrideControl_Read</td>
<td>E_MPBus_Override_6wayMPIV</td>
</tr>
<tr>
<td>rSetPoint_Read</td>
<td>LREAL</td>
</tr>
<tr>
<td>rRelativePos_Read</td>
<td>LREAL</td>
</tr>
<tr>
<td>rRelativeFlow_Read</td>
<td>LREAL</td>
</tr>
<tr>
<td>bErrorStateFlowSensorErr</td>
<td>BOOL</td>
</tr>
<tr>
<td>st_StateEV</td>
<td>st_StateEV</td>
</tr>
<tr>
<td>rAbsoluteFlow_UnitSel</td>
<td>LREAL</td>
</tr>
</tbody>
</table>

**bBusy**: This bit is set for as long as the function block is active.

**bError**: This output goes TRUE as soon as an error occurs. This error is described via the `iErrorId` variable.

**iErrorId**: This output outputs an error code in the event of an error (see `MP_Error`). `bError` goes TRUE at the same time.

**nOverrideControl_Read**: Current override control mode (see `E_MPBus_Override_6wayMPIV`).

**rSetPoint_Read**: Set value in % (0…100%).

**rRelativePos_Read**: Relative position in % (0…100%).

**rAbsolutePos_Read**: Absolute position in °.

**rRelativeFlow_Read**: Relative flow rate in % (0…100%).

**bErrorStateFlowSensorErr**: Flow sensor is faulty.

**bErrorStateActuator_can_not_move**: Drive cannot move.

**st_StateEV**: Only devices from 27 March 2014 (see `St_StateEV`).

**rAbsoluteFlow_UnitSel**: Absolute flow rate in UnitSel (0…4294967295).

### Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4022.14</td>
<td>Tc2_MPBus from 3.4.8.0</td>
</tr>
</tbody>
</table>

TwinCAT from v3.1.4022.14
4.1.1.7  MP_EPIV_R6_Parameter

This function block is used to parameterize drives of series EP..R-R6+BAC.

**MP_Address** is used to specify the MP-Bus device with which the function block is to communicate. **bStart** activates communication with the MP-Bus device. **bBusy** indicates that the function block is active. **bError** is used to indicate an error in communication with the drive. The type of the error can be read with **iErrorID**.

### VAR_INPUT

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MP_Address</td>
<td>USINT := 1;</td>
<td>MP-Bus address of the slave.</td>
</tr>
<tr>
<td>bStart</td>
<td>BOOL;</td>
<td>A rising edge starts the function block. If this remains continuously TRUE, the function block will be activated cyclically with a period specified in <strong>TMPolling</strong>.</td>
</tr>
<tr>
<td>bRead_Write</td>
<td>BOOL;</td>
<td>If FALSE then READ only; if TRUE then READ and WRITE.</td>
</tr>
<tr>
<td>strDataKL6771</td>
<td>DataKL6771;</td>
<td>The data structure with which the <strong>KL6771() [16]</strong> function block must be linked (see <strong>DataKL6771 [58]</strong>).</td>
</tr>
<tr>
<td>nUnitSelection_Write</td>
<td>E_MP_EP_R_R6_UnitSel := E_MP_l_h;</td>
<td>Scaling for <strong>rAbsoluteFlow_UnitSel</strong> (see <strong>E_MP_EP_R_R6_UnitSel [52]</strong>).</td>
</tr>
<tr>
<td>bControlMode_Write</td>
<td>BOOL;</td>
<td>FALSE: position-controlled, TRUE: flow-controlled.</td>
</tr>
<tr>
<td>rVmaxSeq1_Write</td>
<td>LREAL;</td>
<td>0...100%.</td>
</tr>
<tr>
<td>rVmaxSeq2_Write</td>
<td>LREAL;</td>
<td>0...100%.</td>
</tr>
</tbody>
</table>

### VAR_OUTPUT

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bBusy</td>
<td>BOOL;</td>
<td>This bit is set for as long as the function block is active.</td>
</tr>
<tr>
<td>bError</td>
<td>BOOL;</td>
<td>This output goes TRUE as soon as an error occurs. This error is described via the <strong>iErrorId</strong> variable.</td>
</tr>
<tr>
<td>iErrorId</td>
<td>MP_Error;</td>
<td></td>
</tr>
<tr>
<td>strMP_Serial_Number</td>
<td>MP_Serial_Number;</td>
<td></td>
</tr>
<tr>
<td>nUnitSelection_Read</td>
<td>E_MP_EP_R_R6_UnitSel;</td>
<td></td>
</tr>
<tr>
<td>bControlMode_Read</td>
<td>BOOL;</td>
<td></td>
</tr>
<tr>
<td>rVmaxSeq1_Read</td>
<td>LREAL;</td>
<td></td>
</tr>
<tr>
<td>rVmaxSeq2_Read</td>
<td>LREAL;</td>
<td></td>
</tr>
<tr>
<td>rAbsVnom_InitSel</td>
<td>LREAL;</td>
<td></td>
</tr>
<tr>
<td>rAbsVnom_l_h</td>
<td>LREAL;</td>
<td></td>
</tr>
<tr>
<td>rAbsVnom_gpm</td>
<td>LREAL;</td>
<td></td>
</tr>
</tbody>
</table>
iErrorId: This output outputs an error code in the event of an error (see MP_Error [56]). bError goes TRUE at the same time.

strMP_Serial_Number: Structure for the serial number (see MP_Serial_Number [59]).

nUnitSelection_Read: Setting the scaling (see E_MP_EP_R R6_UnitSel [52]).

bControlMode_Read: FALSE: position-controlled, TRUE: flow-controlled.

rVmaxSeq1_Read: Maximum sequence speed 1 in % (0...100%).

rVmaxSeq2_Read: Maximum sequence speed 2 in % (0...100%).

rAbsVnom_InitSel: Nominal volume (see rAbsoluteFlow_UnitSel).

rAbsVnom_l_h: Nominal volume in l/h (0...4294967295).

rAbsVnom_gpm: Nominal volume in gpm (0...4294967295).

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4022.14</td>
<td>Tc2_MPBus from 3.4.8.0</td>
</tr>
</tbody>
</table>

### 4.1.1.8 MP_EV

This function block is used to control a control ball valve of series P6..W..EV-BAC. For more information please visit www.belimo.ch.

MP_Address is used to specify the MP-Bus device with which the function block is to communicate. bStart activates communication with the MP-Bus device. bBusy indicates that the function block is active. If bStart remains TRUE, the device is addressed cyclically with a period specified by the time in TMPolling. The time should be set longer than 1 second. bError is used to indicate an error in communication with the drive. The type of the error can be read with iErrorID.
VAR_INPUT

MP_Address : USINT := 1;
bStart : BOOL;
strDataKL6771 : DataKL6771;
TMpolling : TIME := t#10s;
nOverrideControl : E_MPBus_Override := MPBus_Override_Auto;
nRelSetpoint : INT;

MP_Address: MP-Bus address of the slave.

bStart: A rising edge starts the function block. If this remains continuously TRUE, the function block will be activated cyclically with a period specified in TMpolling.

strDataKL6771: The data structure with which the KL6771() function block must be linked (see DataKL6771).

TMpolling: The time for which the function block should address the drive. Default 10 s, minimum time 1 s.

nOverrideControl: In Override mode the relative set value nRelSetpoint is ignored (see E_MPBus_Override).

VAR_OUTPUT

bBusy : BOOL;
bError : BOOL;
iErrorId : MP_ERROR;
nAbsPos : INT;
rT1_SI : REAL;
rT2_SI : REAL;
rDELTA_T : REAL;
rP_SI : REAL;
nE_COOLING_SI : DINT;
nE_HEATING_SI : DINT;
nRelFlow : INT;
rAbsFlow_SI : REAL;
nRelPos : INT;
nRelSetPointRead : INT;
bErrorStateT1 : BOOL;
bErrorStateT2 : BOOL;
bErrorStateFlowSensorErr : BOOL;
bErrorStateMechanicalOverload : BOOL;
st_StateEV : St_StateEV;
bPositionSetByHand : BOOL;

bBusy: This bit is set for as long as the function block is active.

bError: This output goes TRUE as soon as an error occurs. This error is described via the iErrorId variable.

iErrorId: This output outputs an error code in the event of an error (see MP_ERROR).
bError goes TRUE at the same time.

nAbsPos: Absolute Position in °.

rT1_SI: Temperature 1 (remote) in °C.

rT2_SI: Temperature 2 (embedded) in °C.

rDELTA_T: Delta temperature in °C.

rP_SI: Energy in kWh.

nE_COOLING_SI: Cooling energy in kWh.

nE_HEATING_SI: Heating energy in kWh.

nRelFlow: Relative flow rate in %.

rAbsFlow_SI: Absolute flow rate in l/min.

nRelPos: Relative Position in %.
nRelSetPointRead: The set value is interpreted either as position set value or as advance set value (relative to Vmax) in %.

bErrorStateT1: Error temperature sensor T1.

bErrorStateT2: Error temperature sensor T2.

bErrorStateFlowSensorErr: Flow sensor is faulty.

bErrorStateMechanicalOverload: Mechanical overload detected.

st_StateEV: Only devices from 27 March 2014 (see St_StateEV [60]).

bPositionSetByHand: The drive position was changed manually.

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>

4.1.1.9 MP_EV_Parameter

This function block is used to parameterize drives.

*MP_Address* is used to specify the MP-Bus device with which the function block is to communicate. *bStart* activates communication with the MP-Bus device. *bBusy* indicates that the function block is active. *bError* is used to indicate an error in communication with the drive. The type of the error can be read with *iErrorID*.

VAR_INPUT

| MP_Address : USINT := 1; |
| bStart : BOOL; |
| strDataKL6771 : DataKL6771; |
| dwPassword : DWORD; |
| nControlMode : E_MPBus_ControlMode := MPBus_ControlMode_Disable; |
| nDeltaTLimitation : E_MPBus_DeltaTLimitation := MPBus_DeltaTLimitation_Disable; |
| rSetPoint : REAL := 0.0; |
| rSpFlow_DeltaT_Si : REAL := 0.0; |

**MP_Address**: MP-Bus address of the slave.

**bStart**: A rising edge starts the function block. If this remains continuously TRUE, the function block will be activated cyclically with a period specified in *TMPolling*.

**strDataKL6771**: The data structure with which the *KL6771() [16]* function block must be linked (see DataKL6771 [58]).

**dwPassword**: The drive password is usually 0x0000.

**nControlMode**: Specifies the control mode (see *E_MPBus_ControlMode [54]*).

**nDeltaTLimitation**: dT limitation (see *E_MPBus_DeltaTLimitation [55]*).

**rSetPoint**: dT limit value.
rSpFlow_DeltaT_Si: Flow rate at saturation.

VAR OUTPUT

bBusy : BOOL;
bError : BOOL;
iErrorId : MP_ERROR;

bBusy: This bit is set for as long as the function block is active.
bError: This output goes TRUE as soon as an error occurs. This error is described via the iErrorId variable.
iErrorId: This output outputs an error code in the event of an error (see MP_ERROR[56]). bError goes TRUE at the same time.

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.32</td>
<td>Tc2_MPBus from 3.4.6.0</td>
</tr>
</tbody>
</table>

4.1.1.10 MP_MPX

For BELIMO room sensor MS24A-R..-MPX
- MS24A-R01-MPX temperature
- MS24A-R02-MPX temperature, CO2
- MS24A-R03-MPX temperature, VOC
- MS24A-R04-MPX temperature, CO2, VOC
- MS24A-R05-MPX temperature, air humidity
- MS24A-R06-MPX temperature, air humidity, CO2
- MS24A-R07-MPX temperature, air humidity, VOC
- MS24A-R08-MPX temperature, air humidity, CO2, VOC

VAR_INPUT

MP_Address : USINT := 1;
bStart : BOOL;
strDataKL6771 : DataKL6771;
TMpolling : TIME := t#10s;
bTemp_C_F : BOOL;

MP_Address: MP-Bus address of the slave.
bStart: A rising edge starts the function block. If this remains continuously TRUE, the function block will be activated cyclically with a period specified in TMPolling.

strDataKL6771: The data structure with which the KL6771() [16] function block must be linked (see DataKL6771 [58]).

TMPolling: The time for which the function block should address the drive. Default 10 s. Minimum time 1 s.

bTemp_C_F: FALSE = °C / TRUE = °F

VAR_OUTPUT

bBusy : BOOL;
bError : BOOL;
iErrorId : MP_Error;
strMPX_ERR : MP_BUS_MPX_ERROR;
iU : INT;
bDigital : BOOL;
iTemp : INT;
iCO2 : INT;
iVOC : INT;
iHumidity : INT;
bFlushStatus : BOOL;

bBusy: This bit is set for as long as the function block is active.

bError: This output goes TRUE as soon as an error occurs. This error is described via the iErrorId variable.

iErrorId: This output outputs an error code in the event of an error (see MP_Error [56]). bError goes TRUE at the same time.

strMPX_ERR: Sensor error messages (see MP_BUS_MPX_ERROR [58]).

iU: 0..10 V UNIT 1 mV

bDigital: DI 24 V

iTTemp: 0..50 °C Unit:0.01 °C

iCO2: 0..2000 ppm Unit:1 ppm

iVOC: 0..2000 ppm Unit:1 ppm (pseudo)

iHumidity: 10...90 % Unit: 0.01 %

bFlushStatus: VOC gradient threshold exceeded, FALSE = air quality OK, 1 TRUE = air quality not OK, flush

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>

4.1.1.11 MP_PTH

MP_PTH

- MP_Address
- bBusy
- bStart
- bError
- iSensorTyp
- iErrorId
- strDataKL6771
- iDampness
- TMpolling
- iPressure
- iTemperature
- iYSensor
- bY_PTH_Error
This function block is used to control and monitor a PTH sensor.

*MP_Address* is used to specify the MP-Bus device with which the function block is to communicate. *bStart* activates communication with the MP-Bus device. *bBusy* indicates that the function block is active. If *bStart* remains TRUE, the device is addressed cyclically with a period specified by the time in *TMPolling*. The time should be set longer than 1 second. *bError* is used to indicate an error in communication with the sensor. The type of the error can be read with *iErrorId*.

If an external sensor is connected to the sensor, *iSensorTyp* should be used to specify the sensor type. If no sensor is connected, the value "0" should be entered, or the variable should be left blank. A digital sensor should be parameterized with "3". The state of the sensor is output through the variable *iYSensor*.

**VAR_INPUT**

```plaintext
VAR_INPUT
MP_Address := USINT := 1;
bStart := BOOL;
iSensorTyp := INT;
strDataKL6771 := DataKL6771;
TMpolling := TIME := t#10s;
```

*MP_Address*: MP-Bus address of the slave.

*bStart*: A rising edge starts the function block. If this remains continuously TRUE, the function block will be activated cyclically with a period specified in *TMpolling*.

*iSensorTyp*: "0" or blank: no sensor is connected; "1": an analog sensor is connected with voltage output in mV; "2": an output of a resistance in ohms - 1.0 ohm; "3": an output of a resistance in ohms - 0.1 ohm; "4": digital sensor

*strDataKL6771*: The data structure with which the KL6771() function block must be linked (see DataKL6771 [58]).

*TMpolling*: The time for which the function block should address the drive. Default 10 s. Minimum time 1 s.

**VAR_OUTPUT**

```plaintext
VAR_OUTPUT
bBusy := BOOL;
bError := BOOL;
iErrorId := MP_Error;
iDampness := INT;
iPressure := INT;
iTemperature := INT;
iYSensor := INT;
byPTH_Error := BYTE;
```

*bBusy*: This bit is set for as long as the function block is active.

*bError*: This output goes TRUE as soon as an error occurs. This error is described via the *iErrorId* variable.

*iErrorId*: This output outputs an error code in the event of an error (see MP_Error [56]). *bError* goes TRUE at the same time.

*iDampness*: Relative humidity in 0.01%

*iPressure*: Differential pressure, output in 0.1 Pa.

*iTemperature*: Temperature in 0.01 °C

*iYSensor*: Y-input, iSensorTyp = "1": voltage 0..10 V output in mV; iSensorTyp = "2": resistance output in 1.0 ohm; iSensorTyp = "3": resistance output in 0.1 ohm; iSensorTyp = "3": digital switch 0 or 1

*byPTH_Error*: Sensor error - 0 - no error.
### byPTH_Error Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Servicing error</td>
</tr>
<tr>
<td>1</td>
<td>Error message, sensor faulty</td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Sensor (temperature/humidity) faulty</td>
</tr>
<tr>
<td>5</td>
<td>A/D converter (pressure) faulty</td>
</tr>
<tr>
<td>6</td>
<td>A/D converter (Y-input) faulty</td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
</tbody>
</table>

### Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>

#### 4.1.1.12 MP_RoomSensor

This function block is used to read room sensors. **MP_Address** is used to specify the MP-Bus device with which the function block is to communicate. **bStart** activates communication with the MP-Bus device. If **bStart** remains TRUE, the device is addressed cyclically with a period specified by the time in **TMPolling**. The time should be set longer than 1 second. **bError** is used to indicate an error in communication with the drive. The type of the error can be read with **iErrorID**.

**VAR_INPUT**

```plaintext
MP_Address : USINT := 1;
bStart : BOOL;
bReadDISensor : BOOL;
strDataKL6771 : DataKL6771;
TMpolling : TIME := t#10s;
```

**MP_Address**: MP-Bus address of the slave.

**bStart**: A rising edge starts the function block. If this remains continuously TRUE, the function block will be activated cyclically with a period specified in **TMPolling**.

**bReadDISensor**: If TRUE, the DI sensor is read and the result is available in **bDigitalInput**.

**strDataKL6771**: The data structure with which the **KL6771() [16]** function block must be linked (see **DataKL6771 [58]**).

**TMpolling**: The time for which the function block should address the drive. Default 10 s, minimum time 1 s.
VAR_OUTPUT

bBusy : BOOL;
bError : BOOL;
iErrorId : MP_ERROR;
rSensTemp : LREAL;
rDewPointTemp : LREAL;
rHumid : LREAL;
uiCo2 : UINT;
stUnit : STRING;
bDigitalInput : BOOL;
bSensorError : BOOL;
st_SensorType : STRING;

bBusy: This bit is set for as long as the function block is active.

bError: This output goes TRUE as soon as an error occurs. This error is described via the iErrorId variable.

iErrorId: This output outputs an error code in the event of an error (see MP_ERROR [ [ 56 ] ]). bError goes TRUE at the same time.

rSensTemp: Sensor temperature in °C or °F.

rDewPointTemp: Temperature of the calculated dew point in °C or °F.

rHumid: Humidity in percent (% 0.01).

uiCo2: CO2 content in ppm.

stUnit: C = °C or F = °F, ? = not read.

bDigitalInput: DI sensor read if bReadDI Sensor is TRUE.

bSensorError: One of the sensors has an error.

st_SensorType: Sensor type '?' = not read / type 'unknown' = number unknown.

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4024.10</td>
<td>Tc2_MPBus from 3.4.12.0</td>
</tr>
</tbody>
</table>

4.1.1.13 MP_RoomSensor_Parameter

This function block is used for the parameterization of room sensors.

MP_Address is used to specify the MP-Bus device with which the function block is to communicate. bRead reads the parameters, bWrite writes them to the room sensor. bBusy indicates that the function block is active. bError is used to indicate an error in communication with the drive. The type of the error can be read with iErrorID.

VAR_INPUT

MP_Address : USINT := 1;
bWrite : BOOL;
bRead : BOOL;
MP_Address: MP-Bus address of the slave.

bWrite: A positive edge starts the function block and writes the parameter.

bRead: A positive edge starts the function block and reads the parameter.

strDataKL6771: The data structure with which the KL6771() function block must be linked (see DataKL6771 [58]).

bUnitSelTemp: FALSE = °C, TRUE = °F.

iOffsetCo2_Write: OffsetCO2 [ppm] -500..500.

rOffsetTemp_Write: OffsetTemp [UnitSel] -15..15 °C (-27..27 °F)

rOffsetHumidity_Write: OffsetHumidity [%] -20..+20.

rOffsetDewPointTemp_Write: OffsetDewPointTemp [UnitSel] -15..15 °C (-27..27 °F).

VAR_OUTPUT

bBusy: This bit is set for as long as the function block is active.

bError: This output goes TRUE as soon as an error occurs. This error is described via the iErrorId variable.

iErrorId: This output outputs an error code in the event of an error (see MP_ERROR [56]). bError goes TRUE at the same time.

stUnit: C = °C or F = °F, ? = not read.

iOffsetCO2: OffsetCO2 [ppm].

rOffsetHumidity: OffsetHumidity [%] 0.01.

rOffsetTemp: OffsetTemp [°C or °F] 0.01.

rOffsetDewPointTemp: DewOffsetTemp [°C or °F] 0.01.

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4024.10</td>
<td>Tc2_MPBus from 3.4.12.0</td>
</tr>
</tbody>
</table>
4.1.1.14 MP_Smoker

This function block is used to control and monitor a fire damper.

**MP_Address** is used to specify the MP-Bus device with which the function block is to communicate. **bStart** activates communication with the MP-Bus device. **bBusy** indicates that the function block is active. If **bStart** remains TRUE, the device is addressed cyclically with a period specified by the time in **TMPolling**. The time should be set longer than 1 s, maximum 30 s. **bError** is used to indicate an error in communication with the drive. The type of the error can be read with **iErrorID**.

A rising edge at **bReset** clears any pending error messages from the drive. A telegram is thus sent to the drive that acknowledges the errors in the drive.

If they persist, they remain set. This applies to all **bErr_*** error bits.

**bOpen_Close** is used to open or close the fire damper. TRUE causes the fire damper to open, while FALSE closes it. **bOpen** indicates that the damper is open, **bClosed** indicates that it is closed. If both bits are FALSE, the drive is currently opening or closing.

A rising edge at **bTest** initiates a test run on the fire damper. Errors that have been set can be cleared with this if they are no longer present.

### VAR_INPUT

<table>
<thead>
<tr>
<th>MP_Address</th>
<th>bBusy</th>
</tr>
</thead>
<tbody>
<tr>
<td>bStart</td>
<td>bError</td>
</tr>
<tr>
<td>bTest</td>
<td>iErrorID</td>
</tr>
<tr>
<td>bOpen_Close</td>
<td>bOpen</td>
</tr>
<tr>
<td>bReset</td>
<td>bClosed</td>
</tr>
<tr>
<td>strDataKL6771</td>
<td>bErr_MecTrv</td>
</tr>
<tr>
<td>TMpolling</td>
<td>bErr_MecOmd</td>
</tr>
<tr>
<td></td>
<td>bErr_EmFlt</td>
</tr>
<tr>
<td></td>
<td>bErr_DmoTrv</td>
</tr>
<tr>
<td></td>
<td>bErr_DctAlm</td>
</tr>
<tr>
<td></td>
<td>bErr_SmAlm</td>
</tr>
</tbody>
</table>

**MP_Address**: MP-Bus address of the slave.

**bStart**: A rising edge starts the function block. If this remains continuously TRUE, the function block will be activated cyclically with a period specified in **TMPolling**.

**bTest**: A rising edge starts the test run at a fire damper.

**bOpen_Close**: TRUE opens a damper, while FALSE closes a damper.

**bReset**: A rising edge resets the drive's error messages.

**strDataKL6771**: The data structure with which the **KL6771()** function block must be linked (see **DataKL6771**).

**TMPolling**: The time for which the function block should address the drive. Default 10 s. Minimum time 1 s.
VAR_OUTPUT
bBusy : BOOL;
bError : BOOL;
iErrorId : MP_Error;
bOpen : BOOL;
bClosed : BOOL;
bErr_MecTrv : BOOL;
bErr_MecOvld : BOOL;
bErr_EmFlt : BOOL;
bErr_DmpTrv : BOOL;
bErr_DctAlm : BOOL;
bErr_SmAlm : BOOL;

bBusy: This bit is set for as long as the function block is active.

bError: This output goes TRUE as soon as an error occurs. This error is described via the iErrorId variable.

iErrorId: This output outputs an error code in the event of an error (see MP_Error). bError goes TRUE at the same time.

bOpen: Fire damper is open.

bClosed: Fire damper is closed.

bErr_MecTrv: Drive error, "Positioning angle exceeded"; the drive has passed more than 10° beyond the adaptation position.

bErr_MecOvld: Drive error, "Overload"; the set position could not be reached.

bErr_EmFlt: Drive error, "Safety-relevant error"; ambient temperature above 72°C or motor temperature above 85°C. Error can only be reset at the factory.

bErr_DmpTrv: Drive error "Damper test error" is canceled if the test run is OK.

bErr_DctAlm: Drive error, "Channel temperature too high"; the drive is swinging backwards and forwards.

bErr_SmAlm: Drive error, "Smoke alarm"

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>

4.1.1.15 MP_UST_3

This function block is used to control and monitor a multi-IO module UST3.
**MP_Address** is used to specify the MP-Bus device with which the function block is to communicate. **bStart** activates communication with the MP-Bus device. **bBusy** indicates that the function block is active. If **bStart** remains TRUE, the device is addressed cyclically with a period specified by the time in **TMPolling**. The time should be set to greater than one second. **bError** indicates an error in the communication with the drive. The type of the error can be read with **iErrorID**.

The input data **bRelay_R1** to **bRelay_R3** switch the relays R1 to R3 (PIN 15 to PIN 17). The variable **iUa** switches the analog output 0...10 V to PIN 5. One digit corresponds to 1 mV.

The data structure **SETTINGS** is used for parameterization of the UST3. The scaling of the analog input data can be set, and the resistance measurement can be enabled on PIN 4. The scaling of the measured resistance value can be changed for the resistance measurement. This can be done during operation. **iU1** is the analog input on PIN 3. One digit corresponds to 1 mV, or 250 µV if the scaling in the data structure **SETTINGS** was changed. The same applies to the analog inputs **iU2** (PIN 4) and **iU3** (PIN 7). **iU2** (PIN 4) and **iU3** (PIN 7) can also be used for resistance measurement. This must be set via the data structure **SETTINGS**. **bDigital_S1** to **bDigital_S3** correspond to the digital inputs of UST3, PIN 7 to PIN 9.
All data is automatically polled by the KL6771. The polling speed depends on the number of connected MP-Bus devices and the set polling time. The digital inputs are unsuitable for connecting buttons or sensors, which only issue short pulses. In order to be able to register a change in signal level reliably, it must be present for at least one second.

**VAR_INPUT**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>MP_Address</td>
<td>USINT := 1;</td>
</tr>
<tr>
<td>bStart</td>
<td>BOOL;</td>
</tr>
<tr>
<td>strDataKL6771</td>
<td>DataKL6771;</td>
</tr>
<tr>
<td>TMpolling</td>
<td>TIME := t#10s;</td>
</tr>
<tr>
<td>bRelay_R1</td>
<td>BOOL;</td>
</tr>
<tr>
<td>bRelay_R2</td>
<td>BOOL;</td>
</tr>
<tr>
<td>bRelay_R3</td>
<td>BOOL;</td>
</tr>
<tr>
<td>iUa</td>
<td>UINT;</td>
</tr>
<tr>
<td>SETTINGS</td>
<td>UST3_SET;</td>
</tr>
</tbody>
</table>

**MP_Address:** MP-Bus address of the slave.

**bStart:** A rising edge starts the function block. If this remains continuously TRUE, the function block will be activated cyclically with a period specified in *TMpolling*.

**strDataKL6771:** The data structure with which the KL6771() function block must be linked (see DataKL6771). 

**TMpolling:** The time for which the function block should address the drive. Default 10 s. Minimum time 1 s.

**bRelay_R1:** Relay PIN 15

**bRelay_R2:** Relay PIN 16

**bRelay_R3:** Relay PIN 17

**iUa:** Analog output PIN 5 (1 mV = 1 digit)

**SETTINGS:** Data structure for setting the scaling and the resistance measurement (see UST3_SET). 

**VAR_OUTPUT**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>bBusy</td>
<td>BOOL;</td>
</tr>
<tr>
<td>bError</td>
<td>BOOL;</td>
</tr>
<tr>
<td>iErrorId</td>
<td>MP_Error;</td>
</tr>
<tr>
<td>iU1</td>
<td>INT;</td>
</tr>
<tr>
<td>iU2_YSensor</td>
<td>INT;</td>
</tr>
<tr>
<td>iU3</td>
<td>INT;</td>
</tr>
<tr>
<td>bDigital_S1</td>
<td>BOOL;</td>
</tr>
<tr>
<td>bDigital_S2</td>
<td>BOOL;</td>
</tr>
<tr>
<td>bDigital_S3</td>
<td>BOOL;</td>
</tr>
<tr>
<td>iResistor</td>
<td>INT;</td>
</tr>
</tbody>
</table>

**bBusy:** This bit is set for as long as the function block is active.

**bError:** This output goes TRUE as soon as an error occurs. This error is described via the iErrorId variable.

**iErrorId:** This output outputs an error code in the event of an error (see MP_Error). bError goes TRUE at the same time.

**iU1:** analog input PIN 3 (1 digit = 1 mV or 1 digit = 250 µV)

**iU2_YSensor:** analog input PIN 4 (1 digit = 1 mV or 1 digit = 250 µV)

**iU3:** analog input PIN 7 (1 digit = 1 mV or 1 digit = 250 µV)

**bDigital_S1:** digital input PIN 8

**bDigital_S2:** digital input PIN 9

**bDigital_S3:** digital input PIN 7

**iResistor:** resistance value PIN 4
### Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>

#### 4.1.1.16 MP_VAV

This function block is used to control and monitor a volume flow regulator.

**MP_Address** is used to specify the MP-Bus device with which the function block is to communicate. **bStart** activates communication with the MP-Bus device. **bBusy** indicates that the function block is active. If **bStart** remains TRUE, the device is addressed cyclically with a period specified by the time in **TMPolling**. The time should be set longer than 1 second. **bError** is used to indicate an error in communication with the drive. The type of the error can be read with **iErrorID**.

A rising edge at **bReset** clears any pending error messages from the drive. This resets errors that affect the output variables **bErr_MecOcld**, **bErr_ActHunt** and **bErrMecTrv**. If the error itself is still present, the drive will set these error bits again.

**SetPoint** is used to adjust the volume flow rate from 0..100%. The current position of the damper can be read through **ActValue**.

If a sensor is connected to the drive, **iSensorTyp** should be used to specify the sensor type. If no sensor is connected, the value "0" should be entered, or the variable should be left blank. A digital sensor should be parameterized with "1". The state of the sensor can be interrogated through **bMP_Sensor_Digi**. Analog sensors "2...6" are output in variable **iMP_Sensor_Analog**.

A rising edge at the **bOpen** or **bClose** inputs opens or closes the damper of the drive. A falling edge at these two inputs clears the command again.

**MaxVol** and **MinVol** can be used to store a maximum and minimum flow rate in the drive. A rising edge at **bSet** writes the data to the drive. You can obtain the current value from the output data **act_MaxVol** and **act_MinVol**. The current volume flow rate is output in the **AirVolume_m3h** variable.

```plaintext
VAR_INPUT
MP_Address : USINT := 1;
bStart     : BOOL;
bSet       : BOOL;
bOpen      : BOOL;
bClose     : BOOL;
bReset     : BOOL;
iSensorTyp : INT;
SetPoint   : USINT;
MaxVol     : WORD;
MinVol     : WORD;
strDataKL6771 : DataKL6771;
TMPolling  : TIME := t#10s;
```


MP_Address: MP-Bus address of the slave.

bStart: A rising edge starts the function block. If this remains continuously TRUE, the function block will be activated cyclically with a period specified in TMPolling.

bSet: A rising edge writes the MaxVol and MinVol data to the drive.

bOpen: A rising edge opens the dampers of the drive, while a falling edge cancels the forced ventilation.

bClose: A rising edge closes the dampers of the drive, while a falling edge cancels the forced closure.

bReset: A rising edge resets the drive's error messages.

iSensorTyp: 0: no sensor connected, 1: digital sensor connected, 2: analog sensor connected (0..35 V), 3..6: Output of a resistor in Ohm (3..5 applies to PT1000, NI1000 and NI1000LuS; 6 applies to NTC). To convert to a temperature, use the corresponding conversion functions.

SetPoint: 0..100 % set volume flow

MaxVol: 30..100 % maximum volume flow

MinVol: 0..100 % minimum volume flow

strDataKL6771: The data structure with which the KL6771() [16] function block must be linked (see DataKL6771 [58]).

TMpolling: The time for which the function block should address the drive. Default 10 s, minimum time 1 s.

VAR_OUTPUT

bBusy: This bit is set for as long as the function block is active.

bError: This output goes TRUE as soon as an error occurs. This error is described via the iErrorId variable.

iErrorId: This output outputs an error code in the event of an error (see MP_Error [56]). bError goes TRUE at the same time.

ActValue: Contains the current position of the drive (0..100%).

bMP_Sensor_Digi: If a digital sensor is connected, its state is indicated through this variable. iSensorTyp must be "1".

iMP_Sensor_Analog: If an analog sensor is connected, its value is indicated through this variable. iSensorTyp must be in the range "2..6".

AirVolume_m3h: Output of the volume flow rate in m³/h.

act_MaxVol: Maximum set volume flow rate in %.

act_MinVol: Minimum set volume flow rate in %.

bErr_ActHunt: Drive error, "Regulating oscillation"; the drive is swinging backwards and forwards.

bErr_MecTrv: Drive error, "Positioning angle exceeded"; the drive has passed more than 10° beyond the adaptation position.

bErr_MecOvld: Drive error, "Overload"; the set position could not be reached.
## Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>

### 4.1.1.17 MP_VRU_Configuration

This function block is used to configure the VAV drives VRU-D3-BAC, VRU-M1-BAC and VRU-M1R-BAC (max. 8 slaves). For more information please visit www.belimo.ch.

**MP_Address** is used to specify the MP-Bus device with which the function block is to communicate. **bStart** activates communication with the MP-Bus device. **bBusy** indicates that the function block is active. **bError** is used to indicate an error in communication with the drive. The type of the error can be read with **iErrorID**.

**VAR_INPUT**

```
MP_Address : USINT := 1;
bStart : BOOL;
bRead_Write : BOOL;
dwPassword : DWORD;
strDataKL6771 : DataKL6771;
rMax : LREAL := 100;
rMin : LREAL;
bSetpointSource : BOOL;
bOperationMode : BOOL := TRUE;
nSensor1Type : E_MP_VRU_Sensor1Type := MPBus_VRU_Sensor_Active;
```

**MP_Address:** MP-Bus address of the slave.

**bStart:** A positive edge starts the function block.

**bRead_Write:** FALSE = READ only. TRUE = READ and WRITE.

**dwPassword:** The password for the drives. Usually 0x0000.

**strDataKL6771:** The data structure with which the KL6771() function block must be linked (see DataKL6771).

**rMax:** Max in % (20...100 %).

**rMin:** Min in % (0...rMax). **rMin** must be smaller than **rMax**.

**bSetpointSource:** TRUE = bus; FALSE = analog

**bOperationMode:** TRUE = overpressure; FALSE = underpressure

**nSensor1Type:** Sensor 1 type (see E_MP_VRU_Sensor1Type).
VAR_OUTPUT

bBusy : BOOL;
bError : BOOL;
iErrorId : MP_Error;
rNominalVolumetricFlow : LREAL;
rNominalDeltaPressure : LREAL;
rMinRead : LREAL;
rMaxRead : LREAL;
bSetpointSourcesRead : BOOL;
bOperationModeRead : BOOL;
nSensor1TypeRead : E_MP_VRU_Sensor1Type;
nApplication : E_MP_VRU_Application;
bControlMode : BOOL;
nRoomPressureCascade : E_MP_VRU_RoomPressureCascade;

bBusy: This bit is set for as long as the function block is active.

bError: This output goes TRUE as soon as an error occurs. This error is described via the iErrorId variable.

iErrorId: This output outputs an error code in the event of an error (see MP_ERROR[{56}]). bError goes TRUE at the same time.

rNominalVolumetricFlow: Nominal volume flow in m³/h (0...60.000).

rNominalDeltaPressure: Nominal differential pressure in Pa (0...10.000).

rMinRead: Min in % (0...rMax). rMin must be smaller than rMax.

rMaxRead: Max in % (20...100 %).

bSetpointSourcesRead: TRUE = bus; FALSE = analog

bOperationModeRead: TRUE = overpressure; FALSE = underpressure

nSensor1TypeRead: Sensor 1 type (see E_MP_VRU_Sensor1Type[{54}]).

nApplication: Visualization of the application selected by the manufacturer (see E_MP_VRU_Application [{52}]).

bControlMode: Visualization of the control function selected by the manufacturer. TRUE = volume flow control; FALSE = position control

nRoomPressureCascade: Room pressure cascade control (see E_MP_VRU_RoomPressureCascade[{54}]).

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4024.22</td>
<td>Tc2_MPBus from 3.4.13.0</td>
</tr>
</tbody>
</table>
4.1.1.18  MP_VRU_Process

This function block is suitable for the VAV drives VRU-D3-BAC, VRU-M1-BAC and VRU-M1R-BAC (max. 8 slaves). For more information please visit www.belimo.ch.

**MP_Address** is used to specify the MP-Bus device with which the function block is to communicate. **bStart** activates communication with the MP-Bus device. **bBusy** indicates that the function block is active. If **bStart** remains TRUE, the device is addressed cyclically with a period specified by the time in **TMPolling**. The time should be set longer than 1 second. **bError** is used to indicate an error in communication with the drive. The type of the error can be read with **iErrorID**.

### VAR_INPUT

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>MP_Address</td>
<td>USINT := 1;</td>
</tr>
<tr>
<td>bStart</td>
<td>BOOL;</td>
</tr>
<tr>
<td>strDataKL6771</td>
<td>DataKL6771;</td>
</tr>
<tr>
<td>TMpolling</td>
<td>TIME := t#10s;</td>
</tr>
<tr>
<td>rSetpoint</td>
<td>LREAL;</td>
</tr>
<tr>
<td>nOverrideControl</td>
<td>E_MP_VRU_OverrideControl;</td>
</tr>
<tr>
<td>nCommand</td>
<td>E_MP_VRU_Command;</td>
</tr>
<tr>
<td>nDataRead</td>
<td>BYTE;</td>
</tr>
</tbody>
</table>

**MP_Address**: MP-Bus address of the slave.

**bStart**: A positive edge starts the function block. If a TRUE is permanently present, the function block becomes cyclically active with the time **TMPolling**.

**strDataKL6771**: The data structure with which the KL6771() function block must be linked (see **DataKL6771**).

**TMpolling**: The time for which the function block should address the drive. Default 10 s, minimum time 1 s.

**rSetpoint**: Value in % (0.. 100 %).

**nOverrideControl**: Setpoint override (see **E_MP_VRU_OverrideControl**).

**nCommand**: Command for service and test functions of the drive (see **E_MP_VRU_Command**).

**nDataRead**: 0xFF - read all data; bit 0 - read relative position; bit 1 - read absolute position; bit 2 - read relative volume flow rate; bit 3 - read absolute volume flow rate; bit 4 - read value sensor 1; bit 5 - read analog setpoint; bit 6 - read relative differential pressure; bit 7 - read absolute differential pressure

### VAR_OUTPUT

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>bBusy</td>
<td>BOOL;</td>
</tr>
<tr>
<td>bError</td>
<td>BOOL;</td>
</tr>
<tr>
<td>iErrorId</td>
<td>MP_Error;</td>
</tr>
</tbody>
</table>
strMP_Serial_Number : MP_Serial_Number;
st_MP_VRU_ServiceInfo : St_MP_VRU_ServiceInfo;
rSetpoint_Read : LREAL;
nOverrideControl_Read : E_MP_VRU_OverrideControl;
nCommand_Read : E_MP_VRU_Command;
rRelativePosition : LREAL;
rAbsolutePosition : LREAL;
rRelativeVolumetricFlow : LREAL;
rAbsoluteVolumetricFlow : LREAL;
rSensor1Value : LREAL;
rSetpointAnalog : LREAL;
rRelativeDeltaPressure : LREAL;
rAbsoluteDeltaPressure : LREAL;

bBusy: This bit is set for as long as the function block is active.

bError: This output goes TRUE as soon as an error occurs. This error is described via the iErrorId variable.

iErrorId: This output outputs an error code in the event of an error (see MP_ERROR [56]). bError goes TRUE at the same time.

strMP_Serial_Number: Structure for the serial number (see MP_Serial_Number [59]).

st_MP_VRU_ServiceInfo: Fault and service information (see St_MP_VRU_ServiceInfo [59]).

rSetpoint_Read: Setpoint.

nOverrideControl_Read: Override (see E_MP_VRU_OverrideControl [53]).

nCommand_Read: Command (see E_MP_VRU_Command [53]).

rRelativePosition: Relative position in %. Value -1 means data disabled (see VAR_INPUT nDataRead).

rAbsolutePosition: Absolute position in °. Value -1 means data disabled (see VAR_INPUT nDataRead).

rRelativeVolumetricFlow: Relative volume flow rate in %. Value -1 means data disabled (see VAR_INPUT nDataRead).

rAbsoluteVolumetricFlow: Absolute volume flow rate in m³/h. Value -1 means data disabled (see VAR_INPUT nDataRead).

rSensor1Value: Value of sensor 1 in mV/Ohm. Value -1 means data disabled (see VAR_INPUT nDataRead).

rSetpointAnalog: Analog setpoint in %. Value -1 means data disabled (see VAR_INPUT nDataRead).

rRelativeDeltaPressure: Relative differential pressure %. Value -1 means data disabled (see VAR_INPUT nDataRead).

rAbsoluteDeltaPressure: Absolute differential pressure in Pa. Value -1 means data disabled (see VAR_INPUT nDataRead).

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4024.22</td>
<td>Tc2_MPBus from 3.4.13.0</td>
</tr>
</tbody>
</table>
4.1.1.19 MP_Window

This function block is used to control and monitor a window ventilation system (FLS).

MP_Address is used to specify the MP-Bus device with which the function block is to communicate. bStart activates communication with the MP-Bus device. bBusy indicates that the function block is active. If bStart remains TRUE, the device is addressed cyclically with a period specified by the time in TMPolling. The time should be set longer than 1 second. bError is used to indicate an error in communication with the FLS. The type of the error can be read with iErrorID.

ManualMode can be used to enable or disable manual operation. The type of ventilation can be specified through WindowSettings.

### VAR_INPUT

- MP_Address : USINT := 1;
- bStart : BOOL;
- WindowSettings : Data_Window;
- ManuelMode : BOOL;
- strDataKL6771 : DataKL6771;
- TMPolling : TIME := t#10s;

**MP_Address**: MP-Bus address of the slave.

**bStart**: A rising edge starts the function block. If this remains continuously TRUE, the function block will be activated cyclically with a period specified in TMPolling.

**WindowSettings**: Nominal ventilation settings (see DataWindow [51]).

**ManuelMode**: FALSE = manual operation permitted, TRUE = manual operation disabled.

**strDataKL6771**: The data structure with which the KL6771() [16] function block must be linked (see DataKL6771 [58]).

**TMPolling**: The time for which the function block should address the sensor. Default 10 s. Minimum time 1 s.

### VAR_OUTPUT

- bBusy : BOOL;
- bError : BOOL;
- iErrorId : MP_Error;
- WindowState : Data_Window;
- Window_Mech_Error : BOOL;
- Memory_Error : BOOL;

**bBusy**: This bit is set for as long as the function block is active.

**bError**: This output goes TRUE as soon as an error occurs. This error is described via the iErrorId variable.

**iErrorId**: This output outputs an error code in the event of an error (see MP_Error [56]). bError goes TRUE at the same time.

**WindowState**: Target settings for ventilation (see Data_Window [51]).

**Window_Mech_Error**: The window or drive is blocked.

**Memory_Error**: The drive has a memory error. Reprogram or replace.
Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>

### 4.1.1.20 MPL_DamperLinearActuator

This function block is used to control and monitor a drive of a damper and of a globe valve.

- **MP_Address** is used to specify the MP-Bus device with which the function block is to communicate. **bStart** activates communication with the MP-Bus device. **bBusy** indicates that the function block is active. If **bStart** remains TRUE, the device is addressed cyclically with a period specified by the time in **TMPolling**. The time should be set longer than 1 second. **bError** is used to indicate an error in communication with the drive. The type of the error can be read with **iErrorID**.

- **SetPoint** is used to adjust the position of the damper from 0..100%. The current position of the drive can be read through **ActValue**.

#### VAR_INPUT

- **MP_Address**: MP-Bus address of the slave.
- **bStart**: A rising edge starts the function block. If this remains continuously TRUE, the function block will be activated cyclically with a period specified in **TMPolling**.
- **SetPoint**: 0..100% the set damper position specified for the drive.
- **strDataKL6771**: The data structure with which the [KL6771] function block must be linked (see [DataKL6771]).
- **TMPolling**: The time for which the function block should address the drive. Default 10 s, minimum time 1 s.

#### VAR_OUTPUT

- **bBusy**: This bit is set for as long as the function block is active.
- **ActValue**: Contains the current position of the drive (0..100%).
- **iErrorID**: This output outputs an error code in the event of an error (see [MP_ERROR]). **bError** goes TRUE at the same time.
- **bError**: This output goes TRUE as soon as an error occurs. This error is described via the **iErrorID** variable.
4.1.2 Functions

<table>
<thead>
<tr>
<th>Function blocks</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NI1000_LuS_TO_INT [48]</td>
<td>This function calculates a temperature from the value of an NI1000 L&amp;S resistor.</td>
</tr>
<tr>
<td>NI1000_TO_INT [48]</td>
<td>This function calculates a temperature from the value of an NI1000 resistor.</td>
</tr>
<tr>
<td>NTC_TO_INT [49]</td>
<td>This function calculates a temperature from the value of an NTC resistor.</td>
</tr>
<tr>
<td>PT1000_TO_INT [49]</td>
<td>This function calculates a temperature from the value of a PT1000 resistor.</td>
</tr>
</tbody>
</table>

4.1.2.1 NI1000_LuS_TO_INT: INT

This function calculates a temperature from the value of an NI1000 L&S resistor.

Connect this function to iMP_Sensor_Analog. As output, you receive an INT variable that represents the temperature with a resolution of 0.01°C (20.5°C, for example, is represented as 2050).

The lowest valid value of 872 ohms corresponds to -26°C. If the value is smaller than this, 16#7FFD is output.

The largest valid value of 1586 ohms corresponds to 115°C. If the value is greater than this, 16#7FFE is output.

VAR_INPUT

MP_Value: WORD;

MP_Value: Input for an ohmic NI1000 L&S sensor

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>

4.1.2.2 NI1000_TO_INT: INT

This function calculates a temperature from the value of an NI1000 resistor.

Connect this function to iMP_Sensor_Analog. As output, you receive an INT variable that represents the temperature with a resolution of 0.01°C (20.5°C, for example, is represented as 2050).

The lowest valid value of 867 ohms corresponds to -25°C. If the value is smaller than this, 16#7FFD is output.

The largest valid value of 1583 ohms corresponds to 95°C. If the value is greater than this, 16#7FFE is output.

VAR_INPUT

MP_Value: WORD;

MP_Value: Input for an ohmic NI1000 sensor
### Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>

### NTC_TO_INT: INT

This function calculates a temperature from the value of an NTC resistor.

Connect this function to `iMP_Sensor_Analog`. As output, you receive an INT variable that represents the temperature with a resolution of 0.01°C (20.5°C, for example, is represented as 2050).

The lowest valid value of 104 ohms corresponds to 145°C. If the value is smaller than this, 16#7FFD is output.

The largest valid value of 48555 ohms corresponds to -20°C. If the value is greater than this, 16#7FFE is output.

**VAR_INPUT**

```
MP_Value : WORD;
```

**MP_Value**: Input for an ohmic NTC sensor

### PT1000_TO_INT: INT

This function calculates a temperature from the value of a PT1000 resistor.

Connect this function to `iMP_Sensor_Analog`. As output, you receive an INT variable that represents the temperature with a resolution of 0.01°C (20.5°C, for example, is represented as 2050).

The lowest valid value of 862 ohms corresponds to -35°C. If the value is smaller than this, 16#7FFD is output.

The largest valid value of 1592 ohms corresponds to 155°C. If the value is greater than this, 16#7FFE is output.

**VAR_INPUT**

```
MP_Value : WORD;
```

**MP_Value**: Input for an ohmic PT1000 sensor
## 4.1.3 Error codes

<table>
<thead>
<tr>
<th>Value (hex)</th>
<th>Value (dec)</th>
<th>Value (enum)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0000</td>
<td>0</td>
<td>NO_MP_ERROR</td>
<td>No error.</td>
</tr>
<tr>
<td>0x0001</td>
<td>1</td>
<td>WRONG_TERMINAL</td>
<td>Incorrect terminal connected.</td>
</tr>
<tr>
<td>0x0002</td>
<td>2</td>
<td>NO_ANSWER_FROM_KL6771</td>
<td>No answer from the KL6771 MP-bus master terminal. This message usually means that there is no connection to the terminal. Terminal linked to the variables in the System Manager? Terminal plugged in incorrectly? Everything corrected, everything translated and re-read into the System Manager?</td>
</tr>
<tr>
<td>0x0003</td>
<td>3</td>
<td>NO_LINK_TO_STRUCTURE_strData</td>
<td>Check the link to the structure DataKL6771.</td>
</tr>
<tr>
<td>0x000A</td>
<td>10</td>
<td>WRONG_MP_ADDRESS_IS_0</td>
<td>MP-bus address is 0. Only addresses between 1 and 8 are allowed.</td>
</tr>
<tr>
<td>0x000B</td>
<td>11</td>
<td>WRONG_MP_ADDRESS</td>
<td>MP-bus address is &gt;8. Only addresses between 1 and 8 are allowed.</td>
</tr>
<tr>
<td>0x0015</td>
<td>21</td>
<td>WRONG_SET_POINT</td>
<td>Incorrect set point.</td>
</tr>
<tr>
<td>0x0019</td>
<td>25</td>
<td>MP_BUS_TIMEOUT_NO_ANSWER_FROM_SLAVE</td>
<td>MP-Bus timeout, no response from slave.</td>
</tr>
<tr>
<td>0x001F</td>
<td>31</td>
<td>KL6771_TIME_OUT</td>
<td>KL6771 Timeout.</td>
</tr>
<tr>
<td>0x0020</td>
<td>32</td>
<td>MP_ADDRESS_IS_IN_USE</td>
<td>MP-Bus address is in use.</td>
</tr>
<tr>
<td>0x0021</td>
<td>33</td>
<td>MP_DISABLED</td>
<td>MP-Bus locked.</td>
</tr>
<tr>
<td>0x0057</td>
<td>87</td>
<td>MP_BUS_ERROR</td>
<td>MP-Bus error.</td>
</tr>
<tr>
<td>0x0058</td>
<td>88</td>
<td>MP_NO_ANSWER_ON_EVENT</td>
<td>MP-Bus no response to an event.</td>
</tr>
<tr>
<td>0x0059</td>
<td>89</td>
<td>MP_NO_ANSWER</td>
<td>MP-Bus no response.</td>
</tr>
<tr>
<td>0x005A</td>
<td>90</td>
<td>MP_COM_BREAK</td>
<td>MP-Bus abort communication.</td>
</tr>
<tr>
<td>0x0062</td>
<td>98</td>
<td>MP_LENGTH_PARITY_ERROR</td>
<td>MP-Bus length parity error.</td>
</tr>
<tr>
<td>0x0063</td>
<td>99</td>
<td>MP_CROSS_PARITY_ERROR</td>
<td>MP-Bus cross parity error.</td>
</tr>
<tr>
<td>0x0065</td>
<td>101</td>
<td>MP_MASTER_CONFLICT_ERROR</td>
<td>MP-Bus MASTER_CONFLICT_ERROR.</td>
</tr>
<tr>
<td>0x0066</td>
<td>102</td>
<td>MP_GAP_TIMEOUT_ERROR</td>
<td>MP-Bus GAP Timeout.</td>
</tr>
<tr>
<td>0x0067</td>
<td>103</td>
<td>MP_NO_ANSWER_SLAVE</td>
<td>MP-Bus no response from slave.</td>
</tr>
<tr>
<td>0x006E</td>
<td>110</td>
<td>MP_ANSWER_ERROR_FLAG</td>
<td>MP-Bus error bit in response telegram is set.</td>
</tr>
<tr>
<td>0x006F</td>
<td>111</td>
<td>MP_ANSWER_WRONG_LEN</td>
<td>MP-Bus wrong telegram length.</td>
</tr>
<tr>
<td>0x0070</td>
<td>112</td>
<td>MP_ANSWER_WRONG_TELEG</td>
<td>MP-Bus wrong telegram received.</td>
</tr>
<tr>
<td>0x0073</td>
<td>115</td>
<td>MP_ANSWER_WITH_ERROR</td>
<td>Response contains an error.</td>
</tr>
<tr>
<td>0x00C8</td>
<td>200</td>
<td>MP_ERROR_WrongDeviceFamily</td>
<td>Wrong device family.</td>
</tr>
<tr>
<td>0x800A</td>
<td>32778</td>
<td>MP_ANSWER_Reserve</td>
<td>Reserve.</td>
</tr>
<tr>
<td>0x800B</td>
<td>32779</td>
<td>MP_ANSWER_UnknownCommand</td>
<td>Unknown command.</td>
</tr>
<tr>
<td>0x800C</td>
<td>32780</td>
<td>MP_ANSWER_WrongOrNoPassword</td>
<td>Wrong or no password.</td>
</tr>
<tr>
<td>0x800D</td>
<td>32781</td>
<td>MP_ANSWER_CommandExecution</td>
<td>Command execution.</td>
</tr>
<tr>
<td>0x800E</td>
<td>32782</td>
<td>MP_ANSWER_ParameterError</td>
<td>Parameter error.</td>
</tr>
<tr>
<td>0x800F</td>
<td>32783</td>
<td>MP_ANSWER_UnknownId</td>
<td>Unknown ID.</td>
</tr>
<tr>
<td>0x8010</td>
<td>32784</td>
<td>MP_ANSWER_SizeMismatch</td>
<td>Wrong size.</td>
</tr>
<tr>
<td>0x8011</td>
<td>32785</td>
<td>MP_ANSWER_IllegalBlockNr</td>
<td>Invalid block number.</td>
</tr>
<tr>
<td>0x8012</td>
<td>32786</td>
<td>MP_ANSWER_InternalBusBusy</td>
<td>Internal bus is busy.</td>
</tr>
<tr>
<td>0x80FF</td>
<td>33023</td>
<td>MP_ANSWER_ReservedForFuture</td>
<td>Reserve.</td>
</tr>
</tbody>
</table>
4.2  **DUTs**

### 4.2.1  Enums

<table>
<thead>
<tr>
<th>Data types</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data_Window</td>
<td>Ventilation type.</td>
</tr>
<tr>
<td>E_MP_EP_R6_UnitSel</td>
<td>Scaling.</td>
</tr>
<tr>
<td>E_MP_VRU_Application</td>
<td>Visualization of the application selected by the manufacturer.</td>
</tr>
<tr>
<td>E_MP_VRU_Command</td>
<td>Commands for service and test functions of the drive.</td>
</tr>
<tr>
<td>E_MP_VRU.OverrideControl</td>
<td>Setpoint override.</td>
</tr>
<tr>
<td>E_MP_VRU_RoomPressureCascade</td>
<td>Room pressure cascade control.</td>
</tr>
<tr>
<td>E_MP_VRU_Sensor1Type</td>
<td>External sensor at input S1.</td>
</tr>
<tr>
<td>E_MPBus_ControlMode</td>
<td>Control mode.</td>
</tr>
<tr>
<td>E_MPBus_DeltaLimitation</td>
<td>Delta T (dT) limitation.</td>
</tr>
<tr>
<td>E_MPBus.Override</td>
<td>Override mode.</td>
</tr>
<tr>
<td>E_MPBus.Override_6wayMPIV</td>
<td>Override control mode.</td>
</tr>
<tr>
<td>MP_ERROR</td>
<td>Error messages.</td>
</tr>
<tr>
<td>UST3_EX</td>
<td>Voltage scaling.</td>
</tr>
<tr>
<td>UST3_R_SET</td>
<td>Resistance scaling.</td>
</tr>
</tbody>
</table>

#### 4.2.1.1  Data_Window

This ENUM can be used to specify the ventilation method.

```plaintext
TYPE Data_Window :
  (
    Window_Close := 8,
    Window_Unlock := 9,
    Window_Open := #160A,
    Window_20 := #160B,
    Window_40 := #160C,
    Window_60 := #160D,
    Window_80 := #160E,
    Window_100 := #160F,
    Auto_Close := 1,
    Auto_5_15min := 2,
    Auto_8_30min := 3,
    Auto_10_50min := 4,
    Auto_open := 5,
    Auto := 0
  )
END_TYPE
```

**Window_Close**: Closes the window

**Window_Unlock**: If you use "Unlock", use the switch in the window to restart the MP-Bus communication

**Window_Open**: Opens the window

**Window_20**: Opens the window 20%

**Window_40**: Opens the window 40%

**Window_60**: Opens the window 60%

**Window_80**: Opens the window 80%
Window_100: Opens the window 100%
Auto_Close: Automatic closing
Auto_5_15min: Automatic ventilation every 5..15 minutes
Auto_8_30min: Automatic ventilation every 8..30 minutes
Auto_10_50min: Automatic ventilation every 10..50 minutes
Auto_open: Automatic ventilation open
Auto: Automatic mode

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>

4.2.1.2 E_MP_EP_R_R6_UnitSel

Scaling.

```plaintext
TYPE E_MP_ER_R_R6_UnitSel :
{
  E_MP_m3_s := 0,
  E_MP_m3_h := 1,
  E_MP_l_s := 2,
  E_MP_l_min := 3,
  E_MP_l_h := 4,
  E_MP_gpm := 5,
  E_MP_cfm := 6
};
END_TYPE
```

E_MP_m3_s: Sets scaling to m3/s.
E_MP_m3_h: Sets scaling to m3/h.
E_MP_l_s: Sets scaling to l/s.
E_MP_l_min: Sets scaling to l/min.
E_MP_l_h: Sets scaling to l/h.
E_MP_gpm: Sets scaling to gpm.
E_MP_cfm: Sets scaling to cfm.

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4022.14</td>
<td>Tc2_MPBus from 3.4.8.0</td>
</tr>
</tbody>
</table>

4.2.1.3 E_MP_VRU_Application

Visualization of the application selected by the manufacturer.

```plaintext
TYPE E_MP_VRU_Application :
{
  MPBus_VRU_Application_FlowControl := 0,
  MPBus_VRU_Application_PressureControl := 1,
  MPBus_VRU_Application_RoomPressureControl := 2,
  MPBus_VRU_Application_FlowMeasurement := 3
};
END_TYPE
```

MPBus_VRU_Application_FlowControl: Flow control.
MPBus_VRU_Application_PressureControl: Pressure control.

MPBus_VRU_Application_RoomPressureControl: Room pressure control.

MPBus_VRU_Application_FlowMeasurement: Flow measurement.

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4024.22</td>
<td>Tc2_MPBus from 3.4.13.0</td>
</tr>
</tbody>
</table>

4.2.1.4  E_MP_VRU_Command

Commands for service and test functions of the drive.

```pascal
TYPE E_MP_VRU_Command :

MPBus_VRU_Command_None := 0,
MPBus_VRU_Command_Adaption := 1,
MPBus_VRU_Command_Test := 2,
MPBus_VRU_Command_Sync := 3
); END_TYPE
```

MPBus_VRU_Command_None: None.

MPBus_VRU_Command_Adaption: Adaptation.

MPBus_VRU_Command_Test: Test.

MPBus_VRU_Command_Sync: Sync.

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4024.22</td>
<td>Tc2_MPBus from 3.4.13.0</td>
</tr>
</tbody>
</table>

4.2.1.5  E_MP_VRU_OverrideControl

Setpoint override.

```pascal
TYPE E_MP_VRU_OverrideControl :

MPBus_VRU_Override_None := 0,
MPBus_VRU_Override_Open := 1,
MPBus_VRU_Override_Close := 2,
MPBus_VRU_Override_Max := 3,
MPBus_VRU_Override_Min := 4,
MPBus_VRU_Override_Reserve := 5,
MPBus_VRU_Override_Motor_Stop := 6,
MPBus_VRU_Override_Vnom_Pnom := 7
); END_TYPE
```

MPBus_VRU_Override_None: None.

MPBus_VRU_Override_Open: Open.

MPBus_VRU_Override_Close: Close.

MPBus_VRU_Override_Max: Maximum.

MPBus_VRU_Override_Min: Minimum.

MPBus_VRU_Override_Reserve: Reserve.

MPBus_VRU_Override_Motor_Stop: Motor stop.
MPBus_VRU_Override_Vnom_Pnom: Vnom / Pnom.

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4024.22</td>
<td>Tc2_MPBus from 3.4.13.0</td>
</tr>
</tbody>
</table>

4.2.1.6 E_MP_VRU_RoomPressureCascade

Room pressure cascade control.

```
TYPE E_MP_VRU_RoomPressureCascade :
{
  MPBus_VRU_RoomPressureCascade_Disabled := 0,
  MPBus_VRU_RoomPressureCascade_Enabled := 1,
  MPBus_VRU_RoomPressureCascade_EnabledFast := 2
};
END_TYPE
```

MPBus_VRU_RoomPressureCascade_Disabled: Room pressure cascade control disabled.

MPBus_VRU_RoomPressureCascade_Enabled: Room pressure cascade control enabled.

MPBus_VRU_RoomPressureCascade_EnabledFast: Room pressure cascade control enabled (fast).

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4024.22</td>
<td>Tc2_MPBus from 3.4.13.0</td>
</tr>
</tbody>
</table>

4.2.1.7 E_MP_VRU_Sensor1Type

External sensor at input S1.

```
TYPE E_MP_VRU_Sensor1Type :
{
  MPBus_VRU_Sensor_None := 0,
  MPBus_VRU_Sensor_Active := 1,
  MPBus_VRU_Sensor_Passive := 2,
  MPBus_VRU_Sensor_Switch := 4
};
END_TYPE
```

MPBus_VRU_Sensor_None: None.

MPBus_VRU_Sensor_Active: Active.


MPBus_VRU_Sensor_Switch: Switch.

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4024.22</td>
<td>Tc2_MPBus from 3.4.13.0</td>
</tr>
</tbody>
</table>

4.2.1.8 E_MPBus_ControlMode

Control mode.

```
TYPE E_MPBus_ControlMode :
{
  MPBus_ControlMode_PosCtrl := 0,
  MPBus_ControlMode_FlowCtrl := 1,
  MPBus_ControlMode_PowerCtrl := 2,
};
END_TYPE
```

54 Version: 1.8 TE1000
MPBus_ControlMode_Disable := 16#FF
); END_TYPE

**MPBus_ControlMode_PosCtrl:** Control based on position.

**MPBus_ControlMode_FlowCtrl:** Control based on flow rate.

**MPBus_ControlMode_PowerCtrl:** Control based on amount of energy.

**MPBus_ControlMode_Disable:** Disabled.

### Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>

#### 4.2.1.9 **E_MPBus_DeltaTLimitation**

Delta T (dT) limitation. Details can be found in the Belimo Energy Valve documentation.

**TYPE E_MPBus_DeltaTLimitation :**

```plaintext
{  MPBus_DeltaTLimitation_Disable := 0,
   MPBus_DeltaTLimitation_dT_Manager := 1,
   MPBus_DeltaTLimitation_dT_ManagerScal := 2
};
END_TYPE
```

**MPBus_DeltaTLimitation_Disable:** dT disabled.

**MPBus_DeltaTLimitation_dT_Manager:** Simple dT limitation.

**MPBus_DeltaTLimitation_dT_ManagerScal:** Extended dT limitation.

### Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.32</td>
<td>Tc2_MPBus from 3.4.6.0</td>
</tr>
</tbody>
</table>

#### 4.2.1.10 **E_MPBus_Override**

Override mode.

**TYPE E_MPBus_Override :**

```plaintext
{  MPBus_Override_None := 0,
   MPBus_Override_Auto := 1,
   MPBus_Override_Close := 2,
   MPBus_Override_Open := 3,
   MPBus_Override_Vnom := 4,
   MPBus_Override_Vmax := 5,
   MPBus_Override_Stop := 6,
   MPBus_Override_Pnom := 7,
   MPBus_Override_Pmax := 8
};
END_TYPE
```

### Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>
4.2.1.11 E_MPBus Override 6wayMPIV

Override control mode.

TYPE E_MPBus_OVERRIDE_6wayMPIV :
{
  MPBus_6wayMPIV_None    := 0,
  MPBus_6wayMPIV_Seq1Open := 1,
  MPBus_6wayMPIV_Seq2Open := 2,
  MPBus_6wayMPIV_Close   := 3,
  MPBus_6wayMPIV_Seq1Vmax := 4,
  MPBus_6wayMPIV_Seq2Vmax := 5
};
END_TYPE

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4022.14</td>
<td>Tc2_MPB from 3.4.8.0</td>
</tr>
</tbody>
</table>

4.2.1.12 MP_ERROR

Library error messages

TYPE MP_ERROR :
{
  NO_MP_ERROR             := 0,
  WRONG_TERMINAL          := 1,
  NO_ANSWER_FROM_KL6771   := 2,
  WRONG_MP_ADDRESS_IS_0   := 10,
  WRONG_MP_ADDRESS        := 11,
  WRONG_SET_POINT         := 21,
  MP_BUS_TIMEOUT_NO_ANSWER_FROM_SLAVE := 25,
  KL6771_TIME_OUT         := 31,
  MP_ADDRESS_IS_IN_USE   := 32,
  MP_DISABLED             := 33,
  MP_BUS_ERROR            := 87,
  MP_NO_ANSWER_ON_EVENT   := 88,
  MP_NO_ANSWER            := 89,
  MP_COM_BREAK            := 90,
  MP_LENGTH_PARITY_ERROR  := 98,
  MP_CROSS_PARITY_ERROR   := 99,
  MP_MASTER_CONFLICT_ERROR := 101,
  MP_GAP_TIMEOUT_ERROR    := 102,
  MP_NO_ANSWER_SLAVE      := 103,
  MP_ANSWER_ERROR_FLAG    := 110,
  MP_ANSWER_WRONG_LEN     := 111,
  MP_ANSWER_WRONG_TELEG   := 112
};
END_TYPE

NO_MP_ERROR: No error

WRONG_TERMINAL: Incorrect terminal connected

NO_ANSWER_FROM_KL6771: No response from KL6771. This message usually means that there is no connection to the terminal. Terminal linked to the variables in the System Manager? Terminal plugged in incorrectly? Everything revised, compiled and read again in the System Manager?

WRONG_MP_ADDRESS_IS_0: MP-Bus address is 0. Only addresses between 1 and 8 are allowed.

WRONG_MP_ADDRESS: MP-Bus address is >8. Only addresses between 1 and 8 are allowed.

WRONG_SET_POINT: Incorrect setpoint

MP_BUS_TIMEOUT_NO_ANSWER_FROM_SLAVE: MP-Bus timeout, no response from slave

KL6771_TIME_OUT: KL6771 Timeout

MP_ADDRESS_IS_IN_USE: MP-Bus address is in use

MP_DISABLED: MP-Bus locked
MP_BUS_ERROR: MP-Bus error
MP_NO_ANSWER_ON_EVENT: MP-Bus no response to an event
MP_NO_ANSWER: MP-Bus no response
MP_COM_BREAK: -MP-Bus abort communication
MP_LENGTH_PARITY_ERROR: MP-Bus length parity error
MP_CROSS_PARITY_ERROR: MP-Bus cross parity error
MP_MASTER_CONFLICT_ERROR: MP-Bus MASTER_CONFLICT_ERROR
MP_GAP_TIMEOUT_ERROR: MP-Bus GAP Timeout
MP_NO_ANSWER_SLAVE: MP-Bus no response from slave
MP_ANSWER_ERROR_FLAG: MP-Bus error bit in response telegram is set
MP_ANSWER_WRONG_LEN: MP-Bus wrong telegram length
MP_ANSWER_WRONG_TELEG: MP-Bus wrong telegram received.

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>

4.2.1.13 UST3_EX

Voltage scaling

```plaintext
TYPE UST3_EX :
{
    Ex_1mV := 0,
    Ex_250uV := 1
}
END_TYPE
```

Ex_1mV: Scaling 0..11 V
Ex_250uV: Scaling 0..3 V

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>

4.2.1.14 UST3_R_SET

Resistance scaling

```plaintext
TYPE UST3_R_SET :
{
    R_1Ohm := 0,
    R_250mOhm := 1,
    R_4Ohm := 2
}
END_TYPE
```

R_1Ohm: Scaling 0..20 kOhm
R_250mOhm: Scaling 0..5 kOhm
R_4Ohm: Scaling 0..262 kOhm
### Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>

#### 4.2.2 Structures

<table>
<thead>
<tr>
<th>Data types</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DataKL6771</td>
<td>Links the send and receive blocks with the function block KL6771.</td>
</tr>
<tr>
<td>MP_BUS_MPX_ERROR</td>
<td>Error messages of the &quot;MPX&quot; sensors.</td>
</tr>
<tr>
<td>MP_Serial_Number</td>
<td>Serial number of the device.</td>
</tr>
<tr>
<td>St_MP_VRU_ServicInfo</td>
<td>Fault and service information.</td>
</tr>
<tr>
<td>St_StateEV</td>
<td>Information on the state of the EV.</td>
</tr>
<tr>
<td>UST3_SET</td>
<td>Data structure for setting the scaling and the resistance measurement.</td>
</tr>
</tbody>
</table>

##### 4.2.2.1 DataKL6771

Links the send and receive blocks with the function block KL6771

```plaintext
TYPE DataKL6771 :
    STRUCT
        OrderNumber : BYTE;
        ReciveData : BOOL;
        SendData : BOOL;
        Error : BOOL;
        ErrorID : MP_Error;
        pNumber : DWORD;
    END_STRUCT
END_TYPE
```

- **OrderNumber**: Internal byte
- **ReceiveData**: Data is being received.
- **SendData**: Data is being sent.
- **Error**: This output goes TRUE as soon as an error occurs. This error is described via the ErrorID variable.
- **ErrorID**: This output outputs an error code in the event of an error (see MP_Error). Error goes TRUE at the same time.
- **pNumber**: Internal pointer

#### Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>

##### 4.2.2.2 MP_BUS_MPX_ERROR

Error messages of the "MPX" sensors (function block MP_MPX).  

```plaintext
TYPE MP_BUS_MPX_ERROR :
    STRUCT
        MP_BUS_MPX_TempSensorErr : BOOL;
        MP_BUS_MPX_HumiditySensorErr : BOOL;
        MP_BUS_MPX_CO2SensorErr : BOOL;
        MP_BUS_MPX_VocSensorErr : BOOL;
    END_STRUCT
END_TYPE
```

- **MP_BUS_MPX_TempSensorErr**: The temperature sensor is faulty
**MP_BUS_MPX_HumiditySensorErr**: The humidity sensor is faulty

**MP_BUS_MPX_CO2SensorErr**: The CO2 sensor is faulty

**MP_BUS_MPX_VocSensorErr**: The VOC sensor is faulty

**Requirements**

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>

### 4.2.2.3 MP_Serial_Number

Serial number of the device

```plaintext
TYPE MP_Serial_Number :
  STRUCT
    YearAndWeek : WORD;
    DayAndNumber : WORD;
    DeviceFamily : BYTE;
    TestStation : BYTE;
    FamilySuffix : BYTE;
  END_STRUCT
END_TYPE
```

- **YearAndWeek**: Year and week
- **DayAndNumber**: Day and number
- **DeviceFamily**: Device family
- **TestStation**: Test station
- **FamilySuffix**: Device family suffix

### Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>

### 4.2.2.4 St_MP_VRU_ServiceInfo

Fault information.

```plaintext
TYPE St_MP_VRU_ServiceInfo :
  STRUCT
    bError_dP_Sensor : BOOL;
    bReverseAirflowDetected : BOOL;
    bAirflowNotReached : BOOL;
    bFlowInClosedPosition : BOOL;
    bInternalActivity : BOOL;
    bGearDisengaged : BOOL;
    bBusWatchdogTriggered : BOOL;
    bActuatorDoseNotFitToApplication : BOOL;
    bPressureSensorWrongConnected : BOOL;
    bPressureSensorNotReached : BOOL;
    bError_dP_SensorOutOfRange : BOOL;
  END_STRUCT
END_TYPE
```

- **bError_dP_Sensor**: Error dp sensor.
- **bReverseAirflowDetected**: Reverse airflow detected.
- **bAirflowNotReached**: Airflow not reached.
- **bFlowInClosedPosition**: Flow in closed position.
**bInternalActivity**: Internal activity.

**bGearDisengaged**: Gear unit disengaged.

**bBusWatchdogTriggered**: Bus watchdog triggered.

**bActuatorDoseNotFitToApplication**: Actuator does not fit the application.

**bPressSensorWrongConnected**: Pressure sensor connected incorrectly.

**bPressureSensorNotReached**: Pressure sensor not reached.

**bError_dP_SensorOutOfRange**: dP sensor out of range.

### Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4024.22</td>
<td>Tc2_MPBus from 3.4.13.0</td>
</tr>
</tbody>
</table>

#### 4.2.2.5 St_StateEV

Information on the state of the EV.

```plaintext
TYPE St_StateEV :
STRUCT
  bFlow_with_closed_valve : BOOL;
  bAir_bubbles   : BOOL;
  bFlow_not_reached: BOOL;
  bPower_not_realized: BOOL;
  bGear_disengaged : BOOL;
END_STRUCT
END_TYPE


bAir_bubbles: Too many air bubbles (system venting inadequate). Flow measurement no longer exact. EV switches from volume flow control to position control.

bFlow_not_reached: Flow rate not reached, despite the fact that the ball valve is fully open. Check hydraulics, switch on pump or increase pump pressure.

bPower_not_realized: Power not reached. In addition to position control and volume flow control, the EV can be used to supply power. Primary side provides too little flow or dT.

bGear_disengaged: Gear disengagement active. Manual adjustment possible on site. For other MP drives this information can be read with MP_Get_State.

### Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.32</td>
<td>Tc2_MPBus from 3.4.6.0</td>
</tr>
</tbody>
</table>

#### 4.2.2.6 UST3_SET

Data structure for setting the scaling and the resistance measurement

```plaintext
TYPE UST3_SET :
STRUCT
  E1   : UST3_Ex;
  E2   : UST3_Ex;
  E3   : UST3_Ex;
  R_SET : UST3_R_set;
  R_ON_OFF : BOOL;
END_STRUCT
END_TYPE

E1: parameter U1 (see UST3_Ex [57])
```
E2: parameter U2 (see UST3_Ex [57])
E3: parameter U3 (see UST3_Ex [57])
R_SET: parameter Y (see UST3_R_set [57])
R_ON_OFF: Measurement R or U

Requirements

<table>
<thead>
<tr>
<th>Development environment</th>
<th>required TC3 PLC library</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwinCAT from v3.1.4020.14</td>
<td>Tc2_MPBus from 3.3.5.0</td>
</tr>
</tbody>
</table>

4.3 Integration into TwinCAT

4.3.1 KL6771 with CX5120

This sample describes how a simple PLC program for MP-Bus can be written in TwinCAT and how it is linked with the hardware. The task is to control an individual damper drive and change it with a button.

Sample: https://infosys.beckhoff.com/content/1033/tcplclib_tc2_mpbus/Resources/zip/6222419595.zip

Hardware

Setting up the components

The following hardware is required:

- 1x CX5120 Embedded PC
- 1x digital 4-channel input terminal KL1104 (for the Open and Close functions)
- 1x KL6771 MP-Bus master terminal
- 1x KL9010 end terminal

Set up the hardware and the MP-Bus components as described in the associated documentation.

This sample assumes that the address of the damper drive is known.

Software

Creation of the PLC program

Create a new "TwinCAT XAE project" and a "Standard PLC project".

Add the library Tc2_MPBus under "References" in the PLC project.

Generate a Global Variable List with the name GVL_MPBus and create the following variables:

```plaintext
VAR_GLOBAL
  bOpen          AT %I* : BOOL;
  bClose         AT %I* : BOOL;
  arrKL6771_IN   AT %I* : ARRAY [0..11] OF BYTE;
  arrKL6771_OUT  AT %Q* : ARRAY [0..11] OF BYTE;
  stDataKL6771   : DataKL6771;
END_VAR
```

bOpen: Input variable for the Open button.
bClose: Input variable for the Close button.
arrKL6771_IN: Input variable for the MP-Bus terminal.
arrKL6771_OUT: Output variable for the MP-Bus terminal.
**Programming**

**stDataKL6771:** Required for communication with MP-Bus (see DataKL6771 [58]).

All MP-Bus function blocks must be called in the same task.

Therefore, create a MAIN program (CFC) in which the KL6771 [16] and MP_DamperLinearActuator [21] function blocks are called. Make sure to link arrKL6771_IN, arrKL6771_OUT and stDataKL6771 in the communication block.

The input SetPoint is set depending on the selected function. Link the global variables bOpen and bClose with an auxiliary variable.

Go to the task configuration and give the task a lower interval time.

Further conditions can be found in the description of the function block KL6771 [16].

**I/O configuration**

Select the CX as target system and initiate a search for its hardware. In the project instance within the PLC section, you can see that the input and output variables are assigned to the task.

Now link the global variables with the inputs and outputs of the Bus Terminals.

The linking of MP-Bus variables is described in detail below.

Right-click the array arrKL6771_IN and select "Change Link".
Under "I/O Configuration" select the terminal, select "All Types" and "Continuous", then select "ParameterStatus", "InputData1" to "InputData10" with the left mouse button and the >SHIFT< key. Then click "OK".

You can now check the connection. To do this, go to the KL6771 and open it. All terminal data should now show a small arrow. If that is the case, then proceed in exactly the same way with the outputs.
5 Appendix

5.1 Support and Service

Beckhoff and their partners around the world offer comprehensive support and service, making available fast and competent assistance with all questions related to Beckhoff products and system solutions.

Beckhoff’s branch offices and representatives

Please contact your Beckhoff branch office or representative for local support and service on Beckhoff products!

The addresses of Beckhoff’s branch offices and representatives round the world can be found on her internet pages: https://www.beckhoff.com

You will also find further documentation for Beckhoff components there.

Beckhoff Support

Support offers you comprehensive technical assistance, helping you not only with the application of individual Beckhoff products, but also with other, wide-ranging services:

- support
- design, programming and commissioning of complex automation systems
- and extensive training program for Beckhoff system components

Hotline: +49 5246 963 157
Fax: +49 5246 963 9157
e-mail: support@beckhoff.com

Beckhoff Service

The Beckhoff Service Center supports you in all matters of after-sales service:

- on-site service
- repair service
- spare parts service
- hotline service

Hotline: +49 5246 963 460
Fax: +49 5246 963 479
e-mail: service@beckhoff.com

Beckhoff Headquarters

Beckhoff Automation GmbH & Co. KG
Huelshorstweg 20
33415 Verl
Germany

Phone: +49 5246 963 0
Fax: +49 5246 963 198
e-mail: info@beckhoff.com
web: https://www.beckhoff.com
More Information:
www.beckhoff.com/te1000