



[www.asix.com.pl](http://www.asix.com.pl)

**Communication Drivers**  
Parameterization of Drivers  
Included in Asix Package

*Doc. No ENP8002*  
*Version: 2016-09-28*

**ASKOM**<sup>®</sup> and **asix**<sup>®</sup> are registered trademarks of ASKOM Spółka z o.o., Gliwice. Other brand names, trademarks, and registered trademarks are the property of their respective holders.

All rights reserved including the right of reproduction in whole or in part in any form. No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage and retrieval system, without prior written permission from the ASKOM.

ASKOM sp. z o. o. shall not be liable for any damages arising out of the use of information included in the publication content.

Copyright © 2016, ASKOM Sp. z o. o., Gliwice



ASKOM Sp. z o. o., ul. Józefa Sowińskiego 13, 44-121 Gliwice,  
tel. +48 32 3018100, fax +48 32 3018101,

<http://www.askom.com.pl>, e-mail: office@askom.com.pl

# Table of Contents

1 Drivers and Transmission Protocols .....	5
1.1 Driver of ADAM Protocol.....	10
1.2 Aggregate Driver .....	16
1.3. AirPointer - Driver for Data Exchange Between ASIX and Air Monitoring Station AirPointer Developed by Recordum Messtechnik GmbH, Austria.....	20
1.4 AK - Driver of AK Protocol for Emerson MLT2 Analyzers .....	25
1.5 AM_SA85 - Driver of MODBUS PLUS Protocol for AM-SA85-000 Card .....	31
1.6 AREVA - Driver for Communication with MiCOM Devices .....	35
1.7 AS511 - Driver of AS511 Protocol for SIMATIC S5 PLCs .....	40
1.8 AS512 - Driver of AS512 Protocol for SIMATIC S5 PLCs .....	44
1.9 AS512S7 - Driver of AS512 Protocol for SIMATIC S7 PLCs.....	47
1.10 BACnetIP - Driver for BACnet/IP Protocol Devices.....	50
1.11 BAZA - Driver for Access to Database .....	59
1.12 BUFOR - Driver of General Purpose .....	65
1.13 Calec - Driver of CALECMCP Device Protocol .....	68
1.14 CAN_AC_PCI - Driver of CANBUS Protocol for CAN ACx PCI Card.....	72
1.15 CANOPEN - Driver of CANBUS Protocol for PCI 712 NT Card .....	78
1.16 CipAB - Driver for Communication with Logix5000 Series Controllers by Allen- Bradley .....	83
1.17 COMLI - Driver of COMLI Protocol.....	90
1.18 CPIII - Driver to Communicate with Control Panels CP-III/E Used to Control Compressors of MYCOM (MAYEKAWA) .....	95
1.19 CZAZ - Driver for Communication with CZAZ-U and CZAZ-UM .....	103
1.20 DataPAF - Driver of DataPAF Energy Counter Protocol.....	109
1.21 DDE Driver .....	119
1.22 Diva - Driver for Data Exchange with DIVA Industrial Cameras by VDG Security B.V. ....	124
1.23 DIms - Driver of Electric Energy Counter of ZxD/ZxG/ZxQ Type of Landys&Gyr Company.....	129
1.24 DImsTcpi - Driver for Landys&Gyr Electricity Meters.....	140
1.25 DMS285 - Driver of Protocol for DURAG DMS 285 Analyzers .....	151
1.26 DMS500 - Driver of Protocol for DURAG DMS 500 Analyzers .....	166
1.27 DNP3 - Driver of Distributed Network Protocol Version 3 for Electrical Power Engineering Control and Supervision Systems .....	173
1.28 DP - Driver of PROFIBUS DP Network Protocol for PROFOboard Board.....	187
1.29 DP5412 - Driver of PROFIBUS DP Protocol for CP5412 Card.....	194
1.30 DSC - Driver of DSC PLC Protocol.....	199
1.31 DXF351 - Driver of Compant XF351 Device Protocol .....	202

1.32 E2tangoTcpiip - Protocol Driver for e2TANGO Controllers by Elektrometal Energetyka .....	206
1.33 Ecl - Driver for Communication with ECL Comfort 210/310 Controllers Developed by Danfoss .....	213
1.34 EcoMUZ - Driver of ecoMUZ Protocol.....	220
1.35 EcoMuz2 - Driver of ecoMUZ-2 Protocol of JM Tronik.....	224
1.36 EQABP - Driver for Electricity Meters EQABP of POZYTON .....	236
1.37 Esser - Honeywell Esser 8008 Fire Protection Unit Driver.....	240
1.38 FESTO - Driver of Diagnostic Interface for FESTO PLCs .....	245
1.39 FILE2ASIX - Driver for Data Import from Files .....	248
1.40 FP1001 - Driver of METRONIC Measurer Protocol.....	252
1.41 GFCAN - Driver of CANBUS Protocol for CanCard.....	260
1.42 Global - Exchange of Data Between the Asix System Application and a Swap File .....	265
1.43 IEC61850 - Driver for IEC61850 Protocol .....	269
1.44 K3N - Driver of OMRON K3N Meters Family Protocol .....	279
1.45 K-BUS - Driver of Protocol for VIESSMANN Decamatic Boiler PLCs .....	285
1.46 Lb480 - Driver for Communication with Concentrator LB-480 by LAB_EL Elektronika Laboratoryjna through Ethernet .....	290
1.47 LG - Driver of Dedicated Protocol of LG Master-K and Glofa GM PLCs .....	297
1.48 Logo - DRIVER of Logo OBA5 controllers from SIEMENS.....	303
1.49 LUMBUS - Driver for LUMEL Meters.....	307
1.50 LZQM - Driver of LZQM Electricity Meters .....	316
1.51 M200 - Spirax Sarco M200 Flow Computer Driver .....	322
1.52 MACMAT - Driver of GAZ_MODEM Protocol for MACMAT Station .....	326
1.53 Max1000 - Drajwer Protokołu Systemu MAX-1000 Firmy ULTRAK.....	340
1.54 M-BUS - Driver of M-BUS Protocol .....	344
1.55 MEC - Driver of MEC07 and MEC08 Heat Meter Protocol .....	351
1.56 MegaMuz - Driver for Communication with Microprocessor Protection Devices of MegaMuz type by JM-Tronik, Warsaw .....	357
1.57 MegaMuz2 - Driver for Communication with Microprocessor Protection Devices of MegaMuz2 Type by JM-Tronik, Warsaw .....	368
1.58 MegaMuz_TCPIP - Driver for Communication with Microprocessor Protection Devices of MegaMuz Type by JM-Tronik through Ethernet .....	381
1.59 MegaMuz2_TCPIP - Driver for Communication with Microprocessor Protection Devices of MegaMuz2 Type by JM-Tronik through Ethernet .....	392
1.60 MELSECA - Driver of MITSUBISHI MELSEC-A PLC Protocol.....	404
1.61 MEVAS - Driver of MEVAS Analyzers .....	408
1.62 MicroSmart - Driver for Data Exchange with MicroSmart Controllers from IDEC421	
1.63 MODBUS - Driver of MODBUS/RTU Protocol for MASTER Mode .....	425

## Communication Drivers

1.64 MODBUS_TCPIP - Driver of MODBUS_TCP/IP Protocol for OPEN MODBUS/TCP Mode .....	435
1.65 MODBUSSLV - Driver of MODBUS/RTU Protocol for SLAVE Mode .....	450
1.66 MPI - Driver of MPI Protocol for SIMATIC S7 PLCs .....	455
1.67 MPS - Driver of MPS Protocol for Power Network Parameter Meters .....	458
1.68 MSP1X - Driver of Protocol for MSP-1x ELMONTEX PLCs .....	462
1.69 Muel - MUEL System Driver .....	466
1.70 MultiMuz - Driver for MultiMUZ Microprocessor-Based Security Devices from JM-Tronik .....	470
1.71 MultiMuz_tcpip - Driver for MultiMUZ Microprocessor-Based Security Devices from JM-Tronik .....	480
1.72 MultiMuz3 - Driver for Communication with Microprocessor Protection Devices of MultiMUZ3 Type by JM-Tronik .....	488
1.73 MultiMuz3_tcpip - Driver for MultiMUZ Microprocessor-Based Security Devices from JM-Tronik.....	498
1.74 MUPASZ - Driver of MUPASZ Device Protocol .....	509
1.75 Mupasz710_RS - Driver of Microprocessor Protecting Devices of MUPASZ 710 Type .....	513
1.76 MupaszRtu - Driver for Data Exchange with Microprocessor Devices of Mupasz2001G, Mupasz07 and Mupasz Compact G01 Type Developed by ITR, Warsaw .....	525
1.77 MupaszRtu_TCPIP - Driver of Microprocessor Protecting Devices of MUPASZ 710 Type .....	534
1.78 Mus04 - Driver for Data Exchange with MUS-04 Control Devices from ELEKTROMETAL S.A. ....	546
1.79 MUZ - Driver of Protocol for MUZ Devices.....	551
1.80 NCP - Driver for MN-Series Drivers from Invensys .....	561
1.81 NetLink - Driver of MPI/Profibus Protocol for SIMATIC S7 by using NetLink Lite SYSTEME HELMHOLZ Module .....	565
1.82 NetLinkPro - Driver for Communication with the S7 Controllers .....	569
1.83 NETWORK Driver.....	573
1.84 NONE Driver .....	575
1.85 NordicRF - driver of Nordic ID RF 601 Terminal Made by NordicID Company ...	578
1.86 OMRON - Driver of HOSTLINK Protocol.....	582
1.87 OmronTcpi - Driver for Data Exchange with Omron Controllers Supporting the FINS/UDP and FINS/TCP Protocols .....	589
1.88 OPC Driver .....	595
1.89 PA5 - PA5 Flow Counter Protocol Driver.....	601
1.90 Pmc4000 - POLON 4800 Fire Protection Station Driver According to Protocol PMC-4000.....	605
1.91 PPI - Driver of PPI Protocol for SIMATIC S7 200 PLCs .....	610
1.92 Protherm300 - Driver of PROTHERM 300 DIFF PLC Protocol .....	617

1.93 PROTRONICPS - Driver of PROTRONICPS Regulator Protocol.....	621
1.94 S700 - Driver S700 of MAIHAK Analyzer Protocol .....	625
1.95 S7_TCPIP - Driver for Data Exchange with SIMATIC Controllers with Use of Ethernet.....	632
1.96 Sapis7 - Driver of Sapis7 Protocol for SIMATIC S7 PLCs .....	638
1.97 S-BUS - Driver of S-BUS Protocol for SAIA-Burgess Electronics PLCs .....	642
1.98 SbusTcpi - Driver of S-Bus Ethernet Protocol .....	650
1.99 Si400 - Driver of Protocol for Sintony Si 400 Alarm Central of SIEMENS .....	655
1.100 SINECH1 - Driver of Ethernet Network Protocol for SIMATIC S5 PLCs.....	662
1.101 SINECL2 - Driver of PROFIBUS Protocol for SIMATIC S5 PLCs .....	666
1.102 SNG - Driver to Exchange Data with SNG Devices .....	671
1.103 SNMP - Driver for Reading and Writing Object Values Using SNMPv1 and SNMPv2c Protocol.....	676
1.104 SNPX - Driver of SNPX Protocol for GE Fanuc PLCs.....	680
1.105 SPA - Driver of SPA Protocol .....	685
1.106 Srio - Driver of SRIO 500M Protocol.....	691
1.107 SRTP - Driver of SRTP Protocol.....	705
1.108 TALAS - Driver of TALAS Analyzer Protocol .....	710
1.109 TwinCAT - Driver of ADS Protocol for TwinCAT System .....	715
1.110 UniDriver.....	722
1.111 Vantage - Driver for Data Exchange with Weather Stations of Vantage Pro Family .....	723
1.112 Wago - Wago Controllers Protocol Driver .....	728
1.113 ZdarzenieZmienna Driver .....	735
1.114 ZxD400 - Driver of Protocol of Landys & Gyr ZxD400 Electric Energy Counters .....	740

# 1 Drivers and Transmission Protocols

The Asix system includes a set of drivers that handle the following types of data transfer with controllers of an industrial process.

Driver	Protocol
<b>ADAM</b>	- protocol for ADAM-4000 modules of ADVANTECH
<b>AGGREGATE</b>	- the driver allows definition of variables, values of which are generated as a result of calculations performed on other variables of the Asix system (source variables)
<b>AirPointer</b>	- driver for data exchange between ASIX and air monitoring station AirPointer developed by Recordum Messtechnik GmbH, Austria
<b>AK</b>	- the AK protocol allows data exchange between Asix system computers and Emerson MLT2 analyzers
<b>AM_SA85</b>	- protocol for communication with the Modbus Plus network of Schneider Automation
<b>AREVA</b>	- allows to exchange data between Asix and digital protection devices MiCOM of AREVA; the list of serviced devices includes MiCOM P127 and MiCOM P34x series
<b>AS511</b>	- protocol using programmer interface of SIMATIC PLCs of SIEMENS
<b>AS512</b>	- protocol of CP524/525 communication processors for SIMATIC PLCs of SIEMENS
<b>AS512S7</b>	- protocol of CP340 communication processors for SIMATIC S7 PLCs of SIEMENS
<b>BACnetIP</b>	- protocol of devices with BACnet/ IP protocol interface
<b>BAZA</b>	- the driver enables data import from databases to the Asix system
<b>BUFOR</b>	- general purpose protocol for information exchange with user programs by means of a shared memory
<b>Calec</b>	- Calec MCP driver retrieving the current values of variables from CALEC MCP devices of Aquametro according to the protocol described in the document "MCP Datenauslesung mit dem lowlevel Protokoll"
<b>CAN_AC_PCI</b>	- protocol for data exchange between SELECONTROL MAS PLCs of Selectron Lyss AG and Asix system computers
<b>CANOPEN</b>	- CANOPEN network protocol of SELECTRON MAS PLCs of Selectron Lyss AG
<b>CipAB</b>	- protocol of Logix5000 series controllers by Allen-Bradley using EtherNet/IP protocol in Explicit Messaging mode
<b>COMLI</b>	- protocol (COMunication Link) for communication with ABB SattCon, AC 800C, AC 800M, AC 250 PLCs; data are transferred via RS-232 or RS-485 serial interfaces
<b>CPIII</b>	- Driver to Communicate with Control Panels CP-III/E Used to Control Compressors of MYCOM (MAYEKAWA)
<b>CZAZ</b>	- allows to exchange data between Asix system and digital protection devices CZAZ-U and CZAZ-UM of ZEG-Energetyka
<b>DATAPAF</b>	- protocol for connection with DataPAF energy counters
<b>DDE</b>	- driver defining a channel of the ASMEN module referring to variables shared by a DDE server
<b>Dlms</b>	- driver is used for data exchange between the Asix system and electric energy counters of ZxD/ZxG/ZxQ type of Landys & Gyr company, using the DLMS protocol
<b>DlmsTcpiip</b>	- driver for Landys&Gyr Electricity Meters through Ethernet
<b>DMS285</b>	- protocol for emission meter D-MS285 computers
<b>DMS500</b>	- protocol for emission meter D-MS500 computers (previous name - DURAG)

<b>DNP3</b>	- driver of Distributed Network Protocol Version 3 for Electrical Power Engineering Control and Supervision Systems
<b>DP</b>	- protocol for devices compatible with PROFIBUS DP by using a PROFIBoard card
<b>DP5412</b>	- protocol for devices compatible with PROFIBUS DP by using Siemens cards
<b>DSC</b>	- protocol for data exchange between Asix system computers and DSC 2000 controllers
<b>DXF351</b>	- protocol for communication with Compant DXF351 devices of Endress+Hauser
<b>E2tangoTcpiip</b>	- driver is used for data echange between the Asix system and electric energy counters of Landys & Gyr company, using the DLMS protocol. The communication is performed through Ethernet
<b>Ecl</b>	- protocol for for ECL Comfort 210/310 Controllers Developed by Danfoss
<b>EcoMUZ</b>	- protocol for data exchange between the Asix system and Microprocessor Protecting ecoMUZ Devices made by JM Tronik
<b>EcoMuz2</b>	- driver is used for data exchange between the Asix system and Microprocessor Protecting ecoMuz-2 devices made by JM Tronik. The transmission is performed through serial links by means of serial interfaces in the RS-485 standard.
<b>EQABP</b>	- protocol for data exchange between the Asix system and electricity meters EQABP, produced by Zakład Elektronicznych Urządzeń Pomiarowych POZYTON sp. z o.o. in Częstochowa
<b>Esser</b>	- protocol for data exchange between the Asix system and the Honeywell Esser 8008 fire protection unit control system
<b>FESTO</b>	- protocol using a diagnostic interface for FESTO PLCs
<b>FILE2ASIX</b>	- the driver enables data import from text files to the Asix system
<b>FP1001</b>	- protocol for water and steam flow monitors of METRONIC Kraków
<b>GFCAN</b>	- protocol of CAN network with use of communication card of Garz & Fricke Industrieautomation GmbH
<b>GLOBAL</b>	- protocol for data exchange between the Asix system application and a so-called swap file, which is a container for the driver's current variable parameters (name, status, value, timestamp)
<b>IEC61850</b>	- IEC61850 protocol
<b>K3N</b>	- protocol for data exchange between K3N meters family of OMRON and Asix system computers
<b>K-BUS</b>	- protocol K-BUS used for data exchange between VIESSMANN Dekamatic boilers controllers connected to a Dekatel-G (or Vitocom 200) concentrator and Asix system computers
<b>LG</b>	- protocol of LG Master-K and Glofa GM PLCs
<b>Logo</b>	- driver of Logo OBA5 controllers from SIEMENS
<b>LUMBUS</b>	- protocol for data exchange between RG72 controllers manufactured by Lubuskie Zakłady Aparatów Elektrycznych (Electrical Measuring Instrument Works) "LUMEL" in Zielona Góra and Asix system computers
<b>LZQM</b>	- protocol for data exchange with LZQM electricity meters, produced by POZYTON Electronic Measuring Instruments Facility in Częstochowa
<b>M200</b>	- for data exchange between the Asix system and the M210G flow computer from Spirax Sarco
<b>MACMAT</b>	- GAZ-MODEM protocol used for communication with MACMAT stations
<b>MAX1000</b>	- communication based on the so-called "Network protocol" with the MAX 1000 camera management system by ULTRAK
<b>M-BUS</b>	- subset of standard protocol for data reading from measuring devices used by MULTICAL heat meters of KAMSTRUP A/S
<b>MEC</b>	- protocol of data exchange between Asix system and MEC07 and MEC08 heat meters manufactured by Instytut Techniki

	<p>Cieplnej (Institute of Thermal Technology) in Lodz. Data are transferred with use of a standard RS-232 interface.</p>
<b>MegaMuz</b>	<p>- protocol for microprocessor protection devices of MegaMuz type developed by JM-Tronik, Warsaw</p>
<b>MegaMuz_TCPIP</b>	<p>- protocol for microprocessor protection devices of MegaMuz type developed by JM-Tronik, Warsaw. Communication is carried out via Ethernet.</p>
<b>MELSECA</b>	<p>- protocol of A1SJ71C24-R2 communication processor for MELSEC-A PLCs</p>
<b>MEVAS</b>	<p>- protocol for data exchange between the MEVAS emission meter computer produced by Lubuskie Zakłady Aparatów Elektrycznych (Electrical Measuring Instrument Works ) "LUMEL" in Zielona Góra and Asix system computers</p>
<b>MicroSmart</b>	<p>- driver for data exchange with MicroSmart controllers from IDEC</p>
<b>MODBUS</b>	<p>- subset of standard communication protocol used by AEG Modicon GE Fanuc PLCs</p>
<b>MODBUS_TCPIP</b>	<p>- protocol of data exchange between Asix system and computers/devices by means of the MODBUS protocol on the basis of Ethernet with the TCP/IP protocol</p>
<b>MODBUSSLV</b>	<p>- MODBUS protocol, in which Asix operates as SLAVE</p>
<b>MPI</b>	<p>- protocol of MPI interface of SIMATIC S7 PLCs of SIEMENS; a serial interface</p>
<b>MPS</b>	<p>- serial interface protocol for MPS measuring gauges of a power network from OBR Metrologii Elektrycznej in Zielona Góra</p>
<b>MSP1X</b>	<p>- protocol MSP1X used for data exchange between MSP1X PLCs of ELMONTEX and Asix system computers</p>
<b>CtMuel</b>	<p>- protocol used to exchange data between the Asix system and a MUEL system maintenance computer</p>
<b>MultiMuz</b>	<p>- used to exchange data between the asix system and te MultiMUZ microprocessor-based security devices manufactured by Warsaw-based JM-Tronik</p>
<b>MultiMuz TCPIP</b>	<p>- communication is executed in Ethernet with the use of TCP or UDP protocol to exchange data between the Asix system and te MultiMUZ microprocessor-based security devices manufactured by Warsaw-based JM-Tronik.</p>
<b>MultiMuz3</b>	<p>- protocol for MultiMUZ3 microprocessor-based security devices manufactured by Warsaw-based JM-Tronik</p>
<b>MultiMuz3 TCPIP</b>	<p>- communication is executed in MODBUS RTU mode on TCPIP with the use of Ethernet and port 10502 to exchange data between the Asix system and the MultiMUZ3 microprocessor-based security devices manufactured by Warsaw-based JM-Tronik.</p>
<b>MUPASZ</b>	<p>- hollow (virtual) channel protocol</p>
<b>Mupasz710_RS</b>	<p>- for microprocessor protecting devices of MUPASZ 710 type made by Instytut Tele- i Radiotechniczny in Warsaw. The communication is made with use of serial link and the MODBUS protocol.</p>
<b>MupaszRtu</b>	<p>- protocol for Mupasz2001G, Mupasz07 and Mupasz Compact G01 type devices by ITR, Warsaw</p>
<b>MupaszRtu_TCPIP</b>	<p>- driver is used for data exchange between Asix and the microprocessor protecting devices of MUPASZ 710 type. The communication is performed in MODBUS RTU mode via Ethernet link, using port no. 502.</p>
<b>Mus04</b>	<p>- allows data between the Asix system and the microprocessor-based control devices MUS-04 manufactured by ELEKTROMETAL S.A. from Cieszyn to be exchanged</p>
<b>MUZ</b>	<p>- protocol for data exchange between Microprocessor Security Devices of MUZ-RO type;</p>
<b>NCP</b>	<p>- is used to exchange data between the Asix system and MN-series controllers from Invensys (former Satchwell)</p>

<b>NETWORK</b>	- protocol used by the NETSRV.EXE program is a special type of protocol. It does not provide a physical link to a controller, but only connection to any other computer that has such a link
<b>NetLink</b>	- is used for data exchange between Asix computers and SIMATIC S7 PLCs by using an MPI/Profibus bus and a NetLink Lite SYSTEME HELMHOLZ module
<b>NetLinkPro</b>	- to communicate with the S7 controllers via the NETLink PRO gateway
<b>NONE</b>	- NONE protocol enables: <ul style="list-style-type: none"> <li>• Asix application testing in simulation mode,</li> <li>• data exchange between Asix programs by means of process variables;</li> </ul>
<b>NordicRF</b>	- protocol used to exchange data with Nordic ID RF 601 barcode scanner
<b>OMRON</b>	- enables data exchange between OMRON PLCs and Asix system computers
<b>OmronTcpiip</b>	- protocol used for data exchange between Asix system and Omron controllers supporting the FINS/UDP and FINS/TCP protocols
<b>OPC</b>	- the driver defining a channel of ASMEN module retrieving the variables shared by an OPC server
<b>PA5</b>	- protocol to exchange data between the Asix system and the PA-5 transducers manufactured by Poznań-based POWOGAZ S.A. Water Meter Factory
<b>Pmc4000</b>	- is used for data exchange between the asix system and POLON 4800 fire protection station according to protocol PMC-4000
<b>PPI</b>	- protocol for SIEMENS S7-200 PLCs
<b>Protherm300</b>	- driver for data exchange between Protherm 300 DIFF PLCs of Process-Electronic GmbH and Asix system computers
<b>PROTRONICPS S700</b>	- PROTRONIC PS protocol of Hartmann & Braun
<b>S7_TCPIP</b>	- protocol S700 used for data exchange between Maihak 3700 gas analyzers and Asix system computers
<b>S7_TCPIP</b>	- is used for data exchange with SIMATIC S7-series controllers through the Ethernet connection with the use of a standard computer network card
<b>SAPIS7</b>	- protocol of SIMATIC S7 PLCs with use of the MPI interface or PROFIBUS communication processor (an implementation of S7 function)
<b>S-BUS</b>	- S-BUS protocol used for data exchange between PCD PLCs of SAIA Burgess Electronics and Asix system computers
<b>SbusTcpiip</b>	- driver for data exchange between family of PCD SAIA-Burgess PLCs and Asix system computers
<b>Si400</b>	- is used for data exchange between Asix computers and a Sintony Si 400 alarm central of SIEMENS
<b>SINECH1</b>	- protocol of CP1430 communication processors of SIMATIC S5 PLCs (Ethernet)
<b>SINECL2</b>	- protocol of CP5430 communication processors of SIMATIC S5 PLCs of SIEMENS
<b>SNG</b>	- protocol used for data exchange between the Asix system and SNG systems provided by the Warsaw based Synergia Tech company, via an Ethernet link
<b>SNMP</b>	- driver can read and write objects values using SNMPv1 and SNMPv2c protocol - manage the various elements of telecommunications networks, such as routers, switches, computers and telephone exchanges. The driver performs its functions by using the SNMP Management API
<b>SNPX</b>	- driver for data exchange between Asix system computers and GE Fanuc 90-30 PLCs as well as GE Fanuc 90 CMM and PCM modules
<b>SPA</b>	- protocol used for communication with devices connected to SPA bus of the ABB company
<b>Srio</b>	- protocol used for exchange of data with the SRIO 500M hub (ABB-produced), which provides communication between the

## Communication Drivers

	host system and the devices connected to the SPA bus, such as protection relays and signalling devices
<b>SRTP</b>	- driver used for data exchange between the Asix system and GE Fanuc Automation VersaMax Nano/Micro PPLCs using an IC200SET001 converter and WersaMax 90 PLCs using the IC693CMM321 communication module; via Ethernet with the TCP/IP protocol
<b>TALAS</b>	- protocol of TALAS emission computers
<b>TwinCAT</b>	- driver for data exchange between the Asix system and the TwinCAT system of Beckhoff Industrie Elektronik
<b>Vantage</b>	- protocol for readout of current data from weather stations of Vantage Pro family developed by Davis Instruments Corp., USA
<b>UniDriver</b>	
<b>Wago</b>	- protocol used to exchange data between the Asix system and Wago controllers.
<b>ZDARZENIE ZMIENNA</b>	- driver for generating process variables of WORD type (16-bit word) on the basis of actual values of alarm events in the Asix system
<b>ZxD400</b>	- driver of protocol of electric energy counters of ZxD400 type manufactured by Landys & Gyr

The package of available protocols will be systematically expanded. The ASKOM company is ready to develop on customer request any transmission protocol according to the rules defined in the price list of the Asix system.

## 1.1 Driver of ADAM Protocol

### Driver Use

The ADAM driver is used for data exchange with ADAM-4000 series modules developed by Advantech. The transmission is performed with use of serial interfaces via standard serial ports of a computer (using the converter) or by using an additional card with an RS485 interface.

The only Asix system requirement is that the ADAM modules should be configured to the following data transfer mode:

- number of character bits 10 (1 start bit, 8 character bits, 1 stop bit),
- no parity check,
- checksum.

Parameterization of ADAM driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the ADAM driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name: logical name of the transmission channel*  
*Driver: ADAM*

**Adam** tab:

**Channel parameters:**

*Network number of ADAM module* - network number of the ADAM module;  
*Identifier of ADAM module type* - identifier of the ADAM module type. At present the following types are implemented:

- 1 - ADAM-4011
- 2 - ADAM-4012
- 3 - ADAM-4013
- 4 - ADAM-4017
- 5 - ADAM-4018
- 6 - 8-channel pulse counter Mcom-1 (an equivalent of ADAM-4080D)
- 7 - ADAM-4050
- 8 - ADAM-4052
- 9 - ADAM-4060
- 10 - ADAM-4053
- 11 - ADAM-4080
- 12 - ADAM-4021

*Port* - serial port name;  
*Transmission speed in bauds* - transmission speed.

The *Transmission speed in bauds* parameter is an optional parameter. Its default value is 9600 (Bd).

#### EXAMPLE

An exemplary item which declares the use of transmission channel operating according to the ADAM protocol is given below:

*Channel / Name:* CHAN1

Channel / Driver: ADAM  
 Network number of ADAM module: 1  
 Identifier of ADAM module type: 5  
 Transmission speed in bauds: 9600 Bd

## Addressing the Process Variables

The syntax of symbolic address which is used for process variables belonging to the ADAM driver channel is as follows:

*VARIABLE\_TYPE* *variable\_index* [*.subchannel\_no*]

where:

- VARIABLE\_TYPE* - string identifying the variable name in the ADAM protocol;
- variable\_index* - variable index within a given type;
- subchannel\_no* - subchannel number for multichannel modules or a bit number for digital in/out modules.

The following symbols of process variable types are allowed:

- R - read only variable;
- W - write only variable;
- RW - read/write variable.

Depending on the ADAM module type, various ranges of *variable\_index* and *subchannel\_no* are allowed. Process variables implemented at present are given below:

Table. Types of Implemented Process Variables Serviced by ADAM Modules.

Symb. Address	Variable Type in Device	Type of Raw Variable	Device Type	Data Format
	<b>Variables Only for Reading</b>			
R1	Read analog Input	Float	ADAM-4011	Eng. units
R1	Read analog Input	Float	ADAM-4012	Eng. units
R1	Read analog Input	Float	ADAM-4013	Eng. units
R1	Current readback	Float	ADAM-4021	Eng. Units
R1.0	Read analog Input 0	Float	ADAM-4017	Eng. units
R1.1	Read analog Input 1	Float	ADAM-4017	Eng. units
R1.2	Read analog Input 2	Float	ADAM-4017	Eng. units
R1.3	Read analog Input 3	Float	ADAM-4017	Eng. units
R1.4	Read analog Input 4	Float	ADAM-4017	Eng. units
R1.5	Read analog Input 5	Float	ADAM-4017	Eng. units
R1.6	Read analog Input 6	Float	ADAM-4017	Eng. units
R1.7	Read analog Input 7	Float	ADAM-4017	Eng. units

R1.0	Read analog Input 0	Float	ADAM-4018	Eng. units
R1.1	Read analog Input 1	Float	ADAM-4018	Eng. units
R1.2	Read analog Input 2	Float	ADAM-4018	Eng. units
R1.3	Read analog Input 3	Float	ADAM-4018	Eng. units
R1.4	Read analog Input 4	Float	ADAM-4018	Eng. units
R1.5	Read analog Input 5	Float	ADAM-4018	Eng. units
R1.6	Read analog Input 6	Float	ADAM-4018	Eng. units
R1.7	Read analog Input 7	Float	ADAM-4018	Eng. units
R1.0	Read counter/frequency value channel 0	Dword	MCom-1	Hex
R1.1	Read counter/frequency value channel 1	Dword	MCom-1	Hex
R1.2	Read counter/frequency value channel 2	Dword	MCom-1	Hex
R1.3	Read counter/frequency value channel 3	Dword	MCom-1	Hex
R1.4	Read counter/frequency value channel 4	Dword	MCom-1	Hex
R1.5	Read counter/frequency value channel 5	Dword	MCom-1	Hex
R1.6	Read counter/frequency value channel 6	Dword	MCom-1	Hex
R1.7	Read counter/frequency value channel 7	Dword	MCom-1	Hex
R2.0	Read timer interval value channel 0	Dword	MCom-1	0.1 sec incr.
R2.1	Read timer interval value channel 1	Dword	MCom-1	0.1 sec incr.
R2.2	Read timer interval value channel 2	Dword	MCom-1	0.1 sec incr.
R2.3	Read timer interval value channel 3	Dword	MCom-1	0.1 sec incr.
R2.4	Read timer interval value channel 4	Dword	MCom-1	0.1 sec incr.
R2.5	Read timer interval value channel 5	Dword	MCom-1	0.1 sec incr.
R2.6	Read timer interval value channel 6	Dword	MCom-1	0.1 sec incr.
R2.7	Read timer interval value channel 7	Dword	MCom-1	0.1 sec incr.

*Table. Types of Implemented Process Variables Serviced by ADAM Modules (continuation).*

<b>Symb. Address</b>	<b>Variable Type in Device</b>	<b>Type of Raw Variable</b>	<b>Device Type</b>	<b>Data Format</b>
R1.0	Read digital Input 0	Word	ADAM-4050	0/1
R1.1	Read digital Input 1	Word	ADAM-4050	0/1
R1.2	Read digital Input 2	Word	ADAM-4050	0/1
R1.3	Read digital Input 3	Word	ADAM-4050	0/1
R1.4	Read digital Input 4	Word	ADAM-4050	0/1
R1.5	Read digital Input 5	Word	ADAM-4050	0/1
R1.6	Read digital Input 6	Word	ADAM-4050	0/1
RW1	Analog Data Out/Last value readback	Float	ADAM-4021	Eng. Units
RW1.0	Read/Write digital Output 0	Word	ADAM-4050	0/1
RW1.1	Read/Write digital Output 1	Word	ADAM-4050	0/1
RW1.2	Read/Write digital Output 2	Word	ADAM-4050	0/1
RW1.3	Read/Write digital Output 3	Word	ADAM-4050	0/1
RW1.4	Read/Write digital Output 4	Word	ADAM-4050	0/1
RW1.5	Read/Write digital Output 5	Word	ADAM-4050	0/1
RW1.6	Read/Write digital Output 6	Word	ADAM-4050	0/1
RW1.7	Read/Write digital Output 7	Word	ADAM-4050	0/1
R1.0	Read digital Input 0	Word	ADAM-4052	0/1
R1.1	Read digital Input 1	Word	ADAM-4052	0/1
R1.2	Read digital Input 2	Word	ADAM-4052	0/1
R1.3	Read digital Input 3	Word	ADAM-4052	0/1
R1.4	Read digital Input 4	Word	ADAM-4052	0/1
R1.5	Read digital Input 5	Word	ADAM-4052	0/1
R1.6	Read digital Input 6	Word	ADAM-4052	0/1
R1.7	Read digital Input 7	Word	ADAM-4052	0/1
RW1.0	Read/Write digital Output 0	Word	ADAM-4060	0/1
RW1.1	Read/Write digital Output 1	Word	ADAM-4060	0/1
RW1.2	Read/Write digital Output 2	Word	ADAM-4060	0/1
RW1.3	Read/Write digital Output 3	Word	ADAM-4060	0/1
R1.0	Read counter/frequency value - channel 0	Dword	ADAM-4080	Hex
R1.1	Read counter/frequency value - channel 1	Dword	ADAM-4080	Hex
RW1.0	Read/Write initial counter - channel 0	Dword	ADAM-4080	Hex
RW1.1	Read/Write initial counter - channel 1	Dword	ADAM-4080	Hex
W1.0	Clear counter - channel 0	Dword	ADAM-4080	Hex
W1.1	Clear counter - channel 1	Dword	ADAM-4080	Hex

Table. Types of Implemented Process Variables Serviced by ADAM Modules (continuation).

Symb. Address	Variable Type in Device	Type of Raw Variable	Device Type	Data Format
R1.0	Read digital Input 0	Word	ADAM-4053	0/1
R1.1	Read digital Input 1	Word	ADAM-4053	0/1
R1.2	Read digital Input 2	Word	ADAM-4053	0/1
R1.3	Read digital Input 3	Word	ADAM-4053	0/1
R1.4	Read digital Input 4	Word	ADAM-4053	0/1
R1.5	Read digital Input 5	Word	ADAM-4053	0/1
R1.6	Read digital Input 6	Word	ADAM-4053	0/1
R1.7	Read digital Input 7	Word	ADAM-4053	0/1
R1.8	Read digital Input 8	Word	ADAM-4053	0/1
R1.9	Read digital Input 9	Word	ADAM-4053	0/1
R1.10	Read digital Input 10	Word	ADAM-4053	0/1
R1.11	Read digital Input 11	Word	ADAM-4053	0/1
R1.12	Read digital Input 12	Word	ADAM-4053	0/1
R1.13	Read digital Input 13	Word	ADAM-4053	0/1
R1.14	Read digital Input 14	Word	ADAM-4053	0/1
R1.15	Read digital Input 15	Word	ADAM-4053	0/1

The ADAM driver is installed as a DLL automatically.

## Driver Configuration

The driver configuration is defined in the *Current Data* module, in the channel operating according to ADAM driver.

### **Default length of reply**

Meaning - using the default answer length;  
 YES - waiting for the maximal possible answer length or character timeout;  
 NO - it is used if the answer length is not known in order not to wait for a timeout by the answer; otherwise the default answer length is used and a character timeout is waited.

Default value - NO.

### **Checksum**

Meaning - using the checksum in transfers PC <--> ADAM.  
 Default value - YES.

### **Read timeout**

Meaning - timeout for answer as a multiple of 100 milliseconds.  
 Default value - 15, i.e. 1500 ms.

### **Character timeout**

Meaning - character timeout as a multiple of 10 ms.  
 Default value - 5, i.e. 50 ms.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

computer name - *list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).*

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

computer name - *list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).*

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

number - *time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.*

The default value - by default, no time for data validity is set.

## 1.2 Aggregate Driver

### Driver Use

The Aggregate driver allows definition of variables the values of which are generated as a result of calculations performed on other variables of the Asix system (source variables). Archive values of source variables are used for an aggregate calculation. Usage of archive values allows to prevent any discontinuities in case of the Asix system restart.

Parameterization of Aggregate driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the Aggregate driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name: logical name of the transmission channel*

*Driver: Aggregate*

### Addressing the Process Variables

The address part of variable declaration for the Aggregate driver takes the following form:

*aggregate\_name aggregate\_parameters*

where:

*aggregate\_name* - name of the aggregate;

*aggregate\_parameters* - aggregate parameters, delimited with white space.

The driver may realize following aggregates:

**Table 4. Types of Aggregates Executed by the Driver Aggregate.**

<b>Aggregate Name</b>	<b>Way of Calculations</b>
Average (Average)	As a result, the weighted average of the source variable in the calculation period is obtained.
Max (max)	As a result, the maximum value of the source variable in the calculation period is obtained
Min (min)	As a result, the minimum value of the source variable in the calculation period is obtained.

**Parameters of above aggregates take the following form:**

***Variable\_name:Archive\_Type Period Threshold [A]***  
***[L[lower\_limit]:[upper\_limit]]***

where:

- Variable\_name* - name of the source variable which is connected with the aggregate;
- Archive\_type* - one letter code determining the type of the archive in which source variable values are saved;
- Period* - calculation period of the aggregate;
- Threshold* - minimum number of correct measures, in percentages, needed for aggregate calculations;
- A* - parameter determines whether calculation performing time should be adjusted to the calculation period;
- Lower\_limit* - lower limit value; if the source variable value is lower than the value of *lower\_limit*, then the *lower\_limit* value is used instead; the parameter may be used up to version 1.01.000 of the driver;
- Upper\_limit* - upper limit value; if the source variable value is higher than the value of *upper\_limit*, the then *upper\_limit* value is used instead; the parameter may be used up to version 1.01.000 of the driver;

*Period* determines the calculation period. The calculation period is given in the same manner as the time interval specification for ASPAD, ie. in the form of numbers and units:

*<number><unit> [<number><unit> [...]]*

where:

- <number>* - number of given subsequently time units;
- <unit>* - determines the time unit which may be:
  - s - second,
  - m - minute,
  - g lub h - hour,
  - d - day (24 hours) .

In case when unit is missing, the minute is taken as a default unit of the calculation period.

The result of aggregate calculations is said to be good if the interest rate of correctly read samples (given in percentages) is equal to *Threshold*. Default value of *Period* is 5 minutes, and of *Threshold* - 80 percentages. For the *threshold* correct measurement calculation the source variable valid time is taken into account, according to the parameterization of archiving this variable (sampling period). It means, *threshold* is calculated as a ratio of the sum of all correctness times of

measurements and calculation period length. In case of an average, the calculation result is a weighted average in relation to the measurement correctness time. Values of variables, for which time stamp is greater or equal to beginning of the calculation period and lower than the end of it, are taken into consideration for calculations of aggregates. The aggregate calculation occurs after the end of the calculation period.

The last, optional parameter *A* determines the time instance at which an aggregate will be calculated. If the parameter is omitted, the aggregate will be calculated after each reading of the source variable (in stepwise manner). If *A* is the last parameter, then the aggregate calculation time is adjusted to a multiplicity of the aggregate calculation period. The aggregate calculation result type is adjusted to the type of conversion function given in the variable declaration.

### EXAMPLE

An exemplary variable declaration:

```
Temp_sr, Temp-średnia, SREDNIA Temperatura:B 10 70 A, Srednie, 1, 1,
NOTHING_FP
```

The variable `Temperatura_sr` declared above is an average value of the variable `Temperatura`. The period, over which the variable was averaged, is 10 minutes and in order to obtain a correct value of the average, at least 70 percent of correct measurements are needed. Archive values, placed in the B archive, of the variable `Temperatura` are used for the average creation. The aggregate calculation instant will be adjusted to the calculation period multiplicity, i.e. calculations will be performed at 00:00:00, 00:10:00, 00:20:00 and so on.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

**Purpose** - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

**Option value:**

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

**The default value** - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

**Purpose** - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

## Communication Drivers

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.3. AirPointer - Driver for Data Exchange Between ASIX and Air Monitoring Station AirPointer Developed by Recordum Messtechnik GmbH, Austria

### Driver Use

The AirPointer is used for data exchange between ASIX and air monitoring station AirPointer developed by Recordum Messtechnik GmbH, Austria. The transmission is performed with use of HTTP protocol using info.php and download.php websites of monitoring station web server.

The driver allows to read current and historical 1-, 5- and 30-minute averages of each of parameters measured by the monitoring station. The current average value is transmitted by the driver as the current value of process variable.

Parameterization of AirPointer driver is performed with the use of Architect module.

### Declaration of transmission channel

Declaration of the transmission channel using the AirPointer driver requires a channel with the following parameters to be added to the **Current data** module of Architect:

**Standard** tab:

**Name:** *logical name of the transmission channel*

**Driver:** AirPointer

**AirPointer** tab:

**Channel parameters:**

*ServerIP= IP\_address; Password=text; User=text; AlarmNr=number*

where:

<i>ServerIP</i>	- monitoring station IP address;
<i>Password</i>	- access password to info.php and download.php sites of the station;
<i>User</i>	- monitoring station user name,
<i>AlarmNr</i>	- alarm number when communication with the monitoring station is broken.

**EXAMPLE:**

*ServerIP =192.168.200.160; Password=password123; User=user; AlarmNr=100*

### Addressing the Process Variables

The syntax of symbolic address which is used for process variables belonging to the AirPointer driver channel is as follows:

*P<Idparam>.<avgNo>*

where:

## Communication Drivers

*avgNo* - number that identifies the average number. Acceptable values:

- 1 -1-minute average
- 2 -5-minute average
- 3 -30-minute average

*Idparam* - number that identifies the parameter registered by the station.  
The list of acceptable parameters is available by the site info.php.

All the variable are of the FLOAT type.

Exemplary variable declarations:

ZM_01_01, NO 1 min average,	P1.1	PTY, 1, 1, NOTHING_FP
ZM_01_02, NO 5 min average,	P1.2	PTY, 1, 1, NOTHING_FP
ZM_01_03, NO 30 min average,	P1.3	PTY, 1, 1, NOTHING_FP
ZM_06_01, SO2 1 min average,	P6.1	PTY, 1, 1, NOTHING_FP
ZM_06_02, SO2 5 min average,	P6.2	PTY, 1, 1, NOTHING_FP
ZM_06_03, SO2 30 min average,	P6.3	PTY, 1, 1, NOTHING_FP

## Driver Configuration

AirPointer driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **CTAIRPOINTER**:

- tworzenie pliku logu,
- tworzenie pliku logu danych historycznych,
- rozmiar pliku logu,
- log danych przesłanych ze stacji,
- horyzont pozyskiwania danych historycznych,
- timeout ponownego zapytania o dane historyczne,
- log danych historycznych przekazywanych do Aspada.

**Section name: CTAIRPOINTER**

**Option name: LOG\_FILE**

**Option value: log\_file\_name**

Meaning: The item allows to define a file which all the diagnostic messages of the driver will be written to. The item is used for diagnostic purpose.

Parameter:

*log\_file\_name*

Default value: By default, the log file is not created.

**Section name: CTAIRPOINTER**

**Option name: HISTORY\_LOG\_FILE**

**Option value: log\_file\_name**

Meaning: The item allows to define a file which all the messages about reading historical data from monitoring station will be written to.

Parameter:

*log\_file\_name*

Default value: By default, the log file is not created.

**Section name: CTAIRPOINTER**

**Option name: LOG\_FILE\_SIZE**

**Option value: number**

Meaning: The item allows to define the size of the log file defined with the use of LOG\_FILE / LOG\_OF\_HISTORICAL\_TELEGRAMS options.

Parameter:  
*number* - size of the log file in MB.

Default value: 10 MB.

**Section name: CTAIRPOINTER**

**Option name: LOG\_OF\_TELEGRAMS**

**Option value: YES/NO**

Meaning: The item allows to write the data sent from the monitoring station to the log file. The item should be used only while the Asix start-up.

Default value: NO.

**Section name: CTAIRPOINTER**

**Option name: LOG\_OF\_HISTORICAL\_TELEGRAMS**

**Option value: YES/NO**

Meaning: The item allows to write the data sent from the monitoring station to the log file (declared with the use of **HISTORY\_LOG\_FILE** item). The item should be used only while the Asix start-up.

Default value: NO.

**Section name: CTAIRPOINTER**

**Option name: HISTORY\_SCOPE**

**Option value: number**

Meaning: The position determines how far the driver will turn back in time in the monitoring station archives when reading historical data.

Parameter:  
*number* - time in days.

Default value: 20.

**Section name: CTAIRPOINTER**

**Option name: HISTORY\_ACCESS\_TIMEOUT**

**Option value: number**

Meaning: The position determines which value will be returned to Aspad as timeout of Pozycja określa, jaka wartość zostanie zwrócona do Aspada jako re-query timeout about the data currently unavailable.

Parameter:  
*number* - time in minutes.

Default value: 5.

**Section name: CTAIRPOINTER**

**Option name: ASPAD\_HISTORICAL\_BUFFER\_LOG**

**Option value: YES/NO**

Meaning: The item allows to write the data sent to Aspad to the log file (declared with the use of HISTORY\_LOG\_FILE item). The item should be used only while the Asix start-up.

Default value: NO.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

The default value

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.
- by default, no time for data validity is set.

## 1.4 AK - Driver of AK Protocol for Emerson MLT2 Analyzers

### Driver Use

The driver of the AK protocol allows data exchange between Asix system computers and Emerson MLT2 analyzers. Communication is realized over the RS-485 serial links.

The driver allows only to read the data available by the READ type command, except commands assigned to service purposes as well as commands with the CODE type attribute.

Parameterization of AK driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the AK driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* AK

**CtAK** tab:

**Channel parameters:**

*Port=number* [*;Baudrate=number*] [*;RecvTimeout*] [*;CharTimeout*]

where:

<i>Port</i>	- number of a serial port COM;
<i>Baudrate</i>	- transmission speed between a computer and a device; the following speeds are available: 300, 600,1200,2400, 4800, 9600, 19200 Bd; by default - 19200 Bd;
<i>RecvTimeout</i>	- timeout (w milliseconds) between sending the last character of a query and receiving the first character of a response; by default - 200 milliseconds;
<i>CharTimeout</i>	- timeout (w milliseconds) between the successive response characters; by default - 30 milliseconds;

**NOTICE** The parameters passed in the channel declaration have to be compatible with the parameters set for communication ports of PLCs handled by this channel.

**EXAMPLE**

An exemplary declaration of the channel with standard timeouts on COM2:

Channel / Name: CHANNEL  
 Driver: CtAK  
 Channel Parameters: Port=2

## Addressing Process Variables

The syntax of symbolic address which is used for the variables belonging to the AK driver channel is presented below:

*<Type>.<AnalNo>.<ChanNo>[.<RangeNo>].<Index>*

where:

- Type* - variable type - a name of the command (CODE) used by the protocol for reading particular categories of variables from the analyzer are used as *Type*;
- AnalNo* - analyzer number;
- ChanNo* - channel analyzer number;
- RangeNo* - *Range* number of a given channel (if it is used when addressing the variable - see: statement of commands);
- Index* - specific interpretation of *Index* for the *Type* parameter is as follows:
  - a/ number of an element in *Range* (if *Range* is used when addressing the variable - see: statement of commands),
  - b/ number of elements in the data array returned by the command (if the command returns a data array). In particular the array dimension can equal 1.

```
/* concentration: analyzer no. 1, channel; 2, index 1
JJ_01, concentration,           AIKO.1.2.1, CHANNEL, 1, 1, NOTHING_FP
/* pressure: analyzer no. 1, channel 3, index 1
JJ_02, pressure,               ADRU.1.3.1, CHANNEL, 1, 1, NOTHING_FP
```

### Input/Output States and Calibration State

The driver allows to read Input/Output states of DIO cards by the service command "ASVC Kn S615 b". The following variable address should be used for reading Input/Output states:

*ASVC.<AnalNo>.<ChanNo><DIONo>.<Index>*

where:

- AnalNo* - analyzer number,
- ChanNo* - channel analyzer number,
- DIONo* - number of a DIO card the calibration status is transferred by,
- Index* - offset in the array of data read from the DIO card. The following values are allowable:

- 1 - states 8 inputs I1 ... I8      I1 - the least significant bit)
- 2 - states 8 inputs O1 ... O8      (O1 - the least significant bit)
- 3 - states 8 inputs O9 ... O16      (O9 - the least significant bit)

4 - states 8 inputs O17 ... O24 (O17 - the least significant bit)

The O24 output state includes the current calibration status.

### EXAMPLE

Examples for reading input/output states:

```
# analyzer no. 1, channel 2, DIO no. 3, input state I1 ... I8  
JJ_01, state I1...8, ASVC.1.2.3.1, PT3, 1, 1, NOTHING
```

```
# analyzer no. 3, channel 1, DIO 4, output state O1 ... O24  
JJ_02, state O1...24, ASVC.3.1.4.2, PT3, 1, 1, NOTHING_DW
```

### State of Transmission with the Analyzer

The driver stores information on the status of the last transmission session and on the value of the *Error Code* field transferred from the analyzer during the last transmission (if the transmission ended properly). Reading this information is realized by the following symbolic addresses:

IERR.< *Analnr*>.< *Channr*>

ICOM.< *Analnr* >

where:

- Analnr* - analyzer number,
- Channr* - number of the analyzer channel.

IERR returns the value of the *Error Code* field of the last transmission session in a given channel of the analyzer.

ICOM returns the status of the last transmission session with a given analyzer (0 - correct, 1 - transmission error).

## Driver Configuration

AK driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **CTAK** section.

- ☑ **Section name:** CTAK
- ☑ **Option name:** LOG\_FILE
- ☑ **Option value:** log\_filename

Meaning - the item allows to define a file which all the diagnostic messages of the driver will be written to. The item is used for diagnostic purpose.

Default value - by default, the log file is not created.

- ☑ **Section name:** CTAK
- ☑ **Option name:** LOG\_FILE\_SIZE
- ☑ **Option value:** number

Meaning - the item allows to define the size of the log file.

Default value - by default, the log file is not created.

Parameter:

*number* - size of the log file in MB.

- ☑ **Section name:** CTAK
- ☑ **Option name:** LOG\_OF\_TELEGRAMS
- ☑ **Option value:** YES/NO

Meaning - the item allows to write the content of telegrams sent/received by the driver to the log file (declared by means of the item LOG\_FILE). Writing the content of telegrams should be used only while the Asix start-up.

Default value - NO; by default, the driver does not write the content of telegrams to the log file.

## EXAMPLE

An exemplary driver parameterization section:

- ☑ *Section name:* CTAK
- ☑ *Option name:* LOG\_FILE
- ☑ *Option value:* d:\tmp\ctAk\ak.log

- ☑ *Section name:* CTAK
- ☑ *Option name:* LOG\_FILE\_SIZE
- ☑ *Option value:* 3

- ☑ *Section name:* CTAK
- ☑ *Option name:* LOG\_OF\_TELEGRAMS
- ☑ *Option value:* YES

## Statement of Commands

There are implemented data types (commands of the AK protocol) and parameters used when addressing variables of the particular types below.

Table 5. List of Implemented Commands of the AK Protocol for the AK Driver.

Type (command)	Addressing by Range
AAEG	Yes
AALI	Yes
AANG	Yes
AEMB	No
AGRW	No
AIKG	No
AIKO	No
AKAK	Yes
AKAL	Yes
AKON	???
ALCH	Yes
ALIK	No
ALIN	Yes
ALKO	Yes
ALST	No
AM90	No
AMBA	Yes
AMBE	Yes
AMBU	Yes
AMDR	No
AQEF	No
ASOL	No
ASTF	Nie
ASTZ	No
ASYZ	No
AT90	No
ATEM	Nie
ATOL	No
AVEZ	No
Options	
ABST	No
AKEN	No
AKOW	Yes
AUKA	Yes
ASVC	Yes (DIO number is used instead of Range)

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose

- this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify

variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

- Option value:  
 YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

- Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

- Option value:  
 YES / NO
- The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

- Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.
- The default value - by default, no time for data validity is set.

## 1.5 AM\_SA85 - Driver of MODBUS PLUS Protocol for AM-SA85-000 Card

### Driver Use

The AM\_SA85 driver is used for data exchange between the Modbus Plus network of Schneider Automation and Asix system computers provided with the AM-SA85-000 card of Schneider Automation.

Parameterization of AM\_SA85 driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the AM\_SA85 driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel

*Driver:* AM\_SA85

**AM\_SA85** tab:

**Channel parameters:**

*address* [, *adapter* [, *discrete* [, *registers*]]]

where:

*Modbus Plus network address* - address in the Modbus Plus network in the ASCII string form of R1.R2.R3.R4.R5 format, where R1 ... R5 represent sequent levels of routing to device in the network;

*Adapter* - adapter number;

*Maximal number of discrete values read out in a single telegram* - max. number of discrete values read in an individual telegram;

*Maximal number of registers read out in a single telegram* - max. number of registers read in an individual telegram.

Default values:

adapter - 0,

discrete - 120,

registers - 120.

### EXAMPLE

The definition of logical name CHANNEL operating according to the AM\_SA85 protocol and exchanging data with the controller no. 10 (other parameters are default ones):

*Channel / Name:* CHANNEL

*Driver:* AM\_SA85

*Modbus Plus network address:* 10.0.0.0.0

The AM\_SA85 driver is loaded as a DLL automatically.

## Addressing the Process Variables

The symbolic address syntax is compatible to the way of addressing used for the MODBUS protocol.

## Driver Configuration

The driver configuration is defined in the *Current Data* module, in the channel operating according to AM\_SA85 driver.

### **Log file**

Meaning - the item allows to define a file to which all the diagnostic messages of the AM\_SA85 driver and the information about the content of telegrams received by the drive will be written. If the item doesn't define a full path, the log file will be created in the current directory. The log file should be used only during the Asix system start-up.

Default value - by default, the file log isn't created.

### **EXAMPLE**

```
LOG_FILE=D:\asix\AM_SA85.LOG
```

### **Log file size**

Meaning - it allows to determine a log file size in MB.

Default value - by default, 1 MB.

### **Log of telegrams**

Meaning - the item allows writing to the log file (declared with use of the LOG\_FILE item) the contents of telegrams transmitted during data exchange between Asix and Modbus Plus network controllers; writing the telegrams content to the log file should be used only during the Asix system start-up.

Default value - by default, the contents of telegrams isn't written to the log file.

### **Send timeout**

Meaning - defines timeout for sending a query to a controller; the timeout is given as a multiple of 0.5 second.

Default value - by default, the item is set to 10 (5 seconds).

**Receive timeout**

Meaning - defines timeout for receiving an answer from a controller; the timeout is given as a multiple of 0.5 second.  
Default value - by default, the item is set to 10 (5 seconds).

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.  
Option value:  
YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).  
The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.  
Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).  
The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.  
Option value:  
YES / NO  
The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.6 AREVA - Driver for Communication with MiCOM Devices

### Driver Use

The AREVA driver is the extension of MODBUS driver in the area of generating alarms in the reaction to events proceeded in a device. The current manual focuses only on this extension. Other information you can find in the MODBUS user's manual.

Parameterization of AREVA driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the AREVA driver requires a channel with the following parameters to be added to the Current data module:

**Standard** tab:

*Name:* logical name of the transmission channel

*Driver:* AREVA

**Areva** tab:

**Channel parameters:**

*address, series, port[, baud, character, parity, stop, max\_i/o, max\_register]*

where:

<i>address</i>	- address of a device in a MODBUS network;
<i>series</i>	- number of device series: 2 lub 4;
<i>port</i>	- serial port name (maximal number of serviced ports: 32);
<i>baud</i>	- transmission speed expressed in bauds; max: 115 kBd;
<i>character</i>	- number of bits in a transmitted character;
<i>parity</i>	- parity check type (even, odd, none);
<i>stop</i>	- number of stop bits,
<i>max_i/o</i>	- maximal number of inputs/outputs, the value of which may be transferred by devices within one cycle (max 127*16 i/o states);
<i>max_register</i>	- maximal number of registers, the state of which may be transferred by the device within one cycle (max 127 registers).

Parameters baud, character, parity, stop, max\_i/o, max\_register are optional.

### Addressing the Process Variables

Declarations of variables are the same as in MODBUS driver.

## Driver Configuration

AREVA driver parameters are declared in the Miscellaneous module, the *Directly entered options* tab.

The driver parameters can be declared in 'AREVA' section as well as in sections named in the same manner as the transmission channel. The values defined in 'AREVA' section become global ones for all devices. In other sections there are placed the parameters concerning individual devices. Some parameters may be used only in 'AREVA' section, other may appear in all sections.

### Example

```
AREVA1= AREVA, 10, 4, COM1, 57600, 8, none, 1, 32, 64
AREVA2= AREVA, 10, 4, COM2, 57600, 8, none, 1, 32, 64
AREVA3= AREVA, 10, 4, COM2, 57600, 8, none, 1, 32, 64
```

*Section name:* AREVA  
*Option name:* Global\_alarms  
*Option value:* TAK

*Section name:* AREVA  
*Option name:* Recv\_timeout  
*Option value:* 500

*Section name:* AREVA1  
*Option name:* Recv\_timeout  
*Option value:* 1000

The *Global\_alarms* parameter refers to all devices. As devices are not individual parameterized, the parameter can be used only in 'AREVA' section. The *Recv\_timeout* parameter placed in 'AREVA' section defines 500 millisecond timeout for all devices with the exception of a device declared in 'AREVA1' channel (for 'AREVA1' the parameter has 1 second declaration).

There are parameters for AREVA driver described below. "Global parameter" means that a parameter can be declared only in 'AREVA' section.

- Section name: AREVA**
- Option name: Global\_alarms**
- Option value: YES|NO**

Meaning: the item controls the way of transferring alarms read from devices to the alarm system of Asix; **global parameter**.

Default value: by default, the alarms are transferred to the alarm system as global alarms (transferred to the alarm system by means of the function `AsixAddAlarmGlobalMili()`). Setting the value of the item GLOBAL\_ALARMS on NO causes that the alarms are transferred to the alarm system by means of the function `AsixAddAlarmMili()`.

- Section name: AREVA**
- Option name: Event\_check\_period**
- Option value: number**

Meaning: the option declares the interval (in seconds) between checking the state of events and alarm generation in two subsequent devices connected to the same serial port; **global parameter**.

Default value: 10

**Section name: AREVA**

**Option name: Field\_number**

**Option value: number**

Meaning: the option declares a field number for a given device; it is transmitted as an alarm parameter.

Default value: -1

**Section name: AREVA**

**Option name: Log\_file**

**Option value: file\_name**

Meaning: the item allows to define a file to which all diagnostic messages of AREVA driver and all messages describing the telegrams received and sent by AREVA driver will be written; if LOG\_FILE does not define the full path, then the log file will be created in the current directory; the log file should be used only while the Asix start-up; **global parameter**.

Default value: by default, the log file is not created.

**Section name: AREVA**

**Option name: Transmission\_delay**

**Option value: number**

Meaning: the item allows to determine a time interval (as a multiple of 10 milliseconds) between two successive operations on a communication bus; **global parameter**.

Default value: by default, the item assumes a value of 1 (10 milliseconds).

**Section name: AREVA**

**Option name: number\_of\_repetitions**

**Option value: number**

Meaning: the item allows to define maximal number of trials to do the command in case of transmission errors; global parameter.

Default value: 3

**Section name: AREVA**

**Option name: Recv\_timeout**

**Option value: number**

Meaning: allows to specify a waiting time for arriving the first character of an answer sent from a specified device.

Default value: 1000

**Section name: AREVA**

**Option name: Areva2\_Alarms / Areva4\_Alarms**

**Option value: yes/no**

Meaning: if the parameter is set to 'yes' alarm servicing of specified device series is turned on; global parameter.

Default value: yes

**Section name: AREVA**

**Option name: Alarms2\_base / Alarms4\_base**

**Option value: number**

Meaning: allows to specify a number to be added to the alarm number of a device in order to obtain the alarm number in Asix system; global parameter.

Default value: 500 for 2 series; 5000 for 4 series

**Section name: AREVA**

**Option name: Alarms\_base**

**Option value: number**

Meaning: allows to specify numeration of alarms for each device.

Default value: 500 for 2 series; 5000 for 4 series

- ☑ **Section name:** AREVA
- ☑ **Option name:** Numer\_pola
- ☑ **Option value:** field\_number

Meaning: specifies the field number for a given device; it is passed as the alarm parameter.

Default value: -1

## Numeration of Alarms

The 2 series:

Recalculation of the event number into the alarm number:

$$\text{Asix\_alarm\_number} = \text{event\_number} * 16 + \text{bit\_number} + \text{alarms\_base}$$

Bits are counted starting with BIT 0.

Numeration of bits and the meaning of individual bits is described in a device documentation.

The 4 series:

Numeration is based on the documentation: P34x\_EN\_GC\_H54.pdf "MiCOM P342, P343, P344. Generator Protection Relays. Software Version 0320. Hardware Suffix J" - "Event Record Specification for Courier and MODBUS Interfaces" table on page 117.

Range	Meaning
0 - 199	General events. The list of events is available at the end of the table in the mentioned above documentation.
200 - 231	Relay contact events, it is the number of bit counted starting with 0.
300 - 331	Opto-isolated input events; it is the number of bit counted starting with 0.
400 - 491	Latched alarms. The number of alarm can be read from the "Event index 3x10011" column. There is the text: "Bit 15=state, bits 0-14=nn "nn" - alarm number.
500 - 595	Self reset alarms. Numeration as above.
1000 - 2599	Protection events. It is the number of bit counted starting with 0.

## Alarm Parameters

Alarm parameters:

1. number of field
2. 16-bits word connected with the event and passed by a device

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.7 AS511 - Driver of AS511 Protocol for SIMATIC S5 PLCs

### Driver Use

The AS511 driver is used for data exchange with SIMATIC S5 PLCs by means of a programmer interface. The transmission is performed with use of serial interfaces of standard serial ports of Asix system computers provided with the RS232C converter - current loop 20 mA. The operation of Asix with PLCs with use of the AS511 protocol doesn't require any controller's program adaptation.

The AS511 driver of the Asix system may be used for data exchange with the following PLC types: S5-90U, S5-95U, S5-100U, S5-115U, S5-135U.

Parameterization of AS511 driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the AS511 driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: AS511

**AS511** tab:

**Channel parameters:**  
*port*, [*baud*, *character*, *parity*, *stop*]

where:

<i>port</i>	- serial port name;
<i>baud</i>	- transmission speeds in bauds; the transmission speed must be equal to 9600 bauds;
<i>character</i>	- number of bits in a transmitted character;
<i>parity</i>	- parity check type (even, odd, none).

The parameters *baud*, *character*, *parity*, *stop* i *buffer* are optional. When they are omitted, the default values are as follows:

- transmission speed - 9600 Bd,
- number of bits in a character - 8,
- parity check type - parity check,
- number of stop bits - 2.

#### EXAMPLE

An exemplary item defining the use of transmission channel operating according to the AS511 protocol is given below:

*Channel / Name*: CHANNEL  
*Driver*: AS511  
*Port*: COM1

The transmission channel with the logical name CHAN1 has the following parameters defined:

- AS511 protocol using a serial interface,
- port COM1,
- transmission speed of 9600 Bd,
- transmitted character length - 8 bits,
- parity check,
- two stop bits.

## Addressing the Process Variables

The syntax of symbolic address which is used for variables belonging to the AS511 driver channel is as follows:

*variable\_type* [*db\_number*.]*variable\_index*

where:

- variabletype* - string identifying the variable type in the controller;
- db\_number* - optional number of a data block; it is used only in case of process variables which map the content of words in data blocks;
- variable\_index* - variable index within a given type. In case of data blocks it is the word no. in a data block.

The following symbols of process variables are allowed:

- EA - states of outputs, transferred in bytes,
- EAW - states of outputs, transferred in words,
- EE - states of inputs, transferred in bytes,
- EEW - states of inputs, transferred in words,
- EM - states of marks (flags), transferred in bytes,
- EMW - states of marks (flags), transferred in words,
- EZ - states of counters, transferred in words,
- ET - states of clocks, transferred in words,
- ED - values of words in data blocks,
- EL - values of double words in data blocks,
- EG - values of double words in data blocks, treated as a floating-point number in KG format,

### EXAMPLES

- ED10.22 - word no. 22 in the data block no. 10
- EL20.32 - double word placed in words no. 32 and no. 33 in the data block no. 20
- EZ50 - counter no. 100

## Driver Configuration

AS511 driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **AS511** section.

- Section name: AS511**
- Option name: BLOCK**
- Option value: YES/NO**

Meaning - allows reading the whole data block.

Default value - YES

**NOTE** The AS511 driver (from 1.23 version) allows reading words of data placed in data blocks by reading the whole block instead of determined part of cache (like it was in the previous version). It allows data reading from the 115F controller. Reading the whole block is possible if the parameter 'block' is placed in the AS511 section.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

## Communication Drivers

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.8 AS512 - Driver of AS512 Protocol for SIMATIC S5 PLCs

### Driver Use

The AS512 driver is used for data exchange with SIMATIC S5 PLCs provided with the CP524/CP525 communication processor. The transmission is performed with use of serial interfaces of standard or additional serial ports of a computer.

The controller software must be prepared for cooperation with Asix, i.e.

- program in a CPU controller must include calls of functional blocks handling receiving and sending telegrams with use of a CP524/CP525 communication processor (*SEND\_ALL*, *RECV\_ALL*). The number of calls of these blocks within a controller operation cycle define the number of telegrams, which may be sent during the cycle between the computer and controllers !
- software of communication processor must use the 3964R procedure and transmission speed must be the same as the rate declared in the transmission channel of ASMEN.

Parameterization of AS512 driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the AS512driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: AS512

**AS512** tab:

**Channel parameters:**

*port*, [*,baud,character,parity,stop,cpu*]

where:

<i>port</i>	- serial port name,
<i>baud</i>	- transmission speed in bauds; the transmission speed should not exceed the value of 38400 bauds,
<i>character</i>	- number of bits in a transmitted character,
<i>parity</i>	- parity check type (even,odd,none),
<i>stop</i>	- number of stop bits,
<i>cpu</i>	- CPU number in the controller, to which the carried out operation refers.

The parameters *baud*, *character*, *parity*, *stop*, *cpu* i *buffer* are optional. In case of omitting them the default values are as follows:

- transmission speed - 9600 Bd,
- number of bits in a character - 8,
- parity check type - parity check,
- number of stop bits - 1,
- CPU number - 0.

### EXAMPLE

An example item, which defines the use of transmission channel operating according to the AS512 protocol, is given below:

**Channel / Name: CHAN1**

**Driver: AS512**

**Channel parameters: COM1,4800,8,even,1,2**

The transmission channel of the logical name CHAN1 has the following parameters:

- AS512 protocol using a serial interface,
- port COM1,
- transmission speed of 4800 Bd,
- transmitted character length - 8 bits,
- parity check,
- one stop bit,
- data exchange concerns the CPU no. 2.

## Addressing the Process Variables

During declaration of process variable its symbolic address is entered. It is used as an unique definition of the controller variable, the value of which will be assigned to the process variable in Asix. The syntax of symbolic address which is used for the variables belonging to the AS512 driver channel, is presented below:

*variable\_type [db\_number.]variable\_index*

where:

- variable\_type* - string identifying the variable type in the controller;
- db\_number* - optional number of a data block; it is used only in case of process variables which are the content mapping of words in data blocks;
- variable\_index* - variable index within a given type. In case of data blocks, it is the word no. in a data block.

The following symbols of process variables types (following the names of variable types used by SIEMENS) are permitted:

- EA - states of outputs, transferred in bytes,
- EE - states of inputs, transferred in bytes,
- EM - states of marks (flags), transferred in bytes,
- EZ - states of counters, transferred in words,
- ET - states of clocks, transferred in words,
- ED - values of words in data blocks,
- EL - values of double words in data blocks,
- EG - values of double words in data blocks, treated as a number in the KG floating-point format.

### EXAMPLES

- ED10.22 - word no. 22 in the data block no. 10
- EZ100 - counter no. 100

The AS512 driver is loaded as a DLL automatically.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value

- NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value

- by default, no time for data validity is set.

## 1.9 AS512S7 - Driver of AS512 Protocol for SIMATIC S7 PLCs

### Driver Use

The AS512S7 driver is used for data exchange with SIMATIC S7 PLCs provided with the CP340 communication processor. The transmission is performed with use of serial interfaces in standard serial ports of a computer according to the AS512 protocol.

The ASKOM company offers the software for the SIMATIC S7 PLC that enables data exchange with Asix according to the AS512 protocol.

Parameterization of AS512S7 driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the AS512S7 driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: AS512S7

**AS512S7** tab:

**Channel parameters:**

*port*, [, *bauds*, *character*, *parity*, *stop*, *cpu*]

where:

<i>port</i>	- name of the serial port,
<i>bauds</i>	- transmission speed in bauds,
<i>character</i>	- number of bits in a transmitted character,
<i>parity</i>	- parity check type (even, odd, none),
<i>stop</i>	- number of stop bits,
<i>cpu</i>	- number of the CPU (to which the carried out operation refers) in the controller.

The parameters *bauds*, *character*, *parity*, *stop*, *cpu* are optional. When they are omitted, the default values are as follows:

- transmission speed - 9600 Bd,
- number of bits in a character - 8,
- parity check type - parity check,
- number of stop bits - 1,
- CPU number - 0.

**EXAMPLE**

An example item, which defines the use of transmission channel operating according to the AS512S7 protocol, is given below:

*Channel / Name:* CHAN1  
*Driver:* AS512S7  
*Channel parameters:* COM1,4800,8,even,1,2

The transmission channel of the logical name CHAN1 has the following parameters defined:

- AS512S7 protocol using a serial interface,
- port COM1,
- transmission speed of 4800 Bd,
- transmitted character length - 8 bits,
- parity check,
- one stop bit,
- data exchange concerns the CPU no. 2.

During the declaration of process variable its symbolic address is entered. It is an unique definition of the controller variable, the value of which will be assigned to the process variable in Asix.

## Addressing the Process Variables

The syntax of symbolic address which is used for the variables belonging to the AS512S7 driver channel is presented below:

*variable\_type* [*db\_number.*]*variable\_index*

where:

*variable\_type* - string identifying the variable type in the controller;  
*db\_number* - optional number of a data block; it is used only in case of process variables, which are the content mapping of words in data blocks;  
*variable\_index* - variable index within a given type. In case of data blocks, it is the word no. in a data block.

In the AS512S7 protocol only the access to words in data blocks is implemented. For this reason there is only one type of process variables allowed:

ED - values of words in data blocks.

**EXAMPLES**

ED10.22 - word no. 22 in the data block no. 10

The AS512S7 driver is loaded as a DLL automatically.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

- Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.
- Option value:  
YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

- Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.
- Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

- Purpose - with this option deployed, local control in a channel will be disabled.
- Option value:  
YES / NO
- The default value - NO.

### **Data Validity Period**

- Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.
- Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.
- The default value - by default, no time for data validity is set.

## 1.10 BACnetIP - Driver for BACnet/IP Protocol Devices

### Driver Use

The BACnetIP driver is designed for data exchange between ASIX and devices with BACnet/ IP protocol interface.

The driver allows to read (and possibly write) the properties of the following BACnet standard objects:

AI	- analog input object,
AO	- analog output object,
AV	- analog value object,
AVG	- averaging object,
BI	- binary input object,
BO	- binary output object,
BV	- binary value object,
CAL	- calendar object,
DEV	- device object,
LP	- life-safety point object,
LZ	- life-safety zone object,
MI	- multi-state input object,
MO	- multi-state output object,
MV	- multi-state value object,
SCH	- schedule object,

and signal events transmitted from devices using telegrams of the UnconfirmedEventNotification and ConfirmedEventNotification type.

The driver has been designed according to ISO 16484-5: 2003.

Parameterization of BACnetIP driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel operating according to the BACnetIP driver protocol requires adding a new channel with the following parameters to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: BACnetIP

**BACnetIP/Channel Parameters:**

<i>IP Address of the Computer</i>	- IP address of the Asix computer card;
<i>IP Address of the Device</i>	- device IP address;
<i>ID of the Device</i>	- device ID;
<i>Event Definitions File</i>	- name of the file containing definitions of events,

## Communication Drivers

<i>Alarms Offset</i>	- offset added to the number of each alarm defined for the device in the event definition file;
<i>Write Priority</i>	- priority of controls for 'commandable' objects (AO, BO, MO, AV, BV, MV); allowable range of the parameter is 1 - 16. Default value is 16 (lowest priority);
<i>Port</i>	- computer port number used to connect with a PLC; default value: 0 - port is provided by the Windows system;
<i>Alarm Number</i>	- alarm number reported to the alarm system when there is no communication in the channel;
<i>Log File</i>	- log file name; if not specified - messages will be written to the driver log;
<i>Log File Size</i>	- log file size in MB; default value: 10;
<i>Log of Telegrams</i>	- declaration of saving telegrams to the log file.

### EXAMPLE

Below is an example of the KANAL channel declaration. The device ID is set on 348380. Communication is by the network card with the address 10.10.105.1 and the device IP address is 10.10.105.212. Definitions of events generated by the PLC are in the file Dev348380.def. And the offset 100 is added to each alarm reported to the Asix system.

Name: KANAL  
IP Address of the Computer : 10.10.105.1  
IP Address of the Device: 10.10.105.212  
ID of the Device: 348380  
Event Definitions File: Dev348380.def  
Alarms Offset: 100  
Write Priority: 14

## Addressing the Variables

The driver supports only the following BACnet standard objects:

AI	- analog input object,
AO	- analog output object,
AV	- analog value object,
AVG	- averaging object,
BI	- binary input object,
BO	- binary output object,
BV	- binary value object,
CAL	- calendar object,
DEV	- device object,
LP	- life-safety point object,
LZ	- life-safety zone object,
MI	- multi-state input object,
MO	- multi-state output object,
MV	- multi-state value object,
SCH	- schedule object.

The syntax of symbolic address which is used for the variables belonging to the BACnetIP driver channel is presented below:

*objectType.instance[.property[.item]]*

where:

*objectType* - BACnet object name (from the list above);

*instance*           - object instance number;  
*property*           - object property ID;  
*item*                - element of object property if the object property is an array (eg. element of 'date-;list' property of a calendar).

The syntax for SCHEDULE object is as follows:

*objectType.instance.property.dayNr.item.[VAL / TM]*

where:

*dayNr*             - day number of the week (Monday 0, Sunday 7),  
*item*               - array element for a given day (indexed from 0),  
 VAL               - value of the 'value' parameter for the 'item' element,  
 TM                 - value of the 'time' parameter for the 'item' element.

## EXAMPLE

An exemplary variable definitions:

*Name:* X01  
*Description:* device\_max\_apdu\_length  
*Address:* DEV.62  
*Channel:* KAN  
*Elements Count:* 1  
*Sample Rate:* 1  
*Conversion Function:* NIC

*Name:* X02  
*Description:* analog-value\_1\_present\_value  
*Address:* AV.1.85  
*Channel:* KAN  
*Elements Count:* 1  
*Sample Rate:* 1  
*Conversion Function:* NIC\_FP

*Name:* X03  
*Description:* analog-value\_1\_object\_name  
*Address:* AV.1.77  
*Channel:* KAN  
*Elements Count:* 50  
*Sample Rate:* 1  
*Conversion Function:* NIC\_TEXT

*Name:* X04  
*Description:* binary-value\_1\_present\_value  
*Address:* BV.1.85  
*Channel:* KAN  
*Liczba elementów:* 1  
*Sample Rate:* 1  
*Conversion Function:* NIC

*Name:* X05  
*Description:* device\_local\_time  
*Address:* DEV.57  
*Channel:* KAN  
*Elements Count:* 20  
*Sample Rate:* 1  
*Conversion Function:* NIC\_TEXT

## Communication Drivers

*Name:* X06  
*Description:* device\_local\_date  
*Address:* DEV.56  
*Channel:* KAN  
*Elements Count:* 20  
*Sample Rate:* 1  
*Conversion Function:* NIC\_TEXT

*Name:* X07  
*Description:* item nr 6 of calendar\_5  
*Address:* CAL.5.23.6  
*Channel:* KAN  
*Elements Count:* 30  
*Sample Rate:* 1  
*Conversion Function:* NIC\_TEXT

*Name:* Y20  
*Description:* 'value' elem. 4 środa SCH nr 1  
*Address:* SCH.1.123.2.4.VAL  
*Channel:* KAN  
*Elements Count:* 1  
*Sample Rate:* 1  
*Conversion Function:* NIC\_BYTE

*Name:* Y21  
*Description:* 'time' elem. 3 wtorek SCH nr 2  
*Address:* SCH.2.123.1.3.TM  
*Channel:* KAN  
*Elements Count:* 30  
*Sample Rate:* 1  
*Conversion Function:* NIC\_TEXT

### CALENDAR Object Variable Values

The CALENDAR object variable values are always expressed in ASCII strings and take one of the following formats:

<b>yyyy-mm-dd</b>	(date)
<b>yyyy-mm-dd - yyyy-mm-dd</b>	(date range)
<b>D.W.M</b>	(day, week, month)

where:

<b>yyyy</b>	- number of year
<b>mm</b>	- number of month
<b>dd</b>	- number of month day
<b>D</b>	- number of week day: 1 - Monday ... 7 - Sunday * - every day of week
<b>W</b>	- number of week in month: 1 - days from 1 to 7 2 - days from 8 to 14 3 - days from 15 to 21 4 - days from 22 to 28 5 - days from 29 to 31 6 - last 6 days of month * - every week of month

**M** - number of month:  
 1 - January  
 ...  
 12 - December  
 \* - every month

Notice:

When saving, it is allowed that the control value will be an empty ASCII string - such a control value means a request to delete the item from the calendar. If a deleted item from the calendar is not the last item, the next items will be shifted forward by one position (Calendar does not allow to save undefined items).

### SCHEDULE Object Variable Values

The SCHEDULE object variable values depend on the schedule parameter type.

For the parameter 'time' they are expressed by ASCII string of the following format:

**hh:mm:ss.zz**

where:

**hh** - hour  
**mm** - minute  
**ss** - second  
**zz** - hundredths of a second

For the parameter 'value' they are expressed by the number of one of the following types:

bool  
 unsigned  
 signed  
 real  
 double

If the parameter 'value' has the value NULL in a PLC, the variable takes the value 0 and receives the status OPC\_QUALITY\_OUT\_OF\_SERVICE.

Notice:

When saving, it is allowed that the control value will be an empty ASCII string for the parameter 'time' - such a control value means a request to delete the item from the schedule (both 'time' as well as 'value'). If a deleted item from the schedule is not the last item, the next items will be shifted forward by one position (SCHEDULE does not allow to save undefined items).

## Signalling Events

The driver supports the 'ConfirmedEventNotification' and 'UnconfirmedEventNotification' telegrams. The way the telegram is interpreted depends on the 'Notify-Type' field value of a received telegram:

- a/ if the field 'Notify-Type' = 'alarm' and the field 'To-State' = 'normal', the driver signals the end of the Asix alarm,
- b/ if the field 'Notify-Type' = 'alarm' and the field 'To-State' != 'normal', the driver signals the start of the Asix alarm,
- c/ Notify-Type = 'event', the driver reports the start and the end of the Asix alarm (telegram content is treated as an event).

The telegram content about an event is converted into Asix system alarms based on event definitions read from an event definition file. The event definition file name is placed in the transmission channel declaration (*Event Definitions File*). No *Event Definitions File* parameter declared in the channel definition causes the telegrams about events in a given channel are ignored by the driver.

The *Alarms Offest* parameter (placed in the channel declaration) is added to each of the Asix alarm read from the event definition.

When using Asix.**Evo**, all the parameters of a given event are transferred to the alarm system.

When using **classical** Asix, only the parameters specified in the definition of a given event in the event definition file are transferred.

The syntax of event definition is as follows:

***asixAlarmNr, BacnetObject, eventType [,par0[,par1[,par2[,par3]]]]***

where:

- asixAlarmNr* - alarm number in Asix system
- BacnetObject* - BACnet ObjectIdentifier and instanceNr of BACnet object that generated an alarm in a PLC,
- eventType* - name of BACnet alarm type (values assumed by the enumerator **BACnetEventType**):
  - CHANGE\_OF\_BITSTRING
  - CHANGE\_OF\_STATE
  - CHANGE\_OF\_VALUE
  - FLOATING\_LIMIT
  - OUT\_OF\_RANGE
- parn* - number of BACnet alarm parameter the value of which will be transferred to the Asix alarm system; parameter type: WORD or FLOAT (depending on the BACnet parameter number and alarm type); parameters are numerated starting from 0.

Notice:

1. Types of parameters for each alarm type is described in ISO 16484-5:2003 - **BACnetNotificationParameters** type.
2. The total number of bytes occupied by the values of the parameters transferred with an alarm to the Asix alarm system can not exceed 8 bytes (limitation of Asix alarm system). You can then transmit 2 parameters of FLOAT type, 2 parameters of DWORD type, 4 parameters of WORD type and so on.

Examples of event definitions:

10, AV.2, OUT\_OF\_RANGE, 0, 3

The meaning of the above definition:

alarm of OUT\_OF\_RANGE type coming from the object *analog-value instance 2* generates the Asix alarm no. 10. Together with the alarm the following parameter values are passed:

parameter nr 0	- exceeding-value	(FLOAT)
parameter nr 3	- exceeded limit	(FLOAT)

20, AV.3, FLOATING\_LIMIT, 0, 2

The meaning of the above definition:

alarm of FLOATING\_LIMIT type coming from the object *analog-value instance 3* generates the Asix alarm no. 20. Together with the alarm the following parameter values are passed:

parametr nr 0 - reference-value (FLOAT)  
parametr nr 2 - setpoint-value (FLOAT)

## Control

For BACnet objects with the property 'priority-array' ('commandable' objects) the controls are performed using the commands transferring a control value and a priority. By default, the controls with priority are performed for the objects: AO, BO, MO, AV, BV, MV.

The priority value is declared separately for each of channels with the use of the parameter *Write Priority*. If the parameter is not used in the channel declaration, the default value shall be 16 (lowest priority).

The priority value defined for the channel is valid for all BACnet objects supported in that channel. That value can be overridden for any object by entering a desired priority value (from 1 to 16) into the virtual property with the ID 512 of the object. To achieve this functionality, you have to create an additional variable the symbolic address of which will appeal to the virtual property with the ID 512 of the object.

### Removing Controls

For the BACnet objects with the property 'priority-array' ('commandable' objects) you can remove an active control (placed in the array 'priority-array'). Removal of control consists in writing the value NULL into the array 'priority-array' at the position corresponding to the priority with which controls of the object are performed.

To achieve the above functionality you need to create an additional variable the symbolic address of which refers to the virtual property with the ID 513 of the object. The operation of removal of control consists in writing any control value to a such variable.

## Driver Parameters

The driver configuration is defined in the *Current Data* module, in the channel operating according to BACnetIP driver, on the tabs **BACnetIP / Driver parameters**.

### Log File

Meaning: text log file to which driver status messages are stored; it is used for diagnostic purposes.

Parameter:

*log\_file\_name* - file name

Default value: file is not created

### Events Log File

Meaning: text log file to which messages about events transferred from PLCs are stored; it is used for diagnostic purposes.

Parameter:

*log\_file\_name* - file name

Default value: file is not created

### Log File Size

## Communication Drivers

Meaning: option is used to determine the log file size defined by the Log file and Event Log File options.  
Parameter:  
*number* - log file size in MB  
Default value: 10

### **Log of Telegrams**

Meaning: option allows writing the contents of messages transmitted between the driver and the controllers to the Log File (declared using the Log File option). The option should only be used during the Asix system start-up. The option value can be overridden by using the same option but in channel parameters.

Parameter:  
YES / NO  
Default value: by default, the option is disabled

### **Event Definitions Path**

Meaning: option specifies the full path to the directory where the driver will search for event definition files declared in the channel declaration.

Parameter:  
*path* - path to the directory  
Default value: no path declared

## EXAMPLE

An exemplary declaration of driver parameters:

*Log File*  
d:\tmp\CtBACnetIP\bacnet.log

*Events Log File*  
d:\tmp\CtBACnetIP\events.log

*Log File Size*  
30

*Log of Telegrams*  
TAK

*Event Definitions Path*  
d:\tmp\EventsDef

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the

names of network computers which will be permitted to exercise remote control.

Option value:  
 YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
 YES / NO  
 The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.11 BAZA - Driver for Access to Database

### Driver Use

The BAZA driver allows to import data into the Asix system from databases. The access to database is realized on the basis of the ADO technology. BAZA makes all the data stored in databases available to the Asix system. The received data may (but they needn't) be stamped with a status and time. The driver also allows reading from other sources like an Excel spreadsheet. If data are stamped with a time, the driver allows to complete historical data in ASPAD archives. When data stored in a database are not stamped with a time, the data newly received by the driver are stamped with a current time. If a datum is not stamped with a status, the status *proper datum* will be assigned to it.

Parameterization of BAZA driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the BAZA driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: BAZA

**BAZA** tab:

**Channel parameters:**

**database**

where:

*database* - field that determines a database.

It can be:

- file name; the name has to allow to distinguish it from the database name i.e. it has to include the "." or "\" character; when declaring a file name as *database*, it is assumed that it is a Microsoft Jet database (Microsoft.Jet.OLEDB.4.0);

- database name; in such case the database is assumed to be serviced by an SQL Server on a local station (SQLOLEDB);

- connection string put in quotation marks; this specification form allows to determine an arbitrary database, i.e. to determine the following parameters: database server localization (e.g. remote computer), user name, password, timeout of establishing a connection etc.; this form allows to specify a database as a DSN name;

- section name, put in square brackets, in which elements of the connection string are placed; this form is used in case of a long connection string.

### EXAMPLE

An exemplary declaration of the transmission channel:

Microsoft Jet database:

*Channel / Name:* Measurements1  
*Driver:* BAZA  
*Channel parameters:* c:\Pomiary.mdb

Database defined by DSN data source (of computer or user):

*Channel / Name:* Measurements2  
*Driver:*  
*Channel parameters:* "DSN=Pomiary"

Database defined by file DSN data source:

*Channel / Name:* Measurements3  
*Driver:*  
*Channel parameters:* "FILEDSN=C:\BAZA\Pomiary.dsn"

Database defined by UDL file (Microsoft Data Link):

*Channel / Name:* Measurements4  
*Driver:* Baza  
*Channel parameters:* "File Name=C:\BAZA\Pomiary.UDL"

SQL database named "Pomiary" on local computer:

*Channel / Name:* Measurements5  
*Driver:* BAZA  
*Channel parameters:* Pomiary

SQL database named "Pomiary" on computer "Emisja":

*Channel / Name:* Measurements6  
*Driver:* BAZA  
*Channel parameters:* "Provider=SQLOLEDB.1;Data Source=Emisja; Initial Catalog = Pomiary; Integrated Security=SSPI;Persist Security Info=False"

## Addressing the Variables

The syntax of symbolic address which is used for the variables belonging to the BAZA driver channel is presented below:

*array\_declaration*[,*value\_field*[.*time\_field*][*.status\_field*]]

where:

*array\_declaration* - expression that determines an array in the database (set of records);  
*value\_field* - field name (column) containing the datum value;  
*time\_field* - field name (column) containing the data time (Data/Time type);  
*status\_field* - field name (column) containing the data status (numerical type - OPC status).

*Value\_field* may be omitted if *array\_declaration* determines the array containing one column.

If *time\_field* is omitted, a current time is taken.

If *status\_field* is omitted, the *proper data* status is assigned to the variable value.

*Array\_declaration* can be:

- name of the array placed in the database;
- query, put in apostrophes or round brackets, sending to the database by the driver in order to data readout;
- symbolic name in the  $\$(name)$  form. The name determines the query the syntax of which is defined in the [BAZA] section in an initialization file.

In the most simple and typical case the array is determined by its name. If, for example, the database contains the array named Pomiary containing Temperature, Pressure, Time and Status columns, then variable addresses would have the following form:

Pomiary.Temperature.Time.Status  
Pomiary.Pressure.Time.Status

#### EXAMPLE

When the more complex rule defining the records is necessary, other address forms may be used, i.e. with query text, e.g.:

(SELECT \* FROM Pomiary WHERE ...). Temperature.Time.Status

The queries have to be constructed to determine a record set ordered decreasingly according to the time. When reading current data, the driver modifies the query to read the newest record (the driver adds TOP 1 fraze). When reading historical data (only when address contains *time* field), the driver adds or modifies the WHERE fraze to receive data from the specified time range.

The other form of using the queries is use of a query name. The query is defined in the [BAZA] section in an initialization file. The name use allows:

- shortening the address in case of using many variables with the same query but different value fields;
- avoidance of errors in case of necessity of using the characters (which are interpreted by ASMEN in a different manner) in the query;
- optimization of a query number, i.e. if the array of many variables is the result of the query, then the driver sends only one query instead of one by one for each variable.

An exemplary declaration of query name use:

Definition of the variable address:

Name: Temperature

Address:  $\$(QUERY1)$ .Temperature.Time.Status

Name: Pressure

Address:  $\$(QUERY1)$ .Pressure.Time.Status

Initialization file:

[BAZA]

Query1 = SELECT \* FROM Pomiary WHERE ...

<b>NOTICE</b> <i>The quotation marks ("") should NOT be used in the address.</i>
--

## Driver Configuration

BAZA driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The BAZA driver may be configured using the **BAZA** section placed in the application initialization file or sections having the same name as channel names. The parameters placed in the BAZA section apply to all driver channels. The parameters placed in other sections refer to a specified channel. If the parameter is defined both in the BAZA section and in the channel section, then the parameter referring to a specified channel has a higher priority.

**Section name: BAZA**

**Option name: No\_TOP**

**Option value: YES/NO**

Meaning - if the item has value NO, then the driver puts in the SQL query the fraze TOP limiting a number of read records. Some databases don't allow using the fraze TOP - then one should declare the value YES.

Default value - NO.

**Section name: BAZA**

**Option name: log\_file**

Meaning - defines a file to which all the diagnostic messages of the driver and the information about the content of telegrams received by the driver will be written.

Default value - lack.

**Section name: BAZA**

**Option name: Max\_history**

**Option value: number**

Meaning - determines a time period from the current moment backwards, for which historical data, saved in the station memory, will be read.

Default value - 30.

**Parameter:**

*number* - number in days.

**Section name: BAZA**

**Option name: Record\_optimize**

**Option value: number**

Meaning - the parameter refers to variables determined by the name of the array. If the item has the value Yes, then only one (common for all variables) SQL query, causing readout of the record containing only the fields that occur in variable addresses, will be formulated for all variables placed in the array. If the parameter has the value No, all fields in array will be read.

Default value - yes.

**Section name: BAZA**

**Option name: Order**

**Option value: yes/no**

Meaning - if the parameter has the value Yes, the driver will formulate an SQL query in such way that read records will be ordered in according to the time fields. If the parameters has the value No, the records will not be ordered.

Default value - yes.

**Section name: BAZA**

**Option name: History\_records**

**Option value: number**

Meaning - the parameter determines the maximal number of records read from the database once during historical data reading. The parameter has a sense only when *No\_TOP* parameter takes the value No.

Default value - 1000.

**Parameters:**

*number* - number of records.

- Section name: BAZA**
- Option name: UTC**
- Option value: yes/no**

Meaning - determines whether the time written in the database is an UTC time (Universal Time Coordinate or Greenwich Mean Time). If the parameter has the value Yes, the time is an UTC time, if the parameter has the value No - the time is a local time.

Default value - no.

## Optimization of Field Number in a Record

The driver formulates one SQL query, common for all variables contained in the same array, causing readout of record including only the fields that occur in variable addresses. If the address even of one variable includes the name of the field that doesn't occur in the array, then reading of all array variables ends with an error. To determine which of variables has an incorrect name, one should declare *Record\_optimize=No* - thanks to such solution the error will refer only to wrongly declared variables. If the error refers to the time field name, one should additionally set the *Order=No* parameter - but absence of ordering may cause reading incorrect data.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:  
YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.  
The default value - by default, no time for data validity is set.

## 1.12 BUFOR - Driver of General Purpose

### Driver Use

The BUFOR driver allows data exchange between the Asix system and any program developed by the user to transfer the process variables.

The transmission channel based on the BUFOR protocol is created by two programs:

- driver of the protocol BUFOR for general purpose (included in the Asix package),
- driver of process data transmission developed by the user.

The user program must be implemented in the form of a process operating in Windows XP / 2000 / NT 4.0 environments.

The data exchange between the BUFOR driver and the user program is performed by means of an exchange file (memory mapped file). The synchronization of the access to a memory mapped file is carried out by using a mutex object.

Parameterization of BUFOR driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the BUFOR driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: BUFOR

**BUFOR** tab:

**Channel parameters:**

*FILE\_MMF, USER, PAR1, PAR2, PAR3*

where:

FILE\_MMF - memory mapped file name;  
USER - user process name;  
PAR1..PAR3 - parameters which are transferred to the user process.

The description of data structures and rules of cooperation of a user process with the BUFOR driver is included in the file DrBufor.hlp.

## Addressing the Process Variables

The syntax of symbolic address which is used for process variables belonging to the BUFOR driver channel is as follows:

*Ivariable\_index*

where:

*I* - constant element of symbolic address for the BUFOR driver channel;  
*variable\_index* - variable index in an array of user driver variables. The first variable has the index 1.

The other parameters in a process variable declaration have typical meaning.

The BUFOR driver is loaded as a DLL automatically.

## Possibility of Mutex Name Declaration

It is possible to declare a mutex name in the MUTEX option declared in the *Miscellaneous* module, the *Directly entered options* tab:

- Section name: the same as name of the channel that uses BUFOR driver*
- Option name: MUTEX*
- Option value: mutex\_name*

Example:

*Section name: CHANNEL*  
*Option name: BUFOR*  
*Option value: TEST\_MMF, userebuf, par1 par2 par3*

Mutex declaration:

*Section name: CHANNEL*  
*Option name: MUTEX*  
*Option value: TEST\_MUTEX*

When a mutex name is declared, the BUFOR driver operates according to the scheme:

- the driver creates the user's process by means of the procedure *CreateProcess()* and expects that the process will create a mutex with the name given in the MUTEX position; while creating a mutex, the process must occupy it at once!
- while giving the mutex name declared in the MUTEX position, the driver calls the *OpenMutex()* function; if *OpenMutex ()* returns an error, the driver will finish the channel initialization with the error;
- the driver expects till the user's program releases mutex and after that realizes the stage of verification of memory mapped file contents.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.13 Calec - Driver of CALECMCP Device Protocol

### Driver Use

The Calec driver is designed to retrieve current values of variables from the CALEC MCP meter of Aquametro.

Calec is a dynamic link library (DLL) with an interface meeting the requirements of the UniDriver module. The Calec driver together with UniDriver create a driver meeting the requirements of the ASMEN module.

Parameterization of Calec driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the Calec driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: *Calec*

**Calec** tab:

**Channel parameters:**

*driver parameters*

where:

*driver parameters* - configuration parameters of Calec described in the further part of the specification.

During loading and initialization of an Asix application, Calec receives from the ASMEN module an address of each process variable retrieved from the source definition of variables (in sequence and one time). The variables supplied by the driver are read-only.

#### Configuration Parameters

The driver supports the following configuration parameters described in the following table.

**Table 6. Configuration Parameters of CTCalec.**

Parameter Name	Meaning	Default Value
<i>Port</i>	Indicates the name and parameters of the serial port the device is connected to. For detail description, see below.	N/A- parameter required
<i>ReadingPeriod</i>	Indicates the period (in seconds); a variable value will be updated every <i>ReadingPeriod</i> in the internal driver buffer.	10 s

<i>ReadingTimeout</i>	The maximal time of variable reading from the device (in milliseconds), it is recommended not to be less than 1200.	1200 ms
-----------------------	---	---------

### EXAMPLE

An example of the Calec channel declaration (all items must be written in one line):

*Channel / Name:* CalecMCP

*Driver:* Calec

*Channel parameters:* Port=COM1:9600:8:even:1, ReadingPeriod=10, ReadingTimeout=1200

### Configuration Parameters of Serial Port

The full syntax of the serial port declaration is as follows:

Port=<*port name*>:<*rate*>:<*character*>:<*parity*>:<*stop*>

where:

- port name* - serial port name in the operation system, e.g. COM1 or COM2;
- rate* - baud rate of serial transmission;
- character* - number of bits in a transmitted character;
- parity* - parity check type (odd, even or none);
- stop* - number of stop bits, it may be entered 1, 1.5 (1.5 bits) or 2.

The simplified syntax includes *port name* only:

Port = <*port name*>

For the other parameters default values are assigned respectively: 9600:8:even:1

## Addressing the Process Variables

A variable address consists of a hexadecimal address of the variable in the device and of the type of variable value coding separated by a colon. As the type of variable value it may be assumed FLOAT for floating-point variables and FIX for fixed-point variables.

### EXAMPLE

- 2000:FIX Fixed-point variable to be retrieved from the address 2000 in hexadecimal format,
- 2080:FLOAT Floating-point variable to be retrieved from the address 2080 in hexadecimal format.

For floating-point variables the conversion function NOTHING\_FP should be used and for fixed-point variables the conversion function NOTHING.

## Period of Variable Refreshing (Updating)

The reading time of one variable is usually in the range from 520 ms to 630 ms (average 570 ms) but for some variables (in a tested device it was the measurement named Temperaturdifferenz) it may reach 1300 ms. When defining a driver reading period and variable refreshing period it is necessary to take into consideration that it should be physically possible to read the required variable in the defined refreshing period. Generally the driver reading period and the variable refreshing period are the same.

## Protocol Functions Excluded from Implementation

In the documentation it is mentioned that there is at least one *text* type variable in the device. Because the transmission method of this data type is not described, it is not supported by the driver.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:  
 YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

## Communication Drivers

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.14 CAN\_AC\_PCI - Driver of CANBUS Protocol for CAN ACx PCI Card

### Driver Use

The CAN\_AC\_PCI driver is used for data exchange between SELECONTROL MAS PLCs of Selectron Lyss AG and Asix system computers by using the CAN network. The Asix system computer must be provided with a card of communication processor CAN\_AC1 or CAN\_AC2 of Softing GmbH.

Parameterization of CAN\_AC\_PCI driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the CAN\_AC\_PCI driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: CAN\_AC\_PCI

**CAN\_AC\_PCI** tab:

**Channel parameters**:

*interface\_no*

where:

*interface\_no* - number identifying the CAN\_AC1/CAN\_AC2 card interface by means of which the transmission with the CAN network is executed. In the CAN\_AC1 card the interface no. 1 may be used exclusively.

The CAN\_AC\_PCI driver is loaded as a DLL automatically.

### Addressing the Process Variables

Values of process variables are transferred in telegrams sent by CAN based controllers connected to a CAN network. Each telegram consists of at most 8 bytes that may be identified as:

- bytes indexed 1 - 8 (type BY),
- 16-bit numbers indexed 1 - 4 (type WD),
- 32-bit numbers indexed 1 - 2 (type DW).

The CAN\_AC\_PCI driver distinguishes the following types of access to process variables:

- read-only (type R\_),
- write-only (type W\_),
- read/write (type RW\_).

The addressing of process variables consists in indication of:

- access type (R\_, W\_ or RW\_);
- variable type (BY, WD, DW);
- telegram no. (for variables with RW\_ access type it is the telegram number used to read the variable),

## Communication Drivers

- index within the telegram (for variables with RW\_ access type it is the index in the telegram used to read the variable),
- for variables with RW\_ access type it is necessary to declare in addition:
  - a/ telegram no. used to read the variable,
  - b/ index in the telegram used to write variable.

The syntax of symbolic address which is used for variables belonging to the CAN\_AC\_PCI driver channel is as follows:

`<access_type><variable_type><tel>.<index>[.<tel>.<index>]`

where:

<i>access_type</i>	- access type to the process variable:
R_	- read-only,
W_	- write-only,
RW_	- read/write,
<i>variable_type</i>	- process variable type:
BY	- variable of the byte type,
WB	- variable of the 16-bit number type,
DW	- variable of the 32-bit number type,
<i>tel</i>	- telegram no.,
<i>index</i>	- index within the telegram.

### EXAMPLE

X1, byte no 2 of telegram 31,R_BY31.2,	NONE, 1, 1, NOTHING_BYTE
X2, word no 3 of telegram 31,R_WD31.3,	NONE, 1, 1, NOTHING
X3, burner state, RW_BY31.1.35.3,	NONE, 1, 1, NOTHING_BYTE
X4, valve setting,RW_WD32.1.34.1,	NONE, 1, 1, NOTING

Value of the variable X3 is transferred to the Asix system by means of the byte no. 1 of the telegram no. 31. The transfer of the variable X3 consists in sending from Asix a telegram no. 35, the byte no. 3 of which includes the required state of the variable X3.

## Driver Configuration

CAN\_AC\_PCI driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **CAN\_AC\_PCI** section.

- ☑ **Section name:** CAN\_AC\_PCI
- ☑ **Option name:** TRANSMISSION\_SPEED
- ☑ **Option value:** interface\_no,baud\_id

Meaning - the item is used to declare a transmission speed in the CAN network.

Default value - by default, the transmission speed is assumed to be 1 MB.

Parameter:

*interface\_no* - interface no. of the CAN\_AC card (for the CAN\_AC1 card always 1),

*baud\_id* - identifier of transmission speed in the CAN network:

- 1 - 1 MB
- 2 - 500 kB
- 3 - 250 kB
- 4 - 125 kB
- 5 - 100 kB
- 6 - 50 kB
- 7 - 20 kB

#### EXAMPLE

An example of declaration of transmission speed of 20 kB (the CAN network numbered 1):

*Section name:* CAN\_AC\_PCI  
*Option name:* TRANSMISSION\_SPEED  
*Option value:* 1,7

- ☑ **Section name:** CAN\_AC\_PCI
- ☑ **Option name:** REFRESH\_CYCLE
- ☑ **Option value:** number

Meaning - the item is used to declare a time interval between two successive signals allowing the CAN\_AC card driver to read data from the CAN network.

Default value - by default, the CAN\_AC\_PCI driver reads data every 0.5 second.

Parameter:

*number* - number of 0.5-second intervals, which must pass between two successive signals allowing the CAN\_AC board driver to read data from the CAN network.

#### EXAMPLE

An example of declaration of data reading every 1 second:

*Section name:* CAN\_AC\_PCI  
*Option name:* REFRESH\_CYCLE  
*Option value:* 2

- ☑ **Section name:** CAN\_AC\_PCI
- ☑ **Option name:** NETWORK\_CONTROL
- ☑ **Option value:** number

Meaning - the item allows to test the reception of telegrams from the CAN network. It defines the maximal time (in seconds) between reception of successive telegrams with the same number. In case of exceeding this time the process variables from such telegram will be provided with an error status. If additionally in the same time any telegram wasn't received from the CAN network the message about a lack of telegrams in the network is generated in 'Control Panel'.

Default value - by default, the CAN\_AC\_PCI driver doesn't check reception of telegrams.

Parameter:

*number* - maximal number of seconds, which may pass between successive telegrams of the same number.

**EXAMPLE**

An example of checking the reception of telegrams every 5 seconds:

*Section name:* CAN\_AC\_PCI

*Option name:* NETWORK\_CONTROL

*Option value:* 5

- Section name:** CAN\_AC\_PCI
- Option name:** TELEGRAM\_TRACE
- Option value:** YES/NO

Meaning - the item controls transferring to the operator panel the messages about telegrams that have been received from the CAN network. A message includes the number of CAN network, the number of telegram, number of bytes and telegram contents in hexadecimal form.

Default value - by default, the contents of telegrams are not displayed.

- Section name:** CAN\_AC\_PCI
- Option name:** CONTROL\_TRACE
- Option value:** YES/NO

Meaning - the item controls transferring to the operator panel the messages about control telegrams that have been sent from the Asix system computer to controllers. A message includes the number of CAN network, the number of telegram, number of bytes and telegram contents in hexadecimal form.

Default value - by default, the contents of telegrams are not displayed.

- Section name:** CAN\_AC\_PCI
- Option name:** LOG\_FILE
- Option value:** file\_name

Meaning - the item allows to define a file to which all messages, describing the telegrams received from the CAN network, will be written. If LOG\_FILE does not define the full path, then the log file will be created in the current directory. The log file should be used only while the Asix start-up.

Default value - by default, the log file is not created.

- Section name:** CAN\_AC\_PCI
- Option name:** USE\_CONTROL\_VALUE
- Option value:** YES/NO

Meaning - the driver has two pools of telegrams: sending and receiving ones. The sending telegrams are used by Asix to send control data whereas the receiving telegrams contain actual copies of telegrams sent from controllers and are a source of values of process variables for Asix. USE\_CONTROL\_VALUE allows to copy a value of the process variable (of W type) from a sending telegram directly to a buffer of the receiving telegram. The copying concerns the receiving telegram with the same number as the sending telegram assigned to the control variable and is executed only after that have done the control operation correctly. In this way the values of process variables are used by driver as actual values of process variables. This state lasts up to the moment the real values of process variables under consideration will be read from a controller. The CONTROL\_VAR\_CHECK item allows to change the copying mode of control values.

Default value - by default, the control values are not copied to the buffers of receiving telegrams of the driver.

## Checking Control Variables

The declaration of W\_ type variables (control variables) are checked by default. It is possible to use only one variable of this type in one telegram.

- Section name:** CAN\_AC\_PCI
- Option name:** CONTROL\_VAR\_CHECK
- Option value:** YES/NO

Meaning - the item permits to change default settings and enables using telegram to send control value by means of more than one variable of W\_ type. Individual control values are send sequentially, i.e. by sending a separate telegram transferring a value of one control variable only, the other parts of the telegram are filled in with zeroes.

## Change of Default Settings for Updating Mode of Receiving Buffers by Means of Control Variables

- Section name:** CAN\_AC\_PCI
- Option name:** USE\_RW\_DECLARATION
- Option value:** YES/NO

Meaning - the item allows to change the mode of copying control variables to buffers of driver receiving telegrams. The item is import for W\_ type variables only, and is effective if it is used together with the CONTROL\_VAR\_CHECK=YES item. The result of using the item under consideration is copying the control variable to the buffer of receiving telegram (with the number specified in the process variable declaration in the position of telegram that is used to read the variable value).

Default value - by default, the driver buffers are not updated according to declarations of RW\_ type variables.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

## Communication Drivers

- computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.15 CANOPEN - Driver of CANBUS Protocol for PCI 712 NT Card

### Driver Use

The CANOPEN driver is used for data exchange between SELECONTROL MAS PLCs of Selectron Lyss AG and Asix system computers by using the CAN network. The Asix system computers must be provided with the PCI\_712 NT communication processor board and the PCI712 CanLib32 software package of Selectron Lyss AG.

Parameterization of CANOPEN driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the CANOPEN driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: CANOPEN

**CANOPEN** tab:

*Channel parameters*:  
*card\_no*

where:

*card\_no* - PCI\_712 NT card no., by means of which the transmission with the CAN network is executed. In the present version the CANOPEN driver can operate with one PCI\_712 NT board.

The CANOPEN driver is loaded as a DLL automatically.

### Addressing the Process Variable

Values of process variables are transferred in telegrams sent by controllers connected to the CAN network. Each telegram consists of at most 8 bytes, which can be identified as:

bits with indexes 1 - 8 (type BY),  
 16-bit numbers with indexes 1 - 4 (type WD),  
 32-bit numbers with indexes 1 - 2 (type DW).

The CANOPEN driver distinguishes the following types of access to process variables:

read-only (type R\_),  
 write-only (type W\_),  
 read/write (type RW\_).

Addressing the process variables consists in indication of:

- access type (R\_, W\_ or RW\_);
- variable type (BY, WD, DW);

- telegram no. (for the variables with RW\_ access type it is the number of telegram used to read the variable);
- index within the telegram (for the variables with RW\_ access type it is the index of telegram used to read the variable);
- for the variables of RW\_ access type it is necessary to declare additionally:
  - number of telegram used to write the variable,
  - index of telegram used to write the variable.

The syntax of symbolic address which is used for variables belonging to the CANOPEN driver channel is as follows:

`<access_type><variable_type><tel>.<index>[.<tel>.<index>]`

where:

- |                      |  |
|----------------------|--|
| <i>access_type</i>   | - type of access to process variables: |
| R_                   | - read-only,                           |
| W_                   | - write-only,                          |
| RW_                  | - read/write,                          |
| <i>variable_type</i> | - type of process variables:           |
| BY                   | - variable of byte type,               |
| WB                   | - variable of 16-bit number type,      |
| DW                   | - variable of 32-bit number type,      |
| Tel                  | - telegram no.,                        |
| Index                | - index within the telegram.           |

#### EXAMPLE

```
X1, byte no 2 of telegram 31, R_BY31.2, NONE, 1, 1, NOTHING_BYTE
X2, word no. 3 of telegram 31, R_WD31.3, NONE, 1, 1, NOTHING
X3, state of burners,RW_BY31.1.35.3, NONE, 1, 1, NOTHING_BYTE
X4, valve setting,RW_WD32.1.34.1, NONE, 1, 1, NOTHING
```

The X3 variable value is transferred to Asix in byte no. 3 of the telegram 31. The setting of X3 variable consists in sending from Asix the telegram no. 34, the byte no. 3 of which contains required value of X3 variable.

## Driver Configuration

CANOPEN driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **CANOPEN** section.

**Section name: CANOPEN**

**Option name: TRANSMISSION\_SPEED**

**Option value: network\_no,baud\_id**

Meaning - the item is used to declare the transmission speed in the CAN network.

Default value - by default, the transmission speed is assumed to be 1 MB.

Parameter:

network\_no - CAN network no. (in the present version always the network no. 1),

baud\_id - identifier of transmission speed in the CAN network:

- |   |          |
|---|----------|
| 1 | - 1 MB   |
| 2 | - 500 kB |
| 3 | - 250 kB |
| 4 | - 125 kB |
| 5 | - 100 kB |

6 - 50 kB  
7 - 20 kB

### EXAMPLE

An example of declaration of transmission speed of 20 kB (the CAN network numbered 1):

*Section name:* CANOPEN

*Option name:* TRANSMISSION\_SPEED

*Option value:* 1,7

### Frequency of Data Reading from PCI\_712 NT Card

***Section name:*** CANOPEN

***Option name:*** REFRESH\_CYCLE

***Option value:*** *number*

Meaning - the item is used to declare a time interval between successive signals allowing the PCI\_712 NT card driver to generate information on messages received from the CAN network.

Default value - by default, the CANOPEN driver send signals every 0.5 sec.

*number* - equal to 0.5-second intervals, which must pass between successive signals allowing the PCI\_712 NT card driver to generate information on messages received from the CAN network.

### EXAMPLE

The declaration of sending a permission signal every 1 sec:

*Section name:* CANOPEN

*Option name:* REFRESH\_CYCLE

*Option value:* 2

### Checking Reception of Telegrams from CAN Network CAN

***Section name:*** CANOPEN

***Option name:*** NETWORK\_CONTROL

***Option value:*** *number*

Meaning - the item allows to test reception of telegrams from the CAN network. It defines the maximal time (in seconds) between receptions of successive telegrams with the same number. In case of exceeding this time the process variables bound with such telegram will be provided with an error status. If additionally in the same time any telegram wasn't received from the CAN network a message about a lack of telegrams in network is generated in '*Control Panel*'.

Default value - by default, the CANOPEN driver doesn't check reception of telegrams.  
*number* - maximal number of seconds which may pass between successive telegrams with the same number.

### EXAMPLE

An example of checking reception of telegrams every 5 seconds:

*Section name:* CANOPEN

*Option name:* NETWORK\_CONTROL

*Option value:* 5

### Tracing Telegrams Received from CAN Network

- Section name:** CANOPEN
- Option name:** TELEGRAM\_TRACE
- Option value:** YES/NO

Meaning - the item controls transferring to the operator panel the messages about telegrams that have been received from the CAN network. A message includes the number of CAN network, the number of telegram, number of bytes and telegram content in hexadecimal form.

Default value - by default, the contents of telegrams are not displayed.

#### EXAMPLE

The declaration of tracing the received telegrams:

*Section name:* CANOPEN  
*Option name:* TELEGRAM\_TRACE  
*Option value:* YES

### Tracing of Control Telegrams

- Section name:** CANOPEN
- Option name:** CONTROL\_TRACE
- Option value:** YES/NO

Meaning - the item controls transferring to the operator panel the messages about control telegrams that have been sent from the Asix system computer to controllers. A message includes the number of CAN network, the number of telegram, number of bytes and telegram contents in hexadecimal form.

Default value - by default, the contents of telegrams are not displayed.

#### EXAMPLE

The declaration of tracing the control telegrams:

*Section name:* CANOPEN  
*Option name:* CONTROL\_TRACE  
*Option value:* YES

- Section name:** CANOPEN
- Option name:** LOG\_FILE
- Option value:** *file\_name*

Meaning - the item allows to define a file to which all the messages, describing telegrams received from the CAN network, will be written. If LOG\_FILE does not define the full path, a log file will be created in the current directory. The log file should be used only while the Asix system start-up.

Default value - by default, the log file is not created.

#### EXAMPLE

*Section name:* CANOPEN  
*Option name:* LOG\_FILE  
*Option value:* D:\asix\CAN.LOG

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

- Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.
- Option value:  
 YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

- Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.
- Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

- Purpose - with this option deployed, local control in a channel will be disabled.
- Option value:  
 YES / NO
- The default value - NO.

### **Data Validity Period**

- Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.
- Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.
- The default value - by default, no time for data validity is set.

## 1.16 CipAB - Driver for Communication with Logix5000 Series Controllers by Allen-Bradley

### Driver Use

The CtCipAB driver is used to exchange data between the Asix system and Logix5000 series controllers by Allen-Bradley using EtherNet/IP protocol in Explicit Messaging mode. The driver requires installation of Asmen.dll module, version 5.9 or higher.

### Declaration of Transmission Channel

Declaration of the transmission channel operating according to the CtCipAB driver protocol requires adding a channel with the following parameters to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* CipAB

**CipAB/Channel Parameters** tab:

<b>IP Address of the Device</b>	- IP Address of the device,
<b>Alarm Number</b>	- (optionally) alarm number in the absence of communication with the controller.

#### EXAMPLE

Channel declaration:  
*Name:* CHANNEL  
*Driver:* CtCipAB  
*Channel Parameters:*  
*IP Address of the Device:* 10.10.105.21  
*Alarm Number:* 100

### Declaration of Variables

The symbolic address of a variable is based on the tag name in the PLC software. In case of symbolic addresses referencing to global tags, only the tag name is specified; in case of symbolic addresses referencing to local tags, the guideword Program should be specified: local program name and the tag name.

The driver provides access to the tags of one of the below types:

Boolean  
8-Bit Int  
16-Bit Int  
32-Bit Int  
32-Bit Unsigned  
32-Bit Float  
String

Examples:

*Name:* G\_01  
*Description:* global tag GBOOL1 of Boolean type  
*Address:* GBOOL1  
*Channel:* CHANNEL1  
*Element Count:* 1  
*Sample Rate:* 1  
*Conversion Function:* NOTHING

*Name:* G\_02  
*Description:* global tag GDINT of Dword type  
*Address:* GDINT  
*Channel:* CHANNEL1  
*Element Count:* 1  
*Sample Rate:* 1  
*Conversion Function:* NOTHING\_DW

*Name:* G\_03  
*Description:* global tag GFLOAT of Float type  
*Address:* GFLOAT  
*Channel:* CHANNEL1  
*Element Count:* 1  
*Sample Rate:* 1  
*Conversion Function:* NOTHING\_FP

*Name:* G\_04  
*Description:* global tag GSTRING10 of String type  
*Address:* GSTRING10  
*Channel:* CHANNEL1  
*Element Count:* 10  
*Sample Rate:* 1  
*Conversion Function:* NOTHING\_TEXT

*Name:* L\_01  
*Description:* local tag Prg1.TINT of Float type  
*Address:* Program:Prg1.TINT  
*Channel:* CHANNEL1  
*Element Count:* 11  
*Sample Rate:* 1  
*Conversion Function:* NOTHING\_FP

*Name:* L\_02  
*Description:* local tag Prg2.TBOOL of Boolean type  
*Address:* Program:Prg2.TBOOL  
*Channel:* CHANNEL1  
*Element Count:* 11  
*Sample Rate:* 1  
*Conversion Function:* NOTHING

It is also possible to access array elements:

*Name:* G\_05  
*Description:* global tag GTABINT[1] of Int type  
*Address:* GTABINT[1]  
*Channel:* CHANNEL1  
*Element Count:* 1  
*Sample Rate:* 1  
*Conversion Function:* NOTHING

fragments or whole tables:

## Communication Drivers

*Name:* G\_06  
*Description:* 10 first elements of GTABFP table  
*Address:* GTABFP[0]  
*Channel:* CHANNEL1  
*Element Count:* 10  
*Sample Rate:* 1  
*Conversion Function:* NOTHING\_FP

and access to structural elements:

*Name:* G\_07  
*Description:* access to field ACC of TIMER structure  
*Address:* GTIMER.ACC  
*Channel:* CHANNEL1  
*Element Count:* 1  
*Sample Rate:* 1  
*Conversion Function:* NOTHING\_DW

## Fast Variable Refresh Rate

The variables with the **Fast variable refresh rate** attribute assigned are refreshed by the driver in line with the period specified by the **Fast refresh period** option. The variable values read out in this mode are entered directly into the Asmen cache, thus allowing for fast refresh of the application's visual elements.

Fast refresh rate is used only by the application's visual elements.

## Communication Testing

The **CipAB / Driver Parameters - UniDriver** tab enables to specify the parameters referring to detailed error and diagnostic information. This information is saved in the *UniDriver.<current data>.log* file.

The parameters present on the *Driver Parameters - UniDriver* tabs refer to all the channels having such a tab in their driver declaration. The parameters set in the *Driver Parameters - UniDriver* tab in the definition of one of these channels are automatically displayed in the *Driver Parameters - UniDriver* tabs in the definitions of the other channels.

### **Path to the directory in which the log file will be created**

Purpose - if this option exists, then the log file is inserted in the specified directory.

Default value - by default the log file is inserted in the Asix system directory.

Parameter:

*path\_to\_directory* - path to the directory in which the log file will be created.

### **Traced Variables**

Purpose - for each variable whose name is in the list, information about its value, quality and stamp will be entered in the log file while processing this variable.

The default value - none.

Parameter:

*variable\_list* - list of variable names in the Asix system, separated by commas.

## Driver Configuration

The driver is configured using options present on the tabs: **CipAB / Driver Parameters**. These options are used for declaring the following:

- creating a log file,
- the log file size,
- log of telegrams,
- reception timeout,
- IP address of the computer card, with the use of which controllers are operated,
- fast refresh period.

### **Log File**

Purpose: The text log file to which driver status messages are stored is used for diagnostic purposes.

Option value:

*log\_file\_name*

The default value: By default, the log file is not created.

### **Log File Size**

Purpose: This option is used to determine the log file size defined using the *Log file* option.

Option value:

*number*

The default value: - the size of the log file in MB.  
By default the log file size is 10 MB.

### **Log of Telegrams**

Purpose: This option allows writing the contents of telegrams communicated between the driver and the controllers to the log file. The item in question should only be used during the Asix system start-up.

Option value:

YES | NO

The default value: By default the value of the item is NO.

### **Response Timeout**

Purpose: This option specifies the maximum time throughout which the driver waits for a response from the controller.

Option value:

*number*

The default value: - time in milliseconds.  
The default value is 500 (milliseconds).

**IP Address of the Computer**

Purpose: This option is used to specify the IP address of the computer's network card, with the use of which all driver channels communicate with the controllers.

Option value:

*IP\_address* - network card address in IPv4 notation.  
*The default value:* If this option is not defined, the system network card will be used.

**Fast Refresh Period**

Purpose: used to define the refresh period for the variables with the *Fast Refresh* attribute assigned.

Option value:

*number\_ms* - fast refresh period (from 100 to 990).  
The default value: The default value is 200 (milliseconds).

EXAMPLE

Driver parameterization:

*Driver:* CipAB  
*Log File=*d:\tmp\CtCipAB\ec1.log  
*Log File Size=*20  
*Log of Telegrams=*YES  
*IP Address of the Computer:* 10.10.110.24  
*Fast Refresh Period:* 250

## Channel Parameters

The channel is configured using options present on the tabs: **CipAB / Channel Parameters 2**. These options are used for declaring the following:

- creating a log file,
- the log file size,
- log of telegrams,
- reception timeout,
- IP address of the computer card, with the use of which controllers are operated,
- fast refresh period.

**Log File**

Purpose: The text log file to which the contents of telegrams sent and received in a given channel are stored is used for diagnostic purposes.

Option value:

*log\_file\_name*

**Log File Size**

Purpose: This option is used to determine the log file size defined using the *Log file* option.

Option value:  
*number* - the size of the log file in MB.

**Log of Telegrams**

Purpose: This option allows writing the contents of telegrams communicated in a given channel to the log file. The item value overrides the setting specified by the item of the same name present in the driver section. The item in question should only be used during the Asix system start-up.

Option value:  
YES | NO

**Response Timeout**

Purpose: This option specifies the maximum time throughout which a given channel waits for a response from the controller. The item value overrides the setting specified by the item of the same name present in the driver section.

Option value:  
*number* - time in milliseconds.

**IP Address of the Computer**

Purpose: This option is used to specify the IP address of the computer's network card, with the use of which the channel communicates with the controller. The item value overrides the setting specified by the item of the same name present in the driver section.

Option value:  
*IP\_address* - network card address in IPv4 notation.

**Fast Refresh Period**

Purpose: this option is used to define the refresh period for the variables with the *Fast Refresh* attribute assigned. The item value overrides the setting specified by the item of the same name present in the driver section.

Option value:  
*ms\_number* - fast refresh period (from 100 to 990).  
The default value: The default value is 200 (milliseconds).

**EXAMPLE**

Channel parameterization:

*Channel Name:* CHANNEL1  
*Log File=*d:\logi\kanal1.log  
*Log File Size=*20  
*Log of Telegrams=*YES  
*IP Address of the Computer:* 10.10.115.25  
*Response Timeout:* 750  
*Fast Refresh Period:* 250

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:  
YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.17 COMLI - Driver of COMLI Protocol

### Driver Use

The driver is designed for data exchange between the Asix system and SattConxx and other ABB PLCs supporting the COMLI (**COM**munication **LI**nk) protocol. The data exchange is performed by means of a serial interface in the RS-232 or RS-485 standard.

In the present driver version the following functions of COMLI protocol are implemented:

- Transfer individual I/O bits,
- Transfer I/O bits or a register,
- Request individual I/O bits,
- Request several I/O bits or registers,
- Transfer date and time,
- Acknowledge.

Parameterization of COMLI driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the COMLI driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* COMLI

**COMLI** tab:

**Channel parameters:**

*slave\_no, port [ , baud] [ ,parity] [ ,data\_type]*

where:

*slave\_no* - slave no. assigned to the PLC;  
*port* - name of the serial port through which the connection with the PLC will be executed;  
*baud* - option: transmission speed; by default 9600;  
*parity* - option: parity check; by default ODD;  
*data\_type* - option: representation of ASCII data (ASCII) or binary (BINARY); by default it is BINARY.

#### EXAMPLE

Below there is an example of declaration of transmission channel named CHANNEL, that is used for communication with the controller no. 3 through the port COM2 in default mode, i.e. 9600 Bd, 8 bit character, check parity ODD, binary representation of data:

*Channel / Name:* CHANNEL  
*Driver:* COMLI  
*Channel parameters:* 3, COM2

## Types of Process Variables

In the driver the following types of process variables are defined:

<b>IO</b>	- I/O states,
<b>RG</b>	- values of registers,
<b>TM</b>	- writing date and time to a PLC.

The values of **IO** and **RG** type variables may be read and written whereas the values of **TM** type variable may only be written.

## Addressing the Process Variables

The syntax of symbolic address which is used for variables belonging to the COMLI driver channel is as follows:

*<Type><Index>*

where:

<i>Type</i>	- name of variable type,
<i>Index</i>	- address of the variable within the <i>Type</i> type of the variable.

Ranges of indexes are following:

type <b>IO</b> :	0 - 16383,
type <b>REG</b> :	0 - 3071,
type <b>TM</b> :	no index is given.

### EXAMPLE

Examples of declaration of process variables:

```
JJ_1, state I/O number 1,IO1,CHANNEL,1,1,NOTHING  
JJ_2, register number 10,RG10,CHANNEL,1,1,NOTHING  
JJ_40, writing date and time every 1 minute, TM, CHANNEL,12,60,NOTHING_BYTE
```

## Synchronization of Date and Time with the Controller

In the driver a mechanism of synchronization of date and time between Asix and PLCs is included.

The synchronization is activated for each transmission channel separately with use of option *Time\_synchronization* declared in Architect in *Miscellaneous*, on *Directly entered options* tab:

Section name: ASMEN  
Option name: TIME\_SYNCHRONIZATION  
Option value: *channel,variable*

where:

<i>channel</i>	- name of the channel used to communicate with a PLC.
<i>variable</i>	- name of an ASMEN variable belonging to the channel <i>channel</i> and using for synchronization of date and time.

The synchronization of date and time consists in writing to a PLC a frame containing the current date and time of Asix. The frame is written by using the function for writing the date and time of the COMLI protocol according to the frequency assigned to the *variable*. The variable type must be the type **TM** (supporting date and time), the number of elements

assigned to the *variable* must be 12 (the size of frame of date and time). The function NOTHING\_BYTE must be used as a calculation function.

### EXAMPLE

An example of the definition of time synchronization every 1 minute for the channel CHANNEL by using the variable SYNCHRO is given below.

Channel declaration:

*Channel / Name:* CHANNEL  
*Driver:* COMLI  
*Channel parameters:* 2, COM1

Declaration of the variable SYNCHRO:

*Section name:* ASMEN  
*Option name:* DATA  
*Option value:* COMLI.DAT

The declaration of the variable SYNCHRO is found in the file *COMLI.DAT* and has the following form:

SYNCHRO, synchronization of date and time, TM, CHANNEL,12, 60, NOTHING\_BYTE

Synchronization parameters:

*Variable:* SYNCHRO  
*Description:* synchronization of date and time  
*Address:* TM  
*Channel:* CHANNEL  
*Elements Count:* 12  
*Sample Rate:* 60  
*Conversion Function:* NOTHING\_BYTE

## Driver Configuration

COMLI driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **COMLI** section.

- Section name:** COMLI
- Option name:** RECV\_TIMEOUT
- Option value:** slave\_no, number

Meaning - for each slave it is possible to define the maximal time, which may elapse between sending query and receiving response (so-called receiving timeout).

Default value - by default, the item has a value of 2000.

Parameter:

- slave\_no* - slave no. assigned to the controller.
- number* - value of timeout in milliseconds.

- Section name: COMLI**
- Option name: LOG\_FILE**
- Option value: file\_name**

Meaning - the item allows to define a file to which all the diagnostic messages of the driver and the information about the content of telegrams sent/received by the driver will be written. If the item doesn't define a full path the log file will be created in the current directory. The log file should be used only while the Asix start-up.

Default value - by default, the log file is not created.

- Section name: COMLI**
- Option name: LOG\_FILE\_SIZE**
- Option value: number**

Meaning - the item allows to define the size of the log file in MB.

Default value - the default size of the log file amounts 1 MB.

Parameter:  
    *number* - size of the log file in MB

- Section name: COMLI**
- Option name: LOG\_OF\_TELEGRAMS**
- Option value: YES/NO**

Meaning - the item allows to write the content of telegrams sent/received by the driver to the log file (declared by means of the item LOG\_FILE). Writing the content of telegrams should be used only while the Asix start-up.

Default value - by default, the driver does not write the content of telegrams to the log file.

- Section name: COMLI**
- Option name: NUMBER\_OF\_REPETITIONS**
- Option value: number**

Meaning - the item allows to define maximal number of trials to do the command in case of transmission errors.

Default value - by default, max. 3 repetitions are executed.

Parameter:  
    *number* - number of repetitions.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

- Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:  
    YES / NO  
    *computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**☑ Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**☑ Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
 YES / NO

The default value - NO.

**☑ Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.18 CPIII - Driver to Communicate with Control Panels CP-III/E Used to Control Compressors of MYCOM (MAYEKAWA)

### Driver Use

Driver is used to exchange data between the Asix system and control panels CPIII/E used to control compressors of MYCOM (MAYEKAWA). Communication is performed using a serial communication in RS-485 standard.

The drivers realizes the following functions:

- readout of current analog and binary values,
- readout and write of setpoints,
- readout of alarms and declaring them to the Asix alarm system.

Parameterization of CPIII driver is implemented using the Architect program.

### Declaration of Transmission Channel

Declaration of transmission channel operating according to the protocol of driver CPIII requires to add to the module *Current data* the following parameters:

**Standard** tab:

*Name:* logical name of the transmission channel

*Driver:* CPIII

**CPIII / Channel parameters** tab:

**Identification:**

*Address of a device in a network;*

**Port:**

*Port;*

**Transmission parameters:**

*Transmission speed in bauds;*

*Word length;*

*Parity checking (n-none, e-even parity, o-odd parity, m-mark, s-space);*

*Stop bits.*

If the transmission parameters are omitted, the following values are default: 9600,7,e,1

**Alarms base:**

*Number of first alarm*

**Global alarms:**

*The way of transferring alarms read from remote devices to the alarm system of the Asix system*

**An exemplary channel declaration:**

Driver: CPIII

Channel parameters:

Address of a device in a network: 10,  
Port: COM1

**Declaration of Process Variables**

Addressing variables of CPIII driver is based on the description of the communication protocol of CPIII device, but value of each variable differs to this in the description of the protocol. In particular, do not use multipliers described in the protocol because the driver performs the relevant operations of multiplication /division.

Table. Declaration of address of CPIII driver variables.

Address	Meaning	Conversion function of NOTHING_XXX type
D1	Analog input 1	NOTHING_FP
D2	Analog input 2	NOTHING_FP
D3	Analog input 3	NOTHING_FP
D4	Analog input 4	NOTHING_FP
D5	Analog input 5	NOTHING_FP
D6	Analog input 6	NOTHING_FP
D7	Analog input 7	NOTHING_FP
D8	Analog input 8	NOTHING_FP
D9	Analog input 9	NOTHING_FP
D10	Analog input 10	NOTHING_FP
D11	Analog input 11	NOTHING_FP
D12	Analog input 12	NOTHING_FP
D13	Analog input 13	NOTHING_FP
D14	Analog input 14	NOTHING_FP
D15	Analog input 15	NOTHING_FP
D16	Analog input 16	NOTHING_FP
D17	Operation time in hours	NOTHING_DW
D18	like D17	NOTHING_DW
D19	Capacity control manipulated value (MV), calculated by PID controller, wanted slide valve position.	NOTHING_FP
D20	Superheat process value (PV); actual value of evaporator superheat in case of temperature control. example: chiller unit equipped with linear expansion valve.	NOTHING_FP

Communication Drivers

D21	Superheat manipulated value (MV), calculated by PID controller.	NOTHING _FP
D22	Liquid injection process value PV.	NOTHING _FP
D23	Liquid injection manipulated value (MV) , calculated by PID controller.	NOTHING _FP
D24	fixed data : 000000	NOTHING _FP
D25	Setpoint unit start, cut-in value for controller. (Equivalent of D1 in the command of setpoint entry for communication protocol of CPIII device).	NOTHING _FP
D26	Setpoint unit stop, cut-out value for controller. (Equivalent of D2 in the command of setpoint entry for communication protocol of CPIII device).	NOTHING _FP
D27	Capacity control setpoint. (Equivalent of D3 in the command of setpoint entry for communication protocol of CPIII device).	NOTHING _FP
D28	Setpoint superheat control - only applicable in case of temperature control. (Equivalent of D4 in the command of setpoint entry for communication protocol of CPIII device)	NOTHING _FP
D29	Setpoint Liquid Injection, setpoint for liquid injection oil cooling. (Equivalent of D5 in the command of setpoint entry for communication protocol of CPIII device).	NOTHING _FP
D30	fixed data : 000000	NOTHING _FP
D31	Current mode: 10 : manual mode 15 : manual mode + datacomms on 20 : auto 25 : auto mode + datacomms on 30 : remote mode 35 : remote mode + datacomms on 40 : remote - auto mode 45 : remote -auto mode + datacomms on	NOTHING _BYTE
D31.n	Current mode divided into individual elements: D31.0 - has the value of 1 for automatic mode, and 0 for manual D31.1 - has the value of 1 for "datacomms" and 0 in other case D31.2 - has the value of 1 for remote mode ("remote") and 0 in other case	NOTHING _BYTE
D32	Digital inputs	NOTHING _DW
D32.n	A single bit of digital input "n" can take values form 0 to 31. It corresponds to digital inputs from 1 to 32, e.g. D32.0 corresponds to digital input of 1. The variable takes the value of 1 if the digital input is set, and 0 otherwise	NOTHING _BYTE
D33	The status of all alarms	NOTHING _DW
D33.n	The status of a single alarm. The variable takes the value of 1 if the alarm is active, and 0 otherwise. The number "n" identifies a single alarm and is the	NOTHING _BYTE

	<p>number of bit in status word of alarms in the range of 0-31.</p> <p>Examples:</p> <p>D33.0 - "start fail" (the word of alarm status 00000001)</p> <p>D33.1 - "oil pressure low" (the word of alarm status 00000002)</p> <p>D33.2 - "discharge pressure high" (the word of alarm status 00000004)</p> <p>D33.3 - "discharge temperature high" (the word of alarm status 00000008)</p> <p>itd.</p>	
D100	<p>Special controls.</p> <p>The variable takes the value of:</p> <p>0 - "remote start" OFF &amp; "100% lock" OFF</p> <p>1 - "remote start" OFF &amp; "100% lock" ON</p> <p>2 - "remote start" ON &amp; "100% lock" OFF</p> <p>3 - "remote start" ON &amp; "100% lock" ON</p> <p>(Equivalent of D6 in the order of setpoints entry of communication protocol of CPIII device).</p>	NOTHING_BYTE
D100.n	<p>Special control by individual elements.</p> <p>D100.0 - "100% lock"</p> <p>D100.1 - "remote start"</p> <p>OFF corresponds to 0 and ON corresponds to 1.</p>	NOTHING_BYTE

## Alarm Generation

The driver of CPIII device with the interval not longer than the value determined by *Event check period* checks the state of lists of device events and generates alarms for all new found events. Alarm check is done for each data readout from the device, also when data is read from the device to refresh variable values. So alarm check may be done more frequently than it is determined by *Event check period* - when there are variables which refresh period is shorter than the value of this parameter. The number of Asix system alarm is created as the sum of bit number in the status word of CPIII device alarms and the value determined by the parameter *Alarms base*. If there is no *Alarms base* declaration the alarms will not be generated.

Generating an alarm, the driver transfers the address of CPIII device as the parameter of this alarm. This address may be used in the text associated with the given alarm.

### Example:

If the parameter *Alarms base* has the value of 5 and the alarm 'start fail' appears (the mask of which has the value of 00000001 (bit no. 0)), the alarm no. 5 will be generated. Simultaneously, the variable with address D33.0 will take the value of 1.

## Parameters of Channel / CPIO Driver

The driver configuration is defined in the *Current Data* module, in the channel operating according to CPIO driver, on tabs: *Channel parameters*, *Driver parameters*.

Parameters located on tabs of driver parameters applies to all channels with the same driver declared. In the case of several channels operating on the same driver - parameters set on tabs *Driver parameters* for one channel are automatically displayed on the same tabs for the rest of channels.

Parameters located on tabs of channel parameters applies to one channel for a specific device.

### **Channel parameters 2** tab:

#### **Event check period**

Meaning - interval between checking the state of events and generating alarms.  
 Default value - 30  
 Parameter:  
*number* - number in seconds.

#### **Log file**

Meaning - the item allows to define a file to which all the diagnostic messages of the driver and the information about the content of telegrams sent/received by the driver will be written. If the item doesn't define a full path the log file will be created in the current directory. The log file should be used only while the Asix start-up.  
 Default value - by default, the log file is not created.  
 Parameter:  
*Log file* - nazwa pliku logu;  
*Maximum size of log file* - number in MB; 0 - unlimited;

#### **Retries**

Meaning - number of retries in case of transmission error.  
 Default value - 3  
 Parameter:  
*number* - positive number.

#### **Timeout**

Meaning - maximum time of waiting for a device response.  
 Default value - 500  
 Parameter:  
*number* - time in milliseconds;

#### **Special control**

Meaning - parameter specifies a default value for the D100 variable in the situation when the driver has got a first command to write the variable from the range of D25-D29. The values of all the variables for write - i.e. D25-D29 and D100 are written simultaneously during one operation of transmission. Before that operation the driver reads values of the variables D25-D29 and sends them to a device in unchanged form (except the variable the value of which is recommended to be changed). Only the value of demanded variable is changed. It is not possible to read the previous value of the variable D100 - but some variable has to be sent to a device - so, the parameter *Special control* specifies the value of D100 in above case. The parameter does not have sense if the first

operation of writing concerns the variable D100, i.e. when the user specified its value before changing values of other variables.

Default value - 0

**Pressure control**

Meaning - parameter specifies the multiplier used for reading/writing the variables D25, D26 and D27. For 'Yes' declaration the multiplier takes the value of 100 - for 'No' declaration the multiplier takes the value of 10.

Default value - 'No'

**Driver parameters** tab:

**Alarms base**

Meaning - parameter allows to specify the way of alarm numbering. After adding this parameter to the number of bit in the alarm status word - the number of alarm in Asix system is got.

Default value - -1 (no alarm handling).

Parameter:

*number*

- positive number.

**Global alarms**

Meaning - parameter controls the way of transmission of alarms read from remote devices to the alarm system of Asix.

Default value - by default, alarms are transmitted to the alarm system as global alarms (by the function `AsixAddAlarmGlobalMili()`). Setting the value to NO causes that alarms are transmitted to the alarm system by the function `AsixAddAlarmMili()`.

**Event check period**

Meaning - parameter allows to specify the interval between checking the state of events and alarm generation.

Default value - 30.

Parameter:

*number*

- time in seconds.

**Log file**

Meaning - parameter allows to specify the name of log file and the size of the log file.

Default value - the log file is not created and maximum size of log file by default equals 0 - that means unlimited.

Parameter:

*number*

- time in seconds.

**Driver parameters 2** tab:

**Retries**

Meaning - parameter allows to specify the maximal number of tries to do the command in case of transmission errors.

Default value - 3.

Parameter:

*number*

**Timeout**

Meaning - parameter allows to specify the maximal timeout for the response from an individual device.

Default value - 500.

Parameter:  
*number* - time in milliseconds.

**Special control**

Meaning - parameter specifies a default value for the D100 variable in the situation when the driver has got a first command to write the variable from the range of D25-D29. The values of all the variables for write - i.e. D25-D29 and D100 are written simultaneously during one operation of transmission. Before that operation the driver reads values of the variables D25-D29 and sends them to a device in unchanged form (except the variable the value of which is recommended to be changed). Only the value of demanded variable is changed. It is not possible to read the previous value of the variable D100 - but some variable has to be sent to a device - so, the parameter *Special control* specifies the value of D100 in above case. The parameter does not have sense if the first operation of writing concerns the variable D100, i.e. when the user specified its value before changing values of other variables.

Default value - 0

**Pressure control**

Meaning - parameter specifies the multiplier used for reading/writing the variables D25, D26 and D27. For 'Yes' declaration the multiplier takes the value of 100 - for 'No' declaration the multiplier takes the value of 10.

Default value - 'No'

**Example of driver parameterization**

Declaration of channel:

*Name:* CPIII\_1

*Channel parameters:* 5, COM1,57600,8,none,1

*Name:* CPIII\_2

*Channel parameters:* 5, COM2

*Name:* CPIII\_3

*Channel parameters:* 6, COM2

Declaration of parameters for the channel CPIII:

*Global alarms:* Yes

*Event check period:* 60

Declaration of parameters for the channel CPIII\_2:

*Event check period:* 30

*Global alarms* applies to all devices. *Event check period* is different for the device of CPIII\_2 channel.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.19 CZAZ - Driver for Communication with CZAZ-U and CZAZ-UM

### Driver Use

The driver of serial link RS-485 that allows to exchange data between Asix system and digital protection devices CZAZ-U and CZAZ-UM of ZEG-Energetyka.

Parameterization of CZAZ driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the CZAZ driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: CZAZ

**CZAZ** tab:

**Channel parameters:**  
*address, port, [baud, character, parity, stop]*

where:

<i>address</i>	- device address in network;
<i>port</i>	- serial port name;
<i>baud</i>	- transmission speed in bauds;
<i>character</i>	- number of bits in a transmitted character;
<i>parity</i>	- parity check type (even, odd, none);
<i>stop</i>	- number of stop bits,

**An exemplary channel declaration:**

*Name* (of channel): ASMEN  
*Driver*: CZAZ  
*Channel parameters*: 10, COM1,115200,8,none,1

### Declaration of Process Variables

The syntax of symbolic address which is used for variables belonging to the CZAZ driver channel is as follows:

*object\_id[.element\_nr[bit\_nr]]*

where:

<i>object_id</i>	- hexadecimal identifier of an object according to CZAZ device documentation. It is also possible to use the name of an object instead of the number.
------------------	---

- element\_nr* - The number of element within an object. The first element has the number 0. If *element\_nr* is omitted, it is taken 0.
- bit\_nr* - The number of bit within an element. The first bit has the number 0.

### Example

In the documentation of CZAZ device there is the 0102 object named Signal, which contains three elements: Signal1, Signal2 i Signal3.

Address	Meaning
102	first element of the object - Signal1 (as 32-bit word)
0102	as a/m
Signal	as a/m
102.0	as a/m
Signal.0	as a/m
102.1	second element of the 102 object - Signal2 (as 32-bit word)
102.1.4	Bit named SPZ5 in the second element of 102 object

Variables referring to objects of OB type are 8-bit elements (NOTHING\_BYTE conversion function).

Variables referring to objects of 2B type are 16-bit words (NOTHING conversion function).

Variables referring to objects of 4B type are 32-bit words (NOTHING\_DW conversion function).

Variables referring to individual bits in objects are 8-bit elements (NOTHING\_BYTE conversion function).

Variables referring to elements of ULONG type objects are 32-bit words (NOTHING\_DW conversion function).

Variables referring to elements of UWORD type objects are 16-bit words (NOTHING\_DW conversion function).

The first element of 0110 object (Counter) is 32-bit word (NOTHING\_DW conversion function), and the rest elements are 16-bit words (NOTHING conversion function).

The driver allows access to the first element of 0004 object (Time). It is 32-bit number (NOTHING\_DW conversion function). To display this element on visualization masks it is possible to use the NUMBER object with the format %D. The driver permits to write such a variable into a device and set (in this way) its time. The number of 0004 object milliseconds are unavailable.

The driver allows access to the following objects: 0001, 0002, 0004, 0011, 0012, 0013, 0014, 0100, 0101, 0102, 0105, 0110, 0111, 0112, 0114, 0115, 0119, 011B, 011D, 011E, 0120, 0121, 0125, 0126, 0200, 0201, 0202, 0203, 0204, 0208, 0209, 020A, 020B, 020D, 020E, 020F, 0210, 0211, 0212, 0213, 0221, 0222, 0223, 0224, 0225, 0227, 0228, 0229, 022A, 022C, 022D, 0230, 0231, 0232, 0233, 0234, 0235, 0236, 0300, 0311, 0312, 0313, 0314, 031A, 0320, 0321, 0330, 0400, 0401, 0402, 0500, 0501, 0502 and 0503.

## Generation of Alarms

The driver of CZAZ device with the interval defined by the *Event\_check\_period* parameter checks the state of device event lists and generates alarms for all new detected events. The number of Asix system alarm is created as the sum of the number of CZAZ device event and the value defined by the *Alarms\_base* parameter. The buffer of CZAZ device events can include up to 500 new events. For each event the alarm of event beginning / end is generated. Therefore it is possible to generate 1000 alarms at the same time for each serial link that CZAZ devices are connected to, while the default alarm buffer size of Asix system equals 200. To prevent losing alarms, the buffer size should be set with use of the *INPUT\_BUFFER\_SIZE* parameter of *ALARMS\_SYSTEM* section (The option is not directly supported by Architect module, so to declare it for the application, you should use the *Miscellaneous* module (in Architect), *Directly entered options* tab.).

## Driver Configuration

CZAZ driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The CZAZ driver may be configured using the 'CZAZ' section as well as sections having the same name as channel names. The parameters placed in the 'CZAZ' section apply to all devices and have global meaning. The parameters placed in other sections refer to a specified device. Some parameters can be declared only for the 'CZAZ' section, other can be placed in all sections.

### Example

Channel declaration:

*Name: CZAZ1*  
*Channel parameters: 10, COM1,57600,8,none,1*

*Name: CZAZ2*  
*Channel parameters: 10, COM2,57600,8,none,1*

*Name: CZAZ3*  
*Channel parameters: 10, COM2,57600,8,none,1*

Driver parameters:

*Section name: CZAZ*  
*Option name: Global\_alarms*  
*Option value: yes*

*Section name: CZAZ*  
*Option name: Time\_sync*  
*Option value: 50*

*Section name: CZAZ1*  
*Option name: Time\_sync*  
*Option value: 120*

The *Global\_alarms* parameter refers to all devices. As devices are not individual parameterized, the parameter can be used only in 'CZAZ' section. The *Time\_sync* parameter placed in 'CZAZ' section defines 50 second time interval to control the time of all devices with the exception of CZAZ1 channel device, as in the 'CZAZ1' section one has defined the time interval equaled 120 seconds.

There are some driver parameters below. The 'global parameter' means the parameter declared only in 'CZAZ' section.

**Section name: CZAZ**

**Option name: Alarms\_base**

**Option value: number**

Meaning: allows to specify numeration of alarms for each device.

Default value: -1 (no alarm service).

**Section name: CZAZ**

**Option name: Global\_alarms**

**Option value: yes/no**

Meaning: the item controls the way of transferring alarms read from remote devices to the alarm system of Asix; **global parameter**.

Default value: by default, the alarms are transferred to the alarm system as global alarms (transferred to the alarm system by means of the function AsixAddAlarmGlobalMili()). Setting the value of the item GLOBAL\_ALARMS on NO causes that the alarms are transferred to the alarm system by means of the function AsixAddAlarmMili()).

**Section name: CZAZ**

**Option name: Event\_check\_period**

**Option value: number**

Meaning: the option declares the interval (in seconds) between checking the state of events and alarm generation in two subsequent devices connected to the same serial port; **global parameter**.

Default value: 10

**Section name: CZAZ**

**Option name: Field\_number**

**Option value: number**

Meaning: the option declares a field number for a given device; it is transmitted as an alarm parameter.

Default value: -1

**Section name: CZAZ**

**Option name: Log\_file**

**Option value: file\_name**

Meaning: the item allows to define a file to which all diagnostic messages of CZAZ driver and all messages describing the telegrams received and sent by CZAZ driver will be written; if LOG\_FILE does not define the full path, then the log file will be created in the current directory; the log file should be used only while the Asix start-up; **global parameter**.

Default value: by default, the log file is not created.

**Section name: CZAZ**

**Option name: Transmission\_delay**

**Option value: number**

Meaning: the item allows to determine a time interval (as a multiple of 10 milliseconds) between two successive operations on a communication bus; **global parameter**.

Default value: by default, the item assumes a value of 1 (10 milliseconds).

**Section name: CZAZ**

**Option name: Number\_of\_repetitions**

**Option value: number**

Meaning: the item allows to define maximal number of trials to do the command in case of transmission errors; **global parameter**.

Default value: 3

**Section name: CZAZ**

**Option name: Recv\_timeout**

**Option value: number**

Meaning: allows to specify a waiting time for arriving the first character of an answer sent from a specified device.

Default value: 1000

**Section name: CZAZ**

**Option name: Time\_sync**

**Option value: number**

Meaning: allows to specify a time (in seconds) between successive time readouts from a device. If the parameter equals 0, the time is not read and all variables get a Windows time stamp.

Default value: 600

**Section name: CZAZ**

**Option name: Time\_difference**

**Option value: number**

Meaning: allows to specify the time difference (in milliseconds) between a device time and Windows time. After exceeding the time defined by *Time\_difference*, the time in CZAZ device will be set and equalized with the Windows time. The parameter is taken into consideration only if the *Time\_sync* parameter differs from 0.

Default value: 0 (time in CZAZ device is not changed).

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

 **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

 **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.20 DataPAF - Driver of DataPAF Energy Counter Protocol

### Driver Use

The DataPAF protocol is designed for communication with DataPAF energy counters provided with the EPROM 2.1 13.Aug.96 or later one. For the communication a COM interface is used.

Parameterization of DataPAF driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the DataPAF driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: *DataPAF*

**DataPAF** tab:

**Channel parameters:**

*COMn*

where:

*COMn* - number of the serial port to which the DataPAF energy counter is connected.

Each defined channel may have its own section, the name of which is a logical name of the channel. It contains parameters for the given channel. Some channels may have a common serial port.

Also the given port *COMn* may have its own section named **DataPAF:n**. It defines parameters of the serial port.

### Definition of Variables

#### EXAMPLE

An example of the ASMEN variable definition:

```
3010E15m1, DATAPAF Średnia 15min energii kan. 1, XEN1, CHAN1, 1, 60, NOTHING_FP  
3010EBDSH, DATAPAF suma energii biernej kanałów w pop.okres, PEN11.8, CHAN1, 1,  
3600, NOTHING_FP
```

#### The List of All the Variable Types Supported by the DataPAF Driver

c - signifies the number of channel 1..30

s - time zone 0..8

**Table 7. The List of All the Variable Types Supported by the DataPAF Driver.**

<b>Name</b>	<b>Description</b>	<b>Type</b>	
ENc.s	The current value of energy in the current accounting period in a given time zone.	FLOAT	
PENc.s	The value of energy in the whole previous accounting period in a given time zone.	FLOAT	
MMc.s	Maximum power in the current accounting period in a given time zone.	FLOAT	
PMMc.s	Maximum power in the previous accounting period in a given time zone.	FLOAT	
MMIc	Minute power. The power for the previous minute.	FLOAT	
IMIc	Minute impulses. The number of impulses in the previous minute.	WORD	
IMPC	Impulses in integration period. The number of impulses in the last whole period of integration.	WORD	
BRKc	Break in integration period. The variable takes the value of 1 when a break in the line feeding impulses to counter occurred during the last period of integration.	BYTE	
POFc	The loss of voltage in integration period. The variable can take the following values: 0 - no loss; 1 - short voltage loss; 2 - long voltage loss; 3 - long and short voltage loss;	BYTE	
XENc	Energy value in the period of integration. Energy value in the last full period of integration. It is the value of IMP variable multiplied by the current value of input multiplier, taking into account the value of <i>energy_error</i> parameter.	FLOAT	
IMTc	Impulses in the current period of integration.	WORD	
Lc	Value of the counter in the current accounting period.	FLOAT	
PLc	Value of the counter in the previous accounting period.	FLOAT	
TIM	Current time of electricity meter as a string of bytes.	TEXT	
DAT	Current date of electricity meter as a string of bytes.	TEXT	
DTIM	Current time and date of electricity meter as a string of bytes.	TEXT	
ALRc	Alarms associated with the channel. Alarms are presented as a mask of bits:	BYTE	

	Bit 4 - a break in the input line (10H) Bit 5 - excess of power limit (20H)		
GALR	Counter alarms. Alarms are presented as a mask of bits: Bit 1 - no place for registration (2H) Bit 2 - used up batteries (4H) Bit 3 - break in the line of synchronization (8H)	DWORD	
INMULc	Value of input multiplier. It is the value to be multiplied by the number of impulses to obtain the equivalent energy value.	FLOAT	
INMULCNTc	Counter number.	WORD	
INMULKUKIc	Factor $K_u \cdot K_i$ .	FLOAT	
INMULCONSTc	Constant of the counter.	DWORD	
INMULDIGc	Number of digits.	WORD	
MCHAN	Maximum number of channels.	WORD	
PNT	Reference point.	WORD	
REG	Region.	WORD	
COL	Collector.	WORD	
CHAN	Number of channels.	WORD	
SUM	Number of sums.	WORD	
INTEG	Integration period in minutes.	WORD	
PPS	Beginning of the previous accounting period.	DWORD	
PPE	End of the previous accounting period.	DWORD	
CPS	Beginning of the current accounting period.	DWORD	
CPE	End of the current accounting period.	DWORD	

## Historical Data

Historical data are available for the following variable types: POF, IMP, BRK and XEN.

## Driver Configuration

DataPAF driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The default values of serial interfaces are retrieved from the section **DataPAF**, if section exists. This section is used to configure all the channels for the DataPAF protocol, which are declared in this system.

- Section name: DataPAF**
- Option name: NUMBER**
- Option value: number**

**Or**

- Section name: DataPAF**
- Option name: serial\_Number**
- Option value: number**

Meaning - determines an identification number of the DataPAF counter.  
Input of incorrect number makes the communication impossible.

Default value - 1234.

Parameter:  
*number* - counter number.

- Section name: DataPAF**
- Option name: baud**
- Option value: number**

**Or**

- Section name: DataPAF**
- Option name: bps**
- Option value: number**

Meaning - determines transmission speed.

Default value - 4800.

Parameter:  
*number* - transmission speed in Bd.

- Section name: DataPAF**
- Option name: diagnostics**
- Option value: number**

Meaning - declaring this position with value 14 will cause outputting the diagnostic information connected with time synchronization to the log file.

Default value - 0.

Parameter:  
*number* - 14.

- Section name: DataPAF**
- Option name: parity**
- Option value: parity\_parameter**

Meaning - determines parity;

Default value - N.

Parameter:  
*parity\_parameter* - allowed value:

n	- no parity bit,
o	- odd parity check,
e	- even parity check,
m	- mark,
s	- space.

**Section name: DataPAF**

**Option name: stop\_bits**

**Option value: number**

Meaning - determines a number of stop bits.

Default value - 1.

Parameter:

*number* - allowed values are 1 and 2.

**Section name: DataPAF**

**Option name: word**

**Option value: number**

**Or**

**Section name: DataPAF**

**Option name: word\_length**

**Option value: number**

Meaning - determines word length.

Default value - 8.

Parameter:

*number* - allowed values are from the range of 5 to 8.

**Section name: DataPAF**

**Option name: time\_out**

**Option value: number**

**Or**

**Section name: DataPAF**

**Option name: timeout**

**Option value: number**

Meaning - waiting time for the DataPAF answer.

Default value - 1000.

Parameter:

*number* - in milliseconds.

**Section name: DataPAF**

**Option name: repetitions**

**Option value: number**

Meaning - number of repetitions of communication operations ended with an error.

Default value - 3.

Parameter:

*number* - number of repetitions.

**Section name: DataPAF**

**Option name: max\_Time\_Difference**

**Option value: number**

Meaning - determines the maximal difference between the Asix system time and the DataPAF counter time, after which warnings will be generated in 'Control Panel'. The station time can be read with an interval defined by the parameter *time\_Check*.

Default value - 60.  
 Parameter:  
     *number* - number in seconds.

**Section name: DataPAF**  
 **Option name: time\_Check**  
 **Option value: number**  
 Meaning - interval, with which the current time of the counter is read.  
 Default value - 180.  
 Parameter:  
     *number* - number in seconds.

**Section name: DataPAF**  
 **Option name: system\_sync**  
 **Option value: number**  
 Meaning - maximal difference between the Asix system time and the DataPAF counter time, after which a system time synchronization with the counter time will occur. If the parameter value is 0, then the system time is not synchronized with the counter time.  
 Default value - 0.  
 Parameter:  
     *number* - number in seconds.

**Section name: DataPAF**  
 **Option name: period\_Check**  
 **Option value: number**  
 Meaning - interval, with which the change of current calculation period of the counter is checked.  
 Default value - 180.  
 Parameter:  
     *number* - number in seconds.

**Section name: DataPAF**  
 **Option name: max\_history**  
 **Option value: number**  
 Meaning - determines a time period from the current moment backwards, for which historical data, saved in station memory, will be read.  
 Default value - 35.  
 Parameter:  
     *number* - number in days.

**Section name: DataPAF**  
 **Option name: history\_Buffer\_Removal**  
 **Option value: number**  
 Meaning - the parameter determines a time period, after which buffers containing historical data, read for needs of an archiving module, are removed.  
 Default value 120.  
 Parameter:  
     *number* - the time is given in minutes.

**Section name: DataPAF**

**Option name: CRC16**

**Option value: YES/NO**

Meaning - determines if the CRC16 validity check has to be used in the communication with the counter. If NO is given, then the sum of sequent bytes will be calculated.

Default value - YES.

**Section name: DataPAF**

**Option name: log**

**Option value: file\_name**

Meaning - the parameter determines the name of file, to which additional diagnostic information will be written.

Default value - lack.

**Section name: DataPAF**

**Option name: alarm\_Code**

**Option value: number**

Meaning - the parameter determines the number of an alarm, generated by the driver in case of loss and re-establishing of connection with the station. The value of -1 (default) causes that alarms are not generated. In a situation of a connection loss, a number specifying the cause of connection loss is transferred together with the alarm code:  
0 - complete lack of any answer from the station;  
1 - timeout;  
2 - line errors (frame, parity, overrun errors);  
3 - checksum errors;  
4... - other errors.  
This number determines the end status of the last attempt of connection establishing.

Default value - lack.

**Section name: DataPAF**

**Option name: energy\_error**

**Option value: number**

Meaning - determines situations, when the status of the XEN variable value assumes an error value and it is the sum of the following values:  
1 error when a short decay of voltage;  
2 error when a long decay of voltage;  
4 error when a pause in input line of the impulse counter.

Default value - 0.

**Section name: DataPAF**

**Option name: mult**

**Option value: number**

Meaning - defines a value of all input multipliers. Input multipliers are used for calculating the energy on the basis of the impulse number. If this parameter is given, then the driver will not read values of multipliers from the energy counter. If beside the parameter *mult*, parameters determining values of multipliers for individual channels

(*multn*) are also used, then the parameter *mult* must be placed before all parameters *multn*.

Default value - lack.

Parameter:  
*number* - the multiplier is a floating-point number.

**Section name: DataPAF**

**Option name: multn**

**Option value: number**

Meaning - defines an input multiplier value for a channel with the number *n* (1-31).

Input multipliers are used for calculating the energy on the basis of the impulse number. If this parameter is given, then the driver will not read values of multipliers from the energy counter. If the parameter *mult* is also given, then it must be placed before all parameters *multn*.

Default value - lack.

Parameter:  
*number* - the multiplier is a floating-point number.

## Driver Parameterization Examples

### EXAMPLE

Channel declaration:

*Channel / Name:* KOMIN 2  
*Driver:* DataPAF  
*Channel parameters:* COM2

Driver parameters:

*Section name:* DataPAF: 2  
*Option name:* baud  
*Option value:* 19200

*Section name:* DataPAF: 2  
*Option name:* Number  
*Option value:* 4800

In the above example the station named KOMIN 2, connected to the port COM2, is defined. The transmission speed of 19200 bps will be used.

In case of time synchronization of the Asix station with the time of a chosen counter - the definition of this time should be placed in the ini section for the given counter. For this purpose the parameter *system\_sync*, to which the value of allowed difference of times in seconds is assigned, is used.

### EXAMPLE

Channel declaration:

*Channel / Name:* CHAN1  
*Driver:* DataPAF

*Channel parameters:* COM2

*Channel / Name:* KANAL2

*Driver:* DataPAF

*Channel parameters:* COM4

Driver parameters:

*Section name:* DataPAF:2

*Option name:* baud

*Option value:* 19200

*Section name:* DataPAF:2

*Option name:* Number

*Option value:* 4800

Driver parameters:

*Section name:* CHAN1

*Option name:* system\_sync

*Option value:* 5

*Section name:* CHAN1

*Option name:* log

*Option value:* DATAPAF.log

*Section name:* CHAN1

*Option name:* diagnostics

*Option value:* 14

In the example above the system time will be synchronized with the time of an energy counter connected to the port COM2. By placing the records *diagnostics=14* and *log=DATAPAF.log* in the counter section, it is possible to receive the record of time synchronization diagnostics in the file *DATAPAF.log*. The time is synchronized with an accuracy of 1 second.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

 **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
 YES / NO

The default value - NO.

 **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.21 DDE Driver

### Driver Use

The driver DDE is used to define the channel of the ASMEN module referring the variables shared by the driver of industrial controller implemented as a DDE server (called further shortly a DDE server).

In the new version of the driver while defining the ASMEN module channel, besides a DDE server name, also a name of the service registered by the DDE server and a DDE connection name have to be declared. Thanks to association of a DDE connection with a channel instead of with each variables separately, the way of variable definition is significantly simplified and the definitions of variables are more legible.

Supporting the variable groups allows the DDE driver to receive data from a DDE server in form of groups of variables and not only in form of single variables. Thanks to such solution the performance of data transmission may increase enormously. The maximal transmission performance by using the DDE protocol amounts up to 150 single-element groups of variables per second (at a computer with PCU Pentium 166 MHz) and descends slowly when the group size increases. If a group contains only one variable, then it is possible to transfer 150 variables per second whereas the group size amounts e.g. 25 variables, then it is possible to transfer 3000 variables per second.

A DDE server within each connection may make available a special variable, the value of which defines whether the variables retrieved from this connection have correct values, whether the connection with an industrial controller works correctly. The DDE driver assumes that such special variable has the *Status* name, but it is possible to define such variable separately for each DDE connection.

### External Specification

Declaration of the transmission channel utilizing the DDE driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: DDE

**DDE** tab:

**Channel parameters:**

*service, topic*

where:

*service* - service name registered by the DDE server;  
*topic* - topic name of a connection supported by the DDE server.

The names of connection service and topic are specific for each driver of the industrial controller implemented in form of a DDE server and their description may be found in the documentation of a given driver.

**EXAMPLE**

Examples of channel definitions.

*Channel / Name:* KanAdam  
*Driver:* DDE  
*Channel parameters:* Adam, E2018

The ASMEN channel named KanAdam is associated with a DDE driver and the topic of connection named E2018, made available by the DDE server named Adam.

*Channel / Name:* KanGE  
*Driver:* DDE  
*Channel parameters:* GESNP, GE

The ASMEN channel named KanGE is associated with the DDE driver and the topic of connection named GE, shared by the DDE server named GENSP.

**Variable Definitions**

After the DDE driver start-up the ASMEN module transfers to the DDE driver an information about process variables taken from the definition file of variables. The definition of a variable used by the DDE driver is as follows:

*asmen\_variable, description, "item | number\_in\_group | type", channel\_name, quantity, refreshing\_frequency, conversion\_function*

where:

- |                        |   |
|------------------------|---|
| <i>asmen_name</i>      | - variable name of the ASMEN module;  |
| <i>item</i>            | - name of a variable or a group of variables accessed by the DDE server, maximal length of this name amounts 255 characters (255 characters, it is the maximal length of text served by the DDE protocol);                          |
| <i>number_in_group</i> | - number of a variable in the group of variables to which the ASMEN variable refers; the enumeration begin from 1; if the DDE server as the <i>item</i> sends variables individually, then this parameter should have a value of 1; |
| <i>type</i>            | - type of the ASMEN variable.   |

As the *type* parameter it may be given the letter:  
 S - short,  
 W - word,  
 L - long,  
 D - double word,  
 F - float.

The variable type definition is necessary to make possible the variable conversion to the binary form, used by the ASMEN module. The data from the DDE server are sent in text form and it is not possible to define their type unambiguously, therefore its type in variable definition is defined as public.

If a variable is to be not only read but also written, then the source of such variable in a DDE server must be a single variable. If to the ASMEN variable, which is not a part of variables group, you try to give a new value from the application level of Asix, then an error occurs. If it were possible to give a new variable to only one variable in the group, the DDE driver while sending this group to the server should send also new values of the other variables in this group. The use of recently sent values of the other variables in the group as their new values might cause hard perturbations.

## EXAMPLE

Examples of Variable Definitions

Z1, "O1, 1, W", KanAdam, 1, 1, NOTHING

The ASMEN variable Z1 is associated with the first variable in the variable group named O1 taken from the DDE server specified in the definition of the KanAdam channel.

Z2, "%I1, 5, s", KanalGE, 1, 1, NOTHING

The ASMEN variable Z2 is associated with the fifth variable in the variable group named %I1 taken from the DDE server specified in the definition of the KanGE channel.

## Driver Parameters

DDE driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

- Section name: DDE**
- Option name: StatusVariable**
- Option value: service, topic, item**

Meaning - the item of an initialization file, it defines the name of a variable handled by the DDE server. It allows to monitor the connection status of the DDE server with an industrial controller. If the value of this variable equals 1, the connection with the controller works correctly; if the variable value equals 0, the connection with the controller is broken. In order to define a status variable it is necessary in the initialization application file to define the item *StatusVariable* and to give to it a value in the format *service, topic, item*.

Default value - by default, it is assumed that there is no status variable within the connection.

Parameter:

*service* and *topic* - define the connection;

*item* - is the name of status variable within this connection.

A DDE server may support many DDE connections simultaneously and for each of them a status variable may be defined separately.

## EXAMPLE

Examples of definition of status variable:

*Section name:* DDE

*Option name:* StatusVariable

*Option value:* TPERM, S1, Connected

Within the connection TPERM, S1 the name of the status variable is Connected.

## Thread Priority of the Driver

PriData – value of thread priority of the driver sending data to the ASMEN module, the default value is set to 1.

## Internal Specification

The driver, at the first reference to the DDE server described by a pair of *service, topic*, establishes a connection with this server and keeps it on continuously up to ending the activity of the driver or of the DDE server. At the first reading of the first variable belonging to the variable group a refreshing of this variables group begins. The new values are written to the internal buffer of the driver and transferred to the ASMEN module in case of reading of refresh type. In case of ordinary reading the data are read directly from the DDE server. If it is not possible to begin refreshing, the variable value is read at each reference of the ASMEN module to it. Writing operations are carried out always synchronously and they update the internal buffer of the variable.

In case of breaking the connection with a DDE server, the DDE driver tries to rebuild this connection.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.22 Diva - Driver for Data Exchange with DIVA Industrial Cameras by VDG Security B.V.

### Driver Use

The Diva driver is used for the control of the DIVA video management solution developed by VDG Security B.V. The number of commands executed by the driver is limited to execution of the *selectCamera()* and *selectLayout()* commands.

Data exchange between the driver and DIVA platform was realized based on "Diva HTTP Interface Version 2.3.1" documentation developed by VDG Software B.V. and affiliated Companies.

### Declaration of Transmission Channel

Declaration of the transmission channel operating according to the Diva driver protocol requires adding a channel with the following parameters to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* Diva

**Diva** tab:

**Channel Parameters:**

**serverIp=IPaddress; password=text; user=text  
 [;alarmNo=number][;timeout=number]**

where:

serverIP	- IP address of the DIVA platform server,
password	- access password to the DIVA platform server,
user	- name of the DIVA platform user,
AlarmNo	- number of the Asix system alarm which is generated in the absence of communication with the DIVA platform server,
timeout	- timeout of the HTTP protocol operation expressed in seconds (by default, 3 seconds).

#### EXAMPLE

*Name:* KANAL

*Driver:* CtDiva

*Channel Parameters:* serverIP=10.10.104.73; password=abcd; user=admin;  
 AlarmNo=100

### Declaration of Variables

The driver provides access only to virtual variables which are used to:

- present the names assigned by the driver to the DIVA platform workstations,

- define the command parameters for *selectLayout()* and *selectCamera()*,
- execute the commands *selectLayout()* and *selectCamera()*,
- present the connection status to the DIVA platform server.

The following symbolic addresses are acceptable:

- ALIAS** - a unique name (alias) specified for the server (workstation) of the DIVA platform by the driver (read-only),
- CAMERA** - the name of a camera (read/write),
- DIVA** - a unique name (alias) of a DIVA device on which the operation *selectLayout()* or *selectCamera()* will be executed (read/write),
- LAYOUT** - the layout name (read/write),
- MONITOR** - the number of the DIVA platform workstation monitor (read/write),
- PANEL** - the name of the panel in the layout specified by a symbolic address (read/write),
- SEL\_CAMERA** - execution of the command *selectCamera()* using the parameters specified in the other symbolic addresses (read-only),
- SEL\_LAYOUT** - execution of the command *selectLayout()* using the parameters specified in the other symbolic addresses (read-only),
- STATUS** - connection status with the DIVA platform server (0 - o.k., 1 - error) (read-only).

Values of the variables of the **MONITOR**, **SEL\_CAMERA**, **SEL\_LAYOUT** symbolic addresses are numbers, therefore the **NIC** conversion function should be used for these variables. Values of the variables including any of the other symbolic addresses are texts, therefore the **NIC\_TEXT** conversion function should be used for these variables.

For the purposes of the DIVA servers and workstations identification a symbolic address **ALIAS** has been introduced. This address allows to access the unique name assigned by the driver to the individual DIVA platform devices. The reason for the introduction of symbolic addresses of **ALIAS** type was that the names assigned to the DIVA servers (workstations) can be identical, and differentiating the DIVA platform devices by a multi-digit **divaid** unique identifier is inconvenient.

The symbolic address **ALIAS** requires providing a parameter (separated from the key **ALIAS** word with a dot), whose value equals the reference number assigned to a DIVA platform device by the driver. The minimum value of the parameter is 1, whereas the maximum value is dependant on the DIVA platform configuration.

Assigning aliases to the DIVA platform servers (workstations) is entered in the driver log during periodical execution of the *getDivaList()* command. Addressing can have the form of the following example:

```
alias : NVDR1master name : NVDR1master divald : 8678015003039002295 type : 'Diva server'
alias : NVDR1master_1 name : NVDR1master divald : 8678015003039002295 type : 'Diva client'
alias : NVDR2 name : NVDR2 divald : 435039309 type : "
alias : NVDR3 name : NVDR3 divald : 586093958 type : "
alias : Nvdr1master name : Nvdr1master divald : 15551739365488560896 type : 'Diva client'
alias : Nvdr1master_1 name : Nvdr1master divald : 16755657203230560766 type : 'Diva client'
alias : Nvdr1master_2 name : Nvdr1master divald : 23456757203230500111 type : 'Diva client'
```

where:

- alias* - a unique name specified for the workstation by the driver,
- name* - a name specified for the workstation by the DIVA platform,
- divaid* - a unique identifier specified for the workstation by the DIVA platform,
- type* - type of the workstation in the DIVA platform.

### EXAMPLE

Example of variable declarations:

```
J01, DIVA server name, DIVA, CHANNEL, 20, 1, NIC_TEXT
J01, layout name, LAYOUT, CHANNEL, 20, 1, NIC_TEXT
J03, monitor number, MONITOR, CHANNEL, 1, 1, NIC
```

J04, select camera,	SEL_CAMERA,	CHANNEL, 1, 1, NIC
J04, select layout,	SEL_LAYOUT,	CHANNEL, 1, 1, NIC
J05, alias of device No. 1,	ALIAS.1,	CHANNEL, 1, 1, NIC_TEXT
J06, alias of device No. 2,	ALIAS.2,	CHANNEL, 1, 1, NIC_TEXT
J07, alias of device No. 3,	ALIAS.3,	CHANNEL, 1, 1, NIC_TEXT

### Selection of Camera

The process of selection of a camera consists of two steps:

- defining the parameters for the *selectCamera()* command through:
  - entering the camera name to the variable of the CAMERA symbolic address,
  - entering the layout name to the variable of the LAYOUT symbolic address,
  - entering the panel name to the variable of the PANEL symbolic address,
  - entering the Diva server name to the variable of the DIVA symbolic address, if the command *selectCamera()* is to refer to a given server only,
- executing the *selectCamera()* command through:
  - entering any value to the variable of the SEL\_CAMERA symbolic address.

### Selection of Layout

The process of selection of a layout consists of two steps:

- defining the parameters for the *selectLayout()* command through:
  - entering the DIVA server name to the variable of the DIVA symbolic address,
  - entering the layout name to the variable of the LAYOUT symbolic address,
  - entering the number of a monitor to the variable of the MONITOR symbolic address.
- executing the *selectLayout()* command through:
  - entering any value to the variable of the SEL\_LAYOUT symbolic address.

## Driver Configuration

The Diva driver parameters are declared in the *Miscellaneous* module in the *Directly entered options* tab:

In the section called **CTDIVA** it is possible to include declaration items:

- creating a log file,
- the log file size,
- configuration data log file storing the data read from the DIVA webserver and the commands sent to the DIVA webserver.

**Section name: CTDIVA**

**Option name: LOG\_FILE**

**Option value: log\_file\_name**

Meaning - the text log file to which driver status messages are stored is used for diagnostic purposes.

The default value - by default, the log file is not created.

**Section name: CTDIVA**

**Option name: LOG\_FILE\_SIZE**

**Option value: number**

Meaning - this option is used to determine the log file size defined using the *LOG FILE* option.

Option value:

*number* - the size of the log file in MB.

The default value - by default, the log file size is 10 MB.

**Section name: CTDIVA**

**Option name: LOG\_OF\_TELEGRAMS**

**Option value: YES | NO**

Meaning - this option allows writing to the log file (declared using the *LOG\_FILE* item) the configuration data of the DIVA platform read using the *getDivaList()*, *getDeviceList()*, *getLayoutList()* commands, as well as the values of the commands sent to the DIVA webserver.

The default value - by default, the option value is set to NO.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.23 DImS - Driver of Electric Energy Counter of ZxD/ZxG/ZxQ Type of Landys&Gyr Company

### Driver Use

The driver is used for data exchange between the Asix system and electric energy counters of ZxD/ZxG/ZxQ type of Landys & Gyr company, using the DLMS protocol. The communication is performed through RS-485/RS-232 interface of the counters. The driver can not be used for data exchange using optical link.

The driver allows for:

- reading registers of the class 1, 3, 4, 5
- reading profiles of power, accounting periods and event logs (registerf of the class 7),
- recording in text files the profiles of power, accounting periods and event log.

Parameterization of DImS driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the CtDImS driver requires a channel with the following parameters to be added to the **Current data** module:

**Standard** tab:

*Name: logical name of the transmission channel*

**Driver: DImS**

**DImS/Channel parameters** tab:

where:

<i>Serial port</i>	- serial port name (COM),
<i>Baud rate</i>	- baud rate between a computer and a device. The following values are possible: 2400, 4800, 9600, 19200, 38400 Bd. The default value is 9600 Bd,
<i>Period</i>	- period time (in seconds) between consecutive readouts of counter registers. Default values is 10 seconds.

**EXAMPLE**

Exemplary channel declaration on COM2 port:

*Serial port: 2;*  
*Period: 20;*  
*Baud rate: 9600*

**Declaration of Current Variables**

The syntax of current variable address:

*[counter\_name]/register\_OBIS\_code*

where:

*counter\_name* - (optional). It contains the unique address name of the counter used in multipoint installations to identify individual counters. *counter\_name* corresponds to 'Device address' according to PN-EN 61107. When connected point-to-point the *counter\_name* may be omitted,

*register\_OBIS\_code* - register OBIS code of the counter compatible with the readout list which has been entered to the counter by the producer at the parameterization stage. If the last element of the OBIS (group F) code is the value 255, this element can be omitted.

**NOTICE:**

The register values transmitting numbers the driver converts to DOUBLE numbers.

**EXAMPLE**

```
/* 1-0:0.0.1 - id number of the counter 1*/
JJ_01, id number of the counter 1, "/1-0:0.0.1", CHANNEL, 1, 1, NOTHING_FP

/* 1-1:1.8.0 - register of taken active energy*/
JJ_02, register of taken active energy, "/1-1:1.8.0", CHANNEL, 1, 1, NOTHING_FP
```

**Declaration of Power Profile Variables**

The access to power profile values needs the profile log file to be created, to which the driver will write the data of power profile data (read from the counter with the use of appropriate

register of the class 7). The data from the latest entry in the profile log file is shared by the driver as current values of power profiles (with the time stamp taken from the profile log). The older entries in profile log file are shared as historical values of power profiles (through access interface to historical data).

The address of power profile variable is built on the basis of current variable address by adding to it the suffix **.P**.

The syntax of power profile variable address is as follows:

*[counter\_name]/ register\_OBIS\_code.P*

where:

- |                           |  |
|---------------------------|--|
| <i>counter_name</i>       | - it has the same meaning as for current variable declaration,   |
| <i>register_OBIS_code</i> | - it has the same meaning as for current variable declaration,   |
| <i>.P</i>                 | - suffix used to distinguish the power profile variable address from the address of the corresponding current variable (if it exists). |

**EXAMPLE**

Profile_1.4.0, profile 1.4.0,	"/1-1:1.4.0.P",	CHANNEL, 1, 1, NOTHING_FP
Profile_2.4.0, profile 2.4.0,	"/1-1:2.4.0.P",	CHANNEL, 1, 1, NOTHING_FP

**Notice:**

The number and addresses of power profile variable addresses are specific for each counter and are depended on the way of counter parameterization.

## Declaration of Variables of Accounting Periods

The access to values of accounting periods needs the file of accounting periods, to which the driver will record the data of accounting periods, to be created. The data is read from the counter with the use of the appropriate register of the class 7). The data from the latest entry in the accounting period log file is shared by the driver as current values of accounting periods (with the time stamp taken from the profile log). The older entries in accounting period log file are shared as historical values of accounting periods (through access interface to historical data).

The address of accounting period variable is built on the basis of current variable address by adding to it the suffix **.A**.

The syntax of variable address is as follows:

*[counter\_name]/ register\_OBIS\_code.A*

where:

- |                     |  |
|---------------------|--|
| <i>counter_name</i> | - it has the same meaning as for current variable declaration, |
|---------------------|--|

*register\_OBIS\_code* - it has the same meaning as for current variable declaration,

.A - suffix used to distinguish the accounting period variable address from the address of the corresponding current variable (if it exists).

**EXAMPLE**

Period\_1.8.1, rejestr arch 1.8.1, "/1-1:1.8.1.A", CHANNEL, 1, 1, NOTHING\_FP  
 Period\_1.8.2, rejestr arch 1.8.2, "/1-1:1.8.2.A", CHANNEL, 1, 1, NOTHING\_FP

**Notice:**

The number and addresses of accounting period variable addresses are specific for each counter and are depended on the way of counter parameterization.

## Driver Parameters

The driver configuration is defined in the *Current Data* module, in the channel operating according to Dlms driver.

 **Log file**

Meaning - The item allows to define a file where all diagnostic messages of the driver are written.

Default value - The log file is not created.

Parameter:

*file\_name*

 **Log file size**

Meaning - The item allows to specify the size of log file defined by the option: *Log file*.

Default value - 10 MB.

Parameter:

*number*

- log file size in MB.

 **Log of telegrams**

Meaning: The item allows to write to the log file the contents of telegrams sent between the driver and network clients. Writing the contents of telegrams to the log file should be used only while the Asix start-up.

**Default value: NO.**

Option value:

YES/NO

 **History log file**

Meaning - The item allows to define a file where all diagnostic messages about operations performed with the use of interface of acces to variable historical values of power profiles and accounting periods. History log file is common for all counters.

The size of history log file is the same as for driver log file.

Default value

- The log file is not created.

Parameter:

*log\_file\_name*

**Receive timeout**

Meaning - The option defines the timeout for the first response sign from the counter.  
Default value - 1000.  
Option value:  
    *number* - timeout in milliseconds.

**Char timeout**

Meaning - The option defines the maximal time between successive characters of response from the counter.  
Default value - 100.  
Option value:  
    *number* - timeout in milliseconds.

**Time synchronization**

Meaning - Period of time synchronization.  
Default value - 0 - no synchronization.  
Option value:  
    *number* - time in minutes.

**Suspended Readout Period**

Purpose: The option specified the time during which the counter readout is suspended around midnight (i.e. readout disabled - *seconds* - midnight - *seconds* - readout enabled). When determining the value of the option, allow for some offset during the disabled readout time, because the disable time may occur during current data transmission (a dozen or so seconds ?). For other data categories the offset is of several seconds. During the suspended readout time, the values of current variables are given status OPC\_QUALITY\_UNCERTAIN.  
Option value:  
    *seconds* - time measured in seconds.  
The default value: The default value is 0 (i.e. the readout is not suspended).

**Suspended Readout – Every Day**

Purpose: This option is used to suspend the counter readout on a daily basis. It suspends the data readout at midnight every day.  
The default value: By default value of the position is NO (readout is not suspended).

**Suspended Readout – Every Decade**

Purpose: This option is used to suspend the counter readout on a decade basis. It suspends the data readout at midnight when the decade changes.  
The default value: By default value of the position is NO (readout is not suspended).

**Suspended Readout – Every Month**

Purpose: This option is used to suspend the counter readout on a monthly basis. It suspends the data readout at midnight when the month changes.  
The default value: By default value of the position is NO (readout is not suspended).

**EXAMPLE**

An exemplary declaration

*Log file:* d:\tmp\CtDlms\dlms.log  
*Log file size:* 10  
*Log of telegrams:* YES  
*Receive timeout:* 1500  
*Character timeout:* 150

## Parameterization of Individual Counters

### Case 1

When using a point-point system the counter is parameterized using tabs activated by pushing the button *Add point-to-point* on the *Dlms/Counters* tab.

**Basic** tab:

**Client address**

Meaning

- the option defines the value of the parameter 'Client Address' in the parameters DLMS connection of the channel.

Default value

- 16.

Option value:

*number*

- client address in decimal format.

**Server address**

Meaning

- the option defines the value of the parameter 'Physical Server' in the parameters DLMS connection of the channel.

Default value

- 1.

Option value:

*number*

- server address in decimal format.

**History data upload period**

Meaning

- the option defines the frequency of query sending by the driver to complete the historical data of power profiles, accounting periods and events.

Default value

- by default, the driver completes historical data by sending queries every minute.

Option value:

*number*

- server address in decimal format.

**Diagnostics** tab:

**Log file**

Meaning

- The item allows to define a file where all diagnostic messages generated in the given channel are written.

Default value

- Messages generated in the channel are written to the driver log file.

Parameter:

*file\_name*

**Log file size**

Meaning

- The item allows to specify the size of log file size defined by the option: *Log file*.

Default value

- The log file size is inherited from the driver log file.

Parameter:

*number*

- log file size in MB.

**Log of telegrams**

Meaning:

The item allows to write to the log file the contents of telegrams sent in the given channel. Writing the contents of telegrams to the log file should be used only while the **Asix** start-up.

Default value:

is inherited from the driver settings.

Option value:

YES/NO

**Events** tab:

**Events log**

Meaning

- The events read from the counter are written to text 'csv' file. Each event is written in a separate line. The events in the file are ordered by increasing timestamps. An exemplary event form is as follows:

P.98;1;2005-05-10  
08:14:33;0048;;3;;;F.F;;1.8.0;kWh;024;00000000;0000.0000

Default value

The item allows to define a file where event log will be written.  
- The log file is not created.

**Events log file size**

Meaning

- The item allows to specify the size of log file defined by the option: *Events log*.

Default value

- 10.

Option value:

*number*

- log file size in MB.

**Without events log**

Meaning

- Using the option it is possible to switch off reading the event log.

Default value

- the driver reads event log.

**Events log read period**

Meaning - The option determines the cycle of reading the event log from the counter.

Default value - Every 1 hour.

Option value:

*number* - time in hours.

 **Events log history scope**

Meaning - The option declares how far into the past history of events the driver will reach, reading event log.

Default value - By default, the driver goes back 130 days since the current 24 hour, reading the history of events.

Option value:

*number of days*

**Power profiles tab:** **Power profiles log**

Meaning - Power profiles are read from the counter and saved to a text file in the format 'csv'. Each power profile is written in a separate line. Power profiles are ordered by increasing timestamps. An exemplary form of a singular power profile is as follows (the profile is divided into 2 text lines for the better clarity):

```
P.01;1;2005-05-10 12:45:00;0008;15;6;1.5.0;kW;5.5.0;kvar;8.5.0;kvar;
32.7;V;52.7;V;72.7;V;5.123000;24.2438900;123.654320;---.;---.;---
```

The option allows to define the full path to the file where profile log will be written.

Default value - The log file is not created.

 **Power profiles log file size**

Meaning - The item allows to specify the size of log file defined by the option: *Power profiles log*.

Default value - 10.

Option value:

*number* - log file size in MB.

 **Without power profiles log**

Meaning - The option switches off reading power profiles.

Default value - By default, the driver reads the power profile log.

 **Power profiles log read period**

Meaning - The option determines the cycle of reading the power profile log from the counter.

Default value - Every 1 hour.

Option value:

*number* - time in hours.



**Billing periods log read period**

Meaning - The option determines the cycle of reading the accounting period log from the counter.

Option value:

*number* - time in hours.

Default value - Every 1 hour.

 **Billing periods log history scope**

Meaning - The option declares how far into the past history of events the driver will reach, reading accounting period log.

Default value - By default, the driver goes back 130 days since the current 24 hour, reading the history of accounting periods.

**EXAMPLE**

An exemplary counter declaration.

*Log file:* c:\tmp\CtDlms\kanal.log

*Events log:* c:\tmp\CtDlms\book.log

*Events log file size:* 15

*Events log read period:* 2

*Power profiles log:* c:\tmp\CtDlms\profile.log

*Power profiles log file size:* 20

*Power profiles log read period:* 4

*Billing periods log:* c:\tmp\CtDlms\billing.log

*Log of telegrams:* TAK

*Events log history scope:* 40

**Case 2**

When using a few counters - there are parameterized with the use of tabs activated by the pressing the button *Add new counter* on the *Dlms/Counters* tab.

The tabs and parameters are the same as for parameterization of one counter in point-point installation.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

 **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:  
YES / NO

## Communication Drivers

- computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

- Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

- computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

- Purpose - with this option deployed, local control in a channel will be disabled.
- Option value:  
YES / NO
- The default value - NO.

### **Data Validity Period**

- Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

- number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.
- The default value - by default, no time for data validity is set.

## 1.24 DlmsTcpip - Driver for Landys&Gyr Electricity Meters

### Driver Use

The driver is used for data exchange between the Asix system and electric energy counters of Landys & Gyr company, using the DLMS protocol. The communication is performed through Ethernet.

The driver allows for:

- reading registers of the class 1, 3, 4, 5
- reading profiles of power, accounting periods and event logs (register of the class 7),
- recording in text files the profiles of power, accounting periods and event log.

Parameterization of CtDlmsTcpip driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the CtDlmsTcpip driver requires a channel with the following parameters to be added to the **Current data** module:

**Standard** tab:

*Name: logical name of the transmission channel*

**Driver: Dlms**

**DlmsTcpip/Channel parameters** tab:

where:

<i>IP Address of the Counter</i>	- IP Address of the counter,
<i>IP Port of the counter</i>	- IP Port of the counter,
<i>IP Address of the computer</i>	- IP address of the computer used by the given channels; the address given by Windows is taken as default,
<i>Period</i>	- interval (in seconds) between consecutive readouts of counter records. The default value is 10 seconds.

### EXAMPLE

Exemplary channel declaration on COM2 port:

```
Serial port: 2;
Period: 20;
Baud rate: 9600
```

## Declaration of Current Variables

The syntax of current variable address:

*[counter\_name]/register\_OBIS\_code*

where:

*counter\_name* - (optional). It contains the unique address name of the counter used in multipoint installations to identify individual counters.  
*counter\_name* corresponds to 'Device address' according to PN-EN 61107. When connected point-to-point the *counter\_name* may be omitted,

*register\_OBIS\_code* - register OBIS code of the counter compatible with the readout list which has been entered to the counter by the producer at the parameterization stage. If the last element of the OBIS (group F) code is the value 255, this element can be omitted.

### NOTICE:

The register values transmitting numbers the driver converts to DOUBLE numbers.

### EXAMPLE

```
/* 1-0:0.0.1 - id number of the counter 1*/
JJ_01, id number of the counter 1, "/1-0:0.0.1", CHANNEL, 1, 1, NOTHING_FP
```

```
/* 1-1:1.8.0 - register of taken active energy*/
JJ_02, register of taken active energy, "/1-1:1.8.0", CHANNEL, 1, 1, NOTHING_FP
```

## Declaration of Variables of Power Profiles

The access to power profile values needs the profile log file to be created, to which the driver will write the data of power profile data (read from the counter with the use of appropriate register of the class 7). The data from the latest entry in the profile log file is shared by the driver as current values of power profiles (with the time stamp taken from the profile log). The older entries in profile log file are shared as historical values of power profiles (through access interface to historical data).

The address of power profile variable is built on the basis of current variable address by adding to it the suffix **.P**.

The syntax of power profile variable address is as follows:

*[counter\_name]/register\_OBIS\_code.P*

where:

*counter\_name* - it has the same meaning as for current variable declaration,  
*register\_OBIS\_code* - it has the same meaning as for current variable declaration,  
**.P** - suffix used to distinguish the power profile variable address from the address of the corresponding current variable (if it exists).

**EXAMPLE**

Profile_1.4.0,	profile 1.4.0,	"/1-1:1.4.0.P",	CHANNEL, 1, 1, NOTHING_FP
Profile_2.4.0,	profile 2.4.0,	"/1-1:2.4.0.P",	CHANNEL, 1, 1, NOTHING_FP

**Notice:**

The number and addresses of power profile variable addresses are specific for each counter and are depended on the way of counter parameterization.

### Declaration of Variables of Billing Periods

The access to values of accounting periods needs the file of accounting periods, to which the driver will record the data of accounting periods, to be created. The data is read from the counter with the use of the appropriate register of the class 7). The data from the latest entry in the accounting period log file is shared by the driver as current values of accounting periods (with the time stamp taken from the profile log). The older entries in accounting period log file are shared as historical values of accounting periods (through access interface to historical data).

The address of accounting period variable is built on the basis of current variable address by adding to it the suffix **.A**.

The syntax of variable address is as follows:

*[counter\_name]/register\_OBIS\_code.A*

where:

<i>counter_name</i>	- it has the same meaning as for current variable declaration,
<i>register_OBIS_code</i>	- it has the same meaning as for current variable declaration,
<i>.A</i>	- suffix used to distinguish the accounting period variable address from the address of the corresponding current variable (if it exists).

**EXAMPLE**

Period_1.8.1,	rejestr arch 1.8.1,	"/1-1:1.8.1.A",	CHANNEL, 1, 1, NOTHING_FP
Period_1.8.2,	rejestr arch 1.8.2,	"/1-1:1.8.2.A",	CHANNEL, 1, 1, NOTHING_FP

**Notice:**

The number and addresses of accounting period variable addresses are specific for each counter and are depended on the way of counter parameterization.

### Driver Parameters

Dlms driver parameters are declared in the *DlmsTcpip/Driver parameters* tabs:

- creating a log file,
- the log file size,
- log of telegrams,
- history log file,
- reception timeout,

## Communication Drivers

- response character timeout,
- Computer IP address,
- Time synchronization,
- Suspended readout period,
- suspended readout (every day, every decade, every month).

Nazwy opcji związanych z plikiem logu nawiązują do konwencji stosowanej w innych drajwerach Asmena.

### **Log file**

Meaning - The item allows to define a file where all diagnostic messages of the driver are written.

Default value - The log file is not created.

Parameter:

*file\_name*

### **Log file size**

Meaning - The item allows to specify the size of log file defined by the option: *Log file*.

Default value - 10 MB.

Parameter:

*number*

- log file size in MB.

### **Log of telegrams**

Meaning: The item allows to write to the log file the contents of telegrams sent between the driver and network clients. Writing the contents of telegrams to the log file should be used only while the Asix start-up.

Default value: NO.

Option value:

**YES/NO**

### **History log file**

Meaning - The item allows to define a file where all diagnostic messages about operations performed with the use of interface of acces to variable historical values of power profiles and accounting periods. History log file is common for all counters.

The size of history log file is the same as for driver log file.

Default value - The log file is not created.

Parameter:

*log\_file\_name*

### **Receive timeout**

Meaning - The option defines the timeout for the first response sign from the counter.

Default value - 1000.

Option value:

*number*

- timeout in milliseconds.

### **Character timeout**

Meaning - The option defines the maximal time between successive characters of response from the counter.

Default value - 100.  
 Option value:  
     *number* - timeout in milliseconds.

**IP Address of the Computer**

Purpose - this option is used to specify the IP address of the computer's network card, with the use of which the driver communicates with counters.  
 The default value - if this option is not declared, then the driver will use the network adapter provided by the system, which can prevent communication with the controller.  
 Option value:  
     *IP address* - network card address in IPv4 notation

**EXAMPLE**

An exemplary declaration

*Log file:* d:\tmp\CtDlms\dlms.log  
*Log file size:* 10  
*Log of telegrams:* YES  
*Receive timeout:* 1500  
*Character timeout:* 150

**Time Synchronization**

Purpose: This item enables time synchronization between Asix and the controller, with specified intervals between subsequent synchronisations.  
 The default value: The default value is 0 - no time synchronization.  
 Parameter:  
     *number* - time in minutes.

**Options related to the possibility of suspending the counter readout while the counter is determining daily, decade and monthly closures.**

**Suspended Readout Period**

Purpose: The option specified the time during which the counter readout is suspended around midnight (i.e. readout disabled - *seconds* - midnight - *seconds* - readout enabled). When determining the value of the option, allow for some offset during the disabled readout time, because the disable time may occur during current data transmission (a dozen or so seconds?). For other data categories the offset is of several seconds. During the suspended readout time, the values of current variables are given status OPC\_QUALITY\_UNCERTAIN.  
 Option value:  
     *seconds* - time measured in seconds.  
 The default value: The default value is 0 (i.e. the readout is not suspended).

**Suspended Readout – Every Day**

Purpose: This option is used to suspend the counter readout on a daily basis. It suspends the data readout at midnight every day.  
 The default value: By default value of the position is NO (readout is not suspended).

**Suspended Readout – Every Decade**

Purpose: This option is used to suspend the counter readout on a decade basis. It suspends the data readout at midnight when the decade changes.  
The default value: By default value of the position is NO (readout is not suspended).

**Suspended Readout – Every Month**

Purpose: This option is used to suspend the counter readout on a monthly basis. It suspends the data readout at midnight when the month changes.  
The default value: By default value of the position is NO (readout is not suspended).

## Declaration of Counters

### Case 1

If there is a single counter (when point-point installation is used) - it is configured in tabs created by means of the **DLmsTcpip/Counter** tab with the **Add point-to-point** button.

**Basic** tab:

**Client address**

Meaning - the option defines the value of the parameter 'Client Address' in the parameters DLMS connection of the channel.

Default value - 16.

Option value:  
*number* - client address in decimal format.

**Server Logical Address**

Purpose - the logical address of the server as 'upper HDLC address'.

The default value - the default option value is 1.

Option value:  
*number* - the address of the server as a decimal.

**Server Physical Address**

Purpose - the physical address of the server as 'lower HDLC address'.

The default value - the default option value is 0.

Option value:  
*number* - the address of the server as a decimal.

**Advanced** tab: **History Data Upload Period**

Purpose: - the option is used to declare how often the driver issues requests to fill in historical data concerning power profiles, accounting periods and events.

The default value: - by default the driver fills in historical data issuing requests every minute.

 **Time Synchronization**

Purpose: This item enables time synchronization between Asix and the controller, with specified intervals between subsequent synchronisations.

The default value: The default value is 0 - no time synchronization.

Parameter:  
*number* - time in minutes.

**Suspended Readout** tab: **Suspended Readout Period**

Purpose: The option specified the time during which the counter readout is suspended around midnight (i.e. readout disabled - *seconds* - midnight - *seconds* - readout enabled). When determining the value of the option, allow for some offset during the disabled readout time, because the disable time may occur during current data transmission (a dozen or so seconds ?). For other data categories the offset is of several seconds. During the suspended readout time, the values of current variables are given status OPC\_QUALITY\_UNCERTAIN.

Option value:  
*seconds* - time measured in seconds.

The default value: The default value is 0 (i.e. the readout is not suspended).

 **Suspended Readout – Every Day**

Purpose: This option is used to suspend the counter readout on a daily basis. It suspends the data readout at midnight every day.

The default value: By default value of the position is NO (readout is not suspended).

 **Suspended Readout – Every Decade**

Purpose: This option is used to suspend the counter readout on a decade basis. It suspends the data readout at midnight when the decade changes.

The default value: By default value of the position is NO (readout is not suspended).

 **Suspended Readout – Every Month**

Purpose: This option is used to suspend the counter readout on a monthly basis. It suspends the data readout at midnight when the month changes.

The default value: By default value of the position is NO (readout is not suspended).

**Diagnostics** tab:

**Log file**

Meaning

- The item allows to define a file where all diagnostic messages generated in the given channel are written.

Default value

- Messages generated in the channel are written to the driver log file.

Parameter:

*file\_name*

**Log file size**

Meaning

- The item allows to specify the size of log file size defined by the option: *Log file*.

Default value

- The log file size is inherited from the driver log file.

Parameter:

*number*

- log file size in MB.

**Log of telegrams**

Meaning:

The item allows to write to the log file the contents of telegrams sent in the given channel. Writing the contents of telegrams to the log file should be used only while the Asix start-up.

Default value:

is inherited from the driver settings.

Option value:

YES/NO

**Events** tab:

**Events log**

Meaning

- The events read from the counter are written to text 'csv' file. Each event is written in a separate line. The events in the file are ordered by increasing timestamps. An exemplary event form is as follows:

P.98;1;2005-05-10  
08:14:33;0048;;3;;;F.F;;1.8.0;kWh;024;00000000;0000.0000

Default value

The item allows to define a file where event log will be written.  
- The log file is not created.

**Events log file size**

Meaning

- The item allows to specify the size of log file defined by the option: *Events log*.

Default value

- 10.

Option value:

*number*

- log file size in MB.

**Without events log**

Meaning - Using the option it is possible to switch off reading the event log.  
 Default value - the driver reads event log.

 **Events log read period**

Meaning - The option determines the cycle of reading the event log from the counter.  
 Default value - Every 1 hour.  
 Option value:  
   *number* - time in hours.

 **Events log history scope**

Meaning - The option declares how far into the past history of events the driver will reach, reading event log.  
 Default value - By default, the driver goes back 130 days since the current 24 hour, reading the history of events.  
 Option value:  
   *number of days*

**Power profiles tab:** **Power profiles log**

Meaning - Power profiles are read from the counter and saved to a text file in the format 'csv'. Each power profile is written in a separate line. Power profiles are ordered by increasing timestamps. An exemplary form of a singular power profile is as follows (the profile is divided into 2 text lines for the better clarity):

```
P,01;1;2005-05-10 12:45:00;0008;15;6;1.5.0;kW;5.5.0;kvar;8.5.0;kvar;
32.7;V;52.7;V;72.7;V;5.123000;24.2438900;123.654320;---.-;---.-;---.-
```

The option allows to define the full path to the file where profile log will be written.

Default value - The log file is not created.

 **Power profiles log file size**

Meaning - The item allows to specify the size of log file defined by the option: *Power profiles log*.

Default value - 10.

Option value:  
   *number* - log file size in MB.

 **Without power profiles log**

Meaning - The option switches off reading power profiles.  
 Default value - By default, the driver reads the power profile log.

 **Power profiles log read period**

Meaning - The option determines the cycle of reading the power profile log from the counter.  
 Default value - Every 1 hour.



**Billing periods log read period**

Meaning - The option determines the cycle of reading the accounting period log from the counter.

Option value:

*number* - time in hours.  
Default value - Every 1 hour. •

 **Billing periods log history scope**

Meaning - The option declares how far into the past history of events the driver will reach, reading accounting period log.

Default value - By default, the driver goes back 130 days since the current 24 hour, reading the history of accounting periods.

**EXAMPLE**

An exemplary counter declaration.

```
Log file: c:\tmp\CtDlms\kanal.log
Events log: c:\tmp\CtDlms\book.log
Events log file size: 15
Events log read period: 2
Power profiles log: c:\tmp\CtDlms\profile.log
Power profiles log file size: 20
Power profiles log read period: 4
Billing periods log: c:\tmp\CtDlms\billing.log
Log of telegrams: TAK
Events log history scope: 40
```

**Case 2**

If there are several counters - they are configured in tabs created by means of the **DlmsTcip/Counters** with the **Add new counter** button.

The same tabs and parameters as in case 1 (for a single counter where the point-point installation is used).

 **Server Physical Address**

Purpose - the option is used to declare the counter's 'lower HDLC address', a parameter necessary to establish a connection in multidrop configurations.

The default value - the default option value is 0.

Option value:

*number* - the address of the server as a decimal.

**For point-to-point connections, the 'lower HDLC address' does not apply.**

## 1.25 DMS285 - Driver of Protocol for DURAG DMS 285 Analyzers

### Driver Use

The DMS285 protocol driver is designed to establish the communication between an Asix system computer and a DURAG D-MS285 computer for emission monitoring. The driver operates with devices supporting the protocol of version 1.22.

Parameterization of DMS285driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the DMS285driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: DMS285

**DMS285/Channel parameters** tab:

<i>Serial port</i>	- number of the serial port to which the network of DMS285 controllers is connected.
<i>Name</i>	- name of the DMS285 station; by default, a logical channel name is assumed as a station name; name is completed with spaces to the length of 8 characters.
<i>Alarm code</i>	- number of alarm generated by the driver in the case of loss and re-establishing connection with the station. A value of 1 (default) causes no generation of alarms. When losing the connection, the following number characterizing the cause of communication loss is transmitted with the alarm code:  0 - complete lack of any response from the station;  1 - timeout;  2 - line errors (errors of frame, parity, overrun);  3 - checksum errors;  4 - other errors.

This number determines the status of the end of the last attempt to establish communication.

Default value: none

## Driver Configuration

DMS285 driver parameters are declared on *Driver parameters* tabs.

**Option name: Baud**

**Option value: *number***

Meaning - determines transmission speed.

Default value - 9600.

Parameter:

*number* - number in Bd.

**Option name: Parity**

**Option value: *parity\_parameter***

Meaning - determines parity.

Default value - e.

Parameter:

*parity\_parameter* - default value:

n	- no parity bit,
o	- odd parity check,
e	- even parity check,
m	- mark
s	- space.

**Option name: Stop bits**

**Option value: *number***

Meaning - determines number of stop bits.

Default value - 1.

Parameter:

*number* - admissible values are 1 and 2.

**Option name: Word length**

**Option value: *number***

Meaning - word length.

Default value - 8.

Parameter:

*number* - allowed values are from 5 to 8.

**Option name: Timeout**

**Option value: *number***

Meaning - waiting time for answer from DMS285A.

Default value - 10.

Parameter:

*number* - number in seconds.

**Option name: Log**

**Option value: *file\_name***

Meaning - parameter determines a file name, to which additional diagnostic information will be written.

Default value - lack.

**Option name: Bad Data Classes**

**Option value: *class1, class2, ..., classN***

Meaning - parameter determines numbers of classes, which cause invalidity of data 44.5K.n and 44.5M.n. The class number is taken up from the variable 44.1.n (Klassenangabe der Konz.).

Default value - lack (-) (class value has no influence on the validity of read data).

Parameter:  
*class1, class2, ..., classN* - numbers of classes.

**Option name: Bad Data Status**

**Option value: *status1, status2, ..., statusN***

Meaning - parameter determines values of the variable 43.3.n (Zustand des Kanals - Channel state), for which values of variables 43.7K.n, 43.7M.n, 43.9K.n, 43.9M.n, 43.10K.n and 43.10M.n are assumed as invalid.

Default value - lack (-) (value of the variable 43.3.n has no influence on validity of read data).

Parameter:  
*status1, status2, ..., statusN* - values of the variable 43.3.n (Zustand des Kanals).

**Option name: Bad Data Classes 2**

**Option value: *class1, class2, ..., classN***

Meaning - parameter determines numbers of classes, invalidity of data 44.5K.n and 44.5M.n while calculating 48-hour averages. The class number is taken up from the variable 44.1.n (Klassenangabe der Konz.).

Default value - lack (-) (class value has no influence on the validity of read data).

Parameter:  
*class1, class2, ..., classN* - numbers of classes.

**Option name: Minimal Measurements**

**Option value: *number***

Meaning - minimal number of measurements required for calculation of a 48-hour average.

Default value - lack.

Parameter:  
*number* - number of measurements.

**Option name: Max Time Difference**

**Option value: *number***

Meaning - determines the maximal difference in seconds between the Asix system time and DMS295 station time, after exceeding of which warnings will be output to 'Control Panel'. Station time is read only during reading of 43 and 50 type variables.

Default value - 60.

Parameter:  
*number* - number in seconds.

**Option name: AutoSync**

**Option value: number**

Meaning - determines the maximal difference between the Asix system time and the DMS285 station time, after exceeding of which the driver will set the station time on the Asix system time. Time synchronization occurs only during data reading from the station. The minimal value of this parameter equals 10 seconds. If a smaller value is given, then 10 seconds will be assumed.

Default value - no time synchronization.

Parameter:  
*number* - number in seconds.

**Option name: MaxAutoSync**

**Option value: number**

Meaning - determines the maximal value between the Asix system time and the DMS285 station time, after exceeding of which the driver will **not** synchronise the station time even if this difference exceeds the value determined by the *AutoSync* parameter.

Default value - 6800.

Parameter:  
*number* - number in seconds.

**Option name: Time limiter**

**Option value: number**

Meaning - determines the maximal deviation of data time in hours in relation to the current system time, whose exceeding causes a data is found invalid. The time of a data read from DURAG computer must be contained in the time interval current time +/- deviation, so that it might be found valid. If the parameter has a value of 0 then data time is not checked.

Default value - 24.

Parameter:  
*number* - number in hours.

**Option name: Alarm Code**

**Option value: alarm\_number**

Meaning - parameter determines the number of alarm generated by the driver in case of loss and re-establishing the connection with the station. The value of -1 (default) causes that the alarms are not generated. In case of connection loss, one of the following number specifying the reason for the dropped connection is relayed with the alarm code:

0 - complete lack of any answer from the station,

1 - timeout,

2 - errors of lines (errors of border, parity, overrun),

3 - errors of checksum,

4 - other errors.

This number determines the status of the end of last attempt made to establish a connection.

Default value - lack.

Parameter:  
*Alarm\_number* - alarm number.

**Option name: Max history**

**Option value: number**

Meaning - determines a time period, counted from the current moment backwards, for which historical data, stored in the station memory, will be read.

Default value - 20.

Parameter:  
*number* - number in days.

**Option name: Var\_44\_1\_n**

**Option value: variable\_name,channel\_number**

Meaning - name of the variable 44.1.n and archive type, which is saved in the archive and used simultaneously for calculation of 48-hour average 44.105K.n. After the variable name the archive type should be given after a comma. *n* signifies the channel number.

Default value - lack.

Parameter:  
*variable\_name* - determines the name of 44.1.n variable; *n* signifies the channel number.  
*channel\_number* - number of the channel.

**NOTE** While configuring the variable for ASPAD the option *DO\_NOT\_PACK* should be **absolutely** given!!!

**Option name: Var\_44\_5K\_n**

**Option value: variable\_name,channel\_number**

Meaning - name of the variable 44.5K.n and archive type, which is saved in the archive and used simultaneously for calculation of 48-hours average 44.105K.n. After the variable name the archive type should be given after a comma. *n* signifies the channel number.

Default value - lack.

Parameter:  
*variable\_name* - determines the name of 44.5K.n variable; *n* signifies the channel number.  
*channel\_number* - number of the channel.

## Examples of Driver Configuration

### EXAMPLE 1

Channel declaration:

*Chanel / Name:* KOMIN 2

*Driver:* DMS285

*Channel parameters:*

*Serial port:* COM2

Driver parameters:

*Option name:* baud

*Option value:* 19200

In the above example a station named KOMIN 2 is connected to the COM2 port. The transmission speed of 19200 bps will be used.

## EXAMPLE 2

Channel declarations:

*Chanel / Name:* KOMIN 1  
*Driver:* DMS285  
*Channel parameters:*

*Serial port:* COM2

*Chanel / Name:* KOMIN 2  
*Driver:* DMS285  
*Channel parameters:*

*Serial port:* COM2

*Chanel / Name:* KOMIN 3  
*Driver:* DMS285  
*Channel parameters:*

*Serial port:* COM2

*Chanel / Name:* KOMIN 4  
*Driver:* DMS285  
*Channel parameters:*

*Serial port:* COM3

*Chanel / Name:* KOMIN 5  
*Driver:* DMS285  
*Channel parameters:*

*Serial port:* COM3

*Chanel / Name:* KOMIN 6  
*Driver:* DMS285  
*Channel parameters:*

*Serial port:* COM4

Driver parameters:

Default values for all stations:

*Option name:* speed  
*Option value:* 19200

*Option name:* Invalidity\_Status  
*Option value:* 1, 6, 14

Default values for stations connected to the COM3 port:

*Option name:* speed  
*Option value:* 9600

Other parameters:

For channel: KOMIN 2  
*Option name:* Bad Data Status  
*Option value:* 5

For channel: KOMIN 4  
*Option name:* Bad Data Status  
*Option value:* -

In the above example the stations of names from KOMIN 1 up to KOMIN 6 are defined. The stations KOMIN 1, KOMIN 2 and KOMIN 3 are connected to the COM2 port. The stations KOMIN 4 and KOMIN 5 are connected to the COM3 port. The station KOMIN 6 is connected to the COM4 port. All the serial ports except COM3 will work with a speed of 19200 baud. The COM3 port will work with a speed of 9600 baud. All the stations except the stations KOMIN 2 and KOMIN 4 will use invalidity statuses 1, 6 and 14. The station KOMIN 2 uses a value of 5 as an invalidity status. The station KOMIN 4 does not use any invalidity status - setting parameter "-" was necessary to change the default values.

## Defining the Process Variables

The variable definition is based on the DMS285 protocol description. The list of all the types of variables is given at the end of this chapter.

$$\text{type.subtype} \left[ \begin{array}{c} K \\ M \end{array} \right] \left[ \text{idx} \right] \left[ \begin{array}{c} Mxx \\ \text{subfield} \end{array} \right] \left[ \text{channel} \right]$$

where:

- |                 |   |
|-----------------|---|
| <i>type</i>     | - defines information type e.g.:<br>43 - instantaneous values,<br>44 - integrals,<br>17 - parameters;   |
| <i>subtype</i>  | - number of given information, e.g. 43 for instantaneous values defines actual current intensity in a given channel;  |
| K,P.            | - concentration/flow (only if the subtype contains data for both these categories);   |
| <i>idx</i>      | - index - only for indexed variables, e.g. classification; index is a number $\geq 1$ ;   |
| <i>channel</i>  | - channel number; a channel number may be given only for the values related to the channel; in case of general information it should be omitted.  |
| <i>Mxx</i>      | - bit mask; xx is a number in hexadecimal code; on the datum received from the station, the AND operation with the number xx is executed;   |
| <i>subfield</i> | - subfield name; for data representing time the following subfields are defined:<br><br>SEC, MIN, HOUR, DAY, MONTH, YEAR    - DWORD type,<br>TIME, DATE, DATETIME                - TEXT type. |

In order to display values TIME, DATE, DATETIME the object STRING may be applied. The function NOTHING\_TEXT must be use as the conversion function. The length of displayed string is given as the counter of elements.

**EXAMPLE**

An example of variable declaration:

```
ACT_TIME, actual date and time of the station, 43.1.DATETIME.1, CHANNEL-
DMS285, 20,30, NOTHING_TEXT
```

Format of date and time:

```
dd-mm-rrrr gg:mm:ss.
```

**EXAMPLE**

```
43.11.1      - actual instantaneous value for the analog channel 1
43.1.DAY     - actual time of the station - month day number
43.7K.1     - actual concentration for the channel 1
43.2[2]     - digital inputs 16-23
```

## Time of Data

The data are transferred by the driver to the Asix system together with the time of their reception (time of the DMS285 station). In case of other types than the type 43, which does NOT contain actual time, the time is established on the ground of previously received packet of the type 43 (variable 43.1) and time of its reading. The time defined by the variable 44.20 is assigned to the variables 44.1, 44.5K.n and 44.5M.n. The time of the variables of the packet 48 is rounded down to the hour 00:00. The variables of the packet 48 arriving at 00:00 +/- 2 min are treated as invalid. The variables 44.105M.n and 44.105K.n have always the time 00:00.

## Historical Data

For the variables 44.5K and 44.5M it is possible to read historical data.

## List of All the Variable Types Supported by DMS285 Driver

See the following tables.

**Table 8. List of All the Variable Types Supported by DMS285 Driver.**

<b>Nazwa</b>	<b>Opis</b>	<b>Typ</b>
0.1.n	Analog-Eingangstrom	FLOAT
1.1.n	Physikalische Grosse	FLOAT
2.1.n	Momentane Konzentration	FLOAT
3.1.n	Momentaner Massenstrom	FLOAT
4.1.n	Festintegral Konzentration	FLOAT
5.1.n	Festintegral Massenstrom	FLOAT
6.1.n	Integrationszeit	WORD
6.2.n	Messzeit	WORD
6.3.n	Wartungszeit	WORD
6.4.n	Ausser-Betriebs-Zeit	WORD
6.5K.n	Festintegral Konzentration	FLOAT
6.5M.n	Festintegral Massenstrom	FLOAT

**Table 9. List of All the Variable Types Supported by DMS285 Driver (continuation)**

Nazwa	Opis	Typ
7.1.n	Letzter klassierter Konzentration	FLOAT
8.1.n	Letzter klassierter Massenstrom	FLOAT
9.1.n	Aktueller Tagesmittelwert Konzentration	FLOAT
10.1.n	Aktueller Tagesmittelwert Massenstrom	FLOAT
11.1.n	Letzter klassierter Tagesmittelwert Konzentration	FLOAT
12.1.n	Letzter klassierter Tagesmittelwert Massenstrom	FLOAT
13.1.n	Bisherige Summe der Jahresemission	FLOAT
14.1.n	Vorjahressumme der Emission	FLOAT
15.1.n	Tagesbetriebszeit in min	WORD
16.1.n	Jahresbetriebszeit in min	DWORD
<b>Parametryzacja stacji</b>		
17.1.n	Parametr (n- numer parametru)	FLOAT
:	<b>Fehlerzustand</b> (x - fehlt)	
18.1.x	Untere Grenze	WORD
18.2.x	obere Grenze	WORD
18.3.x	Tagesmittelwert-uberschreitung	WORD
18.4.x	Ersatzwert	WORD
18.5.x	Allg. fehler	WORD
19.1[x].n	Klasseninhalt Konzentration x - numer klasy	WORD
20.1[x].n	Klasseninhalt Massenstrom x - numer klasy	WORD
21.1[1].n - 21.1[3].n	Aktuelles Mischungsverhältnis	FLOAT
22.1[1].n - 22.1[3].n	Integriertes Mischungsverhältnis	FLOAT
23.1[1] - 23.1[16]	Statussignale	BYTE
38.1.n	Trend der Konzentration	FLOAT
38.2.n	Freilast der Konzentration	FLOAT
39.1.n	Trend der Massenstrome	FLOAT
39.2.n	Freilast der Massenstrome	FLOAT
43.1	Datum/Uhrzeit	WORD
43.2[1] - 43.2[8]	Digital-Eingänge	BYTE
43.3.n	Zustand des Kanals	BYTE
43.4.n	Momentanes Mischungsverhältnis 1	FLOAT
43.5.n	Momentanes Mischungsverhältnis 2	FLOAT

**Table 10. List of All the Variable Types Supported by DMS285 Driver (continuation).**

Nazwa	Opis	Typ
43.6.n	Momentanes Mischungsverhältnis 3	FLOAT
43.7K.n	Momentane Konzentration	FLOAT
43.7M.n	Momentaner Massenstrom	FLOAT
43.8K.n	Festintegral Konzentration	FLOAT
43.8M.n	Festintegral Massenstrom	FLOAT
43.9K.n	hochgerechnete Konzentration	FLOAT
43.9M.n	hochgerechnete Massenstrom	FLOAT
43.10.c	Physikalische GroÙe	FLOAT
43.11.c	Analog Eingangstrom	FLOAT
43.12.n	Messzeit	WORD
43.13.n	abgelaufene Integrationszeit	WORD
43.14.n	Tagesbetriebszeit	WORD
43.15.n	Wartungszeit	WORD
43.16.n	Nummer der Klassierung	BYTE
43.17[1] - 43.17[18]	Fehlerzustand	BYTE
43.18K.n	Freilast Konzentration	FLOAT
43.18M.n	Freilast Mass.	FLOAT
43.19K.n	concentration limit	FLOAT
43.19M.n	flow limit	FLOAT
<b>UWAGA</b>	Status zmiennych 43.7.n, 43.9.n i 43.10.n jest uzalezniony od wartosci zmiennej 43.3.n (Zustand des Kanals). Jesli wartosc zmiennej 43.3.n przyjmuje jedna z wartosci okrelonych w parametryzacji stacji to zmiennym przypisywany jest status bledu.	
44.1.n	Klassenangabe der Konzentrationsklassierung	BYTE
44.2.n	Integr Mischungsverhältnis 1	FLOAT
44.3.n	Integr Mischungsverhältnis 2	FLOAT
44.4.n	Integr Mischungsverhältnis 3	FLOAT
44.5K.n	Letzte klassierte Konzentration	FLOAT
44.5M.n	Letzter klassierter Massenstrom	FLOAT
44.6K.n	Akt Tagesmittelwert Konzentration	FLOAT
44.6M.n	Akt Tagesmittelwert Massenstrom	FLOAT
44.7K.n	Letzter klassierter Tagesmittelwert Konzentration	FLOAT
44.7M.n	Letzter klassierter Tagesmittelwert Massenstrom	FLOAT
44.8.n	Summe Jahresemission	FLOAT
44.9.n	Jahresbetriebszeit	DWORD
44.10.c	Integrierte physik. GroÙen	FLOAT
44.11[1] - 44.11[4]	Hauptalarm	BYTE
44.12K.n	Klassenbreite Tagesmittelwert Konzentration	FLOAT
44.12M.n	Klassenbreite Tagesmittelwert Massenstrom	FLOAT
44.13K.n	limit_concentration	FLOAT
44.13M.n	limit_mass	FLOAT
44.14K.n	Trendwert Tagesmittel Konz.	FLOAT
44.14M.n	Trendwert Tagesmittel Mass.	FLOAT
44.15[1] - 44.15[3]	Anlagenname*	BYTE
44.16K.n	Monatsmittelwert Konzentration	FLOAT
44.16M.n	Monatsmittelwert Massenstrom	FLOAT
44.17.n	Massenstrom klassiert in Klasse	BYTE
44.18.n	Zustand Integral	BYTE
44.19K.n	Überschreitungsmerker Konz.	BYTE
44.19M.n	Überschreitungsmerker Mass.	BYTE
44.20	Zeitpunkt der klassierung	WORD
44.21[1] - 44.21[4]	Störung integr. Physikalische werte V3.22	BYTE

**Table 11. List of All the Variable Types Supported by DMS285 Driver (continuation).**

Nazwa	Opis	Typ
44.105K.n.	Letzte klassierte Konzentration - orednia 48-godzinna zmiennej 44.5K.n	FLOAT
44.105M.n.	Letzter klassierter Massenstrom - orednia 48-godzinna zmiennej 44.5M.n	FLOAT
<b>UWAGA</b>	Status zmiennych 44.5K.n i 44.5M.n jest uzależniony od wartości zmiennej 44.1.n (Klassenangabe der Konzentrationsklassierung). Jeśli wartość zmiennej 44.1.n przyjmuje jedną z wartości określonych w parametryzacji stacji to zmiennym przypisywany jest status błędu. Zmiennym 44.5K.n i 44.5M.n jest przypisywany czas określony zmienna 44.20.	
45.1.n	Summe vorjahremmission	FLOAT
45.2.n	Klassenbreiten KK	FLOAT
45.3.n	Klassenbreiten KL2K	FLOAT
45.4.n	Klassenbreiten KL3K	FLOAT
45.5.n	Klassenbreiten KM	FLOAT
45.6.n	Klassenbreiten KL2M	FLOAT
45.7.n	Klassenbreiten KL3M	FLOAT
45.8.n	Integrationszeiten	WORD
	<b>Aktueller Stand der Klasseninhalte in Binar</b>	
46.4.n	Dzienny czas pracy [min]	WORD
46.5.n	Roczny czas pracy [min]	DWORD
46.6.n	Czas początkowy	DWORD
46.8[1].n - 48.8[3].n	Czasy przestojów REA	WORD
46.9.n	Wyłączenie sieci - klasa 0	WORD
46.10[1].n - 48.10[21].n	klasa 1-21 (roczne, stężenie)	WORD
46.11.n	klasa 24 (roczna, stężenie)	WORD
46.12.n	klasa 25 (roczna, stężenie)	WORD
46.13.n	klasa 26 (roczna, stężenie)	WORD
46.14.n	klasa 23 (roczna, stężenie)	WORD
46.15.n	klasa 35 (roczna, stężenie)	WORD
46.16[1].n - 48.16[22].n	klasa 1-22 (dzienne, stężenie)	WORD
46.17.n	klasa 23 (dzienna, stężenie)	WORD
46.18.n	klasa 25 (dzienna, stężenie)	WORD
46.19.n	klasa 27 (dzienna, stężenie)	WORD
46.20.n	klasa 28 (dzienna, stężenie)	WORD
46.21.n	klasa 29 (dzienna, stężenie)	WORD
46.22.n	klasa 35 (dzienna, stężenie)	WORD
46.23.n	klasa 30 (dzienna, stężenie)	WORD
46.24.n	klasa 31 (dzienna, stężenie)	WORD
46.25.n	klasa 32 (dzienna, stężenie)	WORD
46.26.n	klasa 33 (dzienna, stężenie)	WORD
46.27[1].n - 48.27[21].n	klasa 1-21 (roczna, strumień masy)	WORD
46.28.n	klasa 27 (roczna, strumień masy)	WORD
46.29.n	klasa 28 (roczna, strumień masy)	WORD
46.30.n	klasa 29 (roczna, strumień masy)	WORD
46.31.n	klasa 23 (roczna, strumień masy)	WORD
46.32.n	klasa 34 (roczna, strumień masy)	WORD

**Table 12. List of All the Variable Types Supported by DMS285 Driver (continuation).**

Nazwa	Opis	Typ
46.33[1].n - 48.33[22].n	klasa 1-22 (dzienna, strumień masy)	WORD
46.34.n	klasa 23 (dzienna, strumień masy)	WORD
46.35.n	klasa 30 (dzienna, strumień masy)	WORD
46.36.n	klasa 31 (dzienna, strumień masy)	WORD
46.37.n	klasa 32 (dzienna, strumień masy)	WORD
46.38.n	klasa 36 (dzienna, strumień masy)	WORD
46.39.n	średnia wartość miesięczna stężenia	FLOAT
46.40.n	średnia wartość miesięczna strumienia masy	FLOAT
46.41.n	roczna emisja	FLOAT
46.42.n	ostatnie średnio-dobowe stężenie	FLOAT
46.43.n	ostatnia średnio-dobowa emisja	FLOAT
	<b>Klasseninhalte beim lt. Tageswechsel in Binar</b>	
47.4.n	Dzienny czas pracy [min]	WORD
47.5.n	Roczny czas pracy [min]	DWORD
47.6.n	Czas początkowy	DWORD
47.8[1].n - 48.8[3].n	Czasy przestojów REA	WORD
47.9.n	Wyłączenie sieci - klasa 0	WORD
47.10[1].n - 48.10[21].n	klasa 1-21 (roczna, stężenie)	WORD
47.11.n	klasa 24 (roczna, stężenie)	WORD
47.12.n	klasa 25 (roczna, stężenie)	WORD
47.13.n	klasa 26 (roczna, stężenie)	WORD
47.14.n	klasa 23 (roczna, stężenie)	WORD
47.15.n	klasa 35 (roczna, stężenie)	WORD
47.16[1].n - 48.16[22].n	klasa 1-22 (dzienna, stężenie)	WORD
47.17.n	klasa 23 (dzienna, stężenie)	WORD
47.18.n	klasa 25 (dzienna, stężenie)	WORD
47.19.n	klasa 27 (dzienna, stężenie)	WORD
47.20.n	klasa 28 (dzienna, stężenie)	WORD
47.21.n	klasa 29 (dzienna, stężenie)	WORD
47.22.n	klasa 35 (dzienna, stężenie)	WORD
47.23.n	klasa 30 (dzienna, stężenie)	WORD
47.24.n	klasa 31 (dzienna, stężenie)	WORD
47.25.n	klasa 32 (dzienna, stężenie)	WORD
47.26.n	klasa 33 (dzienna, stężenie)	WORD
47.27[1].n - 48.27[21].n	klasa 1-21 (roczna, strumień masy)	WORD
47.28.n	klasa 27 (roczna, strumień masy)	WORD
47.29.n	klasa 28 (roczna, strumień masy)	WORD
47.30.n	klasa 29 (roczna, strumień masy)	WORD
47.31.n	klasa 23 (roczna, strumień masy)	WORD
47.32.n	klasa 34 (roczna, strumień masy)	WORD
47.33[1].n - 48.33[22].n	klasa 1-22 (dzienna, strumień masy)	WORD
47.34.n	klasa 23 (dzienna, strumień masy)	WORD
47.35.n	klasa 30 (dzienna, strumień masy)	WORD
47.36.n	klasa 31 (dzienna, strumień masy)	WORD
47.37.n	klasa 32 (dzienna, strumień masy)	WORD
47.38.n	klasa 36 (dzienna, strumień masy)	WORD
47.39.n	średnia wartość miesięczna stężenia	FLOAT
47.40.n	średnia wartość miesięczna strumienia masy	FLOAT
47.41.n	roczna emisja	FLOAT
47.42.n	ostatnie średnio-dobowe stężenie	FLOAT
47.43.n	ostatnia średnio-dobowa emisja	FLOAT

**Table 13. List of All the Variable Types Supported by DMS285 Driver (continuation).**

Nazwa	Opis	Typ
	<b>Gesamt-Klassinh. b. lt. Tagesw. in Binar</b>	
48.4.n	Dzienny czas pracy [min]	WORD
48.5.n	Roczny czas pracy [min]	DWORD
48.6.n	Czas początkowy	DWORD
48.8[1].n - 48.8[3].n	Czasy przestojów REA	WORD
48.9.n	Wyłączenie sieci - klasa 0	WORD
48.10[1].n - 48.10[21].n	klasa 1-21 (roczne, stężenie)	WORD
48.11.n	klasa 24 (roczna, stężenie)	WORD
48.12.n	klasa 25 (roczna, stężenie)	WORD
48.13.n	klasa 26 (roczna, stężenie)	WORD
48.14.n	klasa 23 (roczna, stężenie)	WORD
48.15.n	klasa 35 (roczna, stężenie)	WORD
48.16[1].n - 48.16[22].n	klasa 1-22 (dzienne, stężenie)	WORD
48.17.n	klasa 23 (dzienna, stężenie)	WORD
48.18.n	klasa 25 (dzienna, stężenie)	WORD
48.19.n	klasa 27 (dzienna, stężenie)	WORD
48.20.n	klasa 28 (dzienna, stężenie)	WORD
48.21.n	klasa 29 (dzienna, stężenie)	WORD
48.22.n	klasa 35 (dzienna, stężenie)	WORD
48.23.n	klasa 30 (dzienna, stężenie)	WORD
48.24.n	klasa 31 (dzienna, stężenie)	WORD
48.25.n	klasa 32 (dzienna, stężenie)	WORD
48.26.n	klasa 33 (dzienna, stężenie)	WORD
48.27[1].n - 48.27[21].n	klasa 1-21 (roczna, strumień masy)	WORD
48.28.n	klasa 27 (roczna, strumień masy)	WORD
48.29.n	klasa 28 (roczna, strumień masy)	WORD
48.30.n	klasa 29 (roczna, strumień masy)	WORD
48.31.n	klasa 23 (roczna, strumień masy)	WORD
48.32.n	klasa 34 (roczna, strumień masy)	WORD
48.33[1].n - 48.33[22].n	klasa 1-22 (dzienne, strumień masy)	WORD
48.34.n	klasa 23 (dzienna, strumień masy)	WORD
48.35.n	klasa 30 (dzienna, strumień masy)	WORD
48.36.n	klasa 31 (dzienna, strumień masy)	WORD
48.37.n	klasa 32 (dzienna, strumień masy)	WORD
48.38.n	klasa 36 (dzienna, strumień masy)	WORD
48.39.n	średnia wartość miesięczna stężenia	FLOAT
48.40.n	średnia wartość miesięczna strumienia masy	FLOAT
48.41.n	roczna emisja	FLOAT
48.42.n	ostatnie średnio-dobowe stężenie	FLOAT
48.43.n	ostatnia średnio-dobowa emisja	FLOAT

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.26 DMS500 - Driver of Protocol for DURAG DMS 500 Analyzers

### Driver Use

The DMS500 driver is designed for data exchange between the D-MS500 emission computer and the Asix system by using serial interfaces. The driver supports devices with implemented company software in versions DMS500 v. 1.23, 1.55, 1.59.

Parameterization of DMS500 driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the DMS500 driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* *DMS500*

**DMS500** tab:

*Serial port* - number of the serial port to which the network of DMS500 controllers is connected.

*Name* - name of the DMS285 station; by default, a logical channel name is assumed as a station name; name is completed with spaces to the length of 8 characters.

*Report* - the item allows to declare a place where the report read from a DMS computer will be sent by appropriate setting of pseudo-variable *REPORT*. As the purpose place of the report:

- printer name,
- disk file name

may be entered.  
 Default value - by default, the report transferred from a DMS computer is sent to a printer (LPT1).

Parameter:  
*name* - printer name or disk file name.

The DMS500 driver is loaded as a DLL automatically.

## Addressing the Process Variables

The definition of process variables is based on the DMS protocol description titled "Beschreibung der Kommunikation D-EVA mit DMS500".

The syntax of symbolic address which is used for variables belonging to the DMS500 driver channel is as follows:

*type.subtype [K|M] [idx] [.Mxx|.subfield] [.channel]*

where:

<i>type</i>	- define the type of information: 43 - instantaneous values, 44 - integrals, 45 - parameters, 46 - classification of analog inputs - current results, 47 - classification of analog inputs - yearly results, 48 - classification of analog inputs - daily results, 56 - classification of digital inputs - current results, 57 - classification of digital inputs - yearly results, 58 - classification of digital inputs - daily results;
<i>subtype</i>	- number of required information; e.g. 13 for instantaneous values defines actual current intensity in a given channel;
<i>K,M</i>	- concentration/flow (only when a subtype contains data for both these categories);
<i>idx</i>	- index - only for indexed variables, e.g. classification; an index is a number bigger or equal to 1;
<i>Mxx</i>	- bit mask; xx is a number in hexadecimal code; on the data received from a DMS computer the AND operation with the number of xx is executed;
<i>Subfield</i>	- subfield name; for time values the following subfields are defined: SEC, MIN, HOUR, DAY, MONTH, YEAR;
<i>Channel</i>	- number of the channel; the channel number may be given only for the variables related to the channel. In case of general data it should be omitted.

### EXAMPLES

43.14.1	actual instantaneous value of current for the analog channel 1
43.1.DAY	actual time of DMS computer - number of month day
43.17K.1	actual concentration for the channel 1
46.4K[5].1	actual value of the class 5 (concentration) for the channel 1

## REPORT Special Variable

The special variable **REPORT** is a pseudo-variable of the 16-bit word type. Writing a given value to the REPORT variable causes reading an appropriate report (defined by this value) from the DMS computer. The report will be sent to a place defined in the **REPORT** item.

## Access to Historical Data

The DMS500 driver enables the ASPAD module to access to the following historical data:

44.9K	- class number for concentration,
44.9M	- class number for flow,
44.16K	- concentration value,
44.16M	- flow value.

## Time of Data

The data are transferred to the Asix system together with time of their retrieving. In case of the type 45, for which the data packet does **not** contain current time (field no. 1 or 10 for types 56,57,58), the time is determined on the ground of previously received packet provided with time and on the ground of the time of its reading. Data packets do not contain the time for the type 45.

## Incorrect Data

For the data 43.14 and 43.15, the DMS computer transfers status data (43.13). If any bit (specified in the protocol description) will be set, then such data is treated as incorrect.

## Driver Configuration

DMS500 driver parameters are declared on *Driver parameters* tabs.

**Option name: Transmission speed**

**Option value: number**

Meaning - the item is used to declare a transmission speed.  
 Default value - by default, the transmission speed is equal to 9600 Bd.  
 Parameter:  
     *number* - transmission speed in bauds.

**Option name: Parity**

**Option value: check\_type**

Meaning - the item is used to declare the kind of parity check.  
 Default value - by default, the parity check is even.  
 Parameter:  
     *check\_type* - identifier of the kind of parity check:  
             n - no parity bit,  
             o - parity odd,  
             e - parity even,  
             m - mark,  
             s - space.

**Option name: Stop bits**

**Option value: number**

Meaning - the item is used to declare a number of stop bits.  
 Default value - by default, it is assumed to be 1 stop bit.  
 Parameter:  
     *number* - number of stop bits: 1 or 2.

**Option name: Word length**

**Option value: number**

Meaning - the item is used to declare a number of bits in a transmitted character.  
 Default value - by default, it is assumed that a character is of 8 bits in length.  
 Parameter:  
     *number* - number of bits in a character (5 to 8).

**Option name: Timeout**

**Option value: number**

Meaning - the item is used to declare waiting time for an answer from the DMS computer.

## Communication Drivers

Default value - by default, it is assumed 10 seconds.  
Parameter:  
    *number* - waiting time for an answer in seconds.

**Option name: Auto sync**

**Option value: number**

Meaning - the item is used to declare an automatic synchronization of an Asix system computer clock with a DMS computer clock. The parameter value determines maximal drift of time. The synchronization occurs when this drift limit is exceeded. The time from the DMS computer is received only during reading other data.

Default value - by default, the time is not synchronized.

Parameter:

*number* - 0 (no synchronization) or maximal variance of times, (in seconds), after which the synchronization occurs.

**Option name: History Buffer Removal**

**Option value: number**

Meaning - the item is used to declare the time, after which buffers containing historical data that were read for needs of ASPAD are removed.

Default value - by default, history buffers are removed after 30 minutes.

Parameter:

*number* - time (in minutes), after which history buffers are removed.

**Option name: Max History Buffers**

**Option value: number**

Meaning - the item is used to declare the maximal number of history buffers containing historical data that were read for needs of ASPAD. One buffer contains historical data for one channel. It is stored in the memory for a period defined in the item *History\_Buffer\_Removal*. One buffer occupies 30 bytes of memory. If archived data are stored by the DMS computer each 30 minutes then for 24 hours 48 buffers are needed.

Default value - by default, 50000 buffers are used.

Parameter:

*number* - maximal number of buffers for historical data.

**Option name: Time Difference**

**Option value: number**

Meaning - the item is used to declare the time difference between clock indication of an Asix system computer and a DMS computer. Even if the time displayed on the DMS computer is the same as the time of the Asix system computer the time received by means of a serial interface, as a number of seconds from 01.01.1970, may be different from the actual time of the DMS computer.

Default value - by default, this time is equal 3600 seconds.

Parameter:

*number* - time difference in seconds.

**Option name: Report**

**Option value: name**

Meaning - the item allows to declare a place where the report read from a DMS computer will be sent by appropriate setting of pseudo-variable *REPORT*. As the purpose place of the report:

- printer name,
- disk file name

may be entered.

Default value - by default, the report transferred from a DMS computer is sent to a printer (LPT1).

Parameter:  
*name* - printer name or disk file name.

**Option name: Max history**

**Option value: *number***

Meaning - the item allows to declare a period of time counted from the current moment backwards, for which historical data in DMS computer memory will be read.

Default value - by default, it is a period of 35 days.

Parameter:  
*number* - time in days.

### EXAMPLE 1

In the example a DMS computer named SIERSZA is defined. It is connected to the COM2 port. If the difference between the Asix system computer time and the DMS computer time will exceed 5 seconds, then a clock synchronization occurs.

Channel declaration:

*Chanel / Name:* SIERSZA

*Driver:* DMS500

*Chanel parameters:*

*Serial port:* COM2

Driver parameters:

*Option name:* Auto sync

*Option value:* 50

### EXAMPLE 2

In the example the DMS computers with the following names are defined:

SIERSZA1,  
 SIERSZA2,  
 SIERSZA3,  
 SIERSZA4,  
 SIERSZA5,  
 SIERSZA6.

The DMS computers named as follows are connected to the COM2 port of the Asix system computer:

SIERSZA1,  
 SIERSZA2,  
 SIERSZA3.

The DMS computers named as follows are connected to the COM3 port of the Asix system computer:

SIERSZA4,  
 SIERSZA5.

The DMS computer named SIERSZA6 is connected to the COM4 port.

The COM2 and COM4 ports work with a speed of 19200 baud. The COM3 port works with a speed of 9600 baud.

## Communication Drivers

The clocks of all DMS computers (except SIERSZA6) are synchronized with the clock of the Asix system computer when the time difference exceeds 60 seconds. The clock of the DMS computer SIERSZA6 is not synchronized.

Report read by means of the pseudo-variable REPORT are printed on the printer LPT1: An exception is the DMS computer SIERSZA6, the reports of which are stored in the file C:\RAP\SIERSZA6.RAP.

Channel declaration:

*Chanel / Name:* SIERSZA1  
*Driver:* DMS500  
*Chanel parameters:*  
*Serial port:* COM2

*Chanel / Name:* SIERSZA2  
*Driver:* DMS500  
*Chanel parameters:*  
*Serial port:* COM2

*Chanel / Name:* SIERSZA3  
*Driver:* DMS500  
*Chanel parameters:*  
*Serial port:* COM2

*Chanel / Name:* SIERSZA4  
*Driver:* DMS500  
*Chanel parameters:*  
*Serial port:* COM3

*Chanel / Name:* SIERSZA5  
*Driver:* DMS500  
*Chanel parameters:*  
*Serial port:* COM3

*Chanel / Name:* SIERSZA6  
*Driver:* DMS500  
*Chanel parameters:*  
*Serial port:* COM4

Driver parameters:

Default values for all DMS computers (parameters declared on *Driver parameters* tabs for one of the channel are valid for all the channels - unless you declare different values for the parameters on *Driver parameters* tabs for individual channel):

*Option name:* Auto Sync  
*Option value:* 0

*Option name:* REPORT  
*Option value:* C:\RAP\SIERSZA6.RAP

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**☑ Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**☑ Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**☑ Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

**☑ Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.27 DNP3 - Driver of Distributed Network Protocol Version 3 for Electrical Power Engineering Control and Supervision Systems

### Driver Use

The following description concerns the DNP3 driver. The driver services the communication with the use of serial links, TCP and UDP protocol.

### Declaration of Transmission Channel

Declaration of the transmission channel using the DNP3 driver requires a channel with the following parameters to be added to the Current data module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* DNP3

**DNP3** tab:

**Channel parameters:**

*communication\_parameters, local\_address, remote\_address*

where:

*communication\_parameters* - determines the way of communication;  
*local\_address* - DNP3 address used by the driver;  
*remote\_address* - DNP3 address of remote device.

Communication parameters for serial links are as follows:

*serial\_port\_name[:baud:data\_bits:parity:stop\_bits]*

**Example:** "COM1:19200:8:n:1"

The parity may be the letter "n" (no parity bit), p (even parity), o (odd parity check), m (mark), s (space).

If the connection is supposed to work with parameters 9600:8:n:1, it is possible to omit them.

Communication parameters for the TCP and UDP protocol have the form:

***protocol\_name:ip\_address[:port\_number][/[ip\_local\_address]:local\_port]***

where:

<i>protocol_name</i>	- TCP or UDP;
<i>ip_address</i>	- IP address of a remote device (the name in DNS system is also possible);
<i>port_number</i>	- port number on which a remote device waits for communication;
<i>ip_local_address</i>	- local address of the network card to be used to establish connection;
<i>local_port</i>	- local port to be used to establish connection.

If the device waits for the port 2000 to establish connection, the *port\_number* parameter can be omitted.

The parameter *ip\_local\_address* is declared when there are multiple network adapters that can be used and only one of them can be used.

The *local\_port* parameter is declared when the remote device waits for establishing connection with the use of one specific port.

### EXAMPLE

Examples of channel declarations:

DNP3, COM1, 3, 4

This declaration is used to connect with a device of the address 4 via a serial port. The local address it is 3.

DNP3,TCP:192.168.0.15, 3, 4

This declaration is used to connect with a remote device with the address 192.168.0.15 with the use of TCP protocol. The remote device waits for connection on the port 2000, so it has been omitted. The remote device has the address DNP3 4. The local DNP3 address is 3.

DNP3,UDP:192.168.0.15|:20000, 3, 4

This declaration is used to connect with a remote device with the address 192.168.0.15 with the use of UDP protocol. The remote device waits for UDP packet reception on the port 20000, so it has been omitted. The device sends answers to received packets to the port 20000 and therefore the port on which the driver waits for these answers.

The device directs answers to received packets to the port 20000 and then the port, on which the driver waits these responses, has been declared openly. The remote device has the address DNP3 4. The local DNP3 address is 3.

The channel parameters are specified in the item ***Driver/channel Parameters***.

## Addressing the Process Variables

The following variables are available:

*Tab. The set of variables for the driver DNP3.*

Symbol	Description	Size of Conversion Function
In	The state of binary input with the address n. If the status bit "online" is set to 0 or the "communication lost" status bit is set, the variable receives bad quality.	1 byte

Communication Drivers

In.Onl or In.0	The "online" bit of the binary input with the address n.	1 byte
In.RS or In.1	The "restart" bit state of the binary input with the address n.	1 byte
In.CL or In.2	The state of "communication lost" status bit of the binary input with the address n.	1 byte
In.RF or In.3	The state of "remote forced" status bit of the binary input with the address n.	1 byte
In.LF or In.4	The state of "local forced" status bit of the binary input with the address n.	1 byte
In.CF or In.5	The state of "chatter filter" status bit of the binary input with the address n.	1 byte
In.Res or In.6	The state of "reserved" status bit of the binary input with the address n.	1 byte
In.ST or In.7	The binary input state with the address n. The quality of the variable with this address does not depend on bits: "communication lost" and "online".	1 byte
In.FL	The state of all status bits (flags) of binary input with the address n. Bits have the same numbers as in the addresses In.m (number m) above. The quality of the variable with this address does not depend on bits: "communication lost" and "online".	1 byte
IGn-m	The group of binary inputs with the addresses from n to m inclusive. The state of input n takes the bit 0 and the following bits correspond to the following inputs. If the status bit "communication lost" is set for any input or the "online" status bit is not set - the quality of variable is bad. As for individual binary inputs, you can add a dot and the number or the name of status bit to receive the group of values of these statuses.	1, 2 or 4 bytes depending on the number of bits
DIn	The state of 2-bit binary input with the address n. Addressing of inputs is the same as for binary inputs In, but the status bit "reserved" has the value of the second input bit.	1 byte
On	The state of binary output with the address n. If the status bit "online" has the value of 0 or the status bit "communication lost" is set - the variable has the value bad.	1 byte
On.Onl or On.0	The state of bit "online" of binary output with the address n.	1 byte
On.RS or On.1	The state of bit "restart" of binary output with the address n.	1 byte
On.CL or On.2	The state of status bit "communication lost" of binary output with the address n.	1 byte
On.RF or On.3	The state of status bit "remote forced" of binary output with the address n.	1 byte
On.LF or On.4	The state of status bit "local forced" of binary output with the address n.	1 byte
On.Res1 or On.5	The state of status bit "reserved" of binary output with the address n.	1 byte
On.Res2 or On.6	The state of status bit "reserved" of binary output with the address n.	1 byte
On.ST or On.7	The state of binary output with the address n. The quality of the variable with this address is not depended on the bits: "communication lost" i "online".	1 byte
On.FL	The state of all status bits (flags) of binary output with the address n. Bits have the same numbers as in the addresses In.m (number m) above. The quality of the variable with this address does not depend on bits: "communication lost" and "online".	1 byte
On. Code	Determining how to control the binary output. One of the following values is written to the variable: 0 - no operation	1 byte

	<p>1 - set output for the time specified by the variable with the address On.OnTime, and next reset it for the time specified by the variable OffTime (pulse on)</p> <p>2 - reset output for the time specified by the variable with the address On.OffTime, and next set it for the time specified by the variable On.OnTime (pulse off)</p> <p>3 - set output to 1 (latch on)</p> <p>4 - set output to 0 (latch off)</p>	
On.Close	Determining how to control the output: 0 - no operation or output activation 1 - perform the operation "close" 2 - perform the operation "trip"	1 byte
On.Count	The value of the variable determines how many times the specified control have to be performed (by default, 1)	1 byte
On. Queue	If the variable is non-zero, the control operation is set at the end of queue for re-execution (by default, 0)	1 byte
On.Clear	If the variable is non-zero, the existing elements will be removed from the control queue (by default, 0)	1 byte
On.OnTime	It specifies the setting time of binary output to 1 for operations: "pulse on" and "pulse off"	4 bytes
On.OffTime	It specifies the setting time of binary output to 0 for operations: "pulse on" i "pulse off"	4 bytes
On.Exec	Writing the appropriate value to the variable initiates the control. The values are as follows: 2 - direct operate 3 - direct operate, no acknowledge 4 - operate (the operation have to be acknowledged by the operation <i>select</i> (5)) 5 - select 6 - the driver will perform first the operation (5), and next (4)	1 byte
On. CtlSts	After sending control command the device sends the status of request back. The variable gets the value equal to this status. Possible status values: 0 - control request has been accepted, initiated or queued 1 - request has been rejected because the request of the operation "operate" (4). It was sent too late after sending the request "select" (5) 2 - request was rejected because the operation "operate" (4) was not preceeded by an operation "select" (5) 3 - request was rejected because of formatting errors 4 - request was rejected because the binary output data does not perform such an operation 5 - request was rejected because the queue is full or the output is active 6 - request was rejected because of hardware problems	1 byte
AIn	The state of analog input with the address n. If the status bit "online" is set to 0 or the status bit "communication lost" is set, the variable takes the quality bad.	2 or 4 bytes, depending on the size of the data in device
AIn.Onl or AIn.0	The state of analog input with the address n. If the status bit "online" has the value of 0 or the status bit "communication lost", the variable has the quality bad. The state of bit "online" of analog input with the address n.	1 byte
AIn.RS or AIn.1	The state of bit "restart" of analog input with the address n.	1 byte

## Communication Drivers

AIn.CL or AIn.2	The state of status bit "communication lost" of analog input with the address n.	1 byte
AIn.RF or AIn.3	The state of status bit "remote forced" of analog input with the address n.	1 byte
AIn.LF or AIn.4	The state of status bit "remote forced" of analog input with the address n.	1 byte
AIn.OR or AIn.5	The state of status bit of analog input with the address n.	1 byte
AIn.RC or AIn.6	The state of status bit "reference check" of analog input with the address n.	1 byte
AIn.Res or AIn.7	The state of status bit "reserved" of analog input with the address n.	1 byte
AIn.FL	The state of all status bits (flags) of analog input with the address n. Bits have the same numbers as in addresses above for AIn.m (number m) addressing. The variable quality with this address is not depended on the bits: "communication lost" and "online".	1 byte
AIn.Val	The value of analog input independed on status bits: "communication lost" and "online".	2 or 4 bytes, depending on the size of the data in device
AOn	The state of analog output with the address n. If the status bit "online" has the value of 0 or the status bit "communication lost" is set, the variable takes the quality bad.	2 or 4 bytes, depending on the size of the data in device
AOn.Onl or AOn.0	The state of bit "online" of analog output with the address n.	1 byte
AOn.RS or AOn.1	The state of bit "restart" of analog output with the address n.	1 byte
AOn.CL or AOn.2	The state of status bit "communication lost" of analog output with the address n.	1 byte
AOn.RF or AOn.3	The state of status bit "remote forced" of analog output with the address n.	1 byte
AOn.Res1 or AOn.4	The state of status bit "reserved" of analog output with the address n.	1 byte
AOn.Res2 or AOn.5	The state of status bit "reserved" of analog output with the address n.	1 byte
AOn.Res3 or AOn.6	The state of status bit "reserved" of analog output with the address n.	1 byte
AOn.Res4 or AOn.7	The state of status bit "reserved" of analog output with the address n.	1 byte
AOn.FL	The state of all status bits (flags) of analog output with the address n. The bits have the same numbers as in addresses above of AIn.m (number m) addressing. The variable quality with that address is not depended on the bits: "communication lost" and "online".	1 byte
AOn.CtIVal	The new value of analog output.	2 or 4 bytes, depending on the size of the data in device
AOn.Exec	Writing the appropriate value to the variable initiates the control. The values are as follows: 2 - direct operate 3 - direct operate, no acknowledge 4 - operate (the operation have to be acknowledged by the operation <i>select</i> (5)) 5 - select 6 - the driver will perform first the operation <i>select</i> (5), and next <i>operate</i> (4)	1 byte
AOn.CtISts	The control operation control. The meaning of the status is the same as for binary outputs.	1 byte
IIN	Internal Indications - set of tags attached by the device to any transmitted information at the application layer.	2 bytes
IIN.BCAST or IIN.0	The device received a message of "broadcast" type.	1 byte
IIN.C1 or IIN.1	The device has events of the 1 class in the buffer.	1 byte
IIN.C2 or IIN.2	The device has events of the class 2 in the buffer.	1 byte
IIN.C3 or IIN.3	The device has events of the class 3 in the buffer.	1 byte

IIN. TSYNC or IIN.4	The device requires time synchronization.	1 byte
IIN.LOC or IIN.5	Some or all the binary inputs are in local mode.	1 byte
IIN. DTRBL or IIN.6	The device signals problems.	1 byte
IIN. RSTRT or IIN.7	There was a device restart.	1 byte
IIN. NIMPL or IIN.8	Function not implemented.	1 byte
IIN. OUNK or IIN.9	Unknown object.	1 byte
IIN. INVPARAM or IIN.10	Invalid parameter.	1 byte
IIN. OVRFL or IIN.11	Event buffer overflow.	1 byte
IIN. EXEC or IIN.12	Control request is correct but the other control is being executed at the moment.	1 byte
IIN. CFGCORR or IIN.13	The device configuration is damaged.	1 byte
IIN.Res1 or IIN.14	Tag reserved.	1 byte
IIN.Res1 or IIN.15	Tag reserved.	1 byte

## Control

The control of binary outputs means sending the set of data describing requested operation (control relay output block) to a device. For individual elements of this set the variables of the following addresses can be assigned: On.Code, On.Close, On.Clear, On.Queue, On.OnTime and On.OffTime (n means the number of binary output). The variable On.Code has to be defined at least. By writing specific values to these variable the content of sent data set is established. The meaning of these values is described in the section about addressing variables. In addition, you can define a variable On.CtISts, which takes the status of control performing after its sending. Sending the control is initiated by the recording of a specific value to the variable with the address On.Exec.

In the case of analog outputs, define the variable with the address AOn.CtIVal, to which the new value of analog output is written. Writing to the variable with the address AOn.Exec an appropriate value (with the same meaning as the value On.Exec) will initiate the control. The result of the operation will be written to the variable with the address AOn.CtISts.

Control can be realized by performing a complex action, consisting of a series of component operations setting the values of individual variables. The last action in the series should write an appropriate value to the variable with the address On.Exec.

An exemplary action sequence:

```
SET_BITS On.Codej 3_type_variable_name
```

```
SET_BITS On.Exec 2_type_variable_name
```

can be used to set the output with the use of the "direct operate" operation.

## Alarms

The change of DNP3 driver variable value can be linked with the generation of alarms. To do this, the additional parameter - alarm definition should be added to the variable address - including:

- alarm definition may take the form of:

```
ABalarm_no bit_no
```

or

*AXalarm\_no bit\_no*

or

*ALalarm\_no relacja wartość*

and alarms associated with controls which are described later.

AB-type alarms are similar to the bitmap strategy of Asix system. Alarm\_no defines the number of the first alarm. This alarm is associated with the variable value bit\_no bit. Successive bits of variable value are associated with alarms of consecutive numbers. Setting the bit to 1 generates a new alarm and resetting the bit to 0 ends the alarm.

AX alarms operates in the same way as AB alarms, but the definition applies only to one specific bit bit\_no. Other bits are not analyzed. The first bit of the value has the number 0.

The AL-type alarm is generated when the *value* of the variable meets the *relation*. The following relations are possible:

= or ==	- equality
!= or <>	- inequality
>	- greater than
>=	- greater or equal
<	- less than
<=	- less than or equal

**Examples:**

A17[AL300=1:AL301=3]

The alarm 301 will be generated when the value of the variable (switch position) equals 1 (ON). The alarm 301 will be generated when the value of the variable equals 3 (intermediate switch position).

A17[AL500>32,76]

The alarm will be generated when the value of the variable exceeds 32,76.

A17[AX600 1]

Setting the bit no. 1 of the variable value will generate the alarm number 600. The parameter "Global\_alarms" determines the type of generated alarms and if it is set to "Yes", global alarms are generated. As the parameter applies to the all driver, it can be declared in the section [DNP3DRV].

The first parameter of the generated alarm has the value that equals the value of the variable.

**Alarms Associated with Control**

Execution of control of binary and analog output can be associated with the generation of alarms.

These alarms take the form:

CAOalarm\_no - alarm is generated when the operation "latch on" or "pulse on" or analog output control occurred

CAFalarm\_no - alarm is generated when the operation "latch off" or "pulse off" occurred

CATalarm\_no - alarm is generated when the operation "trip" occurred

CACalarm\_no - alarm is generated when the operation "close" occurred

The first parameter of the generated alarm is the name of the user logged into the Asix system.

## Driver/Channel Parameters

**NOTICE:** The driver parameters can be entered in the section with the same name as the name of communication channel or in the section named DNP3DRV. Entering any parameter in the section DNP3DRV allows to avoid entering the same parameter for each device - all the parameters from the section DNP3DRV become default values for all devices. However, if an individual device must have a value other than declared in the section DNP3DRV, such a parameter should be entered in the section with the same name as the communication channel name. The parameters from the communication channel section have a higher priority than the ones from the section DNP3DRV.

The parameters *Log* and *Dump* can contain the string "%s", which will be replaced with the channel name for the parameter *Log* and the serial port name for the parameter *Dump*. Placing the parameter *Log* with string "%s" in the section DNP3DRV will create log file for each device. If the log file is not required for a device, you must specify "-". The parameters defining a time period (e.g. the interval between consecutive queries about new events) have values in milliseconds. The letter designation of time units may be used: d - days, g i h - hours, m - minutes, s - seconds, ms - milisekundy, e.g. 2h 10m 5s - which means 2 hours, 9 minutes and 5 seconds.

**Section name: DNP3DRV**

**Option name: AIPollPd**

**Option value: number**

Meaning: The interval between reading analog inputs. If the parameter equals 0, the analog inputs are not read.

Option value:  
*number* - time in milliseconds.

Default value: 0.

**Section name: DNP3DRV**

**Option name: AppTmo**

**Option value: number**

Meaning: The timeout for device response.

Option value:  
*number* - time in milliseconds (application layer).

Default value: 3000.

**Section name: DNP3DRV**

**Option name: AppRetries**

**Option value: number**

Meaning: Retries number to send requests to the device if the previous ones failed (application layer).

Option value:  
*number*

Default value: 3.

**Section name: DNP3DRV**

**Option name: BIPollPd**

**Option value: number**

Meaning: The time interval between reading the binary inputs. If the parameter has the value of 0, binary inputs are not read. The parameter is used if the device does not inform automatically about the change of binary input states.

Option value:  
*number* - time in milliseconds

Default value: 0

**Section name: DNP3DRV**

**Option name: ConfMode**

**Option value: never/ fragment/ always**

Meaning: The message acknowledge mode in the data link layer:  
**never** - the driver does not require confirmation of messages by the device;  
**fragment** - the driver requires confirmations for messages consisting of multiple parts;  
**always** - the driver always requires confirmation of sent messages.

Default value: never.

**Section name: DNP3DRV**

**Option name: ConfTmo**

**Option value: number**

Meaning: Timeout for the device response (data link layer). The parameter is relevant only if the parameter ConfMode has the value other than "never".

Option value:  
*number* - time in milliseconds

Default value: 2000.

**Section name: DNP3DRV**

**Option name: DFCPd**

**Option value: number**

Meaning: The time interval between sending by the driver the message "Request Link Status" to check the device status after receiving the information that its buffers may overflow - i.e. when the device set bits DFC (data flow control). The parameter of data link layer.

Option value:  
*number* - time in milliseconds

Default value: 500.

**Section name: DNP3DRV**

**Option name: LTestPd**

**Option value: number**

Meaning: The time interval between sending by the driver the message "Test Link". The message is sent when the link is inactive. The parameter is of data link layer. The value of 0 disables the link test.

Option value:  
*number* - time in milliseconds

Default value: 1m.

- Section name: DNP3DRV**
- option name: EventMode**
- Option value: none / poll / detect / unsol**

Meaning: Event service mode:  
**none** - the device does not generate events;  
**poll** - the driver polls cyclically the device for the new events with the interval specified by the parameter EPollPd;  
**detect** - while starting the driver sends a message that allows to determine if the device sends information about new events spontaneously;  
**unsol** - the device on own initiative sends information on new events spontaneously.

Default value: detect.

- Section name: DNP3DRV**
- Option name: EPollPd**
- Option value: number**

Meaning: The time interval between polling the device for new events by the driver.

Option value:  
*number* - time in millisecond

Default value: 5000.

- Section name: DNP3DRV**
- Option name: IntgPd**
- Option value: number**

Meaning: The time interval between the driver polls the device for current state of all data (integrity period). The value of 0 disables the device polling; it is done by the driver at the device start-up time and after device reboot.

Option value:  
*number* - time in milliseconds

Default value: 0.

- Section name: DNP3DRV**
- Option name: MaxFrame**
- Option value: number**

Meaning: The maximum length of the message sent to the device. Parameter of data link layer.

Option value:  
*number*

Default value: 255.

- Section name: DNP3DRV**
- Option name: MaxFragment**
- Option value: number**

Meaning: The maximum length of a fragment of data sent by the device. Transport layer parameter.

Option value:  
*number*

Default value: 2048.

- Section name: DNP3DRV**
- Option name: SendInterval**
- Option value: number**

Meaning: The minimum time length between consecutive messages sent to the device. Parameter of data link layer.

Option value:  
*number* - time in milliseconds

Default value: 50.

**Section name: DNP3DRV**

**Option name: Retries**

**Option value: number**

Meaning: The number of retries of sending messages to the device if the previous attempts failed. The parameter is used if the parameter ConfMode requires confirmations. The parameter of data link layer.

Option value:

*number*

Default value: 3.

**Section name: DNP3DRV**

**Option name: TMeasDly**

**Option value: YES/NO**

Meaning: The parameter determines if the device has the function of transmission delay measurement. The measurement is performed at the driver starting. If the parameter is set to NO, then the measurement is not performed. The delay value can be given by the parameter then.

Default value: YES.

**Section name: DNP3DRV**

**Option name: TSyncDly**

**Option value: number**

Meaning: The parameter determines the length of time that elapses from the moment of data preparation by the driver until the data are received by the driver. The driver automatically measures the delay if the device enables it and the parameter TMeasDly has the value YES.

Option value:

*number*

- time in milliseconds

Default value: 0.

**Section name: DNP3DRV**

**Option name: TSyncMode**

**Option value: NoSync / Master / FMaster / Slave**

Meaning: Time synchronization mode. The parameter can have the following values:

**NoSync** - no time synchronization;

**Master** - the driver sets the time of device when the device requests it;

**FMaster** - the driver sets the time of device periodically with the interval specified by the parameter TSyncPd;

**Slave** - the driver reads the time of device cyclically to determine the difference of times and to give time stamp to the data read from the device that does not have this stamp.

Default value: Master.

**Section name: DNP3DRV**

**Option name: TSyncPd**

**Option value: number**

Meaning: The time interval between setting or reading by the driver the time of a device depending on the parameter TSyncMode.

Option value:

*number*

- time in hours

Default value: 1h.

**Section name: DNP3DRV**

**Option name: Type**

**Option value XServer / Mikronika**

Meaning: The device type. The parameter can be set to "XServer" or "Mikronika". Don't use the parameter for other devices.

Default value: no default value

**Section name: DNP3DRV**

**Option name: UTC**

**Option value: YES/NO**

Meaning: The parameter can take the value NO or YES and determines if the device supports UTC time.

Default value: YES.

**Section name: DNP3DRV**

**Option name: Diag**

**Option value: information\_group\_name\_list**

Meaning: The parameter specifies the type of diagnostic information stored in Log parameter file. This is a list of information group names separated by commas (e.g. alhdr, objs, task, hdrs, rsptmr, tsync, iod).

Default value: no default value

**Section name: DNP3DRV**

**Option name: Log**

**Option value: file\_name**

Meaning: The name of a file to which all the diagnostic information will be written, optionally with the full path to this file. After the file name you can declare the maximum file size (in megabytes). Name and size separate by a comma.

Default value: no default value

**Section name: DNP3DRV**

**Option name: Dump**

**Option value: file\_name**

Meaning: The name of file to which all the information sent to/from the device will be written using the serial interface. After the comma you can declare in the option the maximum file size in megabytes. The parameter can be entered only in the section DNP3DRV. If the driver uses multiple serial ports, then you can specify the string "%s" in the file name. The string will be replaced with the name of the serial port. If you use a sequence of "%s" other characters should be added to the file name, because the file name can not be the same as the name of the serial port (e.g. %sData.dmp).

Default value: no default value

## Exemplary Driver Configuration

Channel 1

Name: DNP3\_1

Driver: DNP3

DNP3 tab:

Channel parameters:

## Communication Drivers

UDP:192.168.0.31,3,4

### Channel 2

Name: DNP3\_2  
Driver: DNP3

#### DNP3 tab:

Channel parameters:  
UDP:192.168.0.32,3,4

### Channel 3

Name: DNP3\_3  
Driver: DNP3

#### DNP3 tab:

Channel parameters:  
UDP:192.168.0.33,3,4

Parameters declared on the tab: Architect > *Fields and Computers* > *Miscellaneous* > *Directly entered options*

Section name	Parameter name	Parameter value
DNP3DRV	Log	%s.log
DNP3DRV	EventMode	Poll
DNP3DRV	EPollPd	10s
DNP3_3	Log	-
DNP3_2	EPollPd	5s

In the example above, three channels were defined: DNP3\_1, DNP3\_2 and DNP3\_3. For the devices DNP3\_1 and DNP3\_2 the following log file will be created: "DNP3\_1.log" and "DNP3\_2.log". The log file will not be created for the device DNP3\_3. All the three devices will be cyclically polled for events - but the devices: DNP3\_1 i DNP3\_3 will be polled every 10 seconds and the device DNP3\_2 every 5 seconds.

## Advanced Driver Configuration

The parameters of a channel are declared in the *Advanced* tab:

#### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:  
YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).  
 The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\* \* \*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.28 DP - Driver of PROFIBUS DP Network Protocol for PROFOboard Board

### Driver Use

The DP driver is used for data exchange in the PROFIBUS network with devices operating according to the PROFIBUS DP standard. The Asix system computer must be provided with the PROFIboard NT communication processor board and with the PROFIboard NT v 5.20 software package of Softing GmbH.

Parameterization of DP driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the DP driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: DP

**DP** tab:

**Channel parameters:**

*card\_no, address, section\_name*

where:

- board\_no* - number of the PROFIboard NT board which is used for communication with a given device (DP slave). In the present version the DP driver can control only one PROFIboard NT board;
- address* - address assigned to the DP device;
- section\_name* - name of section in the application INI file, where the configuration parameters of the given DP device are contained.

The DP driver is loaded as a DLL automatically.

### Configuration of DP Devices

Configuration parameters of each DP device (DP slave) should be declared in a separate section of the application INI file. These parameters should be accessible in the device specification or in the GSD file supplied by the manufacturer with the device. The section name is assigned by the user and must be unique in the application INI file.

The individual configuration parameters are transferred in separate items of the section. Each item is as follows:

*parameter=value*

where:

parameter - type of configuration parameter;  
value - parameter value (in decimal format !)

Types of configuration parameters, which **must be transferred** by the application designer are given below:

IDENT_NUMBER	device identifier	(PNO ident. number)
GROUP_IDENT	group identifier	(group member bits)
USER_PRM_DATA		(prm_user_data)
.. variable number of USER_PRM_DATA items		
.		
USER_PRM_DATA		(prm_user_data)
MODULE_ID		(cfg_data)
.. variable number of MODULE_ID items		
.		
MODULE_ID		(cfg_data)
<i>aat_data</i> parameters are optional and if not declared they are set to 0.		
NUMBER_OF_INPUTS		(aat_data)
NUMBER_OF_OUTPUTS		(aat_data)
OFFSET_OF_INPUTS		(aat_data)
OFFSET_OF_OUTPUTS		(aat_data)

*slave\_data* parameters are optional and if not declared they are omitted.

SLAVE_DATA		(slave_user_data)
.. variable number of SLAVE_DATA items		
.		
SLAVE_DATA		(slave_user_data)

## Addressing the Process Variables

The values transferred from modules connected to the DP device are written to an I/O buffer of the DP driver, in order resulting from arrangement of Input/Output modules in the cassette of the DP device. The addressing process variables requires an indication of:

- buffer type (input buffer or output buffer);
- byte no. (in the buffer), where the value of a given input/output is stored; depending on the type of process variable, the variable value occupies one byte (one-byte type variable) or 2 successive bytes (2-byte type variable);
- type of the variable (one-byte or 2-byte).

The syntax of symbolic address which is used for variables belonging to the DP driver channel is as follows:

<type><index>

where:

<i>type</i>	- type of process variables:
IB	- byte from the input buffer,
IW	- 2 successive bytes from the input buffer treated as a fixed-point unsigned number in INTEL format,
IDW	- 4 successive bytes from the input buffer treated as a double word in INTEL format,
IFP	- 4 successive bytes from the input buffer treated as a floating-point number in INTEL format,
IWM	- 2 successive bytes from the input buffer treated as a fixed-point unsigned number in MOTOROLA format,
IDWM	- 4 successive bytes from the input buffer treated as a double word in MOTOROLA format,
IFPM	- 4 successive bytes from the input buffer treated as a floating-point number in MOTOROLA format,

OB	- byte from the output buffer,
OW	- 2 successive bytes from the output buffer treated as a fixed-point unsigned number in INTEL format,
ODW	- 4 successive bytes from the output buffer treated as a double word in INTEL format,
OFFP	- 4 successive bytes from the output buffer treated as a floating-point number in INTEL format,
OWM	- 2 successive bytes from the output buffer treated as a fixed-point unsigned number in MOTOROLA format,
ODWM	- 4 successive bytes from the output buffer treated as a double word in MOTOROLA format,
OFFPM	- 4 successive bytes from the output buffer treated as a floating-point number in MOTOROLA format;
<i>Index</i>	- number of the byte in the input/output buffer.

**EXAMPLE**

IB9	- 9-th byte from the area of inputs
IW2	- word created from the 2-nd and 3-rd byte of the input area (INTEL format)
IWM2	- word created from the 3-rd and 2-nd byte of the input area (MOTOROLA format)
IDW5	- double word created from the 5-th, 6-th, 7-th and 8-th byte of the input area (INTEL format)
IDWM5	- double word created from the 8-th, 7-th, 6-th and 5-th byte of the input area (MOTOROLA format)

## Driver Configuration

DP driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **DP** section.

- Section name:** DP
  - Option name:** TRANSMISSION\_SPEED
  - Option value:** *network\_no,baud\_id*
- Meaning - the item is used to declare the transmission speed in the PROFIBUS DP network.
- Default value - by default, the transmission speed is set to 1,5 MB.
- Parameter:
- network\_no* - number of the PROFIBUS DP network (in the present version always set to 1),
  - baud\_id* - identifier of transmission speed in a PROFIBUS DP network:
    - 0 - 9,6 kB
    - 1 - 19,2 kB
    - 2 - 93,75 kB
    - 3 - 187,5 kB
    - 4 - 500 kB
    - 5 - 750 kB
    - 6 - 1,5 MB
    - 7 - 3 MB
    - 8 - 6 MB
    - 9 - 12 MB
    - 11 - 45,45 kB

**EXAMPLE**

A declaration of transmission speed of 750 kB:

*Section name:* DP  
*Option name:* TRANSMISSION\_SPEED  
*Option value:* 1, 5

- Section name:** DP
- Option name:** REFRESH\_CYCLE
- Option value:** number

Meaning - the item is designed to declare an interval between successive data readings from buffers of the PROFIBoard NT board to the structures of the DP driver.

Default value - by default, the DP driver reads data from buffers of the PROFIBoard NT board every 0.5 second.

Parameter:  
*number* - number of 0.5-second intervals, which must pass between successive data readings from buffers of the PROFIBoard NT board.

**EXAMPLE**

A declaration of data reading every 1 second:

*Section name:* DP  
*Option name:* REFRESH\_CYCLE  
*Option value:* 2

- Section name:** DP
- Option name:** CONSOLE
- Option value:** YES/NO

Meaning - the item allows creating a console window where the DP driver messages, concerning the status of communication between an Asix system computer and DP devices, are displayed.

Default value - by default, the console window is not created.

- Section name:** DP
- Option name:** LOG\_FILE
- Option value:** file\_name

Meaning - the item allows to define a file where all messages of the DP driver, concerning to the status of communication between an Asix system computer and DP devices, will be written. If the item does not define the full path, then the log file is created in the current directory.

Default value - by default, the log file is not created.

**EXAMPLE**

An example item declaring a transmission channel using the DP protocol for the communication with ET200U no. 7 unit is given below. The following input/output boards are connected to the ET200U (in order of their arrangement in the list):

- Digital Output module (8 outputs) 6ES5 461-8MA11 (identifier 32),
- Analog Input module (4 inputs) 6ES5 464-8ME11 (identifier 83),
- Digital Input module (8 inputs) 6ES5 431-8MA11(identifier 16).
- Digital Output module (8 outputs) 6ES5 461-8MA11 (identifier 32).

## Communication Drivers

Channel declaration:

*Channel / Name:* CHAN1  
*Driver:* DP  
*Channel parameters:* 1,7,SIEM8008

The transmission channel named CHAN1 has the following parameters defined:

- DP protocol,
- communication by means of the PROFIBUS NT board with the number of 1,
- DP device has number 7 in the PROFIBUS DP network,
- configuration parameters of DP device are contained in the section SIEM8008.

The content of the section [SIEM8008] defining the sample configuration of the ET200U is as follows (all the values are decimal):

*Section name:* SIEM8008  
*Option name:* IDENT\_NR  
*Option value:* 32776

*Section name:* SIEM8008  
*Option name:* GROUP\_IDENT  
*Option value:* 0

*Section name:* SIEM8008  
*Option name:* USER\_PRM\_DATA  
*Option value:* 0

*Section name:* SIEM8008  
*Option name:* MODULE\_ID  
*Option value:* 32

*Section name:* SIEM8008  
*Option name:* MODULE\_ID  
*Option value:* 83

*Section name:* SIEM8008  
*Option name:* MODULE\_ID  
*Option value:* 32

*Section name:* SIEM8008  
*Option name:* MODULE\_ID  
*Option value:* 16

In the configuration under consideration the area of inputs has 9 bytes. The meaning of the bytes is as follows:

bytes 1,2	- analog input 1
bytes 3,4	- analog input 2
bytes 5,6	- analog input 3
bytes 7,8	- analog input 4
byte 9	- digital input byte

In the configuration under consideration the area of outputs has 2 bytes. The meaning of the bytes is as follows:

byte 1	- digital output byte (the first module 6ES5 451-8MA11)
byte 2	- digital output byte (the second module 6ES5 451-8MA11)

### EXAMPLE

Example declarations of process variables are given below:

```
# X1 - digital output - 1-st byte of output buffer
X1, OB1, CHAN1, 1, 1, NOTHING_BYTE
# X2 - digital output - 2-nd byte of output buffer
X2, OB2, CHAN1, 1, 1, NOTHING_BYTE
# X3 - digital input - 9-th byte of input buffer
X3, IB9, CHAN1, 1, 1, NOTHING_BYTE
# X4 - analog input 1 - 1-st and 2-nd bytes of input buffer
X4, IW1, CHAN1, 1, 1, NOTHING
# X5 - analog input 2 - 3-rd and 4-th byte of input buffer
X5, IW3, CHAN1, 1, 1, NOTHING
# X 6 - analog input 3 - 5-th and 6-th byte of input buffer
X6, IW5, CHAN1, 1, 1, NOTHING
# X7 - analog input 4 - 7-th and 8-th byte of input buffer
X7, IW7, CHAN1, 1, 1, NOTHING
```

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

## Communication Drivers

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.  
The default value - by default, no time for data validity is set.

## 1.29 DP5412 - Driver of PROFIBUS DP Protocol for CP5412 Card

### Driver Use

The DP5412 driver is used for data exchange in the PROFIBUS network with devices operating according to the PROFIBUS DP standard. The CP5412(A2) communication processor card and the DP-5412 software package (version 4.1 or higher) or the CP5613 card with the SIEMENS DP-5613 package must be installed on the Asix system computer.

Parameterization of DP5412 driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the DP5412 driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: DP5412

**DP5412** tab:

**Channel parameters:**

*card\_no*, *address*

where:

*board\_no* - number of the CP5412 (A2) or CP5613 card used for communication with the given DP device (DP slave). In the present version the DP5412 driver may handle only one CP5412(A2) or CP5613 card,

*address* - address assigned to the DP device.

The DP5412 driver is loaded as a DLL.

### Configuration of DP Devices

The DP devices (DP slave) configuration is performed by the COM PROFIBUS program included in the DP-5412 package.

The application designer must ensure the compatibility of numbers assigned to the DP devices during the DP network configuration with the program COM PROFIBUS and of DP device numbers declared in the Asix application INI file.

## Addressing the Process Variables

The values transferred from modules connected to the DP device are written to an input buffer and to an output buffer of the DP5412 driver, in order according to the arrangement of Input/Output modules in the DP device cassette. The addressing the process variables consists in indication of:

- buffer type (input buffer or output buffer);
- byte no. (in the buffer), where the value of a given input/output is stored; depending on the type of process variable, the variable value occupies one byte (one-byte type variable) or 2 successive bytes (2-byte type variable);
- type of the variable (one-byte or 2-byte).

The syntax of symbolic address which is used for variables belonging to the DP5412 driver channel is as follows:

*<type><index>*

where:

<i>type</i>	- type of process variables:
IB	- byte from the input buffer,
IW	- 2 successive bytes from the input buffer treated as a fixed-point unsigned number in INTEL format,
IDW	- 4 successive bytes from the input buffer treated as a double word in INTEL format,
IFP	- 4 successive bytes from the input buffer treated as a floating-point number in INTEL format,
IWM	- 2 successive bytes from the input buffer treated as a fixed-point unsigned number in MOTOROLA format,
IDWM	- 4 successive bytes from the input buffer treated as a double word in MOTOROLA format,
IFPM	- 4 successive bytes from the input buffer treated as a floating-point number in MOTOROLA format,
OB	- byte from the output buffer,
OW	- 2 successive bytes from the output buffer treated as a fixed-point unsigned number in INTEL format,
ODW	- 4 successive bytes from the output buffer treated as a double word in INTEL format,
OFP	- 4 successive bytes from the output buffer treated as a floating-point number in INTEL format,
OWM	- 2 successive bytes from the output buffer treated as a fixed-point unsigned number in MOTOROLA format,
ODWM	- 4 successive bytes from the output buffer treated as a double word in MOTOROLA format,
OFPM	- 4 successive bytes from the output buffer treated as a floating-point number in MOTOROLA format;
<i>Index</i>	- number of the byte in the input/output buffer.

### EXAMPLE

IB9	- 9-th byte from the area of inputs
IW2	- word created from the 2-nd and 3-rd byte of the area of inputs (INTEL format)
IWM2	- word created from the 3-rd and 2-nd byte of the area of inputs (MOTOROLA format)
IDW5	- double word created from the 5-th, 6-th, 7-th and 8-th byte of the area of inputs (INTEL format)
IDWM5	- double word created from the 8-th, 7-th, 6-th and 5-th byte of the area of inputs (MOTOROLA format)

## Driver Configuration

DP5412 driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **DP5412** section.

- Section name: DP5412**
- Option name: REFRESH\_CYCLE**
- Option value: number**

Meaning - the item used to declare an interval between successive data readings from buffers of the CP5412(A2) or CP5613 CARD to the DP5412 driver structures.

Default value - by default, the DP5412 driver reads data from buffers of the CP5412(A2) or CP5613 card every 0.5 second.

Parameter:  
*number* - number of 0.5-second intervals, which must pass between successive data readings from buffers of the CP5412 (A2) or CP5613 card.

A declaration of data reading every 1 second:

*Section name:* CTLOGO  
*Option name:* REFRESH\_CYCLE  
*Option value:* 2

- Section name: DP5412**
- Option name: CONSOLE**
- Option value: YES/NO**

Meaning - the item allows to create a console window, where the DP5412 driver messages, concerning to status of communication between an Asix system computer and DP devices, are displayed.

Default value - by default, the console window is not created.

- Section name: DP5412**
- Option name: LOG\_FILE**
- Option value: file\_name**

Meaning - the item allows to define a file where all the DP5412 driver messages, concerning to the status of communication between an Asix system computer and DP devices, will be written. If the item does not define the full path, then the log file is created in the current directory.

Default value - by default, the log file is not created.

### EXAMPLE

An example item declaring a transmission channel using the DP5412 protocol for the communication with ET200U no. 7 is given below. The following input/output modules are connected to the ET200U (in order of their arrangement in the list):

- Digital Output module (8 outputs) 6ES5 461-8MA11,
- Analog Input module (4 inputs) 6ES5 464-8ME11,
- Digital Input module (8 inputs) 6ES5 431-8MA11.

- Digital Output module (8 outputs) 6ES5 461-8MA11.

Channel declaration:

*Channel / Name:* CHAN1  
*Driver:* DP5412  
*Channel parameters:* 1,7

The transmission channel named CHAN1 has the following parameters defined:

- DP5412 protocol,
- communication via the CP5412 (A2) card no. 1,
- DP device is assigned no. 7 in the PROFIBUS DP network.

In the considered configuration the area of inputs has 9 bytes. The meaning of the bytes is as follows:

- bytes 1,2 - analog input 1 (module 6ES5 431-8ME11),
- bytes 3,4 - analog input 2 (module 6ES5 431-8ME11),
- bytes 5,6 - analog input 3 (module 6ES5 431-8ME11),
- bytes 7,8 - analog input 4 (module 6ES5 431-8ME11),
- byte 9 - digital input byte (module 6ES5 431-8MA11).

In the considered configuration the area of outputs has 2 bytes. The meaning of the bytes is as follows:

- byte 1 - digital output byte (the first module 6ES5 451-8MA11)
- byte 2 - digital output byte (the second module 6ES5 451-8MA11)

Exemplary declarations of process variables are given below:

```
# X1 - digital output – 1-st byte of output buffer
X1,    OB1,  CHAN1,  1,  1, NOTHING_BYTE
# X2 - digital output - 2-nd byte of output buffer
X2,    OB2,  CHAN1,  1,  1, NOTHING_BYTE
# X3 - digital input - 9-th byte of input buffer
X3,    IB9,  CHAN1,  1,  1, NOTHING_BYTE
# X4 - analog input 1 - 1-st and 2-nd bytes of input buffer
X4,    IW1,  CHAN1,  1,  1, NOTHING
# X5 - analog input 2 - 3-rd and 4-th byte of input buffer
X5,    IW3,  CHAN1,  1,  1, NOTHING
# X6 - analog input 3 - 5-th and 6-th byte of input buffer
X6,    IW5,  CHAN1,  1,  1, NOTHING
# X7 - analog input 4 - 7-th and 8-th byte of input buffer
X7,    IW7,  CHAN1,  1,  1, NOTHING
```

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:  
YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).  
 The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).  
 The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO  
 The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.  
 The default value - by default, no time for data validity is set.

## 1.30 DSC - Driver of DSC PLC Protocol

### Driver Use

The DSC driver is used for data exchange between an Asix system computer and the DSC 2000 PLC. The data exchange is performed by means of standard serial interfaces of the Asix system computer.

The cooperation of the Asix system with the DSC 2000 PLC does not require any controller's program adaptation.

Parameterization of DSC driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the DSC driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* DSC

**DSC** tab:

Channel parameters:

*id,port*

where:

*id* - number assigned to the DSC PLC,  
*port* - name of the serial port e.g. COM1.

### EXAMPLE

An example declaration of transmission channel based on the DSC Protocol.

*Channel / Name:* CHAN1  
*Driver:* DSC  
*Channel parameters:* 5,COM3

The logical channel named CHAN1 has the following parameters defined:  
- DSC protocol,  
- data exchange is performed with the controller no. 5,  
- COM3 serial port is used for data exchange.

### Addressing the Process Variables

The syntax of symbolic address which is used for variables belonging to the DSC driver channel is as follows:

*I<index>*

where:

- I* - symbol of the variable type, the same for all process variables of the DSC protocol;
- index* - **number in hexadecimal format** identifying the process variable. Legal numbers are only those specified in the *Nummer* item in "Beschreibung der Rechnerschnittstelle", page 3.

All the variables except variables I10 and I11 (status of alarms) are variables the values of which may be read and written. The status of alarms may be read only.

Values of all process variables of the DSC 2000 controller are sent to the Asix system in the form of 16-bit fix-point unsigned number. This principle is valid for floating-point and fixed-point variables. For that reason in order to show a floating-point value of process variable it is necessary to convert the value received from the controller to the floating-point format by means of a conversion function (most often ANALOG\_FP).

### EXAMPLE

Examples of declaration of process variables:

```
# set value of chlorine - floating-point number, two decimal places
X1, I17, CHAN1, 1, 1, ANALOG_FP,0,1000,0.0,10.0
```

```
# set value of pH - floating-point number, two decimal places
X2, I2A, CHAN1, 1, 1, ANALOG_FP,0,1000,0.0,10.0
```

```
# alarms and flags 1 - 16-bit fixed-point number
X3, I10, CHAN1, 1, 1, NOTHING
```

```
# alarms and flags 2 - 16-bit fixed-point number
X4, I11, CHAN1, 1, 1, NOTHING
```

```
# access code to the operator panel - 16-bit fixed-point number
X5, I15, CHAN1, 1, 1, NOTHING
```

The DSC driver is loaded as a DLL automatically.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.31 DXF351 - Driver of Compart XF351 Device Protocol

### Driver Use

The DXF351 driver is used for data exchange between Compart DXF351 devices of Endress+Hauser and an Asix system computer. The communication is performed by using serial interfaces in the RS232C standard.

Compart DXF351 must be set to the following mode:

```
RS 2323 USAGE - PRINTER
DEVICE ID     - any
BAUD RATE    - 9600
PARITY       - NONE
HANDSHAKE    - NONE
```

Settings in PRINT LIST:

```
ERRORS - NO
ALARMS - NO
```

The other items - freely chosen:

```
PRINT INTERVAL - 00:01 (data transfer every 1 minute)
```

Parameterization of DXF351 driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the DXF351 driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

```
Name: logical name of the transmission channel
Driver: DXF351
```

**DXF351** tab:

*Channel parameters:*

```
port
```

where:

```
port - port name: COM1, COM2 etc.
```

#### EXAMPLE

The logical channel CHAN1 declaration working according to the DXF351 protocol on the COM2 port is as follows:

```
Channel / Name: CHAN1
Driver: DXF351
Channel parameters: COM2
```

The DXF351 driver is loaded as a DLL automatically.

## Addressing the Process Variables

The syntax of symbolic address which is used for variables belonging to the DXF351 driver channel is as follows:

*P<index>*

where:

*index* - number of measurement on the list PRINT LIST (see the table below).

**Table 14. PRINT LIST.**

P1	HEAT FLOW
P2	HEAT TOTAL
P3	HEAT GRAND TOTAL
P4	MASS FLOW
P5	MASS TOTAL
P6	MASS GRAND TOTAL
P7	COR. VOLUME FLOW
P8	COR. VOLUME TOTAL
P9	COR. VOL. GRAND TOTAL
P10	VOLUME FLOW
P11	VOLUME TOTAL
P12	VOL. GRAND TOTAL
P13	TEMPERATURE 1
P14	TEMPERATURE 2
P15	DELTA TEMPERATURE
P16	PROCESS PRESSURE
P17	DENSITY
P18	SPEC. ENTHALPY
P19	VISCOSITY
P20	REYNOLDS NUMBER

DXF351 transmits only these parameters, which are enclosed to the list **PRINT LIST** during configuring the Compart DXF351 device (group COMMUNICATION).

**Raw values of all process variables are of FLOAT type.**

An example of variable declarations:

```
X1, Mass Flow ,           P4,  CHAN1, 1, 1, NOTHING_FP
X2, Volume Flow,         P10, CHAN1, 1, 1, NOTHING_FP
X3, Temperature 1,      P13, CHAN1, 1, 1, NOTHING_FP
```

## Driver Configuration

DXF351 driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **DXF351** section.

- Section name: DXF351**
- Option name: LOG\_FILE**
- Option value: file\_name**

Meaning - the item allows to define a file where all diagnostic messages of the DXF351 driver and information about the content of telegrams received by the SPA driver will be written. If the item does not define the full path, then the log file is created in the current directory. The log file should be used only while the Asix start-up.

Default value - by default, the log file is not created.

### EXAMPLE

*Section name:* DXF351  
*Option name:* LOG\_FILE  
*Option value:* D:\ASIX\DXF.LOG

- Section name: DXF351**
- Option name: CHAR\_TIMEOUT**
- Option value: number**

Meaning - the item allows to determine the maximal time, which may pass between successive characters of a data block from DXF351. After having exceeded this time the DXF351 driver assumes that message as finished and begins analyzing the message content.

Default value - by default, the item is set to 600 (millisecond).

Parameter:  
*number* - number in milliseconds.

- Section name: DXF351**
- Option name: LIMIT\_OF\_ERRORS**
- Option value: number**

Meaning - the item allows to define a number of successive transmission errors, after which an error status of measurement is set.

Default value - by default, the item has a value of 3.

Parameter:  
*number* - number of successive transmission errors, after which an error status is set.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

- Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.
- Option value:  
YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

- Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.
- Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

- Purpose - with this option deployed, local control in a channel will be disabled.
- Option value:  
YES / NO
- The default value - NO.

### **Data Validity Period**

- Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.
- Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.
- The default value - by default, no time for data validity is set.

## 1.32 E2tangoTcpip - Protocol Driver for e2TANGO Controllers by Elektrometal Energetyka

### Driver Use

The E2tangoTcpip driver is used to exchange data between the ASIX system and the e<sup>2</sup>TANGO controllers made by Elektrometal Energetyka. Data exchange is carried out via Ethernet.

### Declaration of Transmission Channel

Declaration of the transmission channel operating according to the E2tangoTcpip driver protocol requires adding a channel with the following parameters to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: E2tangoTcpip

**E2tangoTcpip / Channel Parameters** tab:

*Channel parameters*:

*Device Number in the MODBUS Network* - number assigned to the controller in the Modbus network;

*IP Address of the Device* - IP Address of the Controller;

*Port* - TCP port number: 502, 10502, 10503 or 10504. By default it is 502;

*Alarms Offset* - number added to every number of alarm indicated in this channel. By default it is 0;

### EXAMPLE

*Device Number in the MODBUS Network*: 10

*IP Address of the Device* - 10.10.105.21

*Alarms Offset* - 100

### Declaration of Variables

The driver provides the following symbolic addresses:

**CS** - single Coil Status (used only to execute control)

**HR** - single holding register

**HRL** - next 2 holding registers treated as a double word in INTEL syntax

**HRF** - next 2 holding registers treated as a floating point number in INTEL syntax

**HRLM** - next 2 holding registers treated as a double word in MOTOROLA syntax

**HRFM** - next 2 holding registers treated as a floating point number in MOTOROLA syntax

**V1** - virtual variable used to show the Modbus exception number during the last CS type control. 0 indicates that the control was completed without a Modbus exception (variable readout only)

**V2** - virtual variable used to indicate the status of connection with the controller: 1 indicates that the connection with the controller o.k., 0 indicates lack of connection

## Declaration of Events

The definitions of events indicated by the controller are given in an xml file provided by the controller manufacturer.

The name of the xml file and name of the folder in which the file is stored are declared in the driver parameters, however the declarations could be overridden by settings in the channel parameters (e.g. if controllers employ different event definitions).

## Driver Parameters

The E2tangoTcip driver parameters are declared in the E2tangoTcip/Driver Parameters... tabs:

- creating a log file,
- the log file size,
- log of telegrams,
- reception timeout,
- name of the folder with the event definitions file,
- name of the xml file with event definitions,
- event check period,
- time synchronization period with the controller,
- time type used for time synchronization with the controller,
- IP address of the computer card, with the use of which the controller is operated,
- maximum number of registers in a single request,
- number of repetitions in case of a transmission error.

The names of the items related with the log file refer to the convention used in other ASMEN drivers.

### **Log File**

Purpose: the text log file to which driver status messages are stored is used for diagnostic purposes.

The default value: By default, the log file is not created.

Parameter:

*Log\_file\_name* - the size of the log file in MB.

### **Log File Size**

Purpose: this position is used to determine the size of the log file defined using the *Plik logu.* option

The default value: By default the log file size is 10 MB.

Parameter:

*number* - the size of the log file in MB.

**Log of Telegrams**

Purpose: This position allows writing the contents of messages communicated between the driver and the controllers to the log file (declared using the *Log File* item) The item in question should only be used during the ASIX system start-up.

The default value: NO.

Parameter:  
*logical value* - YES/NO

 **Response Timeout**

Purpose: This item specifies the maximum time (in milliseconds) throughout which the driver waits for a response from the controller.

The default value: The default value is 500 milliseconds.

Parameter:  
*number* - value in milliseconds

 **Number of Repetitions**

Purpose: This item is used to specify the number of request repetitions if there is an error/no response from the controller. After the repetition number has been exhausted, the connection with the controller is closed and re-established.

The default value: The default value is 3.

Parameter:  
*number* - number of repetitions in the case of errors/lack of response.

 **IP Address of the Computer**

Purpose: This item is used to specify the IP address of the computer's network card, with the use of which all driver channels communicate with the controller.

The default value: if this item is not defined, the system network card will be used.

Parameter:  
*IP\_address* - network card address in IPv4 notation

 **Max Number of Registers**

Purpose: The item is used to specify the maximum number of registers, whose values can be subject to a single request from the driver.

The default value: 123.

Parameter:  
*number* - maximum number of registers in a single request

 **Definition Files Path**

Purpose: This option is used to specify the name of the folder in which the driver will search for files with definitions of events.

The default value: none.

Parameter:  
*path* - name of the folder with event definitions files.

**Event Definitions File**

Purpose: This option is used to specify the name of the file with event definitions. The file is searched for in the folder specified by the option *Definitions files path*.

The default value: none.

Parameter:  
*name* - name of the event definitions file.

**Event Check Period**

Purpose: The option is used to specify the interval between subsequent checks of the presence of new events. The value set to 0 means that new events will not be checked.

The default value: 10 seconds.

Parameter:  
*number* - the time between readings in seconds.

**Time Synchronization**

Purpose: The option is used to specify the interval between subsequent records of date stamp in the controller. The value set to 0 means that the time will not be synchronized.

The default value: 0 (no time synchronization).

Parameter:  
*number* - time synchronization interval in minutes.

**UTC Time**

Purpose: This option is used to specify time type (local or UTC) recorded to the controller during the time synchronization operation.

The default value: NO.

Parameter:  
*logical value* - YES/NO

**EXAMPLE**

Example of the driver parameters declaration.

*Log File:* d:\tmp\CtE2tangoTcip\vecl.log  
*Log File Size:* 20  
*Log of Telegrams:* YES  
*IP Address of the Computer:* 10.10.110.24  
*Definition Files Path:* d:\EventsDef  
*Event Definitions File:* zdarzenia.xml

**Channel Parameters**

The parameters of a channel are declared in the tabs: **Channel 2 parameters, Channel parameters - events, Diagnostics channel parameters:**

- creating a log file,
- the log file size,
- log of telegrams,

- reception timeout,
- name of the folder with the event definitions file,
- name of the xml file with event definitions,
- event check period,
- time synchronization period with the controller,
- time type used for time synchronization with the controller,
- IP address of the computer card, with the use of which the controller is operated,
- maximum number of registers in a single request,
- number of repetitions in case of a transmission error.

**Log File**

Purpose: The text log file to which the contents of telegrams sent and received in a given channel are stored is used for diagnostic purposes.

The default value: By default the item takes the value defined in the driver options.

Parameter:  
*log\_file\_name*

**Log File Size**

Purpose: This position is used to determine the log file size defined using the *Log file* option.

The default value: By default the item takes the value defined in the driver options.

Parameter:  
*number* - the size of the log file in MB.

**Log of Telegrams**

Purpose: This position allows writing the contents of telegrams communicated in a given channel to the log file. The item value overrides the setting specified by the item of the same name present in the driver parameters. The item in question should only be used during the ASIX system start-up.

The default value: By default the item takes the value defined in the driver section.

Parameter:  
*logical value* - YES/NO

**Response Timeout**

Purpose: this specifies the maximum time throughout which a given channel waits for a response from the controller. The item value overrides the setting specified by the item of the same name present in the driver parameters.

The default value: By default the item takes the value defined in the driver parameters.

Parameter:  
*number* - time out in milliseconds

**Number of Repetitions**

Purpose: This item is used to specify the number of request repetitions if there is an error/no response from the controller. After the repetition number has been exhausted, the connection with the controller is closed and re-established. The item value overrides the setting specified by the item of the same name present in the driver parameters.

The default value: By default the item takes the value defined in the driver parameters.

Parameter:  
*number* - number of repetitions in the case of errors/lack of response.

**IP Address of the Computer**

Purpose: this position is used to specify the IP address of the computer's network card, with the use of which the channel communicates with the controller. The item value overrides the setting specified by the item of the same name present in the driver parameters.

The default value: By default the item takes the value defined in the driver parameters.

Parameter:  
*IP\_address* - network card address in IPv4 notation

**Max Number of Registers**

Purpose: The option is used to specify the maximum number of registers, whose values can be subject to a single request in given channel from the driver. The item value overrides the setting specified by the item of the same name present in the driver parameters.

The default value: By default the item takes the value defined in the driver parameters.

Parameter:  
*number* - maximum number of registers in a single request

**Definition Files Path**

Purpose: This option is used to specify the name of the folder in which the driver will search for files with definitions of events for given channel. The item value overrides the setting specified by the item of the same name present in the driver parameters.

The default value: By default the item takes the value defined in the driver parameters.

Parameter:  
*path* - name of the folder with the event definitions file.

**Event Definitions File**

Purpose: This option is used to specify the name of the file with event definitions for a given channel. The item value overrides the setting specified by the item of the same name present in the driver parameters. The file is searched for in the folder specified by the option *Definitions files path*.

The default value: By default the item takes the value defined in the driver parameters.

Parameter:  
*file\_name* - name of the event definitions file.

**Event Check Period**

Purpose: The option is used to specify the interval between subsequent checks of the presence of new events in a given channel. The item value overrides the setting specified by the item of the same name present in the driver parameters.

The default value: By default the item takes the value defined in the driver parameters.

Parameter:  
*number* - the time between readings in seconds.

**Time Synchronization**

Purpose: The option is used to specify the interval between subsequent records of date stamp in the controller. The value set to 0 means that the time will not be synchronized.

The default value: By default the item takes the value defined in the driver parameters.

Parameter:  
*number* - time synchronization interval in minutes.

**EXAMPLE**

Sample channel section:

*Log File:* d:\logi\kanal1.log

*Log File Size:* 20

*Log of Telegrams:* YES

*IP Address of the Computer:* 10.10.115.25

## 1.33 Ecl - Driver for Communication with ECL Comfort 210/310 Controllers Developed by Danfoss

### Driver Use

The Ecl driver is used to exchange data between the Asix system and ECL Comfort 210/310 controllers by Danfoss. Data exchange is carried out via Ethernet in the Open Modbus Tcp/Ip mode.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the Ecl driver requires adding a channel with the following parameters to the *Current data* module:

**Standard** tab:

*Name: logical name of the transmission channel*  
Driver: Ecl

**Ecl / Channel Parameters** tab:

*[port=number;] serverIP=IPaddress; nr=number [;AlarmNr=number]*

where:

<i>port</i>	- TCP port number in the controller, by default it is 502.
<i>serverIP</i>	- IP address of the controller,
<i>nr</i>	- number assigned to the controller in the Modbus network,
<i>AlarmNr</i>	- alarm number in the absence of communication with the controller.

Example of transmission channel declaration:

**ServerIP=10.10.105.21; Nr=10; AlarmNr=100**

### Process Variable Declaration

The driver provides the following symbolic addresses:

<b>HR</b>	- single holding register
<b>HRL</b>	- next 2 holding registers treated as a double word in INTEL syntax
<b>HRF</b>	- next 2 holding registers treated as a floating point number in INTEL syntax
<b>HRLM</b>	- next 2 holding registers treated as a double word in MOTOROLA syntax
<b>HRFM</b>	- next 2 holding registers treated as a floating point number in MOTOROLA syntax

## Defining Holding Register Address Range

The ECL controller provides a fragmented register address range (the so called PNU) which additionally is specific for the application supported by the controller. At the same time, implementation of the Modbus protocol in the controller requires so that the requests sent to the controller refer only to such register addresses that are defined in the controller. For that reason, while structuring requests, the driver must have a database of a valid register range for the controller. This database is provided by the application developer in the form of a text file, whose consecutive lines include the definitions of holding address ranges for HR or IR registers. There is no need for the map to include all holding register address ranges - it is enough when it includes the definitions of those ranges to which the registers used by the Asix system process variables belong.

A single definition of the range has the following form:

type, start, end

where:

<i>type</i>	- guideword defining the type of registers contained within a given
range (HR or IR),	
<i>start</i>	- address of the first register in a given range,
<i>end</i>	- address of the last register in a given range.

### Example:

definitions of the register address range when the controller provides registers of the following numbers HR10 ÷ HR14, HR21, HR30 ÷ HR50:

HR, 10, 14

HR, 21, 21

HR, 30, 50

## Driver Parameters

The ECL driver parameters are declared in the *Miscellaneous* module in the *Directly entered options* tab:

The driver is configured using a separate section [CTECL]: In this section it is possible to include declaration items:

- creating a log file,
- the log file size,
- log of telegrams,
- file directory including the definitions of holding register address ranges,
- reception timeout,
- maximum number of registers in a single request,
- IP address of the computer card, with the use of which the controller is operated.

The names of the items related with the log file refer to the convention used in other ASMEN drivers.

**Section name:** CTECL

**Option name:** LOG\_FILE

**Option value:** *log\_file\_name*

Purpose: the text log file to which driver status messages are stored is used for diagnostic purposes.

Option value:  
*log\_file\_name*

The default value: by default, the log file is not created.

**Section name:** CTECL

**Option name:** LOG\_FILE\_SIZE

**Option value:** *number*

Purpose: this item is used to determine the size of the log file defined using the LOG\_FILE item.

Option value:  
*number* - the size of the log file in MB.

The default value: by default the log file size is 10 MB.

**Section name:** CTECL

**Option name:** LOG\_OF\_TELEGRAMS

**Option value:** YES/NO

Purpose: The item allows writing the contents of messages communicated between the driver and the controllers to the log file (declared using the LOG\_FILE item) The item in question should only be used during the ASIX system start-up.

The default value: by default the value of the item is NO.

#### File directory including the definitions of holding register address ranges

**Section name:** CTECL

**Option name:** RANGE\_DEFINITIONS\_PATH

**Option value:** *path*

Purpose: This item is used to specify the name of the file in which the driver will search for files including definitions of holding register address ranges of the controllers.

Option value:  
*path* - complete path to the file directory including the definitions of ranges

**Section name:** CTECL

**Option name:** RECV\_TIMEOUT

**Option value:** *number*

Purpose: This item specifies the maximum time throughout which the driver waits for a response from the controller.

Option value:  
*number* - time out in milliseconds.

The default value: the default value is 500 milliseconds.

**Section name:** CTECL

**Option name:** MAX\_REGISTERS

**Option value:** *number*

Purpose: The item is used to specify the maximum number of registers, whose values can be subject to a single request from the driver.

Option value:

*number* - maximum number of registers in a single request  
 The default value: the default value is 123.

- Section name: CTECL**
- Option name: IP\_ADDRESS**
- Option value: IP\_address**

Purpose: This item is used to specify the IP address of the computer's network card, with the use of which all driver channels communicate with the controllers.

Option value: *IP\_address* - network card address in IPv4 notation  
 The default value: if this item is not defined, the system network card will be used.

## EXAMPLE

Driver parameterization:

*Section name:* CTECL  
*Option name:* LOG\_FILE  
*Option value:* d:\tmp\CtEcl\vecl.log

*Section name:* CTECL  
*Option name:* LOG\_FILE\_SIZE  
*Option value:* 20

*Section name:* CTECL  
*Option name:* LOG\_OF\_TELEGRAMS  
*Option value:* YES

*Section name:* CTECL  
*Option name:* IP\_ADDRESS  
*Option value:* 10.10.110.24

*Section name:* CTECL  
*Option name:* RANGE\_DEFINITIONS\_PATH  
*Option value:* d:\RangeDef

## Channel Parameters

The driver is configured using a separate section called Asmen channel. The parameters of the channel are declared in the *Miscellaneous* module, in the *Directly Entered Options* tab. In the section named Asmen channel it is possible to include declaration items:

- creating a log file,
- the log file size,
- log of telegrams,
- reception timeout,
- maximum number of registers in a single request,
- IP address of the computer card, with the use of which the controller is operated,
- name of the file including the definitions of holding register address ranges of the controller.

**Section name:** <channel\_name>

**Option name:** LOG\_FILE

**Option value:** log\_file\_name

Purpose: The text log file to which the contents of telegrams sent and received in a given channel are stored is used for diagnostic purposes.

Option value:  
log\_file\_name

**Section name:** <channel\_name>

**Option name:** LOG\_FILE\_SIZE

**Option value:** number

Purpose: this item is used to determine the size of the log file defined using the LOG\_FILE item.

Option value:  
number - the size of the log file in MB.

The default value: by default the log file size is 10 MB.

**Section name:** <channel\_name>

**Option name:** LOG\_OF\_TELEGRAMS

**Option value:** YES/NO

Purpose: This option allows writing the contents of messages communicated in this channel to the log file (declared using the LOG\_FILE item) The item value overrides the setting specified by the item of the same name present in the driver section.

**Section name:** <channel\_name>

**Option name:** RECV\_TIMEOUT

**Option value:** number

Purpose: This option specifies the maximum time throughout which a given channel waits for a response from the controller. The item value overrides the setting specified by the item of the same name present in the driver section.

Option value:  
number - time out in milliseconds.

**Section name:** <channel\_name>

**Option name:** MAX\_REGISTERS

**Option value:** number

Purpose: The option is used to specify the maximum number of registers, whose values can be subject to a single request from the driver. The item value overrides the setting specified by the item of the same name present in the driver section.

Option value:  
number - maximum number of registers in a single request

**Section name:** <channel\_name>

**Option name:** IP\_ADDRESS

**Option value:** IP\_address

Purpose: this option is used to specify the IP address of the computer's network card, with the use of which the channel communicates with the controller. The item value overrides the setting specified by the item of the same name present in the driver section.

Option value:  
IP\_address - network card address in IPv4 notation

**Section name:** <channel\_name>

**Option name:** RANGE\_DEFINITIONS\_FILE

**Option value:** file\_name

Purpose: this option is used to transfer the name of the file from which the definitions of holding register address ranges of the controller supported by a given channel will be read. The file will be searched

for in the directory specified by the item  
**RANGE\_DEFINITIONS\_FILE.**

Option value:  
*file\_name*

If the item **RANGE\_DEFINITIONS\_FILE** does not exist, it is assumed that the controller supported in a given channel supports the complete address range of the MODBUS protocol.

## EXAMPLE

A sample section for a transmission channel named CHANNEL1 is provided below.

*Section name: <channel\_name>*  
*Option name: LOG\_FILE*  
*Option value: d:\logs\channel1.log*

*Section name: <channel\_name>*  
*Option name: LOG\_FILE\_SIZE*  
*Option value: 20*

*Section name: <channel\_name >*  
*Option name: LOG\_TELEGRAMOW*  
*Option value: TAK*

*Section name: <channel\_name >*  
*Option name: ADRES\_IP*  
*Option value: 10.10.115.25*

*Section name: <channel\_name >*  
*Option name: TIMEOUT\_ODBIORU*  
*Option value: 750*

*Section name: <channel\_name >*  
*Option name: RANGE\_DEFINITIONS\_FILE*  
*Option value: channel1.def*

\*\*\*

Some channel parameters are declared directly in the *Current data > <ECL driver channel name> >* module in the *Advanced* tab:

### **Remote Write Access**

Purpose

- this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

Some channel parameters are declared directly in the *Current data > <ECL driver channel name>* > module in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - By default, no time for data validity is set.

## 1.34 EcoMUZ - Driver of ecoMUZ Protocol

### Driver Use

The EcoMUZ driver is used for data exchange between the Asix system and Microprocessor Protecting ecoMUZ Devices made by JM Tronik. The transmission is performed through serial links by means of serial interfaces in the RS-232 or RS-485 standard, depending on ecoMUZ's serial interface type.

Parameterization of CtEcoMUZ driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the EcoMUZ driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* EcoMUZ

**CtEcoMUZ** tab:

*Channel parameters:*

**Port=number; Nr=number [;Timeout=number]**

where:

*Port* - number of the ecoMUZ's serial port,  
*Nr* - number of an ecoMUZ serviced by the channel,  
*Timeout* - max. timeout for the first character of the controller response (in milliseconds); it is passed 1000 milliseconds by default.

The following constant transmission parameters are passed:

- transmission speed - 9600 Bd,
- number of bits in a character - 8,
- parity check,
- number of stop bits - 1.

### EXAMPLE

Exemplary declarations of channels for communication with the 1064 and 1125 ecoMUZs, by means of the COM2 serial port, and the 1068 ecoMUZ, by means of the COM1 serial port:

*Channel / Name:* K1064  
*Driver:* CtEcoMUZ  
*Channel parameters:* Port=2;Nr=1064

*Channel / Name:* K1066  
*Driver:* CtEcoMUZ  
*Channel parameters:* Port=2;Nr=1066

Channel / Name: K1068  
Driver: CtEcoMUZ  
Channel parameters: Port=1;Nr=1068

## Addressing the Process Variables

The syntax of symbolic address which is used for variables belonging to the EcoMUZ driver channel is as follows:

<type>[.<index>]

where:

- |              |  |
|--------------|--|
| <i>type</i>  | - variable type,                                   |
| <i>index</i> | - index within the type – for the following types: |
|              | • measurements,                                    |
|              | • settings of protections,                         |
|              | • device state.                                    |

Symbolic names of variable types (in parentheses the type of a raw variable value is given):

### Read-only variable types:

- |          |   |   |
|----------|---|---|
| <b>P</b> | - | measurements (FLOAT), index range 1- 4,             |
| <b>Z</b> | - | settings of protections (FLOAT), index range 1 – 6, |
| <b>S</b> | - | device state (BYTE), index range 1 – 3.             |

### Write-only variable types:

- |           |   |  |
|-----------|---|--|
| <b>PP</b> | - | activation of transmitters (WORD),                       |
| <b>KS</b> | - | signalling deletion (WORD),                              |
| <b>ZI</b> | - | protection test performance I>> i I>,                    |
| <b>ZZ</b> | - | performing the test of floating-short-circuit protection |

### Read-only virtual variable:

- |           |   |  |
|-----------|---|--|
| <b>SK</b> | - | communication status (WORD) (1 – o.k., 0 – lack or communication errors) |
|-----------|---|--|

## EXAMPLE

Exemplary variable declarations – the K1064 channel services the 1064 ecoMUZ, the K1066 channel services the 1066 ecoMUZ:

JJ_10, value of current I1	MUZ1064,	P1,	K1064,	1,	1,	NOTHING_FP
JJ_20, value of current I3	MUZ1066,	P3,	K1066,	1,	1,	NOTHING_FP
JJ_30, setting of short-circuit unit current I>>	MUZ1064,	Z1,	K1064,	1,	1,	NOTHING_FP
JJ_40, setting of short-circuit unit time I>>	MUZ1066,	Z4,	K1066,	1,	1,	NOTHING_FP
JJ_50, state of device MUZ1066,		S1,	K1066,	1,	1,	NOTHING_BYTE
JJ_60, configuration of device MUZ1064,		S2,	K1064	1,	1,	NOTHING_BYTE
JJ_70, activation of transmitter MUZ1064,		PP,	K1064,	1,	1,	NOTHING
JJ_80, deletion of signalling of MUZ1066,		KS,	K1066,	1,	1,	NOTHING
JJ_90, status of communication with MUZ1064,		SK,	K1064,	1,	1,	NOTHING
JJ_91, status of communication with MUZ1066,		SK,	K1066,	1,	1,	NOTHING

## Driver Configuration

EcoMUZ driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **CtEcoMUZ** section.

**Section name:** CtEcoMUZ

**Option name:** LOG\_FILE

**Option value:** *log\_file\_name*

Meaning - for diagnostic purposes the text-type log file, which messages about driver operation status are written into, is used.

Default value - by default, the log file is not created.

Defining - manual.

**Section name:** CtEcoMUZ

**Option name:** LOG\_FILE\_SIZE

**Option value:** *number*

Meaning - this item is used to define the size of the log file defined with use of the LOG\_FILE item.

Default value - by default, the log file size is 10 MB.

Defining - manual.

**Section name:** CtEcoMUZ

**Option name:** LOG\_OF\_TELEGRAMS

**Option value:** *YES / NO*

Meaning - this item allows contents of telegrams transferred between the driver and controllers to be written into the log file (declared with use of the LOG\_FILE item). The referred item should only be used in the Asix system start-up stage.

Default value - by default, value of this item is set to NO.

Defining - manual.

## Exemplary Driver Section

*Section name:* CtEcoMUZ

*Option name:* LOG\_FILE

*Option value:* d:\tmp\CtEcoMUZ\muz.log

*Section name:* CtEcoMUZ

*Option name:* LOG\_FILE\_SIZE

*Option value:* 20

*Section name:* CtEcoMUZ

*Option name:* LOG\_OF\_TELEGRAMS

*Option value:* YES

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.35 EcoMuz2 - Driver of ecoMUZ-2 Protocol of JM Tronik

### Driver Use

The ecoMuz2 driver is used for data exchange between the Asix system and Microprocessor Protecting ecoMuz-2 devices made by JM Tronik. The transmission is performed through serial links by means of serial interfaces in the RS-485 standard.

The driver realizes the following functions:

- current states and measurements readout,
- setpoints readout,
- settings readout,
- identifying data readout,
- events readout and reporting them to the Asix alarm system,
- interference readout and registering them in AsLogger (made by ASKOM Sp. z o.o.) database

The driver supports the following ecoMuz-2 types:

2G

BS

LR

PR

SR

TR

Support a device of other type demands modifications in the driver code.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the ecoMuz2 driver requires a channel with the following parameters to be added to the **Current data** module:

**Standard** tab:

**Name:** *logical name of the transmission channel*

**Driver:** EcoMuz2

**EcoMuz2** tab:

**Channel parameters:**

*Device number* - network device number,

*Device type* - ecoMuz-2 type. Identifiers of supported types:

1 - 2G

2 - BS

3 - LR

4 - PR

5 - SR

6 - TR

After selecting the device type push the button *Show device parameters*. The detailed informations on device parameters and event parameters of selected type will be displayed in a web browser (at least IE6).

Run the mentioned window just once and select consecutive device types - the window content will be refreshed automatically.

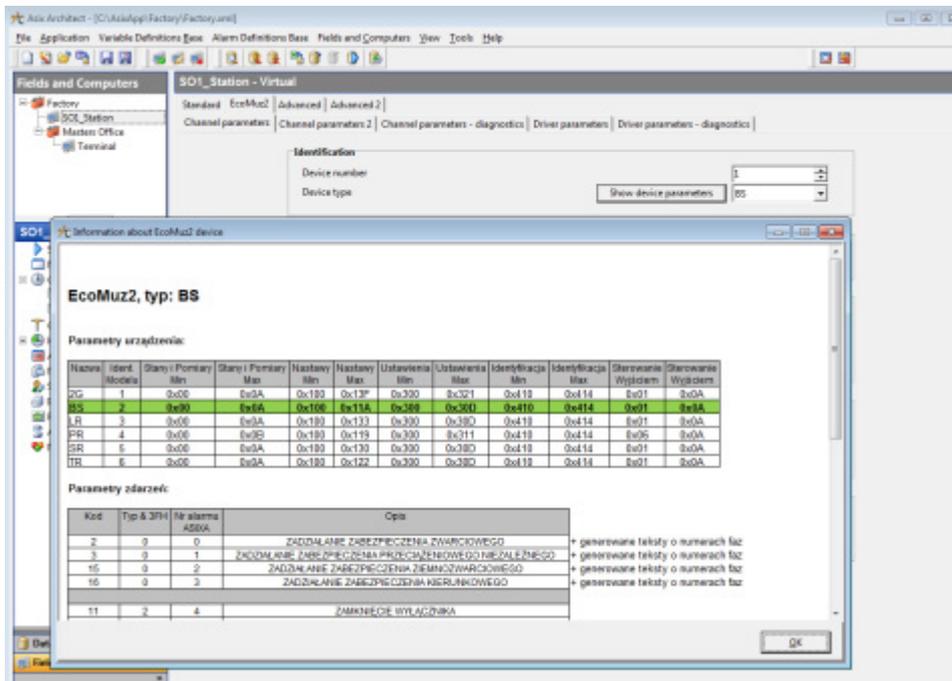


Fig. Information about EcoMuz2 device according to the selected type.

- Alarms offset** - offset added to the calculated Asix alarm number,
- Port** - serial port name,
- Baud rate** - 9600 or 19200 (for 2G type - 9600 only),
- Parity bit checking** - even or none (for 2G type - none only),
- Control variables:**

The name of the variable used to present the transmission status with ecoMuz-2.

- 0 - transmission status ok,
- 1 - no communication/errors of communication,
- 2 - ecoMuz-2 off from the service,

The name of the variable to control the transmission state with ecoMuz-2:

- 1 - request to exclude ecoMuz-2 from the service,
- 0 - include ecoMuz-2 to the service.

**Time synchronization** - period of ecoMuz-2 time synchronization (in seconds). The value 0 means no time synchronization (default value).

**Notice:**

- a/ control variables have to belong to the NOEN type channel.
- b/ constant transmission parameters: number of bits - 8, number of stop bits - 1.

**EXAMPLE**

Below there is an exemplary declaration of the channel: KSR\_67, in which ecoMuz-2 with network number 5, of LR type is serviced, on COM1 port and with alarm offset: 200. The channel state is indicated by the variable: MuzStat05, the channel control is enabled by the variable: MuzCtrl05, time synchronization is done every 60 seconds.

*Name:* KSR\_67

*Driver:* ecoMuz2

*Device number:* 5

*Device type:* 3

*Alarms offset:* 200

*Port:* COM1

*Baud rate:* 19200

*Parity bit checking:* even

*Name of the variable used for showing transmission status:* MuzStat05

*Name of the variable used for controlling transmission state:* MuzCtrl05

*Time synchronization:* 60

## Addressing the Process Variables

The driver makes the following variable types of MODBUS protocol:

CS	- binary value	(write),
HR	- 16-bit register	(readout)

and internal driver variables:

MN	- ecoMuz-2 network number, from which the register is/was read	(readout)
RN	- the number of currently/recently read recorder	(readout)
RQ	- recorder readout request	(write)
RR	- the status of currently/recently read recorder	(readout)
RS	- recorder release reason	(readout)
RT	- recorder release time	(readout)
SN	- the number of recently read recorder sample	(readout)

The syntax of symbolic address which is used for variables belonging to the ecoMuz2 driver channel is as follows:

Type[index]

where:

<i>Type</i>	- variable type,
<i>Index</i>	- index of variable within declared types.

REMARKS:

indexes are not declared for the types: *MN*, *RN*, *RQ*, *RR* and *SN*

- for the types: *RS*, *RT* the range of indexes depends on the number of sectors defined in the device,
- the parameter of sector readout request (*RQ* type) is the sector number; the sectors are numbered w.e.f. 1. If 0 is set as the parameter, that means the interrupt request of sector current readout;
- ranges for *HR* types are depended on the device type,
- to present the sector release time, use the conversion fnction *DATETIME\_MUZ*, that convert the number of *double* type into the following string:

yyyy-mm-dd hh:nn:ss.zzz

**EXAMPLE**

Examples of variable declarations (variable values are taken from the *LR* type device through the channel *KLR\_01*):

ZM_01, I1 value,	HR1, KLR_01, 1, 1, NOTHING
ZM_02, activating protections of 1,	HR6, KLR_01, 1, 1, NOTHING
ZM_03, number of not read events,	HR11, KLR_01, 1, 1, NOTHING
ZM_06, time of release time of recorder 1,	RT1, KLR_01, 1,1,

DATETIME\_MUZ

Example of variable declaration without parameters (variable values taken from the device of *LR* type available through the channel *KLR\_01*):

ZM_07, recorder readout request,	RQ, KLR_01,1,1, NOTHING
ZM_08, number of read recorder,	RN, KLR_01,1,1, NOTHING
ZM_09, status of recorder readout,	RR, KLR_01,1,1, NOTHING
ZM_10, number of the sample read from recorder,	SN, KLR_01,1,1, NOTHING
ZM_11, number of device being read,	MN, KLR_01,1,1, NOTHING

## Interference Recording

The driver allows for automatic registration of interferences in AsLogger database. The registered items can be reviewed only by AsLogger software.

The recording of interference values demands declaration of **Registration of disturbances in AsLogger database** option (**EcoMuz2** tab / **Driver parameters**). It contains:

- MSSQL server name,
- database name,

where the interference values will be written.

The driver does not register the trends automatically - registration has to be initiated any time by the Asix system operator by execution of control with the use of *RQ* type variable. The control value is the number of recorder the value of which should be read from ecoMuz-2 and written to AsLogger database. If control value equals 0, it means the interrupt request of sector current readout.

While the sector readout the readout state tracking is possible with the use of driver internal variables of symbolic addresses:

- a/ *MN* - number of read ecoMuz-2,
- b/ *RN* - number of read recorder,
- c/ *RR* - status of recorder readout:

**0 - readout is not active or finished OK,**  
**1 - readout is in progress,**  
**2 - readout finished with error.**

- d/ *SN* - number of recently read sample.

The recording of interferences is realized according to recording plans. The names of recording plans may be:

- a/ defined by the user,
- b/ generated automatically by the driver.

While the recording the driver writes to AsLogger database the following interference items:

- a/ full set of plan items, i.e.:

- currents: I0, I1, I2, I3
- tensions: U0, U1, U2, U3
- angle: fi0
- inputs states: WE01 ... WE08
- output states: WY01 ... WY08
- states of interference boost states 1: PZ01 ... PZ16
- states of protection working 1: ZZ01 ... ZZ16

The set of parameters and number of plan items concerned the states of protection boost and working are specific for each ecoMuz-2 type,

- b/ creates the plan archive,

c/ wyznacza max i min wartości dla prądów i napięć w rejestrowanym zakłóceniu - będą ew. korygowane przy odczycie kolejnych zakłóceń dla danego planu.

c/ determines max and min values for current and voltage in recorded interference - if need they will be corrected during reading the successive interferences for the given plan.

### Defining the Names of Recording Plans by the User

The user can define the names of recording plans in text files with the extension '.def'. To do this for each communication channel (channel of Asmen) one should create the separate file and write to them the definitions of recording plan names for interferences recorded in this channel. The files with the plan definitions should be located in the common directory the name of which should be declared as the **Directory with files containing registration plan names** option (**EcoMuz2** tab / **Driver parameters**).

The syntax of definition for recording plan name defined in '.def' file is as follows:

**Channel, Cause, Table [, Commentary]**

where:

Channel - Asmen channel name,

## Communication Drivers

<i>Cause</i>	- number identifying the cause of recorder release (number from 0-15),
Table	- recording plan name assigned to the interference,
Commentary	- kommentary

### EXAMPLE

The definitions of recording plan names for interferences:

- release from the current I (release reason no. 0)
- release from the current IO (release reason no. 1)
- release from the tension U0 (release reason no. 2)
- release from the logic (release reason no. 3) in Asmen channel KTR\_06:

KTR\_06, 0, Field06ReleaseI, release from the current I  
KTR\_06, 1, Field06ReleaseIO, release from the current IO  
KTR\_06, 2, Field06ReleaseU0, release from the tension U0  
KTR\_06, 3, Field06Logic, release from the logic

### The Default Recording Plan Names

If the user does not define its own recording plan names, the driver will create the name of recording plan automatically.

The syntax of interference recording plan name generated by the driver is as follows:

#### ***NrxxxpyyyReason***

where:

*xxx* - ecoMuz-2 network number,  
*yyy* - COM port number,  
*Reason* - identifier of recorder release cause.

#### **Example:**

For interference caused by the recorder release from the user's logic in ecoMuz-2 no. 2 and COM5 the recording plan will be:

**Nr002P005logika**

## Event Signalling

Signalling events is checked periodically, the frequency of check is determined by the global option

**Event check period** (see: *Driver parameters*). Event read from the devices are converted to the corresponding numbers in the Asix system alarms.

The conversion is performed based on the mapping prepared for each type of ecoMuz-2 and offset defined in Asmen channel declaration. There are two types of Asix alarms:

- a/ text only,
- b/ text and one numerical parameter.

The events of 0 are converted to Asix alarms of text type + one numerical parameter (of int type) + text parameter (max 3 characters), involving the number of phases.

The ecoMuz-2 events of the type no. 2 are converted to Asix alarms of text type.

#### Example 1:

The event *Turning off the switching from telemechanics* in ecoMuz-2 of LR type (the event of text type) is mapped to the Asix alarm number with the value of 8. If the alarm offset with the value of 400 has been declared in the Asmen channel declaration servicing this protection, then the definition of the alarm in Asix alarm definition file will have the form ( $400 + 8 = 407$ ):

**407, al, turning off the switching from telemechanics**

#### Example 2:

The event *Activation of short-circuit protection* in ecoMuz-2 of LR type (the event of text type and two parameters: numerical and text) is mapped to Asix alarm number with the value of 0. If the alarm offset with the value of 100 has been declared in the Asmen channel declaration servicing this protection, then the definition of the alarm in Asix alarm definition file will have the form ( $100 + 0 = 100$ ):

**100, al, Activation of short-circuit protection %f, fazy %s**

Mapping the events of ecoMuz-2 types on the numbers of Asix alarms are displayed in browser window after selection of the given device type and pushing the button **Show device parameters**.

## Driver Parameters

The driver configuration is defined in the Current Data module, in the channel operating according to ecoMuz2 driver.

#### **Log file**

Meaning: The item allows to define a file where all diagnostic messages of the driver are written.

Default value: By default, the log file is not created.

Option value:

*log\_file\_name*

#### **Disturbances log file**

Meaning: The item allows to define a file where all diagnostic messages on read disturbances are written.

Default value: The log file is not created.

Option value:

*log\_file\_name*

**Log file size**

Meaning: The item allows to specify the size of log file size defined by the option: *Log file*.

Default value: 10 MB.

Option value:  
*number* - rozmiar pliku logu w MB.

**Log of telegrams**

Meaning: The item allows to write to the log file the contents of telegrams sent between the driver and network clients. Writing the contents of telegrams to the log file should be used only while the **Asix** start-up.

Default value: NO.

Option value:  
YES/NO

**Number of repetitions**

Meaning: The option defines the maximal number of connection repetitions in case of transmission error.

Default value: 3.

Option value:  
*number* - the number of connection repetitions.

**Response timeout**

Meaning: The option defines the maximal time of waiting for response from ecoMuz-2 by the driver. The option is global but its value may be overloaded by the channel individual settings.

Default value: 1000.

Option value:  
*number* - timeout in milliseconds.

**Character timeout**

Meaning: The option defines the maximal time between characters of response from EcoMuz-2. The option is global but its value may be overloaded by the channel individual settings.

Default value: 100.

Option value:  
*number* - timeout in milliseconds.

**Event check period**

Meaning: The option defines the number of seconds between event register readout of the consecutive device connected to the same serial port servicing by the driver.

Default value: 10.

Option value:  
*number* - time in seconds.

 **Registration of disturbances in AsLogger database**

Meaning: The option changes over the driver to the mode of disturbances recording. It defines the database in which disturbance samples will be recorded.

Default value: Disturbances are not recorded.

Option value:  
*ServerName* - MSSQL server name  
*DatabaseName* - database name on the *ServerName*

 **Directory with files containing registration plan names**

Meaning: The option defines full path to the directory of files with recording plan definitions.

Default value: -.

Option value:  
*path* - full path to the directory.

## Channel Parameters

The channel parameters are defined in the Current Data module, in the channel operating according to ecoMuz2 driver.

 **Log file**

Meaning: The item allows to define a file where all diagnostic messages of the driver in the given communication channel are written.

Default value: The log file is defined by the driver option.

Option value:  
*log\_file\_name*

 **Log file size**

Meaning: The item allows to specify the size of log file size defined by the option *Log file* of the given channel.

Default value: 10.

Option value:  
*number* - log file size in MB.

**Log of telegrams**

Meaning: The item allows to write to the log file the contents of telegrams sent between the driver and network clients in the given channel. Writing the contents of telegrams to the log file should be used only while the Asix start-up.

Default value: Settings defined by the driver option.

Option value:

YES/NO

**Response timeout**

Meaning: The option defines the maximal time of waiting for response from the device.

Default value: 1000.

Option value:  
*number* - timeout in milliseconds.

**Character timeout**

Meaning: The option defines the maximal time between characters of response from EcoMuz-2. The option overloads the driver option.

Default value: 100.

Option value:  
*number* - timeout in milliseconds.

**Without events**

Meaning: The option is used to switch off the event readout function in the given channel.

Default value: the readout is switch on.

Option value:

YES/NO

**Without registrar**

Meaning: The option is used to switch off the function of recorder interference readout in the channel.

Default value: recorder interference readout is enabled.

Option value:

YES/NO

EXAMPLE

Exemplary driver parameters.

*Driver:* EcoMuz2  
*Log file:* c:\tmp\muz.log  
*Disturbances log file:* c:\tmp\zaklocenia.log  
*Log file size:* 30

Log of telegrams: YES  
 Registration of disturbances in AsLogger database: SERWER\_RO6, BAZA\_RO6  
 Directory with files containing registration plan names: c:\tmp\DefZaklocen

#### EXAMPLE

Exemplary channel parameters:

Name: KANALX  
 Log file: c:\tmp\kanalx.log  
 Log file size: 20  
 Log of telegrams: YES  
 Character timeout: 200  
 Response timeout: 1500

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value

- by default, no time for data validity is set.

## 1.36 EQABP - Driver for Electricity Meters EQABP of POZYTON

### Driver Use

The driver is used to exchange data between the Asix system and electricity meters EQABP, produced by Zakład Elektronicznych Urządzeń Pomiarowych POZYTON sp. z o.o. in Częstochowa.

### Declaration of Transmission Channel

Declaration of the transmission channel using the EQABP driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: EQABP

**EQABP** tab:

*Channel parameters*:  
*Port=port\_number; Baud=transmission\_speed; Period=period\_number*

where:

<i>port_number</i>	- number of serial port to which the meter is connected (1...16); specifying the parameter is mandatory;
<i>transmission_speed</i>	- transmission speed between your computer and device in bauds (300, 1200, 2400, 4800, 9600, 19200 or 38400); default value: 9600;
<i>period_number</i>	- interval between successive readings of current meter indications in seconds; default: 10.

### Declaration of Variables

Identifiers of variables consist of two parts: the address of device (if several devices were connected with one port) and the name of variable, joined by the character '/'. The address of device may be omitted only if there is one device connected with one port, but the character '/' can not be omitted.

Example of variable declarations (logical name of the channel - EQABP).

```
#energy meters P- in zones 1, 2, 3 i 4:
/1.8.1, /1.8.1, /1.8.1, EQABP, 1, 1, NOTHING_FP
/1.8.2, /1.8.2, /1.8.2, EQABP, 1, 1, NOTHING_FP
/1.8.3, /1.8.3, /1.8.3, EQABP, 1, 1, NOTHING_FP
/1.8.4, /1.8.4, /1.8.4, EQABP, 1, 1, NOTHING_FP

#energy meters P+ in zones 1, 2, 3, 4 (X=1, 2, 3, 4)
```

## Communication Drivers

```
/0.8.X,      /0.8.X,      /0.8.X,      EQABP, 1, 1, NOTHING_FP

#energy meters Q+ in zones 1, 2, 3, 4 (X=1, 2, 3, 4)
/2.8.X,      /2.8.X,      /2.8.X,      EQABP, 1, 1, NOTHING_FP

#energy meters Q- in zones 1, 2, 3, 4 (X=1, 2, 3, 4)
/3.8.X,      /3.8.X,      /3.8.X,      EQABP, 1, 1, NOTHING_FP

#sum of energy meters P-/P+/Q+/Q- in all zones (X=1,0,2,3)
/X.8.0,      /X.8.0,      /X.8.0,      EQABP, 1, 1, NOTHING_FP

#power in previous cycle (P+, P-, Q+, Q-)
/0.4.1_0,    /0.4.1_0,    /0.4.1_0,    EQABP, 1, 1, NOTHING_FP
/0.4.1_1,    /0.4.1_1,    /0.4.1_1,    EQABP, 1, 1, NOTHING_FP
/0.4.1_2,    /0.4.1_2,    /0.4.1_2,    EQABP, 1, 1, NOTHING_FP
/0.4.1_3,    /0.4.1_3,    /0.4.1_3,    EQABP, 1, 1, NOTHING_FP

#phase voltages
/97.5.5_0,   /97.5.5_0,   /97.5.5_0,   EQABP, 1, 1, NOTHING_FP
/97.5.5_1,   /97.5.5_1,   /97.5.5_1,   EQABP, 1, 1, NOTHING_FP
/97.5.5_2,   /97.5.5_2,   /97.5.5_2,   EQABP, 1, 1, NOTHING_FP

#currents for each phase
/97.4.4_0,   /97.4.4_0,   /97.4.4_0,   EQABP, 1, 1, NOTHING_FP
/97.4.4_1,   /97.4.4_1,   /97.4.4_1,   EQABP, 1, 1, NOTHING_FP
/97.4.4_2,   /97.4.4_2,   /97.4.4_2,   EQABP, 1, 1, NOTHING_FP

#temporary power active for each phase
/107_1,      /107_1,      /107_1,      EQABP, 1, 1, NOTHING_FP
/107_2,      /107_2,      /107_2,      EQABP, 1, 1, NOTHING_FP
/107_3,      /107_3,      /107_3,      EQABP, 1, 1, NOTHING_FP

#power meters S+ and S-
/4.8.0, /4.8.0, /4.8.0, EQABP, 1, 1, NOTHING_FP
/5.8.0, /5.8.0, /5.8.0, EQABP, 1, 1, NOTHING_FP
```

## Driver Configuration

EQABP driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **CtEQABP** section.

**Section name:** CTEQABP

**Option name:** LOG\_FILE

**Option value:** log\_file\_name

Meaning - for diagnostic purposes the text-type log file, which messages about driver operation status are written into, is used.

Default value - by default, the log file is not created.

Defining - manual.

**Section name: CTEQABP**

**Option name: LOG\_FILE\_SIZE**

**Option value: number**

Meaning - this item is used to define the size of the log file defined with use of the LOG\_FILE item.

Default value - by default, the log file size is 10 MB.

Defining - manual.

**Section name: CTEQABP**

**Option name: LOG\_OF\_TELEGRAMS**

**Option value: YES / NO**

Meaning - this item allows contents of telegrams transferred between the driver and controllers to be written into the log file (declared with use of the LOG\_FILE item). The referred item should only be used in the Asix system start-up stage.

Default value - by default, value of this item is set to NO.

Defining - manual.

## EXAMPLE

Driver parameters:

*Section name:* CTEQABP

*Option name:* LOG\_FILE

*Option value:* d:\tmp\test\licznik.log

*Section name:* CTEQABP

*Option name:* LOG\_FILE\_SIZE

*Option value:* 3

*Section name:* CTEQABP

*Option name:* LOG\_OF\_TELEGRAMS

*Option value:* TAK

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

## Communication Drivers

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.37 Esser - Honeywell Esser 8008 Fire Protection Unit Driver

### Driver Use

The Esser driver allows data exchange between the Asix system and the Honeywell Esser 8008 fire protection unit control system. Communication takes place by means of the fire protection unit RS-232 serial connection operating in the 'RA LCD/POSP not supervised' mode.

Esser driver configuration is performed using the Architect application.

### Transmission Channel Declaration

Declaration of the transmission channel using the Esser driver requires adding a channel with the following parameters to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: Esser

**Esser** tab:

Channel parameters:

*Port*=nrPortu; *Timeout*=timeout; *Alarm*=AlarmuNo

**where:**

*Port* - COM port number.

*Timeout* - Time (in seconds) between two telegrams (from the fire protection unit control system) after which data provided by the driver will be invalidated. The default parameter value is 8 seconds,

*Alarm* - The number of an alarm reported to the Asix alarm system in the absence of communication with the fire protection unit control system.

The transmission is carried out at fixed settings:

600 Bd, 8 bits per character, no parity, 1 stop bit.

### Declaration of Variables

The driver provides the following types of variables:

C - detector state

G - group state

K - communication line state

Z - power state

Depending on the variable type, variable address has the following form:

C.GroupNo.SensorNo[.AlarmNo]  
G.GroupNo[.AlarmNo]  
K[.AlarmNo]  
Z[.AlarmNo]

where:

AlarmNo - is an optional parameter that is used for declaring the base alarm number for alarms that can be generated for a given variable.

All variables are of WORD type.

The value of the variable is a bitmap of all states, which are indicated for the given variable on the Esser fire protection unit control system (a bit value of 1 means that the state is indicated in the control system). The range of indicated states depends on the type of the variable.

Below is a summary of states indicated depending on the variable type together with the assignment of the states indicated to individual bits of the variable:

For the state of the detector:

Bit 0 - ON ( eingeschaltet )  
Bit 1 - OFF ( abgeschaltet )  
Bit 2 - TROUBLE ( stoerung )  
Bit 3 - FIRE ( feuer )  
Bit 4 - FORALARM ( voralarm )  
Bit 5 - TECHALARM ( tech alarm )

For the state of the communication line:

Bit 0 - TROUBLE ( stoerung )

For the group state:

Bit 0 - ON ( eingeschaltet )  
Bit 1 - OFF ( abgeschaltet )  
Bit 2 - TROUBLE ( stoerung )  
Bit 3 - FIRE ( feuer )  
Bit 4 - FORALARM ( voralarm )  
Bit 5 - TECHALARM ( tech alarm )  
Bit 6 - TEST ( pruefbetrieb )  
Bit 7 - ALARM\_HZ ( alarm HZ-gruppe )  
Bit 8 - ACTIVATED ( erregt )

For the power state:

Bit 0 - TROUBLE ( stoerung )

Examples of variable declarations (the base alarm number is given as the last parameter in the symbolic address of each variable; the variables belong to the logical CHANNEL):

JJ\_11, group state 1, G.1.1, CHANNEL, 1, 1, NOTHING

JJ\_12, group state 2, G.2.10, CHANNEL, 1, 1, NOTHING  
 JJ\_13, group state 3, G.3.19, CHANNEL, 1, 1, NOTHING  
 JJ\_16, detector 1 state in group 1, C.1.1.28, CHANNEL, 1, 1, NOTHING  
 JJ\_17, detector 2 state in group 1, C.1.2.34, CHANNEL, 1, 1, NOTHING  
 JJ\_18, detector 1 state in group 2, C.2.1.40, CHANNEL, 1, 1, NOTHING  
 JJ\_19, connection status, K.46, CHANNEL, 1, 1, NOTHING  
 JJ\_20, power status, Z.47, CHANNEL, 1, 1, NOTHING

## Alarms Declaration

If the optional parameter [AlarmNo] is used in the variable address, then the driver will monitor the status of individual bits of a variable, notifying Asix alarm system of:

- a / the initiation of the alarm when the bit value changes from 0 to 1,
- b / the termination of the alarm when the bit value changes from 1 to 0,

Bit 0 of the variable value is assigned the [AlarmNo] alarm number, bit 1 is assigned the [AlarmNo] + 1 alarm number and so on until all indicated states in the type assigned to the variable are exhausted.

When the application is started, variables (which are assigned alarms) are initiated in accordance with the state of alarms retrieved from the Asix alarm system table and the status of these variables is set to OPC\_QUALITY\_GOOD. Other variables assume the value of 0 and the OPC\_QUALITY\_OUT\_OF\_SERVICE status.

## Driver Parameters

The Esser driver parameters are declared in the *Miscellaneous* module, in the *Directly Entered options* tab.

The driver configuration is done using a separate section called CTESSER.

**Section name: CTESSER** •

**Option name: LOG\_FILE**

**Option value: file\_name** •

Purpose: the text log file to which driver status messages are stored is used for diagnostic purposes.

The default value: By default, no log file is created.

**Section name: CTESSER** •

**Option name: LOG\_FILE\_SIZE**

**Option value: number** •

Purpose: this option is used to determine the size of the log file defined using LOG\_FILE item.

Option value:

*number* - the size of the log file in MB.

The default value: the default log file size is 10 MB.

**Section name: CTESSER** •

**Option name: LOG\_OF\_TELEGRAMS**

**Option value: YES | NO** •

Purpose: this option allows writing the contents of messages communicated between the driver and the controllers to the log file (declared using the LOG\_FILE item) The item in question should only be used during the Asix system start-up.

The default value: default value of the option is NO.

### EXAMPLE

Sample driver parameterization

- Section name:* CTESSER
- Option name:* LOG\_FILE
- Option value:* d:\tmp\test\centrala.log

- Section name:* CTESSER
- Option name:* LOG\_FILE\_SIZE
- Option value:* 30

- Section name:* CTESSER
- Option name:* LOG\_OF\_TELEGRAMS
- Option value:* YES

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

**Purpose** - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

**Option value:**

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

**The default value** - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

**Purpose** - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

**Option value:**

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

**The default value** - by default, all computers can read variables.

**Redundancy**

Purpose - with this option searches a redundant channel when connection with a controller is broken in local channel.

Option value:

*computer name* - list including the names of network computers which will be permitted to search a redundant channel.

The default value - by default, when option is enabled, the channel is searched on all computers.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

 **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

 **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.38 FESTO - Driver of Diagnostic Interface for FESTO PLCs

### Driver Use

The FESTO driver is used for data exchange with FESTO FST-103, FST-405, FST IPC PLCs by means of a diagnostic interface. Required version of firmware: 2.20 or later. The transmission is executed by means of serial interfaces in the V24 (RS232C) standard by using the standard serial ports of an Asix system computer.

The operation of the Asix system with FESTO PLCs by using a diagnostic interface does not require any controller's program adaptation.

Parameterization of FESTO driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the FESTO driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* *FESTO*

**FESTO** tab:

*Channel parameters:*  
*port, [baud, character, parity, stop, cpu\_no]*

where:

<i>port</i>	- name of the serial port;
<i>baud</i>	- transmission speed in baud;
<i>character</i>	- number of bits in a transmitted character;
<i>parity</i>	- parity check type;
<i>stop</i>	- number of stop bits,
<i>cpu_no</i>	- CPU number in the controller.

Parameters *baud*, *character*, *parity*, *stop*, *cpu\_no* are optional. In case of omitting them the following default values are assumed:

- transmission speed - 9600 Bd,
- number of bits in a character - 8,
- parity check type - no parity check,
- number of stop bits - 1,
- CPU number - 0.

### EXAMPLE

Example items declaring the use of two transmission channel working according to the protocol of FESTO controller diagnostic interface are given below. In both channels the

communication is performed by means of the same physical interface but the data are exchanged with other CPUs:

*Channel / Name:* CHAN2  
*Driver:* FESTO  
*Channel parameters:* COM1,9600,8,none,1,2,8

*Channel / Name:* CHAN3  
*Driver:* FESTO  
*Channel parameters:* COM1,9600,8,none,1,3,8

In the example above the declaration of channels differs only with the number of CPU. The CHAN2 channel allows for data exchange with the CPU numbered 2 whereas the CHAN3 channel with the CPU no. 3. The other parameters in the channel declarations are identical:

- port COM1,
- transmission speed of 9600 Bd,
- transmitted character length - 8 bits,
- no parity check,
- one stop bit.

## Addressing the Process Variables

The syntax of symbolic address used for process variables supported by the FESTO driver:

*VARIABLE\_TYPE* *variable\_index*

where:

*VARIABLE\_TYPE*           - string identifying the variable type,  
*variable\_index*           - variable index within the given type.

The following symbols of process variable types are allowable (range of variable indexes is specific for different types of controllers):

EW	- input words,
AW	- output words,
ESW	- words of input statuses,
ASW	- words of output statuses,
MW	- words of buffers,
TW	- current readings of timers,
TV	- set values of timers,
TA	- attributes of timers,
T	- readings of timers,
ZW	- current readings of counters ,
ZV	- set values of counters,
Z	- counter readings,
R	- registers.

### EXAMPLES

R15	- register no. 15
EW0	- input word 0
AW8	- input output 8

All process variables are treated as 16-bit unsigned numbers.

The FESTO driver is loaded as a DDL automatically.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.39 FILE2ASIX - Driver for Data Import from Files

### Driver Use

FILE2ASIX is used for importing data into the Asix system from text files of the following structure:

*Opening line*  
*Line containing variable 1 value*  
*Line containing variable 2 value*  
 ...  
*Line containing variable n value*  
*Closing Line*

Each line has the following form:

$P1 <sep> P2 <sep> P3 <sep> P4 <sep> P5$

where:

$<sep>$  - delimiter ';' ;  
*P1* - for opening and closing line it is the file saving date and time in UTC format;  
 - for other lines it is the variable reading time in UTC format;  
*P2* - symbolic address of the variable (it must not contain exclamation character '!');  
*P3* - quality of the variable in one of forms given below:  
     BAD ,  
     UNCERTAIN ,  
     GOOD;  
*P4* - value of the variable in integer or floating-point form ('.'- as a delimiter)  
*P5* - name of the device path.

Fields: *P2*, *P3*, *P4* and *P5* have to remain empty for opening and closing line.

The UTC time format is as follows:

$YYYY-MM-DD <SP> GG:NN:SS,MS$

where:

*YYYY* - 4 digits indicating a year,  
*MM* - 2 digits indicating a month,  
*DD* - 2 digits indicating a day,  
*SP* - white space character,  
*HH* - 2 digits indicating an hour,  
*NN* - 2 digits indicating minutes,  
*SS* - 2 digits indicating seconds,  
*MS* - 2 digits indicating milliseconds.

**NOTICE** *The file is treated as correct if first and last line are exactly the same.*

Parameterization of FILE2ASIX driver is performed with the use of Architect module.

## Declaration of Transmission Channel

Declaration of the transmission channel utilizing the FILE2ASIX driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: FILE2ASIX

**FILE2ASIX/Channel Parameters** tab:

**Data File** - path to file containing variable values;  
**Read Period** - time (given in seconds) between consecutive file reading (10 seconds are the default value).

### EXAMPLE

```
# reading from the file \\komp\c\data\data.csv with 5 second interval
CHANNEL1 = FILE2ASIX, \\KOMP\C\Data\Data.csv, 5
```

```
# reading from the file n:Data\data.csv with 10 second interval (default value)
CHANNEL2 = FILE2ASIX, n:Data\Data.csv
```

## Addressing the Process Variables

The ASMEN process variable address may take one of the forms given below:

```
"access_path ! address"
"address"
```

where:

*access\_path* - generalized path to the device (value of the *P5* field),  
*address* - address of the variable in the path scope (value of the *P2* field).

The second form of addressing ("r; address") is used when an access path is an empty string (*P5* field has no value).

Only a single variable may be passed through the file, thus a number of elements in the ASMEN variable declaration has to be equal to 1.

The driver automatically converts the type of the variable read from the file to a raw type, which is required by the conversion function given in the declaration of the ASMEN variable.

### EXAMPLE

An example of the file which passes the values of ZMIENNA\_FP and ZMIENNA\_WORD variables:

```
2002-10-04 12:23:37,004;;;
2002-10-04 12:23:26,999;ZMIENNA_FP;GOOD;32.4436;PLC:S7[BEL_SPREZ]
2002-10-04 12:23:26,999;ZMIENNA_WORD;GOOD;32;
2002-10-04 12:23:37,004;;;
```

### EXAMPLE

Examples of the ASMEN variable declaration for the above file:

```
JJ_1, "PLC:S7[BEL_SPREZ] ! ZMIENNA_FP", KANAL, 1, 1, NOTHING_FP
JJ_2, "ZMIENNA_WORD", KANAL, 1, 1, NOTHING
```

## Driver Configuration

FILE2ASIX driver parameters are declared on the **Driver Parameters** tab.

**Option name: Log File**

**Option value: file\_name**

Meaning - it allows definition of a file in which all diagnostic messages of the driver will be saved. If position does not define full path, then the log file will be created in the current directory. The log file should only be used at the stage of the Asix system start-up.

Default value - by default, no log file is created.

**Option name: Log File Size**

**Option value: number**

Meaning - it allows to specify the log file size (using MB unit).

Default value - by default, 1MB size is assumed.

Parameter:

*number* - a size of the log file in MB.

**Option name: Data Validity Period**

**Option value: number**

Meaning - this entry is used for monitoring cases when a data file with a given interval reading fail. The entry defines the number of consecutive failed readings, after which status of the variables will be set to BAD.

Default value - by default, a period of data validity is equal to two cycles of data reading from a joint computer.

Parameter:

*number* - a number of failed reading cycles, after which an error status is set.

## Advanced Channel Parameters

The parameters of a channel are declared in the **Advanced** tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.40 FP1001 - Driver of METRONIC Measurer Protocol

### Driver Use

The FP1001 driver is used for data exchange between FP1001 3.W or FP1001 3.4 flow monitors manufactured by METRONIC Kraków and an Asix system computer provided with a serial interface in the RS485 standard.

Parameterization of FP1001 driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the FP1001 driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: FP1001

**FP1001** tab:

*Channel parameters*:

*Device ID* - device no. in the network;  
*Device type* - device type:  
                   1 - steam flow monitor FP1001 3.4,  
                   2 - water flow monitor FP1001 3.W;  
*Port* - name of the serial port;  
*Transmission speed in bauds* - transmission speed (by default 9600).

The FP1001 driver is loaded as a DLL automatically.

### Addressing the Process Variables

The way of addressing the process variables is given in the following tables.

**Table 15. Addressing the FP1001 3.W Meter (Water Monitor).**

Symb. Address	Variable in FP-1001	Conversion Type	Allowed Operation
	<b>Values of basic measurements</b>		
P1	Instantaneous value Fm	Byte7->Float	Reading

Communication Drivers

P2	Instantaneous value Q	Byte7->Float	Reading
P3	Instantaneous value p	Byte4->Float	Reading
P4	Instantaneous value T	Byte3->Float	Reading
P5	Instantaneous value Tw	Byte3->Float	Reading
P6	Calculation value u water	Byte6->Float	Reading
P7	Adder state Fm	Byte10->Float	Reading
P8	Adder state Q	Byte10->Float	Reading
	<b>Alarms and markers</b>		
F1	Marker of unit "M" or "G"	Byte->Word	Reading
F2	Marker of emergency value substitution for T "A" or "	Byte->Word	Reading
F3	Marker of emergency value substitution for "A" or " "	Byte->Word	Reading
F4	State of alarms for Fm : "X", "L" or "H"	Byte->Word	Reading
F5	State of alarms for Q : "X", "L" or "H"	Byte->Word	Reading
F6	State of alarms for p : "X", "L" or "H"	Byte->Word	Reading
F7	State of alarms for T : "X", "L" or "H"	Byte->Word	Reading
	<b>Programmed discrete parameters</b>		
ND1	Input type F (0-20mA, 4-20mA)	Byte->Word	Reading
ND2	Input type p (0-20mA, 4-20mA)	Byte->Word	Reading
ND3	Characteristic	Byte->Word	Reading
ND4	Measurement unit (kg/h and MJ/h or t/h and GJ/h)	Byte->Word	Reading
ND5	Measurement of pressure p (absolute or manometer))	Byte->Word	Reading
ND6	Output 4-20mA (Fm or Q)	Byte->Word	Reading
ND7	Impulse output (Fm or Q)	Byte->Word	Reading
ND8	Alarm Fm (H) (off, on)	Byte->Word	Reading
ND9	Alarm Fm (L) (off, on)	Byte->Word	Reading
ND10	Alarm Q (H) (off, on)	Byte->Word	Reading
ND11	Alarm Q (L) (off, on)	Byte->Word	Reading
ND12	Alarm p (H) (off, on)	Byte->Word	Reading
ND13	Alarm p (L) (off, on)	Byte->Word	Reading
ND14	Alarm T (H) (off, on)	Byte->Word	Reading
ND15	Alarm T (L) (off, on)	Byte->Word	Reading

**Table 16. Addressing the FP1001 3.W Meter (Water Monitor) (continuation).**

Symb. Address	Variable in FP-1001	Conversion Type	Allowed Operation
	<b>Programmed numeric parameters</b>		
NL1	Range F max	Byte5->Float	Reading
NL2	Cut-off level Fc	Byte5->Float	Reading
NL3	Range p max	Byte4->Float	Reading
NL4	Emergency temperature TA	Byte3->Float	Reading
NL5	Emergency water temperature TWA	Byte3->Float	Reading

NL6	Design temperature To	Byte3->Float	Reading
NL7	Output current range	Byte5->Float	Reading
NL8	Alarm level Fm (H)	Byte5->Float	Reading
NL9	Alarm level Fm (L)	Byte5->Float	Reading
NL10	Alarm level Q (H)	Byte5->Float	Reading
NL11	Alarm level Q (L)	Byte5->Float	Reading
NL12	Alarm level p (H)	Byte4->Float	Reading
NL13	Alarm level p (L)	Byte4->Float	Reading
NL14	Alarm level T (H)	Byte3->Float	Reading
NL15	Alarm level T (L)	Byte3->Float	Reading
	<b>Working times</b>		
T1	Working time of device	Byte5+2->Dword	Reading
	<b>Conversion of letter and digital markers on bits</b>		
	letter "A"	bit 0	
	letter "G"	bit 1	
	letter "H"	bit 2	
	letter "L"	bit 3	
	letter "M"	bit 4	
	letter "N"	bit 5	
	letter "S"	bit 6	
	letter "X"	bit 8	
	space	bit 9	
	digit "0"	bit 0	
	digit "1"	bit 1	

**Table 17. Addressing the FP1001 3.4 Meter (Steam Monitor).**

Symb. Address	Variable in FP-1001	Conversion Type	Allowed Operation
	<b>Values of basic measurements</b>		
P1	Instantaneous value Fm	Byte7->Float	Reading
P2	Instantaneous value Fw	Byte6->Float	Reading
P3	Instantaneous value Q	Byte7->Float	Reading
P4	Instantaneous value Qw	Byte7->Float	Reading
P5	Instantaneous value deltaQ	Byte7->Float	Reading
P6	Instantaneous value p	Byte4->Float	Reading
P7	Instantaneous value T	Byte3->Float	Reading
P8	Calculation value p	Byte6->Float	Reading
P9	Calculation value u	Byte6->Float	Reading
P10	Instantaneous value Tw	Byte3->Float	Reading
P11	Calculation value u water	Byte6->Float	Reading
P12	Adder state Fm	Byte10->Float	Reading
P13	Adder state Fw	Byte10->Float	Reading
P14	Adder state Q	Byte10->Float	Reading
P15	Adder state Qw	Byte10->Float	Reading
P16	Adder state delta Q	Byte10->Float	Reading
	<b>Alarms and markers</b>		
F1	Marker of unit "M" or "G"	Byte->Word	Reading
F2	Marker of emergency value substitution for p "A" or "	Byte->Word	Reading
F3	Marker of emergency value substitution for T "A" or "	Byte->Word	Reading
F4	Marker of steam type "N", "S" or "A"	Byte->Word	Reading
F5	State of alarms for Fm : "X", "L" or "H"	Byte->Word	Reading
F6	State of alarms for Q : "X", "L" or "H"	Byte->Word	Reading
F7	State of alarms for p : "X", "L" or "H"	Byte->Word	Reading
F8	State of alarms for T : "X", "L" or "H"	Byte->Word	Reading

**Table 18. Addressing the FP1001 3.4 Meter (Steam Monitor) (continuation).**

Symb. Address	Variable in FP-1001	Conversion Type	Allowed Operation
	<b>Values of conditional adders</b>		
S1	Fm for saturated steam	Byte10->Float	Reading
S2	Fm for >Fm (H)	Byte10->Float	Reading
S3	Fm for <Fm (L)	Byte10->Float	Reading
S4	Fm for >Q (H)	Byte10->Float	Reading
S5	Fm for <Q (L)	Byte10->Float	Reading
S6	Fm for >p (H)	Byte10->Float	Reading

S7	Fm for <p (L)	Byte10->Float	Reading
S8	Fm for >T (H)	Byte10->Float	Reading
S9	Fm for <T (L)	Byte10->Float	Reading
S10	Q for saturated steam	Byte10->Float	Reading
S11	Q for >Fm (H)	Byte10->Float	Reading
S12	Q for <Fm (L)	Byte10->Float	Reading
S13	Q for >Q (H)	Byte10->Float	Reading
S14	Q for <Q (L)	Byte10->Float	Reading
S15	Q for >p (H)	Byte10->Float	Reading
S16	Q for <p (L)	Byte10->Float	Reading
S17	Q for >T (H)	Byte10->Float	Reading
S18	Q for <T (L)	Byte10->Float	Reading
	<b>Programmed discrete parameters</b>		
ND1	Steam type (dry or saturated)	Byte->Word	Reading
ND2	Measurement (p or T)	Byte->Word	Reading
ND3	Measuring method (reducer or Vortex)	Byte->Word	Reading
ND4	Return water (lack, only Tw, Tw and Fw)	Byte->Word	Reading
ND5	Input type F (0-20mA, 4-20mA,0-10kHz,01kHz)	Byte->Word	Reading
ND6	Input type Fw (0-20mA, 4-20mA,0-10kHz,01kHz)	Byte->Word	Reading
ND7	Input type p (0-20mA, 4-20mA,0-10kHz,01kHz)	Byte->Word	Reading
ND8	Input type T (0-20mA, 4-20mA,Pt100)	Byte->Word	Reading
ND9	Characteristic F	Byte->Word	Reading
ND10	Measurement unit (kg/h and MJ/h or t/h and GJ/h)	Byte->Word	Reading
ND11	Pressure measurement p (absolute or manometer.)	Byte->Word	Reading
ND12	Output 4-20mA (Fm or delta Q)	Byte->Word	Reading
ND13	Impulse output (Fm or delta Q)	Byte->Word	Reading
ND14	Alarm Fm (H) (off, on)	Byte->Word	Reading
ND15	Alarm Fm (L) (off, on)	Byte->Word	Reading
ND16	Alarm Q (H) (off, on)	Byte->Word	Reading
ND17	Alarm Q (L) (off, on)	Byte->Word	Reading
ND18	Alarm p (H) (off, on)	Byte->Word	Reading
ND19	Alarm p (L) (off, on)	Byte->Word	Reading
ND20	Alarm T (H) (off, on)	Byte->Word	Reading
ND21	Alarm T (L) (off, on)	Byte->Word	Reading

**Table 19. Addressing the FP1001 3.4 Meter (Steam Monitor) (continuation).**

Symb. Address	Variable in FP-1001	Conversion Type	Allowed Operation
	<b>Programmed Discrete Parameters</b>		
ND22	Conditional adder Fm for saturated steam (off, on)	Byte->Word	Reading
ND23	Conditional adder Fm for >Fm (H) (off, on)	Byte->Word	Reading
ND24	Conditional adder Fm for <Fm (L) (off, on)	Byte->Word	Reading
ND25	Conditional adder Fm for >Q (H) (off, on)	Byte->Word	Reading
ND26	Conditional adder Fm for <Q (L) (off, on)	Byte->Word	Reading

## Communication Drivers

ND27	Conditional adder Fm for >p (H) (off, on)	Byte->Word	Reading
ND28	Conditional adder Fm for <p (L) (off, on)	Byte->Word	Reading
ND29	Conditional adder Fm for >T (H) (off, on)	Byte->Word	Reading
ND30	Conditional adder Fm for <T (L) (off, on)	Byte->Word	Reading
ND31	Conditional adder Q for saturated steam (off, on)	Byte->Word	Reading
ND32	Conditional adder Q for >Fm (H) (off, on)	Byte->Word	Reading
ND33	Conditional adder Q for <Fm (L) (off, on)	Byte->Word	Reading
ND34	Conditional adder Q for >Q (H) (off, on)	Byte->Word	Reading
ND35	Conditional adder Q for <Q (L) (off, on)	Byte->Word	Reading
ND36	Conditional adder Q for >p (H) (off, on)	Byte->Word	Reading
ND37	Conditional adder Q for <p (L) (off, on)	Byte->Word	Reading
ND38	Conditional adder Q for >T (H) (off, on)	Byte->Word	Reading
ND39	Conditional adder Q for <T (L) (off, on)	Byte->Word	Reading
	<b>Programmed Numeric Parameters</b>		
NL1	Range F max	Byte5->Float	Reading
NL2	Cut-off level Fc	Byte5->Float	Reading
NL3	Range Fw max	Byte5->Float	Reading
NL4	Cut-off level for Fw	Byte5->Float	Reading
NL5	Range p max	Byte4->Float	Reading
NL6	Emergency pressure pA	Byte4->Float	Reading
NL7	Design pressure	Byte4->Float	Reading
NL8	Range T min	Byte3->Float	Reading
NL9	Range T max	Byte3->Float	Reading
NL10	Emergency temperature TA	Byte3->Float	Reading
NL11	Design temperature	Byte3->Float	Reading
NL12	Emergency water temperature	Byte3->Float	Reading
NL13	Output current range	Byte5->Float	Reading
NL14	Alarm level Fm (H)	Byte5->Float	Reading
NL15	Alarm level Fm (L)	Byte5->Float	Reading
NL16	Alarm level Q (H)	Byte5->Float	Reading
NL17	Alarm level Q (L)	Byte5->Float	Reading
NL18	Alarm level p (H)	Byte4->Float	Reading
NL19	Alarm level p (L)	Byte4->Float	Reading
NL20	Alarm level T (H)	Byte3->Float	Reading
NL21	Alarm level T (L)	Byte3->Float	Reading

## Driver Configuration

The driver configuration is defined in the *Current Data* module, in the channel operating according to FP1001 driver.

**Log file**

Meaning

- the item allows to define a file to which all the diagnostic messages of the FP1001 driver and the information about the content of the telegrams received and sent by the FP1001 driver will be written. If

- the item doesn't define its full path, the log file will be created in the current directory.
- Default value - by default, the file log isn't created.
- Telegrams log**
- Meaning - the item allows to write to the log file the content of telegrams transmitted between Asix and FP1001 devices. Writing the telegrams content to the log, file should be used only during the Asix start-up.
- Default value - by default, the FP1001 driver does not write the content of telegrams to the log file.
- Transmission delay**
- Meaning - the item is used to declare a pause between transmissions. The pause is expressed as a number of 10 ms intervals.
- Default value - by default, the item is set to 1 (10 msec).
- Updating**
- Meaning - the item defines a number of seconds, after which the driver updates the contents of its internal buffers by reading the measure data from FP1001 devices FP1001.
- Default value - by default, the item is set to 1 (updating every 1 second).
- Number of repetitions**
- Meaning - the item allows to define a number of repetitions in case of occurring the transmission error.
- Default value - by default, the item is set to 0 (no repetitions).

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

- Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.
- Option value:  
YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

- Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

## Communication Drivers

- computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.41 GFCAN - Driver of CANBUS Protocol for CanCard

### Driver Use

The GFCAN driver is used for data exchange between devices with the CAN network interface and an Asix system computer provided with a CAN network communication processor card manufactured by Garz & Fricke Industrieautomation GmbH and provided with the Garz & Fricke CAN driver for Windows NT" version 1.0.

Parameterization of GFCAN driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the GFCAN driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: GFCAN

The GFCAN driver is loaded as a DLL automatically.

### Addressing the Process Variables

Values of process variables are transferred in telegrams sent by controllers connected to the CAN network. Each telegram consists maximally of 8 bytes, which may be identified as:

bytes with indexes 1 - 8	(type BY),
16-bit numbers with indexes 1 - 4	(type WD),
32-bit numbers with indexes 1 - 2	(type DW),
32-bit floating-point numbers with indexes 1 - 2	(type FP).

The GFCAN driver differs the following access types to process variables:

read only	(type R_),
write only	(type W_),
write/read	(type RW_).

The addressing the process variables consists in the indication of:

- access type (R\_, W\_ or RW\_);
- variable type (BY, WD, DW, FP);
- telegram no. (for variables with RW\_ access type it is the telegram number that is used to read the variable);
- index within the telegram (for variables with RW\_ access type it is the index in the telegram that is used to read the variable);
- for variables with RW\_ access type it is necessary to declare additionally:
  - a/ telegram number that is used to read the variable,
  - b/ index in the telegram that is used to write the variable.

The syntax of symbolic address which is used for variables belonging to the GFCAN driver channel is as follows:

`<access_type><variable_type><tel>.<index>[.<tel>.<index>]`

where:

<i>access type</i>	- access type to a process variable:
R_	- only reading,
W_	- only writing,
RW_	- reading and writing,
<i>variable_type</i>	- process variable type:
BY	- variable of the byte type,
WB	- variable of the 16-bit number type,
DW	- variable of the 32-bit number type,
FP	- variable of the 32-bit floating-point number type.
<i>tel</i>	- telegram number,
<i>index</i>	- index within the telegram.

### EXAMPLE

X1, bytes 1-4 of telegram 31,R_FP31.2,	NONE, 1, 1, NOTHING_FP
X2, word no. 3 of telegram 31,R_WD31.3,	NONE, 1, 1, NOTHING
X3, state of burners, RW_BY31.1.35.3,	NONE, 1, 1, NOTHING_BYTE
X4, valve setting, RW_WD32.1.34.1,	NONE, 1, 1, NOTHING

The variable X1 is a variable of the 32-bit floating-point number type transferred to the Asix system in bytes 1,2,3 and 4 of the telegram no. 31.

The variable X2 is a variable of the 16-bit number type, the value of which is transferred to the Asix system by bytes 5 and 6 (third word) of telegram no.31. The value of the variable can't be modified by the application (variable only to read).

Value of the variable X3 is transferred to the Asix system by means of the byte no. 1 of the telegram no. 31. The value exchange of the variable X3 consists in sending from the Asix system the telegram no. 35, the byte no. 3 of which includes the required value of the variable X3.

## Driver Configuration

GFCAN driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **GFCAN** section.

**Section name: GFCAN**

**Option name: TRANSMISSION\_SPEED**

**Option value: baud\_id**

Meaning - the item is used to declare a transmission speed in the CAN network.

Default value - by default, the item is set to 500 kB.

Parameter:

baud\_id - identifier of transmission speed in the CAN network:

1000 -	1 MB
800 -	800 kB
500 -	500 kB
250 -	250 kB
125 -	125 kB

100	-	100 kB
50	-	50 kB
20	-	20 kB
10	-	10 kB

**EXAMPLE**

An exemplary declaration of the transmission speed of 125 kB:

*Section name:* GFCAN  
*Option name:* TRANSMISSION\_SPEED  
*Option value:* 125

- Section name:** GFCAN
- Option name:** NETWORK\_CONTROL
- Option value:** *number*

Meaning - the item allows to test the reception of telegrams from the CAN network. It defines the maximal time between receptions of successive telegrams with the same number. In case of exceeding this time, the process variables bound with such telegram will be provided with an error status. If additionally in the same time any telegram was not received from the CAN network, a message about a lack of telegrams in network is generated in 'Control Panel'.

Default value - by default, the GFCAN driver does not check reception of telegrams.

Parameter:  
*number* - maximal number of seconds, which may pass between successive telegrams with the same number.

**EXAMPLE**

An exemplary declaration of checking reception of telegrams every 5 seconds:

*Section name:* GFCAN  
*Option name:* NETWORK\_CONTROL  
*Option value:* 5

- Section name:** GFCAN
- Option name:** TELEGRAM\_TRACE
- Option value:** YES|NO

Meaning - the item controls transferring to the operator panel the messages about telegrams that have been received from the CAN network. A message includes the number of telegram, number of bytes and content of individual telegrams in hexadecimal form.

Default value - by default, the contents of telegrams are not displayed.

**EXAMPLE**

An exemplary declaration of tracing the content of received telegrams:

*Section name:* GFCAN  
*Option name:* TELEGRAM\_TRACE  
*Option value:* YES

- ☑ **Section name:** GFCAN
- ☑ **Option name:** CONTROL\_TRACE
- ☑ **Option value:** YES|NO

Meaning - the item controls transferring to the operator panel the messages about control telegrams that have been sent from the asix system computer to controllers. A message includes the number of control telegram, number of bytes and telegram contents in hexadecimal form.

Default value - by default, the contents of telegrams are not displayed.

#### EXAMPLE

An example of tracing the control telegrams:

*Section name:* GFCAN  
*Option name:* CONTROL\_TRACE  
*Option value:* YES

- ☑ **Section name:** GFCAN
- ☑ **Option name:** LOG\_FILE
- ☑ **Option value:** file\_name

Meaning - the item allows to define a file to which all diagnostic messages of GFCAN driver and information about content of telegrams received and sent by the GFCAN driver will be written. If the item does not define a full path, the log file will be created in the current directory. The log file should be used only while the asix system start-up.

Default value - by default, the contents of telegrams are not displayed.

- ☑ **Section name:** GFCAN
- ☑ **Option name:** MAX\_MOTOROLA\_TEL
- ☑ **Option value:** number

Meaning - the item allows to specify a maximal number of telegram, the content of which will be converted according to MOTOROLA format. All the telegrams with numbers, which are bigger than declared by means of the item MAX\_MOTOROLA\_TEL, will be converted according to INTEL format.

Default value - by default it is assumed that all telegrams are converted according to INTEL format.

#### EXAMPLE

An example of declaration as a result of which the telegrams with numbers up to 150 inclusive are converted according to MOTOROLA format:

*Section name:* GFCAN  
*Option name:* MAX\_MOTOROLA\_TEL  
*Option value:* 150

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

- Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.
- Option value:  
 YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

- Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.
- Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

- Purpose - with this option deployed, local control in a channel will be disabled.
- Option value:  
 YES / NO
- The default value - NO.

### **Data Validity Period**

- Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.
- Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.
- The default value - by default, no time for data validity is set.

## 1.42 Global - Exchange of Data Between the Asix System Application and a Swap File

### Driver Use

Global driver is used to exchange data between the Asix system application and a so-called swap file, which is a container for the driver's current variable parameters (name, status, value, timestamp). In an application run across multiple computers, the contents of the swap file can be synchronized between different computers. This way changes in variable values occurring on a single computer can be propagated to all other computers.

Parameterization of Global driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the Global driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* Global

**Global / Channel parameters** tab:

*Swap file*  
*Other server directories*  
*Synchronization period*  
*Log file*  
*Log file size*

where:

- Swap file* - name of the swap file in which the variables will be stored;  
default value: global.cfg
- Other server directories* - option is used to determine directories in which the driver will search swap files to synchronize the content of swap file declared in *Swap file* option; the name of searched file is the same as the name of file declared in *Swap file* option. Item applies to installation in which the driver CtGlobal is used to exchange data between computers.
- Synchronization period* - period (in milliseconds) between successive tries to synchronize the contents of a swap file (important for multicomputer applications). The default value is 1000 milliseconds.
- Log file* - all diagnostic messages about driver work are saved into a text log file. Default value: ctglobal.log.
- Log file size* - the item allows to define the log file size in MB; by default, it is assumed that the log file has a size of 10 MB.

**EXAMPLE**

*Name:* KANAL  
*Driver:* CtGlobal

Channel parameters:

*Other server directories:*  
*x:* \tmp\Global  
*y:* \tmp\Global  
*z:* \tmp\Global

*Log file:* d:\tmp\test\kanalGlobal.log  
*Log file size:* 20

The driver provides variables of the scalar and chart types. The driver specifies the size of a variable based on the conversion function type.

The symbolic address may be any string of characters and is ignored by the driver.

The SYNCH symbolic address is introduced to enable the readout of the status of synchronization (UI2 number) with files.

Bit 0 - status of synchronization with the local file  
 Bit 1, 2 ... - status of synchronization with files of other stations  
 A bit with value 0 means correct synchronization.

Examples of variable declarations (variables belong to the logical channel CHANNEL):

*Name:* JJ\_1  
*Description:* zmienna typu WORD  
*Address:* xxx  
*Channel:* KANAL  
*Elements Count:* 1  
*Sample Rate:* 1  
*Conversion Function:* NOTHING

*Name:* JJ\_2  
*Description:* zmienna typu INT  
*Address:* xxx  
*Channel:* KANAL  
*Elements Count:* 1  
*Sample Rate:* 1  
*Conversion Function:* NOTHING\_INT

*Name:* JJ\_3  
*Description:* zmienna typu FLOAT  
*Address:* xxx  
*Channel:* KANAL  
*Elements Count:* 1  
*Sample Rate:* 1  
*Conversion Function:* NOTHING\_FP

*Name:* JJ\_4  
*Description:* zmienna typu DWORD  
*Address:* xxx  
*Channel:* KANAL  
*Elements Count:* 1

*Sample Rate:* 1  
*Conversion Function:* NOTHING\_DW

*Name:* JJ\_5  
*Description:* zmienna typu TEKST  
*Address:* xxx  
*Channel:* KANAL  
*Elements Count:* 20  
*Sample Rate:* 1  
*Conversion Function:* NOTHING\_TEXT

## Driver Parameters

The driver configuration is defined in the *Current Data* module, in the channel operating according to Global driver.

### **Log file**

Meaning - the option allows to define a file to which all diagnostic messages of the Global driver are written.

Option value:  
    *file\_name* - file name.  
Default value: ctglobal.log.

### **Log file size**

Meaning - the item allows to define the log file size.

Option value:  
    *number* - number in MB.  
Default value: 10.

### EXAMPLE

*Log file:* d:\tmp\test\ctGlobal.log  
*Log file size:* 30

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:  
    YES / NO  
    *computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
 YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.43 IEC61850 - Driver for IEC61850 Protocol

### Driver Use

The following document refers to the IEC61850 protocol.

### Declaration of Transmission Channel

The IEC61850 driver channel is a group of Asix system variables supported in the same way, which values are located in a single IED device (Intelligent Electronic Device).

Declaration of the transmission channel is as follows:

**Standard** tab:

**Name:** *logical name of the transmission channel*

**Driver:** IEC61850

**IEC61850** tab:

#### **Channel parameters**

The channel parameters are listed in the section *Channel Parameters*.

The channel parameters can be placed in the section with the same name as the channel name, but the parameters that define the address of remote device have to be declared in the channel declaration. The parameters defining the connection with a remote device - there are: **ip**, **aeq**, **apid**, **psel**, **ssel** and **tsel**. If you have the file of the SCL type (Substation Configuration Language), e.g. **icd** or **cid** file, the above parameters can be replaced by the parameter **scl**. An exemplary channel declaration could be in the following form:

```
IEC1 = IEC61850, scl=Siprotec.icd
```

If, however, address parameters included in the file SCL are not compatible with the reality, they can be corrected by declaring them in the channel declaration. E.g. if the actual device IP address differs, it can be define in the declaration:

```
IEC1 = IEC61850, scl=Turow.cid, IP=192.168.0.5
```

Some of the channel parameters refers to the connection with a remote device but not to an individual specified channel. In such a case the parameters have to be identical in all channel declarations or omitted, with the exception of declaration of first channel for the given connection. These parameters include: **ied**, **ied2**, **log**, **dump**, **ExpITmo**, **itcom**. Multiple channels can be connected with an individual IED device.

The SCL file is not demanded for proper operation of the driver, but its declaration is recommended.

The values of Asix application (except pseudovariables) are the values of IED data attributes. Linking an Asix application variable with a datum specified attribute is realized by declaring the name (precise reference) of datum attribute as the address of the variable. Addressing the variables is described in the section *Variable Declaration*.

- **Channels for Polling**

The channel declaration without the parameters `brcb` and `urcb` defines the group of variables which values will be read from IED by sending to a device read requests in accordance with sampling period (polling) declared for a specific variable.

Due to increased traffic in the network, this type of reading variables is not recommended.

- **Channels with Reporting**

The channel based on the reporting receives the variable values sent spontaneously by a device at the moment of change, actualization or quality change of the variable value. The channel declaration based on reporting has to contain the parameter **brbc** or **urcb**. This parameter specifies the name (exactly reference) of reporting block included in IED. Reporting block contains all the parameters specifying the way of reporting. One channel with reporting can be associated with exactly one reporting block.

#### **DataSet.**

The list of Asix variables to be reported defines the object DataSet included in IED. It contains the list of names (exactly reference) of IED device data attributes which are to be reported. One DataSet can be associated with multiple reporting blocks. If the reporting block is not permanently associated with a DataSet, then its name should be given by the parameter **ds**. In this case, the driver will automatically associate DataSet with the reporting block. If DataSet defined like this does not exist, the driver will create a new DataSet. If the list of data attributes included in the existing DataSet does not include all the Asix application variables defined in the channel, then the driver will remove the previous DataSet and will create the new one. If the existing DataSet includes redundant data attributes, the driver will call an appropriate warnings to the application control panel.

If the device does not allow for dynamic creation of DataSet, then the parameter **ConfDS** = Tak should be declared, which means that DataSet is permanently configured by the use of a suitable program or by the manufacturer. In this case, the driver will call warning to the control panel only when the contents of DataSet is not compatible with the channel variable list.

If the device requires manual configuration of DataSet, you should remember to include the parameters `.q` and `.t` to the DataSet if it includes attributes with which quality and time stamp are associated. The quality (`.q`) is associated with the following attribute names:

`"stVal", "general", "phsA", "phsB", "phsC", "dirGeneral", "dirPhsA", "dirPhsB", "dirPhsC", "dirNeut", "cnt", "actVal", "mag", "range", "cVal", "instMag", "Har", "phsAHar", ".phsBHar", ".phsCHar", "neutHar", "netHar", "resHar", "phsABHar", "phsBCHar", "phsCAHar", "valWTr", "setMag", "neut"`.

All these attributes (except `"instCMag"`) have the associated attribute `.t` being a time stamp. If the attributes `.q` and `.t` are not associated, the driver will assign the current time and the quality good to the variable.

#### **Options of Triggering Reports**

The reports can be triggered as a result of attribute value: change (**dchg**), actualization (**dupd**), quality change (**qchg**), integrity control (**integ**) as well as request of reading

reported values (**GI** - general interrogation). The triggering options are defined by the parameter **TrgOps** which is the list of option names given below in parentheses separated by commas. If the name of option is prefixed with "-" (minus), then the given option is disabled. The option **GI** is determined automatically by the driver and its change by parameterization is ineffective. Similarly, the option **integ** is set automatically if the parameter **IntgPd** of the channel has the value greater than zero. Other options are set by default. If the given option is not required, it should be replaced in the parameter **TrgOps**, prefixing it with "-" (minus). For example, according to the documentation of MICOM device, reporting can not be triggered when the option **dupd** is enabled. In this case, "**TrgOps = -dupd**" should be declared.

In the channels with reporting you can place the variable with the address "@GI" (general interrogation). Writing a non-zero value to this variable will cause triggering the report.

### Channels with Unbuffered Reporting

Channels with unbuffered reporting are used when potential loss of attribute value changes, caused by closing the application, is irrelevant. An unbuffered channel has to include the parameter **urcb**. In the case of unbuffered channels, DataSet can be temporary, as long as the device allows it. A temporary DataSet is removed from IED after connection break. The driver creates automatically the temporary DataSet after establishing connection. For a temporary DataSet the parameter **ds** has the following form:

```
ds = @[dataset_name]
```

If *dataset\_name* is omitted, then the driver will create a DataSet with the same name as the channel name. If a device does not support a temporary DataSet, then they are supported in the way described in the previous section.

### Channels with Buffered Reporting

The channels with buffered reporting are used all changes of attribute values are to be recorded in IED at the time when the Asix application does not work. It applies to especially variables which are associated with alarms. The channel with buffered reporting has to have the parameter **brcb** defined.

Each report stored in the buffered report block IED has a unique identifier with the name **EntryId**. The **EntryId** identifier is transferred together with the report. If the driver IEC61850 will process all the data contained in the report (also generate reports), the value of this identifier is stored in the file named:

```
IEC61850-channel_name.EntryID
```

After closing the Asix application, the file is read and its contents is sent to IED to indicate which parameters have already been handled and to prevent their retransfer and regeneration of false alarms. In the case of change of channel name the name of the file should also be changed adequately. The driver can signal the error of sending ID to IED - when e.g. the device was reset and reporting block did not contain any reports.

Starting the Asix application after a break causes that for a certain time the driver will be receiving reports about events from the past. It may cause showing outdated states of process variables. To avoid this, the driver reads identifier of the latest report and updates the variables only after receiving a current report. This procedure is possible only for devices meeting Technical Issue nr 190. If the device does not meet this requirement (e.g. MICOM P139), then the parameter "**ToESync = Tak**" should be declared - and the driver will use the field **TimeOfEntry** for synchronization. It will also cause sending additional value with each report. To disable synchronization, the parameter "**BRCBSync = nie**" should be declared.

In the buffered channel you can define pseudovvariable with the address **@BufOvfl** (**NIC\_BYTE**), which takes the value of 1 if there was buffer overflow in the reporting block and data loss. The driver can associate an alarm with this variable.

In the case of devices which meet the requirements of Technical Issue 300 (e.g. SIPROTEC), it is necessary to send the ResvTms parameter value to the device to start buffered reporting. There is the time in seconds running from the moment of connection loss with the device, during which the device reserves reporting block for the application establishing connection with the same address as before. After this time, any application in the network can reserve that reporting block. The default value of this parameter is 600 seconds. Notice that after ip address change on which the application is run there will be no possible to run reporting until this time expires.

It is possible to put a variable with the address "@Purge" in the channels with buffered reporting. Writing a non-zero value of this variable will remove all reports from the buffer.

- **Examples of Channel Declaration**

Declaration of buffered channel:

Section	Parameter	Value
ASMEN	IEC1	IEC61850, scl=SIPROTEC.icd, brcb=CTRL/LLN0.brCbA01
IEC1	ds	CTRL/LLN0.AsixDs
IEC1	Log	iec1.log
IEC1	Dump	iec1.xml

The reporting block has the name (reference) "CTRL/LLN0.brCbA01". The block reports the attributes contained in DataSet named "CTRL/LLN0.AsixDs". Diagnostic informations will be recorded to the file "iec1.log" and detailed informations concerning data exchange will be stored in the file "iec1.xml".

Declaration of unbuffered channel.

Section	Parameter	Value
ASMEN	IEC1	IEC61850, scl=SIPROTEC.icd, urcb=CTRL/LLN0.urCbA01
IEC1	ds	@

The reporting block has the name (reference) "CTRL/LLN0.urCbA01". The block reports the attributes contained in dynamically created temporary DataSet named "@IEC1"..

## Variable Declaration

The declaration of a variable associated with a data attribute in IED is as follows:

*datum\_attribute\_reference*[[*additional\_parameters*]]

*Datum\_attribute\_reference* is a device logical name in IEDended with a slash and then a name of logical node, datum name and attribute names separated by dots. The name of logical device should be the same as the one in the given device documentation or in SCL files, i.e. it can not be preceded by the IED name. For example, if the logical device named TurowSystem is in the device physically, then variable names will begin with the name System, e.g.:

Measurements/MmuPriMSTA1.MaxAmps.mag.f

*Dditional\_parameters* have the following form:

*Par1:par2:....*

Possible parameters:

*Alarm\_definition* (described in the section *Alarms*)

P - variable supported in polling mode

CMn - declaration of control model for the variable (described in the section *Control*).

## Control

For MiCOM P139 it is possible to control only the data which was properly configured for this purpose with the use of IED Configurator from MiCOM S1 Studio.

To actuate a given variable, look through the attribute "Oper\$ctlVal" by IEC Browser and determine the address of Asix variable for the attribute, as it is describe in the previous section. For this address add .@CM at the end.

I.E. Control/CSWI1.Pos.Oper.ctlVal.@CM.

For the driver MiCOM P139 it can not be read form the device the way the control has to be performed, it should be indicated by adding "[CMn]" to the address, where "n" is the number of control method determined by the program IED Configurator. The "n" can have the following values:

- 1 - direct-with-normal-security
- 2 - sbo-with-normal-security
- 3 - direct-with-enhanced-security
- 4 - sbo-with-enhanced-security

For example:

Control/CSWI1.Pos.Oper.ctlVal.@CM[CM3]

Giving the parameter CMn for other devices is not necessary if the device has properly set the value of the attribute "ctlModel" of controlled datum. The variable with this address should have the conversion function NOTHING\_BYTE. Writing the value of 1 to this variable will cause activation to the state ON, and the value of 0 will cause actuation to the state OFF. The Asmen channel of controlled variable has to have the parameter "Simple\_Control=Tak" declared.

The string ".Oper" can be omitted in control variable (there is only one dot).

Actuation of variables of "...-enhanced-security" types causes that the device generates the report with control results. To read the results, the following variables have to be defined as follows:

Control/CSWI1.Pos.@Err (conversion function NOTHING\_DW)

and

Control/CSWI1.Pos.@AddCause (conversion function NOTHING\_DW)

According to the IEC61850 standard the variable with the address "...@Err " can take the following values:

- (0) No Error
- (1) Unknown
- (2) Timeout Test Not OK
- (3) Operator Test Not OK

The variable with the address "...@AddCause" can take the following values:

Unknown	0
Not-supported	1
Blocked-by-switching-hierarchy	2
Select-failed	3
Invalid-position	4
Position-reached	5
Parameter-change-in-execution	6
Step-limit	7
Blocked-by-Mode	8
Blocked-by-process	9
Blocked-by-interlocking	10
Blocked-by-synchrocheck	11
Command-already-in-execution	12
Blocked-by-health	13
1-of-n-control	14
Abortion-by-cancel	15
Time-limit-over	16
Abortion-by-trip	17
Object-not-selected	18

## Alarms

The change of IEC61850 driver variable can be associated with alarm generation. To do this, the additional parameter - alarms definition should be added:

The alarm definition can take the following form:

*ABalarm\_no bit\_no*

or

*AXalarm\_no bit\_no*

or

*ALalarm\_no relation value*

AB-type alarms are similar to the bitmap strategy of *Asix*. The *Alarm\_no* defines the first alarm number. This alarm is associated with the bit *bit\_no* of a variable value. Consecutive bits of the variable value are associated with alarms with sequential numbers. Setting the bit to 1 generates a new alarm and setting the bit to 0 end the alarm.

AX alarms work in the same way as AB alarms, but the definition refers only to one particular bit *bit\_no*. Other bits are not analyzed. The first bit of a value has the number 0.

AL-type alarm is generated when the variable value meets the relation. There are the following relations possible:

= or ==	- equality
!= or <>	- inequality
>	- greater than
>=	- greater than or equal
<	- less than
<=	- less than or equal

### Examples:

Control/CSWI2.Pos.stVal[AL300=1:AL301=3]

## Communication Drivers

The alarm 301 will be generated when the value of the variable (switch position) is 1 (ON). The alarm 301 will be generated when the value of the variable is 3 (intermediate switch position).

MEAS/MMXU1.PPV.phsAB.instCVal.mag.f[AL500>32,76]

The alarm will be generated if the variable value exceeds 32,76.

CTRL/LLN0.Mod.stVal[AX600 1]

Setting the bit 1 of the variable value will generate the alarm with the number 600. The parameter "Global\_alarms" defines type generated alarms and if the parameter has the value of "YES", then global alarms are generated. Because the parameter applies to the driver, it can be declared in the section "IEC61850".

### Switching Setpoint Sets

To change the number of current setpoint set, the variable with the address LD/LLN0.SGCB.ActSG should be defined:

where:

LD - is the name of logical device containing sets of setpoints (e.g. System/LLN0.SGCB.ActSG)

Writing the value to this variable will activate the appropriate set of setpoints.

### List of Channel Parameters

Name	Description	Default value
IP	Address of the remote device IED	No default value
aeq	Application entity qualifier	12
appid	Application identifier	1.1.1.999.1
psel	Presentation selector	1
ssel	Session selector	1
tssel	Transport selector	1
ExpITmo	The maximum period of time (in seconds) for initial readout of data from the device.	40
scl	The path and name of the configuration file SCL.	No default value
ied	IED name	IED name in the file *.icd, *.cid
ied2	The name IED - if there is the other than actual name of the device declared.	No default value
itcom	If the option is set to "Yes", the communication module of the driver IEC61850 is started as a separate process (COM server). The option should be used if there is a suspicion that the driver is in conflict with other Asix system drivers using the protocol TCP/UDP. You have to restart the program ITLibCOMServer.exe (COM server) with the use of the command ITLibCOMServer.exe .regserver before using the option.	No
origin	Text specifying the object that executes datum control	Asix
interlock	This parameter determines whether interlock is to be used when executing control operation. Possible values: "Yes", "No".	No

synchrocheck	This parameter determines whether synchrocheck is to be used when executing control operation. possible values: "Yes", "No".	No
test	This parameter determines whether when executing control operation, it have to be performed in test mode. Possible values: "Yes", "No".	No
ConfDS	The parameter determines whether DataSet used in the channel with reporting is to be reconfigured in the device or created by the driver. Possible values: "Yes", "No".	No
ToESync	This parameter determines whether the report field TimeOfEntry instead of the field EntryID is to be used to determine received report as historical or current one. Possible values: "Yes", "No".	No
BRCBSync	The parameter determines whether the channel is to be synchronized to take into account only current reports to determine Asix variable values. Possible values: "Yes", "No".	Yes
TrgOps	The parameter is the list of names of events that trigger report generation. The name can be preceded by a "-" (minus), which means that the name will be disabled. Acceptable names of events that trigger reports: <b>dchg</b> - datum value change <b>qchg</b> - datum quality change <b>dupd</b> - datum value updating <b>GI</b> - general inquiry <b>integ</b> - periodical inquiry <b>all</b> - all options	<b>dchg, dupd, qchg</b>
OptFlds	The parameter is the list of names of report generated by IED components. Acceptable names of events that trigger reports: <b>SqN</b> - consecutive number <b>Toe</b> - report generation time <b>rc</b> - report generation reason <b>ds.</b> - DataSet reference <b>dr</b> - report data reference <b>bo</b> - buffer overflow <b>ei</b> - report ID <b>cf</b> - report configuration version <b>seg</b> - report segment number <b>all</b> - all components of the report	No default value
Simple_Control	This parameter determines whether simplified control is to be used in the channel.	No
ds	Reference to DataSet	No default value
BufTm	Timeout (in milliseconds) from the first event triggering the report to consecutive events before the report will be generated.	-1 (parameter will not be set, i.e. the previous time will be used)
IntgPd	The interval (in milliseconds) of sending periodical reports by IED.	-1 (parameter will not be set, i.e. the previous time will be used)
ResvTms	Time (in seconds) of buffered report block reservation for a particular client after losing connection with IED.	600

Diag	The parameter specifies the kind of additional diagnostic information stored in the file specified by the parameter "Log". It is the list of information group separated by commas.	No default value
Log	The name of the file to which all the diagnostic information will be written. Optionally, the name can be preceded by the full path. After the name you can declare log file size (in MB).	No default value
Dump	The name of the file to which all the information sent to/from IED will be written. After the name you can declare the maximum file size (in MB).	No default value
CurrTm	If the parameter is set to "Yes", all the variables will be stamped with the time equal the current PC time.	No

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

 **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.44 K3N - Driver of OMRON K3N Meters Family Protocol

### Driver Use

The K3N driver is used for data exchange between the OMRON K3N meters family and Asix system computers. The communication is executed according to the CompoWay/F protocol from OMRON via standard serial ports of an Asix computer.

Parameterization of K3N driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the K3N driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: K3N

**K3N** tab:

*Channel parameters*:

*id, port [ , baud, character, parity, stop ]*

where:

<i>id</i>	- controller number in the K3N network;
<i>port</i>	- port name: COM1, COM2 etc.;
optional parameters:	
<i>baud</i>	- transmission speed;
<i>character</i>	- number of bits in a character;
<i>parity</i>	- parity check type;
<i>stop</i>	- number of stop bits.

If the optional parameters are not given, then the factory settings of meter are assumed as default values, i.e.:

- transmission speed: 9600 baud,
- bits in a character: 7,
- parity check: EVEN,
- number of stop bits: 2.

#### EXAMPLE

The declaration of the logic channel named CHAN1, which operates according to the K3N protocol and exchanges data with the meter numbered 1 through the COM2 port with default transmission parameters:

*Channel / Name*: CHAN1

*Driver*: K3N

*Channel parameters*: COM2

The K3N driver is loaded as a DLL automatically.

## Addressing the Process Variables

The syntax of symbolic address which is used for variables belonging to the K3N driver channel is as follows:

`<type>[.<category>.<index>]`

where:

<i>type</i>	- variable type;
<i>category</i>	- category within the type (for some types);
<i>index</i>	- index within the category (for some types).

Symbols of variable types (in parentheses the type of raw variable value is given):

**VW** - values of variables transferred in the form of 16-bit unsigned numbers (WORD). It concerns to Status data variable (category C0, index 3);

**VL** - values of variables transferred in the form of 32-bit signed numbers (LONG). It concerns the variables of the category C0 except the variable Status data;

**PW** - values of parameters transferred in the form of 16-bit unsigned numbers (WORD). It concerns to all the variables of the category 8000 and 8824;

**PL** - values of parameters transferred in the form of 32-bit signed numbers (LONG). It concerns to all the variables of the category C00C and 8824;

**STAT** - status of work mode (WORD);

**INFO** - supplemental information transferred with the status of work mode (WORD);

**RST** - execution of the command *reset of minimal and maximal values* (WORD);

**FCLR** - execution of the command *clear forced-zero setting* (WORD);

**MODE** - setting the operating mode of the meter (*Run or Setting Mode*) (WORD);

**CTRL** - setting the programming mode of the meter (*Remote programming or Local programming*) (WORD).

*Category* and *index* must be given for types **VW**, **VL**, **PW**, **PL**. The list of legal symbols (numbers) of category and index ranges within individual category contains the documentation "*Communication Output-type Intelligent Signal Processor - OPERATION MANUAL*" Cat. No. N96-E1-1 (pkt 1-5 'Memory/Parameters Area Details').

Values of **VW**, **VL**, **PW** or **PL** type variables may be read and modified. A correct reading of the **VW**, **VL**, **PW**, **PL** type variables is possible only when the meter is set in the *Run mode*.

Values of **INFO**, **STAT** type variables may be only read.

The **RST**, **FCLR**, **MODE**, **CTRL** type variables are used only to execute commands controlling the operating mode of the meter and it is not possible to read them.

Sending the control by using a **MODE** type variable causes:

- switching the meter to the *Run mode* if the control value of the variable is set on 0,
- switching the meter to *Setting mode* if the control value of the variable is different from 0.

Sending the control by using a **CTRL** type variable causes:

- switching the meter to the *Local programming* mode if the control value of the variable is set on 0,
- switching the meter to the *Remote programming* mode if the control value is different from 0.

## EXAMPLES OF VARIABLE DECLARATIONS

```
# Present value: variable no. 0 from the category C0
JJ_0, VL.C0.0, CHAN1, 1, 1, NOTHING_LONG

# Maximum value: variable no. 1 from the category C0
JJ_1, VL.C0.1, CHAN1, 1, 1, NOTHING_LONG

# status value (Status data): variable no 3 from the category C0
JJ_2, VW.C0.3, CHAN1, 1, 1, NOTHING

# current operating mode of the meter (Run, Setting mode)
JJ_10, STAT, CHAN1, 1, 1, NOTHING

# supplement information transferred with the meter status
# (HOLD status, RESET status, Local/Remote programming)
JJ_11, INFO, CHAN1, 1, 1, NOTHING

# execution of the command: reset of minimal and maximal values of the meter
JJ_12, RST, CHAN1, 1, 1, NOTHING

# execution of the command: Clear forced-zero setting
JJ_13, FCLR, CHAN1, 1, 1, NOTHING

# execution of the command: Switch mode
JJ_14, MODE, CHAN1, 1, 1, NOTHING

# execution of the command: Remote/Local programming
JJ_14, CTRL, CHAN1, 1, 1, NOTHING

# parameter input range of the K3NX meter
JJ_20, PW.8000.0, CHAN1, 1, 1, NOTHING

# parameter scaling display value 2 of the K3NX meter
JJ_21, PL.C00C.1, CHAN1, 1, 1, NOTHING_LONG

# parameter power supply frequency of the K3NX meter
JJ_22, PW.8824.0, CHAN1, 1, 1, NOTHING

# parameter input mode of the K3NC meter
JJ_30, PW.8000.0, CHAN1, 1, 1, NOTHING

# parameter power failure memory of the K3NC meter
JJ_31, PW.8824.1, CHAN1, 1, 1, NOTHING

# parameter compensation value of the K3NC meter
JJ_32, PW.C82A.0, CHAN1, 1, 1, NOTHING_LONG
```

## Driver Configuration

K3N driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **K3N** section.

**Section name: K3N**  
 **Option name: LOG\_FILE**  
 **Option value: file\_name**  
 Meaning - the item allows to define a file where all diagnostic messages of the K3N driver and information about the contents of telegrams received by the driver will be written. If the item does not define its full path, then the log file is created in the current directory. The log file should be used only while the Asix start-up.  
 Default value - by default, any log file is not created.

**Section name: K3N**  
 **Option name: LOG\_FILE\_SIZE**  
 **Option value: number**  
 Meaning - the item allows to define the log file size in MB.  
 Default value - by default, it is assumed that the log file has a size of 1 MB.

**Section name: K3N**  
 **Option name: LOG\_OF\_TELEGRAMS**  
 **Option value: YES|NO**  
 Meaning - the item allows to write to a log file (declared by using the item LOG\_FILE) the contents of telegrams sent within the communication with the meter. Writing the contents of telegrams to the log file should be used only while the asix start-up.  
 Default value - by default, the telegrams are not written.

**Section name: K3N**  
 **Option name: RECV\_TIMEOUT**  
 **Option value: id,number**  
 Meaning - the item allows to determine a maximal waiting time for arriving the first character of the answer from a given meter. After overflow of this time it is assumed that the considered meter is turned off and the transmission session ends with an error.  
 Default value - by default, it is assumed that the maximal waiting time for the first character of the answer is equal to 1000 milliseconds.  
 Parameter:  
   *id* - meter no. in the K3N network,  
   *number* - time in milliseconds (from 100 up to 5000).

**Section name: K3N**  
 **Option name: CHAR\_TIMEOUT**  
 **Option value: id,number**  
 Meaning - the item allows to determine a maximal time between successive characters of the answer from a given meter. After having exceeded this time it is assumed that the considered meter does not work correctly and the transmission session ends with an error.  
 Default value - by default, it is assumed that the maximal time between successive characters of the answer is equal 50 milliseconds.  
 Parameter:  
   *id* - number of the meter in the network,  
   *number* - time in milliseconds (from 10 up to 300).

- Section name: K3N**
- Option name: NUMBER\_OF\_REPETITIONS**
- Option value: number**

Meaning - the item allows to determine a number of repetitions in case of occurring the transmission error.

Default value - by default, the item receives a value of 0 (no repetition).

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value

- by default, no time for data validity is set.

## 1.45 K-BUS - Driver of Protocol for VIESSMANN Dekamatic Boiler PLCs

### Driver Use

The K-BUS driver is used for data exchange between VIESSMANN Dekamatic boiler controllers connected to the Dekatel-G (Vitocom 200) concentrator and an Asix system computer. For communication with Asix an interface of the RS-232C standard is used.

Parameterization of K-BUS driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the K-BUS driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: K-BUS

**K-BUS** tab:

*Channel parameters*:

*id, port [, alarm\_offset]*

where:

*id* - identifier of the regulator,

*port* - port name: COM1, COM2 etc.,

optional parameters:

*alarm\_offset* - offset added to the number of alarm sent from the regulator. By default, the value of offset equals 0.

The list of identifiers assigned to individual regulators is given below:

- |    |   |   |
|----|---|---|
| 1  | - | Dekamatic-D1/Dekamatic-DE               |
| 2  | - | Dekamatic-D2 (Kesselregelung 2. Kessel) |
| 3  | - | Dekamatic-D2 (Kesselregelung 3. Kessel) |
| 4  | - | Dekamatic-HK (1. und 2. Heizkreis)      |
| 5  | - | Dekamatic-HK (3. und 4. Heizkreis)      |
| 6  | - | Dekamatic-HK (5. und 6. Heizkreis)      |
| 7  | - | Dekamatic-HK (7. und 8. Heizkreis)      |
| 8  | - | Dekamatic-HK (9. und 10. Heizkreis)     |
| 9  | - | Dekamatic-HK (11. und 12. Heizkreis)    |
| 10 | - | Dekamatic-HK (13. und 14. Heizkreis)    |
| 11 | - | Dekamatic-HK (15. und 16. Heizkreis)    |

Transmission parameters are constant:

- 1200 Bd,
- 8 bits of a character,
- parity check: even,
- one stop bit.

**EXAMPLE**

A declaration of the logical channel named CHAN1, which works according to the K-BUS protocol and exchanges the data with the regulator Dekamatic-DE (id 1) through the COM2 port is as follows:

*Channel / Name:* CHAN1  
*Driver:* K-BUS  
*Channel parameters:* 1, COM2

The K-BUS driver is loaded as a DLL automatically.

## Addressing the Process Variables

The syntax of symbolic address which is used for variables belonging to the K-BUS driver channel is as follows:

$V<index>$

where:

$V$  - fixed symbol of the variable type,  
 $index$  - variable index, compatible to the table of addresses of variables for the controller under consideration (given in HEX form).

Raw values of the variables are transferred by the driver as numbers of WORD type.

**EXAMPLES**

Examples of declaration of variables:

```
# max boiler temperature (id 12 HEX)
JJ_1, V12, CHAN1, 1, 1, NOTHING
```

```
# external temperature (id 25 HEX )
JJ_2, V25, CHAN1, 1, 1, NOTHING
```

## Driver Configuration

K-BUS driver parameters are declared in the Miscellaneous module, the *Directly entered options* tab.

The driver is parameterized with use of **K-BUS** section.

- Section name:** K-BUS
- Option name:** LOG\_FILE
- Option value:** file\_name

Meaning - the item allows to define a file where all diagnostic messages of the K-BUS driver and information about the content of telegrams received by the driver are written. If the item does not define its full path, then the log file is created in the current directory. The log file should be used only while the Asix start-up.

Default value - by default, the log file is not created.

- Section name: K-BUS**
- Option name: LOG\_OF\_TELEGRAMS**
- Option value: YES|NO**

Meaning - the item allows to write to the log file (declared by using the item LOG\_FILE) the contents of telegrams sent within the communication with regulators. Writing the contents of telegrams to the log file should be used only while the Asix start-up.

Default value - by default, the telegrams are not written.

- Section name: K-BUS**
- Option name: LOG\_FILE\_SIZE**
- Option value: number**

Meaning - the item allows to define the log file size in MB.

Default value - by default, it is assumed that the log file has a size of 1 MB.

- Section name: K-BUS**
- Option name: RECV\_TIMEOUT**
- Option value: id,number**

Meaning - the item allows to determine a maximal waiting time for arriving the first character of the answer from a given regulator. After this time is over it is assumed that the controller under consideration is turned off and the transmission session ends with an error.

Default value - by default, it is assumed that the maximal waiting for the first character of the answer is equal to 1000 milliseconds.

Parameter:

*id* - number of the regulator,  
*number* - time in milliseconds (from 100 up to 5000).

- Section name: K-BUS**
- Option name: CHAR\_TIMEOUT**
- Option value: id,number**

Meaning - the item allows to determine a maximal time between successive characters of the answer from a given regulator. After having exceeded this time it is assumed that the regulator under consideration does not work correctly and the transmission session ends with an error.

Default value - by default, it is assumed that the maximal time between successive characters of the answer is equal to 50 milliseconds.

Parameter:

*id* - number of the regulator,  
*number* - time in milliseconds (from 10 up to 300).

### **Delay After Having Mapped the Data in Concentrator Dekatel-G (Vitocom 200)**

Dekatel-G (Vitocom 200) concentrators allow simultaneous reading 8 variables maximally. The work mode with the concentrator consists in successive execution of the following functions for the successive group of variables:

- transfer of a list of maximally 8 variables to the concentrator (so called concentrator mapping),
- wait for updating the variables in the concentrator after mapping,
- reading the variables from the concentrator.

- Section name: K-BUS**
- Option name: MAPPING\_DELAY**
- Option value: number**

Meaning - the item allows to determine the time, which must elapse between mapping and the first reading of data from the concentrator so that the read data might be assumed as reliable. In case of too short delay the risk of reading the values of variables which were registered in the concentrator before mapping exists.

Default value - by default, the parameter assumes a value of 35 seconds.

Parameter:  
     *number* - time in seconds.

**NOTE** The protocol specification does not give the formula to calculate the delay after concentrator mapping, therefore this parameter must be determined experimentally by the user.

- Section name: K-BUS**
- Option name: GLOBAL\_ALARMS**
- Option value: YES|NO**

Meaning - the item controls the way of transferring alarms read from regulators to the alarms system of Asix start-up.

Default value - by default, the alarms are transferred to the alarms system as global alarms (transferred to the alarms system by means of the function `AsixAddAlarmGlobalMili()`). Setting the value of the item GLOBAL\_ALARMS on NO causes that the alarms are transferred to the alarms system by means of the function `AsixAddAlarmMili()`.

- Section name: K-BUS**
- Option name: SIGNED\_VARIABLE**
- Option value: YES/NO**

Meaning - the item determines the way of interpretation of the BYTE variable. Setting the value on NO makes the UNSIGNED CHAR interpretation be given to the variable.

Default value - by default, and in the case of setting the variable on YES the variable of the BYTE type assumes the SIGNED CHAR interpretation - it allows transferring the negatives.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

- Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:  
     YES / NO

## Communication Drivers

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).  
The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).  
The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.46 Lb480 - Driver for Communication with Concentrator LB-480 by LAB\_EL Elektronika Laboratoryjna through Ethernet

### Driver Use

The driver Lb480 is used to read the current and historical data from the concentrator LB-480 manufactured by LAB\_EL Elektronika Laboratoryjna (<http://www.label.pl/>).

Sample applications of concentrator LB-480:

- monitoring of server room: temperature measurement at various points, air humidity measurement, flooding sensor, door open sensor with detection of sensor line tamper;
- mini meteo station: measurements of temperature and humidity, atmospheric pressure, wind speed and direction, sunshine;
- local temperature measurement: ability to measure temperature in 8 points using simple and low-cost direct thermistor probes.

Data exchange between Asix and LB-480 is carried out by Ethernet with the use of Modbus/TCP protocol in UDP mode.

### Declaration with Transmission Channel

A separate transmission channel should be defined for each hub. Declaration of the transmission channel requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name* – logical name of the transmission channel  
*Driver* – Lb480

**Lb480 / Channel Parameters:**

*IP address of the device* – IP address of the hub,  
*Device number in the MODBUS network* – hub number in the MODBUS network,  
*Port* – TCP number in the hub; by default: 502.

#### Example

*Name:* CHANNEL

*Driver:* Lb480

*IP address of the device:* 10.10.105.21

*Device number in the MODBUS network:* 10

## Addressing the Process Variables

The syntax of symbolic address is as follows:

**ZM[nrWe].[nrZm]**

where:

**nrWe** - input number (1 – 8)

**nrZm** - variable name (1 – 8) from input defined by the parameter **nrWe**

The driver accepts only the conversion functions for which the raw value of process variable is of DWORD, LONG or FLOAT type.

If for a given conversion function the raw type of process variable is DWORD or LONG, the driver reads values of process variables from the registers 30065 – 30192.

If for a given conversion function the raw type of process variable is FLOAT, the driver reads values of process variables from the registers 30193 – 30320.

Examples:

; X\_ZM1\_1 – variable to show the state of input 1 of variable 1 as a fixed-point number

X\_ZM1\_1, ,ZM1.1, KANALX, 1, 1, NOTHING\_DW

; X\_ZM3\_2 – variable to show the state of input 3 of variable 2 as a floating-point number

X\_ZM3\_2, ,ZM3.2, KANALX, 1, 1, NOTJING\_FP

; X\_ZM2\_2 – variable to show the state 2 of variable 2 as a floating-point number

; using the scaling

X\_ZM2\_2, ,ZM2.2, KANALX, 1, 1, MULTIPLIER\_FP, 2.0

## Historical Data Readout

The driver allows access to the historical data collected in the hub memory. Historical data readout is performed by the special function of Modbus protocol, developed by hub manufacturer.

Access to historical data is performed using the driver options (options are described in the section *Driver Parameterization*).

### Notice:

If the firmware allowing to access historical data is not installed in the hub, historical data readout by the option *History data* should be turned off.

## Driver Parameterization

The Lb480 driver parameters are declared in the *Current data* module in the **Lb480/Driver Parameters** tab... of the channel operating with the Lb480 driver protocol:

- log file creation,

- log file size,
- log of telegrams,
- response timeout,
- IP address of the computer card by which the hub is serviced,

And specific items for service historical data collected in the hub memory:

- log file creation for operations of historical data servicing,
- log of historical data transferred to Aspad (archiving module),
- exclusion of historical data readout.

**Response timeout**

Purpose: this option specifies the maximum time throughout which the driver waits for a response from the hub.

The default value: 500 milliseconds.

Option value:

*number* - time out in milliseconds.

**IP address of the computer**

Meaning: - specifies the IP address of the computer network card by which all the driver channels will communicate with the hubs.

Option value:

*address\_IP* - IPv4 address notation.

Default value: - If this option is not defined, the system network card will be used.

**History data**

Meaning - allows to turn off the readout of historical data in all the channels.

Default value - YES.

Parameter:

YES/NO

**Log file**

Meaning - text log file to which driver status messages are stored; it is used for diagnostic purposes.

Default value - file is not created.

Parameter:

*File name*

**Log File Size**

Meaning - option is used to determine the log file size defined by the *Log file*.  
Default value - 10 MB.  
Parameter:  
    *number* - log file size in MB.

**Log of telegrams**

Meaning - option allows writing the contents of messages transmitted between the driver and the hubs to the log file (declared using the *Log File* option). The option should only be used during the Asix system start-up.  
Parameters:  
    YES/NO  
Default value - option disabled.

**History log file**

Meaning - text log file to which messages about access to historical data in all channels providing historical data; option for diagnostic purposes.  
Default value - log file is not created.  
Parameter:  
    *file\_name*

**Aspad historical buffer log**

Meaning - determines whether values and time stamps of samples sent to Aspad should be written to.  
Default value - NO.  
Parameter:  
    YES/NO

**Exemplary driver configuration:**

*Log file:* d:\tmp\CtLb480\label.log  
*Log file size:* 20  
*Log of telegrams:* YES  
*IP address of the computer:* 10.10.110.24

## Channel Parameters

The parameters of a channel are declared in the **Lb480/Channel parameters** i **Channel parameters – diagnostics** tab:

- log file,
- log file size,
- log of telegrams,
- response timeout,
- IP address of the computer,
- disabling readout of history data.

### **Log file**

Meaning

- text log file to which messages with telegrams sent and received for a given channel will be written to. The option is used for diagnostic purpose.

Default value

- file is not created.

Parameter:

*File name*

### **Log file size**

Meaning

- allows to define the size of the log file defined with the use of *Log file* option.

Default value

- by default, log file size is defined in driver parameters.

Parameter:

*number*

- log file size in MB.

### **Log of telegrams**

Meaning

- option allows writing the contents of telegrams communicated in a given channel to the log file. The item value overrides the setting specified by the item of the same name present in the driver section. The item should only be used during the Asix system start-up.

Parameters:

*YES/NO*

Default value

- file is not created.

### **Response timeout**

Meaning

- option specifies the maximum time throughout which a given channel waits for a response from the hub. The item value overrides the setting specified by the item of the same name present in the driver section.

Default value - 500 ms.  
Parameters:  
    *number* - timeout in milliseconds.

**IP address of the computer**

*Meaning:* - option is used to specify the IP address of the computer network card, with the use of which the channel communicates with the hub. The option value overrides the setting specified by the item of the same name present in the driver section.

*Option value:*

*IP\_address* - network card address in IPv4 notation.

*Default value:* - if the option is not declared, the default IP address is taken from the driver configuration.

**History data**

*Meaning* - option allows you to disable historical data readout in a given channel. The option overrides the same option declared in driver configuration.

Default value - YES.

Parameter:  
    YES/NO

**Exemplary channel declaration:**

*Name (channel):* KANAL1

*Log file:* d:\logi\kanal1.log

*Log file size:* 20

*Log of telegrams:* YES

*IP address of the computer:* 10.10.115.25

*Response timeout:* 750

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

*Purpose* - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

*Option value:*

    YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can

consist of maximum 15 characters; the names are separated by commas).  
 The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorized to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.47 LG - Driver of Dedicated Protocol of LG Master-K and Glofa GM PLCs

### Driver Use

The LG driver is designed to exchange data between the Asix system and LG Industrial Systems Master - K and Glofa GM PLCs with use of an RS232 port. The driver enables access to LG PLC data addressed directly by passing the address of the variable within the device. The driver allows simultaneous handling of many LG PLCs.

Parameterization of CtLG driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the LG driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: CtLG

**CtLG** tab:

*Channel parameters*:

*Port*=*port\_number*; *Speed*=*transmission\_speed*;  
*StopBits*=*number\_of\_stop\_bits*; *ParityBit*=*control\_of\_frame\_parity*;  
*DataBits*=*number\_of\_data\_bits*; *NrOfBlocks*=*number\_of\_blocks*;  
*TimeSynchr*=*address[:period]*;

where:

- Port* - number of the COM serial port;
- Speed* - speed of transmission between computer and device; the following speeds are acceptable: 1200, 2400, 4800, 9600, 19200, 38400, 56000, 57600, 115200, 128000; every speed is given in bits per second, i.e. bauds; a default value is 1200 Bd;
- StopBits* - number of stop bits: 1 or 2; for the GlofaGM6 PLC this parameter is built into the controller on a permanent basis and amounts to 1; a default value of the parameter is 1;
- ParityBit* - defines the method of frame parity control; available options: no, parity\_control, odd\_parity\_contol; as substitutes for these options you may use, respectively: 0, 1, 2; for GlofaGM6 PLC this parameter is built into the controller on a permanent basis and amounts to 0, i.e. no; a default value of the parameter is parity\_control;
- DataBits* - number of data bits per frame: 7 or 8; for the GlofaGM6 PLC this parameter is built into the controller on a permanent basis and amounts to 8; a default value of the parameter is 8;
- NrOfBlocks* - max number of blocks specifying variables in a single read operation; a maximum value of the parameter is 16; the parameter has been introduced because GLOFA PLC properly executes queries for max 4 blocks (inconsistently with specification); a default value of the parameter is 16;
- TimeSynchr* - for Glofa PLCs it is a direct address of the table of 9 bytes, in the controller that PC's system time is to be entered into. The table will be filled in with BCD-code numbers as follows:

time[0] = two younger digits of year  
 time[1] = month  
 time[2] = day of month  
 time[3] = hour  
 time[4] = minutes  
 time[5] = seconds  
 time[6] = day of week (Monday - 0, Tuesday - 1,... Sunday - 6)  
 time[7] = two older digits of year  
 time[8] = 1

e.g. 2001 - 03 - 15 18:30:45 Tuesday: time[0] = 01, time[1] = 03, time[2] = 15,  
 time[3] = 18, time[4] = 30, time[5] = 45, time[6] = 03, time[7] = 20,

On Glofa controller's side, you should execute the command RTC\_SET argument. The argument is a symbolic address of the table of 8 bytes. When declaring this variable, in the Memory allocation field select the Assign(AT) option and pass the direct address of the variable in the transmission channel declaration. Upon the system time is rewritten from the PC to the controller, the value 1 is assigned to the ninth element of the table.

In case of the MASTER-K PLC, TimeSynchr is the address of the table of word type consisting of five elements, which is available in the controller N. The table is filled in with BCD-code numbers as follows:

time[0] = older byte = 2 younger digits of year, younger byte = month (1..12)  
 time[1] = older byte = day of month (1..31), younger byte = hour  
 time[2] = older byte = minutes, younger byte = seconds  
 time[3] = older byte = 2 older digits of year, younger byte = day of week (Sunday - 0, Monday - 1...Saturday - 6)  
 time[4] = 1

With the exception of *time[4]*, these words should be rewritten into a special area of the memory and the appropriate bit should be set.

*period* - default parameter to define the interval in seconds at which time will be rewritten from the PC to the controller. By default, the synchronization time is 60 seconds.

**EXAMPLE**

An example of declaration of channel in which time will be synchronised for the controller numbered 6 (GLOFA type) by the write into the area starting with MB10 (every 25 seconds):

*Channel / Name:* PLC1  
*Driver:* LG  
*Channel parameters:* Port=2; Speed=9600; StopBits=1; ParityBit=odd\_parity\_control;  
 DataBits=8; TimeSynchr=6.MB10:25

## Addressing the Process Variables

### Addressing the Variables in Master - K family

The following direct address is only acceptable:

*ControllerNo.TypeOfDevice.Address*

where:

- Controller no* - number between 0 and 31.
- Type of device* - (see: the table below).

**Table 20. Types of Devices in Master-K Family.**

Type of Device	Range of Device	Read/Write	Bit/Word
P ( Input/Output relay )	P0 ~ P0031 ( 32 words ) P0.0 ~ P31.15 ( 32 × 16 bits )	Read/Write	Both
M ( auxiliary relay )	M0 ~ M191 ( 192 words ) M0.0 ~ M191.15 ( 192 × 16 bits )	Read/Write	Both
K ( keep relay )	K0 ~ K31 ( 32 words ) K0.0 ~ K31.15 ( 32 × 16 bits )	Read/Write	Both
L ( link relay )	L0 ~ L63 ( 64 words ) L0.0 ~ L63.15 ( 64 × 16 bits )	Read/Write	Both
F ( special relay )	F0 ~ F63 ( 64 words ) F0.0 ~ F63.15 ( 64 × 16 bits )	Read	Both
T ( timer contact relay )	T0.0 ~ T0.255 ( 256 bits )	Read/Write	Both
T ( timer elapsed value )	T0 ~ T255 ( 256 words )	Read/Write	Both
C ( counter contact relay )	C0.0 ~ C0.255 ( 256 bits )	Read/Write	Both
C ( counter elapsed value )	C0 ~ C255 ( 256 words )	Read/Write	Both
S ( step controller )	S0 ~ S99 ( 100 sets )	Read/Write	Word only
D ( data register )	D0 ~ D4999 ( 5000 words )	Read/Write	Word only

**NOTE** *T and C devices should not be used for bit addressing because it does not work due to error in the controller's operating system.*

There are two types of variables:

- bit,
- word.

The variable address may contain up to 8 characters (without device type character and '.' character, if any). The address is given in decimal format. When bit is addressed within a word, the bit number (from 0 to 15) is given after a dot. An exception to this rule is Timer and Counter types. As you can see in the above table, bits for these types are addressed from 0 to 255.

For example, O. M5 - word with the address 5  
O. M5.10 - eleventh bit in the word with the address 5

#### **EXAMPLE**

Examples of variable declarations:

JJ\_00, variable of WORD type with address M1, O.M1, PLC1, 1, 1, NIC  
JJ\_01, variable of BIT type with address M5.10, O.M5.10, PLC1, 1, 1, NIC

#### Addressing the Variables in Glofa – GM Family

The direct address has the following format:

*ControllerNo.TypeOfDevice.TypeOfVariable.Address*

where:

*ControllerNo* - defines the controller number and is a number between 0 and 31;

*TypeOfDevice* - defines the device type; the following types are available:

- M (internal memory),
- Q (output),
- I (Input).

The range of addressing for these devices is configurable and depends on the type of device.

All of these devices can be both written to and read from.

*TypeOfVariable* - defines the variable type. The following types are available:

- X - bit,
- B - byte
- W - word,
- D - double word.

For the M device the address is given in decimal format and may contain maximum 13 characters, without the character of the device and variable type, e.g.:

3.MW1 - word with the address 1 from the dictionary no 3

If you want to address a bit in the M device within a byte, word or double word, the bit number (counted from 0) in decimal format should be given after a dot, for example:

4.MW1.14 - fourteenth bit in the word 1 from the dictionary 4

5.MD2.30 - thirteenth bit in the double word 2 from the dictionary 5

Bit can also be addressed directly using the X character, e.g.:

0.MX10 - tenth bit.

In case of addressing Q and I devices, the address is given in decimal format. These are three numbers (*base, slot, number*) separated with the '.' character, e.g.:

2.QX3.1.4 - controller no 2, 3 base, 1 slot, 4<sup>th</sup> bit,

3.IW2.4.1 - controller no 3, 2 base, 4 slot, 1<sup>st</sup> word.

## EXAMPLE

Examples of variable declarations:

JJ\_00, VARIABLE OF WORD TYPE WITH ADDRESS MW1, 0.MW1, PLC1, 1, 1, NIC

JJ\_01, variable of BIT type with address QX3.1.4, 0.QX3.1.4, PLC1, 1, 1, NIC

## Time Marker

Values of variables read from LG are assigned a PC's time stamp.

## Driver Parameterization

LG driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of CtLG section.

**Section name: CtLG**

**Option name: LOG\_FILE**

**Option value: log\_file\_name**

Meaning - for diagnostic purposes the text-type log file into which messages about driver operation status are written is used.

Default value - by default, the log file is not created.

Defining - manual.

**Section name: CtLG**

**Option name: LOG\_FILE\_SIZE**

**Option value: number**

Meaning - this item is used to define the size of the log file defined with use of the LOG\_FILE item.

Default value - by default, the log file size is 1 MB.

Defining - manual.

**Section name: CtLG**

**Option name: LOG\_OF\_TELEGRAMS**

**Option value: yes | No**

Meaning - this item allows contents of telegrams transferred between the driver and controllers to be written into the log file (declared with use of the LOG\_FILE item). The referred item should only be used in the Asix system start-up stage.

Default value - by default, value of this item is set to NO.

Defining - manual.

**EXAMPLE**

An example of the driver section:

*Section name: CtLG*

*Option name: LOG\_FILE*

*Option value: d:\tmp\ctLG\LG.log*

*Section name: CtLG*

*Option name: LOG\_FILE\_SIZE*

*Option value: 3*

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
 YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.48 Logo - DRIVER of Logo OBA5 controllers from SIEMENS

### Driver Use

Logo protocol driver is used to exchange data between Asix system and Logo OBA5 controller from SIEMENS with the use of the programmer link of the mentioned controller. The transmission is executed with the use of serial links by means of standard computer serial ports in RS 232 standard.

Parameterization of Logo driver is performed with the use of Architect application.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the Logo driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: Logo

**CtLg** tab:

*Channel parameters*:  
Port=*number*; Type=*number* [; Timeout=*number*]

where:

*Port* - serial port number,  
*Type* - Logo controller type ID. At the moment, only Logo OBA5 is supported (ID is 1),  
*Timeout* - max. waiting time for the first character of response (in milliseconds). By default, it is 1000 milliseconds.

Transmission parameters are constant and have the following values:

- transmission speed 9600 Bd,
- 8 character bits,
- parity control,
- 1 stop bit.

#### EXAMPLE

Example of the declaration of the KLOGO transmission channel used to communicate with the Logo OBA5 controller through the COM2 serial port:

*Name*: KLOGO  
*Driver*: CtLogo  
*Driver parameters* Port=2; Type=1

## Variable Declaration

The symbolic address of the process variable has the following syntax:

< type> < index>

where:

*type* - variable type,  
*index* - index within type.

The set of types and index range are specific for the determined type of Logo controller.

**NOTE** *Variable values can be only read - the protocol does not allow for the performance of controls.*

Variable types markings (in the brackets, the raw variable value type is given):

Types of variables:

<b>DI</b>	- digital input	(WORD),
<b>DQ</b>	- digital output	(WORD),
<b>DM</b>	- digital flag	(WORD),
<b>AI</b>	- analog input	(WORD),
<b>AQ</b>	- analog output	(WORD),
<b>AM</b>	- analog flag	(WORD).

### EXAMPLE

Examples of variables declaration:

DI1, digital input 1,	DI1, KLOGO,1,1,NIC
DQ3, digital output 3,	DQ3, KLOGO,1,1,NIC
DM13,digital flag 13,	DM13, KLOGO,1,1,NIC
AI2, analog input 2,	AI2, KLOGO,1,1,NIC
AQ1, analog output 1,	AQ1, KLOGO,1,1,NIC
AM2, analog flag 2,	AM2, KLOGO,1,1,NIC

## Driver Parameterization

Logo driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **CTLOGO** section.

- Section name: CTLOGO**
- Option name: LOG\_FILE**
- Option value: log\_filename**

Meaning: the diagnostic purposes are fulfilled by a text log file, where the driver operation status messages are entered.

Default value: by default, the log file is not created.

- Section name: CTLOGO**
- Option name: LOG\_FILE\_SIZE**
- Option value: number**

Meaning: the option is used to determine the log file size.

Option value: *number* - file size in MB.

Default value: default log file size is 10 MB.

- Section name:** CTLOGO
- Option name:** LOG\_OF\_TELEGRAMS
- Option value:** YES | NO

Meaning: option allows the contents of telegrams sent between the driver and controllers to the log file to be saved (declared with use of *LOG\_FILE* option). The option allows the log file size to be determined. The subject option should be used only during the start-up of Asix system.

Default value: by default, the option value is set to NO.

## EXAMPLE

Example of driver section:

*Section name:* CTLOGO  
*Option name:* LOG\_FILE  
*Option value:* d:\tmp\CtLogo\logo.log

*Section name:* CTLOGO  
*Option name:* LOG\_FILE\_SIZE  
*Option value:* 20

*Section name:* CTLOGO  
*Option name:* LOG\_OF\_TELEGRAMS  
*Option value:* YES

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose

- with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).  
 The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
 YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.  
 The default value - by default, no time for data validity is set.

## 1.49 LUMBUS - Driver for LUMEL Meters

### Driver Use

The LUMBUS driver is used for data exchange between RG72 controllers manufactured by Lubuskie Zakłady Aparatów Elektrycznych (Electrical Measuring Instrument Works) "LUMEL" in Zielona Góra and an Asix system computer. The communication is executed by means of serial interfaces in the RS485 standard.

Parameterization of LUMBUS driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the LUMBUS driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

Name: logical name of the transmission channel  
Driver: LUMBUS

**LUMBUS** tab:

*Channel parameters:*

*number, port, baud*

where:

- |               |   |
|---------------|---|
| <i>number</i> | - controller no. in the network,                  |
| <i>port</i>   | - port name: COM1, COM2 etc.,                     |
| <i>baud</i>   | - transmission speed in the range 1200 - 9600 Bd. |

By default it is assumed:

- transmission speed 9600 Bd,
- number of character bits - 8,
- without parity check (PARITY NONE),
- number of stop bits - 1.

#### EXAMPLE

The declaration of the logical channel named CHANNEL, which works according to the LUMBUS driver protocol and exchanges data with the RG72 regulator numbered 1 through the COM2 port with a speed of 4800 Bd, is as follows:

*Channel / Name:* CHANNEL  
*Driver:* LUMBUS  
*Channel parameters:* 1, COM2, 4800

The LUMBUS driver is loaded as a DLL automatically.

## Addressing the Process Variables

The syntax of symbolic address which is used for variables belonging to the LUMBUS driver channel is as follows:

`<type><index>[.subindex]`

where:

- |                 |   |
|-----------------|---|
| <i>type</i>     | - variable type; allowed types:<br>P     - single measurement,<br>PT    - table of measurements,<br>WT    - array of free days,<br>DT    - array of holiday's dates;  |
| <i>index</i>    | - according to the specification given in point 3 of the " <i>Serial interface RS-485 in RG7-07/2 controller</i> " documentation;<br>for single measures <i>index</i> takes the index value assigned to the measurement in the table;<br>for the values transferred in the form of arrays <i>index</i> takes the index value assigned to the array, and the item of the variable under consideration in the array is specified by <i>subindex</i> ; |
| <i>subindex</i> | - applies to the specification of variables transferred in the form of arrays and determines the variable location in the array; the subindex of the first element in the array takes a value of 0.   |

The raw value of measure is of FLOAT type.

The raw value of free day and holiday date is an ASCII string in dd.mm.yyyy format ended by zero (including 11 character).

### EXAMPLES

Examples of declarations of variables the values of which are transferred individually:

```
X13, hour of turning on night reduction, P13,      CHANNEL, 1, 1, NOTHING_FP
X23, set temperature c.w.u,                P23,      CHANNEL, 1, 1, NOTHING_FP
```

Examples of declarations of variables, the values of which are transferred in form of arrays:

```
X39, set temperature in control room,    PT38.0, CHANNEL, 1, 1, NOTHING_FP
X40, ext. temp (A) - initial point of curve, PT38.1, CHANNEL, 1, 1, NOTHING_FP
X50, max. allowed temperature of return, PT48.1, CHANNEL, 1, 1, NOTHING_FP
X56, dead band of c.h.,                  PT52.3, CHANNEL, 1, 1, NOTHING_FP
```

```
X68, economies - holidays day 1,        WT67.0, CHANNEL, 11, 1, NOTHING_TEXT
X69, economies - holidays day 2,        WT67.1, CHANNEL, 11, 1, NOTHING_TEXT
X70, economies - holidays day 3,        WT67.2, CHANNEL, 11, 1, NOTHING_TEXT
```

```
X119, first period of vacation - from,  DT119.0, CHANNEL, 11, 1, NOTHING_TEXT
X120, first period of vacation - from,  DT119.1, CHANNEL, 11, 1, NOTHING_TEXT
X121, second period of vacation - from, DT121.0, CHANNEL, 11, 1, NOTHING_TEXT
X122, second period of vacation - from, DT121.1, CHANNEL, 11, 1, NOTHING_TEXT
X123, third period of vacation - from,  DT123.0, CHANNEL, 11, 1, NOTHING_TEXT
X124, third period of vacation - from,  DT123.1, CHANNEL, 11, 1, NOTHING_TEXT
```

## Driver Configuration

LUMBUS driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **LUMBUS** section.

**Section name: LUMBUS**

**Option name: LOG\_FILE**

**Option value: file\_name**

Meaning - the item allows to define a file where all diagnostic messages of the LUMBUS driver and the information about the contents of telegrams received by the driver are written. If the item does not define the full path, then the LOG\_FILE is created in the current directory. The log file should be used only while the Asix start-up.

Default value - by default, the log file is not created.

### EXAMPLE

*Section name:* LUMBUS

*Option name:* LOG\_FILE

*Option value:* D:\asix\LUMBUS.LOG

**Section name: LUMBUS**

**Option name: LOG\_OF\_TELEGRAMS**

**Option value: YES|NO**

Meaning - the item allows to write to the log file (declared by using the item LOG\_FILE) the contents of telegrams sent within the communication with the RG72 regulator. Writing the contents of telegrams to the log file should be used only while the Asix start-up.

Default value - by default, telegrams are not written.

**Section name: LUMBUS**

**Option name: NUMBER\_OF\_REPETITIONS**

**Option value: number**

Meaning - the item allows to determine a number of repetitions in case of an appearance of transmission error.

Default value - by default, the item assumes a value of 0 (no repetitions).

Table. List of Symbolic Addresses.

Symb. Address	Index	Designation of Measurements From RG72	Conversion Type	Allowed Operation
P9	9		Word->Float	R
P10	10	f.active	Byte->Float	RW
P11	11	Tpwr	Word->Float	RW
P12	12	DeltaT	Float->Float	RW
P13	13	night hours	Char->Float	RW
P14	14	day hours	Char->Float	RW

P15	15	free days	Byte->Float	RW
P16	16	temp. of summer	Float->Float	RW
P17	17	l. Days	Char->Float	RW
P18	18	hours Pom	Char->Float	RW
P19	19	Pump	Byte->Float	RW
P20	20	cw_oszcz	Byte->Float	RW
P21	21	Tzew	Byte->Float	RW
P22	22	St.pompa	Byte->Float	RW
P23	23	T.zad.cw	Float->Float	RW
P24	24	Priority	Byte->Float	RW
P25	25	t.prio	Byte->Float	RW
P26	26	t_pwrcwu	Byte->Float	RW
P27	27	t_progwu	Byte->Float	RW
P28	28	Disinfection	Byte->Float	RW
P29	29	clock mode	Byte->Float	RW
PT31.0	31	Lkan	Char->Float	R
PT31.1	31	selection of curve, Sommer_is	Char->Float	R
P36	36	lock_full	Bit->Float	W
P37	37	lock_part	Bit->Float	W
		co &endash; heating function		
PT38.0	39	T.zad.pk	Float->Float	RW
PT38.1	40	T.zew(A)	Float->Float	RW
PT38.2	41	T.co(A)	Float->Float	RW
PT38.3	42	tg.alfa	Float->Float	RW
PT38.4	43	T.zew(B)	Float->Float	RW
PT38.5	44	tg.beta	Float->Float	RW
PT38.6	45	T.co_max	Float->Float	RW
PT38.7	46	delta_co	Float->Float	RW
P47	47	T.freeze	Float->Float	RW
		co (centr. heat.) - return curve		

Table. List of Symbolic Addresses (continuation).

Symb. Address	Index	Designation of Measurements from RG72	Conversion Type	Allowed Operation
PT48.0	49	T.pwr_min	Float->Float	RW
PT48.1	50	T.pwr_max	Float->Float	RW
PT48.2	51	tg (pwrt)	Float->Float	RW
		Pid		
PT52.0	53	xp co	Int->Float	RW
PT52.1	54	ti co	Int->Float	RW
PT52.2	55	td co	Int->Float	RW
PT52.3	56	2N co	Int->Float	RW
PT52.4	57	H co	Int->Float	RW

Communication Drivers

PT52.5	58	to co	Int->Float	RW
PT52.6	59	tp co	Int->Float	RW
PT52.7	60	xp cw	Int->Float	RW
PT52.8	61	ti cw	Int->Float	RW
PT52.9	62	td cw	Int->Float	RW
PT52.10	63	2N cw	Int->Float	RW
PT52.11	64	H cw	Int->Float	RW
PT52.12	65	to cw	Int->Float	RW
PT52.13	66	tp cw	Int->Float	RW
		Holidays and free days		
WT67.0	68	holidays/free no. 1	Word->ASCII(11)	RW
WT67.1	69	holidays/free no. 2	Word->ASCII(11)	RW
WT67.2	70	holidays/free no. 3	Word->ASCII(11)	RW
WT67.3	71	holidays/free no. 4	Word->ASCII(11)	RW
WT67.4	72	holidays/free no. 5	Word->ASCII(11)	RW
WT67.5	73	holidays/free no. 6	Word->ASCII(11)	RW
WT67.6	74	holidays/free no. 7	Word->ASCII(11)	RW
WT67.7	75	holidays/free no. 8	Word->ASCII(11)	RW
WT67.8	76	holidays/free no. 9	Word->ASCII(11)	RW
WT67.9	77	holidays/free no. 10	Word->ASCII(11)	RW
WT67.10	78	holidays/free no. 11	Word->ASCII(11)	RW
WT67.11	79	holidays/free no. 12	Word->ASCII(11)	RW
WT67.12	80	holidays/free no. 13	Word->ASCII(11)	RW
WT67.13	81	holidays/free no. 14	Word->ASCII(11)	RW
WT67.14	82	holidays/free no. 15	Word->ASCII(11)	RW
WT67.15	83	holidays/free no. 16	Word->ASCII(11)	RW
WT67.16	84	holidays/free no. 17	Word->ASCII(11)	RW
WT67.17	85	holidays/free no. 18	Word->ASCII(11)	RW
WT67.18	86	holidays/free no. 19	Word->ASCII(11)	RW
WT67.19	87	holidays/free no. 20	Word->ASCII(11)	RW

Table. List of Symbolic Addresses (continuation).

Symb. Address	Index	Designation of Measurements from RG72	Conversion Type	Allowed Operation
WT67.20	88	holidays/free no. 21	Word->ASCII(11)	RW
WT67.21	89	holidays/free no. 22	Word->ASCII(11)	RW
WT67.22	90	holidays/free no. 23	Word->ASCII(11)	RW
WT67.23	91	holidays/free no. 24	Word->ASCII(11)	RW
WT67.24	92	holidays/free no. 25	Word->ASCII(11)	RW
WT67.25	93	holidays/free no. 26	Word->ASCII(11)	RW
WT67.26	94	holidays/free no. 27	Word->ASCII(11)	RW
WT67.27	95	holidays/free no. 28	Word->ASCII(11)	RW
WT67.28	96	holidays/free no. 29	Word->ASCII(11)	RW
WT67.29	97	holidays/free no. 30	Word->ASCII(11)	RW
WT67.30	98	holidays/free no. 31	Word->ASCII(11)	RW
WT67.31	99	holidays/free no. 32	Word->ASCII(11)	RW

WT67.32	100	holidays/free no. 33	Word->ASCII(11)	RW
WT67.33	101	holidays/free no. 34	Word->ASCII(11)	RW
WT67.34	102	holidays/free no. 35	Word->ASCII(11)	RW
WT67.35	103	holidays/free no. 36	Word->ASCII(11)	RW
WT67.36	104	holidays/free no. 37	Word->ASCII(11)	RW
WT67.37	105	holidays/free no. 38	Word->ASCII(11)	RW
WT67.38	106	holidays/free no. 39	Word->ASCII(11)	RW
WT67.39	107	holidays/free no. 40	Word->ASCII(11)	RW
WT67.40	108	holidays/free no. 41	Word->ASCII(11)	RW
WT67.41	109	holidays/free no. 42	Word->ASCII(11)	RW
WT67.42	110	holidays/free no. 43	Word->ASCII(11)	RW
WT67.43	111	holidays/free no. 44	Word->ASCII(11)	RW
WT67.44	112	holidays/free no. 45	Word->ASCII(11)	RW
WT67.45	113	holidays/free no. 46	Word->ASCII(11)	RW
WT67.46	114	holidays/free no. 47	Word->ASCII(11)	RW
WT67.47	115	holidays/free no. 48	Word->ASCII(11)	RW
WT67.48	116	holidays/free no. 49	Word->ASCII(11)	RW
WT67.49	117	holidays/free no. 50	Word->ASCII(11)	RW
		Holidays and free days		
DT119.0	119	first period of vacation (from)	Int->ASCII(11)	RW
DT119.1	120	first period of vacation (from)	Int->ASCII(11)	RW
DT121.0	121	second period of vacation (from)	Int->ASCII(11)	RW
DT121.1	122	second period of vacation (from)	Int->ASCII(11)	RW
DT123.0	123	third period of vacation (from)	Int->ASCII(11)	RW
DT123.1	124	third period of vacation (from)	Int->ASCII(11)	RW
DT125.0	125	forth period of vacation (from)	Int->ASCII(11)	RW
DT125.1	126	forth period of vacation (from)	Int->ASCII(11)	RW
DT127.0	127	fifth period of vacation (from)	Int->ASCII(11)	RW
DT127.1	128	fifth period of vacation (from)	Int->ASCII(11)	RW
		presence or lack of sensors		

Table. List of Symbolic Addresses (continuation).

Symb. Address	Index	Designation of Measurements from RG72	Conversion Type	Allowed Operation
PT129.0	130	sensor 1	Byte->Float	RW
PT129.1	131	sensor 2	Byte->Float	RW
PT129.2	132	sensor 3	Byte->Float	RW
PT129.3	133	sensor 4	Byte->Float	RW
PT129.4	134	sensor 5	Byte->Float	RW
PT129.5	135	sensor 6	Byte->Float	RW
		temperature differences for sensors		
PT136.0	137	sensor 1	Byte->Float	RW
PT136.1	138	sensor 2	Byte->Float	RW
PT136.2	139	sensor 3	Byte->Float	RW
PT136.3	140	sensor 4	Byte->Float	RW
PT136.4	141	sensor 5	Byte->Float	RW
PT136.5	142	sensor 6	Byte->Float	RW
		time of full valve opening		

Communication Drivers

PT143.0	144		Byte->Float	RW
PT143.1	145		Byte->Float	RW
		start/stop		
PT146.0	147		Byte->Float	RW
PT146.1	148		Byte->Float	RW
		safety codes		
PT149.0	150		Word->Float	RW
PT149.1	151		Word->Float	RW
PT149.2	152		Word->Float	RW
		current time		
PT153.0	154	Year	Byte->Float	RW
PT153.1	155	Month	Byte->Float	RW
PT153.2	156	Day	Byte->Float	RW
PT153.3	157	Hour	Byte->Float	RW
PT153.4	158	Minute	Byte->Float	RW
		sensor error		
PT173.0	174	error of sensor 1	Char->Float	R
PT173.1	175	error of sensor 2	Char->Float	R
PT173.2	176	error of sensor 3	Char->Float	R
PT173.3	177	error of sensor 4	Char->Float	R
PT173.4	178	error of sensor 5	Char->Float	R
PT173.5	179	error of sensor 6	Char->Float	R
		control signal		
PT180.0	181		Float->Float	R
PT180.1	182		Float->Float	R

Table. List of Symbolic Addresses (continuation).

Symb. Address	Index	Designation of Measurements from RG72	Conversion Type	Allowed Operation
		measured temperatures		
PT183.0	184	temperature 1	Float->Float	R
PT183.1	185	temperature 2	Float->Float	R
PT183.2	186	temperature 3	Float->Float	R
PT183.3	187	temperature 4	Float->Float	R
PT183.4	188	temperature 5	Float->Float	R
PT183.5	189	temperature 6	Float->Float	R
		alarm data		
PT240.0	241	alarm 1	Float->Float	R
PT240.1	242	alarm 2	Float->Float	R
PT240.2	243	alarm 3	Float->Float	R
		remote control of valve of centr. heat. (co) and heat water (cw)		
PT247.0	248		Char->Float	RW
PT247.1	249		Char->Float	RW
		remote control of pump of co and cw		
PT250.0	251		Char->Float	RW

PT250.1	252		Char->Float	RW
		switching on/off the pump of co and cw		
PT253.0	254		Char->Float	W
PT253.1	255		Char->Float	W
		opening the valve of co and cw		
PT256.0	257		Char->Float	W
PT256.1	258		Char->Float	W
		closing the valve of co, cw		
PT259.0	260		Char->Float	W
PT259.1	261		Char->Float	W
		factory settings		
P262	262		Byte->Float	W
P263	263	fact. settings Co	Byte->Float	W
P264	264	fact. settings Cwu	Byte->Float	W
P265	265	fact. settings Others	Byte->Float	W

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

**Purpose** - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

**Option value:**

YES / NO

**computer name** - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

**The default value** - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

**Purpose** - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

**Option value:**

**computer name** - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

**The default value** - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

number

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.50 LZQM - Driver of LZQM Electricity Meters

### Driver Use

The driver LZQM for exchange of data with LZQM electricity meters, produced by POZYTON Electronic Measuring Instruments Facility in Częstochowa. Communication takes place by means of a serial communication according to CLO standard.

Parameterization of the driver is performed by the Architect application.

### Declaration of Transmission Channel

The declaration of transmission channel using the driver LZQM requires the channel with the following parameters to the *Current data* module to be added:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: LZQM

**LZQM** tab:

Channel parameters:

*Port=number* [; *TransmissionSpeed=number*] [; *Period=number*]

where:

*Port* - number of serial port COM.

*TransmissionSpeed* - transmission speed between a computer and a counter.  
Default value: 4800 Bd.

*Period* - time interval (in seconds) between successive readouts of a counter.  
Default values: 10 seconds.

### Declaration of Variables

Variable address is as follows:

"number\_of\_counter/name\_of\_register"

where:

*number\_of\_counter* - number of 8-digit counter (read from a front panel of a counter and possibly completed by leading zeros);

*name\_of\_register* - name of count register in the form of X.X.X read from a reading array of a counter.

### EXAMPLE

Exemplary declaration of variables (logical name of the transmission channel - PTY, number of a counter 121825):

KP\_01, meter reading of energy EP+, "00121825/0.8.0", PTY, 1, 1, nothing\_fp  
KP\_02, meter reading of energy EP-, "00121825/1.8.0", PTY, 1, 1, nothing\_fp

## Declaration of Internal Variables

The data contained in a block of the last accounting period are internal variables of the driver.

*"number\_of\_counter/number\_of\_register.A"*

where:

*number\_of\_counter* - number of 8-digit counter (read from a front panel of a counter and possibly completed by leading zeros);

*number\_of\_register* - name of count register in the form of X.X.X read from a reading array of a counter.

.A - suffix used to distinguish internal variables.

### EXAMPLE

A\_0.6.1, highest power of P+ in zone 1, "00121825//0.6.1.A", PTY, 1, 1, nothing\_fp  
A\_0.8.1, power meter reading EP+ in zone 1, "00121825//0.8.1.A", PTY, 1, 1, nothing\_fp

## Driver Parameters

The LZQM driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **CtLZQM** section.

- Section name: CtLZQM**
- Option name: LOG\_FILE**
- Option value: file\_name**

Meaning: allows to define a file to which all the diagnostic messages of the driver will be written.

Default value: by default, the log file is not created.

**Section name: CtlZQM**

**Option name: LOG\_FILE\_SIZE**

**Option value: number**

Meaning: this item is used to define the size of the log file defined with use of the LOG\_FILE item.

Option value: *number* - log file size in MB.

Default value: by default, the log file size is 1 MB.

**Section name: CtlZQM**

**Option name: LOG\_OF\_TELEGRAMS**

**Option value: YES | NO**

Meaning: this item allows contents of telegrams transferred between driver and controllers to be written into the log file (declared with use of the LOG\_FILE item). The referred item should only be used in the asix system start-up.

Default value: by default, value of this item is set to NO.

Defining: manual.

**Section name: CtlZQM**

**Option name: TRANSMISSION\_DELAY**

**Option value: number**

Meaning: The item defines the time interval between data receiving and sending the next inquiry to the counter. The option can be necessary for reading in registering mode.

Option value: *number* - time in milliseconds.

Default value: 75 ms (according to the manufacturer it can be too few).

**Section name: CtlZQM**

**Option name: NUMBER\_OF\_REPETITIONS**

**Option value: YES | NO**

Meaning: The item specifies the number of retransmissions in the case of communication errors. It applies to register array readout and instantaneous quantity.

Option value: *number* - number of repetitions.

Default value: 0 (no repetitions).

## EXAMPLE

```
[CTLZQM]
LOG_FILE =d:\tmp\test\licznik.log
LOG_FILE_SIZE =30
LOG_OF_TELEGRAMS = YES
```

## Channel Parameters

Parameters declared in the section of the channel (in *Miscellaneous* module > *Directly entered options* tab).

- ☑ **Section name:** <channel\_name>
- ☑ **Option name:** NUMBER\_OF\_REPETITIONS
- ☑ **Option value:** YES | NO

Meaning: The item specifies the number of retransmissions in the case of communication errors. It applies to register array readout and instantaneous quantity.

Option value: *number* - number of repetitions.

Default value: 0 (no repetitions).

## Parameterization of Individual Counters

The driver allows the set of individual parameters concerning service of particular counters to be transferred in separate sections. The name of such section is a composite of the following elements:

*channel\_name:counter\_name*

where:

- channel\_name* - ASMEN's channel name in which the given counter is serviced;
- counter\_name* - number of 8-digit counter (possibly with leading zeros)

### EXAMPLE 1:

ASMEN's channel name    KANAL  
 Counter name            00121825  
 Section name             KANAL:00121825

A parameterization of the counter may be performed by the following items:

- date/time maker,
- log book data.

- ☑ **Section\_name:** *channel\_name:number\_of\_counter*
- ☑ **Option name:** TIME\_MAKER
- ☑ **Option value:** *register\_code [,register\_code]*

Meaning - it allows to define a register (or two registers), that includes the data and time stamp transmitted from the counter. It is assumed that, if one register is declared, it contains data and time in the 'YY-MM-DD hh:mm:ss' format. If the couple of registers is declared, it is assumed that the first register contains data in the 'YY-MM-DD' format, and the second one contains time in the 'hh:mm:ss' format.

Default value - by default, it is assumed that the PC's data and time stamp from the moment of the end of data receiving will be assigned to values transmitted from the counter.

**Section\_name: channel\_name:number\_of\_counter**

**Option name: LOG\_BOOK\_DATA**

**Option value: YES/NO**

Meaning - it allows to declare whether detailed description of parsing of particular event log lines should be entered to the driver log. The log file should be used only while the Asix system start-up.

Default value - by default, event parsing details are not written to the driver log.

### EXAMPLE

There is an exemplary section describing the counter in the channel PTY. The number of a counter is: 00121825.

[PTY:00121825]

TIME\_MAKER = 29., 28.

LOG\_BOOK\_DATA = YES

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

## Communication Drivers

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.51 M200 - Spirax Sarco M200 Flow Computer Driver

### Driver Use

The M200 driver is designed for data exchange between the Asix system and the M210G flow computer from Spirax Sarco. Communication is performed by the use of serial links in standard RS-232.

Parameterization of the M200 driver is performed by the Architect application.

### Declaration of Transmission Channel

The declaration of the transmission channel utilizing the M200 driver requires the channel with the following parameters to the *Current data* module to be added:

#### **Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: M200

#### **CtM200** tab:

Channel parameters:

*Port=number* [*; Baudrate=number*] [*; Parity=number*] [*; WordLen=number*]  
 [*; StopBits=number*] [*; ServiceType=number*] [*; TimeSynchr=number*]

where:

<i>Port</i>	- computer serial port number;
<i>Baudrate</i>	- transmission speed in Bd; by default, 9600 Bd;
<i>Parity</i>	- EVEN, ODD or NONE; by default NONE;
<i>WordLen</i>	- number of character bits (7 or 8); by default 8;
<i>StopBits</i>	- number of stop bits (1 or 2); by default 1;
<i>ServiceType</i>	- RS-232 interface (mode 1) or RS-485 interface with ADAM-4521 modules (mode 2); default mode is mode 1;
<i>TimeSynchr</i>	- time (in minutes) between the subsequent recording of time to the flow computer. It is possible to disable the time synchronization by setting the <i>TimeSynchr</i> parameter to 0. By default, time synchronization is performed every 1 minute.

**NOTE** In the case of the use of addressable ADAM-4521 modules, the following parameters should be set in ADAM-4521 module:

- a/** delimiter - { (factory preset);
- b/** add cr - no (factory preset - yes);
- c/** address in RS-485 network (different address for each module);
- d/** baud rate - compliant with the speed set in the flow computer.

By default, the following transmission parameters are used:

- transmission speed 9600 Bd,
- number of bits in character - 8,
- parity control - NONE,
- number of stop bits - 1,
- communication through RS-232 link.

### EXAMPLE

An example of the declaration of a channel utilizing the M200 driver on the COM2 port in RS-232 mode with default settings and time synchronization every 10 minutes:

*Name:* PLC1

*Driver:* M200

*Driver parameters:* Port=2; TimeSynchr=10

## Addressing Variables

Symbolic address has the following syntax:

*V.address.id*

where:

- |                |   |
|----------------|---|
| <i>address</i> | - address of the addressable ADAM-4251 module, if operating in RS-485 mode. In RS-232 mode, the address is irrelevant, but must be given,   |
| <i>id</i>      | - flow computer parameter id. The following ids are allowed:<br>B - pressure (reading),<br>C - temperature (reading),<br>T - total (reading),<br>R - flow-rate (reading),<br>M - set metric units (writing),<br>P - set imperial units (writing). |

If the heat metering unit is applied, it is also possible to use the following identifiers:

- |                            |            |
|----------------------------|------------|
| E - net energy             | (reading), |
| N - net power              | (reading), |
| W - condensate temperature | (reading). |

All raw variable values are of FLOAT type.

### EXAMPLE

Examples of variables declaration:

```
JJ_01, flowrate,          V.1.R, PLC1, 1, 1, NIC_FP  
JJ_02, Total,            V.1.T,   PLC1, 1, 1, NIC_FP  
JJ_03, Pressure,        V.1.B,  PLC1, 1, 1, NIC_FP  
JJ_04, Temperature,    V.1.C,  PLC1, 1, 1, NIC_FP
```

## Time Stamp

Values of variables read from the M210G flow computer are marked with a PC local time stamp.

## Driver Parameters

M200 driver parameters are declared in the *Miscellaneous* module, on *Directly entered options* tab.

Driver parameterization is performed with the use of a separate section named **CTM200**.

- Section name: CtM200**
- Option name: LOG\_FILE**
- Option value: file\_name**

Meaning: option allows a file to be defined, where all driver diagnostic messages will be saved. If the LOG\_FILE option does not define a full path, a log file will be created in the current folder. The log file should only be used during the start-up of the Asix system.

Default value: by default, a log file is not created.

- Section name: CtM200**
- Option name: LOG\_FILE\_SIZE**
- Option value: number**

Meaning: the option allows the log file size to be determined.

Option value:

*number* - log file size in MB.

Default value: default log file size is 10 MB.

- Section name: CtM200**
- Option name: LOG\_OF\_TELEGRAMS**
- Option value: =YES/NO**

Meaning: option allows the contents of telegrams sent and received from the controller to the log file (declared with use of LOG\_FILE option) to be saved. The subject option should be used only during the start-up of the Asix system.

Default value: by default, the option value is set to NO.

- Section name: CtM200**
- Option name: RECV\_TIMEOUT**
- Option value: number**

Meaning: option allows the timeout for receiving the first response character from the flow computer to be set. This option is set globally for all devices managed by the CTM200 driver.

Option value:

*number* - value expressed in milliseconds.

Default value: by default the option value is 1000 (milliseconds).

- Section name: CtM200**
- Option name: CHAR\_TIMEOUT**
- Option value: number**

Meaning: option allows the timeout between subsequent response characters from the flow computer to be set. This option is set globally for all supported flow computers.

Option value:

*number* - value expressed in milliseconds.

Default value: by default the option value is 50 (milliseconds).

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.52 MACMAT - Driver of GAZ\_MODEM Protocol for MACMAT Station

### Driver Use

The MACMAT driver is used for communication with the MACMAT station. The driver supports the stations signed as Korektor Impulsowy (Impulse Corrector) 01723 CMK 01 97/01/02 manufactured by COMMON Ltd. and PKNMiJ 03-03-93 RP T-Zw5-1 manufactured by the PLUM company.

Parameterization of MACMAT driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the MACMAT driver requires a channel with the following parameters to be added to the Current data module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* MACMAT

**MACMAT** tab:

*Channel parameters:*

*address, COMn*

where:

*n* - number of the serial port to which the network of MACMAT stations is connected;  
*address* - station address.

### Driver Configuration

MACMAT driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

Each defined channel may have its own section, name of which is its logical name i.e. *logical\_name*. The COMn port may also have its own section named **MACMAT:n**. Values defined in such section become default values for all stations connected to a given port. If in the INI file a section named **MACMAT** is placed, then the values placed in such section become default values for all stations supported by the controller. Values placed in the section of a given station (*logical\_name*) have a priority before values placed in the section of a given serial port and the last ones have a priority before values placed in the MACMAT section. If the parameter is not present in any section, then its default value is taken according to the description below. In particular, the initialization file may not include any sections parameterizing stations. Only appropriate channel declaration is required.

Parameters of transmission with use of a serial interface cannot be placed in sections concerning individual stations.

- Section name: MACMAT**
- Option name: Auto\_sync**
- Option value: number**

Meaning - if the parameter is different from 0, then automatic synchronization of the computer clock with the MACMAT station clock. The parameter value determines the minimal time between sequent comparisons of station and computer clocks. A comparison of clocks is performed only during other data reading from the station.

Default value - 3600 (1 hour).

Parameter:  
    *number* - time in seconds.

- Section name: MACMAT**
- Option name: Alt\_port**
- Option value: COMm, metod\_of\_switching\_to\_alternative\_port, metod\_of\_switching\_to\_basic\_port**

Meaning - (See: Definition of alternative ports).

- Section name: MACMAT**
- Option name: AsComm**
- Option value: yes/no**

Meaning - if yes is given, then the driver will use AsComm Connection Manager to establish connections with MACMAT stations.

Default value - No.

- Section name: MACMAT**
- Option name: No\_Errors**
- Option value: Yes/No**

Meaning - if yes is given, then the driver will not output the messages on errors in lines and timeout.

Default value - No.

- Section name: MACMAT**
- Option name: Max\_History\_Buffers**
- Option value: number**

Meaning - determines the maximal number of buffers containing historical data, read for needs of the archiving module. One buffer includes historical data from one time interval for one variable. It is stored in the memory within the time specified by the parameter *Max\_History\_Buffers*. One buffer occupies about 400 bytes of memory and can contain 50 values. If archive data are saved by the station every 15 minutes thus, for 24 hours, 2 buffers for one variable are necessary. Historical buffers are used by the data archiving program ASPAD during completing a B type archive. After the time determined by the parameter *History\_Buffer\_Removal* buffers are removed from the memory.

Default value - 5000.

Parameter:  
    *number* - number of buffers.

- Section name:** MACMAT
- Option name:** Max\_history
- Option value:** number

Meaning - determines a time period in days, counted from the current moment backwards, for which historical data, stored in the station memory, will be read.

Default value - 35.

Parameter:  
*number* - time in days.

- Section name:** MACMAT
- Option name:** Max\_Time\_Difference
- Option value:** number

Meaning - the maximal time difference between indications of a station clock and a computer clock, after exceeding of which synchronization of clocks occurs. The parameter has a meaning only when the *Auto\_sync* parameter is different from zero.

Default value - 60.

Parameter:  
*number* - time in seconds.

- Section name:** MACMAT
- Option name:** Status\_Mask
- Option value:** number

Meaning - a number determining which values of the variable status cause non-validity of the variable value. The variable status is read from the MACMAT station together with its value. This status is a bitmap; meaning of bits is described in the station documentation. The driver executes the logical operation AND on the variable status received from the controller and on the value of the parameter *Status\_Mask*. If the result of this operation is different from zero, then the data value is invalid. The data value is invalid too if the variable status has a value of 0 (i.e. lack of data).

Default value - the default, value of 6 means that values, exceeding the measuring range, are non-validated. As the parameter value, an integer, whose individual bits correspond to appropriate bits of the status, should be given.

Parameter:  
*number* - time in seconds.

- Section name:** MACMAT
- Option name:** Counter\_Ratio
- Option value:** number

Meaning - gas flow counter is transmitted in the form of two floating-point numbers *Vn0* and *Vn1*.

Protocol specification states that the counter value is calculated according to the formula:

$$Vn0 + Vn1 * 10000$$

However, some stations use the formula:

$$Vn0 + Vn1 * 100000$$

Parameter defines the value, by which the quantity *Vn1* should be multiplied:

## Communication Drivers

$Vn0 + Vn1 * Counter\_Ratio.$

Default value - 10000.

**Section name: MACMAT**

**Option name: P\_Ratio**

**Option value: number**

Meaning - according to the MacMAT station specification, the pressure is expressed in kPa. However, some stations transfer the pressure expressed in MPa. It concerns only archive (register) data. The parameter determines, by what factor the pressure, transferred by the station, should be multiplied.

Default value - 1000.

**Section name: MACMAT**

**Option name: baud**

**Option value: number**

Or

**Section name: MACMAT**

**Option name: bps**

**Option value: number**

Meaning - transmission speed.

Default value - 9600.

Parameter:

*number* - value passed in Bd.

**Section name: MACMAT**

**Option name: parity**

**Option value: parity\_parameter**

Meaning - item determines a parity type.

Default value - n.

Parameter:

*number* - parity type:  
n – no parity but,  
o – odd parity check,  
e – even parity check,  
m – mark,  
s – space.

**Section name: MACMAT**

**Option name: retries**

**Option value: number**

Or

**Section name: MACMAT**

**Option name: retry**

**Option value: number**

Meaning - number of transmission repetitions in case of transmission errors.

Default value - 5.

**Section name:** MACMAT

**Option name:** word

**Option value:** number

Or

**Section name:** MACMAT

**Option name:** word\_length

**Option value:** number

Meaning - word length.

Default value - 8.

Parameter:

*number* - number from the range from 5 to 8 bits.

**Section name:** MACMAT

**Option name:** time-out

**Option value:** number

Or

**Section name:** MACMAT

**Option name:** timeout

**Option value:** number

Meaning - waiting time for station answer in seconds.

Default value - 2.

**Section name:** MACMAT

**Option name:** History\_Buffer\_Removal

**Option value:** number

Meaning - parameter determines time after which buffers containing historical data, read for needs of the archiving module, are removed.

Default value - 30.

Parameter:

*number* - the time is given in minutes.

**Section name:** MACMAT

**Option name:** AllErrors

**Option value:** yes/no

Meaning - if the parameter has a value of no, then the information on *timeout* errors will appear in 'Control Panel' only when the transmission missed in spite of attempts of its repetition. If it has a value of yes, then the information on all errors is transmitted to 'Control Panel'.

Default value - no.

**Section name:** MACMAT

**Option name:** RTS

**Option value:** yes/no

Meaning - if yes is given, then data sending to the station will occur by means of the RTS line set on high state and the reception on low state.

Default value - no.

**Section name:** MACMAT

**Option name:** RTS\_OFF\_Delay

**Option value:** yes/no

Meaning - time after which the RTS line will be zeroed after having sent data to the station. The parameter has meaning only when the control by means of the RTS line is switched on.

Default value - 10.

## Communication Drivers

Parameter:

*number* - time in milliseconds.

- Section name: MACMAT**
- Option name: Ignore\_Source\_Address**
- Option value: yes/no**

Meaning - each packet sent by the station includes the station address. The station address is verified by the controller. In case of incompatibility to the station number it is rejected. Setting yes will cause the controller to stop the sender address verification.

Default value - no.

### EXAMPLE 1

Channel declaration:

*Channel / Name: MAC*  
*Driver: MACMAT*  
*Channel parameters: 2,COM2*

Driver parameters:

*Section name: MAC*  
*Option name: Auto\_Sync*  
*Option value: 60*

*Section name: MAC*  
*Option name: Max\_time\_difference*  
*Option value: 10*

In the example above the station named MAC connected to the COM2 port is defined. The synchronization of computer and station clocks will be performed every 1 minute. If the difference is at least 10 seconds, then the synchronization of clocks occurs.

### EXAMPLE 2

Channel declarations:

*Channel / Name: MAC1*  
*Driver: MACMAT*  
*Channel parameters: 1,COM2*

*Channel / Name: MAC2*  
*Driver: MACMAT*  
*Channel parameters: 2,COM2*

*Channel / Name: MAC3*  
*Driver: MACMAT*  
*Channel parameters: 3,COM2*

*Channel / Name: MAC4*  
*Driver: MACMAT*  
*Channel parameters: 4,COM3*

*Channel / Name: MAC5*  
*Driver: MACMAT*  
*Channel parameters: 5,COM3*

Channel / Name: MAC6  
 Driver: MACMAT  
 Channel parameters: 6,COM4

Driver parameters:

Default value for all stations:

Section name: MACMAT  
 Option name: baud  
 Option value: 19200

Default values for stations connected to the COM3 port:

Section name: MACMAT:3  
 Option name: baud  
 Option value: 9600

Other parameters:

Section name: MAC6  
 Option name: Auto\_Sync  
 Option value: 0

In the example above stations with names from MAC1 to MAC6 are defined. The stations MAC1, MAC2 and MAC3 are connected to the COM2 port. The stations MAC4 and MAC5 are connected to the COM3 port. The station MAC6 is connected to the COM4 port. All serial ports except COM3 will work with a speed of 19200 baud. The COM3 port will work with a speed of 9600 baud. The clock of the station MAC6 will not be synchronized.

## Defining the Process Variables

### Current Measure Data

Variables allowing to access to current measure data have the following form:

$B_n$

Where:

$n$  - is the number of a datum according to the station specification:

B1 - value of a gas flow counter  
 B2 -  $Q_n$   
 B3 -  $Q_r$   
 ..... etc.

The value of  $B_n$  variable is a floating-point number.

The variable  $B_0$  is not used.

### Access to Registered Values

For measures marked according to the specification with numbers 0/1 and from 2 to 8, the access to their values remembered by the station as registered data or twenty-four-hours data (for a flow counter) is possible. The value of a registered variable is the value remembered by the station in the last registration period. An access to older measures is possible by the B type archiving. The registered variables have the following form:

R0 - gas flow counter (in the end of last twenty-four hours)  
 R2 -  $Q_n$

R3 - Qr  
 . .  
 R8 - rez2

Access to the list of alarms by using the number of the successive alarm in the list:

Variables allowing to access to the list of alarms have the following form:

*An.type*

where:

- n* - alarm no.
- type* - type of information on an alarm according to the table below.

**Table 26. Type of Information on an Alarm.**

Type Name	Meaning	Type of Received Value
c, code	alarm code (according to the station documentation)	integer number (1 byte)
v, val	increase of counter value in time of alarm duration	floating-point number (4 bytes)
sec0	the second of alarm beginning	integer number (1 byte)
m0, min0	the minute of alarm beginning	integer number (1 byte)
h0, hour0	the hour of alarm beginning	integer number (1 byte)
day0	the day of alarm beginning	integer number (1 byte)
mon0	the month of alarm beginning	integer number (1 byte)
y0,year0	the year of alarm beginning	integer number (2 bytes)
sec1	the second of alarm end	integer number (1 byte)
m1, min1	the minute of alarm end	integer number (1 byte)
h1, hour1	the hour of alarm end	integer number (1 byte)
day1	the day of alarm end	integer number (1 byte)
mon1	the month of alarm end	integer number (1 byte)
y1,year1	the year of end alarm	integer number (2 bytes)

Access to the list of alarms by means of a code of alarms:

Variables allowing to access to the list of alarms by means of an alarm code have the following form:

*En or En.type*

where:

- $n$  - alarm code according to the documentation;  
 $type$  - type of information about the alarm according to the table presented above.

Variable E allows to access to the information about the alarm with a given code. If the list of alarms does not include the code of a required alarm, then a value of 0 (integer type – 1 byte) is returned. If the list contains many alarms with a given code, then the information about the alarm which occurred as latest of all is returned. If the variable type was omitted, then a value of 1 is returned if the alarm with a given code is active, and a value of 0 otherwise. If the variable type is given, then the value from the table presented above is returned.

#### Access to the List of Alarms as Bit Mask

The variable has the form:

$EBn$

where:

- $n$  - number of byte 0-31.

By means of the EB variable it is possible to read information about active alarms in groups of 8 alarms:

- EB0 - alarms with codes 0- 7  
 EB1 - alarms with codes 8-15  
 EB2 - alarms with codes 16-23  
 ...  
 EB31 - alarms with codes 248-255

The variable value is an integer number of 1 byte length. Individual bytes of a variable value are assigned to adequate alarms. If the bit is set, then the alarm corresponding with it is active. The EB type variable allows to bind the alarms with bit strategy of identifying the alarms of Asix.

#### Access to Twenty-Four-Hours Data

The variable has the form:

$Dn$

where:

- $n$  - is a number of data in accordance to the station documentation.

The D0 variable has the same meaning as the R0 variable.

#### Access to Statistical Data

The MACMAT controller enables to access to statistical data referring to the quantity of transmitted data and quantity of errors of the transmission. Variables allowing to access to statistical data have the following form (see: the following table).

**Table 27. Variables Allowing to Access to Statistical Data.**

Address	Meaning	Type of Received Value
SBS	quantity of sent bytes	Integer number (4 bytes)
SBR	quantity of sent bytes	Integer number (4 bytes)
SFS	quantity of sent frames	Integer number (4 bytes)
SFR	quantity of sent frames	Integer number (4 bytes)

SPE	quantity of errors of parity	Integer number (4 bytes)
SFE	(frame errors) quantity of frame errors	Integer number (4 bytes)
SOE	quantity of errors of overrun	Integer number (4 bytes)
SLE	quantity of line errors (the amount of parity errors, frames, overrun and etc.)	Integer number (4 bytes)
STE	quantity of timeout errors	Integer number (4 bytes)
SPRE	quantity of protocol errors	Integer number (4 bytes)
SCE	quantity of checksum errors	Integer number (4 bytes)
SLGE	quantity of logical errors (lack of data in the controller, incorrect address etc.)	Integer number (4 bytes)
SERR	amount of all errors (SLE, STE, SPRE, SCE and SLGE). The saving of any value into ERR variable is causes zeroing the following variables: SBS, SBR, SFS, SFR, SPE, SFE, SOE, SLE, STE, SPRE, SCE and SLGE.	Integer number (4 bytes)
TSBS	quantity of sent bytes (from the beginning of driver working)	Integer number (4 bytes)
TSBR	quantity of received bytes (from the beginning of driver working)	Integer number (4 bytes)
TSFS	quantity of sent frames (from the beginning of driver working)	Integer number (4 bytes)
TSFR	quantity of received frames (from the beginning of driver working)	Integer number (4 bytes)
TSPE	quantity of errors of the parity (from the beginning of driver working)	Integer number (4 bytes)
TSFE	quantity of the frame errors (from the beginning of driver working)	Integer number (4 bytes)
TSOE	quantity of errors of overrun (from the beginning of driver working)	Integer number (4 bytes)
TSLE	quantity of errors of the line (the amount of errors of parity, frames, overrun and etc.)	Integer number (4 bytes)
TSTE	quantity of timeout errors (from the beginning of driver working)	Integer number (4 bytes)
TSPRE	quantity of protocol errors (from the beginning of driver working)	Integer number (4 bytes)
TSCE	quantity of checksum errors (from the beginning of driver working)	Integer number (4 bytes)
TERR	amount of errors determined with following variables: TSLE, TSTE, TSPRE, TSCE, TSOE	Integer number (4 bytes)

Access to Historical Data (for the Version with Access to Historical Data)

The MACMAT controller enables the archive ASPAD module to access to historical data for variables from B1 to B8 and R0 and from R2 to R8 as well as for all variables of D and X type:

- for variables R0 and B1, daily data are read;
- for variables B2 and B3, registered data of flow increment are read.; these data are scaled so that they express a flow per 1 hour; the data are scaled in the basis of registration frequency read from the station;
- for variable B4 to B8 and R2 to R8, adequate registered historical data are read.

## Cooperation with AsComm Manager of Connections

In order to use the AsComm manager of connections to establish connections with MACMAT stations you should declare the following option in the application configuration file with use of Architect module:

*Section name:* MACMAT or MACMAT:n, where n is the number of declared serial port  
*Option name:* AsComm  
*Option value:* Tak

Option should be declared in:

Architect > *Fields and Computers* > *Miscellaneous* module > *Directly entered options* tab

In case of cooperation with the AsComm module, the port number from the channel declaration is used to create the name, which is used by the driver to exchange data with the AsComm module. The name has a form of *MacMAT-n*, where *n* is the number of the serial port from the channel declaration.

### EXAMPLE

Channel declaration:

*Channel / Name:* MAC  
*Driver:* MACMAT  
*Channel parameters:* 2,COM2

Driver parameters:

*Section name:* MACMAT  
*Option name:* AsComm  
*Option value:* Yes

The example above defines driver of the name *MacMAT-2* as a client of AsComm. This name is also the name of the section, in which parameters of connections established by the AsComm module such as modem name, telephone number, etc. are placed. The description of parameters, which should be placed in such section, may be found in the manual of the AsComm module. You should notice that the number of the serial port may refer, but it has not, to the physical serial port. This number signifies the real serial port only when in the section (named 'r;MacMAT-n') for parameterization of connections established by the AsComm module other records defining truly used port (e.g. modem name) are not given.

An example of configuring the AsComm module for dial-up connections is given below:

Channel declaration:

*Channel / Name:* MAC  
*Driver:* MACMAT  
*Channel parameters:* 2,COM2

Driver parameters:

*Section name:* MACMAT  
*Option name:* AsComm  
*Option value:* Yes

*Section name:* MACMAT-2  
*Option name:* switched\_line  
*Option value:* Yes

*Section name:* MACMAT-2  
*Option name:* Modem  
*Option value:* Sportster Flash

*Section name:* MACMAT-2  
*Option name:* Interval  
*Option value:* 5m

*Section name:* MACMAT-2  
*Option name:* Max\_Connection\_Time  
*Option value:* 2m

*Section name:* MACMAT-2  
*Option name:* Number  
*Option value:* 12345678

In order to establish connections the modem Sportster Flash will be used. Connections will be established every 5 minutes with the number 12345678. Maximal connection duration time is equal to 2 minutes.

## Definition of Alternative Ports

The MACMAT driver allows using alternative serial ports in case of communication problems occurring when using the basic port (that appears in the definition of the logical channel). The parameter declaring the alternative port may be inserted into the 'MacMAT' or 'MacMAT:n' section and has the following form:

Alt\_port = COMm, metod\_of\_switching\_to\_alternative\_port,  
metod\_of\_switching\_to\_basic\_port

Option should be declared with use of Architect module in the following way:

*Section name:* MACMAT or MACMAT:n, where n is the number of declared serial port  
*Option name:* Alt\_port  
*Option value:* COMm, metod\_of\_switching\_to\_alternative\_port,  
metod\_of\_switching\_to\_basic\_port

Option should be declared in:

Architect > *Fields and Computers* > *Miscellaneous* module > *Directly entered options* tab

The parametr defines the COMm serial port used in case of communication problems in the COMn port (determined in the logical channel definition). The parameter may occur only one time (only one alternative port is permissible). The COMm alternative port is not allowed to be declared as the basic port in the other ASMEN channel definition.

Switching to the alternative port occurs after fulfilling the following condition:

*Errors\_number[/time\_period]*

When a number of missed reading attempts determined by *Errors\_number* occurs in *time\_period*, switching to the alternative port follows. The quantity of errors also includes the quantity of the transmission operation repetitions performed by the driver. That means, that if the occurrence of 3 errors is the condition of switching and the number of repetitions is 5, switching may occur during the time of current request performing and this request has a chance to be performed correctly through using the alternative channel. If not, the request will be performed with an error status and the switching to the alternative port will occur during performing next ASMEN requests. The *time\_period* parameter may be omitted - in such case, the switching will occur after *Errors\_number* appearance. The *time\_period* parameter is passed in seconds.

The return to using the basic port occurs after a number of seconds determined by the *metod\_of\_switching\_to\_basic\_port* parameter since the moment of switching to the alternative channel. It doesn't mean that the modem connection will be realized all the time (if the alternative channel is that connection). This connection will be served like so far, it means it will be disconnected as a result of the AsComm module parameterization or when all ASMEN requests are realized. The parameters of the serial port should be defined in the 'MacMAT' section if they are supposed to be other than the ones applied in case of the basic port.

### EXAMPLE

An example of the alternative channel parametrization:

Channel declaration:

*Channel / Name:* MAC1  
*Driver:* MACMAT  
*Channel parameters:* 220,COM1

Driver parameters:

*Section name:* MACMAT  
*Option name:* Baud  
*Option value:* 9600

Switching to the alternative channel, after 3 following errors have occurred, and return switching after 2 minutes:

*Section name:* MACMAT  
*Option name:* Alt\_Port  
*Option value:* COM2, 3, 120  
*Section name:* MACMAT  
*Option name:* Alt\_Port  
*Option value:* COM2, 15/60, 120

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

**Purpose** - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

**Option value:**

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

**The default value**

- NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.53 Max1000 - Drajwer Protokołu Systemu MAX-1000 Firmy ULTRAK

**Max1000.** Communication based on the so-called "Network protocol" with the MAX 1000 camera management system by ULTRAK. The driver only carries out the camera switching and time synchronization functions (the driver provides time stamp). Communication takes place using the RS-232 serial interface.

Parameterization of M-Bus driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the Max1000 driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

Name: logical name of the transmission channel

Driver: Max1000

**Max1000** tab:

*Channel parameters:*

```
Port=PortNo; Node=NodeNo [; Baudrate=baud]
[; SynchrNode=SynchrNodeNo [; SynchrUTC=YES|NO] ]
```

where:

**Port** - number of a serial port COM through which Asix communicates with the network MAX-1000.

**Node** - number of a broadcasting node of MAX-1000 network assigned to Asix. This number will be used as a parameter {TX} in telegrams sent to the system MAX 1000 (parameter {RX} will be downloaded from the symbolic address of variable).

**Baudrate** - optional parameter. It is used to determine the transmission speed of serial link. Acceptable parameters are 9600 and 19200. Default: 19200 Bd.

**SynchrNode** - optional parameter. It is used to synchronize the time of MAX-1000. It contains the number of MAX-1000 network node to which the current date and time of Asix will be sent every minute. By default, the driver does not synchronize the time.

**SynchrUTC** - optional parameter. It is used to determine the type of time (UTC or local) sent to MAX-1000 during time synchronization. By default, the driver sends

local time during time synchronization. Transmission is carried out at constant settings: 7 bits of mark, even parity check, 1 stop bit.

#### EXAMPLE

An exemplary declaration of a channel in which a node with the number 5 of MAX-1000 is assigned to Asix:

*Name:* CHANNEL

*Driver:* Max1000

*Driver parameters:* Port=1; Node=5

## Addressing the Process Variables

The driver provides only one type of variable:

**K** - camera

The syntax of symbolic address which is used for variables belonging to the CtMax1000 driver channel is as follows:

K.<CameraNo>.<NodeNo>

where:

**CameraNo** - global number of camera;

**NodeNo** - number of node, in which the camera is installed. The parameter is used as the parameter {RX} of the command switching over between cameras.

All the variables are of WORD type.

Variables are used only to send controls related to the switching of cameras - therefore when reading variables the value of 0 and the status OPC\_QUALITY\_BAD are returned.

For controls (switching the camera) the number of monitor, to which the image of a camera assigned to a given variable will be switched over, should be declared as a control value.

#### EXAMPLE

Exemplary declaration of variables:

JJ\_11, camera 1 node 1, K.1.1, CHANNEL, 1, 1, NOTHING  
JJ\_12, camera 2 node 1, K.2.1, CHANNEL, 1, 1, NOTHING  
JJ\_13, camera 3 node 1, K.3.1, CHANNEL, 1, 1, NOTHING  
JJ\_14, camera 1 node 2, K.1.2, CHANNEL, 1, 1, NOTHING  
JJ\_15, camera 2 node 2, K.2.2, CHANNEL, 1, 1, NOTHING  
JJ\_16, camera 3 node 2, K.3.2, CHANNEL, 1, 1, NOTHING

JJ\_17, camera 1 node 3, K.1.3, CHANNEL, 1, 1, NOTHING

## Parameters of Driver

Max1000 driver parameters are declared in the *Miscellaneous* module, on the *Directly entered options* tab.

The driver is parameterized with use of **CtMax1000** section.

**Section name: CtMax1000**

**Option name: LOG\_FILE**

**Option value: file\_name**

Meaning - allows to define a file where all diagnostic messages of the CtMax1000 driver and information about the contents of telegrams received and sent by the CtMax1000 driver will be written.

Default value - by default, the contents of telegrams are not written to the log file.

**Section name: CtMax1000**

**Option name: LOG\_FILE\_SIZE**

**Option value: number**

Meaning - allows to specify the size of log file in MB.

Default value - 10 MB.

**Section name: CtMax1000**

**Option name: LOG\_OF\_TELEGRAMS**

**Option value: YES/NO**

Meaning - allows to write to the log file (declared by using the item LOG\_FILE) the contents of telegrams sent by the CtMax1000 driver within the reading/writing process variables. The log file should be used only while the Asix start-up.

Default value - NO.

### EXAMPLE

```
[CTMAX1000]
LOG_FILE =d:\tmp\test\max1000.log
LOG_FILE_SIZE =30
LOG_OF_TELEGRAMS =YES
```

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

## Communication Drivers

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose

- with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose

- with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value

- NO.

### **Data Validity Period**

Purpose

- this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value

- by default, no time for data validity is set.

## 1.54 M-BUS - Driver of M-BUS Protocol

### Driver Use

The M-Bus standard was developed as a standard for communication with heat counters and is the most widespread in this branch. This protocol was tested and developed in connection with MULTICAL heat meters of KAMSTRUP A/S.

Parameterization of M-Bus driver is performed with the use of Architect module.

### Driver Configuration

Declaration of the transmission channel utilizing the M-Bus driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* M-Bus

### Defining the Process Variables

The set of variables handled by the driver may be divided into several groups:

- sequential variables, i.e. the variables the address of which is the number of the datum transferred by the M-BUS device;
- variables, the address of which contains the name of measured variable;
- variables, which enable to read the manufacturer's data and other variables.

#### Sequential Variables

The definition of sequential variables needs the sequence in which the M-BUS device sends measured data to be known.

The address of sequential variables is as follows:

$$P_n$$

where  $n$  is a variable order number.

#### Addressing by Means of Variable Name

The variable address is as follows:

$$Name [.Un][.Tn][.Sn]$$

where:

*Name* - name of a measured variable;  
*Un* -  $n$  - unit number (if omitted, then it is assumed to be equal to 0). The unit number is applied when the device consists of several units;

$T_n$                      $n$  - tariff number (if omitted, then it is assumed to be equal to 0);  
 $S_n$                      $n$  - cell number to store historical data (storage) (if omitted, then it is assumed to be equal to 0).

It is allowed to use the following names (see: the following table).

**Table 28. Set of Acceptable Measured Variable Names.**

<b>Name</b>	<b>Meaning</b>
ACCESSNUMBER	Next number of data reading
ACTDURATION	Duration time in seconds
AVGDURATION	Duration time in seconds
BAUDRATE	Transmission speed
BUSADD	Device address
CREDIT	Credit
CUSTOMERLOC	Localization of client
DEBIT	Debit
DIGINPUT	Digital input
DIGOUTPUT	Digital output
EIDENT	Extended identification
ELCURRENT	Current in amperes
ENERGY	Energy
FABRNO	Factory number
FLOWTEMP	Temperature
FVERSION	Firmware version
HVERSION	Equipment version
MANUFACTURER	Manufacturer
MASS	Mass
MASSFLOW	Masse flow
MEDIUM	Code of measured medium
MODEL	Model
ONTIME	Time from turning on
OPERTIME	Work time
PARAMSETID	Identification of parameters
POWER	Power
PRESS	Pressure
RESPDELAY	Delay of device response
RETTEMP	Return temperature
SVERSION	Software version
TEMPDIFF	Difference of temperatures
TIMEPOINT	Time stamp
VOLUME	Volume
VOLFLOW	Volume flow
XVOLFLOW	External volume flow
XTEMP	External temperature

### Addressing the Manufacturer's Data

Manufacturer's data are the data, that are not described in the protocol definition. To read them, the knowledge of manufacturer's data structure of a given device is necessary.

The address of a manufacturer's datum is as follows:

*Mposition.length*

where:

- position* - byte number in the manufacturer's data block, from which a given value begins; the first byte has the number 0;
- length* - data length in bytes.

The driver assumes that the manufacturer's data are expressed in the BCD code.

### Access to Unit Symbol of Measurement

For sequential variables and variables addressed with a variable name it is allowed to define variables that return the symbol of a measured physical unit (e.g. Wh for energy). To do it, you should add /UNIT to the variable address e.g. ENERGY/UNIT. As a conversion function you should define NOTHING\_TEXT. In order to display the unit on technological diagram you can use the object STRING.

### Other Data

Data transferred by the M-BUS device may be provided with a header. Variables allowing to access the data in a header:

**Table 29. Set of Variables Allowing to Access the Data in a Header.**

Address	Meaning	Type
H.IDENT	Identifier of device	DWORD
H.MANUFACTURER	Manufacturer code	
H.VERSION	Version	DWORD
H.MEDIUM	Medium code	BYTE
H.ACCESSNO	Next number of reading	BYTE
H.STATUS	Data status	BYTE

### Data Status

Devices operating according to the M-BUS protocol make available the datum of 1 byte length, the individual bits of which determine the device status in the following way (see: **Table 29**):

**Table 30. Data Statuses for M-BUS Devices.**

Number	Meaning	Byte Number
1	application engaged	0
2	application error	1
3	voltage drop	2
4	constant error	3
5	temporary error	4
6	error specific for the device	5
7	error specific for the device	6
8	error specific for the device	7

Statuses with the successive number from 1 to 5 cause all the data sent by the device to be cancelled by the driver i.e. they take the status *bad datum*. It doesn't apply to the data contained in the header described in point Other Data. This default operation of the driver may be changed by the parameter *Invalid\_States* (see: *Driver Configuration*). The datum containing the device status may be read by the variable H.STATUS described in point Other Data. The first column of the above table determines the byte number, which corresponds to the specified status, in the variable.

Not all the devices make available the statuses 1 and 2. The meaning of statuses 6,7 and 8 is defined by the manufacturer.

## Driver Parameterization

The driver configuration is defined in the *Current Data* module, in the channel operating according to M-Bus driver.

**Channel parameters - transmission** tab:

### Port and transmission parameters

**Port number**

Meaning - It specifies the serial port used for communication. The parameter is obligatory.  
 Default value - COM1.

**Baud rate**

Default value - 9600.

**Word length**

Default value - 8.

**Parity**

Default value - even.

**Number of stop bits**

Default value - 11.

### Address of the device

**Address of the M-BUS device**

Meaning - determines the address of a given M-BUS device. The parameter is an obligatory parameter.  
 Default value - no default value.  
 Parameter:  
     *number* - the parameter is a number from a range 1 to 250.

The parameter *Port* is obligatory. If transmission parameter was omitted, then the default values are taken. The port number must be always given.

\*\*\*

**Channel parameters - advanced** tab: **Refresh period**

Meaning - determines an interval by means of which the driver reads data from the M-BUS device.

Default value - 15.

Parameter:  
    *number* - is passed in seconds.

 **Refresh delay**

Meaning - the parameter determines the minimal time between successive data readings from the M-BUS device. Some devices (e.g. MULTICAL) needs a considerable time to prepare data. The parameter determines the time necessary for preparing data by the M-BUS device.

Default value - 12.

Parameter:  
    *number* - is passed in seconds.

 **Double read**

Meaning - some devices (e.g. MULTICAL) return the data prepared after a previous reading. If the parameter has a value Yes, then driver will do two successive reading in order to obtain the most actual data.

Default value - Yes.

 **Response timeout**

Meaning - the parameter determines a maximal waiting time for an answer. The time is expressed in milliseconds.

Default value - 0.

Parameter:  
    *number* - time in milliseconds.

 **Character timeout**

Meaning - the parameter determines the maximal waiting time to receive one character.

Default value - 0.

Parameter:  
    *number* - time in milliseconds.

\* \* \*

**Channel parameters - advanced 2** tab: **Invalid statuses**

Meaning - the M-BUS device send a status byte with measured data. Each of bits of this byte determines a specified data value. The parameter determines what status bits make the received data invalid.

Default value - +1,+2,+3,+4,+5.

Parameter:  
    *number,number,...* - the parameter has a form of set of bit numbers separated with the '+' character. The least significant bit has the number 1. The default value (1,2) means that the data are found incorrect if the device signals the error "application busy" (1), "application error"

(2), "power drop" (3), "constant error" (4) and "temporary error" (5). The manufacturer may define additional statuses.

\*\*\*

**Channel parameters - alarms** tab:

**Alarm code**

Meaning - the parameter determines the alarm number in the Asix system which is generated by the driver after a lost of connection with the M-BUS device.

Default value - no default value.

**Alarms**

Meaning - is a set of parameters with names from *Alarm0* to *Alarm7*. Each parameter determines a number of Asix system alarm, which will be generated by the driver after appearance of an analogous alarm in the M-BUS device. The meanings of alarms generated by the M-BUS device is specified by the manufacturer.

Default value - no default value.

\*\*\*

**Channel parameters - diagnostics** tab:

**Log file**

Meaning - the parameter value is a name of file where diagnostic information is written. The parameter may be used only for diagnostic purposes.

Default value - no default value.

**Dump file**

Meaning - the parameter value is a name of the file where the data will be written. The parameter may be used only for diagnostic purposes.

Default value - no default value.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose

- with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose

- with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value

- NO.

**Data Validity Period**

Purpose

- this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value

- by default, no time for data validity is set.

## 1.55 MEC - Driver of MEC07 and MEC08 Heat Meter Protocol

### Driver Use

The driver is designed for data exchange between the Asix system and MEC07 and MEC08 heat meters manufactured by the Instytut Techniki Ciepłej (Institute of Thermal Technology) in Łódź. Data exchange is executed by means of a serial interface in the RS-232 standard. For switching serial line between counters a multiplexer, controlled by RTS (switching to the first channel) and DTR (switching to the next channel) lines, is used.

Parameterization of MEC driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the MEC driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

Name: logical name of the transmission channel  
Driver: MEC

**MEC** tab:

*Channel parameters:*  
*port* [,*max\_no*]

where:

- |               |  |
|---------------|--|
| <i>port</i>   | - name of the serial port connected to the multiplexer of MEC heat meters, e.g. COM1,                |
| <i>max_no</i> | - optional number of multiplexer channels to which the MEC heat meters are connected (by default 5). |

#### EXAMPLE

An exemplary declaration of transmission channel CHANNEL used for communication with MEC heat meters connected to the multiplexer channels numbered from 1 to 8. Data exchange is executed through the serial port COM2.

*Channel / Name:* CHANNEL  
*Driver:* MEC  
*Channel parameters:* COM2, 8

### Types of Process Variables

In the driver one type of process variables is defined:

- V** - value of transferred measurement.

All process variables are of FLOAT type and may be only read. A time stamp is given by a heat counter and sent with values of process variables within a common telegram. The time stamp is given with the accuracy of one minute.

The syntax of symbolic address used for variables belonging to the MEC driver channel is as follows:

$$V<nrFabr>.<nrOdb>.<index>$$

where:

- nrFabr* - factory number of the MEC heat meter;
- nrOdb* - number of the receiver from which the measurement is transferred; allowed value is 1 or 2;
- index* - variable number in a table of variables transferred from the meter.

The list of indexes and variables corresponding with them are given in tables 1 and 2.

Raw values of variables read from heat counters should be converted following the conversion function NOTHING\_FP or FACTOR\_FP. The column *Factor* in both tables contains factors **A** of the conversion function FACTOR\_FP for measurements with known conversion method of raw value (factors **B** are equal to 0). For the other measurements the calculation method of real value of the measurement should be agreed with the user of heat meters.

**Table 31. Variables Transferred from MEC07.**

Variable Name	Index	Factor
Factory number	1	1
Number of receiver	2	1
Status	3	1
Time of exceeding Max	4	
Time of exceeding Min	5	
Time of range exceeding	6	
Total time	7	0.1
Time of damage	8	
Total masse	9	10
Number of exceeding	10	
	11	
Number of feed decays	12	
Actual power	13	0.001
Current of break. PDPO	14	
Flux of masse	15	0.01
Ordered power	16	0.001
Delta of energy	17	
Qmax (exceeding)	18	
Qmin (exceeding)	19	
Qsum (total energy)	20	
Duration time of exceeding	21	
Max power of exceeding	22	
Energy of exceeding	23	
Volume of exceeding.	24	
V/Vz of exceeding.	25	
Air temperature (Tpow)	26	0.01
Delta T (Trozn)	27	0.01
Average air temp. (Tsrpow)	28	0.01
Delta T average (Tsrroz)	29	0.01
Temp.of feed. (Tzas)	30	0.01

**Table 32. Variables Transferred from MEC08.**

Variable Name	Index	Factor
Factory number	1	1
Receiver number	2	1
Status	3	1
Pressure	4	0.001
Time of exceeding.	5	
Time in exceeding min.	6	
Time in feed.	7	
Time of range exceeding	8	
Time of range exceed. of condens.	9	
Total time	10	0.1
Time of damage	11	
Close time	12	
Close time of feed	13	
Enthalpy	14	1
Enthalpy of condens.	15	0.1
Total masse of condens..	16	0.1
Total masse	17	0.1
Number of exceedings	18	
Number of feed decays	19	
Steam power	20	0.01
Current of break. PDP0	21	
Current of break. PDP1	22	
Power of condens.	23	0.01
Ordered power	24	0.01
Energy in exceeding	25	
Energy in exceeding min.	26	
Energy in feed..	27	
Energy of condens.	28	0.1
Total energy	29	0.1
Duration time of exceeding.	30	
Max power of exceeding.	31	
Energy of exceeding.	32	
Flux of condens. masse	33	0.01
Flux of masse	34	0.01
Overheat (delta T)	35	0.1

**EXAMPLE**

Examples of declarations of variables.

A0, time 7502,	V7502.1.00, CHANNEL, 17, 1, DATETIME_MEC
A1, pressure 7502,	V7502.1.04, CHANNEL, 1, 1, FACTOR_FP, 0.001, 0
A2, temp. of feed 7502,	V7502.1.37, CHANNEL, 1, 1, FACTOR_FP, 0.1, 0
A3, total time 8502,	V8502.1.07, CHANNEL, 1, 1, FACTOR_FP, 0.1, 0
A4, factory no 7502,	V7502.1.01, CHANNEL, 1, 1, NOTHING_FP

## Driver Configuration

MEC driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **MEC** section.

**Section name: MEC**

**Option name: LOG\_FILE**

**Option value: file\_name**

Meaning - allows to define a file to which all diagnostic driver messages and information about contents of telegrams received by the driver are written. If the item does not define the full path, the log file will be created in the current directory. The log file should be used only in the stage of the Asix start-up.

Default value - log file is not created.

Defining - manual.

**Section name: MEC**

**Option name: LOG\_FILE\_SIZE**

**Option value: number**

Meaning - allows to specify the log file size in MB.

Default value - 1MB.

Defining - manual.

**Section name: MEC**

**Option name: LOG\_OF\_TELEGRAMS**

**Option value: YES|NO**

Meaning - item allows to write to the log file (declared by use of the item LOG\_FILE) the contents of telegrams received by the driver. Writing the contents of telegrams to the log file should be used only in the stage of the Asix system start-up.

Default value - NO.

Defining - manual.

**Section name: MEC**

**Option name: DATA\_VALID\_TIME**

**Option value: number**

Meaning - for each MEC counter the time of the last reading is checked. If it exceeds the declared value, then the data from the actually read MEC counter receive an error status.

Default value - 1min.

Defining - manual.

**Section name: MEC**

**Option name: STROBE\_TIME**

**Option value: number**

Meaning - allows to determine the duration time (in milliseconds) of the RTS and DTR strobe while controlling the switching of multiplexer channels.

Default value - 60 ms.

Defining - manual.

- Section name:** MEC
- Option name:** NUMBER\_OF\_REPETITIONS
- Option value:** number

Meaning - allows to determine the number of repetitions for all channels of multiplexer.  
Default value - 1 repetition.  
Defining - manual.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value

- NO.

**☑ Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value

- by default, no time for data validity is set.

## 1.56 MegaMuz - Driver for Communication with Microprocessor Protection Devices of MegaMuz type by JM-Tronik, Warsaw

### Driver Use

MegaMuz driver is used to exchange data between the Asix system and microprocessor protection devices of MegaMuz type developed by JM-Tronik, Warsaw. Communication is carried out via RS-485.

The driver performs the following functions:

- readout of current status and measurements,
- readout of settings,
- readout of control settings,
- readout of events and reporting them to the ASIX alarm system,
- disturbance record readout and registering in database in the format of AsLogger developed by ASKOM Sp. z o.o,
- controlling outputs.

The driver supports the following protection devices of MegaMuz type:

CR, CR\_87, CR57

LR

LZ, LZ\_87, LZ57

PR, PR\_87

SR, SR\_87

TR, TR\_87

NOTE:

Support of other protection devices of MegaMuz type requires changes in the driver code.

### Declaration of Transmission Channel

Declaration of the transmission channel using the MegaMuz driver requires adding a channel with the following parameters to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: MegaMuz

**MegaMuz/Channel Parameters** tab:

*Device Number* - network device number,

*Device Type* - megaMuz type. Identifiers of supported types:

1 - CR

2 - LR

3 - LZ

4 - PR

5 - SR

6 - TR

7 - CR\_87  
 8 - LZ\_87  
 9 - PR\_87  
 10 - SR\_87  
 11 - TR\_87

*Alarms Offset* - offset added to the calculated Asix alarm number,

*Port* - serial port name,

#### *Control Variables*

- variable used to show the communication status with megaMuz:

0 - transmission o.k.

1 - no communication/communication error

2 - disabling the support for megaMuz,

- variable used to control the communication status with megaMuz:

1 - request to disable the support for megaMuz,

0 - enabling the support for megaMuz.

#### *Time Synchronization*

- time synchronization period with a megaMuz device (in seconds). 0 means no synchronization (default value).

#### Note:

a/ control variables must belong to a channel of NONE type.

b/ fixed communication parameters: baud rate - 9600, the number of bits in a character - 8, the number of stop bits - 1.

c/ the *Control variables* and *Period* options require defining both variables.

Below follows an example of KPR channel declaration, supporting a megaMuz device of the network number 5, PR type, connected to the COM1 serial port, with alarm offset 200. The channel status is indicated by MuzStat05 variable, the channel is controlled by MuzCtrl05 variable, time synchronization is carried out every 60 seconds:

*Device Number:* 5

*Device Type:* 4

*Alarms Offset:* 200

*Port:* COM1

variable used to show the communication status with megaMuz: MuzStat05

variable used to control the communication status with megaMuz: MuzCtrl05

*Time Synchronization:* 60

## Declaration of Variables

The driver provides the following variable types of the MODBUS protocol:

CS - binary value (write),

HR - 16-bit register (read)

HRLL - 2 registers including a number of DWORD type (read)

HRFL - 2 registers including a number of FLOAT type (read)

and the driver's internal variables:

MN - network number of the megaMuz device from which the recorder is/was read (read)

## Communication Drivers

RN - number of currently/recently read recorder (read)

RQ - request for recorder readout (write)

PR - status of currently/recently read recorder (readout)

RS - cause for triggering the recorder (read)

RT - time of triggering the recorder (read)

SN - number of recently read recorder sample (read)

UDC - used to execute the MODBUS RTU protocol commands defined by the user, available only for keys WAUS, WAUW, WDUW or WDUN. The value of a UDC type variable is a string of ASCII characters, whose maximum length is 521 bytes. While writing, it is the content of a command defined by the user; while reading, it is the content of a response to a recently sent user's command. (read/write)

The address of a variable has the following syntax:

*Type[Index]*

where:

*Type* - the variable type,

*Index* - index of a variable within the 'Type' variable type.

### NOTE:

1. for the following types no indexes need to be provided: MN, RN, RR, RQ and SN,
2. in case of RS, RT types the index range is dependant on the number of recorders defined in the device,
3. the parameter of request for recorder readout (RQ type) is the recorder number. The recorders are numbered starting from 1. The parameter set to 0 means a request to terminate the currently conducted recorder readout,
4. the ranges for HR types are dependant on the device type,
5. in order to readout the time of triggering the recorder, the DATAZAS\_MUZ conversion function should be used; as a result, a number of DWORD type will be converted into a string of characters of the following format:

yyyy-mm-dd hh:nn:ss.zzz

Examples of variable declarations (variable values originate from a device of LR type, available through the channel named *KLR\_01*):

ZM_01, value I1,	HRFL28, KLR_01, 1, 1, NIC_FP
ZM_02, protection excitement factor 1,	HR19, KLR_01, 1, 1, NIC
ZM_03, number of non-read events,	HR27, KLR_01, 1, 1, NIC
ZM_04, value U1,	HRFL46, KLR_01, 1, 1, NIC_FP
ZM_06, time of triggering the recorder 1,	RT1, KLR_01, 1,1, DATAZAS_MUZ

Examples of variable declarations without parameters (variable values originate from a device of LR type, available through the channel named *KLR\_01*):

ZM_07, request for recorder readout,	RQ, KLR_01,1,1,NIC
ZM_08, number of the recorder being read,	RN, KLR_01,1,1,NIC
ZM_09, recorder readout status,	RR, KLR_01,1,1,NIC
ZM_10, number of read sample,	SN, KLR_01,1,1,NIC
ZM_11, number of the device being read,	MN, KLR_01,1,1,NIC

## Disturbance Recording

The driver provides for automatic disturbance recording in the AsLogger database. Review of the recorded disturbances is possible only with the use of the AsLogger application.

In order to activate the disturbance log mode, the Disturbance recording in AsLogger database option (in the Driver parameters tab) must be declared, the option specifying:

- MSSQL server name,
- database name,

in which the driver will store the disturbances read.

Automatic disturbance recording is not provided by default - each time it must be initiated by the Asix system operator by the execution of control with the use of a RQ type variable. The control value is the number of the recorder, whose content must be read by megaMuz and recorded in the AsLogger database. The control value set to 0 means a request to terminate the currently conducted recorder readout.

During recorder readout it is possible to view the readout status. To this end, the driver's internal variables of the below symbolic addresses are used:

- a/ *MN* - number of the megaMuz device being read,
- b/ *RN* - number of the recorder being read,
- c/ *RR* - recorder readout status:

0 - disabled or completed o.k.  
 1 - readout in progress  
 2 - readout aborted due to an error

- d/ *SN* - number of recently read recorder sample.

Disturbance recording is carried out in line with registration plans. The names of registration plans can be:

- a/ defined by the user,
- b/ generated automatically by the driver.

While recording, the driver stores in the AsLogger database the following disturbance parameters:

- a/ complete set of the plan's measurement points, including:

currents : I0, I1, I2, I3 (on the secondary side)  
 voltages : I0, I1, I2, I3 (on the secondary side)  
 status of inputs : WE01 ... WE24  
 status of outputs : WY01 ... WY24  
 excitement status of protections 1 : PZ01 ... PZ16  
 tripping of protections 1 : ZZ01 ... ZZ16

The set of parameters and number of the plan's measurement points referring to excitement and tripping of protections are specific for each megaMuz type,

b/ it creates plan archives,

c/ it determines the maximum and minimum values for currents and voltages in the disturbance recorded - if necessary, the values can be further adjusted while reading subsequent disturbances within a given plan.

- **Defining the names of registration plans by the user**

The user can define the names of registration plans in text files with '.def' extension. To this end, for each transmission channel (ASMEN channel) a separate file must be created, to which the definitions of registration plan names for disturbances recorded in a given channel will be entered. The files including the definitions of registration plan names should be stored to a common directory. The name of the directory should be stored in the location **Directory with Files Containing Registration Plan Names** in the driver parameters.

The syntax of the definition of a registration plan name stored in a '.def' file has the following form:

channel, cause, table [, comments]

where:

<i>channel</i>	- name of the ASMEN channel,
<i>cause</i>	- cause for triggering the recorder (number between 0 and 15),
<i>table</i>	- registration plan name assigned to a disturbance,
<i>comments</i>	- comment

- **Default names of registration plans**

If the user has not defined his own names of registration plans, the driver will on its own create the name of a registration plan, based on the following syntax:

NrxxxpyyyPowod

where:

xxx	- network number of the megaMuz device,
yyy	- COM port number,
Reason	- abbreviation identifying the cause for triggering the recorder.

**Example:**

For a disturbance caused by triggering the user's logic recorder, for the megaMuz device number 2 on COM5 a registration plan of the following name will be created:

Nr002P005logika

## Event Signalling

Event signalling is checked cyclically, check frequency is defined by the item **Event Check Period**. The events signalled in megaMuz are converted into respective alarm numbers in the Asix system. The conversion is made based on mapping defined for each megaMuz type

and the offset specified in the ASMEN channel declaration. Two types of Asix alarms are available:

a/ text only

b/ text and one numerical parameter

Events of 0 type are converted into Asix alarms of the following form: text + one numerical parameter (of int type) + text parameter (max. 3 characters), including phase numbers.

MegaMuz events of type No. 2 are converted into Asix alarms of *text* type.

### Example 1

The event *Closing switch from controller* in megaMuz of LR type (event of *text* type) is mapped into Asix alarm No. 34. If an alarm offset of 400 is specified in the ASMEN channel declaration, then the alarm definition in the Asix alarm definitions file will have the following form ( $400 + 34 = 434$ ):

**434, al, closing switch from controller**

### Example 2

The event *Operation of fault protection device* in megaMuz of LR type (event of type *text and two parameters: numerical and text*) is mapped into Asix alarm No. 0. If an alarm offset of 100 is specified in the ASMEN channel declaration, then the alarm definition in the Asix alarm definitions file will have the following form ( $100 + 0 = 100$ ):

**100, al, operation of fault protection device %f, of phase %s**

Mapping of individual megaMuz event types onto Asix alarm numbers is stored in the *NumericalAlarmAsixa.xls* file.

## Driver Parameters

The MegaMuz driver parameters are declared in the *Current data* module in the **MegaMuz / Driver Parameters** tab... of the channel operating in line with the MegaMuz driver protocol.

The parameters present in the *Driver Parameters* tabs refer to all the channels with the same driver declared. If multiple channels operate based on the same driver, the parameters set in the *Driver Parameters* tabs in the definition of one of these channels are automatically displayed in the *Driver Parameters* tabs in the definitions of the other channels.

(The parameters present in the *Channel parameters* tabs are applicable only to the specific channel.)

Driver parameters:

- creating a log file,
- creating a disturbance log file,
- the log file size,
- log of telegrams,
- number of repetitions,
- response timeout,
- inter-character timeout,
- event signalling check period,

## Communication Drivers

- disturbance recording in the AsLogger database.
- file directory including the definitions of registration plans.

The names of the items related with the log file refer to the convention used in other ASMEN drivers.

### ***MegaMuz / Driver Parameters - Diagnostics*** tab:

#### **Log File**

Purpose - the text log file to which driver status messages are stored is used for diagnostic purposes.

The default value - by default, the log file is not created.

Parameter:

*file name*

#### **Log File Size**

Purpose - this option is used to determine the log file size defined using the *Log file* option.

Parameter:

*number* - the size of the log file in MB.

The default value - by default, the log file size is 10 MB.

#### **Log of Telegrams**

Purpose - this option allows writing the contents of messages communicated between the driver and the controllers to the log file (declared using the *Log File* item) The option in question should only be used during the Asix system start-up.

The default value - by default, this option is disabled.

#### **Disturbances Log File**

Purpose: the text log file to which messages about read disturbances are stored is used for diagnostic purposes.

The default value: by default, the log file is not created.

Option value:

*log file name*

### ***MegaMuz / Driver Parameters*** tab:

#### **Number of Repetitions**

Purpose: This option is used to define the maximum number of repetitions in case of a transmission error.

The default value: By default the value of the item is set to 3.

Option value:

*number* - maximum number of repetitions in case of a transmission error.

**☑ Response Timeout**

Purpose: this option specifies the maximum time throughout which the driver waits for a response from megaMuz. This option is of global character and its value can be overridden by individual channel settings.

The default value: the default response timeout value is 1000 milliseconds.

Option value:  
*number* - time out in milliseconds.

**☑ Char Timeout**

Purpose: This option specifies the maximum time between response characters from megaMuz. This option is of global character and its value can be overridden by individual channel settings.

The default value: the default response character timeout value is 100 milliseconds.

Option value:  
*number* - time out in milliseconds.

**☑ Event Check Period**

Purpose - this option specifies the interval (in seconds) before reading the event log of a subsequent device connected to the same serial port supported by the driver.

The default value - the default value is 10 seconds.

Option value:  
*number*

**☑ Registration of disturbances in AsLogger Database**

Purpose: This option switches the driver into disturbance registration mode. The option defines the server name and the database name to which disturbance samples will be stored.

The default value: by default disturbances are not registered.

Option value:  
*ServerName* - MSSQL server name  
*DatabaseName* - database name on the *ServerName*

**☑ Directory with Files Containing Registration Plan Names**

Purpose: this option specifies a complete access path to the directory in which the driver will search for files including the definitions of registration plan names.

The default value: -.

Option value:  
*path* - complete path to the file directory including the definitions of registration plans.

## Channel Parameters

The channel is configured using options present in the **Channel parameters** tabs.

- creating a log file,
- the log file size,

## Communication Drivers

- log of telegrams,
- response timeout,
- response character timeout,
- without events,
- without register.

### **MegaMuz / Channel Parameters - Diagnostics** tab:

#### **Log File**

Purpose - the text log file to which channel status messages are stored is used for diagnostic purposes.

The default value - by default, the driver log file is used.

Parameter:

*file name*

#### **Log File Size**

Purpose - this option is used to determine the channel log file size defined using the *Log file* option.

Parameter:

*number*

- the size of the log file in MB.

The default value - by default, the log file size defined for the driver is used.

#### **Log of Telegrams**

Purpose - this option allows writing the contents of messages communicated in the channel to the channel log file (declared using the *Log File* option). The option in question should only be used during the Asix system start-up.

The default value - by default, the setting defined for the driver is used.

### **MegaMuz / Channel Parameters 2** tab:

#### **Response Timeout**

Purpose: This option specifies the maximum time throughout which the driver waits for a response from megaMuz. The option value overrides the settings in the driver options.

The default value: the default response timeout value is 1000 milliseconds.

Option value:

*number*

- time out in milliseconds.

#### **Char Timeout**

Purpose: This option specifies the maximum time between response characters from megaMuz. The option value overrides the settings in the driver options.

The default value: the default response character timeout value is 100 milliseconds.

Option value:

*number*

- time out in milliseconds.

**Without Events**

Purpose: this option is used to disable readout of the event log in a channel.  
 The default value: by default, readout of the event log is enabled.  
 Option value: YES/NO

 **Without Register**

Purpose: this option is used to disable readout of the disturbance log file in a channel.  
 The default value: by default, readout of the disturbance log file is enabled.  
 Option value: YES/NO

**EXAMPLE**

Driver parameterization:

```
Log File =c:\tmp\muz.log
Disturbances Log File=c:\tmp\zaklocenia.log
Log File Size =30
Log of Telegrams =YES
Registration of disturbances in AsLogger Database =SERWER_RO6, BAZA_RO6
Directory with Files Containing Registration Plan Names =c:\tmp\DefZaklocen
```

Channel parameterization:

```
[CHANNELX]
Log File =c:\tmp\channelx.log
Log File Size =20
Log of Telegrams =YES
Char Timeout =200
Response Timeout=1500
```

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

 **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

## Communication Drivers

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.57 MegaMuz2 - Driver for Communication with Microprocessor Protection Devices of MegaMuz2 Type by JM-Tronik, Warsaw

### Driver Use

MegaMuz2 driver is used to exchange data between the Asix system and microprocessor protection devices of MegaMuz2 type developed by JM-Tronik, Warsaw. Communication is carried out in MODBUS RTU mode via serial link.

The driver performs the following functions:

- readout of current status and measurements,
- readout of events and reporting them to the ASIX alarm system,
- disturbance record readout and registering in database in the format of AsLogger developed by ASKOM Sp. z o.o;

The driver supports the following protection devices of MegaMuz type:

- 01.12

### Declaration of Transmission Channel

Declaration of the transmission channel using the MegaMuz2 driver requires adding a channel with the following parameters to the Current data module:

**Standard** tab:

*Name:* logical name of the transmission channel

*Driver:* MegaMuz2

**MegaMuz2/Channel Parameters** tab:

*Device Number* - network device number,

*Device Type* - megaMuz type. Identifiers of supported types:

1 - 01\_12

*Alarms Offset* - offset added to the calculated Asix alarm number,

*Port* - serial port name,

*Baud Rate* - baud rate from 9600 - 115200 Bd,

*Control Variables*

- variable used to show the communication status with megaMuz2:

0 - transmission o.k.

1 - no communication/communication error

2 - disabling the support for megaMuz2,

- variable used to control the communication status with megaMuz2:

## Communication Drivers

- 1 - request to disable the support for megaMuz2,
- 0 - enabling the support for megaMuz2.

### *Time Synchronization*

- time synchronization period with a megaMuz2 device (in seconds). 0 means no synchronization (default value).

Note:

a/ control variables must belong to a channel of NONE type.

The transmission is carried out without parity (in megaMuz2 communication parameters must be set PARZ. = NO).

Below follows an example of KSR\_67 channel declaration, supporting a megaMuz2 device of the network number 5, PR type, connected to the COM12 serial port, with alarm offset 200 and baud rate 38400. The channel status is indicated by MuzStat05 variable, the channel is controlled by MuzCtrl05 variable, time synchronization is carried out every 60 seconds:

*Device Number:* 5

*Device Type:* 4

*Alarms Offset:* 200

*Port:* COM12

*Baud Rate:* 38400

variable used to show the communication status with megaMuz: MuzStat05

variable used to control the communication status with megaMuz: MuzCtrl05

*Time Synchronization:* 60

## Declaration of Variables

The driver provides the following variable types of the MODBUS protocol:

CS - binary value	(write),
HR -16-bit register	(read)
HRF - 32-bit register of FLOAT type	(read)
HRL - 32-bit register of DWORD/LONG type	(read)

and the driver's internal variables:

RS - cause for triggering the disturbance recorder	(read)
RQ - request for disturbance recorder readout	(write)
RT - time of triggering the disturbance recorder	(read)
MN - network number of the device from which the disturbance recorder is/was read	(read)
RN - number of currently/recently read disturbance recorder	(read)
RR - status of currently/recently read disturbance recorder	(read)
SN - number of recently read disturbance recorder sample	(read)
RSKR - cause for triggering the criterion recorder	(read)
RQKR - request for criterion recorder readout	(write)

RTKR - time of triggering the criterion recorder (read)  
 MNKR - network number of the device from which the criterion recorder is/was read (read)  
 RNKR - number of currently/recently read criterion recorder (read)  
 RRKR - status of currently/recently read criterion recorder (read)  
 SNKR - number of recently read criterion recorder sample (read)

The address of a variable has the following syntax:

*Type[Index]*

where:

*Type* - the variable type,  
*Index* - index of a variable within the 'Type' variable type.

NOTE:

1. for the following types no indexes need to be provided: MN, RN, RR, RQ, SN, MNKR, RNKR, RRKR, RQKR i SNKR;
2. in case of RS, RT, RSKR, RTKR types the index range is dependant on the number of recorders defined in the device; the recorders are numbered starting from 1;
3. the parameter of request for recorder readout (RQ, RQKR type) is the recorder number; the recorders are numbered according to the time of liberation, starting from 1; the parameter set to 0 means a request to terminate the currently conducted recorder readout;
4. in order to readout the time of triggering the recorder, the DATATIME\_MUZ conversion function should be used; as a result, a number of DWORD type will be converted into a string of characters of the following format:

yyyy-mm-dd hh:nn:ss.zzz

Examples of variable declarations:

ZM\_01, RMS current I1, HRL3, KLR, 1, 1, NIC\_DW  
 ZM\_02, RMS voltage U1, HRL27,KLR, 1, 1, NIC\_DW  
 ZM\_03, protection 1 excitement, HRL95,KLR, 1, 1, NIC\_DW  
 ZM\_04, change of settings bank to bank1, CS17, KLR,1,1, NIC  
 ZM\_05, time of triggering the disturbance recorder 1, RT1, KLR, 1,1, DATAZAS\_MUZ  
  
 ZM\_06, request for disturbance recorder readout, RQ, KLR,1,1,NIC  
 ZM\_07, number of the disturbance recorder being read, RN, KLR,1,1,NIC  
 ZM\_08, status of disturbance recorder readout, RR, KLR,1,1,NIC  
 ZM\_09, number of sample read from the disturbance recorder, SN, KLR\_01,1,1,NIC  
  
 ZM\_10, request for reading the criterion recorder, RQKR, KLR,1,1,NIC  
 ZM\_11, number of the criterion recorder being read, RNKR, KLR,1,1,NIC  
 ZM\_12, status of criterion recorder readout, RRKR, KLR,1,1,NIC  
 ZM\_13, number of sample read from the criterion recorder, SNKR, KLR\_01,1,1,NIC

## Disturbance Recording

The driver provides for automatic disturbance recording in the AsLogger database. Review of the recorded disturbances is possible only with the use of the AsLogger application.

In order to activate the disturbance log mode, the Registration of Disturbances in AsLogger Database option (in the Driver parameters - AsLogger tab) must be declared, the option specifying:

- MSSQL server name,
- database name,

in which the driver will store the disturbances read.

Automatic disturbance recording is not provided by default - each time it must be initiated by the Asix system operator by the execution of control with the use of a RQ (disturbance recorder) or RQKR (criteria recorder) type variable. The control value is the number of the recorder, whose content must be read by megaMuz and recorded in the AsLogger database. Control value of 1 means that you should read the newest recorder, the k value means that you should read the k-th recorder, according to the sequence of recording. The control value set to 0 means a request to terminate the currently conducted recorder readout.

During recorder readout it is possible to view the readout status. To this end, the driver's internal variables of the below symbolic addresses are used:

- a/ MN, MNKR - number of the device in MODBUS network,
- b/ RN, RNKR - number of the recorder being read,
- c/ RR, RNKR - recorder readout status:

0 - disabled or completed o.k.

1 - readout in progress

2 - readout aborted due to an error

- d/ SN, SNKR - number of recently read recorder sample.

Disturbance recording is carried out in line with registration plans. The names of registration plans can be:

a/ defined by the user,

b/ generated automatically by the driver.

While recording, the driver stores in the AsLogger database the following parameters:

a/ all parameters stored in a recorder log, referred to in protection protocol documentation,

b/ it creates plan archives,

c/ it determines the maximum and minimum values for currents and voltages in the series recorded - if necessary, the values can be further adjusted while recording subsequent series within a given plan.

- **Defining the names of registration plans by the user**

The user can define the names of registration plans in text files with '.def' extension. To this end, for each transmission channel (ASMEN channel) a separate file must be created, to which the definitions of registration plan names for disturbances recorded in a given channel will be entered. The files including the definitions of registration plan names should be stored to a common directory. The name of the directory should be stored in the location **Directory with Files Containing Registration Plan Names** in the Driver parameters - AsLogger tab.

The syntax of the definition of a registration plan name stored in a '.def' file has the following form:

*channel, type, cause, table [, comments]*

where:

<i>channel</i>	- name of the ASMEN channel,
<i>type</i>	- recorder type (k-criteria, z-disturbance),
<i>cause</i>	- cause for triggering the recorder (as specified in the description of the security protocol),
<i>table</i>	- registration plan name assigned to a disturbance,
<i>comments</i>	- comment;

**Example:**

Defining the names of registration plans for the disturbance recorder:

- operation of fault protection device (cause for triggering No. 2)
- closing switch (cause for triggering No. 31)

in the ASMEN channel of the name KTR\_06:

KTR\_06, Z, 2, Pole06ZadzZabezpZwarciowego, operation of fault protection device  
KTR\_06, Z, 31, Pole06ZamkWylacznika, closing switch

- **Default names of registration plans**

If the user has not defined his own names of registration plans, the driver will on its own create the name of a registration plan, based on the following syntax:

For disturbance recorder:

NrxxxpyyyReason

where:

xxx	- network number of the megaMuz device,
yyy	- COM port number,
Reason	- abbreviation identifying the cause for triggering the recorder.

For criteria recorder:

NrxxxpkyyyReason

where:

- xxx - network number of the megaMuz device,
- yyy - COM port number,
- Reason - abbreviation identifying the cause for triggering the recorder.

**Example:**

For a disturbance caused by triggering the user's logic recorder, for the megaMuz2 device number 2 in the MODBUS network and communication via COM12 port, a registration plan of the following name will be created:

Nr002pz012wyzw\_rej\_od\_log\_użytk

## Event Signalling

Event signalling is checked cyclically, check frequency is defined by the item Event Check Period (Driver Parameters tab). The events signalled in megaMuz2 are converted into respective alarm numbers in the Asix system. The conversion is made based on mapping defined for each megaMuz2 type in the AsixAlarmNumberVerxx.xls file and the offset specified in the ASMEN channel declaration. Two types of Asix alarms are available (according to event type):

- a/ type 01H - int and word values (for events marked with (\*1) text describing the stages is transferred instead of the word)
- b/ type 02H - int and word values
- c/ type 04H - int and word values
- d/ type 08H - int and word values (for events marked with (\*1) the word value includes an identifier of the event cause)
- e/ type 11H - word value
- f/ type 12H - word value
- g/ type 14H - word value
- h/ type 21H (\*2) (\*3) - switch number, identifier of the control method
- i/ type 21H (\*2) - switch number
- j/ type 21H (\*3) - identifier of the control method
- k/ type 21H - word value
- l/ type 52H - word value
- ł/ type 54H - word value
- m/ type 92H - word value
- n/ type 94H - word value

**Example 1:**

The event Closing switch from controller (event of 21H type with the code 161) is mapped into Asix alarm No. 190. If an alarm offset of 400 is specified in the ASMEN channel declaration, then the alarm definition in the Asix alarm definitions file will have the following form (400 + 190 = 590):

**590, a1, closing switch from controller**

## Driver Parameters

The MegaMuz2 driver parameters are declared in the Current data module in the **MegaMuz2 / Driver Parameters tab...** of the channel operating in line with the MegaMuz2 driver protocol.

The parameters present in the Driver Parameters tabs refer to all the channels with the same driver declared. If multiple channels operate based on the same driver, the parameters set in the Driver Parameters tabs in the definition of one of these channels are automatically displayed in the Driver Parameters tabs in the definitions of the other channels.

(The parameters present in the Channel parameters tabs are applicable only to the specific channel.)

Driver parameters:

- creating a log file,
- log file size,
- log of telegrams,
- disturbances log file,
- number of requiry repetitions,
- response timeout,
- timeout between characters when response receiving,
- delay after no response,
- disturbances recording in AsLogger database,
- directory with files of recording (in AsLogger database) plan definition,

The names of the items related with the log file refer to the convention used in other ASMEN drivers.

### **MegaMuz2 / Driver Parameters - Diagnostics** tab:

#### **Log File**

Purpose - the text log file to which driver status messages are stored is used for diagnostic purposes.

The default value - by default, the log file is not created.

Parameter:

*file name*

#### **Log File Size**

Purpose - this option is used to determine the log file size defined using the Log file option.

Parameter:

*number* - the size of the log file in MB.

The default value - by default, the log file size is 10 MB.

**Log of Telegrams**

Purpose: - this option allows writing the contents of messages communicated between the driver and the controllers to the log file (declared using the Log File item) The option in question should only be used during the Asix system start-up.

The default value: - by default, this option is disabled.

**Disturbances Log File**

Purpose: the text log file to which messages about read disturbances are stored is used for diagnostic purposes.

The default value: by default, the log file is not created.

Option value:  
*log file name*

**MegaMuz2 / Driver Parameters** tab:

**Number of Repetitions**

Purpose: This option is used to define the maximum number of repetitions in case of a transmission error.

The default value: By default the value of the item is set to 3.

Option value:  
*number* - maximum number of repetitions in case of a transmission error.

**Response Timeout**

Purpose: this option specifies the maximum time throughout which the driver waits for a response from megaMuz. This option is of global character and its value can be overridden by individual channel settings.

The default value: the default response timeout value is 1000 milliseconds.

Option value:  
*number* - time out in milliseconds.

**Char Timeout**

Purpose: This option specifies the maximum time between response characters from megaMuz. This option is of global character and its value can be overridden by individual channel settings.

The default value: the default response character timeout value is 100 milliseconds.

Option value:  
*number* - time out in milliseconds.

**Event Check Period**

Purpose: - this option specifies the interval (in seconds) before reading the event log of a subsequent device connected to the same serial port supported by the driver.

The default value: - the default value is 10 seconds.

Option value:

*number*

**No Receive Delay**

Purpose: - time after which requests to the driver will be blocked after last request finished with no answer.

The default value: - 0 - no blocking.

Option value:

*number*

**MegaMuz2 / Driver Parameters - AsLogger tab:**

**Registration of disturbances in AsLogger Database**

Purpose: This option switches the driver into disturbance registration mode. The option defines the server name and the database name to which disturbance samples will be stored.

The default value: by default disturbances are not registered.

Option value:

*ServerName* - MSSQL server name

*DatabaseName* - database name on the ServerName

**Directory with Files Containing Registration Plan Names**

Purpose: this option specifies a complete access path to the directory in which the driver will search for files including the definitions of registration plan names.

The default value: -.

Option value:

*path* - complete path to the file directory including the definitions of registration plans.

## Channel Parameters

The channel is configured using options present in the Channel parameters tabs.

- creating a log file,
- the log file size,
- log of telegrams,
- response timeout,
- response character timeout,
- without events,
- no receive delay,
- switch names.

**MegaMuz / Channel Parameters - Diagnostics** tab:

**Log File**

Purpose - the text log file to which channel status messages are stored is used for diagnostic purposes.  
The default value - by default, the driver log file is used.  
Parameter:  
file name

**Log File Size**

Purpose - this option is used to determine the channel log file size defined using the Log file option.  
Parameter:  
*number* - the size of the log file in MB.  
The default value - by default, the log file size defined for the driver is used.

**Log of Telegrams**

Purpose - this option allows writing the contents of messages communicated in the channel to the channel log file (declared using the Log File option). The option in question should only be used during the Asix system start-up.  
The default value - by default, the setting defined for the driver is used.

**MegaMuz / Channel Parameters 2** tab:

**Response Timeout**

Purpose: This option specifies the maximum time throughout which the driver waits for a response from megaMuz. The option value overrides the settings in the driver options.  
The default value: the default response timeout value is 1000 milliseconds.  
Option value:  
*number* - time out in milliseconds.

**Char Timeout**

Purpose: This option specifies the maximum time between response characters from megaMuz. The option value overrides the settings in the driver options.  
The default value: the default response character timeout value is 100 milliseconds.  
Option value:  
*number* - time out in milliseconds.

**Without Events**

Purpose: this option is used to disable readout of the event log in a channel.

The default value: by default, readout of the event log is enabled.

Option value:  
YES/NO

 **No Receive Delay**

Purpose - time after which requests to the driver will be blocked after last request finished with no answer.

The default value - 0 - no blocking.

Option value:  
*number*

**MegaMuz / Channel Parameters - Switch Names** tab: **Switch 1....7 name**

Purpose: Position is used to specify the names of switches that will be forwarded to the alarm system. The name of the switch can not be longer than 3 characters. You can define a name for max 7 switches numbered from 1 to 7.

The default value: default names of switches: 1...7.

Option value:  
*text* - 3-character name of a switch of a defined number.

**EXAMPLE**

Driver parameterization:

Log File =c:\tmp\muz.log  
Disturbances Log File=c:\tmp\zaklocenia.log  
Log File Size =30  
Log of Telegrams =YES  
Registration of disturbances in AsLogger Database =SERWER\_RO6, BAZA\_RO6  
Directory with Files Containing Registration Plan Names =c:\tmp\DefZaklocen

Channel parameterization:

[CHANNEL]  
Log File =c:\tmp\channelx.log  
Log File Size =30  
Log of Telegrams =YES  
Char Timeout =60  
Response Timeout=1500  
Switch 1 name: L1  
Switch 2 name: L2

## Advanced Channel Parameters

The parameters of a channel are declared in the **Advanced** tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:  
YES / NO

computer name - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

computer name - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the **Advanced 2** tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

***Data Validity Period***

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value

- by default, no time for data validity is set.

## 1.58 MegaMuz\_TCPIP - Driver for Communication with Microprocessor Protection Devices of MegaMuz Type by JM-Tronik through Ethernet

### Driver Use

MegaMuz\_TCPIP driver is used to exchange data between the Asix system and microprocessor protection devices of MegaMuz type developed by JM-Tronik, Warsaw. Communication is carried out via Ethernet.

The driver performs the following functions:

- readout of current status and measurements,
- readout of settings,
- readout of control settings,
- readout of events and reporting them to the ASIX alarm system,
- disturbance record readout and registering in database in the format of AsLogger developed by ASKOM Sp. z o.o,
- controlling outputs.

Drajwer obsługuje następujące typy megaMuz:

CR, CR\_87

LR

LZ, LZ\_87

PR, PR\_87

SR, SR\_87

TR, TR\_87

NOTE:

Support of other protection devices of MegaMuz type requires changes in the driver code.

Parameterization of MegaMuz\_TCPIP driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the MegaMuz\_TCPIP driver requires adding a channel with the following parameters to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: MegaMuz

**MegaMuz/Channel Parameters** tab:

*Device Number* - network device number,

*Device Type* - megaMuz type. Identifiers of supported types:

- 1 - CR
- 2 - LR
- 3 - LZ
- 4 - PR
- 5 - SR
- 6 - TR
- 7 - CR\_87
- 8 - LZ\_87
- 9 - PR\_87
- 10 - SR\_87
- 11 - TR\_87

*Alarms Offset* - offset added to the calculated Asix alarm number,

*Number of the TCP Port...* - number of TCP port in megaMuzie (or of the gateway the device is connected to),

*IP Address of the Device...* - IP address of megaMuza (or of the gateway the device is connected to),

#### *Control Variables*

- variable used to show the communication status with megaMuz:

- 0 - transmission o.k.
- 1 - no communication/communication error
- 2 - disabling the support for megaMuz,

- variable used to control the communication status with megaMuz:

- 1 - request to disable the support for megaMuz,
- 0 - enabling the support for megaMuz.

#### *Time Synchronization*

- time synchronization period with a megaMuz device (in seconds). 0 means no synchronization (default value).

#### **Note:**

- control variables must belong to a channel of NONE type.

Below follows an example of KPR channel declaration, supporting a megaMuz device of the network number 5, PR type, communication is realized by the NPort port (port TCP 4004, adres IP 10.10.105.211), with alarm offset 200. The channel status is indicated by MuzStat05 variable, the channel is controlled by MuzCtrl05 variable, time synchronization is carried out every 60 seconds:

*Device Number:* 5

*Device Type:* 4

*Alarms Offset:* 200

*Number of the TCP Port...:* 4004

*IP Address of the Device:* 10.10.105.211

variable used to show the communication status with megaMuz: MuzStat05

variable used to control the communication status with megaMuz: MuzCtrl05

*Time Synchronization:* 60

## Declaration of Variables

The driver provides the following variable types of the MODBUS protocol:

- CS - binary value (write),
- HR - 16-bit register (read)
- HRL - 2 registers including a number of DWORD type (read)
- HRFL - 2 registers including a number of FLOAT type (read)

and the driver's internal variables:

- MN - network number of the megaMuz device from which the recorder is/was read (read)
- RN - number of currently/recently read recorder (read)
- RQ - request for recorder readout (write)
- PR - status of currently/recently read recorder (readout)
- RS - cause for triggering the recorder (read)
- RT - time of triggering the recorder (read)
- SN - number of recently read recorder sample (read)
- UDC - used to execute the MODBUS RTU protocol commands defined by the user, available only for keys WAUS, WAUW, WDUW or WDUN. The value of a UDC type variable is a string of ASCII characters, whose maximum length is 521 bytes. While writing, it is the content of a command defined by the user; while reading, it is the content of a response to a recently sent user's command. (read/write)

The address of a variable has the following syntax:

*Type[Index]*

where:

- Type* - the variable type,
- Index* - index of a variable within the 'Type' variable type.

### NOTE:

1. for the following types no indexes need to be provided: MN, RN, RR, RQ and SN,
2. in case of RS, RT types the index range is dependant on the number of recorders defined in the device,
3. the parameter of request for recorder readout (RQ type) is the recorder number. The recorders are numbered starting from 1. The parameter set to 0 means a request to terminate the currently conducted recorder readout,
4. the ranges for HR types are dependant on the device type,
5. in order to readout the time of triggering the recorder, the DATACZAS\_MUZ conversion function should be used; as a result, a number of DWORD type will be converted into a string of characters of the following format:

yyyy-mm-dd hh:nn:ss.zzz

Examples of variable declarations (variable values originate from a device of LR type, available through the channel named KLR\_01):

- ZM\_01, value I1, HRFL28, KLR\_01, 1, 1, NOTHING\_FP
- ZM\_02, protection excitement factor 1, HR19, KLR\_01, 1, 1, NOTHING
- ZM\_03, number of non-read events, HR27, KLR\_01, 1, 1, NOTHING

ZM\_04, value U1, HRFL46, KLR\_01, 1, 1, NOTHING\_FP  
 ZM\_06, time of triggering the recorder 1, RT1, KLR\_01, 1,1, DATAZAS\_MUZ

Examples of variable declarations without parameters (variable values originate from a device of *LR* type, available through the channel named *KLR\_01*):

ZM\_07, request for recorder readout, RQ, KLR\_01,1,1,NOTHING  
 ZM\_08, number of the recorder being read, RN, KLR\_01,1,1,NOTHING  
 ZM\_09, recorder readout status, RR, KLR\_01,1,1,NOTHING  
 ZM\_10, number of read sample, SN, KLR\_01,1,1,NOTHING  
 ZM\_11, number of the device being read, MN, KLR\_01,1,1,NOTHING

## Disturbance Recording

The driver provides for automatic disturbance recording in the AsLogger database. Review of the recorded disturbances is possible only with the use of the AsLogger application.

In order to activate the disturbance log mode, the *Disturbance recording in AsLogger database* option (in the *Driver parameters tab*) must be declared, the option specifying:

- MSSQL server name,
- database name,

in which the driver will store the disturbances read.

Automatic disturbance recording is not provided by default - each time it must be initiated by the Asix system operator by the execution of control with the use of a RQ type variable. The control value is the number of the recorder, whose content must be read by megaMuz and recorded in the AsLogger database. The control value set to 0 means a request to terminate the currently conducted recorder readout.

During recorder readout it is possible to view the readout status. To this end, the driver's internal variables of the below symbolic addresses are used:

- a/ *MN* - number of the megaMuz device being read,
- b/ *RN* - number of the recorder being read,
- c/ *RR* - recorder readout status:

- 0 - disabled or completed o.k.
- 1 - readout in progress
- 2 - readout aborted due to an error

- d/ *SN* - number of recently read recorder sample.

Disturbance recording is carried out in line with registration plans. The names of registration plans can be:

- a/ defined by the user,
- b/ generated automatically by the driver.

While recording, the driver stores in the AsLogger database the following disturbance parameters:

- a/ complete set of the plan's measurement points, including:
  - currents : I0, I1, I2, I3 (on the secondary side)
  - voltages : I0, I1, I2, I3 (on the secondary side)
  - status of inputs : WE01 ... WE24
  - status of outputs : WY01 ... WY24
  - excitement status of protections 1 : PZ01 ... PZ16
  - tripping of protections 1 : ZZ01 ... ZZ16

The set of parameters and number of the plan's measurement points referring to excitement and tripping of protections are specific for each megaMuz type,

b/ it creates plan archives,

c/ it determines the maximum and minimum values for currents and voltages in the disturbance recorded - if necessary, the values can be further adjusted while reading subsequent disturbances within a given plan.

- **Defining the names of registration plans by the user**

The user can define the names of registration plans in text files with '.def' extension. To this end, for each transmission channel (ASMEN channel) a separate file must be created, to which the definitions of registration plan names for disturbances recorded in a given channel will be entered. The files including the definitions of registration plan names should be stored to a common directory. The name of the directory should be stored in the location *Directory with Files Containing Registration Plan Names* in the MegaMuz\_TCPIP parameters.

The syntax of the definition of a registration plan name stored in a '.def' file has the following form:

*channel, cause, table [, comments]*

where:

<i>channel</i>	- name of the ASMEN channel,
<i>cause</i>	- cause for triggering the recorder (number between 0 and 15),
<i>table</i>	- registration plan name assigned to a disturbance,
<i>comments</i>	- comment

- **Default names of registration plans**

If the user has not defined his own names of registration plans, the driver will on its own create the name of a registration plan, based on the following syntax:

NrxxxpyyyPowod

where:

xxx	- network number of the megaMuz device,
yyy	- COM port number,
Reason	- abbreviation identifying the cause for triggering the recorder.

**Example:**

For a disturbance caused by triggering the user's logic recorder, for the megaMuz device number 2 on COM5 a registration plan of the following name will be created:

Nr002P005logika

## Event Signalling

Event signalling is checked cyclically, check frequency is defined by the item *Event Check Period*. The events signalled in megaMuz are converted into respective alarm numbers in the Asix system. The conversion is made based on mapping defined for each megaMuz type and the offset specified in the ASMEN channel declaration. Two types of Asix alarms are available:

a/ text only

b/ text and one numerical parameter

Events of 0 type are converted into Asix alarms of the following form: text + one numerical parameter (of int type) + text parameter (max. 3 characters), including phase numbers.

MegaMuz events of type No. 2 are converted into Asix alarms of *text* type.

### Example 1

The event *Closing switch from controller* in megaMuz of LR type (event of *text* type) is mapped into Asix alarm No. 34. If an alarm offset of 400 is specified in the ASMEN channel declaration, then the alarm definition in the Asix alarm definitions file will have the following form (400 + 34 = 434):

**434, al, closing switch from controller**

### Example 2

The event *Operation of fault protection device* in megaMuz of LR type (event of type *text and two parameters: numerical and text*) is mapped into Asix alarm No. 0. If an alarm offset of 100 is specified in the ASMEN channel declaration, then the alarm definition in the Asix alarm definitions file will have the following form (100 + 0 = 100):

**100, al, operation of fault protection device %f, of phase %s**

Mapping of individual megaMuz event types onto Asix alarm numbers is stored in the *NumeryAlarmowAsixa.xls* file.

## Driver Configuration

The MegaMuz\_TCPIP driver parameters are declared in the *Current data* module in the **MegaMuz\_TCPIP / Driver parameters** tab... of the channel operating in line with the MegaMuz\_TCPIP driver protocol.

The parameters present in the *Driver Parameters* tabs refer to all the channels with the same driver declared. If multiple channels operate based on the same driver, the parameters set in the *Driver Parameters* tabs in the definition of one of these channels are automatically displayed in the *Driver Parameters* tabs in the definitions of the other channels.

(The parameters present in the *Channel parameters* tabs are applicable only to the specific channel.)

Driver parameters:

- creating a log file,
- creating a disturbance log file,
- the log file size,
- log of telegrams,
- number of repetitions,
- response timeout,
- event signalling check period,
- disturbance recording in the AsLogger database.
- file directory including the definitions of registration plans.

The names of the items related with the log file refer to the convention used in other ASMEN drivers.

**MegaMuz\_TCPIP / Driver Parameters - Diagnostics** tab:

**Log File**

Purpose - the text log file to which driver status messages are stored is used for diagnostic purposes.

The default value - by default, the log file is not created.

Parameter:  
*file name*

**Log File Size**

Purpose - this option is used to determine the log file size defined using the *Log file* option.

Parameter:  
*number* - the size of the log file in MB.

The default value - by default, the log file size is 10 MB.

**Log of Telegrams**

Purpose - this option allows writing the contents of messages communicated between the driver and the controllers to the log file (declared using the *Log File* item) The option in question should only be used during the Asix system start-up.

The default value - by default, this option is disabled.

**Disturbances Log File**

Purpose: the text log file to which messages about read disturbances are stored is used for diagnostic purposes.

The default value: by default, the log file is not created.

Option value:  
*log file name*

**MegaMuz\_TCPIP / Driver Parameters** tab:

**Number of Repetitions**

Purpose: This option is used to define the maximum number of repetitions in case of a transmission error.

The default value: By default the value of the item is set to 3.

Option value:  
*number* - maximum number of repetitions in case of a transmission error.

**Response Timeout**

Purpose: this option specifies the maximum time throughout which the driver waits for a response from megaMuz. This option is of global character and its value can be overridden by individual channel settings.

The default value: the default response timeout value is 1000 milliseconds.

Option value:

*number* - time out in milliseconds.

#### **Event Check Period**

Purpose - this option specifies the interval (in seconds) before reading the event log of a subsequent device connected to the same serial port supported by the driver.

The default value - the default value is 10 seconds.

Option value:

*number*

#### **Registration of disturbances in AsLogger Database**

Purpose: This option switches the driver into disturbance registration mode. The option defines the server name and the database name to which disturbance samples will be stored.

The default value: by default disturbances are not registered.

Option value:

*ServerName*

- MSSQL server name

*DatabaseName*

- database name on the *ServerName*

#### **Directory with Files Containing Registration Plan Names**

Purpose: this option specifies a complete access path to the directory in which the driver will search for files including the definitions of registration plan names.

The default value: -.

Option value:

*path*

- complete path to the file directory including the definitions of registration plans.

## Channel Parameterization

The channel is configured using options present in the **Channel parameters** tabs.

- creating a log file,
- the log file size,
- log of telegrams,
- response timeout,
- without events,
- without register,

**MegaMuz\_TCPIP / Channel Parameters - Diagnostics** tab:

#### **Log File**

Purpose - the text log file to which channel status messages are stored is used for diagnostic purposes.

The default value - by default, the driver log file is used.

Parameter:  
*file name*

**Log File Size**

Purpose - this option is used to determine the channel log file size defined using the *Log file* option.

Parameter:  
*number* - the size of the log file in MB.

The default value - by default, the log file size defined for the driver is used.

**Log of Telegrams**

Purpose - this option allows writing the contents of messages communicated in the channel to the channel log file (declared using the *Log File* option). The option in question should only be used during the Asix system start-up.

The default value - by default, the setting defined for the driver is used.

**MegaMuz\_TCPIP / Channel Parameters 2** tab:

**Response Timeout**

Purpose: This option specifies the maximum time throughout which the driver waits for a response from megaMuz. The option value overrides the settings in the driver options.

The default value: the default response timeout value is 1000 milliseconds.

Option value:  
*number* - time out in milliseconds.

**Without Events**

Purpose: this option is used to disable readout of the event log in a channel.

The default value: by default, readout of the event log is enabled.

Option value:  
YES/NO

**Without Registrar**

Purpose: this option is used to disable readout of the disturbance log file in a channel.

The default value: by default, readout of the disturbance log file is enabled.

Option value:  
YES/NO

**EXAMPLE**

Driver parameterization

*Log File* =c:\tmp\muz.log

*Disturbances Log File* =c:\tmp\disturbances.log

*Log File Size* =30

*Log of Telegrams* =YES

*Registration of Disturbances in AsLogger Database* =SERWER\_RO6, BAZA\_RO6

*Directory with Files Containing Registration Plan Names* =c:\tmp\DefZaklocen

**EXAMPLE**

Driver parameterization

[CHANNELX]

*Log File* =c:\tmp\channelx.log

*Log File Size* =20

*Log of Telegrams* =YES

*Response Timeout* =1500

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose

- this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose

- with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

## Communication Drivers

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose

- with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value

- NO.

### **Data Validity Period**

Purpose

- this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

The default value

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

- by default, no time for data validity is set.

## 1.59 MegaMuz2\_TCPIP - Driver for Communication with Microprocessor Protection Devices of MegaMuz2 Type by JM-Tronik through Ethernet

### Driver Use

MegaMuz2\_TCPIP driver is used to exchange data between the Asix system and microprocessor protection devices of MegaMuz2 type developed by JM-Tronik, Warsaw. Communication is carried out in MODBUS RTU mode on TCPIP via Ethernet and port 10502.

The driver performs the following functions:

- readout of current status and measurements,
- readout of events and reporting them to the ASIX alarm system,
- disturbance record readout and registering in database in the format of AsLogger developed by ASKOM Sp. z o.o;

The driver supports the following protection devices of MegaMuz type:

- 01.12

NOTE:

Support of other protection devices of MegaMuz2 type requires changes in the driver code.

Parameterization of MegaMuz2\_TCPIP driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the MegaMuz2\_TCPIP driver requires adding a channel with the following parameters to the Current data module:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: MegaMuz2\_TCPIP

**MegaMuz2\_TCPIP /Channel Parameters** tab:

*Device Number* - network device number,

*Device Type* - megaMuz type. Identifiers of supported types:

01\_12

*IP Address of the Device* - IP address of megaMuza (or of the gateway the device is connected to),

*Alarms Offset* - offset added to the calculated Asix alarm number,

*Control Variables*

- variable used to show the communication status with megaMuz:
  - 0 - transmission o.k.
  - 1 - no communication/communication error
  - 2 - disabling the support for megaMuz,
- variable used to control the communication status with megaMuz:
  - 1 - request to disable the support for megaMuz,
  - 0 - enabling the support for megaMuz.

*Time Synchronization*

- time synchronization period with a megaMuz device (in seconds). 0 means no synchronization (default value).

**IP Address of the Computer** - IP address of the computer used to connect to the device by the channel.

Note:

- control variables must belong to a channel of NONE type.

Below follows an example of KSR\_67 channel declaration, supporting a megaMuz2 device of the network number 5, address IP 10.10.12.5), with alarm offset 200. The channel status is indicated by MuzStat05 variable, the channel is controlled by MuzCtrl05 variable, time synchronization is carried out every 60 seconds:

*Device Number:* 5

*Alarms Offset:* 200

*IP Address of the Device:* 10.10.12.5

*Variable used to show the communication status with megaMuz:* MuzStat05

*Variable used to control the communication status with megaMuz:* MuzCtrl05

*Time Synchronization:* 60

## Declaration of Variables

The driver provides the following variable types of the MODBUS protocol:

CS	- binary value	(write),
HR	-16-bit register	(read)
HRF	- 32-bit register of FLOAT type	(read)
HRL	- 32-bit register of DWORD/LONG type	(read)

and the driver's internal variables:

RS	- cause for triggering the disturbance recorder	(read)
RQ	- request for disturbance recorder readout	(write)
RT	- time of triggering the disturbance recorder	(read)
MN	- network number of the device from which the disturbance recorder is/was read	(read)
RN	- number of currently/recently read disturbance recorder	(read)
RR	- status of currently/recently read disturbance recorder	(read)
SN	- number of recently read disturbance recorder sample	(read)

RSKR	- cause for triggering the criterion recorder	(read)
RQKR	- request for criterion recorder readout	(write)
RTKR	- time of triggering the criterion recorder	(read)
MNKR	- network number of the device from which the criterion recorder is/was read	(read)
RNKR	- number of currently/recently read criterion recorder	(read)
RRKR	- status of currently/recently read criterion recorder	(read)
SNKR	- number of recently read criterion recorder sample	(read)

The address of a variable has the following syntax:

*Type[Index]*

where:

*Type* - the variable type,  
*Index* - index of a variable within the 'Type' variable type.

NOTE:

1. for the following types no indexes need to be provided: MN, RN, RR, RQ, SN, MNKR, RNKR, RRRKR, RQKR i SNKR;
2. in case of RS, RT, RSKR, RTKR types the index range is dependant on the number of recorders defined in the device; the recorders are numbered starting from 1;
3. the parameter of request for recorder readout (RQ, RQKR type) is the recorder number; the recorders are numbered according to the time of liberation, starting from 1; the parameter set to 0 means a request to terminate the currently conducted recorder readout;
4. in order to readout the time of triggering the recorder, the DATETIME\_MUZ conversion function should be used; as a result, a number of DWORD type will be converted into a string of characters of the following format:

yyyy-mm-dd hh:nn:ss.zzz

Examples of variable declarations:

ZM_01, RMS current I1,	HRL3, KLR, 1, 1, NIC_DW
ZM_02, RMS voltage U1,	HRL27,KLR, 1, 1, NIC_DW
ZM_03, protection 1 excitement,	HRL95,KLR, 1, 1, NIC_DW
ZM_04, change of settings bank to bank1,	CS17, KLR,1,1, NIC
ZM_05, time of triggering the disturbance recorder 1,	RT1, KLR, 1,1, DATACZAS_MUZ
ZM_06, request for disturbance recorder readout,	RQ, KLR,1,1,NIC
ZM_07, number of the disturbance recorder being read,	RN, KLR,1,1,NIC
ZM_08, status of disturbance recorder readout,	RR, KLR,1,1,NIC
ZM_09, number of sample read from the disturbance recorder, SN, KLR_01,1,1,NIC	
ZM_10, request for reading the criterion recorder,	RQKR, KLR,1,1,NIC
ZM_11, number of the criterion recorder being read,	RNKR, KLR,1,1,NIC
ZM_12, status of criterion recorder readout,	RRKR, KLR,1,1,NIC

ZM\_13, number of sample read from the criterion recorder, SNKR, KLR\_01,1,1,NIC  
ZM\_13, nr odczytanej próbki z rejestratora kryterialnego, SNKR, KLR\_01,1,1,NIC

## Disturbance Recording

The driver provides for automatic disturbance recording in the AsLogger database. Review of the recorded disturbances is possible only with the use of the AsLogger application.

In order to activate the disturbance log mode, the Disturbance recording in AsLogger database option (in the Driver parameters tab) must be declared, the option specifying:

- MSSQL server name,
- database name,

in which the driver will store the disturbances read.

Automatic disturbance recording is not provided by default - each time it must be initiated by the Asix system operator by the execution of control with the use of a RQ type variable. The control value is the number of the recorder, whose content must be read by megaMuz and recorded in the AsLogger database. The control value set to 0 means a request to terminate the currently conducted recorder readout.

During recorder readout it is possible to view the readout status. To this end, the driver's internal variables of the below symbolic addresses are used:

- a/ MN, MNKR - number of the device in MODBUS network,
- b/ RN, RNKR - number of the recorder being read,
- c/ RR, RNKR - recorder readout status:

- 0 - disabled or completed o.k.
- 1 - readout in progress
- 2 - readout aborted due to an error

- d/ SN, SNKR - number of recently read recorder sample.

Disturbance recording is carried out in line with registration plans. The names of registration plans can be:

- a/ defined by the user,
- b/ generated automatically by the driver.

While recording, the driver stores in the AsLogger database the following disturbance parameters:

- a/ all the parameters contained in the recorder record, described in the security protocol documentation,
- b/ creates an archive plan,
- c/ determines the maximum and minimum values for currents and voltages in registered series – they will be corrected during other series registration for a given plan.

- **Defining the names of registration plans by the user**

The user can define the names of registration plans in text files with '.def' extension. To this end, for each transmission channel (ASMEN channel) a separate file must be created, to which the definitions of registration plan names for disturbances recorded in a given channel will be entered. The files including the definitions of registration plan names should be stored to a common directory. The name of the directory should be stored in the location Directory with Files Containing Registration Plan Names in the Driver parameters - AsLogger tab.

The syntax of the definition of a registration plan name stored in a '.def' file has the following form:

*channel, type, cause, table [, comments]*

where:

<i>channel</i>	- name of the ASMEN channel,
<i>type</i>	- recorder type (k-criteria, z-disturbance),
<i>cause</i>	- cause for triggering the recorder (number between 0 and 15),
<i>table</i>	- registration plan name assigned to a disturbance,
<i>comments</i>	- comment

Example:

Defining the names of registration plans for the disturbance recorder:

- operation of fault protection device (cause for triggering No. 2)
- closing switch (cause for triggering No. 31)

in the ASMEN channel of the name KTR\_06:

KTR\_06, Z, 2, Pole06ZadzZabezpZwarcowego, operation of fault protection device

KTR\_06, Z, 31, Pole06ZamkWylacznika, closing switch

- **Default names of registration plans**

If the user has not defined his own names of registration plans, the driver will on its own create the name of a registration plan, based on the following syntax:

For disturbance recorder:

NrxxxpyyyReason

where:

xxx	- network number of the megaMuz device,
yyy	- COM port number,
Reason	- abbreviation identifying the cause for triggering the recorder.

For criteria recorder:

NrxxxpkyyyReason

where:

xxx - network number of the megaMuz device,  
yyy - COM port number,  
Reason - abbreviation identifying the cause for triggering the recorder.

**Example:**

For a disturbance caused by triggering the user's logic recorder, for the megaMuz2 device number 2 in the MODBUS network and communication via COM12 port, a registration plan of the following name will be created:

Nr002pz012wyzw\_rej\_od\_log\_uzytk

Event Signalling

Event signalling is checked cyclically, check frequency is defined by the item Event Check Period (Driver Parameters tab). The events signalled in megaMuz2 are converted into respective alarm numbers in the Asix system. The conversion is made based on mapping defined for each megaMuz2 type in the AsixAlarmNumberVerxx.xls file and the offset specified in the ASMEN channel declaration. Two types of Asix alarms are available (according to event type):

- a/ type 01H - *int* and *word values* (for events marked with (\*1) text describing the stages is transferred instead of the *word*)
- b/ type 02H - *int* and *word values*
- c/ type 04H - *int* and *word values*
- d/ type 08H - *int* and *word values* (for events marked with (\*1) the *word value* includes an identifier of the event cause)
- e/ type 11H - *word value*
- f/ type 12H - *word value*
- g/ type 14H - *word value*
- h/ type 21H (\*2) (\*3) - switch number, identifier of the control method
- i/ type 21H (\*2) - switch number
- j/ type 21H (\*3) - identifier of the control method
- k/ type 21H - *word value*
- l/ type 52H - *word value*
- ł/ type 54H - *word value*
- m/ type 92H - *word value*
- n/ type 94H - *word value*

Example 1:

The event *Closing switch from controller* (event of 21H type with the code 161) is mapped into Asix alarm No. 190. If an alarm offset of 400 is specified in the ASMEN channel declaration, then the alarm definition in the Asix alarm definitions file will have the following form (400 + 190 = 590):

**590, a1, closing switch from controller**

## Driver Configuration

The MegaMuz2\_TCPIP driver parameters are declared in the Current data module in the MegaMuz2\_TCPIP / Driver parameters... tab of the channel operating in line with the MegaMuz2\_TCPIP driver protocol.

The parameters present in the Driver Parameters tabs refer to all the channels with the same driver declared. If multiple channels operate based on the same driver, the parameters set in the Driver Parameters tabs in the definition of one of these channels are automatically displayed in the Driver Parameters tabs in the definitions of the other channels.

(The parameters present in the Channel parameters tabs are applicable only to the specific channel.)

Driver parameters:

- creating a log file,
- log file size,
- log of telegrams,
- disturbances log file,
- number of requiry repetitions,
- disturbances recording in AsLogger database,
- event check parity.

The names of the items related with the log file refer to the convention used in other ASMEN drivers.

**MegaMuz2\_TCPIP / Driver Parameters - Diagnostics** tab:

### **Log File**

Purpose - the text log file to which driver status messages are stored is used for diagnostic purposes.

The default value - by default, the log file is not created.

Parameter:

*file name*

### **Log File Size**

Purpose - this option is used to determine the log file size defined using the Log file option.

Parameter:

*number*

- the size of the log file in MB.

The default value

- by default, the log file size is 10 MB.

### **Log of Telegrams**

Purpose - this option allows writing the contents of messages communicated between the driver and the controllers to the log file (declared using the Log File item) The option in question should only be used during the Asix system start-up.

The default value

- by default, this option is disabled.

**Disturbances Log File**

Purpose: the text log file to which messages about read disturbances are stored is used for diagnostic purposes.

The default value: by default, the log file is not created.

Option value:  
*log file name*

**MegaMuz\_TCPIP / Driver Parameters** tab:

**Number of Repetitions**

Purpose: This option is used to define the maximum number of repetitions in case of a transmission error.

The default value: By default the value of the item is set to 3.

Option value:  
*number* - maximum number of repetitions in case of a transmission error.

**Event Check Period**

Purpose - this option specifies the interval (in seconds) before reading the event log of a subsequent device connected to the same serial port supported by the driver.

The default value - the default value is 10 seconds.

Option value:  
*number*

**Registration of disturbances in AsLogger Database**

Purpose: This option switches the driver into disturbance registration mode. The option defines the server name and the database name to which disturbance samples will be stored.

The default value: by default disturbances are not registered.

Option value:  
*ServerName* - MSSQL server name  
*DatabaseName* - database name on the ServerName

**Directory with Files Containing Registration Plan Names**

Purpose: this option specifies a complete access path to the directory in which the driver will search for files including the definitions of registration plan names.

The default value: -.

Option value:  
*path* - complete path to the file directory including the definitions of registration plans.

## Channel Parameterization

The channel is configured using options present in the Channel parameters tabs.

- creating a log file,
- the log file size,
- log of telegrams,
- port number of a computer,
- response timeout,
- response delay,
- without events,
- number of repetitions,
- max. number of registers,
- switch names.

### ***MegaMuz2\_TCPIP / Channel Parameters - Diagnostics*** tab:

#### **Log File**

Purpose - the text log file to which channel status messages are stored is used for diagnostic purposes.

The default value - by default, the driver log file is used.

Parameter:

*file name*

#### **Log File Size**

Purpose - this option is used to determine the channel log file size defined using the Log file option.

Parameter:

*number*

- the size of the log file in MB.

The default value - by default, the log file size defined for the driver is used.

#### **Log of Telegrams**

Purpose - this option allows writing the contents of messages communicated in the channel to the channel log file (declared using the Log File option). The option in question should only be used during the Asix system start-up.

The default value - by default, the setting defined for the driver is used.

### ***MegaMuz2\_TCPIP / Channel Parameters 2*** tab:

#### **Port**

Purpose: Port number of the computer.

The default value: 0 - no port

Option value:

*number*

**Response Timeout**

Purpose: This option specifies the maximum time throughout which the driver waits for a response from megaMuz. The option value overrides the settings in the driver options.

The default value: the default response timeout value is 1000 milliseconds.

Option value:  
*number* - time out in milliseconds.

**Response Delay**

Purpose: This option specifies delay for the response completion.

The default value: 20.

Option value:  
*number* - time out in milliseconds.

**Without Events**

Purpose: this option is used to disable readout of the event log in a channel.

The default value: by default, readout of the event log is enabled.

Option value:  
YES/NO

**Number of Repetitions**

Purpose: This option is used to define the maximum number of repetitions in case of a transmission error.

The default value: By default the value of the item is set to 3.

Option value:  
*number* - maximum number of repetitions in case of a transmission error.

**Max number of registers**

Purpose: This option is used to define the maximum number of registers asked by the driver in a single request.

The default value: 119.

Option value:  
*number*

**EXAMPLE**

Driver parameterization

Log File =c:\tmp\muz.log

Disturbances Log File =c:\tmp\disturbances.log

Log File Size =30

Log of Telegrams =YES

Registration of Disturbances in AsLogger Database =SERWER\_RO6, BAZA\_RO6

Directory with Files Containing Registration Plan Names =c:\tmp\DefZaklocen

**EXAMPLE**

Driver parameterization

[CHANNEL]

Log File =c:\tmp\channelx.log

Log File Size =20

Log o f Telegrams =YES

IP Address of the Computer = 10.10.112.8

Max number of registers = 53

## Advanced Channel Parameters

The parameters of a channel are declared in the **Advanced** tab:

**Remote Write Access**

Purpose

- this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose

- with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the **Advanced 2** tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.60 MELSECA - Driver of MITSUBISHI MELSEC-A PLC Protocol

### Driver Use

The MELSECA driver is used for data exchange between Asix computers and the A1SJ71C24-R2 communication processor of MITSUBISHI MELSEC-A PLCs. The transmission is executed by means of serial interfaces by using standard serial ports of an Asix system computer.

The cooperation of Asix with the controller by using the MELSECA protocol does not require any controller's program adaptation.

Parameterization of MELSECA driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the MELSECA driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: MELSECA

**MELSECA** tab:

**Channel parameters:**

*type, pc\_cpu, port, [baud, character, parity, stop]*

where:

<i>type</i>	- set of realized commands: ACPUR or AnCPU;
<i>pc_cpu</i>	- PC CPU number; in case of connection point-point it should be given ff ( <i>self PC CPU number</i> ),
<i>port</i>	- serial port name;
<i>baud</i>	- transmission speed in baud;
<i>character</i>	- number of bits in a transmitted character;
<i>parity</i>	- parity check type (even, odd, none);
<i>stop</i>	- number of stop bits.

Parameters *baud*, *character*, *parity*, *stop* and *buffer* are optional. In case of omitting them the default values are taken as follows:

- transmission speed - 9600 Bd,
- number of bits in a character - 8,
- parity check type - parity check (none),
- number of stop bits - 1.

### EXAMPLE

An example of declaration of transmission channel operating according to the MELSECA protocol is given below:

Channel / Name: CHAN1

Driver: MELSECA

Channel parameters: AnCPU,ff,COM1,9600,8,even,1

## Addressing the Process Variables

The syntax of symbolic address which is used for variables belonging to the MELSECA driver channel is as follows:

*VARIABLE\_TYPE* *variable\_index*

where:

*VARIABLE\_TYPE* - string identifying the variable type in the MELSECA protocol,  
*variable\_index* - index within a given type.

The following symbol of types of process variables are allowed (in columns on the right side the range of variable indexes for the ACPU and AnCPU sets of commands are given).

	ACPU	AnCPU
X - Input X	0 - 7FF	0 - 7FF
Y - Output Y	0 - 7FF	0 - 7FF
M - Internal relay M	0 - 2047	0 - 8191
L - Latch relay L	0 - 2047	0 - 8191
S - Step relay S	0 - 2047	0 - 8191
B - Link relay B	0 - 3FF	0 - 0FFF
F - Annunciator F	0 - 255	0 - 2047
TS - Timer (contact) T0	0 - 255	0 - 2047
TC - Timer (coil) T	0 - 255	0 - 2047
TN - Timer (present value) T0	0 - 255	0 - 2047
CS - Counter (contact) C	0 - 255	0 - 1023
CC - Counter (coil) C	0 - 255	0 - 1023
CN - Counter (present value) C	0 - 255	0 - 1023
MS - Special relay M	9000 - 9255	9000 - 9255
D - Data register D	0 - 1023	0 - 6143
W - Link register W	0 - 3FF	0 - 0FFF
R - File register R	0 - 8191	0 - 8191
DS - Special register D	9000 - 9255	9000 - 9255

The variable index for types X, Y, B and W is given in hexadecimal form, at the same time the indexes beginning with a letter should be preceded by digit 0, e.g. a correct declaration of input no. E has a form of X0E (declaration XE will be rejected as wrong).

Indexes of the other types are given in decimal format.

### EXAMPLE

X0A2            - value of input no. A2  
 D1010         - value of register D no. 1010

All process variables are treated as 16-bit numbers.

A correct operation of a communication processor A1SJ71C24-R2 requires appropriate setting of switches SW04 - SW12 and MODE (work mode) on the front panel. The MODE switch should be unconditionally set to position 1, because the MELSECA driver is based on the dedicated protocol with number 1. The SW04 switch should be set to ON if the application executes controls (state ON of the switch allows to write data in RUN mode). The SW12 switch should be set to ON (counting and verification of a checksum). The state

of switches SW05 - SW11 should be set according to transmission parameters given in the item declaring a MELSECA transmission channel:

*(transmission speed, number of characters in a word, number of stop bits, manner of parity check).*

The cable connecting a communication processor A1SJ71C24-R2 with an Asix system computer should be made according to the pattern given for the connection with a device which does not use signals DTR/DTS for transmission check (see: *chapter 4.5 External Wiring of documentation 'Computer Link Module type A1SJ71C24-R2'*).

The MELSECA driver is installed as a DLL automatically.

## Driver Configuration

MELSECA driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **MELSECA** section.

**Section name: MELSECA**

**Option name: LOG\_FILE**

**Option value: file\_name**

Meaning - allows to define a file where all diagnostic messages of the MELSECA driver and information about the contents of telegrams received and sent by the MELSECA driver will be written.

Default value - by default, the contents of telegrams are not written to the log file.

**Section name: MELSECA**

**Option name: CHECKSUM**

**Option value: YES/NO**

Meaning - allows to use a checksum in the protocol.

Default value - YES.

**Section name: MELSECA**

**Option name: LOG\_OF\_TELEGRAMS**

**Option value: YES/NO**

Meaning - allows to write to the log file (declared by using the item LOG\_FILE) the contents of telegrams sent and received by the MELSECA driver within the reading/writing process variables.

Default value - NO.

**Section name: MELSECA**

**Option name: LOG\_FILE\_SIZE**

**Option value: number**

Meaning - allows to specify the size of log file in MB.

Default value - 1 MB.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:  
YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.61 MEVAS - Driver of MEVAS Analyzers

### Driver Use

The MEVAS driver is used for data exchange between MEVAS emission computers and an Asix system computer. The communication is executed by means of serial interfaces. The driver realizes the protocol described in "*Bedienungsanleitung Rechnerschnittstelle MEVAS (vorläufige Version 1.00). Telegrammverkehr über eine serielle Schnittstelle. Stand: 26.03.1993*".

Parameterization of MEVAS driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the MEVAS driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: MEVAS

**MEVAS** tab:

*Channel parameters*:  
*COMn*, *Mevas\_Address*

where:

*COMn* - number of the serial port to which the network of MEVAS controllers is connected;  
*Mevas\_Address* - number identifying the MEVAS emission computer; the number is assigned on the stage of the MEVAS emission computer parameterization.

### Driver Configuration

MEVAS driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

Each defined channel may have its own section, the name of which is the logical name of the channel. A given COMn port may also have its own section named **MEVAS:n**. Values defined in such section become the default values for individual stations. The default values for individual serial interfaces are taken from the section named **MEVAS**. Transmission parameters by means of a serial interface cannot be placed in sections concerning individual stations, i.e. they may be located only in sections MEVAS and MEVAS:n.

Communication Drivers

- Section name: MEVAS**
- Option name: baud**
- Option value: number**

**Or**

- Section name: MEVAS**
- Option name: bps**
- Option value: number**

Meaning - determines transmission speed.

Default value - 9600

Parameter:

*number* - number passed in Bd.

- Section name: MEVAS**
- Option name: parity**
- Option value: parity\_number**

Meaning - determines parity.

Default value - n

Parameter:

*number* - allowed values:

n	- no parity bit,
o	- odd parity check,
e	- even parity check,
m	- mark,
s	- space.

- Section name: MEVAS**
- Option name: retries**
- Option value: number**

Meaning - number of repetitions of missed reading operations from a MEVAS station.

Default value - 3

- Section name: MEVAS**
- Option name: stop\_bits**
- Option value: number**

Meaning - determines number of stop bits.

Default value - 1

Parameter:

*number* - allowed values are 1 and 2.

- Section name: MEVAS**
- Option name: word**
- Option value: number**

**Or**

- Section name: MEVAS**
- Option name: word\_length**
- Option value: number**

Meaning - word length.

Default value - 8

Parameter:

*number* - allowed values are from interval 5 to 8.

- ☑ **Section name:** MEVAS
- ☑ **Option name:** `time_out`
- ☑ **Option value:** *number*

Or

- ☑ **Section name:** MEVAS
- ☑ **Option name:** `timeout`
- ☑ **Option value:** *number*

Meaning - waiting time for DMS285 answer.  
 Default value - 10000  
 Parameter:  
   number - number passed in milliseconds.

- ☑ **Section name:** MEVAS
- ☑ **Option name:** `bad_data_statuses`
- ☑ **Option value:** *status1,status2,...,statusN*

Meaning - determines numbers of data status, for which the data are found invalid. Status 13 (no data) causes that the data is always treated as invalid, independently of parameter value.  
 Default value - 0,13  
 Parameter:  
   - format: `status1, status2, ..., statusN`  
   or the character - (lack of the *bad data* status, except 13).

- ☑ **Section name:** MEVAS
- ☑ **Option name:** `log`
- ☑ **Option value:** *log\_file*

Meaning - parameter determines the name of file to which additional diagnostic information will be written.  
 Default value - lack

- ☑ **Section name:** MEVAS
- ☑ **Option name:** `alarm_code`
- ☑ **Option value:** *alarm\_number*

Meaning - parameter determines a number of alarm generated by the driver in case of loss and re-establishing a connection with the station. The value of -1 (by default) causes that alarms are not generated. In a situation of connection loss, the following number specifying the cause of connection loss is transmitted together with an alarm code:  
   0 - complete lack of any answer from the station;  
   1 - timeout;  
   2 - line errors (frame, parity, overrun errors);  
   3 - checksum errors;  
   4 - other errors;  
   5 - MEVAS was reset;  
   6 - timeout on the MEVAS side;  
   7 - checksum error on the MEVAS side.  
 This number determines the end status of last attempt to establish the connection.  
 Default value - -1

- ☑ **Section name:** MEVAS
- ☑ **Option name:** `simulation`
- ☑ **Option value:** *number*

Meaning - if the parameter value is 1, then the driver works in simulation mode and does not communicate with the station. Values of all variables are random.

Default value - 0

- Section name:** MEVAS
- Option name:** Refresh1,...,Refresh10
- Option value:** number

Meaning - parameters determine how often the driver has to transfer to the MEVAS station requests of preparing a new data set for later reading. Each parameter corresponds to a definite variable group. The parameter *Refresh1* refers to all variables. The parameter has a form of two numbers. The first number determines frequency of request sending, the second one determines a shift in time of request sending. For instant, if 60s, 10s is given, then requests of preparing new data may be sent at hours: 12:00:10, 12:01:10, 12:02:10 etc.

The parameter determines only the maximal frequency of request sending. If the driver does not receive requests to read new data from the other components of the Asix system (ASMEN), then requests to prepare new data are not sent to the MEVAS station.

If the parameter *Refresh1* and one of parameters *Refresh2-Refresh10* has a non-zero value, then the requests to prepare a data collection, defined by this last parameter, will be sent at moments fulfilling criteria defined by both the parameters simultaneously i.e. with a frequency equal to the minimal value of both the parameters.

If the parameter *Refresh1* has value 0,0, then corresponding to them other parameters should have non-zero values. Data read from the MEVAS station - except the D28 data (integrals) - receive a time stamp transferred by the MEVAS station at the moment of receiving a request to prepare a new data collection.

Default value - Refresh1=60s,10s (other parameters have values 0,00).  
Parameter:  
    *number* - both the numbers have a format *nnn[s|m/g/h]* where *nnn* determines the time and the letter designation determines the base of time (second, minute, hour, hour respectively). If time unit designation is omitted, then the second is assumed.

- Section name:** MEVAS
- Option name:** Round\_28
- Option value:** yes/no

Meaning - if the parameter has a value yes, then the time of the D28 data is rounded up to a full hour. The rounding refers only to data with time in form *hh:59:00*. Other time values are not rounded. To switch off the rounding one should give no as the parameter value.

Default value - 15

- Section name:** MEVAS
- Option name:** Round\_28\_Time
- Option value:** yes/no

Meaning - time in minutes determining the time rounding range of hour integrals. If the integral time is included in the range: the nearest full hour +/- parameter value, then the integral time will be rounded to the nearest full hour. By example, if the parameter value is 15 minutes, then times of integrals from the range 9:45:00 to 10:15:00 will be rounded to 10:00:00. The rounding occurs if the value of parameter *Round\_28* is yes.

If the parameter value is 0, then the rounding does not occur.  
Default value - 15

- ☑ **Section name:** MEVAS
- ☑ **Option name:** Time\_Read
- ☑ **Option value:** *number*

Meaning - the parameter determines a time interval in seconds, with which the driver updates the station time. The driver cyclically reads the station time with a given interval. The MEVAS station time is used for determining the time, at which requests to prepare new data should be sent to the MEVAS station.

Default value - 3600.

Parameter:  
*number* - time in seconds.

- ☑ **Section name:** MEVAS
- ☑ **Option name:** Max\_D28\_Time
- ☑ **Option value:** *number*

Meaning - determines the way of driver reaction to lack of the D28 data (integrals) during historical data completing. It is the time of data lack, in seconds, after which the driver will assume that historical data do not exist and transfer such information to write in the archive. The parameter is used only in the situation when the MEVAS station reports a lack of any values referring to a definite channel and the value type (Knr/Wsl). The parameter does not concern a situation when only a part of possible 52 historical values is not available. In this last case it is assumed that such situation will not change later.

Default value - 4200

Parameter:  
*number* - time in seconds.

## EXAMPLE

Examples of the MEVAS driver configuration.

Example 1:

Channel declaration:

*Channel / Name:* MVS\_1  
*Driver:* MEVAS  
*Channel parameters:* COM2,3

Driver parameters:

*Section name:* MEVAS:2  
*Option name:* baud  
*Option value:* 19200

In the example above the station named MVS\_1 connected to the COM2 port was defined. The transmission speed of 19200 bps will be used. The station has an identifier of 3.

Example 2:

Channel declarations:

*Channel / Name:* MVS\_1  
*Driver:* MEVAS  
*Channel parameters:* COM1,1

## Communication Drivers

*Channel / Name:* MVS\_2  
*Driver:* MEVAS  
*Channel parameters:* COM2,2

*Channel / Name:* MVS\_2  
*Driver:* MEVAS  
*Channel parameters:* COM3,1

*Channel / Name:* MVS\_2  
*Driver:* MEVAS  
*Channel parameters:* COM4,1

*Channel / Name:*  
*Driver:* MEVAS  
*Channel parameters:* COM5,4

*Channel / Name:*  
*Driver:* MEVAS  
*Channel parameters:* COM6,5

Driver parameters:

Default values for all stations:

*Section name:* MEVAS  
*Option name:* baud  
*Option value:* 19200

*Section name:* MEVAS  
*Option name:* Invalidity\_Status  
*Option value:* 1, 6, 14

Default values for stations connected to the COM3 port:

*Section name:* MEVAS:3  
*Option name:* baud  
*Option value:* 9600

Other parameters:

*Section name:* MVS\_2  
*Option name:* Invalidity\_Status  
*Option value:* 5

*Section name:* MVS\_3  
*Option name:* Invalidity\_Status  
*Option value:* 0,13

In the example above stations with names from MVS\_1 to MVS\_6 connected to ports from COM1 to COM6 are defined. All serial ports except COM3 will work with a speed of 19200 baud. The COM3 port will work with a speed of 9600 baud. All stations except MVS\_2 and MVS\_3 stations will use invalidity statuses 1, 6 and 14. The MVS\_2 station uses a value of 5 as an invalidity status. The MVS\_3 status does not use any invalidity status - setting parameter "-" was necessary to change the default values set in the MEVAS section.

## Time Stamp

Other data than D28 are transferred by driver to the Asix system together with a time received from the MEVAS station while executing a request from the MEVAS station to

prepare new data for reading. The frequency of sending request is specified by parameters *Refresh1*,..., *Refresh10*.

For data D28 (integrals) the MEVAS stations send a set of maximally 52 values. Each of these values is provided with its own time, which usually has a form of hh:59 (for 1-hour cycle of integration). This time is rounded up by driver to a full hour. The rounding may be turned off by means of parameter *Round\_28*. Range of rounding defines a parameter *Round\_Time\_28*.

Although each D28 datum has its own time, for the variables of this type it is also necessary to send to the MEVAS station a request to prepare new data and if the parameter *Refresh1* has a value of 0,0 then the parameter *Refresh5* must have a not zero value in order to read data correctly (historical data too).

## Defining the Process Variables

The variable definition is based on the MEVAS protocol description.

List of all types of variables is given in the end of this point.

*name* [.arg1[.arg2[.arg3]]]

where:

*argn* - may be: number, number preceded by text or text.

Square brackets [ and ] include parts, which may be absent in a variable definition.

The variable name may be a number of the value described in an appropriate query (of D type) of the communication protocol with the MEVAS system. The number may be preceded by a letter D. Variables defining data of other queries of the protocol are the numbers preceded by the query type e.g. X1 to X5 and S1 to S5.

A record <n..m> signifies a numerical value from a range n to m. A vertical line "|" signifies that it is allowed to choose one of text on either side of the line.

In the table below all the types of variables are placed. The column *Number of parameter Refresh* specifies, which of parameters from *Refresh2* to *Refresh10* concerns a given variable. The value of variables from X1 to X10 is a time of preparing a specific data group by the MEVAS station as a result of the lately sent request. Writing any value to these variables causes sending to the MEVAS station a request of preparing new data for reading.

If the *Status* column contains Yes, then a given variable is transferred to the Asix system with the status defined by the parameter *Invalidity\_Statures*. Such variable is accessed by the MEVAS station together with the status defined by the protocol. This status is converted to a numerical value and compared with values defined by the parameter *Invalidity\_Statures*. If the value of converted status is compatible to one of values defined by the parameter *Invalidity\_Statures*, then the data is regarded as invalid. The data with a status of 13 (no data) is always regarded as invalid.

**Table 33. Format of Variable Name.**

Format of Variable Name	TYPE	Number of Parameter Refresh	Status	Record	Example
[D]1.[Knr ch]<1..48>	WORD	2			D1.Knr1
[D]2.[Knr ch]<1..48>	FLOAT	2	Yes		D2.Knr1

Communication Drivers

[D]2.[Knr ch]<1..48>.F VAL	FLOAT	2			D2.Knr1.F
[D]2.[Knr ch]<1..48>.SS STS STA	WORD	2			D2.Knr1.SS
[D]3.[Knr ch]<1..48>	FLOAT	2	Yes		D3.Knr1
[D]3.[Knr ch]<1..48>.[F VAL]	FLOAT	2			D3.Knr1.F
[D]3.[Knr ch]<1..48>.SS STS STA	WORD	2			D3.Knr1.SS
[D]4.[Knr ch]<1..48>	FLOAT	2	Yes		D4.Knr1
[D]4.[Knr ch]<1..48>.F VAL	FLOAT	2			D4.Knr1.F
[D]4.[Knr ch]<1..48>.S STS STA	WORD	2			D4.Knr1.S
[D]4.[Knr ch]<1..48>.int i	WORD	2			D4.Knr1.int
[D]5.[Knr ch]<1..48>	FLOAT	2	Yes		D5.Knr1
[D]5.[Knr ch]<1..48>.F VAL	FLOAT	2			D5.Knr1.F
[D]5.[Knr ch]<1..48>.S STS STA	WORD	2			D5.Knr1.S
[D]5.[Knr ch]<1..48>.int i	WORD	2			D5.Knr1.int
[D]6.[Knr ch]<1..48>	FLOAT	2	Yes		D6.Knr1
[D]6.[Knr ch]<1..48>.F VAL	FLOAT	2			D6.Knr1.F
[D]6.[Knr ch]<1..48>.S STS STA	WORD	2			D6.Knr1.S
[D]6.[Knr ch]<1..48>.int i	WORD	2			D6.Knr1.int
[D]7.[Knr ch]<1..48>.[int i val]	WORD	2			D7.Knr1.int
[D]8.[Knr ch]<1..48>	FLOAT	2	Yes		D8.Knr1
[D]8.[Knr ch]<1..48>.F VAL	FLOAT	2			D8.Knr1.F
[D]8.[Knr ch]<1..48>.S STS STA	WORD	2			D8.Knr1.S
[D]8.[Knr ch]<1..48>.int i	WORD	2			D8.Knr1.int
[D]9.[Knr ch]<1..48>	FLOAT	2	Yes		D9.Knr1
[D]9.[Knr ch]<1..48>.F VAL	FLOAT	2			D9.Knr1.F
[D]9.[Knr ch]<1..48>.S STS STA	WORD	2			D9.Knr1.S
[D]9.[Knr ch]<1..48>.int i	WORD	2			D9.Knr1.int
[D]10.[Knr ch]<1..48>	FLOAT	2	Yes		D10.Knr1
[D]10.[Knr ch]<1..48>.F VAL	FLOAT	2			D10.Knr1.F
[D]10.[Knr ch]<1..48>.S STS STA	WORD	2			D10.Knr1.S
[D]10.[Knr ch]<1..48>.int i	WORD	2			D10.Knr1.int
[D]11.[Knr ch]<1..48>	FLOAT	2	Yes		D11.Knr1
[D]11.[Knr ch]<1..48>.F VAL	FLOAT	2			D11.Knr1.F
[D]11.[Knr ch]<1..48>.S STS STA	WORD	2			D11.Knr1.S
[D]11.[Knr ch]<1..48>.int i	WORD	2			D11.Knr1.int
[D]12.[Knr ch]<1..48>.[F VAL]	FLOAT	2			D12.Knr1.F
[D]12.[Knr ch]<1..48>.Ag cnt	WORD	2			D12.Knr1.Ag
[D]13.[Knr ch]<1..48>.[F VAL]	FLOAT	2			D13.Knr1.F
[D]13.[Knr ch]<1..48>.Ag cnt	WORD	2			D13.Knr1.Ag
[D]14.[Knr ch]<1..48>.[F VAL]	FLOAT	2			D14.Knr1.F
[D]14.[Knr ch]<1..48>.Ag cnt	WORD	2			D14.Knr1.Ag
[D]15.[Knr ch]<1..48>.[F VAL]	FLOAT	2			D15.Knr1.F
[D]15.[Knr ch]<1..48>.Ag cnt	WORD	2			D15.Knr1.Ag

Table 34. Format of Variable Name (continuation).

Format of Variable Name	TYPE	Number of Parameter Refresh	Status	Record	Example
[D]16.[Knr ch]<1..48>.[F VAL]	FLOAT	2			D16.Knr1.F
[D]16.[Knr ch]<1..48>.Ag cnt	WORD	2			D16.Knr1.Ag
[D]17.[Knr ch]<1..48>.[F VAL]	FLOAT	2			D17.Knr1.F
[D]17.[Knr ch]<1..48>.Ag cnt	WORD	2			D17.Knr1.Ag
[D]18.[Knr ch]<1..48>.[I VAL]	WORD	6			D18.Knr1.I
[D]19.[Knr ch]<1..48>.[F VAL]	FLOAT	6			D19.Knr1.F
[D]20.[Knr ch]<1..48>.[F VAL]	FLOAT	6			D20.Knr1.F
[D]21.[Knr ch]<1..48>	DWORD	2			D21.Knr1
[D]21.[Knr ch]<1..48>.val j jjjjmm jjjjmm	DWORD	2			D21.Knr1.val
[D]21.[Knr ch]<1..48>.t ttttmm ttttmm	DWORD	2			D21.Knr1.t
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KLO CLO 0	WORD	3			D24.Bnr1.Bsl1.KLO
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL1 CL1 1	WORD	3			D24.Bnr1.Bsl1.KL1
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL2 CL2 2	WORD	3			D24.Bnr1.Bsl1.KL2
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL3 CL3 3	WORD	3			D24.Bnr1.Bsl1.KL3
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL4 CL4 4	WORD	3			D24.Bnr1.Bsl1.KL4
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL5 CL5 5	WORD	3			D24.Bnr1.Bsl1.KL5
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL6 CL6 6	WORD	3			D24.Bnr1.Bsl1.KL6
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL7 CL7 7	WORD	3			D24.Bnr1.Bsl1.KL7
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL8 CL8 8	WORD	3			D24.Bnr1.Bsl1.KL8
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL9 CL9 9	WORD	3			D24.Bnr1.Bsl1.KL9
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL10 CL10 10	WORD	3			D24.Bnr1.Bsl1.KL10
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL11 CL11 11	WORD	3			D24.Bnr1.Bsl1.KL11
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL12 CL12 12	WORD	3			D24.Bnr1.Bsl1.KL12
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL13 CL13 13	WORD	3			D24.Bnr1.Bsl1.KL13
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL14 CL14 14	WORD	3			D24.Bnr1.Bsl1.KL14
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL15 CL15 15	WORD	3			D24.Bnr1.Bsl1.KL15
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL16 CL16 16	WORD	3			D24.Bnr1.Bsl1.KL16

**Table 35. Format of Variable Name (continuation).**

Format of Variable Name	TYPE	Number of Parameter Refresh	Status	Record	Example
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL17 CL17 17	WORD	3			D24.Bnr1.Bsl1.KL17
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL18 CL18 18	WORD	3			D24.Bnr1.Bsl1.KL18
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL19 CL19 19	WORD	3			D24.Bnr1.Bsl1.KL19
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL20 CL20 20	WORD	3			D24.Bnr1.Bsl1.KL20
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL21 CL21 21	WORD	3			D24.Bnr1.Bsl1.KL21
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL22 CL22 22	WORD	3			D24.Bnr1.Bsl1.KL22
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL23 CL23 23	WORD	3			D24.Bnr1.Bsl1.KL23
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL24 CL24 24	WORD	3			D24.Bnr1.Bsl1.KL24
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL25 CL25 25	WORD	3			D24.Bnr1.Bsl1.KL25
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL26 CL26 26	WORD	3			D24.Bnr1.Bsl1.KL26
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL27 CL27 27	WORD	3			D24.Bnr1.Bsl1.KL27
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL28 CL28 28	WORD	3			D24.Bnr1.Bsl1.KL28
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL29 CL29 29	WORD	3			D24.Bnr1.Bsl1.KL29
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL30 CL30 30	WORD	3			D24.Bnr1.Bsl1.KL30
[D]24.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.KL31 CL31 31	WORD	3			D24.Bnr1.Bsl1.KL31
[D]25.[Bnr BI]<1..96>.[Bsl Typ]<1..3>.Kl cl<1..31>	WORD	3			D25.Bnr1.Bsl1.Kl1
[D]26.[Bnr BI]<1..96>	DWORD	3			D26.Bnr1
[D]26.[Bnr BI]<1..96>.TOT TOTAL h hhhhmm hhhhmm	DWORD	3			D26.Bnr1.TOT
[D]26.[Bnr BI]<1..96>.ACT A aaaamm aaaamm	DWORD	3			D26.Bnr1.ACT
[D]26.[Bnr BI]<1..96>.CNT u uuu	WORD	3			D26.Bnr1.CNT
[D]27.[Bnr BI]<1..96>	DWORD	3			D27.Bnr1
[D]28.[Wnr]<1..52>.[Knr ch]<1..48>.Wsl<1..3>	FLOAT	5	Yes		D28.Wnr1.Knr1.Wsl1
[D]28.[Wnr]<1..52>.[Knr ch]<1..48>.Wsl<1..3>.F VAL	FLOAT	5			D28.Wnr1.Knr1.Wsl1.F

Table 36. Format of Variable Name (continuation).

Format of Variable Name	TYPE	Number of Parameter Refresh	Status	Record	Example
[D]28.[Wnr]<1..52>.[Knr ch]<1..48>.[Wsl<1..3>.TIM TIME	WORD	5			D28.Wnr1.Knr1.Wsl1.TIM
[D]28.[Wnr]<1..52>.[Knr ch]<1..48>.[Wsl<1..3>.SEC	WORD	5			D28.Wnr1.Knr1.Wsl1.SEC
[D]29.<1..16>.day	WORD	4			D29.1.day
[D]29.<1..16>.year	WORD	4			D29.1.year
[D]29.<1..16>.YSEC	DWORD	4			D29.1.YSEC
[D]29.<1..16>.DSEC	DWORD	4			D29.1.DSEC
[D]30	DWORD	9			D30
[D]31.s	WORD	Lack			D31.s
[D]31.p	WORD	Lack			D31.p
[D]32.[Egnr]<1..100>.[Lnr no	WORD	Lack	Yes		D32.Egnr1.Lnr
[D]32.[Egnr]<1..100>.[Knr ch	WORD	10	Yes		D32.Egnr1.Knr
[D]32.[Egnr]<1..100>.[Hk	WORD	10	Yes		D32.Egnr1.Hk
[D]32.[Egnr]<1..100>.[Gk	WORD	10	Yes		D32.Egnr1.Gk
[D]32.[Egnr]<1..100>.[F VAL]	FLOAT	10	Yes		D32.Egnr1.F
[D]32.[Egnr]<1..100>.time t	DWORD	10	Yes		D32.Egnr1.time
[D]33.[Knr ch]<1..48>.[F VAL]	FLOAT	2			D33.Knr1.F
[D]34.[Knr ch]<1..48>.[F VAL]	FLOAT	2			D34.Knr1.F
[D]35.[Knr ch]<1..48>	FLOAT	2	Yes		D35.Knr1
[D]35.[Knr ch]<1..48>.[F VAL]	FLOAT	2			D35.Knr1.F
[D]35.[Knr ch]<1..48>.[S STS STA	WORD	2			D35.Knr1.S
[D]35.[Knr ch]<1..48>.[int i	WORD	2			D35.Knr1.int
S1.start	WORD	N/A			S1.start
S1.end	WORD	N/A			S1.end
S2.start	WORD	N/A			S2.start
S2.end	WORD	N/A			S2.end
S3.start	WORD	N/A			S3.start
S3.end	WORD	N/A			S3.end
S4.start	WORD	N/A			S4.start
S4.end	WORD	N/A			S4.end
S5.start	WORD	N/A			S5.start
S5.end	WORD	N/A			S5.end
X1	DWORD	N/A		Yes	X1
X2	DWORD	N/A		Yes	X2
X3	DWORD	N/A		Yes	X3
X4	DWORD	N/A		Yes	X4
X5	DWORD	N/A		Yes	X5
X6	DWORD	N/A		Yes	X6
X7	DWORD	N/A		Yes	X7
X8	DWORD	N/A		Yes	X8
X9	DWORD	N/A		Yes	X9
X10	DWORD	N/A		Yes	X10

**Table 37. List of Data Statuses.**

Number	Status Character According to Protocol	Description Following the Protocol
0	A	Anlage AUS - Device OFF
1	S	Störung EIN - Failure ON
2	W	Wartung - Maintenance
3	w	Wartung manuell - Manual maintenance
4	T	Test
5	P	Plausibilität AUS - Plausibility OFF
6	M	Verrechnungsfehler - Allocation error
7	E	Ersatzwertsteuerung EIN - Replacement value control ON
8	s	Ersatzwertsteuerung Störung - Failure of replacement value control
9	u	ungültig Anfahrbetrieb - invalid startup
10	U	ungültig (durch Anlage-AUS oder Klassenblockwechsel) - invalid (by device OFF or class block changes)
11	*	nicht belegt - not occupied
12	?	unknown status
13	*****	No data (only for D28 and D32)

## Historical Data

An access to historical data is possible for the D28 type. The data are available from the current day beginning (maximum 52 values). The data for the last hour of the previous day are available only in the period 23:59-00:05 (circa). It means that a pause in communication in this period results in an irreparable loss of values for the last hour of a day. The current values of the data (integral) should be obtained by using a D28 variable with the parameter *Wnr1*.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose

- this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).  
 The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.62 MicroSmart - Driver for Data Exchange with MicroSmart Controllers from IDEC

### Driver Use

The MicroSmart protocol driver allows data with MicroSmart controllers from IDEC to be exchanged.

The transmission is executed with the use of serial ports by means of standard Asix system computer serial ports. Cooperation of the Asix system with the MicroSmart controller does not require any intervention in the driver program.

The parameterization of the MicroSmart driver is performed by the Architect application.

### Declaration of Transmission Channel

The declaration of the transmission channel utilizing the MicroSmart driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: MicroSmart

**MicroSmart** tab:

*Device number* - number between 0-31 or 255; number 255  
can only be used when a device is connected to the  
port

*Port* - serial port name (COM1 ... COM256)

*Transmission speed in bauds*

*Word length*

*Parity checking* - parity control type:

- no control
- even control
- odd control
- mark
- space

*Stop bits* - values: 1; 1.5; 2

*Transmission speed in bauds*, *Word length*, *Parity checking* and *Stop bits* are optional. If those are ignored, the following default values are adopted:

*Transmission speed*: 75,

*Word length*: 7,

*Parity checking*: no parity control,

*Stop bits*: 1

### EXAMPLE

An example of channel declaration:

Name: MicroSmart  
 Driver: MicroSmart  
 Device number: 5  
 Port: COM1

## Addressing the Process Variables

The symbolic address for variables belonging to the MicroSmart channel has the following syntax:

*variable\_type**variable\_index*

where:

- variable\_type* - string identifying the variable type in the controller;
- variable\_index* - variable index within the given type; in the case of data blocks, this is the word number in the data block.

The following marking of the process variables types are allowed:

- Q - single input status
- QB - statuses of outputs, transmitted in bytes,
- QW - statuses of inputs, transmitted in words,
- I - single input status,
- IB - statuses of inputs, transmitted in bytes,
- IW - statuses of inputs, transmitted in words,
- M - single flag status,
- MB - statuses of flags, transmitted in bytes,
- MW - statuses of flags, transmitted in words,
- R - status of single shift register bit,
- RB - statuses of shift registers, transmitted in bytes,
- RW - statuses of shift registers, transmitted in words,
- T - clock status as a byte of the following value:
  - 0 - when the time is measured
  - 1 - when the time measurement is over (timeout)
- TC - current values of clocks, transmitted in words,
- TP - setting values for clocks, transmitted in words,
- TS - clock status bytes
- TPVCS - byte of value 1, when the clock setting value was changed, and of value 0 in the opposite case
- C - counter status as a byte of the following value:
  - 0 - when the counter is counting
  - 1 - when the counting is over (countout)
- CC - current statuses of counters, transmitted in words,
- CP - setting values for counters, transmitted in words,
- CS - counter status bytes
- CPVCS - byte of value 1, when the counter setting value was changed, and of value 0 in the opposite case
- D - data word
- DD - double data word
- E - error code as a word (index may have a value of 0-5)

### EXAMPLES

- M0003 - flag no. 3
- MB0005 - byte containing 8 subsequent flags starting from flag 0005
- T1 - status of clock no. 1

CPVCS7 - byte determining whether the setting value of counter 7 was changed

**NOTE**

The last digit of the variable index I,IB,IW,Q,QB,QW,M,MB,MW is an octal digit.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- by default, all computers can read variables.

**Redundancy**

Purpose - with this option searches a redundant channel when connection with a controller is broken in local channel.

Option value:

*computer name*

- list including the names of network computers which will be permitted to search a redundant channel.

The default value

- by default, when option is enabled, the channel is searched on all computers.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.  
The default value - by default, no time for data validity is set.

## 1.63 MODBUS - Driver of MODBUS/RTU Protocol for MASTER Mode

### Driver Use

The MODBUS driver is used for data exchange with controllers or devices operating according to the MODBUS protocol. The transmission is executed by means of serial interfaces with the use of standard serial ports of an Asix system computer.

The cooperation of Asix with the controller by using the MODBUS protocol does not require any controller's program adaptation for data exchange.

(While implementing the MODBUS protocol the RTU mode, which enables a larger capacity of the interface, was used).

The driver has the following data types implemented:

HR	(holding registers),
IR	(input registers),
CS	(coil status),
IS	(input status).

and the following functions of the MODBUS protocol:

Read Coil Status	(function 01),
Read Input Status	(function 02),
Read Holding Registers	(function 03),
Read Input Registers	(function 04),
Force Single Coil	(function 05),
Preset Single Register	(function 06),
Force Multiple Coils	(function 15),
Preset Multiple Registers	(function 16).

Parameterization of MODBUS driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the MODBUS driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: MODBUS

**MODBUS/Channel parameters** tab:

**Identification**

**Device identifier (slave id)**;

**Transmission parameters**

- Port** - serial port name (max number of handled ports: 32);  
**Transmission speed in bauds** - transmission speed in baud; max transmission speed is equal to 115 kBd;  
**Number of bits in a character** - number of bits in a transmitted character;  
**Parity checking** - parity check type (even,odd,none);  
**Number of stop bits** - number of stop bits;

**Device**

- Maximal number of inputs/outputs...** - maximal number of inputs/outputs, the value of which may be transferred by devices within one cycle (max 127\*16 i/o states);  
**Maximal number of registers...** - maximal number of registers, the state of which may be transferred by the device within one cycle (max 127 registers).

**ASCII mode****Enable ASCII mode for this device**

**Write using 'Preset Multiple Registers' function** - Write the registers using the function no. 16 - *option has been introduced to enable writing in counters of N30 series of LUMEL company (firmware of these counters have only the function no. 16 implemented); by default, writing is realized with the use of the function no. 6 (Preset Single Register).*

Parameters *baud*, *character*, *parity*, *stop*, *max\_i/o*, *max\_register* and *buffer* are optional. In case of omitting them the following default values are taken:

- transmission speed - 9600 Bd,
- number of bits in a character - 8,
- type of parity check - parity check,
- number of stop bits - 1,
- default number of inputs/outputs - 16,
- default number of registers - 4.

**EXAMPLE**

An example of declaration of transmission channel working according to the MODBUS protocol is given below:

*Channel / Name:* CHAN1  
*Driver:* MODBUS  
*Device identifier:* 2  
*Port:* COM1  
*Transmission speed in bauds:* 9600  
*Number of bits in a character:* 8  
*Parity checking:* parzystość  
*Number of stop bits:* 1  
*Maximal number of inputs/outputs...:* 40  
*Maximal number of registers...:* 16

The transmission channel with the logical name CHAN1 has the following parameters defined:

- MODBUS protocol using a serial interface;
- device identifier (slave id) 2;
- port COM1;
- transmission speed of 9600 Bd;
- transmitted character length - 8 bits;
- parity check;
- one stop bit.

## Addressing the Process Variables

The syntax of symbolic address used for variables belonging to the MODBUS driver channel is as follows:

*VARIABLE\_TYPE* *variable\_index*

where:

- variable\_type* - string identifying the variable type in the MODBUS protocol;
- variable\_index* - variable index within a given type.

The following symbols of types of process variables are allowable:

- ES - Exception Status; BYTE type value returned by the function 07 of Modbus protocol - used by controllers servicing the function 07 of Modbus protocol;
- CS - Coil Status ( 0X reference );
- IS - Input Status ( 1X reference );
- HR - Holding Register ( 4X reference );
- IR - Input Register ( 3X reference );
- HRL - 2 successive Holding Registers treated as a double word in INTEL format;
- HRF - 2 successive Holding Registers treated as a floating-point number in INTEL format;
- HRLM - 2 successive Holding Registers treated as a double word in MOTOROLA format;
- HRFM - 2 successive Holding Registers treated as a floating-point number in MOTOROLA format;
- IRL - 2 successive Input Registers treated as a double word in INTEL format;
- IRF - 2 successive Input Registers treated as a floating-point number in INTEL format;
- IRLM - 2 successive Input Registers treated as a double word in MOTOROLA format;
- IRFM - 2 successive Input Registers treated as a floating-point number in MOTOROLA format.

### EXAMPLES

- CS22 - Coil 22
- IS197 - Input 197
- HR118 - Holding Register 118
- IR25 - Input Register 25

The MODBUS driver is loaded as a DLL automatically.

Considering the appearance of controllers using 32-bit registers, the MODBUS driver was extended with the support of 32-bit registers HR and IR:

- HR32L - 32-bit register HR of DWORD type (requires a calculation function

	based on DWORD, e.g. NOTHING_DW);
HR32F	- 32-bit register HR of FLOAT type (requires a calculation function based on FLOAT, e.g. NOTHING_FP);
IR32L	- 32-bit register of DWORD type (requires a calculation function based on DWORD, e.g. NOTHING_DW);
IR32F	- 32-bit register of FLOAT type (requires a calculation function based on FLOAT, e.g. NOTHING_FP).

**EXAMPLE**

X1, register HR no 10 as DWORD, HR32L10, CHAN32, 1, 1, NOTHING\_DW  
 X2, register IR no 20 as FLOAT, IR32F20, CHAN32, 1, 1, NOTHING\_FP

## Driver Configuration

The driver configuration is defined in the *Current Data* module, in the channel operating according to MODBUS driver.

**Driver parameters** tab:

 **Number of repetitions**

Meaning the item allows to define maximal number of trials to do the command in case of transmission errors.

Default value - by default, max. 3 repetitions are executed.

Parameter:

*number* - number of repetitions.

 **Transmission delay**

Meaning - the item allows to declare a time interval between the end of receiving the answer and sending the successive query to the remote device.

Default value - by default, the item assumes a value of transmission time of 3,5 characters.

Parameter:

*number* - time; the maximal value is equal to 55 ms.

 **Block read**

Option value: **NO/YES[,slave1, slave2, ... slaven]**

Meaning - the item enables to set an operation mode, in which values of registers and coils are read individually (the function of block data reading is not used). It is valid for ALL the variables supported by the driver.

Default value - by default, the mode of block data reading is used.

Parameters:

*NO,slave1, slave2,... slaven* - numbers of slaves the variable values will be read from one by one (without block read);

*YES,slave1, slave2,... slaven* - numbers of slaves the variable values will be read from by means of block read.

\*\*\*

**Driver parameters 2** tab:

**Timeouts of receive operation**

Meaning - the item allows to specify a maximal waiting time for arriving the first character of an answer from a specified remote device. After passage of this time it is assumed that the device under consideration does not work correctly and the transmission session ends with an error.

Default value - by default, it is assumed that the maximal waiting time for the first character of an answer is equal to 1000 milliseconds.

Parameter:  
    *slave\_no* - number of slave placed in the declaration of the transmission channel using the MODBUS protocol;  
    *time* - ber from a range of 100 - 65000 milliseconds.

**Timeouts of receiving successive characters**

Meaning - the item allows to specify a maximal waiting time for arriving the successive character of the answer from a specified remote device. After passage of this time it is assumed that the device under consideration does not work correctly and the transmission session ends with an error.

Default value - by default, it is assumed that the maximal waiting time for the successive character of the answer is equal to 100 milliseconds.

Parameter:  
    *slave\_no* - slave no. placed in the declaration of transmission channel using the MODBUS protocol;  
    *time* - number from a range of 10 - 2000 milliseconds.

**Turning off AsComm module**

Option value: **NO/YES**

Meaning - don't use AsComm module for initialization of the connections. AsComm module should be entered only in case of using switched line connections.

Default value - Yes.

\*\*\*

**Driver parameters - diagnostics** tab:

**Log file**

Meaning - the item allows to define a file where all diagnostic messages of MODBUS driver and information about the contents of telegrams received and sent by the MODBUS driver will be written. If the item does not define the full path, then the log file will be created in the current directory. The log file should be used only while the Asix start-up.

Default value - by default, the log file is not created.

Parameter:  
    *log\_file\_name*

**Log file size**

Meaning - determines the log file size in MB.

Default value - 1 MB.

Parameter:  
    *number* - number in MB.

**Log of telegrams**

Meaning	- the item allows to write to the log file (declared by using the item <i>Log file</i> ) the contents of telegrams sent and received by the MODBUS driver within the reading/writing process variables. Writing the contents of telegrams to the log file should be used only while the Asix start-up.
Default value	- by default, the contents of telegrams are not written to the log file.

## Connection by Means of Modem

### EXAMPLE

The MODBUS protocol may also exchange data by means of a modem connection.

The MODBUS driver channel is a client of server AsComm named MODBUS:n, where *n* is a number of the serial port.

If the channel declaration is as follows:

*Channel name:* logical\_name

*Channel driver:* Modbus

*Device identifier:* 4

*Port:* COM3

the name of client will be 'MODBUS:3'

To establish the communication with AsComm it is important to set up the **Switched line** parameter:

Architect > *Fields and Computers* > *Current data* module > declared channel parameters > **AsComm server client** tab

If the modem is connected to an other port than COMn, then you should give the number of this port by means of the parameter **Modem port** or specify the modem name by means of the parameter **Modem name**. You should also give a telephone number and define other parameters required. If MODBUS driver has to communicate with many controllers by means of the same modem, then one should define suitable number of channels assuming the parameter **Modem port** as a virtual transmission channel and specify for each channel an appropriate telephone number.

### EXAMPLE

Channel declarations:

*Name:* Chan1

*Driver:* MODBUS

*Device identifier:* 1

*Port:* COM11

*Transmission speed in bauds:* 9600

*Number of bits In a character:* 8

*Parity checking:* none

*Number of stop bits:* 1

*Maximal number of inputs/outputs:* 16

*Maximal number o registers:* 16

Declaration of 'Chan1' as AsComm server client:
---

*Switched line / Use a modem for connection initialization* - the parameter set up

*Define modem by name:* US Robotics

*Phone number:* 11111111

Name: Chan2  
Driver: MODBUS  
Device identifier: 1  
Port: COM12  
Transmission speed in bauds: 9600  
Number of bits In a character: 8  
Parity checking: none  
Number of stop bits: 1  
Maximal number of inputs/outputs: 16  
Maximal number o registers: 16

Declaration of 'Chan2' as AsComm server client:

*Switched line / Use a modem for connection initialization* - the parameter set up  
Define modem by name: US Robotics  
Phone number: 22222222

In the example above Chan1 will communicate with a controller placed under the telephone number 11111111, and the Chan2 with a controller placed under the telephone number 22222222. The US Robotics modem will be used. The **Modem name** parameter may be replaced by the parameter **Modem port**, which specifies the number of the serial port to which the modem is connected.

You should notice the given above description of using the MODBUS driver on switched links does not include the modem parameterization. The modem configuration depends on types of used modems.

## Channel Parameters

The Modbus channel parameters are declared on *Modbus / Channel Parameters*:

### **Timeouts of receive operation**

Meaning - the item allows to specify a maximal waiting time for arriving the first character of an answer from a specified remote device. After passage of this time it is assumed that the device under consideration does not work correctly and the transmission session ends with an error.

Default value - by default, it is assumed that the maximal waiting time for the first character of an answer is equal to 1000 milliseconds or is taken from the driver parameters (if declared).

Parameter:  
*time* - ber from a range of 100 - 65000 milliseconds.

### **Timeouts of receiving successive characters**

Meaning - the item allows to specify a maximal waiting time for arriving the successive character of the answer from a specified remote device. After passage of this time it is assumed that the device under consideration does not work correctly and the transmission session ends with an error.

Default value - by default, it is assumed that the maximal waiting time for the successive character of the answer is equal to 100 milliseconds or is taken from the driver parameters (if declared).

Parameter:  
*time* - number from a range of 10 - 2000 milliseconds.

**Transmission delay**

- Meaning - the item allows to declare a time interval between the end of receiving the answer and sending the successive query to the remote device.
- Default value - by default, is taken from the driver parameters.
- Parameter:  
*number* - time; the maximal value is equal to 55 ms.

 **Delay after no answer**

- Meaning - option used to block queries in the channel for a period measured in milliseconds, if the last query resulted in lack of response. During the blocking queries all commands directed to the driver are finished with a status indicating a lack of response (without performing any operations on the port). Option overloads the identical value of options declared in the driver options.
- Default value - by default, is taken from the driver parameters.
- Parameter:  
*number* - 0, it operates without blocking the transmission after no response.

 **Log file**

- Meaning - the item allows to define a file where all diagnostic messages of a given MODBUS channel and information about the contents of telegrams received and sent by the MODBUS channel will be written. If the item does not define the full path, then the log file will be created in the current directory. The log file should be used only while the Asix start-up.
- Default value - by default, is taken from the driver parameters.
- Parameter:  
*log\_file\_name*

 **Log file size**

- Meaning - determines the log file size in MB.
- Default value - 1 MB or is taken from the driver parameters (if declared).
- Parameter:  
*number* - number in MB.

 **Log of telegrams**

- Meaning - the item allows to write to the log file (declared by using the item *Log file*) the contents of telegrams sent and received by the MODBUS driver within the reading/writing process variables. Writing the contents of telegrams to the log file should be used only while the Asix start-up.
- Default value - by default, the contents of telegrams is taken from the driver parameters (if declared).

 **Block read**

- Meaning - the item enables to set an operation mode, in which values of registers and coils are read individually (the function of block data reading is not used). It is valid for ALL the variables supported by the driver in a given channel.
- Default value - by default, is taken from the driver parameters (if declared).
- Parameters:  
*YES/NO*

## Additional Channel Parameters

Parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of channel name section.

**Section name:** *<channel\_name>*

**Option name:** CONTROL\_VARIABLE\_PERIOD

**Option value:** *number\_of\_seconds*

Meaning - position changes the reading period of control variable - the use of the position makes sense in the case of paid links.

Default value - by default, the variable reading period is defined in the variable definition database.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

**Redundancy**

Purpose - with this option searches a redundant channel when connection with a controller is broken in local channel.

Option value:

*computer name*

- list including the names of network computers which will be permitted to search a redundant channel.

The default value - by default, when option is enabled, the channel is searched on all computers.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.64 MODBUS\_TCPIP - Driver of MODBUS\_TCP/IP Protocol for OPEN MODBUS/TCP Mode

### Driver Use

The driver MODBUS\_TCPIP is designed for data exchange between the Asix system and other computers/devices with use of the MODBUS protocol realized via an Ethernet network with the TCP/IP protocol. The default operating mode of the MODBUS\_TCPIP driver is the Open Modbus/TCP mode, developed on the base of specification titled: "OPEN MODBUS/TCP Specification" Release 1.0, issued 29.03.1999 by the firm Schneider Electric. The driver allows simultaneous operation in both SLAVE and MASTER modes.

Parameterization of MODBUS\_TCPIP driver is performed with the use of Architect module.

### SLAVE Mode

The SLAVE mode consists in executing the read/write commands for the ASMEN data sent from other computers/devices, which are defined as MASTER in the MODBUS network. It is allowed to connect many computers operating as MASTER in the network.

The ASMEN variables are accessed in the SLAVE mode as MODBUS variables from one of the types specified below:

- CS (coil status),
- HR (holding registers),
- IR (input registers).

Declarations of assignment (mapping) of the ASMEN variables to the MODBUS variables are placed by the application designer in text files, which are read by the driver during starting the Asix system (see: an example on the end of the chapter).

To read the ASMEN variables the following functions of the MODBUS protocol are implemented:

- Read Coil Status (function 01),
- Read Holding Registers (function 03),
- Read Input Registers (function 04).

To write the ASMEN variables the following functions of the MODBUS protocol are used:

- Preset Single Coil (function 05),
- Preset Single Register (function 06),
- Return Query Data (function 08, subfunction 00 00)
- Preset Multiple Coils (function 15),
- Preset Multiple Registers (function 16).

## Declaration of Transmission Channel

Declaration of the transmission channel utilizing the MODBUS\_TCPIP driver requires a channel with the following parameters to be added to the Current data module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* MODBUS\_TCPIP

**MODBUS\_TCPIP/Mode** tab:

*Mode:* Slave

**MODBUS\_TCPIP/Channel parameters** tab:

**Identification**

*Asix application number in MODBUS network  
 Computer number*

**NOTE** It is allowed to declare only one transmission channel executing the SLAVE mode.

## SLAVE Driver Configuration

The driver configuration is defined in the Current Data module, in the channel operating according to MODBUS\_TCPIP driver. The Slave mode has to be switched on (Modbus\_TCPIP > Mode tab).

**Driver parameters** tab:

**IP address**

Meaning - IP address through which the driver communicates with the controller.  
 Default value - the driver uses the IP provided by Windows operation system.

**Refreshing Variables Accessed by the Driver**

The driver may use one of two strategies of handling the variables exported from the Asix system.

**The strategy I** (default) assumes that all the variables, the names of which were declared in the file with mapping declarations, are periodically read by a separate driver thread from the ASMEN cache and placed in a common accessible driver

buffer. Threads supporting connection with clients build response telegrams on the basis of this buffer content, without necessity to access to the ASMEN API.

The period of reading from the ASMEN cache to shared driver buffer is configured by means of the option:

**Data buffer update period**

Meaning - enables declaring the period of reading from the ASMEN cache to shared driver buffer.

Default value - by default the period of reading is equal to 1 second.

Parameter:

*number* - period of reading from the ASMEN cache in seconds.

**The strategy II** (option) assumes that each thread takes data from the ASMEN cache individually in response to the queries sent by the client. As a result the thread reads from the ASMEN cache only these data, which are required by the client at that moment.

The strategy II should be also activated when Asix uses the driver only to import data from an other system. In such mode periodical refreshing of contents of driver buffers has no sense.

The strategy II is activated by means of the item:

**Reading on request**

Values of variables, which are in the ASMEN cache in the moment of executing the read command, may be out-of-date because in the period preceding the client query the refreshing of variables under consideration might be inactive (the Asix system was just run or none of ASMEN clients was interested in refreshing these variables). Therefore the thread preparing a transfer for the client checks the time stamp actually accessed variables values. If the time stamp value is not contained in the range of *credibility\_time* specified in the variable mapping declaration, then the thread reads the variable value directly from the driver and just then returns the answer to the client.

By default, the driver realizes the strategy I with a frequency of updating the content of common accessible buffer equal to 1 second.

\*\*\*

**Driver parameters - log tab:**

**Log file**

Meaning - the item allows to define a file where all diagnostic messages of the driver in the SLAVE mode and the information about contents of telegrams received/sent by the driver in this mode are written. If the item does not define the full path, then the log file is created in the current directory. The log file should be used only while the Asix start-up.

Default value - by default, the log file is not created.

*file name* - log file name.

**Log of telegrams**

Meaning - the item allows to write to the log file the contents of telegrams sent between the driver working in the SLAVE mode and network clients. Writing the contents of telegrams to the log file should be used only while the Asix start-up.

Default value - by default, the driver does not write the contents of telegrams to the log file.

**Log file size**

Meaning - the item allows to specify the log file size.

Default value - by default, the item assumes that the log file has a size of 1 MB.

Parameter:

*number* - log file size in MB.

\*\*\*

**Driver parameters** tab:

**Declaration of MODBUS RTU operation mode**

Meaning - the support of stations operating according to the specification MODBUS RTU requires to use an item.

Parameter:

**IP address** - address of the station operating as the master.  
**OVATION**

**IP address with Read only option**

Meaning - to the Asix system only the clients which have an access permission may connect.

Default value - by default, each client may read and write Asix system variables. It is possible to limit clients access only for reading the variables by using the read only option. In this situation an attempt to write is acknowledged by sending a response telegram of the exception type with a code 1 which signifies an illegal function.

Parameter:

**IP address** - IP address of the client accepted by the driver;  
**Read only** - option allowing the client to read data only.

Number of client declarations not limited.

**Example:**

A declaration of the client with IP address 10.10.10.84 with unrestricted access to the Asix system:

*IP address: 10.10.10.84*

**Connection timeout declared for individual IP address**

Meaning - the maximal time, which may elapse between successive queries from client (connection timeout), is declared for each connected

client. After exceeding the connection timeout, the connection with the client is broken. In case of clients, for which the timeout declaration of is not defined, a default value of 5 minutes is assumed.

Parameter:

- IP address*** - IP address of a client,
- Timeout*** - timeout in minutes

**Example:**

The declaration of 10-minute connection timeout for the client with the address IP 10.10.10.84:

Connection timeout:  
*IP address:* 10.10.10.84  
*Timeout:* 10

***Transferring Status of Variables in Separate Registers and Coils***

Meaning - the MODBUS protocol does not use a status with reference to transferred values of registers and coils. For this reason the status may be joined to the standard MODBUS transfer only in an artificial way. The driver under consideration may transfer the Asix variable status by placing it in the next element following one which contains the variable value. For registers it is the next register, for coils - the next coil. In case of registers the space for status is sufficient (16 bits), for coils the status must be limited to two states (0 - good , 1 - bad).

By example, if the variable value is transferred as the register 10, then the status is transferred as the register 11. If the coil state is transferred in the bit no. 5, then the coil status is transferred in the bit no. 6.

The applied method allows to send the variable value and its status in the same telegram which ensures the data compactness.

The mode of status transferring is declared individually for each client, separately for registers and for coils. Declarations enable:

- transferring status of registers,
- transferring status of coils.

Default value - by default, the status is not transferred either for registers or for coils.

Parameters:

- Registers status* - transferring status of register,
- Coils status* - transferring status of coils,
- IP address* - client IP address.

**Transferring Status of Variables Without Using Separate Registers or Coils**

Meaning - in case of work without transferring statuses in separate registers or coils, a problem of transferring the information that the variable value is invalid (e.g. errors of communication with data source) arises. The convention is assumed, according to which it is possible to declare the value transferred in case of incorrect status for registers (by default, 0xffff).

Parameter:

*IP address* - client IP address;  
*Register error status* - 16-bit value (HEX), transferred in case of an incorrect variable status.

**NOTE** In case of coils, because of a binary character of a variable, there is no possibility to transfer the information about the status in such way that it might differ from the correct variable value. Considering it you should also be assumed as a rule that in the work mode without status transferring the registers should be used.

**Declaration of Mapping MODBUS Variables to the ASMEN Variables**

Declarations of mappings of MODBUS variables to ASMEN variables are placed in text files. Localization of files with d mapping declarations is specified by means of the item:

Meaning - localization of files with declarations of mappings.

Parameter:

*IP address* - client IP address;  
*Map file* - name of a file containing declaration of mappings.

The purpose of connection of a file with a client IP address is to enable individual clients an independent way of mapping.

The number of items with declarations of mapping files is unlimited.

Because the MODBUS protocol does not use any names of variables but numbers of registers and coils, it is necessary to prepare configuration files for mapping MODBUS variables to ASMEN variables.

The declaration of mapping a MODBUS variable to an ASMEN variable is as follows:

```
Asix_name, MODBUS_address [, credibility_time][,
WITHOUT_TIME_STAMP][, TIME_STAMP]
```

where:

*Asix\_name* - process variable name of the Asix system; it must have its equivalent amongst names of process variables in ASMEN files;

*MODBUS\_address* - type and index of a MODBUS variable, by means of which the process variable value is accessed; depending on the MODBUS variable type, its value is transferred as one bit, one register (a variable of SHORT or USHORT type) or two successive registers (a variable of type FLOAT, LONG or ULONG);

Communication Drivers

*validity\_time* - time interval (in seconds), in which the variable value is assumed to be correct. This time is defined as a difference between the time stamp of a query from client and time stamp of an actually accessed variable value. By default, it is equal to 5.

An accepted format of the MODBUS address is as follows:

HR<reg\_no> or IR<reg\_no> or CS<coil\_no>

where:

*reg\_no* - number of register HR or IR;  
*coil\_no* - number of coil.

The number of MODBUS registers used for transferring the variable value is closely related to the ASMEN variable type:

- for WORD type variables it is one register or one coil;
  - for INT16 type variables it is one register;
  - for DWORD, LONG or FLOAT type variables it is two registers;
- or in case of status transfer:
- for WORD type variables it is two registers or two coils;
  - for INT16 type variables it is two registers;
  - for DWORD, LONG or FLOAT type variables it is two registers.

The method of register content interpretation for FLOAT, LONG and ULONG type numbers of MODBUS:

First register																Second register																
1	1	1	1	1	1	1	9	8	7	6	5	4	3	2	1	0	1	1	1	1	1	1	9	8	7	6	5	4	3	2	1	0
5	4	3	2	1	0											5	4	3	2	1	0											
Mantissa – less significant bits																Z		Exponent														

Figure. Format FLOAT (IEEE 745).

First register																Second register																
1	1	1	1	1	1	1	9	8	7	6	5	4	3	2	1	0	1	1	1	1	1	1	9	8	7	6	5	4	3	2	1	0
5	4	3	2	1	0											5	4	3	2	1	0											
Z		More significant bits														Less significant bits																

Figure. Format LONG.

First register																Second register																
1	1	1	1	1	1	1	9	8	7	6	5	4	3	2	1	0	1	1	1	1	1	1	9	8	7	6	5	4	3	2	1	0
5	4	3	2	1	0											5	4	3	2	1	0											
More significant bits																Less significant bits																

Figure. Format ULONG.

**EXAMPLE**

In ASMEN the variables X\_WORD, X\_INT, X\_DWORD, X\_LONG, X\_FLOAT are defined as follows:

X\_WORD, ED120.2, CHAN1, 1, 1, NOTHING  
 X\_INT, ED130.2, CHAN1, 1, 1, NOTHING\_INT  
 X\_DWORD, EL140.2, CHAN1, 1, 1, NOTHING\_DW  
 X\_LONG, EL150.2, CHAN1, 1, 1, NOTHING\_LONG  
 X\_FLOAT, EG160.2, CHAN1, 1, 1, NOTHING\_FP

Mapping the values of these variables to a continuous area of registers beginning from the register HR1 (without status transferring) is as follows:

X\_WORD, HR1  
 X\_INT, HR2

X\_DWORD, HR3  
 X\_LONG, HR5  
 X\_FLOAT, HR7

Mapping the values of these variables to a continuous area of registers beginning from the register HR1 (with status transferring) is as follows:

X\_WORD, HR1  
 X\_INT, HR3  
 X\_DWORD, HR5  
 X\_LONG, HR8  
 X\_FLOAT, HR11

\*\*\*

Parameters are declared in the *Miscellaneous* module, the *Directly entered* options tab.

### ***Declaration of passing timestamps for individual client***

- Section name: MODBUS\_TCPIP\_SLAVE**
- Option name: HR\_REGISTER\_TIME\_STAMP**
- Option value: IP\_address**

Meaning - with this option, timestamps can be declared for each Modbus\_tcpip driver client individually. The value of the option can be overridden for any variable from mapping file by using the parameter WITHOUT\_TIME\_STAMP in the declaration of mapped variable (in the mapping file). It is also possible the scenario in which the option HR\_REGISTER\_TIME\_STAMP will not be used, while the parameter TIME\_STAMP will be used for selected variables in the declaration of a mapped variable. The timestamp is passed as a 32-bit number containing the number of seconds since 01.01.1980 (in UTC). The driver puts the timestamp in two consecutive registers - being after register (registers) containing the value (and status) of mapped variable. Younger register of a timestamp contains more important 16 bits of timestamp.

Default value

- by default, timestamp is not passed to clients.

Parameters:

*IP\_address* - IP address of a client.

- Section name: MODBUS\_TCPIP\_SLAVE**
- Option name: OPC\_STATUS**
- Option value: IP\_address**

Meaning - with this option, passing the OPC status can be declared for each Modbus\_tcpip driver individually.

Default value

- by default, the status created according to the following rule is passed to the clients:  
 status o.k. - 0  
 status !=0 - 1

Parameters:

*IP address* - IP address of a client.

(Notice: Linking the file with the client IP address has to enable independent way of mapping for individual clients)

## MASTER Mode

The MASTER mode is the MASTER function implementation in the MODBUS network protocol (in RTU mode) based on ETHERNET with the TCP/IP protocol.

The MASTER mode has implemented the following data types:

CS	- Coil Statuses;
CSBx	- state of 8 consecutive coils from the coil x no. (BYTE type),
CSDWx	- state of 32 consecutive coils from the coil x no. (DWORD type),
CSWx	- state of 16 consecutive coils from the coil x no. (WORD type),
IS	- Input Statuses;
HR	- Holding Registers;
IR	- Input Registers;
HRL	- 2 successive Holding Registers treated as a double word in INTEL format;
HRF	- 2 successive Holding Registers treated as a floating-point number in INTEL format;
HRLM	- 2 successive Holding Registers treated as a double word in MOTOROLA format;
HRFM	- 2 successive Holding Registers treated as a floating-point number in MOTOROLA format;
IRL	- 2 successive Input Registers treated as a double word in INTEL format;
IRF	- 2 successive Input Registers treated as a floating-point number in INTEL format;
IRLM	- 2 successive Input Registers treated as a double word in MOTOROLA format;
IRFM	- 2 successive Input Registers treated as floating-point number in MOTOROLA format.

From the version Modbus\_tcpip 2.2.1.5:

HRD	- double (oldest HR contains the oldest bit of the number)
HRI	- int64 (oldest HR contains the oldest bits of the number)
HRDM	- double (youngest HR contains the oldest bit of the number)
HRIM	- int64 (youngest HR contains the oldest bits of the number)

Notice: Asmen in the ver. >=5.2 needed.

UDC - used to execute the MODBUS RTU protocol commands defined by the user, available only for keys WAUS, WAUW, WDUW or WDUN. The value of a UDC type variable is a string of ASCII characters, whose maximum length is 521 bytes. While writing, it is the content of a command defined by the user; while reading, it is the content of a response to a recently sent user's command.

### Examples

CS22	- Coil 22
IS197	- Input 197
HR118	- Holding Register 118
IR25	- Input Register 25

and the following functions of the MODBUS protocol:

Read Coil Statuses	(function 01),
Read Input Statuses	(function 02),
Read Holding Registers	(function 03),
Read Input Registers	(function 04),
Preset Single Coil	(function 05),
Preset Single Register	(function 06),
Preset Multiple Registers	(function 16 limited to writing a pair of registers).

The MODBUS driver is loaded as a DLL automatically.

## Declaration of Transmission Channel

Declaration of the transmission channel utilizing the MODBUS\_TCPIP driver requires a channel with the following parameters to be added to the Current data module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* MODBUS\_TCPIP

**MODBUS\_TCPIP/Mode** tab:

*Mode:* Master

**MODBUS\_TCPIP/Channel parameters** tab:

*Port number* - number of the port by means of which the connection with a slave type device numbered number will be executed;  
*IP\_address of the device* - IP address of the device;  
*Device number* - number of the slave type device supported in this channel (by default, 1);  
*Maximal number of binary variables...* - maximal number of binary signals in one query (by default, 32\*8);  
*Maximal number of registers...* - maximal number of registers in one query (by default, 127).

For each slave type device supported in the MASTER mode a separate declaration of transmission channel is required.

## MASTER Mode Configuration

The driver configuration is defined in the *Current Data* module, in the channel operating according to MODBUS\_TCPIP driver (*Modbus\_TCPIP > Driver parameters* tab). The *Master* mode has to be switched on *Modbus\_TCPIP > Mode* tab).

**Driver parameters** tab:

**IP address**

Meaning - IP address through which the driver communicates with the controller.  
 Default value - the driver uses the IP provided by Windows operation system.

## Declaration of Startup Time

Establishing connections with slaves is executed on the startup stage of the driver. The default time of startup duration is equal to 3 seconds. It may be modified by means of the item:

### **Startup time**

Meaning - startup time of the driver, during which connections are established with slaves.  
Default value - 3 s.  
Defining - manual.

### **Write using the 'Preset Multiple Registers' function**

Meaning - setting the function causes that recording the single HR register is performed with the function 16 (not 6).

### **MODBUS RTU**

Meaning - declaration operation in 'MODBUS RTU' mode; support of stations operating according to 'MODBUS RTU' specification.

Parameters:

*Client IP address* - IP address of a device,  
*MODBUS RTU*

### **Receive timeout**

Meaning - declaration of receive timeout.  
Default value - for devices with no timeout declaration - the default value is 5 seconds.

Parameters:

*Client IP address* - IP address of a device,  
*Timeout* - timeout in seconds.

\*\*\*

## **Driver parameters 2** tab:

### **Transmission delay**

Meaning - declaration of timeout between the end of receiving and sending another query to a device with IP address.

Default value - 0.

Parameters:

*IP address* - IP address of a device,  
*number of milliseconds*

### **Delay after no answer**

Meaning - declaration of minimal time that must elapse before sending the next query if the current query resulted in a lack of response.

Default value - 0.

Parameters:

*IP address* - IP address of a device,  
*number of seconds*

\*\*\*

### **Driver parameters - log tab:**

#### **Log file**

Meaning - the item allows to define a file to which all diagnostic messages of the driver generated in the MASTER mode and the information about the contents of telegrams sent/received in this mode by the driver are written. If the item does not define the full path, then the log file is created in the current directory. The log file should be used only while the Asix start-up.

Default value - by default, the log file is not created.

Defining - manual.

Parameter:  
*file\_name* - log file name.

#### **Log of telegrams**

Meaning - the item allows to write to the log file (declared by means of the item LOG\_FILE) the contents of telegrams sent between the driver operating in the MASTER mode and slave type devices. Writing the contents of telegrams should be used only while the Asix start-up.

Default value - by default, the driver does not write the contents of telegrams to the log file.

Defining - manual.

#### **Log file size**

Meaning - the item allows to specify a log file size.

Default value - by default, the log file size is equal to 1 MB.

Parameter:  
*number* - log file size in MB.

## **Channel Parameters (MASTER Mode)**

The channel parameters declared on Modbus\_TCPIP/Channel parameters 2 and Channel parameters - log:

#### **IP address**

Meaning - IP address through which the driver communicates with the controller.

Default value - the driver uses the IP provided by Windows operation system.

#### **Receive timeout**

Meaning - declaration of receive timeout.

## Communication Drivers

Default value - for devices with no timeout declaration - the default value is 5 seconds.

Parameters:

*Client IP address* - IP address of a device,  
*Timeout* - timeout in seconds.

### **Write using the 'Preset Multiple Registers' function**

Meaning - setting the function causes that recording the single HR register is performed with the function 16 (not 6).

### **Delay after no answer**

Meaning - declaration of minimal time that must elapse before sending the next query if the current query resulted in a lack of response.

Default value - 0.

Parameters:

*IP address* - IP address of a device,  
*number of seconds*

### **Log file**

Meaning - the item allows to define a file to which all diagnostic messages of the driver generated in the MASTER mode and the information about the contents of telegrams sent/received in this mode by the driver are written. If the item does not define the full path, then the log file is created in the current directory. The log file should be used only while the Asix start-up.

Default value - by default, the log file is not created.

Defining - manual.

Parameter:

*file\_name* - log file name.

### **Log of telegrams**

Meaning - the item allows to write to the log file (declared by means of the item LOG\_FILE) the contents of telegrams sent between the driver operating in the MASTER mode and slave type devices. Writing the contents of telegrams should be used only while the Asix start-up.

Default value - by default, the driver does not write the contents of telegrams to the log file.

Defining - manual.

### **Log file size**

Meaning - the item allows to specify a log file size.

Default value - by default, the log file size is equal to 1 MB.

Parameter:

*number* - log file size in MB.

\*\*\*

Parameters are declared in the *Miscellaneous* module, the *Directly entered* options tab.

The driver is parameterized with use of channel name section.

**Parameterization of Modbus exception handler with the number 6**

Meaning: The default behavior of the driver is to signal an exception in the control panel and set the error status for the variables attached to the command in response of which the driver received an exception no. 6. This behavior can be changed (for each driver channel individually) through the use of new options.

**Section name:** channel\_name

**Option name:** NUMBER\_OF\_REPETITIONS\_AFTER\_EXCEPTION\_NR\_6

**Option value:** number\_of\_repetitions

Default value: 0 - no repetitions.

**Section name:** channel\_name

**Option name:** TIMEOUT\_AFTER\_EXCEPTION\_NR\_6

**Option value:** number\_of\_milliseconds

Option value:

*number\_of\_milliseconds*

Default value: 50.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

## Communication Drivers

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\* \* \*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.65 MODBUSSLV - Driver of MODBUS/RTU Protocol for SLAVE Mode

### Driver Use

The MODBUSSLV driver is used to access the process variables values of the Asix system to other systems by means of a serial interface and the MODBUS protocol operating in RTU mode. In such connection the Asix system operates as a subordinate device (SLAVE) of the MODBUS protocol.

The MODBUSSLV protocol has the following types of variables implemented:

HR (holding registers),  
 IR (input registers),  
 CS (coil status).

and the following functions of the MODBUS protocol:

Read Coil Status (function 01),  
 Read Holding Registers (function 03),  
 Read Input Registers (function 04),  
 Force Single Coil (function 05),  
 Preset Single Register (function 06),  
 Return Query Data (function 08, subfunction 00 00),  
 Force Multiple Coils (function 15),  
 Preset Multiple Registers (function 16).

The function *Preset Multiple Registers* is limited to write the value of one process variable of the Asix system.

The function *Return Query Data* is needed for proper cooperation of Asix system with Symphony system (ABB).

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the MODBUSSLV driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: MODBUSSLV

**MODBUSSLV/Channel parameters** tab:

*Asix application number in MODBUS network* - number of a remote device of the MODBUS network assigned to Asix;  
*Port* - name of the serial port;  
*Transmission speed in bauds* - transmission speed, max 115 kBd;  
*Number of bits in a character* - number of bits in a character;  
*Parity checking* - parity check type;  
*Number of stop bits* - number of stop bits.

#### EXAMPLE

The declaration of the logical channel named CHAN1, which works according to the MODBUSSLV protocol and with parameters as below:

- number - 4

## Communication Drivers

- port - COM1
- transmission speed - 9600 Bd
- number of bits in a character - 8
- parity check type - parity check
- number of stop bits - 1

is as follows:

```
Channel / Name: KANAL1
Driver: MODBUSSLV
Asix application number in MODBUS network: 1
Port: COM1
Transmission speed in bauds: 9600
Number of bits in a character: 8
Parity checking: even
Number of stop bits: 1
```

The MODBUSSLV driver is loaded as a DLL automatically.

## Addressing the Process Variables

The syntax of symbolic address which is used for variables belonging to the MODBUSSLV driver channel is as follows:

```
name, address [,scale] [,WITHOUT_STATUS |WITH_STATUS]
```

where:

<i>name</i>	- name of the Asix process variable; it must have its equivalent among process variables declared in ASMEN files;
<i>address</i>	- type and number of the register by means of which the value of process variable is accessed; depending on the process variable type its value is transferred by means of one register (WORD or INT16 type variable) or two successive registers (FLOAT or DWORD type variable); there is a possibility to convert FLOAT type variables to a INT16 type variables;
<i>scale</i>	- <i>scale</i> determines a sort of operation ('-' division) and exponent of power of 10 when scalling values (value scalling is currently available for all variable types); it is used in case of converting the value of FLOAT type variables to a INT16 type number with simultaneous scaling (multiplication or division by power of 10), for example;
<i>WITHOUT_STATUS</i>	- variable value is transferred without status;
<i>WITH_STATUS</i>	- variable value is transferred with the status - the status occupies the next register after the register assigned to the variable.

### EXAMPLE

A value of the variable X1 is transferred by means of the register HR1. The variable is of FLOAT type and its value is converted to INT16. Before the conversion the variable value is multiplied by 100. Independently of the value of the item WITH\_STATUS the status (occupying the RH2 register) is transferred after the variable value.

```
X1, HR1, 2, WITH_STATUS
```

The value of the variable X2 is transferred by the HR2 register. Depending on the variable type it occupies only the HR2 register (WORD or INT16 type variable) or HR2 and HR3

registers (FLOAT and DWORD type variable). The variable value is transferred without the status independently of the value of the item WITH\_STATUS.

X2, HR2, WITHOUT\_STATUS

The value of the X3 variable is transferred by means of the HR3 register (if the variable is of WORD or INT16 type) or HR3 and HR4 registers (if the variable is of FLOAT or DWORD type). If the value of the item WITH\_STATUS is YES, then the variable X3 status is transferred in the register HR4 (WORD or INT16 type variable) or HR5 (FLOAT or DWORD type variable).

X3, HR3

## Driver Configuration

The driver configuration is defined in the *Current Data* module, in the channel operating according to ADAM driver.

### **Data file**

Meaning - the item allows to declare a file name in which the declarations of mapping the process variables of Asix to the registers HR and IR of MODBUS. The section may include many DATA FILE items.

Default value - lack of default file.

### Declaration of Transferring Statuses of Variables

#### **Transferring of status**

Meaning - the item allows to define globally the way of transferring the status of Asix system process variables. The value of the item equal to YES signifies that status is sent with the value of the Asix system process variable. The status is transferred in a register following the last register assigned to the process variable.

Default value - by default, the value of the item is equal to NO.

If the variable is of DWORD type and if the following declaration is given:

X1, HR3

then the variable status is transferred in the register HR5 (the X1 variable value is transferred in HR3 and HR4 registers).

If the variable X2 is of WORD type and if the following declaration is given:

X2, HR3

then the variable status is transferred in the register HR4 (the X2 variable value is transferred in the register HR3).

The *Transferring of status* item value equal to NO signifies that the process variable status is not transferred. It is possible to force the global settings by giving the component WITH\_STATUS or WITHOUT\_STATUS in the declaration of the process variable mapping. The declaration WITH\_STATUS signifies that, irrespective of the WITH\_STATUS value, the register following registers with the value of the considered variable contains the variable status. The declaration WITHOUT\_STATUS signifies that, irrespective of the WITH\_STATUS value, the variable status is not transferred.

### **Casting Variables of FLOAT Type to INT16**

**Data type cast**

Meaning - the item equal to YES allows converting the values of FLOAT type to INT16 type. Before converting it is possible to scale the FLOAT value by multiplication by a power of 10. The power value is given optionally in the declaration of the ASMEN variable (component *scale*) mapping.

Default value - by default, the **Data type cast** item value is equal to NO.

The declaration:

X1, HR1, -2

signifies that the value of X1 process variable will be divided by 100, and then it will be converted to a INT16 type number accessed in the register HR1.

The declaration:

X2, HR2, 3

signifies that the value of the X2 process variable will be multiplied by 1000, and then it will be converted to a INT16 type number accessed in the HR2 register.

### **Frequency of Refreshing Contents of Registers**

**Data update period**

Meaning - the values of registers are currently updated by reading data from ASMEN. The item determines a refreshing cycle (in seconds).

Default value - by default, the refreshing cycle is equal to 1 second.

### **Time of Answer Preparation**

**Reply timeout**

Meaning - time of elaborating an answer (in milliseconds) is controlled by the value of the item REPLY\_TIMEOUT. If in the time period passed by the REPLY\_TIMEOUT item the driver is not able to prepare data required by the MASTER, then it does not send any reply.

Default value - by default, the answer preparation time is set to 200 milliseconds.

### **Maximal Number of Registers Given in a Query**

**Number of registers in a telegram**

Meaning - the item determines a maximal number of registers that may be asked by MASTER in one query. If the number of registers specified in this item is too large, then the driver sends an answer of the EXCEPTION type with the code 4 (SLAVE DEVICE FAILURE).

Default value - by default, it is allowed to send queries including up to 128 registers.

**Log file**

Meaning - the item allows to define a file to which all diagnostic messages of the MODBUSSLV driver are written. If the item does not define the full path, then the log file is created in the current directory. The log file should be used only while the Asix start-up.

Default value - by default, the log file is not created.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value

- NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value

- by default, no time for data validity is set.

## 1.66 MPI - Driver of MPI Protocol for SIMATIC S7 PLCs

### Driver Use

The MPI driver is used for data exchange with SIMATIC S7 PLCs by means of the MPI interface. The transmission is executed by serial interfaces in the V24 (RS232C) standard with use of standard serial ports of an Asix system computer.

Operation of the Asix system with the SIMATIC S7 PLCs by using the MPI interface does not require any controller's program adaptation in order to perform data exchange.

Parameterization of MPI driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the MPI driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: MPI

**MPI/Channel parameters** tab:

*Port* - name of the serial port;  
*PC Address* - PC address;  
*Address of controller on MPI bus* - controller address on the MPI bus;  
*Transmission speed in bauds* - transmission speed in bauds;  
*Number of bits in a character* - number of bits in a transmitted character;  
*Parity checking* - parity check type,  
*Number of stop bits* - number of stop bits.

The parameters *PCaddress*, *MPIaddress*, *baud*, *character*, *parity*, *stop* are optional parameters. In case of omitting them the default values are taken:

- transmission speed - 19200 Bd,
- number of bits in a character - 8,
- type of parity check - odd parity check,
- number of stop bits - 1,
- address of S7 controller - 2,
- address of PC - 0.

#### EXAMPLE

Exemplary options declaring the transmission channel operating according to the MPI protocol are given below:

Channel / *Name*: CHAN2  
*Driver*: MPI  
*Port*: COM1  
*PC Address*: 0  
*Address of controller on MPI bus*: 2  
*Transmission speed in bauds*: 19200  
*Number of bits in a character*: 8  
*Parity checking*: kontrola nieparzystości  
*Number of stop bits*: 1

## Addressing the Process Variables

The syntax of symbolic address which is used for variables belonging to the MPI driver channel is as follows:

*VARIABLE\_TYPE* *variable\_index*

where:

*VARIABLE\_TYPE* - string identifying the variable type in the MPI protocol;  
*variable\_index* - variable index within a given type.

The following symbols of process variable types are allowable (the range of variable indexes is specific for different types of controllers):

EA	- output bytes,
EAW	- output words,
EAD	- output double words,
EE	- input bytes,
EEW	- input words ,
EDI	- 16-byte words in INTEL convention,
EDD	- input double words,
EM	- bytes of flags,
EMW	- words of flags,
EMD	- double words of flags,
EZ	- counter word,
ET	- timer word,
ED	- word in a data block,
EL	- double word in a data block,
ER	- floating-point number in a data block.

In case of data in a data block, after having given the type (EL or ED), you should give the data block number ended with a dot and then the word number.

### EXAMPLES

EMW15	- word of flags 15
EE0	- input word 0
EAW8	- output word 8
ED5.3	- word DW3 in data block DB5

The MPI driver is loaded as a DLL automatically.

## Driver Configuration

The driver configuration is defined in the *Current Data* module, in the channel operating according to MPI driver.

### **Data transfer**

Meaning - when the value yes is declared, transferring 32-byte data from DB is realized as a transfer of two 16-byte words; when the value no is declared, transferring 32-byte data from DB is realized as a transfer of one double word.

Default value - no.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

- Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.
- Option value:  
YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

- Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.
- Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

- Purpose - with this option deployed, local control in a channel will be disabled.
- Option value:  
YES / NO
- The default value - NO.

### **Data Validity Period**

- Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.
- Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.
- The default value - by default, no time for data validity is set.

## 1.67 MPS - Driver of MPS Protocol for Power Network Parameter Meters

### Driver Use

The MPS driver is used for communication with MPS power network parameter meters made by OBR Metrologii Elektrycznej in Zielona Góra, by means of a serial interface.

Parameterization of MPS driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the MPS driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* MPS

**MPS** tab:

*Channel parameters:*  
*controller address, COMn*

where:

*COMn* - is the number of the serial port to which the MPS controllers network is connected.

#### EXAMPLE

An example of the channel definition. In this case, the channel is a logical name of the MPS controller.

*Channel / Name:* MPS1  
*Driver:* MPS  
*Channel parameters:* 1,COM2

### Names of MPS Controller Variables

A variable may be defined by means of the name *FCnn* where *nn* is a number of a variable in the controller according to the description of the controller manufacturer. It is allowed also to use symbolic names. Letter case does not matter.

**Table 38. Symbolic Names of MPS Controller Variables.**

FC1 - U1	FC30 - Z1
FC2 - U2	FC31 - Z2
FC3 - U3	FC32 - Z3
FC4 - I1	FC33 - R

FC5 - I2	FC34 - R1
FC6 - I3	FC35 - R2
FC7 - P	FC36 - R3
FC8 - Q	FC37 - FI
FC9 - S	FC38 - FI1
FC10 - f	FC39 - FI2
FC11 - cos	FC40 - FI3
FC12 - P15	FC41 - P1s
FC13 - P1	FC42 - P2s
FC14 - P2	FC43 - P3s
FC15 - P3	FC44 - U1m
FC16 - Q1	FC45 - U2m
FC17 - Q2	FC46 - U3m
FC18 - Q3	FC47 - I1m
FC19 - S1	FC48 - I2m
FC20 - S2	FC49 - I3m
FC21 - S3	FC50 - KsU1
FC22 - cos1	FC51 - KsU2
FC23 - cos2	FC52 - KsU3
FC24 - cos3	FC53 - KsI1
FC25 - sin	FC54 - KsI2
FC26 - sin1	FC55 - KsI3
FC27 - sin2	FC56 - Hour
FC28 - sin3	FC57 - Min
FC29 - Z	FC58 - Sec

### Definition of Nominal Values

In order to improve the reading of measure values you should define nominal value of voltage, of current and of power (UL, IL, P, Q, S). These values may be given for each station. To do this you should entered in Architect in the *Miscellaneous* module, the *Directly entered options* tab:

#### EXAMPLE

*Section name:* ASMEN  
*Option name:* MPS1  
*Option value:* MPS,1,COM2

*Section name:* ASMEN  
*Option name:* MPS2  
*Option value:* MPS,2,COM2

*Section name:* MPS1  
*Option name:* UL  
*Option value:* 660

*Section name:* MPS1  
*Option name:* IL  
*Option value:* 100

*Section name:* MPS2  
*Option name:* UL  
*Option value:* 380

*Section name:* MPS2  
*Option name:* IL  
*Option value:* 100

## Driver Configuration

MPS driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **MPS** section.

Diagnostic option:

- Section name: MPS**
- Option name: Sleep**
- Option value: YES/NO**

Meaning - yes causes a change of the manner of short time period timing.

Default value - default value is equal to no.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

## Communication Drivers

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.68 MSP1X - Driver of Protocol for MSP-1x ELMONTEX PLCs

### Driver Use

The MSP1X driver is used for data exchange between MSP1X PLCs made by ELMONTEX and an Asix system computer. The communication is executed in the RS485 standard according to the protocol developed for MSP1X PLCs by ELMONTEX (no official specification).

Parameterization of MSP1X driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the MSP1X driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* *MSP1X*

**MSP1X** tab:

*Channel parameters:*  
*group\_no, device\_no, port [, baud]*

where:

<i>group_no</i>	- number of the group to which the controller belongs;
<i>device_no</i>	- number of a controller within the group;
<i>port</i>	- port name: COM1, COM2 etc.;
<i>baud</i>	- transmission channel (by default, 9600).

Other parameters are default:

- 8 bits in a character,
- without parity check (NONE),
- 1 stop bit.

#### EXAMPLE

The declaration of the logical channel named CHAN1, which works according the MSP1X protocol and exchanges data with the controller with the number 1, within the group with the number 5, by means of the COM2 port, with default transmission parameters:

*Channel / Name:* CHAN1  
*Driver:* MSP1X  
*Channel parameters:* 5, 1, COM2

The MSP1X driver is loaded as a DLL automatically.

## Addressing the Process Variables

The syntax of symbolic address which is used for variables belonging to the MSP1X driver channel is as follows:

*<type><index>*

where:

*type* - variable type,  
*index* - index within the type.

Symbols of variable types (the raw variable value type is given in parentheses):

<b>AI</b>	- Analog Input	(WORD),
<b>AO</b>	- Analog Output	(WORD),
<b>BI</b>	- Binary Input	(WORD),
<b>BO</b>	- Binary Output	(WORD),
<b>PV</b>	- Preset Value	(WORD),
<b>PD</b>	- Delta Preset Value	(WORD),
<b>IS</b>	- Binary Input Status	(WORD),
<b>OS</b>	- Binary Output Status	(WORD).

### EXAMPLE

Examples of declarations of process variables:

X4, analog input nr 1, AI1, CHAN1, 1, 1, NOTHING  
 X5, binary output nr 15, BO15, CHAN1, 1, 1, NOTHING

## Driver Configuration

MSP1X driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **MSP1X** section.

- Section name: MSP1X**
- Option name: REINITIALIZATION**
- Option value: YES/NO**

Meaning - allows to re-initiate a serial port before each communication session with the controller.

Default value - NO.

Defining - manual.

- Section name: MSP1X**
- Option name: WRITE\_DELAY**
- Option value: number**

Meaning - declares a time interval (in milliseconds) between the data writing to the controller and the next session of data exchange with the controller.

Default value - 1200.

Defining - manual.

**Section name: MSP1X** **Option name: HEADER\_DELAY** **Option value: number**

Meaning - declares a time interval (in milliseconds) between the characters of a command sent to the controller from the Asix system.

Default value - 50.

Defining - manual.

 **Section name: MSP1X** **Option name: DTR\_DELAY** **Option value: number**

Meaning - declares a time interval (in milliseconds) between sending a command to the controller and setting DTR signaling readiness for data receiving by the Asix system.

Default value - 2.

Defining - manual.

 **Section name: MSP1X** **Option name: UPDATE** **Option value: number**

Meaning - declares a time interval (in milliseconds) between the next data reading from the controller to internal driver buffers.

Default value - 5.

Defining - manual.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

 **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

 **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

## Communication Drivers

- computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.69 Muel - MUEL System Driver

### Driver Use

Muel protocol driver is used to exchange data between the Asix system and a MUEL system maintenance computer. Transmission is implemented using standard RS-232 computer serial ports.

Muel driver configuration is performed using the Architect application.

### Transmission Channel Declaration

Declaration of the transmission channel utilizing the Muel driver requires adding a channel with the following parameters to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* Muel

**Muel** tab:

Channel parameters:

*Port=number*

**where:**

*Port*                      - Serial port number

Transmission parameters are fixed at:

baud rate of 9600 Bd,  
 8-bit character  
 no parity,  
 1 stop bit.

#### EXAMPLE

Example of transmission channel declaration:

Port=1

### Declaration of Variables

The general syntax of the process variable symbolic address is as follows:

**<object><type>[.<index>]**

where:

*object* - the object number  
*type* - The variable type,  
*index* - Index within the type;

Variable type symbols (raw variable value type is given in parentheses):

**The types of variables with an index (read):**

**AI** - the voltage on the analogue input, index range 1 - 4 (FLOAT)  
**DI** - digital input state, index range 1 - 8 (WORD)  
**DO** - digital input state, index range 1 - 4 (WORD)

**The types of variables without an index (read):**

**T** - temperature (WORD)  
**STSA** - A status value (WORD)  
**BLTS** - B status value (WORD)  
**DI** - status of all digital inputs (WORD)  
**DO** - status of all digital outputs (WORD)  
**CON** - The connection status of the object (WORD)

**Examples of variable declarations for an object No 38:**

JJ\_11, AI No. 1, 38.AI.1, CHAN1, 1, 1, NOTHING\_FP  
JJ\_13, DI No. 2, 38.DI.2, CHAN1, 1, 1, NOTHING  
JJ\_14, DO No. 3, 38.DO.3, CHAN1, 1, 1, NOTHING  
JJ\_15, temperature, 38.T, CHAN1, 1, 1, NOTHING  
JJ\_16, all DI, 38.DI, CHAN1, 1, 1, NOTHING  
JJ\_17, all DO, 38.DO, CHAN1, 1, 1, NOTHING  
JJ\_18, A status, 38.STSA, CHAN1, 1, 1, NOTHING  
JJ\_19, B status, 38.STSB, CHAN1, 1, 1, NOTHING  
JJ\_20, connection status, 38.CON, CHAN1, 1, 1, NOTHING

## Driver Parameters

The Muel driver parameters are declared in the *Miscellaneous* module, in the *Directly Entered options* tab.

The driver configuration is done using a separate section called **CTMUUEL**.

- Section name:** CTMUUEL
- Option name:** LOG\_FILE
- Option value:** *file\_name*

Purpose: the text log file to which driver status messages are stored is used for diagnostic purposes.

The default value: By default the log file is not created.

- Section name:** CTMUUEL
- Option name:** LOG\_FILE\_SIZE
- Option value:** *number*

Purpose: this option is used to determine the size of the log file defined using LOG\_FILE item.

Option value: *number* - the size of the log file in MB.

The default value: the default log file size is 10 MB.

- Section name:** CTMUEL
- Option name:** LOG\_OF\_TELEGRAMS
- Option value:** YES | NO

Purpose: the option allows writing the contents of messages communicated between the driver and the controllers to the log file (declared using the LOG\_FILE item) The item in question should only be used during the Asix system start-up.

The default value: default value of the option is NO.

- Section name:** CTMUEL
- Option name:** REFRESH\_PERIOD
- Option value:** number

Purpose: this option is used to specify the maximum time (in seconds) after which the driver will read the data independently from the station. This option can be used in version 1.1.2.

The default value: By default, the time between readings is 30 seconds.

Option value:

number - The time between readings in seconds

#### EXAMPLE

Sample driver parameterization

- Section name:** CTMUEL
- Option name:** LOG\_FILE
- Option value:** d:\tmp\CtMuel\muel.log

- Section name:** CTMUEL
- Option name:** LOG\_FILE\_SIZE
- Option value:** 20

- Section name:** CTMUEL
- Option name:** LOG\_OF\_TELEGRAMS
- Option value:** TAK

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

- Remote Write Access**

Purpose

- this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

## Communication Drivers

- computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

- Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

- computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - by default, all computers can read variables.

### **Redundancy**

- Purpose - with this option searches a redundant channel when connection with a controller is broken in local channel.

Option value:

- computer name* - list including the names of network computers which will be permitted to search a redundant channel.
- The default value - by default, when option is enabled, the channel is searched on all computers.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

- Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

- YES / NO
- The default value - NO.

### **Data Validity Period**

- Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

- number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.
- The default value - by default, no time for data validity is set.

## 1.70 MultiMuz - Driver for MultiMUZ Microprocessor-Based Security Devices from JM-Tronik

### Driver Use

The MultiMuz protocol driver is used to exchange data between the Asix system and te MultiMUZ microprocessor-based security devices manufactured by Warsaw-based JM-Tronik. Communication is executed with the use of serial interfaces.

The driver executes the following functions:

- reading current statuses and measurements,
- reading events and reporting them to the Asix alarms system,
- reading troubles and logging them in the database in the format of the DataLogger application, developed by ASKOM Sp. z o.o.

The driver supports the following types of MultiMuz devices:

CR 001  
 LR 001  
 LR 001  
 LR 064  
 LR 067  
 LZ 001  
 PR 001  
 SR 001  
 SR 067  
 SR 072  
 SR.72A  
 SR.73  
 SR.96  
 TR 001  
 MegaMuz2 01.12 (supported in MultiMuz3 mode)  
 LR96

The support of other MultiMuz devices requires changes in the driver code.

The parameterization of the MultiMuz driver is performed by the Architect program.

### Declaration of Transmission Channel

The declaration of the transmission channel operating according to MultiMuz protocol requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

**Name:** logical name of the transmission channel  
**Driver:** *MultiMuz*

**MultiMuz/Channel parameters** tab:

**Identification**

**Device number**

**Device type** - IDs of the supported types:

- CR
- LR
- LR\_64
- LR\_67
- LZ
- LZ\_143\_01
- PR
- SR
- SR\_67
- SR\_72
- SR\_94
- SZR
- TR
- TR\_143\_01

**Transmission parameters**

**Port** - serial port number (COM)

**Alarms offset**

**Offset added to the calculated alarm number sent to the Asix system**

After selecting the device type push the button Show device parameters. The detailed informations on device parameters and event parameters of selected type will be displayed in a web browser (at least IE6).

Run the mentioned window just once and select consecutive device types - the window content will be refreshed automatically.

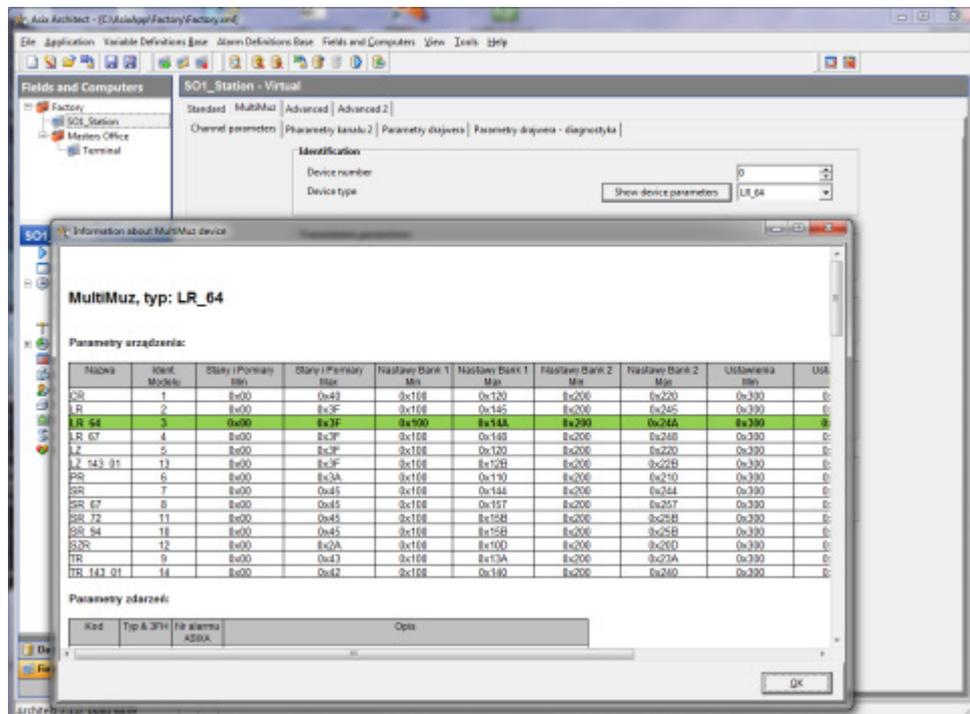


Fig. Information about MultiMuz device according to the selected type.

**Control variables****Name of the variable used for showing transmission status**

(0 - transmission status o.k.; 1 - lack of communication/communication errors; 2 - disabling MultiMuz support)

**Name of the variable used for controlling transmission state**

(0 - on; 1 - off)

**NOTE** Variables: both variable name should be entered. The variables have to belong to the channel of NONE type.

**Time synchronization**

**Period of device current time synchronizing in seconds** (0 - no synchronization)

Parameters of transmission to MultiMuz devices are constant:

- transmission speed 9600 Bd,
- 8 character bits,
- parity control,
- 1 stop bit.

**EXAMPLE**

Below is an example of a KSR\_67 channel declaration, which served by MultiMuz of network number 5, type SR 067 and alarm offset 200. The channel status is presented by variable MuzStat05, channel control is possible with the use of variable MuzCtrl05, time synchronization is performed every 60 seconds. Communication is executed through COM2 port:

*Name: KSR\_67  
 Driver: MULTIMUZ  
 Device number: 5  
 Device type: SR 067  
 Port: COM2  
 Offset added to the calculated alarm number sent to the Asix system: 200  
 Name of the variable used for showing transmission status: MuzStat05  
 Name of the variable used for controlling transmission state: MuzCtrl05  
 Period of device current time synchronizing in seconds: 60*

**Variable Declaration**

The driver provides the following variable types of MODBUS protocol:

CS	- binary value	(writing),
HR	- 16-bit log	(reading)
HRF	- 32-bit log of FLOAT type	(reading)
and internal driver variable:		
RS	- reason for logger triggering	(reading)
RQ	- logger reading request	(writing),
RT	- logger triggering time	(reading)
MN	- network number of MultiMuz, from which the logger is/was read	(reading)
RN	- number of currently/recently read logger	(reading)
RR	- status of currently/recently read logger	(reading)
SN	- number of recently read logger sample	(reading)

**The variable address has the following syntax:**

*Type[Index]*

where:

- Type*: - variable type,
- Index*: - variable index within 'Type' variable type.

**NOTE** Variables: the variable used to monitor the channel status and the variable used to disable the channel support must belong to a channel of NONE type.  
**1/** indexes are not given for types: *MN*, *RN*, *RR*, *RQ* and *SN*;  
**2/** in the case of *RS* and *RT* types, the range of indexes depends on the number of loggers defined in the device;  
**3/** the parameter of the logger reading request (*RQ* type) is the logger number. Loggers are numbered, starting from 1. If 0 is given as a parameter, it is equivalent to the request of stopping the currently executed logger reading;  
**4/** ranges for *HR* and *HRF* types depend on the device type;  
**4/** in order to present the logger triggering time, use the conversion function *DATAZAS\_MUZ*, which converts the *double*-type number to a character string in the following format:

*yyyy-mm-dd hh:nn:ss.zzz*

**EXAMPLE**

Examples of variable declaration (variable values are taken from a device of the *LR 001* type available through the *KLR\_01* channel):

ZM_01, positive active energy meter,	HRF5, KLR_01, 1, 1, NIC_FP
ZM_02, protection triggering 1,	HR19, KLR_01, 1, 1, NIC
ZM_03, measurement I1,	HRF28, KLR_01,1,1,NIC_FP
ZM_04, switch control,	CS2, KLR_01,1,1, NIC
ZM_05, change of settings bank,	CS10, KLR_01,1,1, NIC
ZM_06, logger triggering time 1,	RT1, KLR_01, 1,1, DATAZAS_MUZ

Examples of variable declaration without parameters (variable values are taken from a device of the *LR 001* type available through the *KLR\_01* channel):

ZM_07, logger reading request,	RQ, KLR_01,1,1,NIC
ZM_08, read logger number,	RN, KLR_01,1,1,NIC
ZM_09, logger reading status,	RR, KLR_01,1,1,NIC
ZM_10, read logger sample number,	SN, KLR_01,1,1,NIC
ZM_11, read device number,	MN, KLR_01,1,1,NIC

## Trouble Logging

The driver enables the automatic logging of troubles in the DataLogger application database. In order to activate the trouble logging mode, declare the option *Logging troubles on SQL server in AsLogger program base*, which defines:

- MS SQL server name,
- database name,

where the driver will save the read trouble samples.

The driver does not perform an automatic trouble logging - the logging must be initiated by the Asix system operator each time by the execution of control with the use of the *RQ*-type variable. As the control value, give the number of the logger, whose contents should be read from MultiMuz and saved to the DataLogger program base. If the control value is 0, it is interpreted as the request to stop the currently performed logger reading.

During logger reading, it is possible to track the reading status. This is executed with the use of the internal driver variables of symbolic addresses:

- a/ *MN* - number of read MultiMuz device,
- b/ *RN* - number of read logger,

- c/ *RR* - logger reading status:  
 - inactive or completed ok.,  
 - reading in progress,  
 - reading terminated with error.  
 d/ *SN* - number of recently read sample.

Trouble logging is performed in accordance to the logging plans. The names of the logging plans are not declared in the application (this would be too difficult, due to the number of devices in the switching stations, and due to the fact that each device can log troubles resulting from several causes). The driver assumes that the MultiMuz logging plans names have the following syntax:

*NrxxxpyyReason*

where:

- |               |  |
|---------------|--|
| <i>xxx</i>    | - MultiMuz network number,                                   |
| <i>yy</i>     | - number of serial port, which MultiMuz is connected to,     |
| <i>Reason</i> | - abbreviation determining the reason for logger triggering. |

### EXAMPLE

For trouble caused by a logger triggering by user logic in MultiMuz no. 2 on port COM11, the logging plan of the following name will be created:

*Nr002p11log\_trig\_by\_user\_logic*

Logging plans can be created with the use of the DataLogger program, but can also be created automatically by the driver during its operation. The second case occurs when after trouble reading, the driver does not locate the logging plan of the name appropriate for the read trouble parameters (*NoxxxpyyReason*) in the database. In this case, the driver automatically creates a new logging plan of the name compliant with the syntax *NoxxxpyyReasone*, and adds to it:

- a/ full set of plan items, including:
- currents: I0, I1, I2, I3
  - voltages: U0, U1, U2, U3
  - input states: WE01 ... WE20
  - output states: WY01 ... WY20
  - protection 1 triggering states: PZ01 ... PZ16
  - protection 1 operation states: ZZ01 ... ZZ16

The number of plan items related to protection triggering and operation states is specific for each type of MultiMuz,

- b/ creates a plan archive,  
 c/ sets the maximum and minimum values for currents and voltages in the logged trouble  
 - those will be corrected, if necessary, during reading the further troubles for the given plan.

## Event Signaling

Event signaling is checked cyclically, checking the frequency is determined by the *Event checking period* parameter. Events read from MultiMuz are converted to the appropriate Asix system alarm numbers. This conversion is performed on the basis of mapping presented for each MultiMuz type and offset presented in the ASMENA channel declaration. Two types of Asix alarms are possible:

- a/ text only  
 b/ text and one numerical parameter

Events from MultiMuz of type no. 2 are converted to Asix alarms of *text* type, while the remaining MultiMuz events (types 0, 1, 3 and 4) are changed to Asix alarms of text + one numerical parameter (of *float* type) type.

#### EXAMPLE 1

Event *Closing switch from controller* in MultiMuz of type LR 001 (*text*-type event) is mapped to the Asix alarm number of value 160. If in the declaration of the ASMENA channel supporting MultiMuz LR 001, the alarm offset of value 400 is given, the alarm definition in the Asix alarm definition file will have the form of (400 + 160 = 560):

*560, al, closing switch from controller*

#### EXAMPLE 2

Event *Circuit protection triggering* in MultiMuz of type SR 067 (*text and one numerical parameter*-type event) is mapped to the Asix alarm number of value 0. If in the declaration of the ASMENA channel supporting MultiMuz SR 067, the alarm offset of value 100 is given, the alarm definition in the the Asix alarm definition file will have the form of (100 + 0 = 100):

*100, al, circuit protection triggering %f*

Mapping the events of specific MultiMuz types to Asix alarm numbers is presented in the *NumeryAlarmowAsixa.xls* file.

## Driver Parameters

The MultiMuz driver parameters are declared in the *Current data* module on *MultiMuz* tab of the channel operating on the basis of MultiMuz driver protocol. Parameters placed on the tab *Driver parameters* refers to all the channels operating on the basis of the same driver. If one of that channel is parameterized on the *Driver parameters* tab, settings of the parameters will automatically appear on the *Driver parameters* tabs in definitions of other channels.

#### ***Driver parameters - diagnostic*** tab:

##### **Log file**

Meaning - the diagnostic purposes are fulfilled by a text log file, where the driver operation status messages are entered.

Default value - by default, the log file is not created.

Parameter:

*file\_name*

##### **Log file size**

Meaning - option is used to determine the size of the log file defined with the use of the *Log file* option.

Parameter:

*number*

Default value - log file size in MB.  
- default log file size is 1 MB.

**Log of telegrams**

Meaning - option allows the contents of telegrams sent between the driver and controllers to the log file (declared with use of *Log file* option) to be saved. The subject option should be used only during the start-up of the Asix system.

Default value - by default, the option is disabled.

 **Disturbances log file**

Meaning - the diagnostic purposes are fulfilled by a text log file, where the disturbance readout is entered.

Parameter:  
*file\_name*

Default value - by default, the log file is not created.

\*\*\*

**Driver parameters** tab: **Number of repetition**

Meaning: the option is used to specify the max number of repetitions in the case of transmission errors.

Default value: 3.

Option value:  
*number* - max. liczba powtórzeń w przypadku błędów transmisji.

 **Response timeout**

Meaning: This option specifies the max time the driver waits for responses from MultiMuz. The option is global and its value can be overloaded by individual option from the channel.

Default value: 1000.

Option value:  
*number* - time in milliseconds.

 **Char timeout**

Meaning: The option specifies the maximum time between characters of responses from MultiMuz. The option is global and its value can be overloaded by individual option from the channel.

Default value: 100.

Option value:  
*number* - time in milliseconds.

 **Event check period**

Meaning - option declares the interval (in seconds) between reading the event log of the subsequent device connected to the same serial port supported by the driver.

Default value - by default the option value is 10 seconds.

**No receive delay**

Meaning - time after which requests to the driver will be blocked after last request finished with no answer.

Default value - 0 - no blocking.

**Registration of disturbances in AsLogger database**

Meaning - the option switches the driver to trouble logging mode. This option defines the server name and the database name, in which the trouble samples will be logged.

Default value - by default, the troubles are not logged.

Parameters:

*ServerName* - MSSQL server name

*BaseName* - base name on *ServerName* server

**Directory with files containing registration plan names**

Meaning: This option defines the full path to the directory the driver will use looking for files with definitions of interference plan names.

Default value: -.

Option value:  
    *path* - full path to the file directory.

**EXAMPLE**

Example of driver parameters:

*Log file:* d:\tmp\multimuz\muz.log

*Log file size:* 3

*Telegrams log:* YES

*Troubles logging in SQL server's database of AsLogger:* SERVER\_RO6, BASE\_RO6

## Channel Parameters

**Channel parameters 2** tab:

**Baud rate**

Meaning: The option defines baud rate: 9600 or 19200 (for 2G only 9600).

Default value: - 9600.

**Response timeout**

Meaning: This option specifies the max time the driver waits for responses from MultiMuz. The option overloads the option from driver parameters.

Default value: 1000.  
 Option value:  
     *number* - time in milliseconds.

**Char timeout**

Meaning: The option specifies the maximum time between characters of responses from MultiMuz. The option overloads the option from driver parameters.

Default value: 100.  
 Option value:  
     *number* - time in milliseconds.

**Without events**

Meaning: The option used to disable the function of event reading in the channel.

Default value: Event readout is enabled.  
 Option value:  
     YES/NO

**Without registrar**

Meaning: The option used to disable the function of registrar reading in the channel.

Default value: Event readout is enabled.  
 Option value:  
     YES/NO

**No receive delay**

Meaning - time after which requests to the driver will be blocked after last request finished with no answer.

Default value - 0 - no blocking.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the

## Communication Drivers

- names of network computers which will be permitted to exercise remote control.
- Option value:  
YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

- Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.
- Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - by default, all computers can read variables.

### **Redundancy**

- Purpose - with this option searches a redundant channel when connection with a controller is broken in local channel.
- Option value:  
*computer name* - list including the names of network computers which will be permitted to search a redundant channel.
- The default value - by default, when option is enabled, the channel is searched on all computers.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

- Purpose - with this option deployed, local control in a channel will be disabled.
- Option value:  
YES / NO
- The default value - NO.

### **Data Validity Period**

- Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.
- Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.
- The default value - by default, no time for data validity is set.

## 1.71 MultiMuz\_tcpip - Driver for MultiMUZ Microprocessor-Based Security Devices from JM- Tronik

### Driver Use

The MultiMuz protocol driver is used to exchange data between the Asix system and te MultiMUZ microprocessor-based security devices manufactured by Warsaw-based JM-Tronik. Communication is executed in Ethernet with the use of TCP or UDP protocol.

The driver executes the following functions:

- reading current statuses and measurements,
- reading events and reporting them to the Asix alarms system,
- reading troubles and logging them in the database in the format of the DataLogger application, developed by ASKOM Sp. z o.o.

The driver supports the following types of MultiMuz devices:

CR 001  
 LR 001  
 LR 001  
 LR 064  
 LR 067  
 LZ 001  
 PR 001  
 SR 001  
 SR 067  
 SR.72  
 SR.72+  
 SR.73  
 SR.94  
 SR.94+  
 TR 001  
 megaMUZ TR  
 megaMUZ TR 082

The support of other MultiMuz devices requires changes in the driver code.

The parameterization of the MultiMuz\_TCPIP driver is performed by the Architect program.

### Declaration of Transmission Channel

The declaration of the transmission channel operating according to MultiMuz\_TCPIP protocol requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: MultiMuz\_TCPIP

**MultiMuz\_TCPIP** tab:

*Channel parameters:*

*number, type, alarmOff, port, status\_var, control\_var [,period]*

where:

*number* - network device number,

*type* - MultiMuz type; identifiers of supported types:

- 1 - CR 001
- 2 - LR 001
- 3 - LR 064
- 4 - LR 067
- 5 - LZ 001
- 6 - PR 001
- 7 - SR 001
- 8 - SR 067
- 9 - TR 001
- 13 - megaMUZ TR
- 14 - megaMUZ TR 082

*alarmOff* - offset added to the calculated number of Asix alarm;

*Port* - serial port number (TCPIP) and IP address;

*status\_var* - name of the variable used for channel status monitoring  
(0 - transmission status o.k.; 1 - lack of communication/communication errors; 2 - disabling MultiMuz support)

*control\_var* - name of the variable used for channel disabling  
(1 - request to disable MultiMuz support; 0 - enabling MultiMuz support)

*period* - period of time synchronization with MultiMuz (in seconds). A value of 0 means no time synchronization (default).

**NOTE** Variables: a variable used to monitor the channel status and the variable used to disable the channel support must belong to a channel of NONE type.

Transmission parameters of devices MultiMuz are fixed:

- Baud rate: 9600 Bd,
- 8 bits of mark,
- parity control,
- 1 stop bit.

## Variable Declaration

The driver provides the following variable types of MODBUS protocol:

- CS - binary value (writing),
  - HR - 16-bit log (reading)
  - HRF - 32-bit log of FLOAT type (reading)
- and internal driver variable:
- RS - reason for logger triggering (reading)
  - RQ - logger reading request (writing),
  - RT - logger triggering time (reading)
  - MN - network number of MultiMuz, from which the logger is/was read (reading)
  - RN - number of currently/recently read logger (reading)
  - RR - status of currently/recently read logger (reading)
  - SN - number of recently read logger sample (reading)

UDC - used to execute the MODBUS RTU protocol commands defined by the user, available only for keys WAUS, WAUW, WDUW or WDUN. The value of a UDC type variable is a string of ASCII characters, whose maximum length is 521

bytes. While writing, it is the content of a command defined by the user; while reading, it is the content of a response to a recently sent user's command. (read/write)

### The variable address has the following syntax:

*Type[Index]*

where:

*Type*: - variable type,  
*Index* - variable index within 'r;Type' variable type.

**NOTE** Variables: the variable used to monitor the channel status and the variable used to disable the channel support must belong to a channel of NONE type.

**1/** indexes are not given for types: *MN*, *RN*, *RR*, *RQ* and *SN*;

**2/** in the case of *RS* and *RT* types, the range of indexes depends on the number of loggers defined in the device;

**3/** the parameter of the logger reading request (*RQ* type) is the logger number. Loggers are numbered, starting from 1. If 0 is given as a parameter, it is equivalent to the request of stopping the currently executed logger reading;

**4/** ranges for *HR* and *HRF* types depend on the device type;

**4/** in order to present the logger triggering time, use the conversion function *DATA CZAS\_MUZ*, which converts the *double*-type number to a character string in the following format:

*yyyy-mm-dd hh:nn:ss.zzz*

### EXAMPLE

Examples of variable declaration (variable values are taken from a device of the *LR 001* type available through the *KLR\_01* channel):

ZM_01, positive active energy meter,	HRF5, KLR_01, 1, 1, NIC_FP
ZM_02, protection triggering 1,	HR19, KLR_01, 1, 1, NIC
ZM_03, measurement I1,	HRF28, KLR_01,1,1,NIC_FP
ZM_04, switch control,	CS2, KLR_01,1,1, NIC
ZM_05, change of settings bank,	CS10, KLR_01,1,1, NIC
ZM_06, logger triggering time 1,	RT1, KLR_01, 1,1, DATA CZAS_MUZ

Examples of variable declaration without parameters (variable values are taken from a device of the *LR 001* type available through the *KLR\_01* channel):

ZM_07, logger reading request,	RQ, KLR_01,1,1,NIC
ZM_08, read logger number,	RN, KLR_01,1,1,NIC
ZM_09, logger reading status,	RR, KLR_01,1,1,NIC
ZM_10, read logger sample number,	SN, KLR_01,1,1,NIC
ZM_11, read device number,	MN, KLR_01,1,1,NIC

## Trouble Logging

The driver enables the automatic logging of troubles in the DataLogger application database. In order to activate the trouble logging mode, declare the option *Logging troubles on SQL server in AsLogger program base*, which defines:

- MS SQL server name,
- database name,

where the driver will save the read trouble samples.

The driver does not perform an automatic trouble logging - the logging must be initiated by the Asix system operator each time by the execution of control with the use of the RQ-type variable. As the control value, give the number of the logger, whose contents should be read from MultiMuz and saved to the DataLogger program base. If the control value is 0, it is interpreted as the request to stop the currently performed logger reading.

During logger reading, it is possible to track the reading status. This is executed with the use of the internal driver variables of symbolic addresses:

- a/ *MN* - number of read MultiMuz device,
- b/ *RN* - number of read logger,
- c/ *RR* - logger reading status:
  - inactive or completed ok.,
  - reading in progress,
  - reading terminated with error.
- d/ *SN* - number of recently read sample.

Trouble logging is performed in accordance to the logging plans. The names of the logging plans are not declared in the application (this would be too difficult, due to the number of devices in the switching stations, and due to the fact that each device can log troubles resulting from several causes). The driver assumes that the MultiMuz logging plans names have the following syntax:

*NrxxxpyyReason*

where:

- xxx* - MultiMuz network number,
- yy* - number of serial port, which MultiMuz is connected to,
- Reason* - abbreviation determining the reason for logger triggering.

## Event Signaling

Event signaling is checked cyclically, checking the frequency is determined by the *Event checking period* parameter. Events read from MultiMuz are converted to the appropriate Asix system alarm numbers. This conversion is performed on the basis of mapping presented for each MultiMuz type and offset presented in the ASMENA channel declaration. Two types of Asix alarms are possible:

- a/ text only
- b/ text and one numerical parameter

Events from MultiMuz of type no. 2 are converted to Asix alarms of *text* type, while the remaining MultiMuz events (types 0, 1, 3 and 4) are changed to Asix alarms of *text + one numerical parameter* (of *float* type) type.

### EXAMPLE 1

Event *Closing switch from controller* in MultiMuz of type LR 001 (*text*-type event) is mapped to the Asix alarm number of value 160. If in the declaration of the ASMENA channel supporting MultiMuz LR 001, the alarm offset of value 400 is given, the alarm definition in the Asix alarm definition file will have the form of (400 + 160 = 560):

*560, al, closing switch from controller*

### EXAMPLE 2

Event *Circuit protection triggering* in MultiMuz of type SR 067 (*text and one numerical parameter*-type event) is mapped to the Asix alarm number of value 0. If in the declaration of the ASMENA channel supporting MultiMuz SR 067, the alarm offset of value 100 is given, the alarm definition in the the Asix alarm definition file will have the form of (100 + 0 = 100):

100, al, circuit protection triggering %f

Mapping the events of specific MultiMuz types to Asix alarm numbers is presented in the *NumeryAlarmowAsixa.xls* file.

## Driver Parameters

The MultiMuz\_TCPIP driver parameters are declared in the *Miscellaneous* module, on *Directly entered options* tab.

**Section\_name: MultiMuz\_TCPIP**

**Option\_name: LOG\_FILE**

**Option\_value: file\_name**

Meaning - the diagnostic purposes are fulfilled by a text log file, where the driver operation status messages are entered.

Default value - by default, the log file is not created.

Defining - manual.

Parameter:

*file\_name*

**Section\_name: MultiMuz\_TCPIP**

**Option\_name: LOG\_OF\_TELEGRAMS**

**Option\_value: YES/NO**

Meaning - option allows the contents of telegrams sent between the driver and controllers to the log file (declared with use of *Log file* option) to be saved. The subject option should be used only during the start-up of the Asix system.

Default value - by default, the option is disabled.

Defining - manual.

**Section\_name: MultiMuz\_TCPIP**

**Option\_name: LOG\_FILE\_SIZE**

**Option\_value: number**

Meaning - option is used to determine the size of the log file defined with the use of the *Log file* option.

Parameter:

*number* - log file size in MB.

Default value - default log file size is 1 MB.

Defining - manual.

**Section\_name: MultiMuz\_TCPIP**

**Option\_name: EVENT\_CHECK\_PERIOD**

**Option\_value: number**

Meaning - option declares the interval (in seconds) between reading the event log of the subsequent device connected to the same serial port supported by the driver.

Default value - by default the option value is 10 seconds.

Defining - manual.

**Section\_name: MultiMuz\_TCPIP**

**Option\_name: REGISTRAR**

**Option\_value: server\_name, database\_name**

Meaning - the option switches the driver to trouble logging mode. This option defines the server name and the database name, in which the trouble samples will be logged.

Default value - by default, the troubles are not logged.

Parameters:

ServerName - MSSQL server name

BaseName - base name on *ServerName* server

Defining - manual.

## EXAMPLE

Example of driver parameters:

```
[MULTIMUZ_TCPIP]
LOG_FILE=d:\tmp\multimuz\muz.log
LOG_FILE_SIZE=3
LOG_OF_TELEGRAMS=YES
REGISTRAR=SERWER_RO6, BAZA_RO6
```

## Channel Parameters

The parameters defined for individual channels in the sections with the name of Asmen channel:

**Section name: <channel\_name>**

**Option name: NUMBER\_OF\_REPETITIONS**

**Option value: number**

Meaning - number of repetitions in the case of error.

Default value - no repetitions.

**Section name: <channel\_name>**

**Option name: MAX\_REGISTERS**

**Option value: number**

Meaning - maximal number of registers in one inquiry.

Default value - 119.

**Section name: <channel\_name>**

**Option name: LOG\_OF\_TELEGRAMS**

**Option value: YES/NO**

Meaning - option allows the contents of telegrams sent between the driver and controllers to the log file.

Default value - default settings from the driver section.

**Section name: <channel\_name>**

**Option name: RECONNECT\_TIMEOUT**

**Option value: number\_of\_seconds**

Meaning - timeout for reconnect after depletion of repetition number.

Default value - after depletion of repetition number the driver indicates a transmission error.

**Section name:** *<channel\_name>*

**Option name:** **PORT**

**Option value:** *number*

Meaning - the number of a port through the driver communicates with the controller.

**Section name:** *<channel\_name>*

**Option name:** **IP\_ADDRESS**

**Option value:** *ip\_address*

Meaning - IP address of network card through which the driver communicates with the controller.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

**Redundancy**

Purpose - with this option searches a redundant channel when connection with a controller is broken in local channel.

Option value:

*computer name*

- list including the names of network computers which will be permitted to search a redundant channel.

## Communication Drivers

The default value - by default, when option is enabled, the channel is searched on all computers.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.72 MultiMuz3 - Driver for Communication with Microprocessor Protection Devices of MultiMUZ3 Type by JM-Tronik

### Driver Use

The MultiMuz protocol driver is used to exchange data between the Asix system and the MultiMUZ3 microprocessor-based security devices manufactured by Warsaw-based JM-Tronik.

The driver supports the following MultiMuz3 device types:

- 01.07
- 01.09
- 10.02
- 10.11
- 10.12
- 23.01
- 26.01
- MegaMuz2 01.12 (supported in MultiMuz3 mode)
- LR96

The driver executes the following functions:

- reading current statuses and measurements,
- reading events and reporting them to the Asix alarms system,
- reading disturbances and logging them in the database in the format of the DataLogger application, developed by ASKOM Sp. z o.o.

### Declaration of Transmission Channel

The declaration of the transmission channel operating according to MultiMuz3 protocol requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: MultiMuz3

**MultiMuz3/Channel Parameters** tab:

**Device Number** - network device number;

- Device Type** - device type;
- Alarms Offset** - offset added to the calculated Asix alarm number;
- Port** - port name (COM1, ...);
- Baud Rate** - predefined baud rate;
- Control Variables**
- variable used to show the communication status with device:
    - 0 - transmission o.k.,
    - 1 - no communication/communication error,
    - 2 - disabling the support for device,
  - variable used to control the communication status with device:
    - 1 - request to disable the support for device,
    - 0 - enabling the support for device.
- Time Synchronization**
- time synchronization period with a device (in seconds). 0 means no synchronization (default value).

## Variable Declaration

The driver provides the following variable types of MODBUS protocol:

- CS - binary value (writing),
- HR - 16-bit log (reading)
- HRF - 32-bit log of FLOAT type (reading)
- HRL - 32-bit log of DWORD/LONG type (reading)

and internal driver variable:

- RS - reason for logger triggering (reading)
- RQ - logger reading request (writing),
- RT - logger triggering time (reading)
- MN - network number of MultiMuz, from which the logger is/was read (reading)
- RN - number of currently/recently read logger (reading)
- RR - status of currently/recently read logger (reading)
- SN - number of recently read logger sample (reading)
- RSKR - reason for release of criterion logger (reading)
- RQKR - request for reading a criterion logger (writing)
- RTKR - release time of criterion logger (reading)
- MNKR - network number of the device from which criterion logger is/was read (reading)
- RNKR - number of currently/recently read out criterion logger (reading)
- RRKR - status of currently/recently read out criterion logger (reading)
- SNKR - number of recently read out sample of criterion logger (reading)
- UDC - type for executing commands of MODBUS RTU protocol defined by the user. Available only for the keys: WAUS, WAUW, WDUW or WDUN. UDC variable value is ASCII string with a maximum length of 521 bytes. When writing it is the content of the command created by a user, when reading it is the content of the response to recent sent user's command. (reading/writing)

The variable address has the following syntax:

*Type[Index]*

where:

*Type*: - variable type,  
*Index* - variable index within 'Type' variable type.

**NOTE** Variables: the variable used to monitor the channel status and the variable used to disable the channel support must belong to a channel of NONE type.

**1/** indexes are not given for types: *MN, RN, RR, RQ, SN, MNKR, RNKR, RRKR, RQKR* and *SNKR*;

**2/** in the case of *RS, RT, RSKR, RTKR* types, the range of indexes depends on the number of loggers defined in the device;

**3/** the parameter of the logger reading request (*RQ, RQKR* type) is the logger number. Loggers are numbered, starting from 1. If 0 is given as a parameter, it is equivalent to the request of stopping the currently executed logger reading;

**4/** in order to present the logger triggering time, use the conversion function *DATA CZAS\_MUZ*, which converts the *double*-type number to a character string in the following format:

*yyyy-mm-dd hh:nn:ss.zzz*

#### EXAMPLE

ZM\_07, number of the disturbance recorder being read, RN, KLR\_01,1,1,NOTHING  
 ZM\_08, status of disturbance recorder readout, RR, KLR\_01,1,1,NOTHING  
 ZM\_9, number of sample read from the disturbance recorder, SN, KLR\_01,1,1,NOTHING  
 ZM\_10, request for reading the criterion recorder, RQKR, KLR\_01,1,1,NOTHING  
 ZM\_11, number of the criterion recorder being read, RNKR, KLR\_01,1,1,NOTHING  
 ZM\_12, status of criterion recorder readout, RRKR, KLR,1,1,NOTHING  
 ZM\_13, number of sample read from the criterion recorder, SNKR, KLR\_01,1,1,NOTHING

## Readout of Loggers

The driver enables the automatic logging of troubles in the AsLogger application database. In order to activate the trouble logging mode, declare the option **Registration of Disturbances in AsLogger Database** in the driver options, which defines:

- MS SQL server name,
- database name,

where the driver will save the read trouble samples.

(see more in: *Driver Parameters*)

The driver does not perform an automatic trouble logging - the logging must be initiated by the Asix system operator each time by the execution of control with the use of the RQ-type or RQKR- type variable. As the control value, give the number of the logger, whose contents should be read from MultiMuz and saved to the AsLogger program base. If the control value is 0, it is interpreted as the request to stop the currently performed logger reading.

During logger reading, it is possible to track the reading status. This is executed with the use of the internal driver variables of symbolic addresses:

- a/ *MN, MNKR* - number of read MultiMuz device,
- b/ *RN, RNKR* - number of read logger,

- c/ *RR, RNKR* - logger reading status:
  - 0 - inactive or completed ok.,
  - 1 - reading in progress,
  - 2 - reading terminated with error.
- d/ *SN, SNKR* - number of recently read sample.

Trouble logging is performed in accordance to the logging plans. The names of the logging plans can be:

- 1/ defined by the user;
- 2/ generated automatically by the driver.

When registering, the driver writes into the AsLogger database:

- a/ all parameters contained in the record of logger, described in the documentation protocol of version 01.07;
- b/ creates an archive plan;
- c/ sets max. and min. values for currents and voltages in registered data series - they will be corrected during registration of consecutive series for a given plan.

## Names of Logging Plans Defined by the User

The user can define names of registration plans in text files with the extension '.def'. To do it, for each transmission channel (ASMEN channel) a separate file should be created and definitions of names for logging plans for disturbances logged in a given channel should be written into this file. Files with definitions of plan names should be written into a common directory. The name of this directory should be placed in the item **Directory with Files Containing Registration Plan Names** in the driver parameters. (See more in: *Driver Parameters*)

The syntax of definitions for registration plan placed in the file '.def' is as follows:

***channel, type, cause, table [, comment]***

where:

- channel* - name of ASMEN channel,
- type* - type of logger (K - criterion, Z - disturbances),
- cause* - number of cause (compatible with the specification given in the description of protocol of ver. 01.07)
- table* - name of logging plan,
- comment* - comment

### EXAMPLE:

Defining the names of registration plans for the disturbance recorder:

- operation of fault protection device (cause for triggering No. 2)
  - closing switch (cause for triggering No. 31)
- in the ASMEN channel of the name KTR\_06:

KTR\_06, Z, 2, Pole06ZadzZabezpZwarciewego, operation of fault protection device  
KTR\_06, Z, 31, Pole06ZamkWylacznika, closing switch

## Default Names for Logging Plans

If the user did not define its own names, the driver creates a default name.

The syntax of name for disturbance loggers:

### **NrxxxipyyyCause**

where:

xxx - nuber of device in MODBUS network,  
 yyy - youngest element of device IP address,  
 Cause - shortcut to identify the cause of logger release.

The syntax of name for criterion loggers:

### **NrxxxkryyyCause**

where:

xxx - nuber of device in MODBUS network,  
 yyy - youngest element of device IP address,  
 Cause - shortcut to identify the cause of logger release.

### **EXAMPLE:**

For a disturbance caused by triggering the user's logic recorder, for the megaMuz device number 2 in the MODBUS network with the IP address of 10.10.12.5, a registration plan of the following name will be created:

Nr002ip005wyzw\_rej\_od\_log\_użytk

## Event Signaling

Event signaling is checked cyclically, checking the frequency is determined by the item Event Check Period. Events read from MultiMuz are converted to the appropriate Asix system alarm numbers. This conversion is performed on the basis of mapping presented in the file NumeryAlarmowAsixaVerxx.xls and alarm offset presented in the ASMENA channel declaration. The following event parameters (according to event type) are transferred to Asix alarms:

a/ 01H - *int* and *word* (for events marked with (\*1) text describing the stages is transferred instead of *word*)  
 b/ 02H - *int* and *word*  
 c/ 04H - *int* and *word*  
 d/ 08H - *int* and *word* (for events marked with (\*1) *word* contains event cause ID)  
 e/ 11H - *word*  
 f/ 12H - *word*  
 g/ 14H - *word*  
 h/ 21H (\*2) (\*3) - switch number, ID of control  
 i/ 21H (\*2) - switch number  
 j/ 21H (\*3) - ID of control  
 k/ 21H - *word*

l/ 52H - *word*  
t/ 54H - *word*  
m/ 92H - *word*  
n/ 94H - *word*

**EXAMPLE:**

The event Closing switch from controller (event of 21H type with the code 161) is mapped into Asix alarm No. 190. If an alarm offset of 400 is specified in the ASMEN channel declaration, then the alarm definition in the Asix alarm definitions file will have the following form (400 + 190 = 590):

**590, al, closing switch from controller**

## Driver Parameters

The MultiMuz3 driver parameters are declared on *Driver Parameters* tabs.

**Log File**

Meaning

- the diagnostic purposes are fulfilled by a text log file, where the driver operation status messages are entered.

Default value

- by default, the log file is not created.

Parameter:

*file\_name*

**Log File Size**

Meaning

- option is used to determine the size of the log file defined with the use of the *Log file* option.

Parameter:

*number*

- log file size in MB.

Default value

- default log file size is 10 MB.

**Log of Telegrams**

Meaning

- option allows the contents of telegrams sent between the driver and controllers to the log file (declared with use of *Log file* option) to be saved. The subject option should be used only during the start-up of the Asix system.

Default value

- by default, the option is disabled.

**Disturbances Log File**

Meaning

- the diagnostic purposes are fulfilled by a text log file, where the disturbance messages are entered.

Default value

- by default, the log file is not created.

Defining

- manual.

Parameter:

*file\_name*

**Registration of Disturbances in AsLogger Database**

Meaning - the option switches the driver to disturbance logging mode. This option defines the server name and the database name, in which the disturbance samples will be logged.

Default value - by default, the disturbances are not logged.

Parameters:

*ServerName* - MSSQL server name  
*BaseName* - base name on *ServerName* server

**Directory with Files Containing Registration Plan Names**

Meaning - the item allows to specify the full path to the directory with files containing definitions of disturbance plan names.

Parameters:

*path* - full path to the directory

**Number of Repetitions**

Meaning - the item allows to define maximal number of trials to do the command in case of transmission errors.

Default value - 3.

Defining - manual.

Parameters:

*number* - number of repetitions in case of transmission error.

**Response Timeout**

Purpose - this option specifies the maximum time throughout which the driver waits for a response from the device.

The default value - the default time out value is 1000 milliseconds.

Parameter:

*number* - time out in milliseconds.

**Char Timeout**

*Purpose* - this option specifies the time out (in milliseconds) between sending a request to a device and initiating a response reception operation. The use of this option makes sense only if the response is highly fragmented.

The default value - the default reception timeout value is 50 milliseconds.

Parameter:

*number* - reception timeout value in milliseconds.

**Event Check Period**

Meaning - option declares the interval (in seconds) between reading the event log of the subsequent device connected to the same serial port supported by the driver.  
Default value - by default the option value is 10 seconds.

**No Receive Delay**

*Purpose* - the time throughout which requests to be sent to the driver will be blocked after the last request did not return a response.

*Parameter:*

*number* - time in milliseconds.  
The default value - 0 - no blocking.

## Channel Parameters

The Parameters Defined for Individual Channels Using the **Channel Parameters...** tabs:

**Response Timeout**

Meaning - this option specifies the maximum time throughout which the driver waits for a response from the device.

The default value - the default time out value is 1000 milliseconds.

Parameter:

*number* - time out in milliseconds.

**Char Timeout**

*Meaning* - this option specifies the time out (in milliseconds) between sending a request to a device and initiating a response reception operation. The use of this option makes sense only if the response is highly fragmented.

The default value - the default reception timeout value is 100 milliseconds.

Parameter:

*number* - reception timeout value in milliseconds.

**Without Events**

Meaning - this option is used to disable event readout in a given channel.

The default value - no readout restrictions.

**No Receive Delay**

Meaning - the time throughout which requests to be sent to the driver will be blocked after the last request did not return a response.

Parameter:

*number* - time in milliseconds.  
The default value - 0 - no blocking.

**Switch 1, 2, ...7**

*Purpose*

- this option is used to specify the names of the switches which will be transferred to the alarm system. The name of a switch cannot be longer than 3 characters. Names of up to 7 switches can be defined with numbers assigned from 1 to 7.

*Parameter:*

*text*

The default value

- 3-character switch name.

- by default, the names of switches are 1,2,3, ...7.

 **Log File**

*Purpose*

- the text log file to which channel status messages are stored is used for diagnostic purposes.

The default value

- none.

 **Log File Size**

*Purpose*

- this option is used to determine the log file size defined using the *Log file* option.

*Parameter:*

*number*

The default value

- the size of the log file in MB.

- by default, the log file size is 10 MB.

 **Log of Telegrams**

*Purpose*

- this option allows writing the contents of messages communicated in the channel to the log file (declared using the *Log File* item). The option in question should only be used during the Asix system start-up.

The default value

- by default, this option is disabled.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

 **Remote Write Access**

*Purpose*

- this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the

## Communication Drivers

names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose

- with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose

- with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value

- NO.

### **Data Validity Period**

Purpose

- this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value

- by default, no time for data validity is set.

## 1.73 MultiMuz3\_tcpip - Driver for MultiMUZ Microprocessor-Based Security Devices from JM- Tronik

### Driver Use

The MultiMuz3\_TCPIP protocol driver is used to exchange data between the Asix system and the MultiMUZ3 microprocessor-based security devices manufactured by Warsaw-based JM-Tronik. Communication is executed in MODBUS RTU mode on TCPIP with the use of Ethernet and port 10502.

The driver executes the following functions:

- reading current statuses and measurements,
- reading events and reporting them to the Asix alarms system,
- reading troubles and logging them in the database in the format of the DataLogger application, developed by ASKOM Sp. z o.o.

The driver supports the following MultiMuz3 device types:

- 01.07
- 01.09
- 10.02
- 10.11
- 10.12
- 23.01
- 26.01
- MegaMuz2 01.12 (supported in MultiMuz3 mode)
- 01.12M

The parameterization of the MultiMuz3\_TCPIP driver is performed by the Architect program.

### Declaration of Transmission Channel

The declaration of the transmission channel operating according to MultiMuz3\_TCPIP protocol requires a channel with the following parameters to be added to the Current data module:

**Standard** tab:

*Name:* logical name of the transmission channel

*Driver:* MultiMuz3\_TCPIP

**MultiMuz3\_TCPIP/Channel Parameters** tab:

**Device Number** - network device number,

**Device Type** - device type.

**IP Address of the Device** - IP address of device.

**Alarms Offset** - offset added to the calculated Asix alarm number,

**Control Variables**

- variable used to show the communication status with device:

- 0 - transmission o.k.
- 1 - no communication/communication error
- 2 - disabling the support for device,

- variable used to control the communication status with device:

- 1 - request to disable the support for device,
- 0 - enabling the support for device.

**Time Synchronization**

- time synchronization period with a device (in seconds). 0 means no synchronization (default value).

**IP Address of the Computer** - IP address of computer network card through which the communication in a given channel will be realized.

**EXAMPLE**

An exemplary declaration of the channel KSR\_67, in which MultiMuz3 with network number 5, address IP 10.10.12.5 and alarm offset 200 is supported. The channel state is given by the variable MuzStat05, the channel control is possible by the variable MuzCtrl05 and synchronization is performed every 60 seconds:

*Channel name:* KSR\_67

*Device Number:* 5

*IP Address of the Device:* 10.10.12.5

*Alarms Offset:* 200

*Variable used to show the communication status with device:* MuzStat05

*Variable used to control the communication status with device:* MuzCtrl05

*Time Synchronization:* 60

## Variable Declaration

The driver provides the following variable types of MODBUS protocol:

- CS - binary value (writing),
- HR - 16-bit log (reading)
- HRF - 32-bit log of FLOAT type (reading)
- HRL - 32-bit log of DWORD/LONG type (reading)

and internal driver variable:

- RS - reason for logger triggering (reading)
- RQ - logger reading request (writing),
- RT - logger triggering time (reading)
- MN - network number of MultiMuz, from which the logger is/was read (reading)
- RN - number of currently/recently read logger (reading)
- RR - status of currently/recently read logger (reading)
- SN - number of recently read logger sample (reading)

RSKR - reason for release of criterion logger (reading)  
 RQKR - request for reading a criterion logger (writing)  
 RTKR - release time of criterion logger (reading)  
 MNKR - network number of the device from which criterion logger is/was read (reading)  
 RNKR - number of currently/recently read out criterion logger (reading)  
 RRKR - status of currently/recently read out criterion logger (reading)  
 SNKR - number of recently read out sample of criterion logger (reading)  
 UDC - type for executing commands of MODBUS RTU protocol defined by the user. Available only for the keys: WAUS, WAUW, WDUW or WDUN. UDC variable value is ASCII string with a maximum length of 521 bytes. When writing it is the content of the command created by a user, when reading it is the content of the response to recent sent user's command. (reading/writing)

**The variable address has the following syntax:**

*Type[Index]*

where:

*Type* - variable type,  
*Index* - variable index within 'Type' variable type.

**NOTE** Variables: the variable used to monitor the channel status and the variable used to disable the channel support must belong to a channel of NONE type.

1/ indexes are not given for types: MN, RN, RR, RQ, SN, MNKR, RNKR, RRKR, RQKR and SNKR;  
 2/ in the case of RS, RT, RSKR, RTKR types, the range of indexes depends on the number of loggers defined in the device;

3/ the parameter of the logger reading request (RQ, RQKR type) is the logger number. Loggers are numbered, starting from 1. If 0 is given as a parameter, it is equivalent to the request of stopping the currently executed logger reading;

4/ in order to present the logger triggering time, use the conversion function DATACZAS\_MUZ, which converts the double-type number to a character string in the following format:

yyyy-mm-dd hh:nn:ss.zzz

#### **EXAMPLE**

Examples of variable declaration:

ZM\_01, RMS value of current I1, HRL3, KLR, 1, 1, NOTHING\_FP  
 ZM\_02, RMS value of voltage U1, HRL27, KLR, 1, 1, NOTHING\_DW  
 ZM\_03, release of protections 1, HRL95, KLR,1,1, NOTHING\_DW  
 ZM\_04, change of bank of settings to bank1, CS17, KLR,1,1, NOTHING  
 ZM\_05, release time of disturbance logger 1, RT1, KLR, 1,1, DATETIME\_MUZ  
 ZM\_06, request of reading disturbance logger, RT1, RQ, KLR,1,1,NOTHING  
 ZM\_07, number of disturbance logger being read, RN, KLR,1,1,NOTHING  
 ZM\_08, status of disturbance logger readout, RR, KLR,1,1, NOTHING

ZM\_09, number of sample read from disturbance logger,  
SN, KLR\_01,1,1, NOTHING

ZM\_10, request of reading criterion logger, RQKR, KLR,1,1, NOTHING

ZM\_11, number of criterion logger being read, RNKR, KLR,1,1, NOTHING

ZM\_12, status of reading criterion logger, RRKR, KLR,1,1, NOTHING

ZM\_13, number of sample read from criterion logger, SNKR, KLR\_01,1,1, NOTHING

## Readout of Loggers

The driver enables the automatic logging of troubles in the AsLogger application database. In order to activate the trouble logging mode, declare the option Registration of Disturbances in AsLogger Database in the driver parameters, which defines:

- MS SQL server name,
- database name,

where the driver will save the read trouble samples.

(see more in: Driver Parameters)

The driver does not perform an automatic trouble logging - the logging must be initiated by the Asix system operator each time by the execution of control with the use of the RQ-type or RQKR-type variable. As the control value, give the number of the logger, whose contents should be read from MultiMuz and saved to the AsLogger program base. If the control value is 0, it is interpreted as the request to stop the currently performed logger reading.

During logger reading, it is possible to track the reading status. This is executed with the use of the internal driver variables of symbolic addresses:

- a/ MN, MNKR - number of read MultiMuz device,
- b/ RN, RNKR - number of read logger,
- c/ RR, RNKR - logger reading status:
  - 0 - inactive or completed ok.,
  - 1 - reading in progress,
  - 2 - reading terminated with error.
- d/ SN, SNKR - number of recently read sample.

Trouble logging is performed in accordance to the logging plans. The names of the logging plans can be:

- 1/ defined by the user;
- 2/ generated automatically by the driver.

When registering, the driver writes into the AsLogger database:

- a/ all parameters contained in the record of logger, described in the documentation protocol of version 01.07;

b/ creates an archive plan;

c/ sets max. and min. values for currents and voltages in registered data series - they will be corrected during registration of consecutive series for a given plan.

## Names of Logging Plans Defined by the User

The user can define names of registration plans in text files with the extension '.def'. To do it, for each transmission channel (ASMEN channel) a separate file should be created and definitions of names for logging plans for disturbances logged in a given channel should be written into this file. Files with definitions of plan names should be written into a common directory. The name of this directory should be placed in the item **Directory with Files Containing Registration Plan Names** in the driver parameters. (See more in: Driver Parameters)

The syntax of definitions for registration plan placed in the file '.def' is as follows:

***channel, type, cause, table [, comment]***

where:

<i>channel</i>	- name of ASMEN channel,
<i>type</i>	- type of logger (K - criterion, Z - disturbances),
<i>cause</i>	- number of cause (compatible with the specification given in the description of protocol of ver. 01.07)
<i>table</i>	- name of logging plan,
<i>comment</i>	- comment

### EXAMPLE:

KTR\_06, Z, 2, Pole06ZadzZabezpZwarcowego, zadziałanie zabezpieczenia zwarcowego

KTR\_06, Z, 31, Pole06ZamkWylacznika, zamknięcie wyłącznika

## Default Names for Logging Plans

If the user did not define its own names, the driver creates a default name.

The syntax of name for disturbance loggers:

***NrxxxipyyyCause***

where:

xxx	- nuber of device in MODBUS network,
yyy	- youngest element of device IP address,
Cause	- shortcut to identify the cause of logger release.

The syntax of name for criterion loggers:

***NrxxxkryyyCause***

where:

- xxx - number of device in MODBUS network,
- yyy - youngest element of device IP address,
- Cause - shortcut to identify the cause of logger release.

**EXAMPLE:**

For disturbance caused by logger release in MultiMuz3 no. 2 in MODBUS network and IP address: 10.10.12.5 the following logging plan name will be created:

**Nr002ip005wyzw\_rej\_od\_log\_uzytk**

## Event Signaling

Event signaling is checked cyclically, checking the frequency is determined by the item Event Check Period. Events read from MultiMuz are converted to the appropriate Asix system alarm numbers. This conversion is performed on the basis of mapping presented in the file NumeryAlarmowAsixaVerxx.xls and alarm offset presented in the ASMENA channel declaration. The following event parameters (according to event type) are transferred to Asix alarms:

- a/ 01H - int and word (for events marked with (\*1) text describing the stages is transferred instead of word)
- b/ 02H - int and word
- c/ 04H - int and word
- d/ 08H - int and word (for events marked with (\*1) word contains event cause ID)
- e/ 11H - word
- f/ 12H - word
- g/ 14H - word
- h/ 21H (\*2) (\*3) - switch number, ID of control
- i/ 21H (\*2) - switch number
- j/ 21H (\*3) - ID of control
- k/ 21H - word
- l/ 52H - word
- ł/ 54H - word
- m/ 92H - word
- n/ 94H - word

**EXAMPLE:**

Event Closing switch from controller (event of 21H type with the code 161) is mapped to the Asix alarm number of value 190. If in the declaration of the ASMENA channel supporting the given device, the alarm offset of value 400 is given, the alarm definition in the Asix alarm definition file will have the form of (400 + 190 = 590):

*590, al, closing switch from controller*

## Driver Parameters

The MultiMuz3\_TCPIP driver parameters are declared on **Driver Parameters...** tabs.

### **Log File**

Meaning - the diagnostic purposes are fulfilled by a text log file, where the driver operation status messages are entered.

Default value - by default, the log file is not created.

Parameter:

*file\_name*

### **Log File Size**

Meaning - option is used to determine the size of the log file defined with the use of the Log file option.

Parameter:

*number*

Default value - default log file size is 10 MB.

### **Log of Telegrams**

Meaning - option allows the contents of telegrams sent between the driver and controllers to the log file (declared with use of Log file option) to be saved. The subject option should be used only during the start-up of the Asix system.

Default value - by default, the option is disabled.

### **Disturbances Log File**

Meaning - the diagnostic purposes are fulfilled by a text log file, where the disturbance messages are entered.

Default value - by default, the log file is not created.

Parameter:

*file\_name*

### **Number of Repetitions**

Meaning - the item allows to define maximal number of trials to do the command in case of transmission errors.

Default value - 3.

Parameters:

*number*

- number of repetitions in case of transmission error.

**Event Check Period**

- Meaning - option declares the interval (in seconds) between reading the event log of the subsequent device connected to the same serial port supported by the driver.
- Default value - by default the option value is 10 seconds.

**Registration of Disturbances in AsLogger Database**

- Meaning - the option switches the driver to disturbance logging mode. This option defines the server name and the database name, in which the disturbance samples will be logged.
- Default value - by default, the disturbances are not logged.
- Parameters:
- ServerName* - MSSQL server name
  - BaseName* - base name on ServerName server

**Directory with Files Containing Registration Plan Names**

- Meaning - the item allows to specify the full path to the directory with files containing definitions of disturbance plan names.
- Parameters:
- path* - full path to the directory

**EXAMPLE**

Example of driver parameters:

```
Log File =d:\tmp\multimuz\muz.log
Log File Size =3
Log of Telegrams =YES
Registration of Disturbances in AsLogger Database=SERWER_RO6, BAZA_RO6
```

## Channel Parameters

The MultiMuz3\_TCPIP driver parameters are declared on **Channel Parameters...** tabs.

**Port**

- Meaning - the item allows to define the number of computer port through which the communication in a given channel will be realized.
- Default value - the port assigned by Windows.
- Parameters:
- number* - number of computer port.

**☑ Response Timeout**

Meaning - the item allows to determine a maximal waiting time between sending a query and receiving an answer (so called receiving timeout). The parameter value is passed for all handled MUZ-RO devices globally.

Default value - 500.

Parameters:  
*number* - timeout value in milliseconds.

**☑ Response Delay**

Meaning - the item allows to determine a maximal delay time between sending a query and receiving an answer.

Default value - 20.

Parameters:  
*number* - value in milliseconds.

**☑ Without Events**

Meaning - the item allows to disable events read for a current channel.

Default value - 20.

**☑ Number of Repetition**

Meaning - the item allows to define maximal number of trials to do the command in case of transmission errors.

Default value - set in driver section.

Parameters:  
*number* - number of repetitions in case of transmission error.

**☑ Max Number of Registers**

Meaning - the item allows to define maximal number of registers which the driver may ask for in one command in a given channel.

Default value - 119.

Parameters:  
*number* - max. number of registers.

**☑ Switch name 1, 2, ...7**

Meaning - opcja służy do określenia nazw łączników, które zostaną przekazane do systemu alarmów. Nazwa łącznika nie może być dłuższa niż 3 znaki. Można zdefiniować nazwy dla max 7-miu łączników o numerach od 1 do 7.

Default Value - default names: 1,2,3, ...7.

Parametr:  
*text* - switch name (max 3 characters).

**☑ Log File**

Meaning - the diagnostic purposes are fulfilled by a text log file, where the driver operation status messages are entered.

Default value - by default, the log file is not created.

Parameter:  
*file\_name*

## Communication Drivers

### **Log File Size**

Meaning - option is used to determine the size of the log file defined with the use of the Log file option.

Parameter:  
*number* - log file size in MB.

Default value - default log file size is 10 MB.

### **Log of Telegrams**

Meaning - option allows the contents of telegrams sent between the driver and controllers to the log file (declared with use of Log file option) to be saved. The subject option should be used only during the start-up of the Asix system.

Default value - by default, the option is disabled.

## EXAMPLE

[CHANNEL]

Log File =c:\tmp\channel\_1.log

Log File Size =30

Log of Telegrams = YES

IP Address of the Computer =10.10.112.8

Max Number of Registers =53

\*\*\*

Some channel parameters are declared directly in the *Current data > <MultiMuz3\_TCPIP driver channel name>* > module in the *Advanced* tab:

### **Remote Write Access**

Meaning - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Meaning - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\* \* \*

Some channel parameters are declared directly in the *Current data > <MultiMuz3\_TCPIP driver channel name>* > module in the *Advanced 2* tab:

 **Local Write Denied**

Meaning - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

 **Data Validity Period**

Meaning - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - By default, no time for data validity is set.

## 1.74 MUPASZ - Driver of MUPASZ Device Protocol

### Driver Use

The MUPASZ driver is used for data exchange between MUPASZ or MUPASZ2000 devices and an Asix system computer. The communication is performed with use of serial interfaces in the RS232C or RS485 standard.

Parameterization of MUPASZ driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the MUPASZ driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: MUPASZ

**MUPASZ/Channel parameters** tab:

*Remote device number* - number assigned to the remote device;  
*Serial port* - name of the serial port;  
*Signalization mode...* - number added to the event number with **signalization** executing mode in order to build a unique number of the alarm transferred to the Asix system;  
*Off mode event offset...* - number added to the event number with **shutdown** executing mode in order to build a unique number of the alarm transferred to the Asix system;  
*Block mode event offset...* - number added to the event number with **shutdown with lock** executing mode in order to create a unique number of the alarm transferred to the Asix system.

#### EXAMPLE

The declaration of the logical channel named CHAN1, which works according to the MUPASZ protocol and has parameters as below:

- number of remote device - 4
  - port - COM1
  - number added to the number of event **signalization** - 100
  - number added to the number of event **shutdown** - 200
  - number added to the number of event **shutdown with lock** - 300
- is as follows:

*Channel / Name*: CHAN1  
*Driver*: MUPASZ  
*Remote device number*: 4  
*Serial port*: COM1  
*Signalisation mode...*: 100  
*Off mode event offset...*: 200  
*Block mode event offset...*: 300

The MUPASZ driver is loaded as a DLL automatically.

## Addressing the Process Variables

The syntax of symbolic address which is used for variables belonging to the MUPASZ driver channel is as follows:

`<variable_type><channel>.<index>`

where:

*variable\_name* - process variable type,  
*index* - index of a process variable within the type.

Types of process variable:

P - measurement values (FLOAT),  
 B - state of locks arriving with measures (WORD),  
 L - values of counters (WORD),  
 F - status arriving with measurements and events (WORD).

### EXAMPLE

Example of declarations of variables:

X1, current Io,	P1, CHAN1, 1, 1, NOTHING_FP
X2, I1 accum.,	P31, CHAN1, 1, 1, NOTHING_FP
X3, state of blockade no. 1,	B1, CHAN1, 1, 1, NOTHING
X4, counter of switch openings,	L1, CHAN1, 1, 1, NOTHING
X5, counter of motor startups,	L25, CHAN1, 1, 1, NOTHING
X6, switch state,	F1, CHAN1, 1, 1, NOTHING

## Generating Alarms

Numbers of events generated by a remote device have the same range of variation. In order to be able to uniquely determine which device the event comes from, the MUPASZ driver adds to the event number the number specified in the channel declaration as AISygOf (for signalization), AIWy!Of (for shutdowns) or AIBlokOf (for shutdowns with lock). In such way this number is transferred to the Asix system as an alarm number.

For some alarms the MUPASZ driver may transfer values coming with events (activation time or current). These values may be read by giving a formatting string (%3.0f) in the definition of an alarm message.

In order to transfer alarms the MUPASZ driver uses the function *AsixAddAlarmGlobalMili()* by default. The item GLOBAL\_ALARMS allows to change default settings and to transfer alarms by means of the function *AsixAddAlarmMili()*.

## Driver Parameters

The driver configuration is defined in the *Current Data* module, in the channel operating according to MUPASZ driver.

**Log file**

Meaning - the item allows to define a file to which all diagnostic messages of the MUPASZ driver and information about contents of telegrams received and sent by the MUPASZ driver will be written. If the item does not define the full path, then the log file is created in the current directory. The log file should be used only while the Asix start-up.

Default value - by default, the log file is not created.

Defining - manual.

**Transmission delay**

Meaning - the item allows to specify the time period (as a multiple of 10 msec) between successive operations on the MUPASZ bus.

Default value - by default, the item assumes a value of 1 (10 msec).

Defining - manual.

**Number of repetitions**

Meaning - the item allows to specify a number of repetitions in case of a transmission error.

Default value - by default, the item assume a value of 0 (no repetitions).

Defining - manual.

**Data update**

Meaning - the item allows to define the time period (in seconds), after which the driver should update values of process variables stored in internal driver buffers.

Default value - by default, the item assumes a value of 1.

Defining - manual.

**Current time update**

Meaning - the item allows to define a time period (in seconds), after which an actual time should be sent to remote devices.

Default value - by default, the item assumes a value of 60.

Defining - manual.

**Global alarms**

Meaning - the item controls the way of transferring alarms read from remote devices to the Asix alarm system.

Default value - by default, the alarms are transferred to the alarm system as global alarms (transferred by means of the function *AsixAddAlarmGlobalMili()*). Setting the GLOBAL\_ALARMS item value causes that alarms are transferred to the alarm system by means of the function *AsixAddAlarmMili()*.

Defining - manual.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

- Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.
- Option value:  
 YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

- Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.
- Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

- Purpose - with this option deployed, local control in a channel will be disabled.
- Option value:  
 YES / NO
- The default value - NO.

### **Data Validity Period**

- Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.
- Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.
- The default value - by default, no time for data validity is set.

## 1.75 Mupasz710\_RS - Driver of Microprocessor Protecting Devices of MUPASZ 710 Type

### Driver Use

The Mupasz710\_RS is used for data exchange between Asix and the microprocessor protecting devices of MUPASZ 710 type made by Instytut Tele- i Radiotechniczny in Warsaw. The communication is made with use of serial link and the MODBUS protocol.

The driver realizes the following functions:

- readout of current states and measurements,
- readout of events and reporting them to the Asix alarm system,
- readout of interferences from the recorder and their registration in the AsLogger database (the software developed by Askom).

#### Notice:

Using the functions of the recorder readout and interference registration in AsLogger database needs the appropriate Asix system license.

### Declaration of Transmission Channel

Declaration of transmission channel using the Mupasz710\_RS driver requires a channel with the following parameters to be added to the *Current data* module:

**Standar** tab:

**Name:** logical name of the transmission channel  
**Driver:** Mupasz710\_RS

**Mupasz710\_RS/Channel parameters:**

#### Address

**Device number in the MODBUS network,**

#### Transmission parameters

**Port** - serial port name,

**Baud rate** - baud rate. By default: 115200 Bd,

**Parity bit checking**

**Number of stop bits**

#### Alarms offset

Offset added to the calculated alarm number sent to the Asix system.

#### EXAMPLE

The declaration of the logical channel named CHANNAL, which works according to the Mupasz 710 protocol and has parameters as below:

- *Device number...*: 2,
- *Port*: COM3,
- *Alarms offset*: 200,
- *Baud rate*: 9600,
- *Parity bit checking*: NONE,

- Number of stop bits: 1

## Declaration of Process Variables

Types of process variable for MODBUS protocol:

HR	- 16-bit register	(readout)
HRF	- 32-bit register of FLOAT type	(readout)
HRL	- 32-bit register of DWORD/LONG type	(readout)
UDC	- used to execute the MODBUS RTU protocol commands defined by the user, available only for keys WAUS, WAUW, WDUW or WDUN. The value of a UDC type variable is a string of ASCII characters, whose maximum length is 521 bytes. While writing, it is the content of a command defined by the user; while reading, it is the content of a response to a recently sent user's command. (read/write)	

and internal driver variables:

MN	- network number of device from which sector is being read or has been read	(readout)
RN	- number of currently/recently read sector	(readout)
RQ	- sector readout request	(write)
RR	- status of currently/recently read sector	(readout)
RS	- state of recorder (unavailable/empty/written)	(readout)
RT	- release time of interference written in a sector	(readout)
SN	- current counter of HR registers read from a sector	(readout)

**The syntax of symbolic address:**

***Type[Index]***

where:

*Type* - process variable type,  
*Index* - index of a process variable within the *Type*.

### REMARKS:

1. indexes are not declared for the types: MN, RN, RQ, RR i SN
2. for the types: RS, RT the range of indexes depends on the number of sectors defined in the device,
3. the parameter of sector readout request (RQ type) is the sector number; the sectors are numbered w.e.f. 1. If 0 is set as the parameter, that means the interrupt request of sector current readout,
4. to present the sector release time, use the conversion function DATETIME\_MUZ, that convert the number of double type into the following string:

**yyyy-mm-dd hh:nn:ss.zzz**

### EXAMPLE

Example of variable declarations:

```
ZM_01, face value Un, HR8235, KLR, 1, 1, NOTHING
ZM_02, face value In, HR8236, KLR, 1, 1, NOTHING
ZM_03, release time in the sector 1, RT1, KLR, 1, 1, DATETIME_MUZ
```

ZM\_04, sector readout request RQ, KLR, 1, 1, NOTHING  
ZM\_05, number of read sector, RN, KLR, 1, 1, NOTHING  
ZM\_06, status of sector readout, RR, KLR, 1, 1, NOTHING  
ZM\_07, counter of registers read from the sector SN, KLR, 1, 1, NOTHING\_DW

## Interference Recorder Readout

The driver allows for readout of interferences from the recorder and their registration in the AsLogger database. The registered trends can be reviewed only by AsLogger software.

The recording of interference recorder trends demands declaration of *Registration of disturbances in AsLogger database* option (*Mupasz710\_RS* tab / *Driver parameters - AsLogger*). It contains:

- MSSQL server name,
- database name,

where the trends will be written.

The driver does not register the trends automatically - registration has to be initiated any time by the Asix system operator by execution of control with the use of RQ type variable. The control value is the number of sector the value of which should be read from a device and written to AsLogger database. If control value equals 0, it means the interrupt request of sector current readout.

While the sector readout the readout state tracking is possible with the use of driver internal variables of symbolic addresses:

a/ *MN* - device number in MODBUS network,  
b/ *RN* - number of read sector,  
c/ *RR* - status of sector readout:

0 - readout is not active or finished OK  
1 - readout is in progress  
2 - readout of the latest frame finished OK  
3 - readout of the latest frame finished with error  
4 - readout of the sector finished with error  
5 - readout of the sector finished OK  
6 - readout of the sector stopped by the operator

d/ *SN* - current counter of registers read from the sector.

The recording of register trends is realized according to recording plans. The names of recording plans may be:

a/ defined by the user in text files which are read by the driver at application start-up,  
b/ generated automatically by the driver.

While the trend recording, the driver writes to AsLogger database the following items:

a/ all the parameters from the sector,

b/ creates the plan archive,  
 c/ determines max and min values for currents and voltages in recorded trend - if need they will be corrected during reading the successive trends for the given plan.

The **Number of repetitions of disturbances registrator** parameter (set on *Driver parameters - AsLogger* tab) determines the number of repetitions in the case of transmission error.

- **Defining the Names of Recording Plans by the User**

The user can define the names of recording plans in text files with the extension '.def'. To do this for each communication channel (channel of Asmen), one should create the separate file and write to them the definitions of recording plan names for interferences recorded in this channel. The files with the plan definitions should be located in the common directory the name of which should be declared as the **Disturbance names files path** option (*Mupasz710\_RS* tab / *Driver parameters - AsLogger*).

The syntax of definition for recording plan name defined in '.def' file is as follows:

***Channel, Cause, FormatAi, FormatDi, Table [, Commentary]***

where:

<i>Channel</i>	- Asmen channel name,
<i>Cause</i>	- number identifying the cause of recorder release,
<i>FormatAi</i>	- analog measurement mask in HEX notation (6 digits),
<i>FormatDi</i>	- binary input mask in HEX notation (2 digits),
<i>Table</i>	- recording plan name,
<i>Commentary</i>	- kommentary (option)

**Example:**

Definition for recording plan for recording interferences with the following parameters:

- voltage value U1 < (release cause number 1),  
 - analog mask 0000FF - recording U1, U2, U3, U0 and I1, I2, I3, I0  
 - binary values mask 0001 - recording input states 1 - 16  
 in the channel KTR:

**KTR, 1, 0000FF, 0001, Field05VoltageDropU1**

- **The Default Recording Plan Names**

If the user does not define its own recording plan names, the driver will create the name of recording plan automatically.

The syntax of interference recording plan name generated by the driver is as follows:

**ChannelName.Cause.FormatAi.FormatDi**

where:

ChannelName	- name of Asmen channel in which the interferences were read,
Cause	- identifier of recording release cause,

- FormatAi - analog measurement mask in HEX notation (6 digits),
- FormatDi - binary input mask in HEX notation (2 digits).

**EXAMPLE**

For interference caused by the recorder release from the user's logic in the device operated in Asmen channel named CHANNEL with the registration set as follows:

- analogs: U1, U2, U3, U0, I1, I2, I3, IO
  - and
  - binary measurements: inputs 1-16, inputs 17-32, outputs 1-16
- the following recording plan name will be defined:

**CHANNEL.29.000FF.0023**

- **Identifiers of Recorder Release Reasons**

Below the set of recorder release reasons:

- |    |         |  |
|----|---------|--|
| 01 | U1LE    | voltage U1 as a parameter                      |
| 02 | U2LE    | voltage U2 as a parameter                      |
| 03 | U3LE    | voltage U3 as a parameter                      |
| 04 | U0LE    | voltage Uo as a parameter                      |
| 05 | I1LE    | current I1 as a parameter                      |
| 06 | I2LE    | current I2 as a parameter                      |
| 07 | I3LE    | current I3 as a parameter                      |
| 08 | IOLE    | current Io as a parameter                      |
| 09 | I1LESEC | current I1sec as a parameter                   |
| 10 | I2LESEC | current I2sec as a parameter                   |
| 11 | I3LESEC | current I3sec as a parameter                   |
| 12 | IOLESEC | current Iosec as a parameter                   |
| 13 | U1GT    | voltage U1 as a parameter                      |
| 14 | U2GT    | voltage U1 as a parameter                      |
| 15 | U3GT    | voltage U1 as a parameter                      |
| 16 | U0GT    | voltage Uo as a parameter                      |
| 17 | I1GT    | current I1 as a parameter                      |
| 18 | I2GT    | current I2 as a parameter                      |
| 19 | I3GT    | current I3 as a parameter                      |
| 20 | IOGT    | current Io as a parameter                      |
| 21 | I1GTSEC | current I1sec as a parameter                   |
| 22 | I2GTSEC | current I2sec as a parameter                   |
| 23 | I3GTSEC | current I3sec as a parameter                   |
| 24 | IOGTSEC | current Iosec as a parameter                   |
| 25 | DIN     | release input number as a parameter (1..80)    |
| 26 | DOUT    | release output number as a parameter (1..80)   |
| 27 | SIG     | release register number as a parameter (1..80) |
| 28 | OIN     | release input number as a parameter (1..8)     |
| 29 | USER    | register release by the user:                  |
- parameter 0 - local, 1 - remote

- **Analog Measurement Mask**

There are the set of bits of analog measurement mask below, which may be recorded in interference recorder (lists **LIST\_8** and **LIST\_9** in **list.xml**).

- |    |        |
|----|--------|
| U1 | 0x0001 |
| U2 | 0x0002 |
| U3 | 0x0004 |
| U0 | 0x0008 |
| I1 | 0x0010 |
| I2 | 0x0020 |

I3	0x0040
I0	0x0080
I1sec	0x0100
I2sec	0x0200
I3sec	0x0400
I0sec	0x0800
Ia_1	0x1000
Ia_2	0x2000
Ia_3	0x4000
Ia_4	0x8000
Ia_5	0x0010000
Temp_1	0x0020000
Temp_2	0x0040000
Temp_3	0x0080000
Temp_4	0x0100000
Temp_5	0x0200000
Temp_6	0x0400000
I0sec	0x0800000

- **Binary Measurement Mask**

There are the set of bits of digital measurement mask below, which may be recorded in interference recorder (lists **LIST\_10** in **list.xml**).

IN_01_16	0x0001
IN_17_32	0x0002
IN_33_48	0x0004
IN_49_64	0x0008
IN_65_80	0x0010
OUT_01_16	0x0020
OUT_17_32	0x0040
OUT_33_48	0x0080
OUT_49_64	0x0100
OUT_65_80	0x0200
SIG_01_16	0x0400
SIG_17_32	0x0800
SIG_33_48	0x1000
SIG_49_64	0x2000
SIG_65_80	0x4000
OIN_01_08	0x8000

## Event Signaling

Signaling events is checked periodically, the frequency of check is determined by the global option **Event check period** (see: *Driver parameters*). Event read from the devices are converted to the corresponding numbers in the Asix system alarms.

The conversion is performed based on the following algorithm:

- definitions and the way they are converted are read from the files: **event.xml** and **format.xml**, provided by the device manufacturer,

- driver converts the parameters of the event as it is described in the **event.xml** file w.e.f. no. 1 until depletion of the buffer used to pass event parameters to the Asix alarm system or depletion of the number of event parameters (the driver may report any combination of 16-bit to 32-bit parameters, provided that their total size do not exceed 8 bytes),

- driver converts the value of each parameter according to the rule defined by the format of the parameter described in the format.xml file.

- it is possible to use global files **event.xml** and **format.xml** by entering the names of this files in the global options **Event definitions files** (see: *Driver parameters, Channel parameters 2*) and **Format definitions file** (see: *Driver parameters, Channel parameters 2*).

- it is possible to define the specific files **event.xml** and **format.xml** by entering the names of this files in the given transmission channel options: **Event definitions file** i **Format definitions file**).

- driver searches the files **event.xml** and **format.xml** in the directory the anme of which is given in the global option **Disturbance names files path** (*Mupasz710\_RS* tab / **Driver parameters - AsLogger**).

## Driver Parameters

The driver configuration is defined in the *Current Data* module, in the channel operating according to MupaszRtu\_TCPIP driver. Options are placed on the **Driver parameters...** tabs. These options refer to all the devices.

### **Driver parameters - diagnostics** tab:

#### **Log file**

Meaning: The item allows to define a file where all diagnostic messages of the driver are written.

Default value: By default, the log file is not created.

Option value:  
*log\_file\_name*

#### **Log file size**

Meaning: The item allows to specify the size of log file size defined by the options: *Log file* and *Disturbances log file*.

Default value: 10 MB.

Option value:  
*number* - log file size in MB.

#### **Log of telegrams**

Meaning: The item allows to write to the log file the contents of telegrams sent between the driver and network clients. Writing the contents of telegrams to the log file should be used only while the **Asix** start-up.

Default value: NO.

Option value:  
*YES/NO*

#### **Disturbances log file**

Meaning: The item allows to define a file where all diagnostic messages on read disturbances are written.

Default value: The log file is not created.

Option value:  
*log\_file\_name*

**Driver parameters** tab: **Password**

Meaning: The option defines the password of the user registered in Mupasz 710 device. The password is necessary to read interferences of the recorder. The password entered by this option is a global option and concerned all the devices. The password value can be overloaded by the option declared in the communication channel options.

Default value: "1111".

Option value:  
*password* - four-character text.

 **Number of repetitions**

Meaning: The option defines the number of connection repetitions in the case of response error from a device when reading all types of data (it does not refer to recorder readout). When the number of repetitions is exhausted the connection with the device is closed and re-established.

Default value: 3.

Option value:  
*number* - the number of connection repetitions.

 **Event check period**

Meaning: The option defines the time after which the event register readout should be performed.

Default value: 10.

Option value:  
*number* - number of seconds.

 **Definition files path**

Meaning: The option defines the full path to the directory in which the driver will be searching for files with event definition (**event.xml**) and files with format definitions (**format.xml**).

**Default value:** -.

Option value:  
*path* - full path to the directory.

 **Event definitions file**

Meaning: The option defines the name of XML file with event definition. The option is global one, but may be overloaded by the same option defined in the communication channel parameters.

Option value:  
*file\_name*

 **Format definitions file**

Meaning: The option defines the name of XML file with format definition. The option is the global one, but it may be overloaded by the same option declared in the communication channel parameters.

Option value:  
*file\_name*

**Driver parameters - AsLogger tab:**

**Registration of disturbances in AsLogger database**

Meaning: The option changes over the driver to the mode of writing recorder trend to a database. It concerns samples from disturbance and criterion recorders.

Option value:

*SERVERNAME* - MS SQL SERVER NAME

*DatabaseName* - database name of the ServerName server

Default value: Disturbances are not recorded.

**Number of repetitions of disturbances registrar**

Meaning: The option defines the number of repetitions in case of response error from a device when reading the recorder.

Default value: 0 (no repetition).

Option value:

*number* - number of repetitions.

**Disturbance names files path**

Meaning: The option defines full path to the directory of files with recording plan definitions.

Option value:

*path* - full path to the directory.

## Channel Parameters

The channel parameters are defined in the *Current Data* module, in the channel operating according to MupaszRtu\_TCPIP driver. Options are placed on the *Channel parameters...* tabs. These options refer only to the channel.

**Channel parameters 2 tab:**

**Password**

Meaning: The option defines the password of the user registered in Mupasz 710 device. The password is necessary to read interferences of the recorder. The option may overload the same option declared in the driver parameters.

Default value: By default, the password is defined by the driver option, if not - the following value is taken: "1111".

Option value:

*password* - four-character text.

**Number of repetitions**

Meaning: The option defines the number of query repetitions in the case of response error in that channel. When the number of repetitions is exhausted the connection with the device is closed and re-established. The option may overload the same option declared in the driver parameters.

Default value: By default, the option is defined by the driver option.

Option value:

*number* - the number of connection repetitions.

**Response timeout**

Meaning: The option defines the maximal time of waiting for response from the device.

Default value: 500.

Option value

*number* - timeout in milliseconds.

**Char timeout**

Meaning: The option specifies the maximum time between characters of responses from a device.

Default value: 50.

Option value:

*number* - timeout in milliseconds.

**Event definitions file**

Meaning: The option defines the name of XML file with event definition. The option may overload the same option defined in the driver parameters.

Option value:

*file\_name* - full path to the directory.

**Format definitions file**

Meaning: The option defines the name of XML file with format definition. The option may overload the same option declared in the driver parameters.

Option value:

*file\_name* - full path to the directory.

**Channel parameters - diagnostics** tab:

**Log file**

Meaning: The item allows to define a file where all diagnostic messages of the driver in the given communication channel are written.

Default value: By default, the log file is defined by the driver option.

Option value:

*log\_file\_name*

**Log file size**

Meaning: The item allows to specify the size of log file size defined by the option *Log file* of the given channel.

## Communication Drivers

Default value: 10 MB.

Option value:

*number* - log file size in MB.

### **Log of telegrams**

Meaning: The item allows to write to the log file the contents of telegrams sent between the driver and network clients in the given channel. Writing the contents of telegrams to the log file should be used only while the Asix start-up.

Default value: No

Option value:

YES/NO

### **Examples of Driver Parameters**

*Log file:* c:\tmp\mupasz.log

*Disturbances log file:* c:\tmp\zaklocenia.log

*Log file size:* 20

*Log of telegrams:* YES

**Event check period: 10**

*Registration of disturbances in AsLogger database:* SERWER\_RO6, BAZA\_RO6

*Disturbance names files path:* c:\tmp\DefZaklocen

*Definition files path:* c:\tmp\DefFiles

*Password:* 1234

### **Examples of Channel Parameters**

*Channel > Name:* KANALX

*Log file:* c:\tmp\kanalx.log

*Event definitions file:* kanalx\Event.xml

*Format definitions file:* kanalx\Format.xml

*Password:* 4321

## **Advanced Channel Parameters**

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\* \* \*

The parameters of a channel are declared in the *Advanced 2* tab:

 **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

 **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

# 1.76 MupaszRtu - Driver for Data Exchange with Microprocessor Devices of Mupasz2001G, Mupasz07 and Mupasz Compact G01 Type Developed by ITR, Warsaw

## Driver Use

The Mupasz driver is used to exchange data between the Asix system and Mupasz2001G, Mupasz07 and Mupasz Compact G01 type devices by ITR, Warsaw. Communication takes place by means of serial communication according to RS485 standard.

## Declaration of Transmission Channel

Declaration of the transmission channel operating according to the MupaszRtu driver protocol requires adding a channel with the following parameters to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* MupaszRtu

**MupaszRtu** tab:

*Channel Parameters:*

***nr, typ, defZdarzeń, aIOffset, port, baud, znak, parzystość, stop, maxWeWy, maxReg***

where:

<i>nr</i>	- device number in the Modbus network,
<i>typ</i>	- type of the device, 1 - Mupasz 2001G 2 - Mupasz 07 3 - Mupasz Compact G01
<i>defZdarzeń</i>	- file including the definitions of events (this parameter is used only for the channels supporting Mupasz 07 and Mupasz Compact G01; for the channels supporting Mupasz 2001G this parameter is ignored;
<i>aIOffset</i>	- offset added to the calculated Asix alarm number;
<i>port</i>	- serial port name, e.g.: COM1;
<i>baud</i>	- baud rate;
<i>znak</i>	- number of bits in a character;
<i>parzystość</i>	- parity check;
<i>stop</i>	- number of stop bits;
<i>maxWeWy</i>	- maximum number of inputs/outputs which can be transferred within a single readout from a device;
<i>maxReg</i>	- maximum number of registers which can be transferred within a single readout from a device;

**EXAMPLE**

Example of transmission channel declaration:

Below follows an example of KANAL channel declaration, in which a Mupasz07 controller of the network number 5 is supported. The definitions of events are stored in the \_701L2.def event file, communication is carried out via COM2 port:

*Channel parameters:*

5,2,zdarzen\_701L2.def,200,COM2,115200,8,none,1,112,120

**Process Variables Declaration**

The driver provides the same types of variables which are found in the MODBUS driver; additionally the following types are available:

MN - number of the device from which the recorder is currently being read (WORD),

RN - number of the recorder section currently being read (WORD),

RQ - request for section readout (type used to write only) (WORD),

RN - status of section readout (WORD),

SN - number of elements read out from section (bytes or words) (DWORD).

The above types do not have any parameters.

Examples of variables declarations using specific types:

*# (control value != 0 - request for section readout, 0 - reset of section readout)*

*Name: SECREQ1*

*Description: control of section readout*

*Address: RQ*

*Channel: CHANNEL*

*Number of elements 1*

*Sampling period: 1*

*Conversion function: NOTHING*

*# network number of MUPASZ from which a section is currently being read*

*Name: SECSLV1*

*Description: network number*

*Address: MN*

*Channel: CHANNEL*

*Number of elements 1*

*Sampling period: 1*

*Conversion function: NOTHING*

*# number of section currently being read*

*Name: SECNR1*

*Description: sektor number*

*Address: RN*

*Channel: CHANNEL*

*Number of elements 1*

*Sampling period: 1*

*Conversion function: NOTHING*

*# status of current section readout*

*Name: SECSTAT1*

*Description: sektor status*

*Address:* RR  
*Channel:* CHANNEL  
*Number of elements* 1  
*Sampling period:* 1  
*Conversion function:* NOTHING

# counter of bytes/words read out from section (DWORD !!!)  
*Name:* SECCNT1  
*Description:* counter of elements  
*Address:* SN  
*Channel:* CHANNEL  
*Number of elements* 1  
*Sampling period:* 1  
*Conversion function:* NOTHING\_DW

## Disturbance Recording

The driver provides for automatic disturbance recording in the AsLogger database. In order to activate the disturbance log mode, the **Recorder** option (see below: driver parameters) must be declared, the option specifying:

- MSSQL server name,
  - database name,
  - registration plan name,
- in which the driver will store the disturbance samples read.

Automatic disturbance recording is not provided by default - each time it must be initiated by the Asix system operator by the execution of control with the use of a RQ type variable. The control value is the number of the section, whose content must be read from Mupasz and recorded in the AsLogger database. The control value set to 0 means a request to terminate the currently conducted section readout.

During section readout it is possible to view the readout status. To this end, the driver's internal variables of the below symbolic addresses are used:

- a/ MN - number of the Mupasz read,
- b/ RN - number of the section being read,
- c/ RR - status of section readout:

- 0 - disabled or completed o.k.
- 1 - readout in progress
- 2 - readout aborted due to an error.

- d/ SN - number of elements read out from section (bytes or words depending on Mupasz type).

Disturbance recording is carried out in line with registration plans defined with the use of the AsLogger application. The names of registration plans are provided in the **RECORDER** option (see below: driver parameters).

## Event Signalling

Event signalling is checked cyclically, check frequency is defined by the parameter **EVENT\_CHECK\_PERIOD** (see below: driver parameters).

In case of Mupasz 2001G the events read are converted into respective alarm numbers in the Asix system, in line with the Mupasz 2001G event log documentation.

In case of Mupasz07 and Mupasz Compact G01 devices, the definitions of events are provided in a text file, whose name is specified in the transmission channel definition. It is possible to use the same definition file in many transmission channels. The syntax of an event definition has the following form:

***event\_no, text, par\_1, par\_2 [,option]***

where:

*event\_no* - event number  
*text* - event text  
*par\_1* - type name of the first event parameter  
*par\_2* - type name of the second event parameter  
*option* - optional parameter

The event definition driver uses the following parameters: *nr\_zdarz*, *par\_1* i *par\_2*. They are used to define the conversion method for the parameters transferred with the event. The driver accepts the following parameter type names as the parameter values *par\_1* and *par\_2* (they are compatible with the format

***Typy\_Parametrów\_Zdarzeń\_Format\_D*** (names only in Polish version) of Mupasz family devices):

Brak,  
Kod,  
Numer,  
Temperatura,  
Cos,  
Czas,  
Czas1,  
Ufaz,  
Uo,  
Ifaz,  
Io,  
Iskum,  
Licznik,  
Admitancja,  
Czestotliwosc,  
Rezystancja,  
Moc,  
Operator,  
Rwirnika,  
MocP,  
IrHarm2,  
IrHarm5,  
IfazInTg,  
IfazInPg.

The alarm number transferred by the driver to the Asix system is created by adding the event number received from Mupasz to the offset specified in the declaration of the Asmen transmission channel. Event parameters are converted into FLOAT type and transferred as alarm parameters to the Asix alarm system. Depending on the number of parameters, the alarm transferred to the Asix system includes 0, 1 or 2 parameters.

## Driver Parameters

The MupaszRtu driver parameters are declared in the *Miscellaneous* module, in the *Directly entered options* tab.

In the section called **MUPASZRTU** it is possible to include declaration items:

- creating a log file,
- the log file size,
- log of telegrams,
- event check period,
- event definition path,
- readout of recorders,
- number of repetitions during recorder readout,
- number of repetitions during readout of current data,
- blocking of requests in a channel for a specified period of time after the last request did not return a response,
- number of registers in a single request during recorder readout,
- recorder request timeout.

**Section name:** MUPASZRTU

**Option name:** LOG\_FILE

**Option value:** *log\_file\_name*

Purpose - the text log file to which driver status messages are stored is used for diagnostic purposes.

The default value - by default, the log file is not created.

**Section name:** MUPASZRTU

**Option name:** LOG\_FILE\_SIZE

**Option value:** *number*

Purpose - this option is used to determine the log file size defined using the LOG FILE option.

Option value:  
*number* - the size of the log file in MB.

The default value - by default, the log file size is 1 MB.

**Section name:** MUPASZRTU

**Option name:** LOG\_OF\_TELEGRAMS

**Option value:** YES/NO

Purpose - this option allows writing the contents of messages communicated between the driver and the controllers to the log file (declared using the LOG\_FILE option) The option in question should only be used during the Asix system start-up.

The default value - by default, the option value is set to NO.

**Section name:** MUPASZRTU

**Option name:** EVENT\_CHECK\_PERIOD

**Option value:** *number*

Purpose - this option specifies the interval (in seconds) before reading the event log of a subsequent device connected to the same serial port supported by the driver.

Option value:  
*number* - time declared in milliseconds.

The default value - the default item value is 10 seconds.

**Section name: MUPASZRTU**

**Option name: EVENTS\_DEFINITIONS\_PATH**

**Option value: path**

Purpose - this option specifies a complete access path to the directory including the files with the definitions of events; this option is applicable when devices of Mupasz 07 are used (for devices of Mupasz2001G type it is not possible to define a method for event decoding).

Option value:

*ServerName* - MSSQL server name.

*DatabaseName* - database name on the *ServerName*.

The default value - none.

**Section name: MUPASZRTU**

**Option name: REGISTRAR**

**Option value: ServerName, DatabaseName, Plan1Name, ..., PlanNName**

Purpose - this option switches the driver into disturbance registration mode. The option defines the server name and the database name to which disturbance samples will be stored.

Option value:

*ServerName* - MSSQL server name.

*DatabaseName* - database name on the *ServerName*.

*Plan1Name - PlanNName* - registration plan names for disturbances. The name of at least one plan must be specified

The default value - by default disturbances are not registered.

**Section name: MUPASZRTU**

**Option name: NUMBER\_OF\_REGISTRAR\_REPETITIONS**

**Option value: number**

Purpose - this option specifies the maximum number of repetitions in case of a transmission error during recorder readout.

Option value:

*number* - number of repetitions in case of a transmission error during recorder readout; the parameter value of 1 means that the readout should be repeated until the operation is performed successfully.

The default value - by default, in case of the occurrence of errors, recorder readout is not repeated.

**Section name: MUPASZRTU**

**Option name: NUMBER\_OF\_REPETITIONS**

**Option value: number**

Purpose - this option specifies the number of repetitions in case of a transmission error during readout of current data.

Option value:

*number* - number of repetitions in case of transmission errors during readout of current data (0 means no repetitions).

The default value - by default, 2 repetitions are allowed in case of transmission errors during readout of current data (3 readout attempts in total).

**Section name: MUPASZRTU**

**Option name: NO\_RECEIVE\_DELAY**

**Option value: number\_of\_seconds**

Purpose - this option is used to block requests in a channel for a period of *number\_of\_seconds* if the last request did not return a response; when declared in the MUPASZRTU section, the option is valid for all channels; when declared in a section of the channel name, the option is valid for the specific channel only (it overrides the option value entered in the driver section); throughout the

period when requests are blocked, all commands sent to the driver are ended with a status indicating a lack of response (without any operations carried out via the port).

Option value:

*number*

The default value

- number of seconds.

- by default, this option value is 0 - in such case, the driver operates without blocking transmission after no response has been received.

**Section name: MUPASZRTU**

**Option name: REGISTRAR\_REQUEST\_SCOPE**

**Option value: *max\_number\_of\_request\_registers***

Purpose

- this option specifies the maximum number of registers in a single request during recorder readout.

Option value:

*max\_number\_of\_request\_registers*

The default value

- 125.

**Section name: MUPASZRTU**

**Option name: REGISTRAR\_REQUEST\_TIMEOUT**

**Option value: *number\_of\_milliseconds***

Purpose

- if the value is set to 0, it means that the thread for recorder readout will be of lower priority than the thread sending requests to the application account, which in case of numerous requests to the application account can slow down the recorder readout to last a couple of hours (this mode is used in the existing driver versions). The value of != 0 means that the threads sending requests to the application account and to the recorder account will be of equal priority; the requests sent to the device will interlace and the timeout between consecutive access attempts to the port will equal the parameter value.

Option value:

*number\_of\_milliseconds*

The default value

- the default value is 0.

**Section name: MUPASZRTU**

**Option name: MAX\_REGISTERS**

**Option value: *number***

Purpose

- this option specifies the maximum number of holding registers HR in a single request.

Option value:

*number*

The default value

- the default option value is 120.

## Channel Parameters

The MupaszRtu driver channel parameters declared in the *Miscellaneous* module, in the *Directly entered options* tab):

**Section name: <channel\_name>**

**Option name: NO\_RECEIVE\_DELAY**

**Option value: *number\_of\_seconds***

Purpose

- this option is used to block requests in a channel for a period of *number\_of\_seconds* if the last request did not return a response; when declared in the MUPASZRTU section, the option is

valid for all channels; when declared in a section of the channel name, the option is valid for the specific channel only (it overrides the option value entered in the driver section); throughout the period when requests are blocked, all commands sent to the driver are ended with a status indicating a lack of response (without any operations carried out via the port).

Option value:

*number*

- number of seconds.

The default value

- by default, this option value is 0 - in such case, the driver operates without blocking transmission after no response has been received.

**Section name:** <*channel\_name*>

**Option name:** MAX\_REGISTERS

**Option value:** *number*

Purpose

- this option specifies the maximum number of holding registers HR in a single request.

Option value:

*number*

The default value

- the default option value is 120.

### SAMPLE DRIVER PARAMETERIZATION

*Nazwa sekcji:* MUPASZRTU

*Nazwa opcji:* LOG\_FILE

*Wartość opcji:* d:\tmp\mupasz\drajwer.log

*Nazwa sekcji:* MUPASZRTU

*Nazwa opcji:* LOG\_FILE\_SIZE

*Wartość opcji:* 3

*Nazwa sekcji:* MUPASZRTU

*Nazwa opcji:* LOG\_OF\_TELEGRAMS

*Wartość opcji:* YES

*Nazwa sekcji:* MUPASZRTU

*Nazwa opcji:* NUMBER\_OF\_REPETITIONS

*Wartość opcji:* 0

*Nazwa sekcji:* MUPASZRTU

*Nazwa opcji:* NUMBER\_OF\_REGISTRAR\_REPETITIONS

*Wartość opcji:* 1

*Nazwa sekcji:* MUPASZRTU

*Nazwa opcji:* EVENT\_CHECK\_PERIOD

*Wartość opcji:* 5

*Nazwa sekcji:* MUPASZRTU

*Nazwa opcji:* REGISTRAR

*Wartość opcji:* ServerName, DatabaseName, Plan1t

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Meaning - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Meaning - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value

- NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value

- by default, no time for data validity is set.

## 1.77 MupaszRtu\_TCPIP - Driver of Microprocessor Protecting Devices of MUPASZ 710 Type

### Driver Use

The MupaszRtu\_TCPIP driver is used for data exchange between Asix and the microprocessor protecting devices of MUPASZ 710 type. The communication is performed in MODBUS RTU mode via Ethernet link, using port no. 502.

The driver realizes the following functions:

- readout of current states and measurements,
- readout of events and reporting them to the Asix alarm system,
- readout of interferences from the recorder and their registration in the AsLogger database (software developed by Askom).

#### Notice:

Using the functions of the recorder readout and interference registration in AsLogger database needs the appropriate Asix system license.

### Declaration of Transmission Channel

Declaration of the transmission channel using the MupaszRtu\_TCPIP driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: *MupaszRtu\_TCPIP*

**MupaszRtu\_TCPIP /Channel parameters** tab:

**Device number in the MODBUS network** - device number in the MODBUS network;

Notice: currently the only supported device number in the MODBUS network is 255,

**IP address of the device**,

**Alarms offset**

to the Asix system,

**IP address of the computer**

channel will connect to the computer,

**Port**

**Modbus RTU mode**

- offset added to the calculated alarm number sent

- IP address of the computer through which the

- TCP port number of a device,

- MODBUS RTU format instead of the default Open Modbus TCPIP is used in the channel for data

exchange.

#### EXAMPLE

The declaration of the logical channel named CHAN1, which works according to the Mupasz 710 protocol and has parameters as below:

- IP address 10.10.12.5 and alarm offset: 200;

*Name*: CHAN1

*Driver:* MupaszRtu\_TCPIP

*Channel parameters:*

*Device number in the MODBUS network:* 255

*IP address of the device:* 10.10.12.5

*Alarms offset:* 200

## Addressing the Process Variables

Types of process variable for MODBUS protocol:

HR - 16-bit register	(readout)
HRF - 32-bit register of FLOAT type	(readout)
HRL - 32-bit register of DWORD/LONG type	(readout)

and internal driver variables:

MN - network number of device from which sector is being read or has been read	(readout)
RN - number of currently/recently read sector	(readout)
RQ - sector readout request	(write)
RR - status of currently/recently read sector	(readout)
RS - state of recorder (unavailable/empty/written)	(readout)
RT - release time of interference written in a sector	(readout)
SN - current counter of HR registers read from a sector	(readout)

**The syntax of symbolic address** which is used for variables belonging to the MUPASZ driver channel is as follows:

*Type[Index]*

where:

*Type* - process variable type,  
*Index* - index of a process variable within the type.

### REMARKS:

indexes are not declared for the types: *MN*, *RN*, *RQ*, *RR* and *SN*

- for the types: *RS*, *RT* the range of indexes depends on the number of sectors defined in the device,
- the parameter of sector readout request (*RQ* type) is the sector number; the sectors are numbered w.e.f. 1. If 0 is set as the parameter, that means the interrupt request of sector current readout;
- to present the sector release time, use the conversion function *DATETIME\_MUZ*, that convert the number of *double* type into the following string:

**yyyy-mm-dd hh:nn:ss.zzz**

### EXAMPLE

Example of variable declarations:

ZM_01, face value Un,	HR8235, KLR, 1, 1, NOTHING
ZM_02, face value In,	HR8236, KLR, 1, 1, NOTHING
ZM_03, release time in the sector 1,	RT1, KLR, 1, 1, DATETIME_MUZ
ZM_04, sector readout request	RQ, KLR, 1, 1, NOTHING

ZM_05, number of read sector,	RN,	KLR, 1, 1, NOTHING
ZM_06, status of sector readout,	RR,	KLR, 1, 1, NOTHING
ZM_07, counter of registers read from the sector	SN,	KLR, 1, 1, NOTHING_DW

## Interference Recorder Reading

The driver allows for readout of interferences from the recorder and their registration in the AsLogger database. The registered trends can be reviewed only by AsLogger software.

The recording of interference recorder trends demands declaration of *Registration of disturbances in AsLogger database* option (*MupaszRtu\_TCPIP* tab / *Driver parameters - AsLogger*). It contains:

- MSSQL server name,
- database name,

where the trends will be written.

The driver does not register the trends automatically - registration has to be initiated any time by the Asix system operator by execution of control with the use of RQ type variable. The control value is the number of sector the value of which should be read from a device and written to AsLogger database. If control value equals 0, it means the interrupt request of sector current readout.

While the sector readout the readout state tracking is possible with the use of driver internal variables of symbolic addresses:

a/ *MN* - device number in MODBUS network,  
 b/ *RN* - number of read sector,  
 c/ *RR* - status of sector readout:

0 - readout is not active or finished OK  
 1 - readout is in progress  
 2 - readout of the latest frame finished OK  
 3 - readout of the latest frame finished with error  
 4 - readout of the sector finished with error  
 5 - readout of the sector finished OK  
 6 - readout of the sector stopped by the operator

d/ *SN* - current counter of registers read from the sector.

The recording of register trends is realized according to recording plans. The names of recording plans may be:

a/ defined by the user in text files which are read by the driver at application start-up,  
 b/ generated automatically by the driver.

While the trend recording the driver writes to AsLogger database the following items:

- a/ all the parameters from the sector,
- b/ creates the plan archive,
- c/ determines max and min values for current and voltage in recorded trend - if need they will be corrected during reading the successive trends for the given plan.

### Defining the Names of Recording Plans by the User

The user can define the names of recording plans in text files with the extension '.def'. To do this for each communication channel (channel of Asmen) one should create the separate file and write to them the definitions of recording plan names for interferences recorded in this channel. The files with the plan definitions should be located in the common directory the name of which should be declared as the *Directory with files containing registration plan names* option (*MupaszRtu\_TCPIP* tab / *Driver parameters - AsLogger*).

The syntax of definition for recording plan name defined in '.def' file is as follows:

***Channel, Cause, FormatAi, FormatDi, Table [, Commentary]***

where:

- Channel - Asmen channel name,
- Cause - number identifying the cause of recorder release,
- FormatAi - analog measurement mask in HEX notation (6 digits),
- FormatDi - binary input mask in HEX notation (2 digits),
- Table - recording plan name,
- Commentary - kommentary (option)

Example:

Definition for recording plan for recording interferences with the following parameters:

- voltage value U1 < (release cause number 1),
  - analog mask 0000FF - recording U1, U2, U3, U0 and I1, I2, I3, I0
  - binary values mask 0001 - recording input states 1 - 16
- in the channel KTR:

**KTR, 1, 0000FF, 0001, Field05VoltageDropU1**

### The Default Recording Plan Names

If the user does not define its own recording plan names, the driver will create the name of recording plan automatically.

The syntax of interference recording plan name generated by the driver is as follows:

**ChannelName.Cause.FormatAi.FormatDi**

where:

- ChannelName - name of Asmen channel in which the interferences were read,
- Cause - identifier of recording release cause,
- FormatAi - analog measurement mask in HEX notation (6 digits),
- FormatDi - binary input mask in HEX notation (2 digits).

**Example:**

For interference caused by the recorder release from the user's logic in the device operated in Asmen channel named CHANNEL with the registration set as follows:

- analogs: U1, U2, U3, U0, I1, I2, I3, IO  
and

- binary measurements: inputs 1-16, inputs 17-32, outputs 1-16

the following recording plan name will be defined:

**CHANNEL.29.000FF.0023**

**Identifiers of Recorder Release Reasons**

Below the set of recorder release reasons:

01	U1LE	voltage U1 as a parameter
02	U2LE	voltage U2 as a parameter
03	U3LE	voltage U3 as a parameter
04	U0LE	voltage Uo as a parameter
05	I1LE	current I1 as a parameter
06	I2LE	current I2 as a parameter
07	I3LE	current I3 as a parameter
08	IOLE	current Io as a parameter
09	I1LESEC	current I1sec as a parameter
10	I2LESEC	current I2sec as a parameter
11	I3LESEC	current I3sec as a parameter
12	IOLESEC	current Iosec as a parameter
13	U1GT	voltage U1 as a parameter
14	U2GT	voltage U1 as a parameter
15	U3GT	voltage U1 as a parameter
16	U0GT	voltage Uo as a parameter
17	I1GT	current I1 as a parameter
18	I2GT	current I2 as a parameter
19	I3GT	current I3 as a parameter
20	IOGT	current Io as a parameter
21	I1GTSEC	current I1sec as a parameter
22	I2GTSEC	current I2sec as a parameter
23	I3GTSEC	current I3sec as a parameter
24	IOGTSEC	current Iosec as a parameter
25	DIN	release input number as a parameter (1..80)
26	DOUT	release output number as a parameter (1..80)
27	SIG	release register number as a parameter (1..80)
28	OIN	release input number as a parameter (1..8)
29	USER	register release by the user:

parameter 0 - local, 1 - remote

**Analog Measurement Mask**

There are the set of bits of analog measurement mask below, which may be recorded in interference recorder (lists **LIST\_8** and **LIST\_9** in **list.xml**).

U1	0x0001
U2	0x0002
U3	0x0004
U0	0x0008
I1	0x0010
I2	0x0020
I3	0x0040

## Communication Drivers

I0	0x0080
I1sec	0x0100
I2sec	0x0200
I3sec	0x0400
I0sec	0x0800
Ia_1	0x1000
Ia_2	0x2000
Ia_3	0x4000
Ia_4	0x8000
Ia_5	0x0010000
Temp_1	0x0020000
Temp_2	0x0040000
Temp_3	0x0080000
Temp_4	0x0100000
Temp_5	0x0200000
Temp_6	0x0400000
I0sec	0x0800000

## Binary Measurement Mask

There are the set of bits of binary measurement mask below, which may be recorded in interference recorder (list **LIST\_10** in **list.xml**).

IN_01_16	0x0001
IN_17_32	0x0002
IN_33_48	0x0004
IN_49_64	0x0008
IN_65_80	0x0010
OUT_01_16	0x0020
OUT_17_32	0x0040
OUT_33_48	0x0080
OUT_49_64	0x0100
OUT_65_80	0x0200
SIG_01_16	0x0400
SIG_17_32	0x0800
SIG_33_48	0x1000
SIG_49_64	0x2000
SIG_65_80	0x4000
OIN_01_08	0x8000

## Event Signalling

Signalling events is checked periodically, the frequency of check is determined by the global option **Event check period** (see: *Driver parameters*). Event read from the devices are converted to the corresponding numbers in the Asix system alarms.

The conversion is performed based on the following algorithm:

- definitions and the way they are converted are read from the files: **event.xml** and **format.xml**, provided by the device manufacturer,
- driver converts the parameters of the event as it is described in the **event.xml** file w.e.f. no. 1 until depletion of the buffer used to pass event parameters to the Asix alarm system

or depletion of the number of event parameters (the driver may report any combination of 16-bit to 32-bit parameters, provided that their total size do not exceed 8 bytes),

- driver converts the value of each parameter according to the rule defined by the format of the parameter described in the format.xml file.

- it is possible to use global files **event.xml** and **format.xml** by entering the names of this files in the global options **Event definitions files** (see: *Driver parameters, Channel parameters 2*) and **Format definitions file** (see: *Driver parameters, Channel parameters 2*).

- it is possible to define the specific files **event.xml** and **format.xml** by entering the names of this files in the given transmission channel options: **Event definitions file** i **Format definitions file**).

- driver searches the files **event.xml** and **format.xml** in the directory the name of which is given in the global option **Directory with files containing plan names** (**MupaszRtu\_TCPIP** tab / **Driver parameters - AsLogger**).

## Driver Use

The driver configuration is defined in the *Current Data* module, in the channel operating according to MupaszRtu\_TCPIP driver. Options are placed on the tabs:

**MupaszRtu\_TCPIP / Driver parameters, MupaszRtu\_TCPIP / Driver parameters - AsLogger, MupaszRtu\_TCPIP / Driver parameters - diagnostics.** These options refer to all the devices.

### **Log file**

Meaning: The item allows to define a file where all diagnostic messages of the driver are written.

Option value:  
*log\_file\_name*

Default value: By default, the log file is not created.

### **Log file size**

Meaning: The item allows to specify the size of log file size defined by the options: *Log file* and *Disturbances log file*.

Option value:  
*number* - log file size in MB.

Default value: 10 MB.

### **Disturbances log file**

Meaning: The item allows to define a file where all diagnostic messages on read disturbances are written.

Option value:  
*log\_file\_name*

Default value: The log file is not created.

### **Log of telegrams**

Meaning: The item allows to write to the log file the contents of telegrams sent between the driver and network clients. Writing the contents of telegrams to the log file should be used only while the **Asix** start-up.

## Communication Drivers

Option value: YES/NO  
Default value: NO.

### **Password**

Meaning: The option defines the password of the user registered in Mupasz 710 device. The password is necessary to read interferences of the recorder. The password entered by this option is a global option and concerned all the devices. The password value can be overloaded by the option declared in the communication channel options.

Option value: *password* - four-character text.  
Default value: "1111".

### **Number of repetitions**

Meaning: The option defines the number of connection repetitions in case of transmission error. When the number of repetitions is exhausted the connection with the device is closed and re-established.

Option value: *number* - the number of connection repetitions.  
Default value: 3.

### **Event check period**

Meaning: The option defines the time after which the event register readout should be performed.

Option value: *number* - number of seconds.  
Default value: 10.

### **Definition files path**

Meaning: The option defines the full path to the directory in which the driver will be searching for files with event definition (**event.xml**) and files with format definitions (**format.xml**).

Option value: *path* - full path to the directory.  
Default value: -.

### **Event definitions file**

Meaning: The option defines the name of XML file with event definition. The option is global one, but may be overloaded by the same option defined in the communication channel parameters (**MupaszRtu\_TCPIP** tab / **Channel parameters**).

Option value: *file\_name* - full path to the directory.

### **Format definitions file**

Meaning: The option defines the name of XML file with format definition. The option is the global one, but it may be overloaded by the same option declared in the communication channel parameters (**MupaszRtu\_TCPIP** tab / **Channel parameters**).

Option value: *file\_name*

### **Registration of disturbances in AsLogger database**

Meaning: The option changes over the driver to the mode of writing recorder trend to a database. It concerns samples from disturbance and criterion recorders.

Option value:

*SERVERNAME* - MS SQL SERVER NAME  
*DatabaseName* - database name of the *ServerName* server  
 Default value: Disturbances are not recorded.

**Number of repetitions of disturbances registrar**

Meaning: The option defines the number of repetitions in case of transmission error.

Option value:  
*number* - number of repetitions.

Default value: 0 (no repetition).

**Directory with files containing registratuon plan names**

Meaning: The option defines full path to the directory of files with recording plan definitions.

Option value:  
*path* - full path to the directory.

## Channel Parameters

The channel parameters are defined in the *Current Data* module, in the channel operating according to MupaszRtu\_TCPIP driver. Options are placed on the tabs:

***MupaszRtu\_TCPIP / Channel parameters 2, MupaszRtu\_TCPIP / Driver parameters - diagnostics.*** These options refer only to the channel.

**Password**

Meaning: The option defines the password of the user registered in Mupasz 710 device. The password is necessary to read interferences of the recorder.

Option value:  
*password* - four-character text.

Default value: By default, the password is defined by the driver option, if not - the following value is taken: "1111".

**Number of repetitions**

Meaning: The option defines the number of query in the case of response error in that channel. When the number of repetitions is exhausted the connection with the device is closed and re-established.

Option value:  
*number* - the number of connection repetitions.

Default value: By default, the password is defined by the driver option.

**Response timeout**

Meaning: The option defines the maximal time of waiting for response from the device.

Option value:  
*number* - timeout in milliseconds.

Default value: 500.

**Response delay**

Meaning: The option defines the time of delay between sending the query to the device and starting the operation of getting response. The use

## Communication Drivers

of this option has a sense in case of considerable defragmentation of the response.

Option value:  
*number* - delay in milliseconds  
Default value: 20.

### **Event definitions file**

Meaning: The option defines the name of XML file with event definition. The option may overload the same option defined in the driver parameters.

Option value:  
*file\_name* - full path to the directory.

### **Format definitions file**

Meaning: The option defines the name of XML file with format definition. The option may overload the same option declared in the driver parameters.

Option value:  
*file\_name*

### **Log file**

Meaning: The item allows to define a file where all diagnostic messages of the driver in the given communication channel are written.

Option value:  
*log\_file\_name*

Default value: By default, the log file is defined by the driver option.

### **Log file size**

Meaning: The item allows to specify the size of log file size defined by the option *Log file* of the given channel.

Option value:  
*number* - log file size in MB.  
Default value: 10 MB.

### **Log of telegrams**

Meaning: The item allows to write to the log file the contents of telegrams sent between the driver and network clients in the given channel. Writing the contents of telegrams to the log file should be used only while the **Asix** start-up.

Option value: YES/NO  
Default value: NO.

## Examples of Driver Parameters

```
Driver: MupaszRtu_TCPIP
Log file=c:\tmp\mupasz.log
Disturbance log file=c:\tmp\zaklocenia.log
Log file size=20
Log of telegrams=TAK
Event check period=10
Registration of disturbances in AsLogger database=SERWER_RO6, BAZA_RO6
Disturbance names files path=c:\tmp\DefZaklocen
Definition files path=c:\tmp\DefFiles
Password=1234
```

## Examples of Channel Parameters

*ChannelName*= KANALX  
*Log file*=c:\tmp\kanalx.log  
*Event definitions file*=kanalx\Event.xml  
*Format definitions file*=kanalx\Format.xml  
*Password*=4321

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

## Communication Drivers

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.78 Mus04 - Driver for Data Exchange with MUS-04 Control Devices from ELEKTROMETAL S.A.

### Driver Use

The Mus04 protocol driver allows data between the Asix system and the microprocessor-based control devices MUS-04 manufactured by ELEKTROMETAL S.A. from Cieszyn to be exchanged. The transmission is executed with the use of serial links by means of standard computer serial ports in RS-485 standard.

Parameterization of the Mus04 driver is performed with the use of the Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel operating according to Mus04 protocol requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: Mus04

**CtMus04** tab:

Channel parameter:  
*Port=number; No=number*  
 [[*;* *Timeout=number*]; *CharTimeout=number*]; *TransmissionDelay=number*]

where:

<i>Port</i>	- serial port number,
<i>No</i>	- number of Mus04 supported through this channel,
<i>Timeout</i>	- max. waiting time for the first response character (in milliseconds); by default - 1000 milliseconds;
<i>CharTimeout</i>	- max. waiting time between the response characters (in milliseconds); by default - 100 milliseconds;
<i>TransmissionDelay</i>	- means a beak in milliseconds between the end of receiving the data from the device and sending a next inquiry to the device.

Transmission parameters are constant and have the following values:

- transmission speed 2400 Bd,
- 8 character bits,
- no parity bit,
- 1 stop bit.

#### EXAMPLE

An example of the declaration of the transmission channel for communication with the Mus04 devices no. 1 and 2 with the use of the COM2 serial port and with Mus04 no. 3 with the use of the COM1 serial port.

**Name**: K1

**Driver**: CtMus04

*Channel parameters*: Port=2;No=1

**Name: K2**

**Driver: CtMus04**

*Channel parameters:* Port=2;No=2

**Name: K3**

**Driver: CtMus04**

*Channel parameters:* Port=1;No=3

## Addressing Variables

Symbolic address of the process variable has the following syntax:

*< type> . < index >*

where:

*type* - variable type,  
*index* - index within type - in relation to the variable reading operation.

Variable types markings (in the brackets, the raw variable value type is given) are discussed below:

### Read-only variable types:

**R1** - information on MUS status, index range 1 - 12, index meanings - as in box 161,

**R2** - is not used,

**R3** - function matrix, index range 1 - 9, index meanings - as in box 163,

**R4** - delay times, index range 1 - 8, index meanings - as in box 164,

**R5** - retention times, index range 1 - 8, index meanings - as in box 164,

**R6** - description of input determined by the index, index range 1 - 8,

**R7** - description of output determined by the index, index range 1 - 8,

**R8** - access password, only index 1,

**R9** - input settings, index range 1 - 8, index meanings - as in box 169,

**R10** - output settings, index range 1 -8, index meanings - as in box 170,

**R11** - MW settings, only index 1,

**R12** - settings change information, only index 1,

**R13** - additional settings information, only index 1.

### Write-only (control) variables types:

**W1** - RESET,

**W2** - change of MUS number, it is not implemented,

**W3** - sending new function matrix - buffer must have frame format of 3,

**W3** - sending new delay times - buffer must have frame format of 4,

**W5** - sending new retention times - buffer must have frame format of 5,

**W6** - sending new retention times - buffer must have frame format of 6,

**W7** - sending output description - buffer must have frame format of 7,

**W8** - sending access password - buffer must have frame format of 8,

**W9** - sending new input settings - buffer must have frame format of 9,

**W10** - sending new output settings - buffer must have frame format of 10,

**W11** - sending new MW settings - buffer has a size of 1 byte,

**W12** - deleting bits related to settings change - buffer has a size of 1 byte.

Writing variables of **W6**, **W7** and **W8** types can be executed from **CAPTION** objects.

Writing variables of **W11** and **W12** types can be executed from **NUMBER** objects.

Writing variables of the remaining types can be executed with the use of scripts.

## EXAMPLES

Examples of variable declaration - channel K1 supports the Mus-04 device no.1, channel K2 supports the Mus-04 device no.2:

```
JJ_10, IN1 input status of Mus-04 no. 1,      R1.1, K1, 1, 1, NIC
JJ_11, IN8 input status of Mus-04 no. 2,      R1.8, K2, 1, 1, NIC
JJ_12, battery voltage for Mus-04 no. 1,      R1.10, K1, 1, 1, NIC_FP
JJ_13, supply voltage of Mus-04 no. 2,        R1.11, K2, 1, 1, NIC_FP
JJ_14, description of input no. 1 of Mus-04 no. 2, R6.1, K2, 1, 1, NIC_TEXT
JJ_15, description of output no.5 of Mus-04 no.1, R7.5, K1, 1, 1, NIC_TEXT
JJ_16, delay time no. 2 of Mus-04 no. 1,      R4.2, K1, 1, 1, NIC
JJ_16, settings of inputs no. 3 of Mus-04 no. 1, R9.3, K1, 1, 1, NIC
JJ_17, settings of outputs no. 4 of Mus-04 no. 2, R10.4, K2, 1, 1, NIC
```

Variables used only for execution of controls:

```
JJ_20, sending RESET order to Mus-04 no. 2,    W1, K2, 1, 1, NIC
JJ_21, new matrix settings for Mus-04 no. 2,    W3, K2, 9, 1, NIC_BYTE
JJ_22, new delay times settings for Mus-04 no. 1, W4, K1, 8, 1, NIC_BYTE
JJ_23, new input settings for Mus-04 no. 1,     W9, K1, 8, 1, NIC_BYTE
JJ_24, new output settings for Mus-04 no. 2,    W10, K2, 8, 1, NIC_BYTE
```

## Driver Parameterization

The Mus04 driver parameters are declared in the *Miscellaneous* module, on *Directly entered options* tab.

**Section name: CtMus04**

**Option name: LOG\_FILE**

**Option value: log\_filename**

Meaning - a text log file, where the driver operation status messages are entered.

Default value - by default, the log file is not created.

Defining - manual.

**Section name: CtMus04**

**Option name: LOG\_FILE\_SIZE**

**Option value: number**

Meaning - option is used to determine the size of the log file defined with use of the LOG\_FILE option.

Default value - default log file size is 10 MB.

Parameters:

*number* - log file size in MB.

Defining - manual.

**Section name: CtMus04**

**Option name: LOG\_OF\_TELEGRAMS**

**Option value: YES/NO**

Meaning - allows the contents of telegrams sent between the driver and controllers to the log file (declared with use of LOG\_FILE option) to be saved. The discussed option can be used only during the Asix system start-up.

Default value - by default, the option value is set to NO.

Defining - manual.

**EXAMPLE OF DRIVER SECTION:**

*Section name:* CTMUS04  
*Option name:* LOG\_FILE  
*Option value:* d:\tmp\CtMus04\mus.log

*Section name:* CTMUS04  
*Option name:* LOG\_FILE\_SIZE  
*Option value:* 20

*Section name:* CTMUS04  
*Option name:* LOG\_OF\_TELEGRAMS  
*Option value:* YES

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.79 MUZ - Driver of Protocol for MUZ Devices

### Driver Use

The MUZ driver is used for data exchange between Microprocessor Protecting Devices (MUZ) of the MUZ-RO type made by JMTronik Warsaw and an Asix system computer.

In the current driver version logical channels are connected with serial ports that service MUZ-RO devices, and not with each of MUZ-RO devices separately.

Parameterization of MUZ driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the MUZ driver requires a channel with the following parameters to be added to the Current data module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* MUZ

**MUZ** tab:

*Channel parameters:*  
*port*

where:  
*port*

- address of the serial port (HEX), which the connection with MUZ-RO devices will be realized by.

The separate logical channel declaration is demanded for each serial port used by the MUZ driver.

### Declaration of Device

Declaration of all MUZ-RO devices serviced by the MUZ driver is realized by means of the MUZ position set in the separate **MUZ** section. Each device should have the separate MUZ position.

The MUZ position includes the parameters used for logical channel declaration in the previous driver version. The position syntax is given below:

*channel\_name, id, type, alarmTxtOff, alarmValOff, var\_status, var\_control*

where:

<i>channel_name</i>	- name of the logical channel declared in the [ASMEN] section, which the given MUZ-RO device will be serviced by;
<i>id</i>	- number of MUZ-RO in the network;
<i>type</i>	- identifier of the MUZ type (for MUZ-RO it is 7);
<i>alarmTxtOff</i>	- number added to the number of a text alarm transferred from MUZ-RO;
<i>alarmValOff</i>	- number added to the alarm number with the value transferred from MUZ-RO;
<i>var_status</i>	- variable that shows the status of transmission with MUZ-RO;
<i>var_control</i>	- variable that controls the realization of transmission with MUZ-RO.

Parameters are declared in the Miscellaneous module, the Directly entered options tab.

## Syntax of Symbolic Variable Address

The previous symbolic address of the variable has been extended by the prefix containing the MUZ-RO device number. The present address is as follows:

*Address.IdMUZ*

where:

<i>Address</i>	- the previous address of the symbolic variable;
<i>IdMUZ</i>	- number of the MUZ-RO device, which the variable with the Address address is received from.

## Symbolic Addresses and Kinds of Variables Used in Data Exchange with MUZ Devices

In the operation of data exchange with MUZ devices, the variables are divided into groups differing with data kind and addressing method. To each group a different symbolic address is assigned. The variables are divided into the following groups:

- values of analog measures;
- binary inputs handled together with measurements (state of single bits is transferred);
- binary inputs handled together with measurements (state of 16 successive SPi variables is transferred);
- rated values and settings of protections;
- binary inputs related to settings of protections;
- variable storing states of events with value;
- variable storing states of text events;
- control variables.

The way of configuring the individual groups of variables is described below.

Table. Values of Analog Measures.

Symb. Addr.	Variable in MUZRO	Type of Conversion	Allowed Operation
P1	Current I0	Byte3->Float	Reading
P2	Current I1	Byte3->Float	Reading
P3	Current I2	Byte3->Float	Reading
P4	Current I3	Byte3->Float	Reading
P5	Voltage U0	Byte3->Float	Reading
P6	Voltage U1	Byte3->Float	Reading
P7	Voltage U2	Byte3->Float	Reading
P8	Voltage U3	Byte3->Float	Reading
P9	cos(f1)	Byte4->Float	Reading
P10	cos(f2)	Byte4->Float	Reading
P11	cos(f3)	Byte4->Float	Reading
P12	active power	Byte3->Float	Reading
P13	reactive power	Byte3->Float	Reading
P14	active energy	BCD6->Float	Reading
P15	reactive energy	BCD6->Float	Reading

Table. Binary Inputs Handled Together with Measurements (State of Single Bits is Transferred)

8	Variable in MUZRO	Type of Conversion	Allowed Operation
SP1	switch status, closed	Bit (0/1)->Word	Reading
SP2	switch status, closed	Bit (0/1)->Word	Reading
SP3	freely used by client	Bit (0/1)->Word	Reading
SP4	freely used by client	Bit (0/1)->Word	Reading
SP5	local opening of switch	Bit (0/1)->Word	Reading
SP6	freely used by client	Bit (0/1)->Word	Reading
SP7	freely used by client	Bit (0/1)->Word	Reading
SP8	local closing of switch	Bit (0/1)->Word	Reading
SP9	signal for technological protection	Bit (0/1)->Word	Reading
SP10	signal for technological protection 10	Bit (0/1)->Word	Reading
SP11	signal for technological protection 11	Bit (0/1)->Word	Reading
SP12	signal for technological protection 12	Bit (0/1)->Word	Reading
SP13	signal for technological protection 13	Bit (0/1)->Word	Reading
SP14	signal for technological protection 14	Bit (0/1)->Word	Reading
SP15	signal for technological protection 15	Bit (0/1)->Word	Reading
SP16	signal for technological protection 16	Bit (0/1)->Word	Reading
SP17	signal for technological protection 17	Bit (0/1)->Word	Reading
SP18	lock of function of undervoltage protection	Bit (0/1)->Word	Reading
SP19	not used - 0	Bit (0/1)->Word	Reading
SP20	not used - 0	Bit (0/1)->Word	Reading
SP21	not used - 0	Bit (0/1)->Word	Reading
SP22	not used - 0	Bit (0/1)->Word	Reading
SP23	1 - MUZ signals operation of one of protections	Bit (0/1)->Word	Reading
SP24	not used - 0	Bit (0/1)->Word	Reading

Table. Binary Inputs Handled Together with Measurements (State of 16 Successive SPI Variables is Transferred)

Symb. Addr.	Variable in MUZRO	Type of Conversion	Allowed Operation
SPW	actual status of variables SP1 (Bit 0) - SP16	WORD ->WORD	Reading
SPW	actual status of variables SP17 (Bit0) - SP24	WORD->WORD	Reading
	(Bits B8 - B15 are filled with zeroes)		

Table. Specific Variable in MUZRO.

Symb. Addr.	Variable in MUZRO	Type of Conversion	Allowed Operation
RP1	Number of not read events (operation of protections)		Reading
RP2	event code (1-22) if RP1 != 0		Reading

Table. Rated Values and Settings of Protections.

Symb. Addr.	Variable in MUZRO	Type of Conversion	Allowed Operation
Z1	Voltage Uzn	Byte3<->Float	Read/Write
Z2	Current Izn	Byte3<->Float	Read/Write
Z3	Utilization factor of network Iw	Byte3<->Float	Read/Write
Z4	Coefficient kcdc - current cutter	Byte3<->Float	Read/Write
Z5	Coefficient ki>> - short-circuit protection	Byte3<->Float	Read/Write
Z6	Coefficient ti>> - short-circuit protection	Byte3<->Float	Read/Write
Z7	Coefficient ki> - independent overload protection	Byte3<->Float	Read/Write
Z8	Coefficient ti> - independent overload protection	Byte3<->Float	Read/Write
Z9	Coefficient t1.2 - dependent overload protection	Byte3<->Float	Read/Write
Z10	Coefficient t1.5 - dependent overload protection	Byte3<->Float	Read/Write
Z11	Coefficient t2.0 - dependent overload protection	Byte3<->Float	Read/Write
Z12	Coefficient t3.0 - dependent overload protection	Byte3<->Float	Read/Write
Z13	Coefficient t6.0 - dependent overload protection	Byte3<->Float	Read/Write
Z14	Coefficient ta - dependent overload protection	Byte3<->Float	Read/Write
Z15	Coefficient Ii0> - ground-fault protection	Byte3<->Float	Read/Write
Z16	Coefficient ti0> - ground-fault protection	Byte3<->Float	Read/Write
Z17	Coefficient ku< - undervoltage protection	Byte3<->Float	Read/Write
Z18	Coefficient tu< - undervoltage protection	Byte3<->Float	Read/Write
Z19	Coefficient tt9 - technological protection 9	Byte3<->Float	Read/Write
Z20	Coefficient tt10 - technological protection 10	Byte3<->Float	Read/Write
Z21	Coefficient tt11 - technological protection 11	Byte3<->Float	Read/Write
Z22	Coefficient tt12 - technological protection 12	Byte3<->Float	Read/Write
Z23	Coefficient tt13 - technological protection 13	Byte3<->Float	Read/Write
Z24	Coefficient tt14 - technological protection 14	Byte3<->Float	Read/Write

Communication Drivers

Z25	Coefficient tt15 - technological protection 15	Byte3<->Float	Read/Write
Z26	Coefficient tt16 - technological protection 16	Byte3<->Float	Read/Write
Z27	Coefficient tt17 - technological protection 17	Byte3<->Float	Read/Write
Z28	Coefficient tiE - impulse time on output E	Byte3<->Float	Read/Write
Z29	Coefficient tiF - impulse time on output F	Byte3<->Float	Read/Write

Table. Binary Inputs Related to Settings of Protections.

Symb. Addr.	Variable in MUZRO	Type of Conversion	Allowed Operation
SZ1	on/off - current cutter	Bit (0/1)<->Word	Read/Write
SZ2	on/off - short-circuit protection	Bit (0/1)<->Word	Read/Write
SZ3	op/sign. - overload protection, independent	Bit (0/1)<->Word	Read/Write
SZ4	on/off - overload protection, independent	Bit (0/1)<->Word	Read/Write
SZ5	op/sign. - overload protection, dependent	Bit (0/1)<->Word	Read/Write
SZ6	on/off - overload protection, dependent	Bit (0/1)<->Word	Read/Write
SZ7	op/sign. - ground-fault protection	Bit (0/1)<->Word	Read/Write
SZ8	on/off - ground-fault protection	Bit (0/1)<->Word	Read/Write
SZ9	op/sign. - undervoltage protection	Bit (0/1)<->Word	Read/Write
SZ10	on/off - undervoltage protection	Bit (0/1)<->Word	Read/Write
SZ11	op/sign. - technological protection 9	Bit (0/1)<->Word	Read/Write
SZ12	dep./indep. - technological protection 9	Bit (0/1)<->Word	Read/Write
SZ13	op/sign. - technological protection 10	Bit (0/1)<->Word	Read/Write
SZ14	dep./indep. - technological protection 10	Bit (0/1)<->Word	Read/Write
SZ15	op/sign. - technological protection 11	Bit (0/1)<->Word	Read/Write
SZ16	dep./indep. - technological protection 11	Bit (0/1)<->Word	Read/Write
SZ17	op/sign. - technological protection 12	Bit (0/1)<->Word	Read/Write
SZ18	dep./indep. - technological protection 12	Bit (0/1)<->Word	Read/Write
SZ19	op/sign. - technological protection 13	Bit (0/1)<->Word	Read/Write
SZ20	dep./indep. - technological protection 13	Bit (0/1)<->Word	Read/Write
SZ21	op/sign. - technological protection 14	Bit (0/1)<->Word	Read/Write
SZ22	dep./indep. - technological protection 14	Bit (0/1)<->Word	Read/Write
SZ23	op/sign. - technological protection 15	Bit (0/1)<->Word	Read/Write
SZ24	dep./indep. - technological protection 15	Bit (0/1)<->Word	Read/Write
SZ25	op/sign. - technological protection 16	Bit (0/1)<->Word	Read/Write
SZ26	dep./indep. - technological protection 16	Bit (0/1)<->Word	Read/Write
SZ27	op/sign. - technological protection 17	Bit (0/1)<->Word	Read/Write
SZ28	dep./indep. - technological protection 17	Bit (0/1)<->Word	Read/Write

Table. Variables Storing States of Events with Value.

Symb. Addr.	Variable in MUZRO	Type of Conversion	Allowed Operation
ZW8	Current cutter - phase 0	Bit (0/1)->Word	Read/Write
ZW9	Current cutter - phase 3	Bit (0/1)->Word	Read/Write
ZW10	Current cutter - phase 2	Bit (0/1)->Word	Read/Write
ZW11	Current cutter - phases 2 and 3	Bit (0/1)->Word	Read/Write
ZW12	Current cutter - phase 1	Bit (0/1)->Word	Read/Write
ZW13	Current cutter - phases 1 and 3	Bit (0/1)->Word	Read/Write
ZW14	Current cutter -phases 1 and 2	Bit (0/1)->Word	Read/Write
ZW15	Current cutter - phases 1, 2 and 3	Bit (0/1)->Word	Read/Write
ZW16	Short-circuit protection - phase 0	Bit (0/1)->Word	Read/Write
ZW17	Short-circuit protection - phase 3	Bit (0/1)->Word	Read/Write
ZW18	Short-circuit protection - phase 2	Bit (0/1)->Word	Read/Write
ZW19	Short-circuit protection - phases 2 and 3	Bit (0/1)->Word	Read/Write
ZW20	Short-circuit protection - phase 1	Bit (0/1)->Word	Read/Write
ZW21	Short-circuit protection - phases 1 and 3	Bit (0/1)->Word	Read/Write
ZW22	Short-circuit protection - phases 1 and 2	Bit (0/1)->Word	Read/Write
ZW23	Short-circuit protection - phases 1, 2 and 3	Bit (0/1)->Word	Read/Write
ZW24	Independent overload protection - phase 0	Bit (0/1)->Word	Read/Write
ZW25	Independent overload protection - phase 3	Bit (0/1)->Word	Read/Write
ZW26	Independent overload protection - phase 2	Bit (0/1)->Word	Read/Write
ZW27	Independent overload protection - phases 2 and 3	Bit (0/1)->Word	Read/Write
ZW28	Independent overload protection - phase 1	Bit (0/1)->Word	Read/Write
ZW29	Independent overload protection - phases 1 and 3	Bit (0/1)->Word	Read/Write
ZW30	Independent overload protection - phases 1 and 2	Bit (0/1)->Word	Read/Write
ZW31	Independent overload protection - phases 1, 2 and 3	Bit (0/1)->Word	Read/Write
ZW32	Dependent overload protection - phase 0	Bit (0/1)->Word	Read/Write
ZW33	Dependent overload protection - phase 3	Bit (0/1)->Word	Read/Write
ZW34	Dependent overload protection - phase 2	Bit (0/1)->Word	Read/Write
ZW35	Dependent overload protection - phases 2 and 3	Bit (0/1)->Word	Read/Write
ZW36	Dependent overload protection - phase 1	Bit (0/1)->Word	Read/Write
ZW37	Dependent overload protection - phases 1 and 3	Bit (0/1)->Word	Read/Write
ZW38	Dependent overload protection - phases 1 and 2	Bit (0/1)->Word	Read/Write
ZW39	Dependent overload protection - phases 1, 2 and 3	Bit (0/1)->Word	Read/Write
ZW40	Ground-fault protection	Bit (0/1)->Word	Read/Write

Communication Drivers

ZW64	Undervoltage protection - phase 0	Bit (0/1)->Word	Read/Write
ZW65	Undervoltage protection - phase 3	Bit (0/1)->Word	Read/Write
ZW66	Undervoltage protection - phase 2	Bit (0/1)->Word	Read/Write
ZW67	Undervoltage protection - phases 2 and 3	Bit (0/1)->Word	Read/Write
ZW68	Undervoltage protection - phase 1	Bit (0/1)->Word	Read/Write
ZW69	Undervoltage protection - phases 1 and 3	Bit (0/1)->Word	Read/Write
ZW70	Undervoltage protection - phases 1 and 2	Bit (0/1)->Word	Read/Write
ZW71	Undervoltage protection - phases 1, 2 and 3	Bit (0/1)->Word	Read/Write

Table. Variable Storing States of Text Events.

<b>Symb. Addr.</b>	<b>Variable in MUZRO</b>	<b>Type of Conversion</b>	<b>Allowed Operation</b>
ZT1	Short-circuit PDZ (accelerated protections)	Bit (0/1)->Word	Read/Write
ZT4	Failure of shutdown	Bit (0/1)->Word	Read/Write
ZT5	Failure of switching on	Bit (0/1)->Word	Read/Write
ZT7	Opening of remote control	Bit (0/1)->Word	Read/Write
ZT8	Closure of remote control	Bit (0/1)->Word	Read/Write
ZT38	Technological protection 9	Bit (0/1)->Word	Read/Write
ZT39	Technological protection 10	Bit (0/1)->Word	Read/Write
ZT40	Technological protection 11	Bit (0/1)->Word	Read/Write
ZT41	Technological protection 12	Bit (0/1)->Word	Read/Write
ZT 48	Technological protection 13	Bit (0/1)->Word	Read/Write
ZT49	Technological protection 14	Bit (0/1)->Word	Read/Write
ZT50	Technological protection 15	Bit (0/1)->Word	Read/Write
ZT51	Technological protection 16	Bit (0/1)->Word	Read/Write
ZT52	Technological protection 17	Bit (0/1)->Word	Read/Write
ZT116	Bad clock setting	Bit (0/1)->Word	Read/Write
ZT117	Checksum error (MUZ failure)	Bit (0/1)->Word	Read/Write

Table. Control Variables.

Symb. Addr.	Variable in MUZRO	Type of Conversion	Allowed Operation
KS1	Quitting signals of protection activation (writing any number)	Word->Word	Writing
	and zeroing statuses of all variables ZTi and ZWi		
	in internal driver buffer		
SW1	Opening (0) and closure of breaker (1)	Word->Word	Writing
KZ1	Zeroing statuses of all variables ZTi and ZWi	Word->Word	Writing
	in internal driver buffer (writing any number)		

#### Meanings of Abbreviations

- On - protection switched on (1),
- Off - protection switched off (0),
- Op - protection causes opening of a breaker (1),
- Sygn - protection causes activation of a signalization (0),
- Dep. - protection is active when a breaker is closed (0),
- Indep. - protection is active independently.

## Declaring the Permission for Control

Group handling the MUZ-RO devices has an effect on passing control permissions. It results from the fact that the ASMEN positions declaring control permissions refer to the individual logical channels. In the previous MUZ driver version (v 1.3.1.) the logical channel was assigned to the singular MUZ-RO device and it allowed passing permissions for each of MUZ-ROs individually. In the current group version of the driver the logical channel may group a few MUZ-RO devices and permissions for control refer to all the MUZ-RO devices handled by the given logical channel.

## Configuring the Driver

MUZ driver parameters are declared in the Miscellaneous module, the Directly entered options tab.

- Section name:** MUZ
- Option name:** RECV\_TIMEOUT
- Option value:** *number*

Meaning - the item allows to determine a maximal waiting time between sending a query and receiving an answer (so called receiving timeout). The parameter value is passed for all handled MUZ-RO devices globally.

Default value - 1000.

Parameters:  
*number* - timeout value in milliseconds.

**Section name: MUZ**

**Option name: LOG\_FILE**

**Option value: file\_name, file\_size**

Meaning - allows to define a file to which all diagnostic driver messages and information about the content of telegrams sent/received by the driver will be written. If the item LOG\_FILE does not defines the full path then the log file will be created in the current directory.

Default value - log file is not created.

Parameters:

*file\_name* - log file name;

*file\_size* - log file size in MB;

**Section name: MUZ**

**Option name: DATA\_UPDATE**

**Option value: number**

Meaning - the item allows to specify an acceptable time difference between the time stamp of the variable stored in the driver cache and the current system time. After exceeding the acceptable difference the driver performs reading the variable from the MUZ-RO device.

Default value - by default the item assumes the value of 0.

Parameters:

*number* - time value in seconds;

**Section name: MUZ**

**Option name: TIME\_UPDATE**

**Option value: number**

Meaning - the item allows to specify the global period of updating the time between the Asix system and the MUZ-RO device.

Default value - by default the item assumes the value of 1.

Parameters:

*number* - time value in minutes.

**Section name: MUZ**

**Option name: LOG\_OF\_TELEGRAMS**

**Option value: YES/NO**

Meaning - the item allows writing to the log file (declared with use of LOG\_FILE item) the contents of telegrams transmitted during the data exchange between the Asix system and MUZ-ROs; writing the telegrams content to the log file should be used only during starting the Asix system.

Default value - by default the file is not created.

**Section name: MUZ**

**Option name: NUMBER\_OF\_REPETITIONS**

**Option value: number**

Meaning - the item allows to define maximal number of trials to do the command in case of transmission errors.

Default value - by default the repetitions are not realized.

Parameters:

*number* - number of repetitions in case of transmission error.

## Advanced Channel Parameters

The parameters of a channel are declared in the Advanced tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

computer name - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the Advanced 2 tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

number - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.80 NCP - Driver for MN-Series Drivers from Invensys

### Driver Use

The NCP driver is used to exchange data between the Asix system and MN-series controllers from Invensys (former Satchwell). Communication is performed by the use of serial links in standard RS-485.

Parameterization of the NCP driver is performed by the Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel operating according to NCP protocol requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel

*Driver:* NCP

**CTNCP** TAB:

*Channel parameter:*

*Port=port\_number, No.=slave\_number*

where:

*Port* - COM serial port number,

*No* - MN controller number set on the controller switch.

**Transmission parameters are constant and have the following values:**

- speed **9600** bps,
- 8 data bits,
- no parity control,
- 1 stop bit.

### EXAMPLE

Below is an example of the declaration of the CHANNEL channel, where the MN-440 controller with a network number of 5 is supported. The communication is performed through the port COM2:

*Name:* CHANNEL

*Driver:* CtNCP

*Channel parameters:* Port=2; No.=5

## Variable Declaration

The driver provides the following variable types:

B	- byte,
R	- 16-bit number,
RL	- 32-bit number,
RF	- floating point number.

The variable address has the following syntax:

*<Type><ArrNo><Index>*

where:

Type	- variable type name,
ArrNo	- data array number in the controller,
Index	- byte number in the array specified by 'r;ArrNo'.

### EXAMPLE

Examples of variables declaration:

X11, byte 0 from array 114, B114.0, CHANNEL, 1, 1, NIC\_BYTE

X12, WORD consisting of bytes 2 and 3 of array 110, R110.2, CHANNEL, 1, 1, NIC

X13, DWORD consisting of bytes from 4 to 7 of array 8, RL8.4, CHANNEL, 1, 1, NIC\_DW

X14, FLOAT consisting of bytes from 8 to 11 of array 12, RF12.8, CHANNEL, 1, 1, NIC\_FP

## Driver Parameterization

The NCP driver parameters are declared in the *Miscellaneous* module, on *Directly entered options* tab.

- Section name:** CTNCP
- Option name:** LOG\_FILE
- Option value:** *log\_filename*

Meaning - the diagnostic purposes are fulfilled by a text log file, where the driver operation status messages are entered.

Default value - by default, the log file is not created.

- Section name:** CTNCP
- Option name:** LOG\_FILE\_SIZE
- Option value:** *number*

Meaning - the option is used to determine the log file size.

Parameter:

*number* - log file size in MB.

Default value - log file size is 1 MB.

- Section name:** CTNCP
- Option name:** LOG\_OF\_TELEGRAMS
- Option value:** YES | NO

Meaning - option allows the contents of telegrams sent between the driver and controllers to the log file (declared with use of *LOG\_FILE* option) to

## Communication Drivers

be saved. The subject option should be used only during the start-up of the Asix system.  
Default value - NO.

**Section name: CTNCP**

**Option name: MAX\_BUFFER\_LEN**

**Option value: number**

Meaning - option determines the maximum number of bytes which can be asked for in a single readout request.

Default value - 64.

### EXAMPLE

Example of driver parameterization:

*Section name: CTNCP*

*Option name: LOG\_FILE*

*Option value: d:\tmp\CtNcp\ncp.log*

*Section name: CTNCP*

*Option name: LOG\_FILE\_SIZE*

*Option value: 3*

*Section name: CTNCP*

*Option name: LOG OF TELEGRAMS*

*Option value: YES*

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

 **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
 YES / NO

The default value - NO.

 **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.81 NetLink - Driver of MPI/Profibus Protocol for SIMATIC S7 by using NetLink Lite SYSTEME HELMHOLZ Module

### Driver Use

The NetLink driver is used for data exchange between Asix computers and SIMATIC S7 PLCs by using an MPI/Profibus.

The advantage of NetLink driver is that it uses inexpensive and easy to use the NetLink Lite module which is the gateway of Ethernet for MPI or PROFIBUS data bus of S7 controllers. This method of communication is an attractive alternative to solutions based on SOFTNET software and CP5611/CP5613 cards of SIEMENS.

NetLink Lite module is manufactured by Systeme Helmholtz GmbH, and is distributed in Poland by MEDIOTECH ([www.mediotech.pl](http://www.mediotech.pl)).

Limitations of using NetLink Lite:

- a) the module can handle maximally 2 clients at the same time;
- b) during communication between a Netlink Lite module and PLCs (more than one) it is necessary to take into account a time needed for the module to switch over between PLCs (an exemplary configuration having been tested consisted of 3 PLCs, and the time of switching amounted to about 80 msec/PLC).

The NetLink driver needs ASMEN v. 4.6.8 or newer one.

Parameterization of NetLink driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the NetLink driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: *NetLink*

**NetLink** tab:

*Channel parameters*:

*IP, port, address* [, *contr\_var* [, *alarm\_nr*] [, *error\_signal*]]

where:

- IP* - IP address assigned to the NetLink Lite module;
- port* - port number (1099);
- address* - PLC address on the MPI/Profibus Network;
- contr\_var* - variable name used for control of the PLC's RUN-STOP state;
- alarm\_nr* - alarm number generated when changing the PLC's RUN-STOP state; by default the alarm is not generated;
- error\_signal* - setting an error status for all the variables in a defined channel in case of switching the PLC to STOP; by default, an error status is set.

## Addressing Process Variables

The rules for creating symbolic addresses of variables belonging to the channel that uses the NetLink driver are the same as for the SAPI S7 driver - see: SAPI S7 - Driver of SAPI S7 Protocol for SIMATIC S7 PLCs.

## Driver Configuration

NetLink driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **NETLINK** section.

- Section name: NETLINK**
- Option name: STATISTICS**
- Option value: yes/no**

Meaning - the item allows to display (every 1 minute) information about number of transmission sessions that have been carried-out, average transmission time and number of transmission errors. The item was developed as a designer support on the stage of the Asix system start-up.

Default value - by default, the transmission statistics is not displayed.

- Section name: NETLINK**
- Option name: LOG\_FILE**
- Option value: file\_name**

Meaning - the item allows to define a file to which all NetLink driver messages concerning operations performed by the driver are written. If the item does not define the full name, then the log file is created in the current directory.

Default value - by default, the log file is not created.

- Section name: NETLINK**
- Option name: LOG\_FILE\_SIZE**
- Option value: number**

Meaning - declares the log file size.

Default value - 1MB.

Defining - manual.

Parameters:

number - the log file size in MB;

- Section name: NETLINK**
- Option name: TELEGRAM\_LOG**
- Option value: YES/NO**

Meaning - declaration of writing the contents of telegrams sent and received by the NetLink driver within reading/writing process variables to the log file declared in the item LOG\_FILE.

Default value - NO.

### Time Synchronization

See: [SAPIS7 - Driver of SAPIS7 Protocol for SIMATIC S7 PLCs.](#)

### Signalization of the Controller's STOP State

See: [SAPIS7 - Driver of SAPIS7 Protocol for SIMATIC S7 PLCs.](#)

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

#### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

#### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- by default, all computers can read variables.

#### **Redundancy**

Purpose - with this option searches a redundant channel when connection with a controller is broken in local channel.

Option value:

*computer name*

- list including the names of network computers which will be permitted to search a redundant channel.

The default value

- by default, when option is enabled, the channel is searched on all computers.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.82 NetLinkPro - Driver for Communication with the S7 Controllers

The NetLinkPro driver (manufactured by SYSTEME HELMHOLZ and distributed in Poland by [www.mediotech.pl](http://www.mediotech.pl)) task is to communicate with the S7 controllers via the NETLink PRO gateway (an Ethernet <-> MPI/Profibus gateway).

Basic features of the NETLink PRO driver include:

**a/** at the Ethernet side it may support simultaneously up to 6 PC computers

**b/** at the MPI/PROFIBUS side it may support simultaneously up to 12 controllers.

The NetLinkPro driver requires the ASMEN module version 4.13.1 or higher.

Parameterization of NetLinkPro driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the NetLink driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel

*Driver:* NetLinkPro

**NetLinkPro** tab:

*Channel parameters:*

*deviceNr, slaveNr* [, *contr\_var* [, *alarm\_nr* [, *error\_signal*]]]

where:

- |                     |  |
|---------------------|--|
| <i>deviceNr</i>     | - number of a NetLink Pro module (with use of configuration software);                                     |
| <i>slaveNr</i>      | - number of a controller in MPI/PROFIBUS network;  |
| <i>contr_var</i>    | - variable name used for control of a PLC's RUN-STOP state;  |
| <i>alarm_nr</i>     | - alarm number generated when changing the PLC's RUN-STOP state; by default the alarm is not generated;    |
| <i>error_signal</i> | - setting an error status for all the variables in a defined channel in case of switching the PLC to STOP. |

### Addressing Process Variables

The rules for creating symbolic addresses of variables belonging to the channel that uses the NetLinkPro driver are the same as for the AS512 driver - see: *AS512 - Driver of AS512 Protocol for SIMATIC S5 PLCs* (in "Communication Drivers - User's Manual").

The set of process variables types for NetLinkPro driver protocol has been enriched (compared to AS512 protocol) with the following items:

- EDI - 16-byte words in INTEL convention,
- ER - floating-point number in a data block,
- EB - contents of the data blocks treated as bytes.

The NetLinkPro driver is loaded as a DLL automatically.

## Parameters of NetLinkPro Driver

NetLinkPro driver parameters are declared in: Architect > *Miscellaneous* module > *Directly entered options* tab.

The driver is parameterized with use of **NETLINKPRO** section.

**Section name: NETLINKPRO**

**Option name: LOG\_FILE**

**Option value: file\_name**

Meaning - the item allows to define a file to which all NetLinkPro driver messages concerning operations performed by the driver are written. If the item does not define the full name, then the log file is created in the current directory.

Default value - by default, the log file is not created.

**Section name: NETLINKPRO**

**Option name: TELEGRAM\_LOG**

**Option value: file\_name**

Meaning - declaration of writing the contents of telegrams sent and received by the NetLinkPro driver within reading/writing of process variables to the log file declared in the LOG\_FILE item.

Default value - NO.

**Section name: NETLINKPRO**

**Option name: MAX\_BUFFER\_LEN**

**Option value: number**

Meaning - the length of telegrams accepted by an MPI interface depends on the CPU type of S7 controllers and on a program executed by them.

Default value - by default, length of a telegram equals 220 bytes.

Parameter:

*number* - length of telegrams passed in bytes.

**Section name: NETLINKPRO**

**Option name: CONSOLE**

**Option value: YES|NO**

Meaning - the item allows to create a console window where NetLinkPro driver messages concerning operations executed by the driver are displayed currently.

Default value - by default, any console window is not created.

**Section name: NETLINKPRO**

**Option name: STATISTICS**

**Option value: YES|NO**

Meaning - the item allows to display (every 1 minute) information about number of transmission sessions that have been carried-out, average transmission time and number of transmission errors. The item was developed as a designer support on the stage of the Asix system start-up.

Default value - by default, the transmission statistics is not displayed.

**Section name: NETLINKPRO**

**Option name: NUMBER\_OF\_CHECK\_READINGS**

**Option value: number**

Meaning - the item specifying a minimal number of successive readings of control variable (of a constant value) which cause a signalization of the controller STOP status.

Default value - by default, the item assumes a value of 3.

**Section name: NETLINKPRO**

**Option name: SERIALISING**

**Option value: YES|NO**

Meaning - declaration of servicing the transmission from the S7 by transferring single YES or many NO queries.

Default value - YES.

**Section name: NETLINKPRO**

**Option name: WITHOUT\_MULTIPLE\_READ**

**Option value: YES|NO**

Meaning - it allows to block the mode of multiple read creation and enables driver operation in the mode of single read.

Default value - No; in the current version the s7\_multiple\_read\_req function is used by default (it allows to build a multiple read) for maximal use of the length of the telegram buffer sent between PC and S7.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:  
YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).  
 The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).  
 The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO  
 The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.  
 The default value - by default, no time for data validity is set.

## 1.83 NETWORK Driver

The **NETWORK** protocol used by the NETSRV.EXE program is a special type of protocol. It does not provide a physical link to a controller, but only connection to any other computer that has such a link. The driver of the network channel, that have established the link, sends a query to all servers, defining the logical name of a channel. The query is responded by only those servers, where such a channel has been previously declared. If more that one server responds the query, the one with the less loaded is chosen to establish the link. If the chosen server is turned off during operation, network link will be established with another server.

The network channel can be used for creating the network terminals that use current variables of Asix.

## Declaration of Transmission Channel

Declaration of the transmission channel using the NETWORK driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* NETWORK

## Driver Configuration

**NETWORK** tab:

- Option name:** List of network names of data servers
- Option value:** *network\_server\_names*

Meaning	- if data server names are not defined then the workstations can connect to any data server with a required transmission channel declared (the channel with the same name as the channel with the NETWORK driver declared). Otherwise the set of servers will be limited to the list declared in the option.
Default value	- NO.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**☑ Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**☑ Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**☑ Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

**☑ Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.84 NONE Driver

### Driver Use

The NONE driver uses the NONE protocol internally served by ASMEN. That protocol doesn't realize a physical connection with the controller. It may be used for:

- application testing in simulation mode,
- data exchange between Asix applications by means of process variables.

### Declaration of Transmission Channel

Declaration of the transmission channel using the NONE driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel

*Driver:* NONE

### Driver Configuration

**NONE** tab:

**Option name:** *Writing data and status*

**Option value:** YES[NO]

Meaning - the option allows to determine a time period after exceeding of which the current time should be sent to remote devices.

Default value - NO.

The item **Writing data and status** causes that in the channel only writing with variable value supplemented with date and status is possible. At present this property is available only from the level of the **AsixConnect** function (write2()).

A status, time and variable value are a subject to the following rules:

- to the moment of the first writing, while reading, the status AVD\_BAD is returned;
- while writing, the time given in writing parameters, new status and new variable value is written to the variable;
- the reading following the writing returns parameters of the lately performed writing (value, status and time);
- writing to the variable by using previous writing functions ends with an error code of 24;
- for needs of DEMO (program AS32\_demo.exe) objects may perform writes to variables belonging to the NONE channel with the item **Writing data and status** declared.

**Option name: *Passive channel*** **Option value: YES[NO]**

Meaning	- the option allows to treat the channel as a passive one. The option determines the way the Asix interprets the lack of new data in the channel: - when passive channel - the lack of new data means that the previous data with statuses is still valid; - when active channel - (default value) the lack of new data means communication errors. The option is meaningful only when the option <i>Writing data and status</i> is used at the same time.
Default value	- NO.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

 **Remote Write Access**

Purpose	- this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.
Option value: YES / NO <i>computer name</i>	- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).
The default value	- NO - by default, only local computers can modify variable values.

 **Remote Read Limitation**

Purpose	- with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.
Option value: <i>computer name</i>	- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
The default value	- by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

 **Local Write Denied**

Purpose	- with this option deployed, local control in a channel will be disabled.
---------	---

## Communication Drivers

Option value:

YES / NO

The default value - NO.

### **Data Validity Period**

Purpose

- this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value

- by default, no time for data validity is set.

## 1.85 NordicRF - driver of Nordic ID RF 601 Terminal Made by NordicID Company

### Driver Use

**NordicRF**. The exchange of data with Nordic ID RF 601 barcode scanner. Communication takes place using the RS-232 serial interface of the base station.

Parameterization of NordicRF driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the NordicRF driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: NordicRF

**CtNordicRF** tab:

*Channel parameters*:

Port=*PortNo* [;*ValidTimeout*=*timeout*] [;*Alarm*=*AlarmNo*]

where:

Port - number of serial port COM.

*ValidTimeout* - time (in minutes) between the two telegrams from a terminal after which data received by the driver will be invalidated. Default value: 0 (without invalidating data).

*Alarm* - number of an alarm called to alarm system of Asix when exceeding *ValidTimeout*.

Transmission is realized at constant settings:

19200 Bd, 8 bites of mark, no parity control, 1 stop bit.

### Declaration of Variables

The driver provides one type of variables:

**D - text type**

The syntax of variable address is as follows:

**D.<termId>.<position>**

where:

*termId* - ID of the terminal (16-bit unsigned)

*position* - number of position on the terminal display; the value of position is assigned to a variable. By default, two following parameters can be read from the terminal:

1/ Count - position 26

2/ Code - position 46

**NOTE:**

All variables have the BSTR type, therefore you should use the conversion function NOTHING\_TEXT.

**EXAMPLE**

Exemplary declarations of variables for the terminal with IDs: 30002 and 30004 (variables belong to the logical channel CHANNEL):

JJ\_11, field Count, D.30002.26, CHANNEL, 1,1,NOTHING\_TEXT

JJ\_12, field Code, D.30002.46, CHANNEL, 1,1,NOTHING\_TEXT

jj\_21, field Count, D.30004.26, CHANNEL, 1,1,NOTHING\_TEXT

JJ\_22, field Code, D.30004.46, CHANNEL, 1,1,NOTHING\_TEXT

## Parameter of Drivers

NordicRF driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **NETLINKPRO** section.

- Section name: CtNordicRF**
- Option name: LOG\_FILE**
- Option value: file\_name**

Meaning	- it allows to define a file to which all diagnostic messages of the driver and the information about the content of telegrams received by the driver will be written.
Default value	- log file is not created.

**Section name: CtNordicRF** **Option name: LOG\_FILE\_SIZE** **Option value: number**

Meaning - it allows to determine a log file size in MB.

Default value - 10 MB

 **Section name: CtNordicRF** **Option name: LOG\_OF\_TELEGRAMS** **Option value: YES|NO**

Meaning - the item allows to write to the log file (declared by using an item LOG\_FILE) the content of telegrams received by the driver.

Writing the content of telegrams should be used only while the Asix start-up.

Default value - NO

**EXAMPLE**

```
[CTNORDICRF]
```

```
LOG_FILE =d:\tmp\test\centrala.log
```

```
LOG_FILE_SIZE =30
```

```
LOG_OF_TELEGRAMS =YES
```

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

 **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

 **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

## Communication Drivers

- computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.86 OMRON - Driver of HOSTLINK Protocol

### Driver Use

The OMRON driver protocol is used for data exchange with OMRON PLCs. The transmission is executed by means of serial interfaces HOSTLINK by using standard serial ports of an Asix system computer.

The cooperation of Asix with the controller by using the OMRON protocol does not require any controller's program adaptation. Before executing controls the driver switches the controller in MOTOROLA mode (if the controller is in the run mode). After control is ended, the controller is switched to mode, in which was operating before executing the control.

Parameterization of CtM200 driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel utilizing the OMRON driver requires a channel with the following parameters to be added to the Current data module:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: OMRON

**OMRON** tab:

*Channel parameters*: *type*[,*id*],*port*,[*baud*,*character*,*parity*,*stop*]

where:

<i>type</i>	- connection type - SLINK (single link), MLINK (multi link);
<i>id</i>	- controller identifier (unit number), it is used when the connection type was marked as MLINK (multi link);
<i>port</i>	- name of the serial port (COM1 or COM2);
<i>baud</i>	- transmission speed in baud;
<i>character</i>	- number of bits in a transmitted character;
<i>parity</i>	- parity check type (even,odd,none0);
<i>stop</i>	- number of stop bits.

The parameters *baud*, *character*, *parity*, *stop* are optional. In case of omitting them the following default values are assumed:

- transmission speed - 9600 Bd,
- number of bits in a character - 7,
- parity check type - parity check (even),
- number of stop bits - 2.

#### EXAMPLE

An exemplary declaration of transmission channel working according to the OMRON protocol:

*Channel / Name*: CHAN1

*Driver:* OMRON

*Channel parameters:* MLINK,0,COM1,9600,7,even,2

The transmission channel with the logical name CHAN1 has the following parameters defined:

- OMRON protocol using a serial interface working in MLINK (multi-link);
- identifier of the controller (unit number) 0;
- port COM1;
- transmission channel of 9600 Bd;
- transmitted character length - 7 bits;
- parity check;
- two stop bits.

## Transmission Channel Parameters

The transmission channel is parameterized by using a separate section with the name the same as the name of the channel. Parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab:

- using the FINS command,
- network number with the controller,
- node number in the network assigned to the controller.

### Using the FINS command

**Section name:** *channel\_name*

**Option name:** PROTOKOL\_FINS

**Option value:** YES/NO

Meaning - it forces using the commands of FINS during the data exchange with the controller.

Default value - NO. By default, the communication is done using C-MODE commands.

If the options: DNA and DA1 are not used in the channel section, then for the communication in the FINS mode it is assumed that:

- controller network number (DNA),
- controller node number (DA1)

have the value of 0.

### Controller network number

**Section name:** *channel\_name*

**Option name:** DNA

**Option value:** *number*

Meaning - the option sets the controller network number to the value of DNA and it forces the use of commands of FINS during data exchange with the controller.

Parameter:

*number* - number of network of the controller.

Default value - 0 (local network).

### Controller node number

**Section name:** *channel\_name*

**Option name:** DA1

**Option value:** *number*

Meaning - The option sets the node number of the controller to the value of the parameter DA1 and forces the use of commands of FINS during data exchange with the controller.

Parameter:

*number* - controller node number.

Default value - 0 (internal communication in local PLC).

## Addressing the Process Variables

The syntax of symbolic address which is used for variables belonging to the OMRON driver channel is as follows:

*variable\_type variable\_index*

where:

*variable\_type* - string identifying the variable type in the OMRON protocol,

*variable\_index* - variable index within a given type.

**Types of variables available in C-MODE** (in parentheses there is a range of variable indexes).

IR - Internal Relay, (0 - 235, 300 - 511),  
 HR - Holding Relay, (0 - 99),  
 AR - Auxiliary Relay, (0 - 27),  
 LR - Link Relay, (0 - 63),  
 DM - Data Memory, (0 - 6143).

All the process variables are treated as 16-bit numbers.

### Types of variables available in FINS:

16-bit types (WORD):

CIO - CIO Area  
 T - Timer Area  
 C - Counter Area  
 WR - Work Area  
 HR - Holding Area  
 AR - Auxiliary Bit Area

32-bit types (DWORD):

ARL - Auxiliary Bit Area  
 CIOL - CIO Area  
 CL - Counter Area  
 DML - DM Area  
 HRL - Holding Area  
 TL - Timer Area  
 WRL - Work Area

32-bit types (FLOAT):

## Communication Drivers

ARF - Auxiliary Bit Area  
CIOF - CIO Area  
CF - Counter Area  
DMF - DM Area  
HRF - Holding Area  
TF - Timer Area  
WRF - Work Area

### 64-bit types (DOUBLE):

ARD  
CD  
CIOD  
DMD  
HRD  
TD  
WRD - Auxiliary Bit Area  
- Counter Area  
- CIO Area  
- DM Area  
- Holding Area  
- Timer Area  
- Work Area

### 64-bit types (INT64):

ARI  
CI  
CIOI  
DMI  
HRI  
TI  
WRI - Auxiliary Bit Area  
- Counter Area  
- CIO Area  
- DM Area  
- Holding Area  
- Timer Area  
- Work Area

## EXAMPLES

Exemplary variable declarations:

Name: JJ\_02  
(C6 i C7 jako DWORD)  
Address: CL6  
Channel: K1  
ElementsCount: 1  
SampleRate: 1  
ConversionFunction: NOTJING\_DW

Name: JJ\_03  
(DM480 i DM481 as FLOAT)  
Address: DMF480  
Channel: K1  
ElementsCount: 1  
SampleRate: 1  
ConversionFunction: NOTJING\_FP

Name: JJ\_04  
(DM480 - DM483 as DOUBLE)

Address: DMD480  
 Channel: K1  
 ElementsCount: 1  
 SampleRate: 1  
 ConversionFunction: NOTJING\_DOUBLE

Name: JJ\_05  
 (DM500 - DM503 as INT64)  
 Address: DMI480  
 Channel: K1  
 ElementsCount: 1  
 SampleRate: 1  
 ConversionFunction: NOTJING\_INT64

## Driver Configuration

OMRON driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

- Section name: OMRON**
- Option name: LOG\_FILE**
- Option value: file\_name**

Meaning - it allows to define a file to which all diagnostic messages of the driver and the information about the content of telegrams received by the driver will be written. If the item does not define the full path, then the log file will be created in the current directory. The log file should be used only while the Asix start-up.

Default value - log file is not created.

Defining - manual

- Section name: OMRON**
- Option name: LOG\_FILE\_SIZE**
- Option value: number**

Meaning - it allows to determine a log file size in MB.

Default value - 1MB

Defining - manual

- Section name: OMRON**
- Option name: LOG\_OF\_TELEGRAMS**
- Option value: YES|NO**

Meaning - the item allows to write to the log file (declared by using an item LOG\_FILE) the content of telegrams received by the driver. Writing the content of telegrams should be used only while the Asix start-up.

Default value - NO

Defining - manual

- Section name: OMRON**
- Option name: MAX\_BUFFER\_LEN**
- Option value: number**

Meaning - maximal length of answer telegrams (counted in bytes). Maximal value is equal to 118.

Default value - 118 bytes

Defining - manual

**Section name: OMRON**

**Option name: CHARACTER\_TIMEOUT**

**Option value: *id,number***

Meaning - option allows to specify the maximum time (in seconds) between consecutive characters of response from the specified device. After this time, it is considered that this device is not working properly and the transmission session is completed with an error.

Parameter:

*id*

- number of the meter in K3N network;

*number*

- time in milliseconds (from 10 to 300).

Default value

- 50 milliseconds.

**Section name: OMRON**

**Option name: RECV\_TIMEOUT**

**Option value: *number***

Meaning - specifies the maximum time for waiting for device response.

Default value - 0 (no repetition)

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.87 OmronTcpiip - Driver for Data Exchange with Omron Controllers Supporting the FINS/UDP and FINS/TCP Protocols

### Driver Use

OmronTcpiip protocol driver is used for data exchange between Asix system and Omron controllers supporting the FINS/UDP and FINS/TCP protocols. The driver provides communication with the CS/CJ and CV Series controllers.

OmronTcpiip driver configuration is performed using the Architect application.

### Transmission Channel Declaration

Declaration of the transmission channel operating according to the OmronTcpiip driver protocol requires adding a channel with the following parameters to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel

*Driver:* OmronTcpiip

**OmronTcpiip/Channel parameters** tab:

<b>IP address of the device</b>	- controller IP address,
<b>Node number of the device</b>	- controller node number (UDP mode only)
<b>Node number of the PC</b>	- PC node number (UDP mode only)
<b>Alarm number</b>	- The alarm number reported to Asix alarm system in the absence of communication with the controller,
<b>Timeout</b>	- Max. time (in milliseconds) to wait for a response from the controller. By default, 500 milliseconds,
<b>Port</b>	- Controller TCPIP port number supporting the FINS protocol. The default port is 9600.

Example of transmission channel declaration for communication with the controller, IP address 10.10.105.212 in UDP mode (DA1 = 212, SA1 = 50) with time synchronization every minute:

IP=10.10.105.212; DA1=212; SA1=50; Time synchronization=60

Example of transmission channel declaration for communication with the controller, IP address 10.10.105.212 in TCP mode with time synchronization every minute:

IP=10.10.105.212; Time synchronization=60

## Declaration of Variables

The general syntax of the process variable symbolic address is as follows:

*<variable\_type><variable\_index>*

List of 16-bit types (WORD):

AR - Auxiliary Bit Area  
 C - Counter Area  
 CIO - CIO Area  
 DM - DM Area  
 HR - Holding Area  
 T - Timer Area  
 WR - Work Area

List of 32-bit types (DWORD):

ARL - Auxiliary Bit Area  
 CL - Counter Area  
 CIOL - CIO Area  
 DML - DM Area  
 HRL - Holding Area  
 TL - Timer Area  
 WRL - Work Area

List of 32-bit types (FLOAT):

ARF - Auxiliary Bit Area  
 CF - Counter Area  
 CIOF - CIO Area  
 DMF - DM Area  
 HRF - Holding Area  
 TF - Timer Area  
 WRF - Work Area

List of 64-bit types (DOUBLE):

ARD - Auxiliary Bit Area  
 CD - Counter Area  
 CIOD - CIO Area  
 DMD - DM Area  
 HRD - Holding Area  
 TD - Timer Area  
 WRD - Work Area

List of 64-bit types (INT64):

ARI - Auxiliary Bit Area  
 CI - Counter Area  
 CIOI - CIO Area  
 DMI - DM Area  
 HRI - Holding Area  
 TI - Timer Area  
 WRI - Work Area

The driver does not validate the variable index range.

Examples of variable declarations:

JJ\_01, HR number 100 WORD, HR100, K1, 1, 1, NIC  
JJ\_02, C6 and C7 as a DWORD, CL6, K1, 1, 1, NIC\_DW  
JJ\_03, DM480 and DM481 as FLOAT, DMF480, K1, 1, 1, NIC\_FP  
JJ\_04, DM480 - DM483 as DOUBLE, DMD480, K1, 1, 1, NIC\_DOUBLE  
JJ\_05, DM500 - DM503 as INT64, DMI480, K1, 1, 1, NIC\_INT64

## Driver Parameters

The OmronTcpi driver is declared using options from the *Driver parameters* tab.

**Log file**

Meaning - the text log file to which driver status messages are stored is used for diagnostic purposes.

Default value - Log file is not created.

Parameter:

*log\_file\_name*

**Log file size**

Meaning - determines the log file size in MB.

Default value - 10 MB.

Parameter:

*number* - number in MB.

**Log of telegrams**

Meaning - This option allows you to save the contents of messages received by the driver in the log file (declared using the *Log file* option) Writing of the messages in the log file should be used only during the Asix system start-up.

Default value - NO.

Parameter:

*YES/NO*

**IP address of the computer**

Meaning - Allows the declaration of a network adapter through which the driver communicates with the controller.

Default value - By default, the option value is set to NO.

Option value:

*IPaddress* - IPv4 address notation.

## Sample Driver Parameterization

*Log file:* d:\tmp\CtOmronTcpip\mvi.log

*Log file size:* 20

*Log of telegrams:* YES

*IP address of the computer:* 10.10.105.23

## Channel Parameters

The transmission channel is configured using a separate options from the *Channel parameters 2* tab.

### **Log file**

Meaning - the text log file to which driver status messages are stored is used for diagnostic purposes.

Default value - Log file declared in driver options.

Parameter:

*log\_file\_name*

### **Log file size**

Meaning - determines the log file size in MB.

Default value - declared in driver options.

Parameter:

*number* - number in MB.

### **Log of telegrams**

Meaning - This option allows you to save the contents of messages received by the driver in the log file (declared using the *Log file* option) Writing of the messages in the log file should be used only during the Asix system start-up.

Default value - declared in driver options.

Parameter:

*YES/NO*

### **UDP mode**

Meaning - Allows you to set the UDP mode of communication.

Default value - NO, by default, the communication mode is set to TCP.

Parameter:

*YES/NO*

### **UTC tme synchronization**

Meaning - Allows you to specify a time synchronization using UTC time format instead of local time.

Default value - NO, by default, the local time is communicated to the controller.

Parameter:

*YES/NO*

### **Time synchronization**

Meaning - the period of time synchronization.

Default value - 1.

Parameter:

*number* - time in seconds (0 - off)

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

- Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.
- Option value:  
YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

- Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.
- Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - by default, all computers can read variables.

### **Redundancy**

- Purpose - with this option searches a redundant channel when connection with a controller is broken in local channel.
- Option value:  
*computer name* - list including the names of network computers which will be permitted to search a redundant channel.
- The default value - by default, when option is enabled, the channel is searched on all computers.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

- Purpose - with this option deployed, local control in a channel will be disabled.
- Option value:  
YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.88 OPC Driver

### Driver Use

The OPC driver is used for data exchange between Asix system computers and any industrial PLC or SCADA program that has an access to a data server compatible with the OPC 1.0 or OPC 2.05 specification. (OPC specification is available at <http://www.opcfoundation.org>).

The OPC driver handles operations of read/write from/into a controller of:

1. simple variables containing scalar values like:
  - a. Byte - 8-bit unsigned number;
  - b. Word - signed or unsigned word (16-bit number);
  - c. Double Word - signed or unsigned double word (32-bit number);
  - d. Single Precision - 32-bit real number;
2. single dimension table variables consisting of simple variables (*see: 1*);
3. text variables.

### Changes in OPC Driver - Version 2.0

- Adding the OPC server service compatible with the OPC 2.05 specification.
- Adding the possibility of specifying the access path for variables in the transmission channel.
- Adding the possibility of blocking the operation of writing to the OPC server.
- Driver extension with read/write operation of text / table variables.
- Elimination of large OPC server loading reducing considerably its capacity in case when a variable archiving period was too long in relation to their sampling period.

### Declaration of Transmission Channel

The declaration of the transmission channel operating according to OPC protocol requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* OPC

**OPC/Channel parameters/Main** tab:

The basic and the only obligatory OPC driver option is  **Identifier of OPC server in Windows system.**

**Update mechanism** - mechanism of updating values from the OPC server. Available values:

- Automatic selection,
- Asynchronous according to OPC specification 1.0,
- Asynchronous according to OPC specification 2.05,
- Synchronous.

**Blocking of all writes to OPC server** - if the option has the value 'Yes', then all write operations to the OPC server are blocked.

**Period of reading all measurements** - reading time in [ms]. The option should be used only for the OPC servers that contain errors and don't always send information on changes in values of variables. Cyclic reading time should be adequate long (tens of seconds) so that the OPC server is not overloaded.

### EXAMPLE

An exemplary declaration of the channel recalling to the OPC server registered under the *Matrikon.OPC.Simulation* identifier:

*Channel / Name:* Matrikon  
*Driver:* OPC  
*Identifier of OPC Server:* Matrikon.OPC.Simulation

An exemplary declaration of the channel using all options:

*Channel / Name:* Matrikon  
*Driver:* OPC  
*Identifier of OPC Server:* Matrikon.OPC.Simulation  
*Blocking of all writes to OPC server:* yes,  
*Items always active:* yes,  
*OPC version:* 2

### **OPC/Channel parameters/Advanced** tab:

**Status item of channel** - option allows to declare a variable used to signal the channel status (information on damage or proper operation of the channel). There are two possibilities: declaration of values of variables denoting the proper operation of the channel (then other value of the variable denotes an error in the channel) or declaration of values signaling a damage of the channel (then other value of the variable denotes a proper operation of the channel).

**Variables always active** - every variable is updated by an OPC server regardless of whether the variable is at the moment used in the Asix system.

**OPC/Channel parameters/Remote server** tab:

**Remote server** - when using this option, the designer is solely responsible for proper configuration of protections on both computers so that the client and server can exchange data with each other using DCOM.

Using this option requires on first tab *Identification* to be used, because OPC server isn't registered on the computer on which the Asix system works.

If no server name is declared, the local computer will be used.

**Data of the user on whose account the OPC server will be run** - the option needs the option *Remote server* to be declared. The data of the user account:

*Domain or computer name;*

*User name;*

*User password.*

The other group of options is declared by **OPC/Channel parameters/Diagnostics - Log** tab:

**Log of successful initializations**

Meaning - allows to store in the log the information on successful initializations of item on the OPC server.

Default value - NO.

**Log of successful writes**

Meaning - the information on successful end of the operation of writing to the OPC server is placed in the log.

Default value - NO.

**Log of transmitted items**

Meaning - the information on the items transmitted by the server is placed in the log.

Default value - NO.

**Log of statistics of data change**

Meaning - the information on the number of items transmitted by the server and time of data processing by the driver is placed in the log.

Default value - NO.

**OPC/Channel parameters/Diagnostics - Trace** tab:

**Traced names**

Meaning - for each variable, which name is on the list, the information on its value, quality and stamp will be written to the log (during the variable processing).

Default value - lack.

Parameter:

*variable\_list* - list of the variable names in the Asix system, delimited with commas.

## Defining the Process Variables

Defining the variables in the Asix system is described in *Asix, User Manual*, in chapter *Asmen - Communication Manager* (see: *Declaration of Process Variables*).

When defining the variable (recalling to an OPC server), the variable identifier from the OPC server variable base should be declared as the variable address. The syntax of the identifier depends on the specific OPC server and is described in its documentation. If the identifier contains small letters, comma or white space, then the identifier (the whole one) should be put in quotation marks.

Notice: It is possible to display the window of selecting a variable from the OPC server and to configure the address of the variable using this window. The window is activated in Architect > variable definition database editor > by clicking in the field of the *Address* column.

See: informations on import of variable definitions from the OPC server: Architect.CHM/PDF, 3.2.1. *Creating the new application - Wizard - base import from OPC*.

## Type Matching

While initializing the variable in the OPC server, the OPC driver sends a requested type of the variable to it. This type is defined according to the conversion function that has been assigned to the variable in the Asix VariableBase. If the variable type requested by the OPC driver differs from the type declared in the OPC variable base, then the OPC server usually accepts the requested type and performs the proper conversion during the data transmission. If the OPC server is not able to perform the proper conversion, then an error is notified at the moment of variable initialization. This fact is also signaled in the list of system messages (in *'Control Panel'*) as the message *"Incorrect variable type"*. The variable the initialization of which finished with an error will have a bad status.

If the error connected with the variable type misfit in the Asix system and the OPC server occurs, then the other analogical conversion function (but operating with the variable type handled by the OPC server) should be used. The variable types used by conversion functions are described in the Asix system documentation, in chapter *ASMEN - Communication Manager/Conversion Functions* (the type of variables requested by the OPC server is named *type of PLC variable*).

The variable type misfit most often occurs when the conversion functions that operate on unsigned numbers are used. It is because some OPC servers don't handle such numbers at all. In such case, it is impossible to use the following functions: TIME, CIRCLE1, COUNTER, MASK, FACTOR, FACTOR\_DW, NEGBIT, NEGBIT\_DW, NOTHING, NOTHING\_BYTE, NOTHING\_DD, NOTHING\_DW, ON/OFF, SHIFT\_L, SHIFT\_R, SLIDER, SLIDER1, SLIDER1\_FP.

There are three conversion functions handling any type of values being received from the controller: GRADIENT, AVERAGE, TABLE. For these functions, the OPC driver always requests from the OPC server sending the variable values in the form of floating-point numbers - because the floating-point number is always these functions' output type.

## Communication Testing

The OPC driver enters on the message list of '*r;Control Panel*' information on important events like driver loading, driver connection to the OPC server, variable initialization performance. The information on possible errors at the driver initialization and operation stage is also entered on the list.

Detailed information on errors and diagnostic information is placed in the OPC log file. The driver log file is named *UniDriver,<current\_date>.log* and is placed in the Asix system directory by default. There are two groups of options defining the kind of information written to the log file. The first group options are declared with use of **OPC/Driver parameters - UniDriver** tab:

**Path to the directory in which the log file will be created**

Meaning - the log file is placed in the defined directory.

Default value - by default, the log file will be created in the Asix directory.

Parameter:

*record\_track* - path to the directory in which the log file will be created.

**Traced names**

Meaning - for each variable, which name is on the list, the information on its value, quality and stamp will be written to the log (during the variable processing).

Default value - lack.

Parameter:

*variable\_list* - list of the variable names in the Asix system, delimited with commas.

All the options concerning the communication test may be changed during the Asix system operation. After the modification and writing the application initialization file on the disc, the new option values will be retrieved from the log by the OPC driver and will begin to have an effect on the range of information to be written to the log.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify

variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

- Option value:  
 YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

- Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - by default, all computers can read variables.

**Redundancy**

Purpose - with this option searches a redundant channel when connection with a controller is broken in local channel.

- Option value:  
*computer name* - list including the names of network computers which will be permitted to search a redundant channel.
- The default value - by default, when option is enabled, the channel is searched on all computers.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

- Option value:  
 YES / NO
- The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

- Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.
- The default value - by default, no time for data validity is set.

## 1.89 PA5 - PA5 Flow Counter Protocol Driver

### Driver Use

The Pa5 protocol driver allows the exchange of data between the Asix system and the PA-5 transducers manufactured by Poznań-based POWOGAZ S.A. Water Meter Factory.

The communication can be performed in two modes:

- A/** point-to-point - the Asix system and the transducer are connected with the RS-232 link,
- B/** multidrop – communication with the transducer is performed by using the addressable ADAM-4521 modules connected into RS-485 network.

Parameterization of the PA5 driver is performed by the Architect application.

### Declaration of Transmission Channel

The declaration of the transmission channel operating according to Pa5 protocol requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* PA5

**Pa5** tab:

*Channel parameters:*

**Port=number** [**;ReceiveTimeout=number**] [**;ServiceType=number**]  
**;****ScanPeriod=number** [**;CharTimeout=number**]

where:

*Port* - computer's COM serial port number,  
*ReceiveTimeout* - waiting time for the first response character (in milliseconds); by default - 500 milliseconds;  
*CharTimeout* - time between response characters (in milliseconds); by default - 100 milliseconds;  
*ServiceType* - RS-232 interface (mode 1) or RS-485 interface with ADAM-4521 modules (mode 2); default mode is mode 2;  
*ScanPeriod* - time in milliseconds between subsequent readings of the transducer; by default - 30 seconds (manufacturer's recommendation). Parameter applies to all transducers supported in the given channel.

**NOTE** *Communication with the transducer is performed by using standard transmission parameters i.e. speed of 1200 Bd, 8 character bits and 1 stop bits.*

**NOTE** *In the case of the use of addressable ADAM-4521 modules, the following parameters should be set in ADAM-4521 module:*

**a/** delimiter - { (factory preset);  
**b/** add cr - yes (factory preset);  
**c/** address in RS-485 network (different address for each module);  
**d/** baud rate - 1200 Bd (transmission speed used in PA-5 transducer).

## EXAMPLE

An example of the declaration of the transmission channel using COM2, work mode with addressable ADAM-4521 modules and transducer reading every 40 seconds:

Name: CHANNEL  
 Driver: CtPA5  
 Channel parameters: Port=2; ScanPeriod =40

## Variable Declaration

The variable address has the following syntax:

V.<address>.<index>

where:

V	- variable name,
address	- transducer address (important for RS-485 network using addressable ADAM-4521 modules. In case of RS-232 connections, parameter value can be custom),
index	- values from 1 to 4 are assumed:
	1 - order no. of the water counter type,
	2 - current flow percentage value,
	3 - current A counter status,
	4 - current B counter status,

All variables return values of FLOAT type.

## EXAMPLE

Examples of variables declaration:

JJ_01, meter A of transducer no. 1,	V.1.3, CHANNEL, 1, 1, NIC_FP
JJ_02, meter B of transducer no. 2,	V.2.4, CHANNEL, 1, 1, NIC_FP

## Driver Parameters

The PA5 driver parameters are declared in the *Miscellaneous* module, on *Directly entered options* tab.

- Section name:** CtPA5
- Option name:** LOG\_FILE
- Option value:** log\_filename

Meaning - the diagnostic purposes are fulfilled by a text log file, where the driver operation status messages are entered.

Default value - by default, the log file is not created.

**Section name: CtPA5**

**Option name: LOG\_FILE\_SIZE**

**Option value: number**

Meaning - option is used to determine the size of the log file defined with use of the LOG\_FILE option.

Default value - default log file size is 10 MB.

Parameter:

*number* - log file size in MB.

**Section name: CtPA5**

**Option name: LOG\_OF\_TELEGRAMS**

**Option value: YES | NO**

Meaning - option allows the contents of telegrams sent between the driver and controllers to the log file (declared with use of LOG\_FILE option) to be saved. The subject option should be used only during the start-up of the Asix system.

Default value - by default, the option value is set to NO.

**EXAMPLE**

Example of driver parameters:

*Section name: CtPA5*

*Option name: LOG\_FILE*

*Option value: d:\tmp\ctpa5\pa5.log*

*Section name: CtPA5*

*Option name: LOG\_FILE\_SIZE*

*Option value: 20*

*Section name: CtPA5*

*Option name: LOG\_OF\_TELEGRAMS*

*Option value: YES*

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
 YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.90 Pmc4000 - POLON 4800 Fire Protection Station Driver According to Protocol PMC-4000

### Driver Use

The Pmc4000 protocol driver is used for data exchange between the Asix system and POLON 4800 fire protection station according to protocol PMC-4000. The communication is executed with the use of a serial link in RS-232 standard.

The parameterization of the Pmc4000 driver is performed by the Architect module.

### Declaration of Transmission Channel

The declaration of the transmission channel operating according to Pmc4000 protocol requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* Pmc4000

**CtPmc4000** tab:

*Channel parameters:*  
**Port=number; StationNr=number; Baud=number**

where:

*Port* - computer's COM serial port number,  
*StationNo* - station number,  
*Baud* - transmission speed (2400, 4800 or 9600 Bd). By default, 9600 Bd.

#### EXAMPLE

An example of the declaration of the transmission channel communicating with the fire protection station numbered 2 with the use of the COM2 serial port at the default transmission speed (9600 Bd):

*Name:* CHANNEL  
*Driver:* CtPmc4000  
*Channel parameters:* Port=2; StationNo=2

### Variable Declaration

The variable address has the following syntax:

*V.<LineNo>.<ElemNo>.<IONo>.<ZoneNo>[.<baseAlarmNo>]*

where:

<i>V</i>	- variable name,
<i>LineNo</i>	- line number,
<i>ElemNo</i>	- given line element number,
<i>IONo</i>	- input/output number of the given element,
<i>ZoneNo</i>	- zone number,
<i>AlarmNo</i>	- base alarm number.

All variables return values of WORD type.

Line, element, IO and zone numbers are determined by the fire protection station configuration.

The variable value is the bitmap of the fire protection station's element status whose address was given in the variable declaration. Bits have the following meaning:

bit 0	- fire alarm status;
bit 1	- control status;
bit 2	- test status;
bit 3	- blocking status;
bit 4	- technical alarm status;
bit 5	- failure status;
bit 6	- unmaskable failure status;
bits 7 - 15	- are not used.

If the bit value is 1, the given bit is active.

For each variable the user can define a set of alarms assigned to specific variable bitmap bits. Alarms are reported, as soon as the status of any bit is changed. To guarantee the correct driver operation, the aforementioned alarms should occupy a coherent alarm number area, starting from *baseAlarmNo*. Subsequent alarm numbers, starting from *baseAlarmNo*, have the following meanings:

+ 0	- fire alarm;
+ 1	- control performed;
+ 2	- test;
+ 3	- blocking;
+ 4	- technical alarm;
+ 5	- failure;
+ 6	- unmaskable failure;

Asix alarms are reported according to the following rule: **start** (change from 0 to 1) and **end** (change from 1 to 0). The time at which the alarm occurred is marked with a fire protection station timestamp. It is not possible to automatically pass on the alarm text sent from the fire protection station to the Asix alarm. It is recommended that the alarm texts defined in the Asix system should be compliant with the alarm texts defined in the fire protection station.

### Specific supported variables

The variables of the following addresses are specifically supported:

V.0.0.0.0.baseAlarm (line no. 0)

and

V.1.0.0.0.baseAlarm (line no. 1)

The status of **V.0.0.0.0.baseAlarm** contains information about the 2nd level alarm status.

The bits have the following meaning:

bit 0	- 2 <sup>nd</sup> level alarm status;
-------	---------------------------------------

## Communication Drivers

The base alarm number of the variable in question is used for generating the following alarms:

- + 0 - 2<sup>nd</sup> level alarm signaling;
- + 1 - receipt of DELETE information from the station,
- + 2 - receipt of CONFIRMATION information from the station,

The variable can be used to send orders to the fire protection station:

- recording the value 1 to the variable results in sending a DELETE order to the station,
- recording the value 2 to the variable results in sending a CONFIRMATION order to the station,

*V.1.0.0.0.baseAlarm* variable status is used to pass on alarms about station failures.

The base alarm number of the variable is used to generate 107 alarms according to the specification given in the station failure code table of POLON unit ("PMC-4000 - Digital monitoring protocol for POLON 4800 standard"). The first alarm in the specification will be assigned to *baseAlarm* number, the second one - to *baseAlarm + 1*, and so on.

The variable value is accidental.

### EXAMPLES

Examples of variables declaration:

```
# variable used to display the 2nd level alarm status and to send orders  
# DELETE and CONFIRMATION. It generates alarms of the numbers starting from 1  
JJ_01, , V.0.0.0.0.1, PMC, 1, 1, NIC
```

```
# line 1001, element 2, I/O 3, zone 4, alarms starting from number 10  
JJ_02, , V.1001.2.3.4.10, PMC1, 1, 1, NIC
```

```
# line 3002, element 4, I/O 0, zone 0, alarms starting from number 20  
JJ_03, , V.3002.4.0.0.20, PMC1, 1, 1, NIC
```

```
# variable used to pass alarms about station failures:  
JJ_04, , V.1.0.0.0.1000, PMC, 1, 1, NIC
```

Alarm definitions for JJ\_01 variable:

- 1, al, 2nd level fire alarm
- 2, al, information on deleting fire alarm from station
- 3, al, information on confirmation of event or failure

Alarm definitions for JJ\_02 variable:

- 10, al, fire alarm - line 1001 elem 2 io 3 zone 4
- 11, al, control - line 1001 elem 2 io 3 zone 4
- 12, al, test - line 1001 elem 2 io 3 zone 4
- 13, al, blocking - line 1001 elem 2 io 3 zone 4
- 14, al, technical alarm - line 1001 elem 2 io 3 zone 4
- 15, al, failure - line 1001 elem 2 io 3 zone 4
- 16, al, unmaskable failure - line 1001 elem 2 io 3 zone 4

Alarm definitions for JJ\_03 variable:

- 20, al, fire alarm - line 3002 elem 4 io 0 zone 0
- 21, al, control - line 3002 elem 4 io 0 zone 0
- 22, al, test - line 3002 elem 4 io 0 zone 0
- 23, al, blocking - line 3002 elem 4 io 0 zone 0
- 24, al, technical alarm - line 3002 elem 4 io 0 zone 0
- 25, al, failure - line 3002 elem 2 io 0 zone 0
- 26, al, unmaskable failure - line 3002 elem 2 io 0 zone 0

Alarm definitions for JJ\_04 variable:

- 1000, al, uP1 microprocessor system
- 1001, al, uP2 microprocessor system

1005,al, uP1 processor EPROM memory  
 1006,al, configuration memory - uP1 setup

## Driver Parameters

The Pmc4000 driver parameters are declared in the Miscellaneous module, on Directly entered options tab.

**Section name: CtPmc4000**

**Option name: LOG\_FILE**

**Option value: log\_filename**

Meaning - the diagnostic purposes are fulfilled by a text log file, where the driver operation status messages are entered.

Default value - by default, the log file is not created.

**Section name: CtPmc4000**

**Option name: LOG\_FILE\_SIZE**

**Option value: number**

Meaning - option is used to determine the size of the log file defined with the use of LOG\_FILE option. When the size exceeds the amount declared, the driver keeps the current log file contents in the file named log\_filename.old (the name is created from log\_filename given in the LOG\_FILE declaration and extension .old). In this manner, the user is always able to view records of size equal to at least LOG\_FILE\_SIZE MB.

Parameter:

*number* - log file size in MB.

Default value - default log file size is 10 MB.

**Section name: CtPmc4000**

**Option name: LOG\_OF\_TELEGRAMS**

**Option value: YES | NO**

Meaning - option allows the contents of telegrams sent between the driver and controllers to the log file (declared with use of LOG\_FILE option) to be saved. The subject option should be used only during the start-up of the Asix system.

Default value - by default, the option value is set to NO.

### EXAMPLE

Example of driver parameters:

Section name: CtPmc4000

Option name: LOG\_FILE

Option value: d:\tmp\CtPmc4000\pmc.log

Section name: CtPmc4000

Option name: LOG\_FILE\_SIZE

Option value: 20

Section name: CtPmc4000

Option name: LOG OF TELEGRAMS

Option value: YES

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

- Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.
- Option value:  
YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

- Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.
- Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

- Purpose - with this option deployed, local control in a channel will be disabled.
- Option value:  
YES / NO
- The default value - NO.

### **Data Validity Period**

- Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.
- Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.
- The default value - by default, no time for data validity is set.

## 1.91 PPI - Driver of PPI Protocol for SIMATIC S7 200 PLCs

### Driver Use

The communication protocol PPI is used for data exchange between computers with the Asix application and SIEMENS S7 200 PLCs.

Parameterization of PPI driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the PPI driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: PPI

**PPI / Channel parameters** tab:

**Port** - number of the serial port to which the controller is connected;

**Programmable controller address** - address of the PLC.

**Number of repetitions**

Meaning - number of transmission repetitions in case of transmission errors.  
 Default value - 4.  
 Parameter:  
   *number* - number of repetitions.

**Timeout**

Meaning - timeout for the station answer.  
 Default value - 500.  
 Parameter:  
   *number* - time passed in milliseconds.

**Delay**

Meaning - minimal time interval in milliseconds between transmissions of frames.  
 Default value - 25.  
 Parameter:  
   number - time passed in milliseconds.

**All errors**

Meaning - if the parameter has a value of no, the information on timeout errors will appear in 'Control Panel' only when the transmission

missed in spite of attempts of its repetition. If it has a value of yes, the information on all errors is transmitted to 'Control Panel'.  
Default value - no.

**PPI / Channel parameters 2** tab:

**Send frame**

Meaning - maximal length of a sending frame.  
Default value - 117.  
Parameter:  
    *number* - number passed in bytes from the range of 10-260.

**Receive frame**

Meaning - maximal length of a receiving frame.  
Default value - 117.  
Parameter:  
    *number* - number passed in bytes from the range of 10-260.

**Variables**

Meaning - maximal number of variables transferred once.  
Default value - 8.

**Simulation**

Meaning - if yes is given, then data reading/writing from/to a PLC will be simulated.  
Default value - no.

**PC adres**

Meaning - address of a computer.  
Default value - no.  
Parameter:  
    *number* - number passed in bytes from the range of 0-255.

**EXAMPLE 1**

*Channel / Name: S7\_212*  
*Driver: PPI*  
*Port: COM2*  
*Programmable controller address: 5*

In the example above, the station named S7\_212 connected to the COM2 port is defined. The communication with the controller is operating on the basis of default parameters.

**NOTE** *The PPI protocol driver may act together with the AsComm connections manager. In such case, the driver is registered as a client of the AsComm module with a name PPI:n, where n is a number of a serial interface through which the communication with the PLC is executed.*

## Driver Parameters

**PPI / Driver Parameters** tab:

### **Transmission parameters**

**Port**

Meaning - number of serial port (COM1...COM256).

**Speed**

Meaning - option allows to specify the baud rate; the option is not placed in a section of parameters of logical channel.

Default value - 9600.

Parameter:

*number* - number in bauds.

**Parity**

Meaning - option allows to specify the parity type; the option is not placed in a section of parameters of logical channel.

Default value - e.

Parameter:

*parity\_parameter* - one of the following items:

n - no parity,

o - odd,

e - even,

m - mark,

s - space.

**Word length**

Meaning - option allows to specify the length of the word; the option is not placed in a section of parameters of logical channel.

Default value - 8.

Parameter:

*number* - number from the range of 5-8.

**Stop bits**

Meaning - option allows to specify the number of stop bits; the option is not placed in a section of parameters of logical channel.

Default value - 1.

Parameter:

*number* - number of bits.

**AsComm**

Meaning - option allows to specify whether the driver has to work with the communication manager AsComm; the option is not placed in a section of parameters of logical channel.

Default value - no (from the version 1.1).

## Defining the Process Variables

### Measurement Data

The driver executes an access to the following variables.

**Table 48. Variables Serviced by the PPI Driver.**

Symbol	Data Length
Mn.m	BYTE
MBn	BYTE
MWn	WORD
MDn	DWORD
In.m	BYTE
Ibn	BYTE
Iwn	WORD
Idn	DWORD
Qn.m	BYTE
BN	BYTE
PWN	WORD
QDn	DWORD
Vn.m	BYTE
VBn	BYTE
VWn	WORD
VDn	DWORD
Sn.m	BYTE
SBn	BYTE
SWn	WORD
SDn	DWORD
SMn.m	BYTE
SMBn	BYTE
SMWn	WORD
SMDn	DWORD
AIWn	WORD
AQWn	WORD
HCn	DWORD
Cn	WORD
Cn.m	BYTE
Tn	WORD
Tn.m	BYTE
RUN	BYTE

Meaning of symbols placed in the left column (except RUN) is described in the documentation of S7 controllers.

The variable RUN assumes a value of 1 if the controller is in the state RUN, and of 0 otherwise. Writing to the variable RUN causes an activation of the controller. Writing the value of 0 to the variable RUN causes a transition of the controller to the STOP mode. The change of the controller state is possible only at a suitable setting of switches on the controller.

The variables Cn and Tn enable to access to an actual value of counters and timers. The variables Cn.m and Tn enable access to the state (1 or 0) of counters and timers. The value m.may be any number of the range 0 to 7.

The present driver version does not allow to write to the variables Q, AQW, AIW. Writing to other variables is limited by the controller (Cm.n, Tm.n).

#### Access to Pseudo-Variables

The PPI protocol driver enables access to pseudo-variables. The access to pseudo-variables does not cause a physical transmission through a serial interface. Values of pseudo-variables are related with an actual state of a connection with the controller.

**Table 49. Pseudo-Variables Serviced by the PPI Driver.**

<b>Symbol</b>	<b>Meaning</b>	<b>Length</b>
SBS	number of sent bytes	DWORD
SBR	number of received bytes	DWORD
SFS	number of sent frames	DWORD
SFR	number of received frames	DWORD
SPE	number of parity errors	DWORD
SFE	number of frame errors	DWORD
SOE	number of overrun errors	DWORD
SLE	number of line errors (sum of parity, frame, overrun and other errors)	DWORD
STE	number of timeout errors	DWORD
SPRE	number of protocol errors	DWORD
SCE	number of checksum errors	DWORD
SFC	number of unsuccessful connections (by means of AsComm module)	DWORD
SBC	number of broken connections (established by AsComm module)	DWORD
SLGE	number of logical errors (no data in the controller, faulty address etc.).	DWORD
ERR	sum of all errors (SLE, STE, SPRE, SCE, SFC, SBC and SLGE). Writing of any value to ERR variable causes zeroing of variables SBS, SBR, SFS, SFR, SPE, SFE, SOE, SLE, STE, SPRE, SCE, SFC, SBC and SLGE.	DWORD
TSBS	number of sent bytes (from the beginning of driver operation)	DWORD
TSBR	number of received bytes (from the beginning of driver operation)	DWORD
TSFS	number of sent frames (from the beginning of driver operation)	DWORD
TSFR	number of received frames (from the beginning of driver operation)	DWORD
TSPE	number of parity errors (from the beginning of driver operation)	DWORD
TSFE	number of frame errors (from the beginning of driver operation)	DWORD
TSOE	number of overrun errors (from the beginning of driver operation)	DWORD
TSLE	number of line errors (sum of parity, frame, overrun and other errors) (from the beginning of driver operation)	DWORD
TSTE	number of timeout errors (from the beginning of driver operation)	DWORD
TSPRE	number of protocol errors (from the beginning of driver operation)	DWORD
TSCE	number of checksum errors (from the beginning of driver operation)	DWORD

TSFC	number of unsuccessful connections (by means of AsComm module) (from the beginning of driver operation)	DWORD
TSBC	number of broken connections (established by AsComm module) (from the beginning of driver operation)	DWORD
TSLGE	number of logical errors (no data in the controller, faulty address etc.). (from the beginning of driver operation)	DWORD
TERR	sum of errors determined by variables TSLE, TSTE, TSPRE, TSCE, TSFC, TSBC and TSLGE.	DWORD
ONLINE	assume a value of 1 if the last attempt to send any frame ended successfully (i.e. an acknowledge from the controller was received) and 0 otherwise.	BYTE

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.92 Protherm300 - Driver of PROTHERM 300 DIFF PLC Protocol

### Driver Use

The Protherm300 driver is used for data exchange between Asix system computers and Protherm 300 DIFF PLC of Process-Electronic GmbH. The data are transferred over the RS-422 serial link.

Parameterization of Protherm300 driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the Protherm300 driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* Protherm300

**CtProtherm300** tab:

*Channel parameters:*

Port=*port\_number* [;*RecvTimeout=ms\_number*]

where:

*port\_number* - number of the serial link (1 is passed for COM1, 2 for COM2, end so on);  
*ms\_number* - timeout of waiting for the controller response (in milliseconds); it is passed 1000 milliseconds by default.

By default, the following transmission parameters are passed:

- transmission speed 9600 Bd;
- number of bytes in a character;
- parity check - EVEN;
- number of stop bytes - 1.

#### EXAMPLE

An exemplary declaration of the channel using the Protherm300 driver on the serial port COM2 with receiving timeout that equals 2000 ms is as follows:

*Channel / Name:* PLC1  
*Driver:* CTPROTHERM300  
*Channel parameters:* Port=2; RecvTimeout=2000

## Addressing the Process Variables

The syntax of the variable address is as follows:

*V.group.unit.index*

where:

- |              |  |
|--------------|--|
| <i>group</i> | - number of the group to which the controller is assigned (a number form 0 to 9);  |
| <i>unit</i>  | - number assigned to the controller within the group (a number form 0 to 9);   |
| <i>index</i> | - number of the variable in the controller, compatible with the specification ' <i>Variablen-Liste fuer PT300</i> ' (in <i>Protherm300/Protherm300 Kommunikation</i> documentation). |

The raw variables may have one of the following type:

- byte;
- byte table;
- float.

The type of raw variable is defined in the specification '*Variablen-Liste fuer PT300*', in *Protherm300/Protherm300 Kommunikation* documentation with the symbol R411.0, published by Process-Electronic.

### EXAMPLE

Examples of variable declaration.

```
JJ_00, value CO (FLOAT),V.1.2.6,      PLC1, 1, 1, NOTHING_FP
JJ_01, status STAT3 (BAJT),V.1.2.18,  PLC1, 1, 1, NOTHING_BYTE
JJ_02, table of 4 statuses XRELi (BAJT),V.1.2.14,PLC1, 4, 1, NOTHING_BYTE
```

## Time Stamp

The variables read from the Protherm300 system are stamped with a local PC time.

## Driver Configuration

PROTHERM300 driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **CTPROTHERM300** section.

- Section name: CTPROTHERM300**
- Option name: LOG\_FILE**
- Option value: file\_name**

Meaning - the item allows to define a file to which all the diagnostic messages of the driver will be written. If the item LOG\_FILE doesn't define a full path, the log file will be created in the current directory. The log file should be used only during the asix system start-up.

Default value - by default, the log file is not created.

Parameter:

*file\_name* - name of the log file.

**Section name: CTPROTHERM300**

**Option name: LOG\_FILE\_SIZE**

**Option value: number**

Meaning - allows to define the size of the log file in MB.  
Default value - by default, the item assumes that the log file has a size of 1 MB.  
Parameter:  
    *number* - size of the log file in MB.

**Section name: CTPROTHERM300**

**Option name: LOG\_OF\_TELEGRAMS**

**Option value: YES/NO**

Meaning - the item allows writing to the log file (declared with use of the LOG\_FILE item) the contents of telegrams transmitted during the data exchange between the asix system and Protherm 300 DIFF PLC of Process-Electronic GmbH company; writing the telegrams content to the log file should be used only during the asix system start-up  
Default value - NO.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.93 PROTRONICPS - Driver of PROTRONICPS Regulator Protocol

### Driver Use

The PROTRONICPS driver is used for data exchange between PROTRONIC PS regulators of Hartmann & Braun and an Asix system computer. The communication is performed by means of serial interfaces in the RS422 standard.

Parameterization of PROTRONICPS driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the PROTRONICPS driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel

*Driver:* PROTRONICPS

**PROTRONICPS** tab:

*id, port, baud [,character, parity, stop]*

where:

- |             |   |
|-------------|---|
| <i>id</i>   | - device identifier (regulator no. in the network), |
| <i>port</i> | - name of the serial port,                          |
| <i>baud</i> | - transmission speed,                               |

optional parameters:

- |                  |                                  |
|------------------|----------------------------------|
| <i>character</i> | - number of bits in a character, |
| <i>parity</i>    | - parity check type,             |
| <i>stop</i>      | - number of stop bits.           |

Default values:

- 8 bits in a character,
- even parity check (EVEN),
- number of stop bits 1.

The PROTRONICPS driver is loaded as a DLL automatically.

### Addressing the Process Variables

The syntax of symbolic address which is used for variables belonging to the PROTRONICPS driver channel is as follows:

*<type><index>*

where:

<i>type</i>	- variable type; allowable types are:
	STAT - controller status,
	F - status of controller errors,
	WA - analog value,
	BV - binary value,
	Y - controller output;
<i>index</i>	- index within the type (used only for WA, BV and Y):
	WA 0 - 255
	BV 0 - 255
	Y 0 - 10

Variables of STAT and F type may be only read.

Variables of Y type may be only written.

Variables of WA and BV type may be read and written.

Raw values of STAT, F, WA type variables are of WORD type.

Raw values of BV type variables are of BYTE type.

Raw values of Y type variables are of SIGNED SHORT type.

### EXAMPLE

An exemplary declaration of variables:

X1, Controller status,STAT,	CHAN1, 1, 1, NOTHING
X2, Status of errors,F,	CHAN1, 1, 1, NOTHING
X3, Bit BV1,BV1,	CHAN1, 1, 1, NOTHING_BYTE
X4, Analog WA3,WA3,	CHAN1, 1, 1, NOTHING
X5, Controller output no. 0,Y0,	CHAN1, 1, 1, NOTHING
X6, Controller output no. 1,Y1,	CHAN1, 1, 1, NOTHING

## Driver Configuration

PROTRONICPS driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **PROTRONICPS** section.

**Section name: PROTRONICPS**

**Option name: LOG\_FILE**

**Option value: file\_name**

Meaning - the item allows to define a file to which all diagnostic messages of the PROTRONICPS driver and information about contents of telegrams received by the driver are written. If the item does not define the full path, then the log file is created in the current directory. The log file should be used only while the asix start-up.

Default value - by default, the log file is not created.

**Section name: PROTRONICPS**

**Option name: LOG\_OF\_TELEGRAMS**

**Option value: YES|NO**

Meaning - the item allows to write to the log file (declared by LOG\_FILE) the contents of telegrams sent within the communication with the PROTRONIC PS controller. Writing the contents of telegrams to the log file should be used only while the asix start-up.

Default value - by default, telegrams are not written.

**Section name: PROTRONICPS**

**Option name: NUMBER\_OF\_REPETITIONS**

**Option value: YES|NO**

Meaning - the item allows to determine a number of repetitions in case of occurrence of transmission error.

Default value - by default, telegrams are not written.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value

- by default, no time for data validity is set.

## 1.94 S700 - Driver S700 of MAIHAK Analyzer Protocol

### Driver Use

The S700 driver is used for data exchange between Maihak S700 gas analyzers and an Asix system computer by use of the AK protocol. The communication is performed via standard serial ports of an Asix computer and the serial interface no. 1 of the analyzer.

**NOTE** Settings *Serial interface # 1* of the analyzer must have the following values:

- 1/ RTS/CTS protocol - without RTS/CTS protocol,
- 2/ XON/XOFF protocol - without XON/XOFF protocol.

Settings *Communication # 1* of the analyzer must have the following values:

- 1/ AK-ID active - ON (1).

Parameterization of S700 driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the S700 driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* S700

**S700** tab:

*Channel parameters:*

*id, port [, baud ,character, parity, stop]*

where:

- id* - analyzer identifier (number AK-ID),
- port* - port name: COM1, COM2 etc.,

*optional parameters:*

- baud* - transmission speed,
- character* - number of bits in a character,
- parity* - parity check type,
- stop* - number of stop bits.

If optional parameters are not given, then default values are as follows:

- transmission speed of 9600 baud,
- 8 bits in a character,
- no parity check (NONE),
- number of stop bits 1.

**EXAMPLE**

The declaration of the logical channel named CHAN1 operating according to the S700 protocol and exchanging data with the analyzer no. 1 via the COM2 port is as follows:

*Channel / Name:* CHAN1

*Driver:* S700

*Channel parameters:* 1, COM2

The S700 driver is loaded as a DLL automatically.

**Addressing the Process Variables**

The syntax of symbolic address which is used for variables belonging to the S700 driver channel is as follows:

`<type>[<index>][.<element>]`

where:

<i>type</i>	- variable type,
<i>index</i>	- index within the type (for some types of variables),
<i>element</i>	- element within the index (for some types of variables).

Types which use no index (in parentheses a type of raw variable value is given):

CZK	- time of calibration measure (WORD),
CZO	- delay time (WORD),
IDA	- analyzer identifier (up to 40 characters) (BYTE),
MNU	- menu language identifier (BYTE),
NRS	- analyzer serial number (BYTE),
PGB	- measure (introducing) of tested gas to the analyzer (WORD),
SPKK	- state of a calibration tray pump (WORD).

Types which need to give an index (in parentheses a type of raw variable value is given):

DWK	- sensibility drift after calibration (FLOAT),
DZK	- zero-point drift of measure substance after calibration (FLOAT),
NSKK	- rated value of measured substance in calibration tray (FLOAT),
PGKW	- measuring (introducing) of standard calibration gas in the analyzer (WORD),
PGKZ	- measuring (introducing) of zero calibration gas to the analyzer (WORD),
SKT	- compensation of measured substance temperature (WORD),
SPKW	- status of a standard calibration gas pump (WORD),
SPKZ	- status of a zero calibration gas pump (WORD),
SPT	- name of measure substance (BYTE),
SPW	- current values of measured substances (FLOAT),
SPZ	- end value of measuring range of measured substance (FLOAT),
STA	- actual analyzer state (WORD).

Types which need to give an index and an element (in parentheses a type of raw variable value is given):

NSPW	- rated value of measured substance in standard calibration gas (FLOAT),
NSPZ	- rated value of measured substance in zero calibration gas (FLOAT).

Types only for reading :

DWK	- sensibility drift after calibration,
DZK	- zero-point drift of calibration substance after calibration,
MNU	- menu language identifier,
NRS	- analyzer serial number,

## Communication Drivers

NSKK	- rated value of measure substance in calibration tray,
SPKK	- status of a calibration tray pump,
SPT	- name of measured substance,
SPW	- current values of measured substance,
SPZ	- limit range value of measured substance,
STA	- current analyzer status.

Types only for writing:

PGB	- measuring (introducing) of tested gas in the analyzer,
PGKW	- measuring (introducing) of standard calibration gas to the analyzer,
PGKZ	- measuring (introducing) of zero calibration gas to the analyzer.

Types for reading and writing:

CZO	- delay time,
CZK	- time of calibration measurement,
IDA	- identifier of the analyzer,
NSPW	- rated value of the measured substance in zero calibration gas,
NSPZ	- rated value of measured substance in zero calibration gas,
SKT	- compensation of measured substance temperature,
SPKW	- status of a standard calibration gas pump,
SPKZ	- status of a zero calibration gas pump.

### EXAMPLE

Exemplary declarations of variables.

```
# names of types (SPT) and ranges (SPZ) of measured substances - with an index
```

```
X1, SPT1, CHAN1, 10, 1, NOTHING_TEXT
X2, SPZ1, CHAN1, 1, 1, NOTHING_FP
X3, SPT2, CHAN1, 10, 1, NOTHING_TEXT
X4, SPZ2, CHAN1, 1, 1, NOTHING_FP
X5, SPT3, CHAN1, 10, 1, NOTHING_TEXT
X6, SPZ3, CHAN1, 1, 1, NOTHING_FP
X7, SPT4, CHAN1, 10, 1, NOTHING_TEXT
X8, SPZ4, CHAN1, 1, 1, NOTHING_FP
X9, SPT5, CHAN1, 10, 1, NOTHING_TEXT
X10, SPZ5, CHAN1, 1, 1, NOTHING_FP
```

```
# state bytes (STA) of analyzer with an index
```

```
X11, STA1, CHAN1, 1, 1, NOTHING
X12, STA2, CHAN1, 1, 1, NOTHING
X13, STA3, CHAN1, 1, 1, NOTHING
X14, STA4, CHAN1, 1, 1, NOTHING
X15, STA5, CHAN1, 1, 1, NOTHING
X16, STA6, CHAN1, 1, 1, NOTHING
X17, STA7, CHAN1, 1, 1, NOTHING
X18, STA8, CHAN1, 1, 1, NOTHING
```

```
# actual values (SPW) of measured substances - with an index
```

```
X21, SPW1, CHAN1, 1, 1, NOTHING_FP
X22, SPW2, CHAN1, 1, 1, NOTHING_FP
X23, SPW3, CHAN1, 1, 1, NOTHING_FP
X24, SPW4, CHAN1, 1, 1, NOTHING_FP
X25, SPW5, CHAN1, 1, 1, NOTHING_FP
```

```
# time pauses (CZO and CZK) - without an index
```

```
X31, CZO, CHAN1, 1, 1, NOTHING
X32, CZK, CHAN1, 1, 1, NOTHING
```

```
# results after calibration: zero drift (DZK), sensibility drift (DWK) - without an index
```

```
X42, DZK1, CHAN1, 1, 1, NOTHING_FP
X43, DWK1, CHAN1, 1, 1, NOTHING_FP
```

X44, DZK2, CHAN1, 1, 1, NOTHING\_FP  
X45, DWK2, CHAN1, 1, 1, NOTHING\_FP

# status of temperature compensation (SKT) - with an index  
X51, SKT1, CHAN1, 1, 1, NOTHING  
X52, SKT2, CHAN1, 1, 1, NOTHING  
X53, SKT3, CHAN1, 1, 1, NOTHING  
X54, SKT4, CHAN1, 1, 1, NOTHING  
X55, SKT5, CHAN1, 1, 1, NOTHING

# identifier of analyzer (IDA) - without an index  
X61, IDA, CHAN1, 42, 1, NOTHING\_TEXT

#serial number of analyzer (IDA) - without an indexes  
X62, NRS, CHAN1, 20, 1, NOTHING\_TEXT

# menu language of analyzer (IDA) - without an index  
X63, MNU, CHAN1, 1, 1, NOTHING\_BYTE

# state of zero calibration gas pump (1 and 2) (SPKZ) - with an index  
X70, SPKZ1, CHAN1, 1, 1, NOTHING  
X80, SPKZ2, CHAN1, 1, 1, NOTHING

# nominal values of zero calibration gases (1 and 2) (NSPZ) - with an index and element  
X71, NSPZ1.1, CHAN1, 1, 1, NOTHING\_FP  
X72, NSPZ1.2, CHAN1, 1, 1, NOTHING\_FP  
X73, NSPZ1.3, CHAN1, 1, 1, NOTHING\_FP  
X74, NSPZ1.4, CHAN1, 1, 1, NOTHING\_FP  
X75, NSPZ1.5, CHAN1, 1, 1, NOTHING\_FP

X81, NSPZ2.1, CHAN1, 1, 1, NOTHING\_FP  
X82, NSPZ2.2, CHAN1, 1, 1, NOTHING\_FP  
X83, NSPZ2.3, CHAN1, 1, 1, NOTHING\_FP  
X84, NSPZ2.4, CHAN1, 1, 1, NOTHING\_FP  
X85, NSPZ2.5, CHAN1, 1, 1, NOTHING\_FP

# status of standard calibration gas pump (3 - 6) (SPKW) - with an index  
X90, SPKW3, CHAN1, 1, 1, NOTHING  
X100, SPKW4, CHAN1, 1, 1, NOTHING  
X110, SPKW5, CHAN1, 1, 1, NOTHING  
X120, SPKW6, CHAN1, 1, 1, NOTHING

# rated values of standard calibration gases (3 - 6) (NSPW) - with an index and element  
X91, NSPW3.1, CHAN1, 1, 1, NOTHING\_FP  
X92, NSPW3.2, CHAN1, 1, 1, NOTHING\_FP  
X93, NSPW3.3, CHAN1, 1, 1, NOTHING\_FP  
X94, NSPW3.4, CHAN1, 1, 1, NOTHING\_FP  
X95, NSPW3.5, CHAN1, 1, 1, NOTHING\_FP

X101, NSPW4.1, CHAN1, 1, 1, NOTHING\_FP  
X102, NSPW4.2, CHAN1, 1, 1, NOTHING\_FP  
X103, NSPW4.3, CHAN1, 1, 1, NOTHING\_FP  
X104, NSPW4.4, CHAN1, 1, 1, NOTHING\_FP  
X105, NSPW4.5, CHAN1, 1, 1, NOTHING\_FP

X111, NSPW5.1, CHAN1, 1, 1, NOTHING\_FP  
X112, NSPW5.2, CHAN1, 1, 1, NOTHING\_FP  
X113, NSPW5.3, CHAN1, 1, 1, NOTHING\_FP  
X114, NSPW5.4, CHAN1, 1, 1, NOTHING\_FP  
X115, NSPW5.5, CHAN1, 1, 1, NOTHING\_FP

X121, NSPW6.1, CHAN1, 1, 1, NOTHING\_FP  
X122, NSPW6.2, CHAN1, 1, 1, NOTHING\_FP  
X123, NSPW6.3, CHAN1, 1, 1, NOTHING\_FP

## Communication Drivers

```
X124, NSPW6.4, CHAN1, 1, 1, NOTHING_FP  
X125, NSPW6.5, CHAN1, 1, 1, NOTHING_FP
```

```
# settings of calibration tray (SPKK) - pump, (NSKK) - rated
```

```
X130, SPKK, CHAN1, 1, 1, NOTHING  
X131, NSKK1, CHAN1, 1, 1, NOTHING_FP  
X132, NSKK2, CHAN1, 1, 1, NOTHING_FP  
X133, NSKK3, CHAN1, 1, 1, NOTHING_FP  
X134, NSKK4, CHAN1, 1, 1, NOTHING_FP  
X135, NSKK5, CHAN1, 1, 1, NOTHING_FP
```

```
# start of zero calibration gas (1 and 2)
```

```
X201, PGKZ1, CHAN1, 1, 1, NOTHING  
X202, PGKZ2, CHAN1, 1, 1, NOTHING
```

```
# start of standard calibration gas measuring (3 - 6)
```

```
X203, PGKW3, CHAN1, 1, 1, NOTHING  
X204, PGKW4, CHAN1, 1, 1, NOTHING  
X205, PGKW5, CHAN1, 1, 1, NOTHING  
X206, PGKW6, CHAN1, 1, 1, NOTHING
```

```
# start of tested gas measuring
```

```
X207, PGB, CHAN1, 1, 1, NOTHING
```

## Driver Configuration

S700driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **S700** section.

**Section name: S700**

**Option name: LOG\_FILE**

**Option value: file\_name**

Meaning - the item allows to define a file to which all diagnostic messages of the S700 driver and information about contents of telegrams received by the driver are written. If the item does not define the full path, then the log file is created in the current directory. The log file should be used only while the Asix start-up.

Default value - by default, the log file is not created.

**Section name: S700**

**Option name: LOG\_OF\_TELEGRAMS**

**Option value: YES|NO**

Meaning - the item allows to write to the log file (declared by use of LOG\_FILE) the contents of telegrams transferred within the communication with a S700 analyzer. Writing the contents of telegrams to the log file should be used only while the Asix start-up.

Default value - by default, telegrams are not written.

**Section name: S700**

**Option name: LOG\_FILE\_SIZE**

**Option value: number**

Meaning - the item allows to specify the log file size in MB.

Default value - by default, the item assumes that the log file has a size of 1 MB.

**Section name: S700**

**Option name: IGNORE\_STATUS\_CHARACTER**

**Option value: YES|NO**

Meaning - in each answer from S700 an internal status byte of the analyzer is transferred. The content of this byte decides about the status of data which are transferred in a given answer from the analyzer. If the byte has a value of 0, then the variables receive a correct status, otherwise they receive an error status. The use of the item IGNORE\_STATUS\_CHARACTER with a value of YES causes that a correct status is assigned to the variables, irrespectively of the status byte content.

Default value - by default, the item has a value of NO.

**Section name: S700**

**Option name: RECV\_TIMEOUT**

**Option value: station\_no,number**

Meaning - the item RECV\_TIMEOUT allows to specify a waiting time for arriving the first character of an answer sent from a specified analyzer. After passage of this time it is assumed that a given analyzer does not work correctly and the transmission session ends with an error.

Default value - by default, it is assumed that the maximal waiting time for the first character of an answer is equal to 1000 milliseconds.

Parameter:

*station\_no* - number AK-ID of the analyzer,

*number* - time in milliseconds (from 100 to 5000).

**Section name: S700**

**Option name: CHAR\_TIMEOUT**

**Option value: station\_no,number**

Meaning - the item allows to determine a maximal time between successive characters of answer from a given analyzer. After passage of this time it is assumed that a given analyzer does not work correctly and transmission session ends with an error.

Default value - by default, it is assumed that the time between successive characters is equal to 50 millisecond.

Parameter:

*station\_no* - number AK-ID of analyzer,

*number* - time in milliseconds (from 10 to 300).

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

- Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.
- Option value:  
YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

- Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.
- Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

- Purpose - with this option deployed, local control in a channel will be disabled.
- Option value:  
YES / NO
- The default value - NO.

### **Data Validity Period**

- Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.
- Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.
- The default value - by default, no time for data validity is set.

## 1.95 S7\_TCPIP - Driver for Data Exchange with SIMATIC Controllers with Use of Ethernet

### Driver Use

The S7\_TCPIP driver is used for data exchange with SIMATIC S7-series controllers through the Ethernet connection with the use of a standard computer network card.

The S7\_TCPIP protocol driver does not require the installation of SIMATIC NET software from SIEMENS on the Asix system computer, nor the adaptation of the controller program for the needs of data exchange.

The parameterization of the S7\_TCPIP driver is performed by the Architect application.

### Declaration of Transmission Channel

The declaration of the transmission channel operating according to S7\_TCPIP protocol requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the logical transmission channel

*Driver*: S7\_TCPIP

**S7\_TCPIP/ Channel parameters** tab:

*Device (controller) IP address*

*Local TSAP address*

*Remote TSAP address*

*IP address* - the IP address of the local computer network adapter through which the channel will connect with the driver;

*UTC Synchronization* - it is possible to use UTC time for time synchronization.

**S7\_TCPIP/ Channel parameters - controlling** tab:

*Control variable* - name of variable used to control the controller's RUN-STOP status;

*Alarm number* - alarm number generated upon the change of the controller's RUN-STOP status; by default, the alarm is not generated;

*Error signal* - setting the error status for all variables in the given channel; if the controller switches into STOP status; by default the error status is set;

### EXAMPLE

Below is an example of the declaration of the transmission channel utilizing protocol S7-TCPIP:

Name: CHAN1  
Driver: S7\_TCPIP  
Server IP address: 10.10.10.40  
Local TSAP address: 10.02  
Remote TSAP address: 10.03

## Process Variable Addressing

The rules for creating symbolic variable addresses belonging to the transmission channel utilizing S7\_TCPIP protocol are the same as in the case of channel utilizing SAPI S7 protocol.

The symbolic address for variables belonging to the S7\_TCPIP channel has the following syntax:

*variable\_type*[*db\_number*.]*variable\_index*

where:

- variable\_type* - string identifying the variable type in the controller;
- db\_number* - optional data block number; used only in the case of process variables being the mapping of word contents in the data blocks;
- variable\_index* - variable index within the given type; in the case of data blocks, this is the word number in the data block.

The following process variable type markings (based on the variable type names used by SIEMENS) are allowed:

- EA - statuses of outputs, transmitted in bytes,
- EE - statuses of inputs, transmitted in bytes,
- EM - statuses of flags, transmitted in bytes,
- EZ - statuses of meters, transmitted in words,
- ET - statuses of clocks, transmitted in words,
- ED - word values in data blocks,
- EL - double word values in data blocks,
- EG - double word values in data blocks, treated as number in floating point format  
KG,
- EDI - 16-bit words in INTEL convention,
- ER - contents of the data blocks treated as floating point numbers,
- EB - contents of the data blocks treated as bytes.

### EXAMPLES

- ED10.22 - word number 22 in data block no. 10
- EZ100 - meter no. 100

## Driver Parameters

Driver parameters are declared in the *Current data* module on **S7\_TCPIP/Driver parameters Driver parameters - logging** tabs of the channel operating on the basis of S7\_TCPIP driver protocol:

- IP address** - the IP address of the local computer network adapter through which the channel will connect with the driver.
- Statistics**  
Meaning - the option allows information on the number of performed transmission sessions, average transmission time and the

- transmission error number to be displayed every minute; this option was developed to support the designer in the system start-up stage.
- Default value - by default, transmission statistics are not displayed.
- Receive timeout**
- Meaning - maximal waiting time for PLC response.
- Default value - 1000
- Parameter:  
*number* - number in milliseconds
- Log file**
- Meaning - option allows the definition of a file, where all S7\_TCPIP driver messages related to operations executed by the driver will be saved. If the *Log file* option does not define a full path, the log file will be created in the current folder.
- Default value - by default, the log file is not created.
- Parameter:  
*file\_name*
- Log file size**
- Meaning - allows log file size in MB to be determined.
- Default value - by default, option takes the value of 1 MB.
- Parameter:  
*Number* - number in MB
- Log of telegrams**
- Meaning - declaration of writing the contents of telegrams sent and received by the S7\_TCPIP driver within the reading/writing process variables to the log file, declared in *Log file* option.
- Default value - by default, the option is disabled.
- Log of telegram contents**
- Meaning - writing telegram contents to the log file.
- Default value - by default, the option is disabled.

## Channel Parameters

- Log file**
- Meaning - option allows the definition of a file, where all S7\_TCPIP driver messages related to operations executed by the driver will be saved. If the *Log file* option does not define a full path, the log file will be created in the current folder.
- Default value - by default, the log file is not created.
- Parameter:  
*file\_name*

**Log file size**

Meaning - allows log file size in MB to be determined.  
Default value - by default, option takes the value of 1 MB.  
Parameter:  
    *Number* - number in MB

**Log of telegrams**

Meaning - declaration of writing the contents of telegrams sent and received by the S7\_TCPIP driver within the reading/writing process variables to the log file, declared in *Log file* option.  
Default value - by default, the option is disabled.

**Log of telegram contents**

Meaning - writing telegram contents to the log file.  
Default value - by default, the option is disabled.

**Control Variable**

Meaning - name of the variable used for controlling RUN-STOP state of the PLC.  
Default value - no variable.  
Parameter:  
    *Text*

**Alarm Number**

Meaning - number of the alarm generated when RUN-STOP state of the PLC changes.  
Default value - 1.  
Parameter:  
    *Number*

**Error Signal**

Meaning - setting an error status for all variables in a given channel in case of the PLC switching over into the STOP state.  
Default value - not available.  
Parameter:  
    YES/NO

**Enable redundancy H for controllers S7-400**

How to configure:

- 1/ establish S7\_tcpip type channel for connection Asix <--> Rack\_0 of controller S7-400 (channel to define application variables in),
- 2/ establish S7\_tcpip type channel for connection Asix <--> Rack\_1 of controller S7-400 (solely for implementation of redundancy H),
- 3/ set options (in channel parameters):

**Enable redundancy H for this channel**

**Location of Redundancy Status** - byte number in the controller's marker (flag) area, in which the controller's programme will place the current status of redundancy H based on values, the so-called "status of an H\_System". The driver assumes the following meanings of bits in the current status of redundancy H:

bit_0	H_R0_MSTR	// 1 - CPU in rack 0 is master
bit_1	H_R1_MSTR	// 1 - CPU in rack 1 is master
bit_2	H_R0_RUN	// 1 - CPU in rack 0 is running
bit_3	H_R1_RUN	// 1 - CPU in rack 1 is running
bit_4	H_INT_ERR	// 1 - internal error

The driver assumes that in byte no\_MB+1 of the markers area the controller's programme will place the current value of the clock-byte, which will be used by the driver to detect the STOP state of the controller.

Note: The process variables of the redundancy controller S7-400 should be defined solely in the channel handling the Asix <--> Rack\_0 connection.

Change of operation: if requests are repeated, the driver ignores responses bearing the identifier of the previous request (in the previous version, the driver would indicate an error and close the connection in such cases).

 **Enable redundancy H for this channel**

Meaning - option allows the redundancy H for the channel (only for S7-400 series controllers).

Default value - no redundancy H.

Parameters:

*Location of redundancy status* - number of the word in the area of controller markers (flags), in which the controller program places:  
 - redundancy status in the lower byte,  
 - current value ('clock byte') in the high byte.

Addresses:

*Device (controller) IP address*

*Local TSAP address*

*Remote TSAP address*

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

 **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

**Redundancy**

Purpose - with this option searches a redundant channel when connection with a controller is broken in local channel.

Option value:  
*computer name* - list including the names of network computers which will be permitted to search a redundant channel.

The default value - by default, when option is enabled, the channel is searched on all computers.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.96 SAPIS7 - Driver of SAPIS7 Protocol for SIMATIC S7 PLCs

### Driver Use

The SAPIS7 driver is used for data exchange with SIMATIC S7 PLCs by means of the MPI interface or a PROFIBUS bus communication processor. In an Asix system computer the SIEMENS CP5611, CP5412(A2) or CP5613 card is used. The data exchange with use of the SAPIS7 protocol is based on so called S7 functions.

Operation of the Asix system with the SIMATIC S7 PLC by use of the SAPIS7 protocol does not require any controller's program adaptation for data exchange.

Parameterization of SAPIS7 driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the SAPIS7 driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: SAPIS7

**SAPIS7/Channel parameters** tab:

<i>Application name</i>	- virtual device name (VFD),
<i>Connection name</i>	- connection name,
<i>Control variable</i>	- name of a variable used to control the RUN-STOP state of the PLC,
<i>Alarm number</i>	- number of the alarm generated when the RUN-STOP state of the PLC changes; by default, alarms are not generated;
<i>Error signal</i>	- setting an error status for all variables of a given channel in case of the PLC switching over into the STOP state; by default, an error status is set;
<i>UTC Synchronization</i>	- use for time synchronization in the UTC format.

Names *VFD* and *connection name* must be compatible to parameters declared by use of the configuration program COML S7 supplied with the communication processor board.

#### EXAMPLE

An exemplary item declaring use of transmission channel operating according to the SAPIS7 protocol is given below:

*Channel / name*: CHAN1  
*Driver*: SAPIS7  
*Channel parameters*: VFD1, S7\_connection1

The logical channel named CHAN1 has defined the following parameters:

- SAPIS7 protocol,

- virtual device name (VFD) - VFD1,
- connection name - S7\_connection1.

Rules of creating the symbolic addresses of variables belonging to the transmission channel using the SAPI S7 protocol are the same ones as in case of a channel using the AS512 protocol.

The set of process variable types used in the SAPI S7 protocol was extended with the following elements in relation to the set offered by the AS512 protocol:

EDI - 16-byte words in INTEL convention,

ER - content of data blocks, treated as floating-point numbers,

EB - content of data blocks, treated as bytes.

The SAPI S7 driver is loaded as a DLL automatically.

## Driver Configuration

The driver configuration is defined in the *Current Data* module, in the channel operating according to SAPI S7 driver.

**SAPI S7/Driver parameters** tab:

**Log file**

Meaning - the item allows to define a file to which all SAPI S7 driver messages concerning operations performed by the driver are written. If the item does not define the full name, then the log file is created in the current directory.

Default value - by default, the log file is not created.

**Log of telegrams**

Meaning - declaration of writing the contents of telegrams sent and received by the SAPI S7 driver within reading/writing process variables to the log file declared in the item Log file.

Default value - NO.

**Maximal length of the buffer**

Meaning - the length of telegrams accepted by an MPI interface depends on the CPU type of S7 controllers and on a program executed by them.

Default value - the default, length of a telegram is equal to 220 bytes.

Parameter:

*number* - length of telegrams passed in bytes.

**Console**

Meaning - the item allows to create a console window where SAPI S7 driver messages concerning operations executed by the driver are displayed currently.

Default value - by default, any console window is not created.

**SAPIS7/Driver parameters 2** tab: **Statistics**

- Meaning - the item allows to display (every 1 minute) information about number of transmission sessions that have been carried-out, average transmission time and number of transmission errors. The item was developed as a designer support on the stage of the Asix system start-up.
- Default value - by default, the transmission statistics is not displayed.

 **Number of check readings**

- Meaning - the item specifying a minimal number of successive readings of control variable (of a constant value) which cause a signalization of the controller STOP status.
- Default value - by default, the item assumes a value of 3.

 **Serialization**

- Meaning - declaration of servicing the transmission from the S7 by transferring single YES or many NO queries.
- Default value - YES.

 **Without multiple read**

- Meaning - it allows to block the mode of multiple read creation and enables driver operation in the mode of single read.
- Default value - No, in the current version the `s7_multiple_read_req` function is used, by default (it allows to build a multiple read), for maximal use of the length of the telegram buffer sent between PC and S7.
- Parameter:  
*number* - length of telegrams passed in bytes.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

 **Remote Write Access**

- Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.
- Option value:  
YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

**Redundancy**

Purpose - with this option searches a redundant channel when connection with a controller is broken in local channel.

Option value:  
*computer name* - list including the names of network computers which will be permitted to search a redundant channel.

The default value - by default, when option is enabled, the channel is searched on all computers.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.97 S-BUS - Driver of S-BUS Protocol for SAIA-Burgess Electronics PLCs

### Driver Use

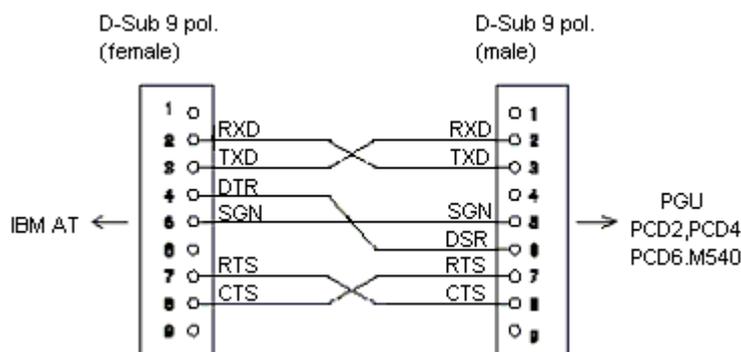
The S-BUS driver is used for data exchange between PCD PLCs of SAIA-Burgess Electronics and an Asix system computer.

The S-BUS protocol is compatible to the specification "SAIA S-Bus for the PCD family", Edition 26/739 E2-05.96, developed by SAIA-Burgess Electronics.

For purposes of communication with the Asix system, the following interfaces of PCD PLCs may be used:

- the PGU interface (RS-232C);
- additional communication interfaces, the number and type of which depend on the controller type and configuration. These interfaces enable data transmission in one of the standards given below:  
RS-232C,  
RS-422,  
RS-485,  
current loop of 20 mA.

**NOTE** For data exchange you should use a cable made according to the specification PCD8.K111.



**Figure. Cable Compatible with the PCD8.K111 Specification.**

Parameterization of S-BUS driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the S-BUS driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: S-BUS

**S-BUS/Channel Parameters** tab:

**Device Number in S-BUS Network** - number of the controller in the S-BUS network;  
**Port** - port name: COM1, COM2 etc.;

*optional parameters*:

**Transmission Speed** - transmission speed in bauds.

If the optional parameters are not given, then by default it is assumed as follows:  
transmission speed of 9600 baud.

**EXAMPLE**

The declaration of the logical channel named CHAN1 operating according to the S-BUS protocol and exchanging data with the controller numbered 1 through the COM2 port with a speed of 9600 baud is as follows:

*Channel / Name*: CHAN1  
*Driver*: S-BUS  
*Channel parameters*: 1, COM2

The S-BUS driver is loaded as a DLL automatically.

## Addressing the Process Variables

The syntax of symbolic address which is used for variables belonging to the S-BUS driver channel is as follows:

*<type><index>*

where:

*type* - variable type,  
*index* - index within the type.

Symbols of variable types (the type of raw variable value is given in parentheses):

**C** - values of counters (DWORD),  
**F** - states of flags (WORD),  
**I** - values of inputs (WORD),  
**K** - current date and time in the form of an 8-bit array (BYTE),  
**O** - values of outputs (WORD),  
**RI** - values of registers treated as 32-bit fixed-point signed numbers (LONG),  
**RF** - values of registers treated as 32-bit floating-point numbers in SAIA format (FLOAT),  
**S** - status (WORD),  
**T** - values of timers (DWORD).

Values of variables of **C**, **F**, **O**, **RI**, **RF**, **T** that may be read and written.

Values of variables of **I**, **S** type that may be only read.

Values of variables of **K** types are used by the driver for time synchronization with a PCD.

The structure of **K** type variable buffer is as follows:

byte 0	- number of a week in a year,
byte 1	- number of a week day (Monday - 1, Sunday - 7),
byte 2	- two least significant digits of a year,
byte 3	- month,
byte 4	- day,
byte 5	- hour,
byte 6	- minute,
byte 7	- second.

Range of indexes for the **S** type is limited to 20 - 27.

#### Variable defining the state of connection with the controller

The variable is defined by address ON and takes value 1 when the last transmission is ended properly as well as when the last transmission is ended with failure. The variable is of WORD type and requires the NOTHING conversion function to be used.

#### **EXAMPLE**

Examples of variable declaration:

```
# values of registers treated as FLOAT
JJ_10, RF1, CHAN1, 1, 1, NOTHING_FP

# values of registers treated as LONG
JJ_11, RI11, CHAN1, 1, 1, NOTHING_LONG

# states of flags
JJ_14, F14, CHAN1, 1, 1, NOTHING

# values of inputs
JJ_14, I14, CHAN1, 1, 1, NOTHING

# values of outputs
JJ_14, O14, CHAN1, 1, 1, NOTHING

# values of counters
JJ_21, C21, CHAN1, 1, 1, NOTHING_DW

# value of status
JJ_40, S20, CHAN1, 1, 1, NOTHING
```

## Driver Configuration

S-BUS driver parameters are declared on the **Driver Parameters** tab.

**Option name: Alarm in Case of Connection Loss and Resume**

**Option value: <device\_number>, <alarm\_number>**

Meaning

- driver of the S-BUS protocol may generate an alarm in case of loss and re-establishing the connection with the station. It is necessary in this case to create the item ALARM in the INI file.

## Communication Drivers

Default value - by default, the alarm is not generated.

Parameters:

- device\_number* - number of the controller in the S-BUS network,
- alarm\_number* - number of the alarm to be generated in case of loss and re-establishing the connection.

**Option name: Number of Repetitions**

**Option value: number**

Meaning - the item allows to specify a number of repetitions in case of a transmission error.

Default value - by default, the item assumes a values of 0 (no repetitions).

**Option name: Mode**

**Option value: <device\_number>, <mode\_name>**

Meaning - up to now, the S-BUS protocol driver has serviced the PARITY transmission mode; from the version 1.02.000 the driver services also BREAK and DATA modes; settings of a proper mode are realized by the MODE parameter.

Default value - omission of the item causes the PARITY mode realization.

Parameters:

- device\_number* - number of controllers in the S-BUS network,
- mode\_name* - one of the following words: PARITY, BREAK or DATA.

**NOTICE** *Using a specific mode, one should remember about proper controller parametrization - the controller should use the same mode.*

**Option name: RTS mode**

**Option value: number**

Meaning - the option sets RTS control mode; useful for RS232/RS485 converters.

Default value - 1

Parameter:

- number* - one of the following numbers that indicate RTS control mode according to API Win32:
  - 0 - RTS\_CONTROL\_DISABLE
  - 1 - RTS\_CONTROL\_ENABLE
  - 2 - RTS\_CONTROL\_HANDSHAKE
  - 3 - RTS\_CONTROL\_TOGGLE

**Option name: Log File**

**Option value: file\_name**

Meaning - the item allows to define a file to which all diagnostic messages of the S-BUS driver and the information about contents of telegrams received by the driver are written. If the item does not define the full path, then the log file is created in the current directory. The log file should be used only while the Asix start-up.

Default value - by default, the log file is not created.

.  
.

**Option name: Log file size**

**Option value: number**

Meaning - the item allows to specify the log file size in MB.

Default value - by default, the item assumes that the log file has a size of 1 MB.

**Option name: Log of Telegrams**

**Option value: YES|NO**

Meaning - the item allows to write to the log file (declared by use of the item LOG\_FILE) the contents of telegrams sent within the communication with the controller. Writing the contents of telegrams to the log file should be used only while the Asix start-up.

Default value - by default, the telegrams are not written.

**Option name: Response timeout**

**Option value: <device\_number>, <timeout in ms>**

Meaning - the item allows to specify a maximal waiting time for arriving the first character of an answer from a given controller. After this time it is assumed that a given controller is switched off and the transmission session ends with an error.

Default value - by default, it is assumed that the maximal waiting time for the first character of an answer is equal to 1000 milliseconds.

Parameters:

*device\_number* - number of the controller in the S-BUS network,  
*timeout in ms* - time in milliseconds (from 100 to 5000).

**Option name: Char Timeout**

**Option value: <device\_number>, <timeout in ms>**

Meaning - the item allows to specify a maximal time between successive characters of an answer from a given controller. After this time it is assumed that the controller does not work correctly and the transmission session ends by an error.

Default value - by default, it is assumed that the maximal time between successive characters of an answer is equal to 50 milliseconds.

Parameters:

*device\_number* - number of the controller in the S-BUS network,  
*timeout in ms* - time in milliseconds (from 10 to 300).

**Option name: Address Timeout**

**Option value: number**

Meaning - the item allows to determine a time period between the character of address and the first character of data in an order sent to the PCD. The time period is necessary to switch over the PGU interface from the mode of address receiving to the mode of data receiving.

Default value - by default, it is assumed that the time period between the address character and the first character of data is equal to 25 milliseconds.

Parameters:

*number* - time in milliseconds.

### Time Synchronization Between the Asix System and SAIA Controllers

In the S-BUS driver there is a mechanism of time synchronization between the Asix system and SAIA controllers. The time synchronization is activated for each channel separately by means of items placed in the ASMEN section.

- ☑ **Section name:** ASMEN
- ☑ **Option name:** TIME\_SYNCHRONIZATION
- ☑ **Option value:** *channel, variable*

Parameters:

- channel* - name of a transmission channel used for communication with a given SAIA controller;
- variable* - name of an ASMEN variable belonging to the channel CHANNEL and used for time synchronization.

The time synchronization consists on cyclic writing to the controller a frame containing an actual Asix time. The frame is written by means of a built-in function for writing the S\_BUS protocol time according to a frequency assigned to variable. The variable type must be the **K** type (clock support), the number of elements assigned to variable must accommodate the time frame, i.e. it must have a size of min. 8 bytes. As a conversion function the NOTHING\_BYTE function must be used.

## Connection by Means of Modem

Driver of the S-BUS protocol is also able to exchange the data by means of a modem, also with use of the PGU interface.

Connection by means of a modem is possible only by using the DATA transmission mode.

S-BUS driver channel *n* is the client of the AsComm server named S:BUS:*n*

where:

- n* - it is the number of the serial port received from the ASMEN channel definition,

e.g.

if channel\_name=S=BUS,1,com2,...

then a client name is S-BUS:2.

To establish a connection on dial-up links by means of the AsComm program, use the option **AsComm Server Client** on the *Driver Parameters - AsComm* tab.

If the modem is connected to the other port than COM*n*, then you should give the number of this port by means of the parameter Port or specify the modem name by means of the parameter Modem. You should also give a telephone number and define other required parameters. If MODBUS driver has to communicate with many controllers by means of the same modem, then one should define suitable number of channels taking the parameter port as a virtual transmission channel and place suitable number of sections in the application configuration file, by specifying in them an appropriate telephone number.

### EXAMPLE

An example of the initialization file content:

*Channel / Name:* Chan1

*Driver:* S-BUS

*Channel parameters:* 1,COM11,9600,8,none,1,16,16

*Channel / Name:* Chan2

*Driver:* S-BUS

*Channel parameters:* 1,COM12,9600,8,none,1,16,16

*Section name:* MODBUS:11  
*Option name:* Switched\_line  
*Option value:* Yes

*Section name:* MODBUS:11  
*Option name:* Modem  
*Option value:* US Robotics

*Section name:* MODBUS:11  
*Option name:* Number

*Option value:* 11111111

*Section name:* MODBUS:12  
*Option name:* Switched\_line  
*Option value:* Yes

*Section name:* MODBUS:12  
*Option name:* Modem  
*Option value:* US Robotics

*Section name:* MODBUS:12  
*Option name:* Number  
*Option value:* 22222222

In the example above Chan1 will communicate with a controller placed under the telephone number 11111111, and the Chan2 with a controller placed under the telephone number 22222222. The US Robotics modem will be used. The *Modem* parameter may be replaced by the parameter *Port*, which specifies the number of the serial port to which the modem is connected.

You should notice that the description of application of the MODBUS driver on switched links does not include any modem configuration guidelines. The modem configuration depends on modem types.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

**Purpose** - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

**Option value:**

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

**The default value**

- NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.98 SbusTcpiip - Driver of S-Bus Ethernet Protocol

### Driver Use

The SbusTcpiip driver is used for data exchange between Asix system computers and PLCs of PCD SAIA-Burgess family by means of the Ethernet S-Bus protocol.

Parameterization of SbusTcpiip driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the SbusTcpiip driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: SbusTcpiip

**CtSbusTcpiip/Channel parameters** tab:

*Device ID in the S-BUS network* - number of the controller in the S-BUS network;  
*Serial port* - number of the TCPIP port of the controller (by default 5050),  
*Device IP address* - IP address of the controller,  
*Synchronization interval* - period (in seconds) for time synchronization with the controller - optional;  
*Timeout* - timeout (in milliseconds) between sending query and receiving response - optional.

#### EXAMPLE

An exemplary declaration of the channel for communication with the controller:

- a. number in the S-BUS network - 3;
- b. number of a TCPIP port - 5050;
- c. IP address - 10.10.10.225;
- d. time synchronization - every 20 seconds;

*Channel*: CHANNEL  
*Driver*: CtSbusTcpiip  
*Device ID in the S-BUS network*: 3  
*Serial port*: 5050  
*Device IP address*: 10.10.10.225  
*Synchronization interval*: 20

## Declaration of Variables

The declaration of variables is the same as in the S-BUS driver. The syntax of the variable address is as follows:

`<type><index>`

where:

*type* - variable type,  
*index* - indexes within the framework of the type.

The notations of variable types (the raw variable type is put in parentheses):

**C** - counter values (DWORD),  
**F** - flag states (WORD),  
**I** - input states (WORD),  
**K** - **current date & time in the form of 8-byte table (BYTE)**,  
**O** - output states (WORD),  
**RI** - values of registers treated as a 32-bit signed number (LONG),  
**RF** - values of registers treated as a 32-bit floating-point number in SAIA format (FLOAT),  
**S** - statuses (WORD),  
**T** - **timer values (DWORD)**.

The variable values of the **C**, **F**, **O**, **RI**, **RF**, **T** type may be read and written.

The variable values of the **I**, **S** type may be only read.

The range of the indexes for the **S** type is from 20 to 27.

### EXAMPLE

Examples of variable declarations.

```
# values of registers treated as FLOAT
JJ_10, , RF1, CHANNEL1, 1, 1, NOTHING_FP
# values of registers treated as LONG
JJ_11, , RI11, CHANNEL1, 1, 1, NOTHING_LONG
# flag states
JJ_14, , F14, CHANNEL1, 1, 1, NOTHING
# input states
JJ_14, , I14, CHANNEL1, 1, 1, NOTHING
# output states
JJ_14, , O14, CHANNEL1, 1, 1, NOTHING
# counter values
JJ_21, , C21, CHANNEL1, 1, 1, NOTHING_DW
# statuses values
JJ_40, , S20, CHANNEL1, 1, 1, NOTHING
```

## Driver Configuration

The driver configuration is defined in the *Current Data* module, in the channel operating according to SbusTcpiip driver.

**CtSbusTcpiip/Driver parameters** tab:

**Log file**

Meaning - it is a text file to which messages about the driver operation state are written; is used for diagnostic purposes.

Default value - by default, the log file is not created.

Defining - manual.

**Log file size**

Meaning - allows to define the size of the log file.  
 Default value - by default, the item assumes that the log file has a size of 1 MB.  
 Parameter:  
     *number* - size of the log file in MB.  
 Defining - manual.

**Telegrams log**

Meaning - the item allows writing to the log file (declared with use of the LOG\_FILE item) the contents of telegrams transmitted during the data exchange between the Asix system and the controllers.  
 Default value - NO.  
 Defining - manual.

**PCD status**

Meaning - the item allows to control the variable status modification depending on the current controller status (PCD own status). If the item has the value YES, then the variable status is not dependent on the current variable of the controller status. If the item is set at value NO, then the variable status is dependent on the controller status - if it differs from 0x52 (RUN state), then the variable status is set at OPC\_QUALITY\_ COMM\_FAILURE.  
 Default value - NO.  
 Defining - manual.

## Channel Parameters

The driver configuration is defined in the *Current Data* module, in the channel operating according to SbusTcpiip driver.

**SbusTcpiip/Channel Parameters** tab:

**Log file**

Meaning - it is a text file to which messages about the driver operation state in the channel are written; is used for diagnostic purposes.  
 Default value - by default, the log file is not created.

**Log file size**

Meaning - allows to define the size of the log file.  
 Default value - by default, the item assumes that the log file has a size of 1 MB.  
 Parameter:  
     *number* - size of the log file in MB.

**Telegrams log**

Meaning - the item allows writing to the log file (declared with use of the *Log File* item) the contents of telegrams transmitted during the data exchange between the Asix system and the controllers.

## Communication Drivers

Default value - NO.

### **IP Address of the Computer**

Meaning - IP address of the computer which is used to connect to the device.

Default value - no address.

### **SbusTcpip/Channel Parameters - redundancy tab:**

#### **PCD3 Redundancy**

Meaning - option allows you to enable redundancy PCD3.M6860 for the channel. The use of redundancy is only possible for the Series M6860.

Default value - PCD3 redundancy is not used.

Parameters:

*Number of Redundancy Status*  
*Device ID in the S-BUS Network*  
*Device TCP Port Number*  
*Device IP Address*

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

#### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

#### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.  
The default value - by default, no time for data validity is set.

## 1.99 Si400 - Driver of Protocol for Sintony Si 400 Alarm Central of SIEMENS

### Driver Use

The Si400 driver is used for data exchange between Asix computers and a Sintony Si 400 alarm central of SIEMENS. The communication is executed by means of serial links in the RS-232 standard.

Additional recommendations:

For proper operation of the alarm system it is demanded to connect only one visualization computer to a Sintony central (despite the central is equipped with 3 interfaces, during the system operation it is possible to use only one of them). The best solution is to connect a computer to the J7 interface with use of a SAQ11 cable recommended by SIEMENS.

Parameterization of Si400 driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the Si400 driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel

*Driver:* Si400

**Si400** tab:

*Channel parameters:*

Port=*port\_nr* [;BaudRate=*baud*] [;Timeout=*ms\_number*]  
[;AlarmOffset=*offset*]USERID=*user\_id*

where:

port	- serial port number (for COM1 it passed 1, for COM2 is passed 2, etc.);
BaudRate	- transmission speed passed in bauds. By default, it is assumed 9600 Bd;
Timeout	- timeout of waiting for the controller response (in milliseconds); it is passed 1000 milliseconds by default;
AlarmOffset	- offset added to the number of each alarm sent from a SINTONY SI 400 central;
USERID	- user identifier sent to SINTONY SI 400 in control frames. Proper values include in the range 0-500.

By default, it is assumed:

- number of character bits - 8,
- without parity check (NONE),
- number of stop bits - 1.

**EXAMPLE**

An exemplary declaration of the channel using CtSi400 on the serial port COM2 with the receiving timeout that equals 2000 ms and the offset, added to each alarm number, equaling 1500 is as follows:

*Channel name:* PLC1

*Driver:* CTSI400

*Channel parameters:* Port=2; Timeout=2000; OffsetAlarm=1500

## Addressing Process Variables

The syntax of the symbolic address for all variables used to monitor the central state is given below:

*<type><index>*

and for variables used to control:

*<type>*

or

*<type><index>*

or

*<type><p\_index>.<r\_index>*

where:

<i>type</i>	- variable type,
<i>index</i>	- index within the type,
<i>p_index</i>	- partition number (0-15),
<i>r_index</i>	- room's number within the partition (0-7).

## Types of Variables Used to Monitor

There are several types of variables used to monitor, which are listed below. The driver permits readout operations to be performed on these variables, while all write operations end with the OPC\_E\_BADRIGHTS error.

The bit 0 means the least significant bit in the following list.

- **IS<index>** - input status

PState	- bits 7-6
Lstate	- bits 5-3
Reserved	- bits 2-0

- **IP<index>** - input parameters:

PSL	- bits 15-14
V.Address	- bits 13-8
Type	- bits 7-4,

## Communication Drivers

SubType, - bits 3-1

- **IA<index>** - input address

room nr - bits 12-9

partition nr - bits 4-0

- **IN<index>** - input name
- **OS<index>** - output status

State - bit 7

OutInTest - bit 6

BlinkBit - bit 5

NbrPulse - bit 4

Reserved - bits 3-0

- **OP<index>** - output parameters

Ltype - bits 7-0

- **OA<index>** - output address

room nr - bits 12-9

partition nr - bits 4-0

- **ON<index>** - output name
- **RP<index>** - room parameters

Rset - bits 15-8 (room1 is in the bit no. 15, room2 is in the bit no. 14, etc.)

Rarm - bits 7-0 (room1 is in the bit no. 7, room2 is in the bit no. 6, etc.)

- **RS<index>** - room status

room1 - bits 15-14

room2 - bits 13-12

etc.

- **PS<index>** - partition status

TAMP - bit 15

PA - bit 14

Fire - bit 13

AM - bit 12

Chime - bit 11

Mess - bit 10

Reserved - bits 9-8

Pstate - bits 7-6

PFA - bit 5

PPA - bit 4

PFS - bit 3

PPS - bit 2

PUS - bit 1

BA - bit 0

- **SS<index>** - system status

ParamCh - bit 15

Evt Overflow - bit 14

Avzone - bits 13 - 8

Mains - bit 7

Batt - bit 6  
 Fuse - bit 5  
 Line - bit 4  
 CMS1 - bit 3  
 CMS2 - bit 2  
 InComm - bit 1  
 Evt queue - bit 0

## Types of Variables Used to Control

There are several types of variables used to control, which are listed below. The driver permits write operations to be performed on these variables, while all readout operations end with the OPC\_E\_BADRIGHTS error.

- **cIB<index>** - input bypass (index = 0..0xFFFF)

Writing any value to the variable will cause the performance of the command for the input with the *index* number.

- **cIS<index>** - input in soak test (index = 0..0xFFFF)

Writing any value to the variable will cause the performance of the command for the input with the *index* number.

- **cOT<index>** - set output in test mode (index = 0..0xFFFF)

Writing any value to the variable will cause the performance of the command for the output with the *index* number.

- **cTO<index>** - toggle output (index = 0..0xFFFF)

Writing any value to the variable will cause the performance of the command for the output with the *index* number.

- **cPP<p\_index>** - partition part-set (p\_index = 0..0xF)

Writing any value to the variable will cause the performance of the command for the partition with the *p\_index* number.

- **cPF<p\_index>** - partition full-set (p\_index = 0..0xF)

Writing any value to the variable will cause the performance of the command for the partition with the *p\_index* number.

- **cPU<p\_index>** - partition unset (p\_index = 0..0xF)

Writing any value to the variable will cause the performance of the command for the partition with the *p\_index* number.

- **cCA<p\_index>** - clear alarm memory in partition (p\_index = 0..0xF)

Writing any value to the variable will cause the performance of the command for the partition with the *p\_index* number.

- **cRF<p\_index>.<r\_index>** - room full-set (p\_index = 0..0xF, r\_index = 0..0xF)

Writing any value to the variable will cause the performance of the command for the room with the *r\_index* number from the partition with the *p\_index* number.

- **cRU<p\_index>.<r\_index>** - room unset (p\_index = 0..0xF, r\_index = 0..0xF)

Writing any value to the variable will cause the performance of the command for the room with the *r\_index* number from the partition with the *p\_index* number.

- **cSR<p\_index>** - partition reset sounders/bells (p\_index = 0..0xF)

Writing any value to the variable will cause the performance of the command for the partition with the *p\_index* number.

- **cIP<index>** - input activation pulse (index = 0..0xFFFF)

Writing any value to the variable will cause the performance of the command for the input with the *index* number.

- **cIL<index>** - input activation latch (index = 0..0xFFFF)

A written value must convert to a double type. If the value 0 is received after conversion, for the input with the *index* number there will be performed the command with the parameter *commutation level* equaling 0; otherwise the parameter will be equal to 1.

- **cAV** - new current A/Video zone

A written value must be an integer from the range <0; 32>. Writing the value 0 will activate the *autoswitch* option.

- **cTC** - transmission to CMS

A written value must have the value 1 or 2 and indicate CMS1 or CMS2 suitably.

- **cER** - reset the pending event buffer

Writing any value to the variable will cause the performance of the command.

IN and ON variables are 16-element BYTE type arrays. Other variables are WORD type values.

All the variables used to control are of the WORD type.

### EXAMPLE

Exemplary variable declarations:

JJ_00, input status 1,	IS1,	PLC1, 1, 1, NIC
JJ_01, output status 10,	OS10,	PLC1, 1, 1, NIC
JJ_02, input 2 name,	IN2,	PLC1, 16, 1, NIC_BYTE
JJ_03, system status,	SS1,	PLC1, 1, 1, NIC
cIB3, input 3 bypass,	cIB3,	PLC1, 1, 1, NIC
cRU1.2, part 1 room 2 unset,	cRU1.2,	PLC1, 1, 1, NIC
cAV, current A/V zone,	cAV,	PLC1, 1, 1, NIC

## Time Stamp

All values of variables read from a SINTONY SI 400 central are stamped with the local PC time.

## Driver Configuration

Si400 driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **CTSI 400** section.

**Section name: CTSI 400**

**Option name: LOG\_FILE**

**Option value: file\_name**

Meaning - the item allows to define a file to which all diagnostic messages of the driver will be written. If the item does not define the full path, then the log file is created in the current directory. The log file should be used only while the Asix system start-up.

Default value - by default, the log file is not created.

Defining - manual.

**Section name: CTSI 400**

**Option name: LOG\_FILE\_SIZE**

**Option value: number**

Meaning - allows to specify the log file size.

Default value - 1MB.

Defining - manual.

Parameter:

*number* - number in MB.

**Section name: CTSI 400**

**Option name: LOG\_OF\_TELEGRAMS**

**Option value: YES|NO**

Meaning - item allows to write to the log file (declared by use of the item LOG\_FILE) the contents of telegrams received by the driver. Writing the contents of telegrams to the log file should be used only in the stage of the Asix system start-up.

Default value - NO.

Defining - manual.

**Section name: CTSI 400**

**Option name: DELAY\_AFTER\_OPEN\_SESSION**

**Option value: number**

Meaning - allows to define a timeout between opening a session and sending the first query about data within this session.

Default value - by default, the value is set to 100.

Defining - manual.

Parameter:

*Number* - timeout value in milliseconds.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value

- NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value

- by default, no time for data validity is set.

## 1.100 SINECH1 - Driver of Ethernet Network Protocol for SIMATIC S5 PLCs

### Driver Use

The SINECH1 driver is used for data exchange between Asix computers and SIMATIC S5 PLCs provided with the CP 1430, via an Ethernet network. An Asix system computer must be provided with the SIEMENS CP1413 card.

The cooperation of the Asix system with the PLC by use of the SINECH1 protocol needs developing the controller's program supporting the CP 1430.

Parameterization of SINECH1 driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the SINECH1 driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel

*Driver:* SINECH1

**SINECH1** tab:

*fTsapPC, fTsapPLC, rTsapPC, rTsapPLC, MAC*

where:

fTsapPC	- name of TSAP used for the reading operation (FETCH) in the Asix computer;
fTsapPLC	- name of TSAP used for the reading operation (FETCH) in the controller;
rTsapPC	- name of TSAP used for the writing operation (RECEIVE) in the Asix computer;
rTsapPLC	- name of TSAP used for the writing operation (RECEIVE) in the controller;
MAC	- MAC network address assigned to the controller.

**NOTE** Names of TSAPs are any 8-character strings. Names of TSAPs declared in the Asix system must have their equivalents among TSAPs declared by means of COM 1430 on the controller side. Reading and writing from/to the controller is executed within separate connections.

### EXAMPLE

An exemplary declaration of transmission channel using the SINECH1 protocol is given below:

*Channel name:* CHAN1

*Driver:* SINECH1

*Channel parameters:* LocFetch,TestFetc,LocRecev,TestRece,08:00:06:01:00:22

The logical channel named CHAN1 has the following parameters defined:

SINECH1 protocol,

LocFetch - name of TSAP used for the reading operation (FETCH) in the Asix system,

TestFetc - name of TSAP used for the reading operation (FETCH) in the controller,

LocRecev - name of TSAP used for the writing operation (RECEIVE) in the Asix computer,

TestRece - name of TSAP used for the writing operation (RECEIVE) in the controller,

08:00:06:01:00:22 - MAC network address assigned to the controller

## Addressing the Process Variables

The rules of creating the symbolic addresses of variables belonging to the SINECH1 driver channel are the same as in case of the channel using the AS512 protocol

The SINECH1 driver is loaded as a DLL automatically.

## Driver Configuration

SINECH1 driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **SINECH1** section.

**Section name: SINECH1**

**Option name: LOG\_FILE**

**Option value: file\_name**

Meaning - the item allows to define a file to which all diagnostic messages of the SINECH1 driver and the information about contents of telegrams received by the driver are written. If the item does not define the full path, then the log file is created in the current directory. The log file should be used only while the Asix start-up.

Default value - by default, the log file is not created.

**Section name: SINECH1**

**Option name: LOG\_OF\_TELEGRAMS**

**Option value: YES|NO**

Meaning - the item allows to write to the log file (declared by use of the item LOG\_FILE) the contents of telegrams sent within the communication

with the controller. Writing the contents of telegrams to the log file should be used only while the Asix start-up.

Default value - by default, the telegrams are not written.

- Section name: SINECH1**
- Option name: LOG\_FILE\_SIZE**
- Option value: number**

Meaning - the item allows to specify the log file size in MB.

Default value - by default, the item assumes that the log file has a size of 1 MB.

- Section name: SINECH1**
- Option name: READ\_ONLY**
- Option value: YES|NO**

Meaning - option to use to the communication with the server providing only the channel for reading.

Default value - by default, NO.

- Section name: SINECH1**
- Option name: CHECKSUM**
- Option value: YES|NO**

Meaning - option to use to the communication with the server that does not use a checksum.

Default value - by default, NO.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

- Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:  
 YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

**Redundancy**

Purpose - with this option searches a redundant channel when connection with a controller is broken in local channel.

Option value:  
*computer name* - list including the names of network computers which will be permitted to search a redundant channel.

The default value - by default, when option is enabled, the channel is searched on all computers.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.101 SINECL2 - Driver of PROFIBUS Protocol for SIMATIC S5 PLCs

### Driver Use

The SINECL2 driver is used for data exchange between Asix computers and SIEMENS SIMATIC S5 PLCs provided with the CP5430 card, with the aid of the SINEC L2 local network based on the FDL protocol (2-nd level of the PROFIBUS standard). An Asix system computer must be provided with the CP5412 (A2) or CP5613 communication processor and system software used to support this processor.

The software of the controller dedicated for operation together with the Asix system must meet the following requirements:

- program in the controller must contain calls of function blocks executing receiving and sending telegrams through the CP5430 according to the FDL protocol;
- number of the node given to the CP5430 must be unique within the local SINEC L2 network;
- parameters of the CP5430 and CP5412(A2) or CP5613 operation must be compatible.

In each transmission channel established between any Asix system computer and any CP5430 communication processor, it is allowed to use 10 subchannels, multiplying in this way a quantity of transferred information.

Parameterization of SINECL2 driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the SINECL2 driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel

*Driver:* SINECL2

**SINECL2** tab:

*Channel parameters:*

*PC\_node, PLC\_node [, subchannel]*

where:

*PC\_node*

- number of the node assigned to the application computer;

*PLC\_node*

- number of the node assigned to the CP5430 processor in the controller with which the connection is to be executed in a given transmission channel;

*subchannel*

- numbers of subchannels which are used in a given transmission channel. An absence of this parameter signifies that all the subchannels (from 1 to 10 inclusive) are used in a given channel.

#### EXAMPLE

Exemplary items declaring transmission channels working according to the SINEC L2 protocol are given below:

*Channel name:* CHAN2  
*Driver:* SINECL2  
*Channel parameters:* 5,1,1,2,3,4

*Channel name:* CHAN3  
*Driver:* SINECL2  
*Channel parameters:* 5,2

The transmission channel with the logical name CHAN2 has the following parameters defined:

- SINECL2 protocol using the SINEC L2 local network;
- number of the node assigned to the computer - 5;
- number of the node assigned to the CP5430 communication processor - 1;
- used subchannels with numbers 1, 2, 3 and 4.

The transmission channel with the logical name CHAN3 has the following parameters defined:

- SINECL2 protocol using the SINEC L2 local network;
- number of the node assigned to the computer - 5;
- number of the node assigned to the CP5430 communication processor - 2;
- all the subchannels are used.

## Addressing the Process Variables

The rules of creating the symbolic addresses belonging to the SINECL2 type channel are the same ones as for AS512 type channels.

The SINECL2 protocol driver requires installing the CP5412(A2) or CP5613 card and the driver of this card supplied in the DP-5412/Windows NT packet by SIEMENS.

The SINECL2 protocol driver is loaded as a DLL automatically.

## Driver Configuration

SINECL2 driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **SINECL2** section.

- Section name:** SINECL2
- Option name:** ALARM\_LOG
- Option value:** YES|NO

Meaning - declaration of writing to the log file the messages about an arrival of alarm telegrams.

Default value - NO.

Defining - manual.

- Section name:** SINECL2
- Option name:** ERROR\_LOG
- Option value:** YES|NO

Meaning - declaration of writing to the log file the messages about transmission errors.

Default value - NO.  
 Defining - manual.

- Section name: SINECL2**
- Option name: MAX\_NOUMBER\_OF\_NODES**
- Option value: number**

Meaning - declaration of maximal number of nodes in the SINECL2 network.  
 Default value - 16.  
 Defining - manual.

- Section name: SINECL2**
- Option name: SAP**
- Option value: id, PC\_SAP, PLC\_SAP**

Meaning - declaration of mapping the SAPs, creating a logical channel named id, on the PC and PLC sides;  
 - by default, 10 pairs of SAPs mapped as follows (id, PC\_SAP, PLC\_SAP) are used:  
 1, 35, 45  
 2, 36, 46  
 3, 37, 47  
 4, 38, 48  
 5, 39, 49  
 6, 40, 50  
 7, 41, 51  
 8, 42, 52  
 9, 43, 53  
 10, 44, 54  
 Defining - manual.

- Section name: SINECL2**
- Option name: STATISTICS**
- Option value: YES|NO**

Meaning - transmission statistics writing to the log file every minute.  
 Default value - NO.  
 Defining - manual.

- Section name: SINECL2**
- Option name: TIMEOUT**
- Option value: number**

Meaning - timeout for an answer from the controller expressed in ticks of duration time equal to 400 ms.  
 Default value - 3.  
 Defining - manual.

- Section name: SINECL2**
- Option name: LOG\_FILE**
- Option value: name**

Meaning - declaration of the log file name with diagnostic messages of the SINECL2 driver.  
 Default value - log file is not created.  
 Defining - manual.

**Section name: SINECL2**

**Option name: CP\_TYPE**

**Option value: number**

Meaning - declaration of the communication processor card type used in the Asix system computer. Two types are allowable:  
1/ CP5613 identified by the number 5613;  
2/ CP4512 (A2) identified by the number 5412.

Default value - 5412 (CP5412 (A2) communication processor).

Defining - manual.

**Section name: SINECL2**

**Option name: DELAY**

**Option value: number**

Meaning - timeout declaration (in milliseconds) after an occurrence of SDA telegram sending error.

Default value - 0.

Defining - manual.

**Section name: SINECL2**

**Option name: ACTIVE\_TELEGRAMS\_SAP**

**Option value: number**

Meaning - the item allows definition of SAP number that is used for servicing active telegrams.

Default value - 0.

Defining - manual.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).  
 The default value - by default, all computers can read variables.

**Redundancy**

Purpose - with this option searches a redundant channel when connection with a controller is broken in local channel.

Option value:

*computer name* - list including the names of network computers which will be permitted to search a redundant channel.  
 The default value - by default, when option is enabled, the channel is searched on all computers.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO  
 The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.  
 The default value - by default, no time for data validity is set.

## 1.102 SNG - Driver to Exchange Data with SNG Devices

### Driver Use

SNG protocol driver is used for data exchange between the Asix system and SNG systems provided by the Warsaw based Synergia Tech company, via an Ethernet link. CtSNG driver configuration is performed using the Architect application.

### Transmission Channel Declaration

Declaration of the transmission channel utilizing the SNG driver requires adding a channel with the following parameters to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: SNG

**SNG** tab:

Channel parameters:

*ServerIP=IPaddress;*      [*Port=number;*]      *SNG=address;*      *AlarmNo=number;*  
*[Timeout=number]*

**where:**

*ServerIP*                    - IP address of the CommServer of the SNG\_Synergia system  
*Port*                         - CommServer port number of the SNG\_Synergia system. Default  
8888,  
*SNG*                         - Source Physical Address of the SNG system assigned to transmission  
channel,  
*Timeout*                    - Max. time (in seconds) between two consecutive frames sent  
by the CommServer (CommServer sends a 'heart-beat' frame, if there are no frames  
generated by the SNG\_Synergia devices). By default, 15 seconds,  
*AlarmNo*                    - The alarm number reported to Asix alarm system in the absence of  
communication with the SNG\_Synergia system.

#### EXAMPLE

Example of a declaration of transmission channels to communicate with the CommServer with an IP address 87.205.117.187 and the default port 8888. SNG address assigned to the channel is 2.1.5:

ServerIP=87.205.117.187; SNG=2.1.5; AlarmNo=100

## Declaration of Variables

The general syntax of the process variable symbolic address is as follows:

`<data_type>: <group_address>: <source_address>: <return_address>`

where:

*data\_type* - Number in the range 1 - 6 determines the type of the SNG data. It should be given only when a variable is used for control.

*group\_address* - group address of the device in the SNG\_Synergia system in the **a/b/c** format.

It should be given only when a variable is used for control.

*source\_address* - The physical address of the source in **a.b.c** format, from which the 'Feedback' messages that supplies the current value of the variable arrive. It should be given only when a variable is fed with data from the messages of the 'feedback' type;

*return\_address* - A group source address in the **a/b/c** format, from which the 'Feedback' messages that supplies the current value of the variable arrive. It should be given only when a variable is fed with data from the messages of the 'feedback' type.

Depending on the application of the variable, its symbolic address can take form:

a/ control only variable:

**`<data_type>: <group_adres>`**

b/ monitoring only variable:

**`<source_address>: <return_address>`**

c/ control and monitoring variable:

**`<data_type>: <group_address>: <source_address>: <return_address>`**

Examples of variable declarations:

# JJ\_01 is only used to control the **On/Off** type device in the 1/2/3 group address.  
JJ\_01, , 1:1/2/3, K1, 1, 1, NIC

# JJ\_02 only supports the 'feedback' from the device with a physical SOURCE address 1.2.3 and  
# group return address 1/0/23 (data type arrives in the 'feedback' frame and is neither **Time** nor  
# **Date** type)  
JJ\_02, , 1.2.3:1/0/23, K1, 1, 1, NIC

# JJ\_03 device IS used to control the **Value** type device with the 1/2/3 group address and receives

```
# 'Feedback' of a different type than Time or Date from the device with the source  
physical address of 1.5.6  
# and group address 1/0/44  
JJ_03, , 6:1/2/3:1.5.6:1/0/44, K1, 1, 1, NIC
```

```
# JJ_04 only supports the Time type 'feedback' from a device with a source physical  
address of 1.2.3  
# and return group address of 1/0/23  
JJ_04, , 1.2.3:1/0/23, K1, 10, 1, NIC_TEXT
```

```
# JJ_05 only supports the Date type 'feedback' from a device with a source physical  
address of 1.2.3  
# and return group address of 1/0/23  
JJ_05, , 1.2.3:1/0/23, K1, 12, 1, NIC_TEXT
```

## Driver Parameters

The SNG driver parameters are declared in the *Miscellaneous* module, in the *Directly Entered options* tab.

The driver configuration is done using a separate section called **CTSNG**

**Section name:** CTSNG

**Option name:** LOG\_FILE

**Option value:** *log\_file\_name*

Purpose: - the text log file to which driver status messages are stored is used for diagnostic purposes.

The default value: - by default, no log file is created.

**Section name:** CTSNG

**Option name:** LOG\_FILE\_SIZE

**Option value:** *number*

Purpose: - this item is used to determine the size of the log file defined using LOG\_FILE item.

Option value:

*number* - the size of the log file in MB.

The default value: - the default log file size is 10 MB.

**Section name:** CTSNG

**Option name:** LOG\_OF\_TELEGRAMS

**Option value:** YES | NO

Purpose: - the item allows writing the contents of messages communicated between the driver and the controllers to the log file (declared using the LOG\_FILE item) The item in question should only be used during the Asix system start-up.

The default value: - default value of the option is NO.

### EXAMPLE

Sample driver parameterization

*Section name:* CTSNG•

*Option name:* LOG\_FILE•

*Option value:* d:\tmp\CtSNG\mvi.log•

*Section name:* CTSNG•

*Option name:* LOG\_FILE\_SIZE•

Option value: 20

Section name: CTSNG

Option name: LOG\_OF\_TELEGRAMS

Option value: YES

## Channel Parameters

The parameters included in the channel section (in the *Miscellaneous* module, the *Directly entered options* tab).

In the section called *<channel\_name>* it is possible to include declaration items:

- create a log file
- the log file size
- messages log.

The purpose of this item is similar to the driver's purpose, but the information stored in files declared in the channel section refer to a single driver channel.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

## Communication Drivers

The default value - by default, all computers can read variables.

### **Redundancy**

Purpose - with this option searches a redundant channel when connection with a controller is broken in local channel.

Option value:  
*computer name* - list including the names of network computers which will be permitted to search a redundant channel.

The default value - by default, when option is enabled, the channel is searched on all computers.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.103 SNMP - Driver for Reading and Writing Object Values Using SNMPv1 and SNMPv2c Protocol

### Driver Use

**SNMP.** The driver can read and write objects values using SNMPv1 and SNMPv2c protocol - manage the various elements of telecommunications networks, such as routers, switches, computers and telephone exchanges. The driver performs its functions by using the SNMP Management API.

### Declaration of Transmission Channel

Declaration of the transmission channel using the SNMP driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* SNMP

**SNMP/Channel parameters** tab:

**Remote device address** - IP address of remote device;

**Community string** - community string defined on remote device; default: public;

**Transmission timeout** - transmission timeout in milliseconds; default: 1000 ms;

**Number of retries** - number of retries of transmission; default: 3;

**Log file** - the item allows to define a file to which all the diagnostic messages of the driver will be written; default: snmp.log.

### Declaration of Variables

Declaration of variable takes the form of identifiers sequence separated by dots, e.g.: ".1.3.6.1.2.1.4.1.0". the last digit is zero for scalar values or index in an array of values (starting form 1). The index may also have a more complex structure depending on the object. Instead of numerical identifiers occurring in the given address you can use symbolic identifiers - if they are defined on local computer (in mib.bin file); for example: ".iso.org.dod.internet.mgmt.mib-2.system.sysDescr.0". If the address begins with a dot - it's absolute address. Otherwise, the address is relative to "iso.org.dod.internet.mgmt.mib-

2". That means: you can declare only "system.sysDescr.0" to get a description of the system. The variable type must be compatible with the type of object.

## Channel Parameters - Trap Processing

Since the version 3 the driver supports the traps. The driver receives messages about the traps coming from remote devices mentioned in the channel definition. Trap processing may be disabled with the use of the parameter Traps. Supporting traps consists in generating alarms and setting the values got in a received trap to a variable.

### **Traps**

Meaning - this option enables trap processing.

Option value:  
YES / NO

The default value - YES - by default, traps are supported.

### **Global alarms**

Meaning - this option controls the type of alarms generated by the driver. If you specify YES, the global alarms will be generated. If not, local alarms will be generated.

Option value:  
YES / NO

The default value - NO - by default, local alarms are generated.

### **Trap mode**

Meaning - together with the trap an additional data can be received. This data is marked by the object identifier and its values can be used to assign the values to the variable with the same identifier (address) as the identifier of received datum. The default way the additional data is received with the trap is declared by the parameter *Trap mode*.

Option value:

-1 - data from traps is not used to assign the values to variables. The variables get values by physical reading from a device in accordance with a declared refresh period.

0 - variables get data only from the data of received traps. The declared refresh period initiates reading values from driver external buffers.

1 - works as in '-1', but while driver start-up the physical readout of values from a device occurs. The physical readout takes place also at the moment the loss of communication is found and if the period longer than declared in the parameter *Traps refreshing* elapsed from the last correct value received.

The default value - -1.

**Notice:** Regardless of the value of the parameter *Trap mode*, trap support for each variable can be defined by adding the string [TR-],[TR0] and [TR1] (for mode: -1,0,1 adequately) to the variable object identifier. E.g. the variable 1.3.6.1.2.1.4.1.0[TR0] will be supported in the mode 0. The mode set in this way has priority over the parameter *Trap mode*.

## Declaration of Channel Parameters for Different Types of Alarms/Traps

### **Events of 'cold start' type**

Parameters:

*Number of the alarm in the Asix system*

*Alarm scope: Local, Global or default* - alarm scope L-Local, G-Global. Empty value means the scope defined by the *Global alarms* option.

*Parameters passed to the Asix system together with the alarm* - as a value one should enter indexes (comma separated) of the items obtained together with the

trap that should be passed to the Asix system together with the alarm. The index '-1' means the time of the event counted from the start of the remote device expressed as a multiple of 10ms value.

**Events of 'warm start' type**

Parameters:

*Number of the alarm in the Asix system*

*Alarm scope: Local, Global or default* - alarm scope L-Local, G-Global. Empty value means the scope defined by the *Global alarms* option.

*Parameters passed to the Asix system together with the alarm* - as a value one should enter indexes (comma separated) of the items obtained together with the trap that should be passed to the Asix system together with the alarm. The index '-1' means the time of the event counted from the start of the remote device expressed as a multiple of 10ms value.

**Events of 'authentication failure' type**

Parameters:

*Number of the alarm in the Asix system*

*Alarm scope: Local, Global or default* - alarm scope L-Local, G-Global. Empty value means the scope defined by the *Global alarms* option.

*Parameters passed to the Asix system together with the alarm* - as a value one should enter indexes (comma separated) of the items obtained together with the trap that should be passed to the Asix system together with the alarm. The index '-1' means the time of the event counted from the start of the remote device expressed as a multiple of 10ms value.

**Events of 'EGP links' type**

Parameters:

*Number of the alarm in the Asix system*

*Alarm scope: Local, Global or default* - alarm scope L-Local, G-Global. Empty value means the scope defined by the *Global alarms* option.

*Parameters passed to the Asix system together with the alarm* - as a value one should enter indexes (comma separated) of the items obtained together with the trap that should be passed to the Asix system together with the alarm. The index '-1' means the time of the event counted from the start of the remote device expressed as a multiple of 10ms value.

**Events of 'link up/down' type**

Parameters:

*Interface no* - number of the interface to which the trap is related

*Alarm no* - number of the alarm in the Asix system

*Alarm scope* - L-local, G-global, empty value means the scope defined by the *Global alarms* option

*Parameters* - parameters passed to the Asix system. As a value one should enter indexes (comma separated) of the items obtained together with the trap, that should be passed to the Asix system together with the alarm. The index '-1' means the time of the event counted from the start of the remote device expressed as a multiple of 10ms value.

**Events of 'enterprise specific' type**

Parameters:

*Event type* - trap number/type

*Alarm no* - number of the alarm in the Asix system

*Alarm scope* - L-local, G-global, empty value means the scope defined by the *Global alarms* option

*Parameters* - parameters passed to the Asix system. As a value one should enter indexes (comma separated) of the items obtained together with the trap that should be passed to the Asix system together with the alarm. The index '-1' means the time of the event counted from the start of the remote device expressed as a multiple of 10ms value.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

- Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.
- Option value:  
YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

- Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.
- Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

- Purpose - with this option deployed, local control in a channel will be disabled.
- Option value:  
YES / NO
- The default value - NO.

### **Data Validity Period**

- Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.
- Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.
- The default value - by default, no time for data validity is set.

## 1.104 SNPX - Driver of SNPX Protocol for GE Fanuc PLCs

### Driver Use

The driver of the SNPX protocol (Series Ninety Protocol) is used for data exchange between Asix system computers and GE\_FANUC 90-30 PLCs as well as GE\_FANUC 90 CMM and PCM modules. The communication is executed by means of serial links.

The driver allows access to the following controller variable types:

- Discrete Inputs (%I),
- Discrete Outputs (%O),
- Discrete Internals (%M),
- Analog Inputs (%AI),
- Analog Outputs (%AO),
- Registers (%R),
- Genius Global Data (%G).

The driver does not handle the following controller variable types (system and temporary types):

- %SA Discrete,
- %SB Discrete,
- %SC Discrete,
- %S Discrete,
- Discrete Temporary (%T).

Parameterization of SNPX driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the SNPX driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel  
*Driver*: SNPX

**SNPX** tab:

*Channel parameters*:

Port = *number*; [Baudrate = *number*;] [ParityBit = *check\_parity\_name*;]  
 [TimeSynchrX = *address[:period]*;] [T4 = *timeout\_break*;] [T2 = *response\_timeout*;]  
 [TBroadCast = *timeout\_broadcast*]

where:

*Port* - number of the COM serial port;  
*BaudRate* - transmission speed between the computer and the controller;  
 there are the following acceptable values: 300, 600, 1200, 2400, 4800, 9600, 19200 Bd; a default value - 19200 Bd;

- ParityBit* - determines the method of frame parity check; there are the following acceptable values: NONE, ODD, EVEN; default value - ODD (odd parity check);
- T4* - timeout (in milliseconds) between sending BREAK and BROADCAST ATTACH; a default value - 50 milliseconds;
- TBroadcast* - timeout between sending BROADCAST ATTACH and sending the first request to the controller; a default value - 2000 milliseconds;
- T2* - timeout (in milliseconds) for receiving the first bit of the response; a default value - 2000 milliseconds;
- SynchrCzasuX* - cyclic (every period in seconds) date & time frame write at the given address in the controller; there are 99 positions of time synchronization from the range of names from SynchrCzasu1 to SynchrCzasu99; if the parameter period is not given, the synchronization is performed every 60 seconds by default; the date & time frame synchronization is compatible with SVCREQ 7 - the procedure of date & time write:

```

struct    dateTime
{
    byte Year;
    byte Month;
    byte Day;
    byte Hour;
    byte Minute;
    byte Second;
    byte DayOfWeek;
    byte NotUsed;    // always 0
    word    wSynchr;    // set to 1 at new date & time frame write
};
    
```

**NOTICE** The parameters given in the channel declaration must be compatible with the parameters set for communication ports of the controllers handled by this channel.

**EXAMPLE**

An exemplary declaration of the channel, in which the controllers with identifiers A123 and B456 are handled, is given below:

- 1/ for the controller with the A123 identifier - time is synchronized by writing to the register area beginning with R10 (every 25 seconds),
- 2/ for the controller with the B456 identifier - time is synchronized by writing to the register area beginning with R10 (with default frequency).

The communication with the controllers is performed over COM2 by means of standard transmission parameters, i.e. 19200 Bd, odd parity control, the first bit of stop and standard timeouts of the SNPX protocol.

*Channel name:* CHANNEL

*Driver:* CtSNPX

*Channel parameters:* Port=2; TimeSynchr1=A123.R10:25; TimeSynchr2=B456.R20

## Declaration of Variables

The driver makes the following variable types available:

- I - Discrete Input (%I) in BIT mode,
- IB - Discrete Input (%I) in BYTE mode,
- IW - Discrete Input (%I) in WORD mode,
- Q - Discrete Output (%I) in BIT mode,
- QB - Discrete Output (%I) in BYTE mode,
- QW - Discrete Output (%I) in WORD mode,

M	- Discrete Internal (%I) in BIT mode,
MB	- Discrete Internal (%I) in BYTE mode,
MW	- Discrete Internal (%I) in WORD mode,
G	- Genius Global Data (%G) in BIT mode,
GB	- Genius Global Data (%G) in BYTE mode,
GW	- Genius Global Data (%G) in WORD mode,
AI	- Analog Input (%AI) in WORD mode,
AO	- Analog Output (%AO) in WORD mode,
R	- Register (%R) treated as WORD,
RL	- two following Registers (%R) treated as DWORD,
RF	- two following Registers (%R) treated as FLOAT,

The syntax of the variable address is as follows:

[<CpuID>.<Type><Index>

where:

<i>CpuID</i>	- CPU identifier;
<i>Type</i>	- variable type name;
<i>Index</i>	- variable address within the framework of the variable type <i>Type</i> .

#### NOTICE

**1.** *CpuID* may be omitted in the variable address exclusively when only one controller is connected to the serial link. In such case, commands sent to the controller contain the identifier set at NULL (the real identifier set in the controller is unimportant).

**2.** *Index* indicates bit number, from which the range of bits ascribed to the variable begins, for discrete variables. *Index* may have one of the following values (by pattern of addressing used in VersaPro), depending on mode of making discrete variables available:

**a/** for BIT mode - any value w.e.f. 1,

**b/** for BYTE mode - value 1, 9, 17, etc. (numbers of the first bit of successive bytes),

**c/** for WORD mode - value 1, 17, 33, etc. (numbers of the first bit of successive words).

#### EXAMPLE

An exemplary variable declaration (the variable values come from the controllers identified by A123 and B456):

JJ_01, Register R3,	A123.R3,	CHANNEL, 1, 1, NOTHING
JJ_02, Analog Input 1,	A123.AI1,	CHANNEL, 1, 1, NOTHING
JJ_03, Discrete Input 3,	B456.I3,	CHANNEL, 1, 1, NOTHING
JJ_04, Discrete Output Byte 9 ,	A123.QB9,	CHANNEL,1,1, NOTHING_BYTE
JJ_05, Genius Global Word 17 ,	A123.GW17,	CHANNEL, 1, 1, NOTHING
JJ_06, Discrete Internal Word 33,	B456.MW33,	CHANNEL, 1, 1, NOTHING

## Driver Configuration

SNPX driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **SNPX** section.

- Section name: SNPX**
- Option name: LOG\_FILE**
- Option value: file\_name**

Meaning - the item allows to define a file to which all the diagnostic messages of the driver will be written.

Default value - by default, the log file is not created.

**Section name: SNPX**

**Option name: LOG\_FILE\_SIZE**

**Option value: number**

Meaning - the item allows to define the size of the log file in MB.  
Default value - by default, the item assumes that the log file has a size of 1 MB.  
Parameter:  
    number - size of the log file in MB.

**Section name: SNPX**

**Option name: LOG\_OF\_TELEGRAMS**

**Option value: YES | NO**

Meaning - the item allows writing to the log file (declared with use of the LOG\_FILE item) the contents of telegrams transmitted during the data exchange between the Asix system and controllers.  
Default value - NO.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.  
Option value:  
    YES / NO  
    *computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).  
The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.  
Option value:  
    *computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).  
The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**☑ *Local Write Denied***

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

**☑ *Data Validity Period***

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.105 SPA - Driver of SPA Protocol

### Driver Use

The SPA driver is used for data exchange between devices manufactured by ABB connected to the SPA bus and an Asix system computer. The communication is executed by means of serial interfaces in the RS232C or RS485 standard.

Parameterization of SPA driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the SPA driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* SPA

**SPA** tab:

*Channel parameters:*

*number, type, port, baud, AITxtOff, AIVaOff, password*

where:

*number* - number assigned to a remote device;  
*type* - remote device type:

1 – SPAJ 141C,  
2 – SPAM 150C,

*port* - serial port name;  
*baud* - transmission speed: 9600 or 4800 - it must be compatible to settings in the remote device;

*AITxtOff* - number added to the number of the text event read from the remote device in order to build an unique alarm number transferred to the Asix system;

*AIVaOff* - number added to the event number in order to build an unique alarm number transferred to the Asix system;

*password* - password allowing write operations to the remote device - it must be compatible to settings in the remote device.

#### EXAMPLE

The declaration of the logical channel named CHAN1 operating according to the SPA protocol and with parameters as below:

- remote device number - 4
- device type - SPAM 150 C
- port - COM1
- transmission speed - 9600 Bd
- number added to the number of a text event - 100
- number added to the number of an event with a value - 200

- password - 123

is as follows:

*Channel name:* CHAN1

*Driver:* SPA

*Channel parameters:* 4, 2, COM1, 9600, 100, 200, 123

The SPA driver is loaded as a DLL automatically.

## Addressing the Process Variables

The syntax of symbolic address which is used for variables belonging to the SPA driver channel is as follows:

*<variable\_type><channel>.<index>*

where:

*variable\_type* - type of the process variable;  
*channel* - channel number in the device from which the process variable is taken;  
*index* - process variable index within the type.

Types of process variables:

I - values of I category data,  
 O - values of O category data,  
 S - values of S category data,  
 V - values of V category data.

The range of used channels, types of supplied process variables, index range within each of types and meaning of individual elements within the type are specific for the remote device type.

A detail specification is included in the documentation of the remote device.

**NOTE** Raw values of all process variables are of *FLOAT* type.

### EXAMPLE

An example of the declaration of variables for the SPAM 150 C device (according to the documentation all the variables are placed in the channel no. 0):

X1, current in phase L1 , I0.1, CHAN1, 1, 1, NOTHING\_FP  
 X2, stimulation of stage Io> , O0.8, CHAN1, 1, 1, NOTHING\_FP  
 X3, coefficient p for thermal unit, S0.3, CHAN1, 1, 1, NOTHING\_FP  
 X4, current I measured during stimulation,V0.21,CHAN1, 1, 1, NOTHING\_FP  
 X5, current I measured during activation,0.41,CHAN1, 1, 1, NOTHING\_FP

## Generating the Alarms

Numbers of events, generated by remote devices, have the same variation range. In order to specify unambiguously which device the event under consideration come from, the SPA driver adds to the event number a number specified in the channel declaration as AllTxtOff (for text events) or AllValOff (for events with a value). In this way created number is transferred to the Asix system as an alarm number.

Beside an alarm number, the SPA driver transfers a number of the remote device from which a given event comes. The device number may be used in a message related to the alarm by giving a formatting string (%3.0f) in the content of the alarm message.

For transferring alarms the SPA driver uses the function *AsixAddAlarmGlobalMili()* by default. The item GLOBAL\_ALARMS allows to change the default settings and to transfer the alarms by means of the *function AsixAddAlarmMili()*.

Types of SPA remote devices implemented in the SPA driver generate only text events.

## Driver Configuration

SPA driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **SPA** section.

- Section name: SPA**
- Option name: LOG\_FILE**
- Option value: file\_name**

Meaning - the item allows to define a file to which all diagnostic messages of the SPA driver and information about contents of telegrams received and sent by the SPA driver will be written. If the item does not define the full path, then the log file is created in the current directory. The log file should be used only while the Asix system start-up.

Default value - by default, the log file is not created.

Defining - manual.

- Section name: SPA**
- Option name: LOG\_OF\_TELEGRAMS**
- Option value: YES|NO**

Meaning - the item allows to write to the log file (declared by use of the item LOG\_FILE) the contents of telegrams sent and received from the SPA bus within reading the process variables. Writing the contents of telegrams to the log file should be used only while the Asix system start-up.

Default value - by default, telegrams are not written.

Defining - manual.

- Section name: SPA**
- Option name: TRANSMISSION\_DELAY**
- Option value: number**

Meaning - the item allows to determine a time interval (as a multiple of 10 milliseconds) between two successive operations on the SPA bus.

Default value - by default, the item assumes a value of 1 (10 milliseconds).

Defining - manual.

- Section name: SPA**
- Option name: NUMBER\_OF\_REPETITIONS**
- Option value: number**

Meaning - the item allows to specify a number of repetitions in case of a transmission error.

Default value - by default, the item assumes a value of 0 (no repetitions).

Defining - manual.

- ☑ **Section name: SPA**
- ☑ **Option name: DATA\_UPDATE**
- ☑ **Option value: number**

Meaning - the item allows to specify a time period (in seconds) after exceeding of which you should update values of process variables stored in internal buffers of the driver.

Default value - by default, the item assumes a value of 5.

Defining - manual.

- ☑ **Section name: SPA**
- ☑ **Option name: TIME\_UPDATE**
- ☑ **Option value: number**

Meaning - the item allows to specify a time period (in seconds) after exceeding of which you should send an actual time to remote devices.

Default value - by default, the item assumes a value of 1.

Defining - manual.

- ☑ **Section name: SPA**
- ☑ **Option name: DATE\_UPDATE**
- ☑ **Option value: number**

Meaning - the item allows to specify a time period (in seconds) after exceeding of which you should send an actual date to remote devices.

Default value - by default, the item assumes a value of 30.

Defining - manual.

- ☑ **Section name: SPA**
- ☑ **Option name: ALARM\_UPDATE**
- ☑ **Option value: number**

Meaning - the item allows to specify a time period (in seconds) which separates the successive cycles of reading alarm buffers of all remote devices supported by individual serial interfaces.

Default value - by default, the item assumes a value of 1.

Defining - manual.

- ☑ **Section name: SPA**
- ☑ **Option name: CHECKSUM**
- ☑ **Option value: YES|NO**

Meaning - the item allows to control the building of checksum in telegrams sent to the SPA bus. If the item has the value NO, then two characters X are inserted in telegram instead of the checksum.

Default value - by default, the checksum is built.

Defining - manual.

- ☑ **Section name: SPA**
- ☑ **Option name: TELEGRAM\_EXCLUSION**
- ☑ **Option value: YES|NO**

Meaning - the item allows to exclude, from the list of supported telegrams, telegrams which are stamped by an addressed device with a code N (illegal range of variables in the telegram or not supported type of variables). The exclusion of telegrams allows to use interfaces efficiently.

Default value - by default, telegrams are excluded.

Defining - manual.

- Section name: SPA**
- Option name: GLOBAL\_ALARMS**
- Option value: YES|NO**

Meaning - the item controls the way of transferring alarms read from remote devices to the alarm system in the Asix system.

Default value - default alarms are transferred to the alarm system as global alarms (transferred to the alarm system by means of the function `AsixAddAlarmGlobalMili()`). Setting the value of the item `GLOBAL_ALARMS` on `NO` causes that alarms are transferred to the alarm system by means of the function `AsixAddAlarmMili()`.

Defining - manual.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:  
YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

### **Redundancy**

Purpose - with this option searches a redundant channel when connection with a controller is broken in local channel.

Option value:  
*computer name* - list including the names of network computers which will be permitted to search a redundant channel.

The default value - by default, when option is enabled, the channel is searched on all computers.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.106 Srio - Driver of SRIO 500M Protocol

### Driver Use

Srio. The exchange of data with the SRIO 500M hub (ABB-produced), which provides communication between the host system and the devices connected to the SPA bus, such as protection relays and signalling devices. Communication with the hub is accomplished via RS232-compliant serial communication, data transport layer - based on the ANSI X3.28 protocol in full-duplex mode with the BCC checksum.

The controller is a gateway between the security network SPA and a computer. The controller communicates with the security network using the SPA protocol, and with the computer using the ANSI X3.28 protocol.

Process variables get from the security network SPA are mapped in SRIO into so-called ANSI variables. The process of defining ANSI variables is made at the stage of setting up SRIO and is performed using the SRIO manufacturer's software tools.

ANSI variables have a unique number belonging to one of four groups marked with symbols: DI, DO, AI and AO. The way of servicing the variable on computer interface depends on the group to which the variable belongs. The variables DI and DO are treated as 16-bit integers unsigned, while the variables AI and AO are treated as 32-bit integers signed.

The driver Srio realizes read and write operations of ANSI variables on computer demand and supports spontaneous messages sent by the controller SRIO related to:

- any change of values in variables belonging to DI or DO group;
- change the values of AI and AO group with a value greater than the value of 'delta';
- the occurrence of event or alarm in security.

Additionally, Srio allows you to:

- configure variables ANSI in a SRIO,
- configure event masks of SPA devices,
- direct communication with securities using the native protocol SPA (SPA messages),
- write the configuration into SPA devices,
- write the SRIO configuration into EEPROM,
- support information on events in external variables of the driver,
- delete information on events in internal variables of the driver,
- link events with selected controls,
- read interference files.

Parameterization of Srio driver is performed with the use of Architect module.

### Declaration with transmission channel

Declaration of the transmission channel using the Srio driver requires a channel with the following parameters to be added to the Current data module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* Srio

**CtSrio** tab:

**Channel parameters:**

port=number; nrPc=number; nrSrio=number; enq=number; timeout=number;  
 nrAlarmu=number; defAnsi=fileName; defAlarmow=fileName [;defSynop=fileName]  
 [;KartZaklocen=directoryPath]

where:

port - number of a serial port COM,  
 nrPc - parameter 'host num' set by the function ansi\_setup of SRIO terminal,  
 nrSrio - parameter 'own num' set by the function ansi\_setup of SRIO terminal,  
 enq - max number of ENQ sent in the absence of DLE ACK/NAK,  
 timeout - timeout (in milliseconds) for first character of query response,  
 nrAlarmu - number of an Asix alarm set when communication is lack,  
 defAnsi - name (path) of the file with ANSI point definition,  
 defAlarmow - name (path) of the file with event definitions,  
 defSynop - name (path) of the file with definitions of relations between controls,  
 spontaneous telegrams and variables of blinking state,  
 kartZaklocen - directory in which files read form disturbance recorders are saved. By  
 default, disturbance files are in subfolder Disturbances (created by the driver in Asix  
 system directory).

Parameters of transmission are fixed at:

- baud rate: 9600 bps,
- 8 bits of data,
- parity check (even),
- 1 stop bit.

## Types of Variables

The driver provides the following types of variables:

**A** - ANSI variable,  
**F** - internal variable that stores information about the occurrence of spontaneous events  
 or changes of ANSI point assigned to that variable in the file of event definition,  
**S** - device status in security network,  
**Z** - internal variable used to maintain information on the occurrence of event assigned to  
 that variable in the file with definitions of relations between controls, spontaneous  
 telegrams and variables of blinking state.

It is assumed that the address of the variable will have the syntax:

**<Type>[.<No>[.<Index>] ]**

where:

Type - name of variable type (A, F, S or Z);  
 No - ANSI number (for A type) or number of device in the security network (for S type).  
 Virtual variables of type F or Z does not require a parameter No.  
 Index - number of bit for variables of A type relating to a DI or DO. It is assumed that  
 variables of A type with address 0-1999 enable access to real variables ANSI in SRIO:  
 0 - 499 variables of DI type  
 500 - 999 variables of DO type  
 1000 - 1499 variables of AI type  
 1500 - 1999 variables of AO type

## Communication Drivers

Variables of A type with addresses 16000 - 16010 will be used to carry out functions related to configuration of SRIO and direct communication with securities using the native protocol SPA (SPA messages):

- 16000 - name of the file with configuration of ANSI variables
- 16001 - request to save the file with configuration of ANSI variables into SRIO
- 16002 - name of the file with configuration of event masks for SPA devices
- 16003 - request to save the file with configuration of event masks for SPA devices
- 16004 - name of the file with configuration of SPA device
- 16005 - request to save the file with configuration of device into SPA
- 16006 - number of currently processed command element
- 16007 - saving configuration of SRIO into EEPROM
- 16008 - content of 'SPA message' sent to SRIO
- 16009 - content of response for 'SPA message'
- 16010 - number of bytes which we ask for when reading form 'SPA bus reply' (default is 64 bytes, min. 16 bytes, max. 255 bytes)

Variables of A type with addresses 16010 - 16019 are used to carry out functions related to handling disturbance recorder:

- 16011 - address of SPA network device which all operations concerning disturbance recorder will apply to
- 16012 - write of any value into this variable initiates readout of disturbance recorder directory from the device which address (in SPA network) is indicated by the variable with the address 16011
- 16013 - number of disturbances recorded in disturbance recorder directory
- 16014 - reserve
- 16015 - write into a variables initiates readout of interference with w number equal to the value of control value
- 16016 - write any value into a variable causes removal form a disturbance recorder an oldest interference
- 16017 - total number of lines in the current read disturbance file (set at the moment of start of reading disturbance file)
- 16018 - counter of lines read from the current read disturbance file (updated during the execution of disturbance file readout)
- 16019 - compression rate used when reading a disturbance file (an integer in the range of 0-5); by default: 0 (without compression)
- 16500 - write any value into a variable causes sending current PC time into SRIO. By default: local time is sent
- 16501 - write the value of 1 into a variable causes forcing refresh of all the ANSI variables. Write the value of 0 into a variable causes refresh the ANSI variables the definitions of which were changed/added as a result of uploading the configuration into a SRIO.

Variables of A type with addresses form 16020 to 16039 are used for the presentation of the names of recorder disturbance file when reading disturbance directories. The names of files are listed from oldest to youngest. The variable with the address 16020 contains the name of the oldest file in interference directory.

### **NOTICE:**

For the variables with the addressses 16020 - 16039 the conversion function NOTHING\_TEXT should be used.

Exemplary declarations of variables:

```
DI_90, , A.90, PT, 1, 1, NOTHING
DI_90_00, , A.90.0, PT, 1, 1, NOTHING
DI_90_01, , A.90.1, PT, 1, 1, NOTHING
DO_580, , A.580, PT, 1, 1, NOTHING
AI_1000, , A.1000, PT, 1, 1, NOTHING_FP
AO_1502, , A.1502, PT, 1, 1, NOTHING_FP
SRIO_FILE_NAME, , A.16000, PT, 255, 1, NOTHING_TEXT
```

```

SRIO_FILE_RQ, , A.16001, PT, 1, 1, NOTHING
SPA_MESS_SD, , A.16008, PT, 80, 1, NOTHING_TEXT
SPA_MESS_RV, , A.16009, PT, 255, 1, NOTHING_TEXT
SPA_LEN, , A.16010, PT, 1, 1, NOTHING
ST_203, , S.203, PT, 1, 1, NOTHING
ZADZ_01, , Z, PT, 1, 1, NOTHING
ZADZ_02, , Z, PT, 1, 1, NOTHING
DIST_UNIT_NR, , A.16011, PT, 1, 1, NOTHING
DIST_READ_RQ, , A.16015, PT, 1, 1, NOTHING
DIST_LINES_CNT, , A.16017, PT, 1, 1, NOTHING
DIST_CURR_LINE, , A.16018, PT, 1, 1, NOTHING
DIST_FILE_1, , A.16020, PT, 255, 1, NOTHING_TEXT
DIST_FILE_2, , A.16021, PT, 255, 1, NOTHING_TEXT
SET_DATE_TIME, , A.16500, PT, 1, 1, NOTHING

```

## Event Processing

Each event sent from a SRIO into a computer has a following parameters:

- number of SPA device;
- number of a channel in SPA device;
- number of an event;
- time stamp in the form: minute, second, millisecond;

For each event get from a SRIO which is to be reported to the Asix alarm system the way of its conversion should be set:

a/ whether it is only an Asix event (simultaneously the beginning and end of Asix alarm is reported)?

b/ whether it is an event which should be treated as an Asix alarm - and if so:

1. whether it is a signalling of the beginning of an Asix alarm (which one?);
2. whether it is a signalling of the end of an Asix alarm (which one /ones?);
3. whether an event should be reported as an Asix event (simultaneously beginning and end)?

c/ whether it is necessary to uphold the fact of appearing of an event in an internal variable.

Definitions of event processing are placed in text files. For each transmission channel (for each SRIO) a separate file is created and the name of the file is a parameter for the declaration of a transmission channel (parameter DefAlarmow). All files with the definitions of event processing can be placed in a common directory which name is given using EVENTS\_DEFINITIONS\_PATH position in the [CtSRIO].

### Definition of Processing of an Event Get From a SRIO

The syntax for the definition of event processing is as follows (all items separated by ' ; '):

```
[#] unit ; channel ; eventNr ; [eventSpec] ; [alarmStartSpec] ; [alarmEndSpec
[,alarmEndSpec]] ; [ZmZadzialanie[.value]] ; [ZmZapis] ; [AnsiAlarm.bitNr.val]
```

where:

# - an optional character that allows you to add a comment

#### Parameters identifying the event

*unit* - number of a SPA device,  
*channel* - number of a channel,

*eventNr* - number of an event,

### Parameters defining the way of event processing

*eventSpec* - definition of an Asix alarm which will be reported as an Asix event (alarmStart + alarmEnd with the same time),  
*alarmStartSpec* - definition of an Asix alarm which will be reported as the beginning of Asix alarm,  
*alarmEndSpec* - definition of Asix alarm / alarms which will be reported as the end of Asix alarm,  
*ZmZadzialanie* - name of a virtual variable which value is set to the value of the parameter value (default: 1) after the event occurrence. It is used to uphold the fact of event occurrence for the operator. The operator can modify the value of variable according to its preference.

### Specific use of event definition to register controls

*ZmZapis* - if the operation of write was made on the variable ZmZapis it takes effect in generation of Asix event (registration of controls in the history of alarms).  
Parameters: unit, channel and eventNr are ignored.

### Specific use of event definition to generate alarms of changing binary digits

*AnsiAlarm* - if the bit bitNr of ANSI point with the number AnsiAlarm will change the value, it results in generating Asix events according to the definitions given in alarmStartSpec and alarmEndSpec. It applies only to ANSI points with numbers up to 999 (binary digits) inclusive. Parameters: unit, channel and eventNr are ignored. By default, change of a bit from 1 into 0 (parameter val=0) is the end of the alarm.

### NOTICE:

If parameters identifying an event will not be found in the event definition file, the event will be ignored by the driver.

### Definition of Asix Alarm

The definition of an Asix alarm has the following syntax:

*<AlarmNo>[.K][.N][.U]*

where:

*AlarmNo* - number of Asix system alarm,

*K* - number of a channel in the event sent from a SRIO,

*N* - number of an event in the event sent from a SRIO,

*U* - number of a device in the event sent from a SRIO;

The parameters: K, N and U are optional and can be used to transfer max. 3 dynamic parameters with an alarm to Asix alarm system (all parameters are of WORD type).

The order of parameters: K, N and U in the definition of Asix alarm is arbitrary.

The order of dynamic parameter transferred to Asix alarm is compatible with the order of parameters in the alarm definition.

### Definition of Asix Alarm for Registering Control

In the case of registering control it is permitted to use a special syntax of the parameter alarmStartSpec in the definition of Asix alarm:

*<AlarmNo>.<value>, ... , <AlarmNo>.<value>*

where:

*AlarmNo* - number of an alarm in Asix system,  
*value* - value of control the sending of which to a SRIO will cause notification of an alarm with the number AlarmNo.

### Examples of the Definition of Event Processing

```
# activate the input 8 of the device 204 - the event 206 of Asix is reported with # the
channel number and device number, upholding the event by the
# variable ZZ_06
204; 8; 1; 206.K.U; ; ; ZZ_06 ; ;

# no communication with the device 203 - the beginning of the alarm 102 is reported with
# the number of device, upholding the event by the variable ZZ_02
203; 0; 53; ; ; 102.U; ; ; ZZ_02 ; ;

# restart of the device 203 - the beginning of the alarm 101 is reported with
# the number of device
203; 0; 50; ; ; 101.U; ; ; ;

# return of the communication with the device 203 - the end of the alarms 101 and 102 is
reported
# with the number of device, upholding the event by the variable ZZ_03
203; 0; 54; ; ; ; 101.U, 102.U ; ; ; ZZ_03 ; ;

# controlling the variable ZZ_01 results in the event 201 of Asix
; ; ; 201; ; ; ; ; ZZ_01 ;

# controlling the variable DO_580 results in: the event 305 of Asix, if the value of control
equals 0 or the event 306, if the value of control equals 1
; ; ; ; 305.0, 306.1 ; ; ; ; DO_580 ;
; ; ; ; 400 ; 400 ; ; ; ; 90.0.1

# the state of bit 2 in ANSI 90 set to 1 results in the beginning of the alarm 402
# the state of bit 2 in ANSI 90 set to 0 results the end of the alarm 402
; ; ; ; 402 ; 402 ; ; ; ; 90.2.1

# the state of bit 5 in ANSI 90 set to 0 results in the beginning of the alarm 405

# the status of bit 5 in ANSI 90 set to 1 results in the end of the alarm 405
; ; ; ; 405 ; 405 ; ; ; ; 90.5.0
```

## Configuration of SRIO and SPA Devices

The driver allows you to:

- configure ANSI variables in a SRIO,
- configure event masks of events of SPA devices supervised by a SRIO,
- configure SPA devices,
- save the current configuration of a SRIO into EEPROM.

The parameters used by the driver when configuring a SRIO and SPA devices are transferred in text files. The format of files is specific for each type of configuration.

### Configuration of ANSI Variables

Configuration of ANSI variables consists in transmission of content of configuration file to a SRIO. The process uses three virtual variables:

## Communication Drivers

- A.16000 - variable contains a full name of the configuration file,
- A.16001 - variable is used to begin the process of writing the configuration to a SRIO,
- A.16006 - variable is used to monitor the number of currently saved declaration from the configuration file.

The process of configuration of ANSI variables consists in execution of the following steps:

- preparation of a text file with declaration of ANSI variables,
- writing a full name of a file with ANSI variable declarations into the virtual variable A.16000,
- writing arbitrary value into the virtual variable A.16001. This record is treated by the driver as the request to write the configuration of ANSI variables into a SRIO.

Adoption of the request of saving the configuration by the driver is signaled by setting the value of 1 in the variable A.16001 and the value of 0 in the variable A.16006.

After saving the configuration of each of ANSI variable into a SRIO the value of the variable A.16006 increases by 1.

After saving the configuration of all variables into a SRIO the variable A.16001 takes the value of 2 for 1 second time, and after that the values of variables A.16001 and A.16006 are set to 0.

Before saving the configuration file the driver compares connectivity of configuration of ANSI points existing in a SRIO with the configuration which the operator intends to save to a SRIO. There are principles which the driver uses during execution of above operation:

a/ it is assumed that at the start of application there are ANSI points which are defined in a configuration file (specified in the declaration of transmission channel),

b/ during the operation of the driver the following items can be written into a SRIO:

- 1/ modifications of ANSI points already existing in a SRIO,
- 2/ definitions of new ANSI points,

c/ definitions identical to the definitions existing in a SRIO are ignored,

d/ driver stores its own map of ANSI points which is updated after each modification and after each operation of saving a new ANSI point.

### **NOTICE:**

The driver does not update the content of configuration file (given in the declaration of transmission channel). It is assumed that any changes in the configuration of ANSI points performed during operation of the application will be updated in the configuration file by the system engineer.

### **Format of ANSI Variable Declaration**

The declaration of ANSI variable has to be compatible with the format created by the SIGTOOL program of ABB company - i.e. it has to be a line of the text consisting of 14 elements separated by space characters and ending with CR and LF characters.

Line beginning with the character # or ; are treated as comment lines.

The meaning of individual elements of the declaration is as follows:

- 1 (SNR) - number of a SRIO
- 2 (ITEM) - beginning number of the declaration in the declaration file
- 3 (ADDR) - identifier of ANSI assigned to the variable
- 4 (BUS) - number of a serial link in a SRIO

- 5 (UNIT) - number of a device in SPA network
- 6 (CHS) - number of the first channel
- 7 (CHE) - number of the last channel
- 8 (IO) - identifier of the variable category (I, O, S, V)
- 9 (DAT1) - number of the first datum
- 10 (DAT2) - number of the last datum
- 11 (DTYPE) - variable type (0 - DI, 1 - AI, 2 - DI, 3 - AO)
- 12 (DFORM) - datum format (0 - binary, 1 - decimal, 2 - hexadecimal)
- 13 (DELTA) - mask of spontaneous telegrams
- 14 (STAT) - status. Individual status bits mean:

- bit 0 - update enable,
- bit 1 - continuous poll enable,
- bit 2 - timed poll enable
- bit 3 - spontaneous transmission enable

Exemplary declarations of ANSI variables:

```
;SNR ITEM ADDR BUS UNIT CHS CHE IO DAT1 DAT2 DTYPE DFORM DELTA STATUS
6 1 1342 2 203 0 0 I 15 15 1 1 20 13
6 2 580 2 203 1 1 OI 1 1 2 0 0 11
6 3 92 2 203 0 0 V 6 6 1 1 0 11
6 4 1500 2 203 0 0 S 15 15 0 0 10 13
6 5 91 2 203 7 7 I 2 3 0 0 0 11
6 6 92 2 203 11 11 I 1 1 0 0 0 11
```

### Configuration of Event Masks of SPA Device

Configuration of event masks of SPA devices consists in transferring the content of configuration file to a SRIO. The process uses three virtual variables:

- A.16002 - variable contains the full name of the configuration file,
- A.16003 - variable is used to begin the process of writing the configuration into a SRIO according to the selected configuration mode,
- A.16006 - variable is used to monitor the number of the currently saved declaration from the configuration file.

The configuration can be realized in one of two modes:

- **add all** - declarations from the configuration file will be added to the configuration existing in a SRIO. If the device declared in the configuration file has its own declaration in a SRIO, the declaration in the SRIO will be replaced by the declaration given in the configuration file,
- **overwrite** - all declarations existing in a SRIO will be deleted and the declarations from the configuration file will be written into the SRIO.

The process of configuring event masks of SPA devices consists in execution of the following steps:

- preparing text file with declarations of device masks,
- entering the full name of the file with declarations of device masks to the variable A.16002,
- entering the identifier of configuration of event masks into the virtual variable A.16003. The entry is treated by the driver as the request to write the configuration of device masks into a SRIO.

Adoption of the configuration write request by the driver is signaled by setting the value of 1 in the variable A.16003 and the value of 0 in the variable A.16006.

After saving the configuration of each of device masks into a SRIO the value of the variable A.16006 increases by 1.

After saving the configuration of all variables into a SRIO the variable A.16003 takes the value of 2 for 1 second time, and after that the values of variables A.16003 and A.16006 are set to 0.

#### **Format of declaration of event mask**

Declaration of the event mask has to be compatible with the format created by the program SIGTOOL of ABB, i.e. it has to be a line of text consisting of 4 elements separated by space characters and ending by the characters CR and LF. Lines beginning with # or ; are treated as comment lines.

The meaning of individual elements of the declaration is as follows:

- 1 (SNR) - number of a SRIO
- 2 (BUS) - number of a serial link in a SRIO
- 3 (UNIT) - number of a device in SPA network
- 4 (UTYPE) - device type

Exemplary declarations of device event masks:

```
; SNR BUS UNIT UTYPE  
6 2 203 2  
6 2 204 3
```

#### **Configuration of SPA Devices**

Configuration of SPA devices consists in transferring SPA commands (SPA messages) read from the configuration file to a SRIO. Commands are sent according to the order of placement in the configuration file. The process uses three virtual variables:

- A.16004 - variable contains the full name of the configuration file,
- A.16005 - variable is used to begin the process of writing the configuration to a SRIO,
- A.16006 - variable is used to monitor the number of the currently sent command of SPA from the declaration of configuration file.

The process of configuration of SPA devices consists in execution of the following steps:

- preparation of a text file with SPA commands which will be sent to a SRIO during the configuration process,
- entering the full name of the file with SPA commands into the virtual variable A.16004,
- entering an arbitrary value into the virtual variable A.16005. The entry is treated by the driver as the request of writing the SPA device configuration into a SRIO,

Adoption of the configuration write request by the driver is signaled by setting the value of 1 in the variable A.16005 and the value of 0 in the variable A.16006.

During saving each of SPA commands into a SRIO the value of the variable A.16006 increases by 1.

After saving all of the SPA commands into a SRIO the variable A.16005 takes the value of 2 for 1 second time, and after that the values of variables A.16005 and A.16006 are set to 0.

#### **Format of SPA Command**

The commands used to configure the SPA devices must use the format of SPA protocol command with the control checksum replaced by the string XX.

Lines of the configuration file starting with a semicolon are treated as a comment.

The content of exemplary file with commands used to configure SPA devices:

```
>204W0V155:255:XX  
>204W0V156:0:XX  
>204W0V157:0:XX
```

```
; STORE COMMANDS
>204WV151:1:XX
```

### **Saving SRIO Configuration into the EEPROM Memory**

To make the changes made in the SRIO configuration permanent, it is necessary to save the modified configuration into the EEPROM memory.

Saving the configuration into the EEPROM memory is initialized by performing write operation at the virtual variable A.16007 (control value does not matter).

Saving the configuration into the EEPROM memory is a long-term operation during which the driver blocks the possibility of performing any changes in the configuration of a SRIO (any trial to perform changes in the configuration ends with the error).

## **Readout of Disturbance Recorder**

The driver allows to the following operations related to disturbance recorder:

- readout of disturbance recorder directory,
- readout of a selected file from the directory of disturbance recorder,
- removal of the oldest file from the disturbance recorder directory.

All operations are performed with the use of SPA command (SPA messages) realized concurrently with routine communication with a SRIO.

Messages about the following operations start and end are displayed in Control Panel of the application.

### **Readout of Disturbance Recorder Directory**

Readout of disturbance recorder directory consists in the following operations:

- entering into the variable A.16011 the number of device (in SPA network) from which the disturbance directory has to be read,
- entering into the variable A.16012 an arbitrary value - this operation will initiate reading the disturbance directory from the device the number of which is set in the variable A.16011.

In the first phase of the operation the number of disturbances stored in the device will be read out (and written into the variable A.16013). After that, if the number of disturbances is nonzero, the names of disturbance files occurs will be read out and written successively into the variables with addresses from the range of A.16020 upwards.

It is assumed for the driver that the number of disturbance files stored in a device will not exceed 20.

### **Readout of Selected File from Disturbance Recorder**

Readout of the disturbance recorder directory consists in the following operations:

- entering into the variable A.16011 the number of device (in SPA network) from which the disturbance directory has to be read out,
- entering into the variable A.16019 the stage of compression of the file being read (acceptable values: 0, 1, 2, 3, 4, 5). By default: 0 (no compression),
- entering into the variable A.16014 the number of disturbance file which we want to read out - this operation will initiate a read file. The oldest file has the number of 1, the youngest file has the number equal to the number of disturbance files read out from the disturbance directory.

In the first stage of the operation the number of lines existing in the disturbance file (and written into the variable A.16017) is read out as well as the current line counter is reset (the variable A.16018).

In successive stages the read out of consecutive lines of the disturbance file and A.16018 variable update occur - it allows to track the progress of reading.

After the correct reading the disturbance file is saved into the directory given in the declaration of transmission channel (or into the default directory). The file is named the name created from the file name preceded by the letter 'h'. Files are saved in ASCII format. The time stamp with the time of the file writing is given to the file (not the time of creation the file in recorder).

### Removing the Oldest File from Disturbance Recorder

Removing the oldest file from the disturbance recorder directory consists in the following operations:

- entering into the variable A.16011 the number of the device (in SPA network) from which the disturbance directory has to be read out,
- entering into the variable A.16016 an arbitrary value - this operation will initiate the process of removing the oldest file from the disturbance recorder.

## Time Synchronization

The variable A. 16500 is used for synchronization. Writing an arbitrary value into this variable results in sending the current PC time into a SRIO. By default, local time is sent. It is also possible to send the time in UTC mode - to do so, create the section with the name of ASMEN channel servicing the SRIO data and write into this section the following item:

UTC\_SYNCHRONIZATION = YES

## Driver Parameterization

Srio driver parameters are declared in the Miscellaneous module, the Directly entered options tab.

The driver is parameterized with use of **CtSrio** section.

**Section name: CTSRIO**

**Option name: LOG\_FILE**

**Option value: file\_name**

Meaning: The item allows to define a file which all the diagnostic messages of the driver will be written to. The item is used for diagnostic purpose.

Default value: by default, the log file is not created.

**Section name: CTSRIO**

**Option name: LOG\_FILE\_SIZE**

**Option value: number**

Meaning: The item allows to define the size of the log file.

Option value:

*number*

- size of the log file in MB.

Default value:

10 MB.

**Section name:** *CTSRIO*

**Option name:** *LOG\_OF\_TELEGRAMS*

**Option value:** *YES/NO*

Meaning: The item allows to write the content of telegrams sent between SRIO and Asix.

Default value: NO.

**Section name:** *CTSRIO*

**Option name:** *DIAGNOSTIC\_PERIOD*

**Option value:** *number*

Meaning: For the purpose of checking the activity of SRIO the driver can periodically send diagnostic messages.

Option value:

*number*

- the period of sending diagnostic messages (in seconds).

Default value: By default, the driver sends diagnostic messages every 10 seconds.

**Section name:** *CTSRIO*

**Option name:** *EVENTS\_DEFINITIONS\_PATH*

**Option value:** *path*

Meaning: Files with definitins are placed in a common directory declared by the item EVENTS\_DEFINITIONS\_PATH.

Option value:

*path*

- name of the directory in which the driver will search files with the definition of events.

## Parameters of Driver Channel

The following items can be entered in the section which name is the name of transmission channel.

- creating log file for a single channel,
- size of the log file for a single channel,
- log of telegrams of a single channel,
- type of the time sent into a SRIO within time synchronization,
- tracking controls,
- exclusion of reading specified ANSI points,
- exclusion of registering alarms of control.

The first three items have the syntax and meaning the same as corresponding items placed in the driver section [CTSRIO], but they apply only to the channel for which they were declared.

### Time synchronization in UTC mode

**Section name:** *<channel\_name>*

**Option name:** *UTC\_SYNCHRONIZATION*

**Option value:** *YES/NO*

Meaning: The item allows you to specify the type of the time sent to a SRIO within the synchronization operation. The item should be used for each transmission channel individually. YES - sending UTC time; NO - sending local time.

Default value: By default, the driver sends local time into a SRIO.

### Tracking controls allows you to

The item allows you to enter into an operator panel the messages applies to individual stages of control operation. The messages contain a time stamp accurate to milliseconds and concern the following stages of control:

- making out the command to write to a queue the telegrams sent into a SRIO,
- sending the command to write into a SRIO,
- receiving from a SRIO the mark ACK for a write command,
- receiving from a SRIO the mark NAK for a write command,
- timeout of waiting for receiving from a SRIO the mark ACK/NAK for a sent write command,
- receiving the spontaneous telegram being the effect of performed control.

By default, the driver does not create a log during control operation.

### Exclusion of selected ANSI points from readout list

- Section name:** *<channel\_name>*
- Option name:** *ANSI\_WRITE\_ONLY*
- Option value:** *number*

Meaning: The item allows you to exclude from the readout list the certain ANSI point. That means that the driver will not send into a SRIO the queries about the value of a given ANSI point (but it will receive spontaneous telegrams concerning the given point). The item should be used for ANSI points for which the readout operation ends with the error with the code 0x40 signaled by a SRIO. For each ANSI point for which you want to turn off the readout function use a separate item.

Option value:  
*nrANSI* - number of ANSI point in decimal form.

### Exclusion of registering alarms from control

- Section name:** *<channel\_name>*
- Option name:** *WITHOUT\_CONTROL\_ALARMS*
- Option value:** *YES/NO*

Meaning: The items allows you to detach registering alarms from control (the alarms defined in the file with definitions of processing events). The item refers to all alarms of control. NO - registering alarms of control. YES - detaching registering alarms from control.

Default value: NO.

## Advanced Channel Parameters

The parameters of a channel are declared in the Advanced tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:  
YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the Advanced 2 tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
 YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.107 SRTP - Driver of SRTP Protocol

### Driver Use

The SRTP driver is designed for data exchange between the Asix system and the GE Fanuc Automation PLCs of VersaMax Nano/Micro, WersaMax and 90 series, by means of SRTP (Service Request Transfer Protocol) using an Ethernet network with the TCP/IP protocol.

The data exchange with the VersaMax Nano/Micro PLCs is realized with the aid of the IC200SET001 converter.

The communication with the WersaMax and 90 PLCs demands the IC693CMM321 converter.

Parameterization of SRTP driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the SRTP driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel

*Driver:* SRTP

**SRTP/Channel parameters** tab:

*Channel parameters:*

*IP address...* - IP address assigned to the IC693CMM321 communication module;

*Port number* - optional port number through which the connection with the IC693CMM321 communication module is executed (by default, 18245).

For each IC693CMM321 module a separate transmission channel declaration is required.

### Types of Process Variables

In the driver the following types of process variables are defined:

<b>I</b>	- Discrete Input (%I) in BIT mode,
<b>IB</b>	- Discrete Input (%I) in BYTE mode,
<b>IW</b>	- Discrete Input (%I) in WORD mode,
<b>Q</b>	- Discrete Output (%Q) in BIT mode,
<b>QB</b>	- Discrete Output (%Q) in BYTE mode,
<b>QW</b>	- Discrete Output (%Q) in WORD mode,
<b>M</b>	- Discrete Internal (%M) in BIT mode,
<b>MB</b>	- Discrete Internal (%M) in BYTE mode,
<b>MW</b>	- Discrete Internal (%M) in WORD mode,
<b>G</b>	- Genius Global Data (%G) in BIT mode,
<b>GB</b>	- Genius Global Data (%G) in BYTE mode,
<b>GW</b>	- Genius Global Data (%G) in WORD mode,

<b>AI</b>	- Analog Input (%AI) in WORD mode,
<b>AQ</b>	- Analog Output (%AQ) in WORD mode,
<b>R</b>	- Register (%R) treated as WORD,
<b>RL</b>	- two successive Registers (%R) treated as DWORD,
<b>RF</b>	- two successive Registers (%R) treated as FLOAT,
<b>TD</b>	- current time and date of the controller.

Values of **I**, **IB**, **IW** and **AI** type variables may be only read whereas values of the other variables may be read and written.

The following types of variables (system, temporary) are NOT supported:

- Discrete SA (%SA),
- Discrete SB (%SB),
- Discrete SC (%SC),
- Discrete S (%S),
- Discrete Temporary (%T).

## Addressing the Process Variables

The syntax of symbolic address which is used for variables belonging to the S RTP driver channel is as follows:

*<Type><Index>*

where:

<i>Type</i>	- name of the variable type;
<i>Index</i>	- variable address within the type <i>Type</i> of the variable.

For discrete variables (**I**, **IB**, **IW**, **Q**, **QB**, **QW**, **M**, **MB**, **MW**, **G**, **GB**, **GW**) *Index* indicates the number of bit from which the range of bits assigned to the variable begins. Depending on the mode of accessing the discrete variables, *Index* may assume the following values:

- a/ for BIT mode - any value beginning from 1;
- b/ for BYTE mode - values 1, 9, 17, etc. (numbers of the first bit of successive bytes);
- c/ for WORD mode - values 1, 17, 33, etc. (numbers of the first bit of successive words).

### EXAMPLE

Examples of variable declarations:

JJ_1, %R1 and %R2 as FLOAT,	RF1,	CHANNEL,1,1,NOTHING_FP
JJ_3, %R3 and %R4 as DWORD,	RL3,	CHANNEL,1,1,NOTHING_DW
JJ_5, %R5 as WORD,	R5,	CHANNEL,1,1,NOTHING
JJ_31, single bit %M1,	M1,	CHANNEL,1,1,NOTHING
JJ_32, bits %Q9 - %Q16 as one byte,	QB9,	CHANNEL,1,1,NOTHING_BYTE
JJ_33, bits %I17 - %I32 as one word,	IW17,	CHANNEL,1,1,NOTHING
JJ_40, writing date and time to PLC,	TD,	CHANNEL,8,15,NOTHING_BYTE

## Date and Time Synchronization with the Controller

A mechanism of date and time synchronization between the Asix system and GE Fanuc PLCs is built in the driver. The synchronization is performed for each transmission channel separately by means of option declared in:

Architect > *Fields and Computers* > *Current data* > channel parameter tab > *Standard* tab

**TIME\_SYNCHRONIZATION**

Parameters:

*variable* - name of the ASMEN variable, belonging to the channel channel, used for date and time synchronization.

The date and time synchronization consists in a cyclic writing to the controller an actual frame containing the current date and time of Asix. The frame is written by use of a built-in SRTP protocol function for writing date and time according to the frequency assigned to *variable*. The variable type must be the **TD** type (support of date and time), number of elements assigned to *variable* must have a size of 8 (size of date and time frame). The function NOTHING\_BYTE must be used as the conversion function.

The declaration of the variable SYNCHRO may be found in the file *SRTP.DAT* and is as follows:

SYNCHRO, date and time synchronization, TD, CHANNEL, 8, 60, NOTHING\_BYTE

## Driver Configuration

The driver configuration is defined in the *Current Data* module, in the channel operating according to SRTP driver.

**Log file**

Meaning - allows to define a file to which all diagnostic driver messages and information about the content of telegrams received by the driver will be written. If the item LOG\_FILE does not define the full path, then the log file will be created in the current directory. The log file should be used only in the stage of the Asix start-up.

Default value - log file is not created.

Defining - manual.

**Log file size**

Meaning - declares a log file size in MB.

Default value - 1MB.

Defining - manual.

**Log of telegrams**

Meaning - allows to write, to the log file (declared by use of the item LOG\_FILE), contents of telegrams sent/received by the driver. Telegrams content write should be used only in the stage of the Asix system start-up.

Default value - NO.

Defining - manual.

**Start-up time**

Meaning - allows declaring a time (in seconds) provided for establishing the network connections with all the IC693CMM321 modules on the stage of the Asix start-up.

Default value - 3s.

Defining - manual.

**Transmission**

Meaning - for each module IC693CMM321 it is determined maximum time between sending requests and receiving responses (receive timeout). After this time the connection will be dropped (and established again).

Default value - 5s.

Parameter

*IP address number* - IP address of the module IC693CMM321,  
- value of timeout (in seconds)

## MaxON Redundancy Parameterization

1. Two Asmen channels should be created and assigned to ID\_A and ID\_B PLCs that make MaxON redundancy.
2. The channel directed to ID\_A PLC is dedicated to variables of Asix application, the channel directed to ID\_B is used by SRTP driver (and should not be used by the application).
3. Next, the parameter **MaxOn redundancy** should be declared in the SRTP driver parameters:

Architekt > *Fields and Computers* > *Current data* > declaration of the transmission channel using the SRTP driver > *SRTP tab / Driver parameters 2*

where:

*ID\_A* - name of the channel directed to ID\_A PLC;  
*ID\_B* - name of the channel directed to ID\_B PLC.

Principle of operation of SRTP driver during MaxON redundancy handling consists in selection of the right channel (*ID\_A* or *ID\_B*) for servicing application variables. The selection strategy bases on current state of status flags read form *ID\_A* and *ID\_B* channels.

There are the meanings for symbolic addresses of system variables:

XN1 - status ID\_A  
XN2 - status ID\_B  
XN3 - status RUN  
XN4 - status MASTER  
XN5 - status SYNC

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
YES / NO

The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.108 TALAS - Driver of TALAS Analyzer Protocol

### Driver Use

The TALAS driver is used for data exchange between TALAS emission computers and the Asix system by use of serial interfaces. The driver was created in order to operate with devices which have a firm software v 2.3 (007)22 installed.

Parameterization of TALAS driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the TALAS driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: TALAS

**TALAS** tab:

Channel parameters:

*COMn*, *baud*

where:

*COMn* - number of the serial port to which the TALAS emission computer is connected;

*baud* - transmission speed expressed in bauds.

The TALAS driver is loaded as a DLL automatically.

### Addressing the Process Variables

The arrays of data of following categories are read from a TALAS computer:

HM	- current half-minute data,
PI	- current partial integrals,
AI	- current values of half-hour integrals,
HI	- half-hour integrals for a current day,
DIW	- daily distribution of half-hour integrals from a previous day,
YIW	- annual distribution of half-hour integrals from a previous day,
DMV	- distribution of an average daily value from a previous day.

From the arrays above the TALAS driver retrieves process data of the following types:

ATM	- current time (of half-minute data) of the TALAS computer, transferred as a number of seconds in DWORD format;
VAL	- variable value in float format;

- STA - variable status in word format;
- OTM - *work time* of variable counted in tenths of second, transferred in long format;
- ITM - integration period in seconds in short format;
- DCL - classes of classification in word format;
- DCT - classification table for the variable, contents of individual classes in word format, maximal table size - 34 elements,
- IVT - table of half-hour integrals for the variable, element is a structure containing time, value (float) and status (word); maximal table size - 48 elements.

Allowable sets: type-data category together with waited format of an address string are given in the table below, where *nn* signifies the successive number (not identifying) of the variable in the variable list of the TALAS computer in the range <1..128> for HM, PI and AI categories and <1..64> for HI, DIW, YIW and DMV categories (see: the table below).

**Table 50. Allowable Sets: Type-Data Category Together with Waited Format of an Address String for the TALAS Driver.**

Address state	Contents	Notes
HM.ATM	dword	actual time of TALAS station in [s]
HM.VAL.nn PI.VAL.nn AI.VAL.nn	float	variable value
HM.STA.nn PI.STA.nn AI.STA.nn	word	variable status
HI.VAL.nn	float	half-hour integral of variable
HI.STA.nn	word	status of the above-mentioned integral
HI.IVT.nn	struct { struct xtime time; float value; word status; }	table; size 48
DIW.VAL.nn YIW.VAL.nn DMV.VAL.nn	float	average value
DIW.OTM.nn YIW.OTM.nn DMV.OTM.nn	long	working time of variable in [0.1s]
DIW.ITM.nn YIW.ITM.nn DMV.TTM.nn	short	integration period in [s]
DIW.DCL.nn.mm YIW.DCL.nn.mm DMV.DCL.nn.mm	word	mm: class number <1..34>
DIW.DCT.nn YIW.DCT.nn	word	table; size 34

DMV.DCT.nn		
------------	--	--

## Driver Configuration

TALAS driver parameters are declared in the *Miscellaneous* module, the *Directly entered options* tab.

The driver is parameterized with use of **TALAS** section.

**Section name: TALAS**

**Option name: baud (or bod or bps)**

**Option value: number**

Meaning - the item is used to declare a transmission speed. The item value has priority over a transmission speed given in the definition of the logical channel.

Default value - by default, the transmission speed is assumed to be equal to 9600 Bd.

Defining - manual.

Parameters:  
     number - transmission speed in bauds.

**Section name: TALAS**

**Option name: parity**

**Option value: check\_type**

Meaning - the item used to declare a method of the parity check.

Default value - by default, it is assumed the even parity check.

Defining - manual.

Parameters:  
     *check\_type* - identifier of the way of parity check:  
         n - no parity bit,  
         o - odd parity check,  
         e - even parity check,  
         m - mark,  
         s - space.

**Section name: TALAS**

**Option name: stop**

**Option value: number**

Meaning - the item is used to declare a number of stop bits.

Default value - by default, it is assumed 1 stop bit.

Defining - manual.

Parameters:  
     *number* - number of stop bits: 1 or 2.

**Section name: TALAS**

**Option name: word (or word\_length)**

**Option value: number**

Meaning - the word item is used to declare a number of bits in a transmitted character.

Default value - by default, it is assumed that the transmitted character has 8 bits.

Defining - manual.

Parameters:  
     *number* - number of bits in a character (from 5 to 8).

**Section name:** TALAS  
 **Option name:** timeout  
 **Option value:** number  
Meaning - the item is used to declare a waiting time for an answer from the TALAS computer.  
Default value - by default, it is assumed 10 seconds.  
Defining - manual.  
Parameters:  
    *number* - waiting time for an answer in seconds.

**Section name:** TALAS  
 **Option name:** KM\_Interval (or Interval)  
 **Option value:** number  
Meaning - the item is used to declare a time interval between readings of values of short-time averages and partial integrals from the TALAS computer.  
Default value - by default, it is assumed to be 30 seconds.  
Defining - manual.  
Parameters:  
    *number* - time interval in seconds.

**Section name:** TALAS  
 **Option name:** IW\_Interval  
 **Option value:** number  
Meaning - the item used to declare a time interval between readings of values of half-hour integrals and actual integrals from the TALAS computer. By default it is assumed to be 30 minutes.  
Default value - by default, it is assumed to be 30 seconds.  
Defining - manual.  
Parameters:  
    *number* - time interval in minutes.

**Section name:** TALAS  
 **Option name:** FD\_Interval  
 **Option value:** number  
Meaning - the item used to declare a time interval between readings of values of distributions from the TALAS computer.  
Default value - by default, it is assumed to be 60 minutes.  
Defining - manual.  
Parameters:  
    *number* - time interval in minutes.

**Section name:** TALAS  
 **Option name:** log  
 **Option value:** name  
Meaning - the item is used to declare a file to which a diagnostic information of the TALAS driver are written. The item is dedicated to test purposes.  
Default value - by default, the file is not created.  
Defining - manual.  
Parameters:  
    *name* - file name.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

- Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.
- Option value:  
YES / NO
- computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

- Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.
- Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

- Purpose - with this option deployed, local control in a channel will be disabled.
- Option value:  
YES / NO
- The default value - NO.

### **Data Validity Period**

- Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.
- Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.
- The default value - by default, no time for data validity is set.

## 1.109 TwinCAT - Driver of ADS Protocol for TwinCAT System

### Driver Use

The TwinCAT driver is designed to exchange data between the Asix system and the TwinCAT system of Beckhoff Industrie Elektronik. The communication between systems is realized through Ethernet in two modes, depending on the firmware a Beckhoff controller is delivered with.

There are two versions of the TwinCAT driver:

1. CtTwinCat.dll - uses the TcAdsDll library provided by Beckhoff.
2. CtTwinCatTcpi.dll - uses ADS/AMS over TCPIP interfaces (without Beckhoff libraries).

CtTwinCAT services the following devices:

- controllers of the CX1000 series;
- TwinCAT PLC (PC based control system);
- controllers of the BC9000 series;
- controllers of the BX9000 series.

Parameterization of TwinCAT driver is performed with the use of Architect module.

### Declaration of Transmission Channel

The TwinCAT driver is loaded by the universal Asix driver - UNIDRIVER.

**The declaration of transmission channel using CtTwinCat.dll has the following form:**

Declaration of the transmission channel using the TwinCAT driver requires a channel with the following parameters to be added to the Current data module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* TwinCAT

**TwinCAT/Channel parameters** tab:

<i>Number of the controller port</i>	- port number of the TwinCAT system router, for example 801 (RTS1), 811 (RTS2);
<i>AMS Net Id of controller</i>	- AMS address (4 first elements) of the TwinCAT system router;

**TwinCAT/Channel parameters 2** tab:

<i>Timeout</i>	- timeout (in milliseconds) for performing an ADS operation in a given channel;
<i>Time synchronization</i>	- address in controller to which time frame will be sent;
<i>Time synchronization period</i>	- time period (in seconds) according to which time frame will be sent to a controller;
<i>Service type</i>	- service type;
<i>Number of STOP state alarm</i>	- it enables generation of alarm when PLC changes its state to STOP.
<i>Version</i>	- version of the software in the controller.

\*\*\*

**The declaration of transmission channel using CtTwinCatTcpip.dll has the following form:**

Declaration of the transmission channel using the TwinCatTcpip driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel  
*Driver:* TwinCatTcpip

**TwinCatTcpip/ Channel parameters** tab:

*AMS Net Id of the asix system (4 first elements)*  
*Controller IP address*  
*AMS Net Id of controller*  
*Number of the controller port*

- AmsNetId address that identifies together with an Asix computer IP a connection Asix <-> controller; both addresses must have their equivalents in the table of controller project routs to make data exchange possible; it is passed that AmsNetId is the same as an Asix computer IP (for simplification of connection parameterization);

*IP address of the controller*  
*Buffer size*  
*UTC synchronization*

The date & time frame structure is given below:

```
struct dateTime
{
    word wYear;
    word wMonth;
    word wDayOfWeek;
    word wDay;
    word wHour;
    word wMinute;
    word wSecond;
    word wMilliseconds;
    word wSynchr;
};
```

where:

wYear	- year	1970 ~ 2106
wMonth	- month	1 ~ 12 (January = 1, February = 2, etc)
wDayOfWeek	- day of week	0 ~ 6 (Sunday = 0, Monday = 1, itd.)
wDay	- day of month	1 ~ 31;
wHour	- hour	0 ~ 23;
wMinute	- minute	0 ~ 59;
wSecond	- second	0 ~ 59;
wMilliseconds	- millisecond	0 ~ 999;
wSynchr	- synchro marker	1 is written with the new data & time frame

*Buffer size* - maximal size (in bytes) of the buffer; this option overwrites value of similar option declared among the TwinCATtcpip driver parameters;

*UTC synchronization* - time synchronization in UTC mode.

**TwinCatTcpip/ Channel parameters 2 tab:**

<i>Timeout</i>	- timeout (in milliseconds) for performing an ADS operation in a given channel;
<i>Time synchronization</i>	- address in controller to which time frame will be sent;
<i>Time synchronization period</i>	- time period (in seconds) according to which time frame will be sent to a controller;
<i>Service type</i>	- service type;
<i>Number of STOP state alarm</i>	- it enables generation of alarm when PLC changes its state to STOP.
<i>Version</i>	- version of the software in the controller.

**TwinCatTcpip/ Channel parameters - history tab:**

<i>Group number</i>	- number of the group (in HEX) in which the historical data are stored;
<i>Historical data offset</i>	- offset (in HEX) in the group from which the historical data starts
<i>Number of variables</i>	- number of historical variables
<i>Query offset</i>	- offset (in HEX) in the group from which the buffer for the commands of the Asix system starts
<i>Reply offset</i>	- offset (in HEX) in the group from which the reply buffer for the commands of the Asix system starts
<i>Query delay</i>	- expected time of execution

**TwinCatTcpip/ Channel parameters - log tab:**

**Log file**

Meaning - the item allows to define a file to which all diagnostic messages of the driver and the information about ADS requests will be written. If the item doesn't define a full path, the log file will be created in the current directory. The log file should be used only during the Asix system start-up.

Default value - by default, the log file is not created.

Parametr:

*file\_name* - name of the log file.

**Log file size**

Meaning - the item allows to define the size of the log file.

Default value - by default, the item assumes that the log file has a size of 1 MB.

Parameter:  
*number* - size of the log file in MB.

**Log of telegrams**

Meaning - the item allows to write the info about performed ADS requests to the log file. This option should be used only during the Asix system start-up.

Default value - by default, the driver does not write the info about ADS requests to the log file.

**Log of telegram contents**

Meaning - the item allows to write the contents of telegram to the log file.

Default value - by default, the driver does not write the info to the log file.

## Types of Process Variable Addresses

Two types of addresses are allowed in the TwinCAT system:

- **symbolic** - variable symbolic address. The form is as follows:

*[<text1>].<text2>*

where:

*text1, text2* - ASCII string.

Symbolic addresses of all variables of the project realized in the TwinCAT system are placed in the project directory, in an XML file with the extension \*.tpy - addresses are entered in sections identified by the key word *<Symbol>*.

### EXAMPLE

.engine  
 MAIN.devUP

- *direct* - address in the form **Group:Offset**. Both **Group** and **Offset** are declared in HEX format and delimited with a colon (:).

### EXAMPLE

F030:03EA

### EXAMPLE

Examples of process variable declarations.

JJ\_00, variable valFloat (REAL), .valFloat, PLC1, 1, 1, NOTHING\_FP  
 JJ\_01, variable valWord (WORD), .valWord, PLC1, 1, 1, NOTHING  
 JJ\_02, variable MAIN.engine (BOOL), MAIN.engine, PLC1, 1, 1, NOTHING\_BYTE  
 JJ\_03, variable with address F030:5 (BYTE), F030:5, PLC1, 1, 1, NOTHING\_BYTE

## Variable Type

For variables with a symbolic address the raw type of a variable is taken from the variable's type declared in the TwinCAT system. This type is converted into the type required by the user.

For variables with a direct address the raw type of a variable is the real type of conversion function defined for the variable. Based on this type:

- the appropriate number of bytes are read/written from/into the address specified by **Group:Offset**;
- the conversion of read bytes, according to the type defined for the variable, is performed.

## Data & Time Stamp

Data read from the driver is stamped with the PC local date & time, evaluated at the moment when the ADS request is completed.

## Driver Configuration

The driver configuration is defined in the *Current Data* module, in the channel operating according to TwinCAT / TwinCatTcpip driver.

### **Log file**

Meaning

- the item allows to define a file to which all diagnostic messages of the driver and the information about ADS requests will be written. If the item doesn't define a full path, the log file will be created in the current directory. The log file should be used only during the Asix system start-up.

Default value

- by default, the log file is not created.

Parameter:

*file\_name*

- name of the log file.

### **Log file size**

Meaning

- the item allows to define the size of the log file.

Default value

- by default, the item assumes that the log file has a size of 1 MB.

Parameter:

*number*

- size of the log file in MB.

### **Log of telegrams**

Meaning

- the item allows to write the info about performed ADS requests to the log file. This option should be used only during the Asix system start-up.

Default value

- by default, the driver does not write the info about ADS requests to the log file.

### **Log of telegram contents**

Meaning

- the item allows to write the contents of telegram to the log file.

Default value

- by default, the driver does not write the info to the log file.

**Buffer size**

Meaning	- the item allows to define the size of the buffer used to perform ADS read/write requests.
Default value	- the default size of the buffer is 1800 bytes.
Parameter:	
<i>number</i>	- size of the buffer in bytes.

## Variable Quick Refresh Function

By default, variables are refreshed by the Asmen - current data module with a frequency of at least 1 second. The function of variable quick refresh is realized by the driver TwinCATtcpip from the version 1.3.5 and it enables to refresh variables with maximum line speed (it does not affect the backup process - variables are archived with a frequency of at least 1 second). The variables being subject to quick refresh function have the attribute *FastRefresh* with the value of 1 declared in variable definition database.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

 **Remote Write Access**

Purpose	- this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.
Option value:	
YES / NO	
<i>computer name</i>	- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).
The default value	- NO - by default, only local computers can modify variable values.

 **Remote Read Limitation**

Purpose	- with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.
Option value:	
<i>computer name</i>	- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
The default value	- by default, all computers can read variables.

 **Redundancy**

Purpose	- with this option searches a redundant channel when connection with a controller is broken in local channel.
---------	---

## Communication Drivers

- Option value:  
*computer name* - list including the names of network computers which will be permitted to search a redundant channel.
- The default value - by default, when option is enabled, the channel is searched on all computers.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value - NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.110 UniDriver

### *Driver parameters - UniDriver* tab:

**NOTICE:** Options defined on this tab for any channel that has this tab are valid for all the channels that have this tab.

**Path to the directory in which the log file will be created**

Meaning - the log file is placed in the defined directory.

Default value - by default, the log file will be created in the Asix directory.

Parameter:

*record\_track* - path to the directory in which the log file will be created.

**Traced variables**

Meaning - for each variable, which name is on the list, the information on its value, quality and stamp will be written to the log (during the variable processing).

Default value - lack.

Parameter:

*variable\_list* - list of the variable names in the Asix system, delimited with commas.

## 1.111 Vantage - Driver for Data Exchange with Weather Stations of Vantage Pro Family

### Driver Use

The Vantage driver is used for readout of current data from weather stations of Vantage Pro family developed by Davis Instruments Corp., USA. Communication is carried out via serial interface.

The driver implements only 'LPS' command described in 'Vantage Pro, Vantage Pro2 and Vantage Vue Serial Communication Reference Manual Rev. 2.6.1 (03/29/2013)' - Davis Instruments Part Number: 07395.801.

### Declaration of Transmission Channel

Declaration of the transmission channel operating according to the Vantage driver protocol requires adding a channel with the following parameters to the Current data module:

**Standard** tab:

*Name*:: logical name of the transmission channel  
*Driver*: Vantage

**Vantage** tab:

*Channel Parameters*:

*Port*=number; *BaudRate* =number; *Timeout*=number; *Period*=number;  
*AlarmNr*=number

where:

<i>Port</i>	- serial port number;
<i>BaudRate</i>	- serial port baud rate; by default 19200;
<i>Timeout</i>	- timeout for a response from the station (in milliseconds); by default 2000 (ms);
<i>Period</i>	- interval (in milliseconds) between consecutive readouts from the station; by default 2 (seconds);
<i>AlarmNr</i>	- number of Asix alarm in the absence of communication with the station; by default the driver does not generate an alarm in the absence of communication.

#### EXAMPLE

Below follows an example of a declaration of the KANAL channel, in which communication is carried out via COM4 port, baud rate is 19200 Bd, the station

is polled every 5 seconds, alarm of number 100 is returned in the absence of communication.

*Port=4; BaudRate=19200; Period=5; AlarmNr=100*

## Declaration of Variables

The symbolic address of a variable has the following syntax:

**C** - communication status with the station (0 - no communication, 1 - o.k.)

**V<offset>** - value of the parameter transferred in the offset position in the response buffer as a response to the command LPS (LOOP2 Packet Format - page 25/26 of the documentation:

'Vantage Pro, Vantage Pro2 and Vantage Vue Serial Communication Reference Manual Rev. 2.6.1 (03/29/2013)' - Davis Instruments Part Number: 07395.801.)

Note:

The driver ver. 1.0 supports only the following parameters:

offset 3 // Pressure, 3 hour trend,  
 offset 7 // Pressure,  
 offset 12 // External temperature,  
 offset 14 // WIND, Instantaneous wind speed,  
 offset 18 // WIND, Average wind speed in last 10 minutes,  
 offset 16 // WIND, Direction,  
 offset 22 // WIND, Speed of 10 minute wind gusts,  
 offset 24 // WIND, Direction of 10 minute wind gusts,  
 offset 30 // Dew point,  
 offset 33 // Outside humidity,  
 offset 52 // Rainfall in 15 minutes.

### EXAMPLE

Examples of variable declarations:

*Name: X1*  
*Description: communication status*  
*Address: C*  
*Channel: CHANNEL1*  
*Elements Count: 1*  
*Sample Rate: 1*  
*Conversion Function: NOTHING*

*Name: X2*  
*Description: pressure\_3\_hours*  
*Address: V3*  
*Channel: CHANNEL1*  
*Elements Count: 1*  
*Sample Rate: 1*  
*Conversion Function: NOTHING*

*Name: X3*

*Description:* pressure  
*Address:* V7  
*Channel:* CHANNEL1  
*Elements Count:* 1  
*Sample Rate:* 1  
*Funkcja przeliczająca:* NOTHING\_FP

*Name:* X4  
*Description:* outside temperature  
*Address:* V12  
*Channel:* CHANNEL1  
*Elements Count:* 1  
*Sample Rate:* 1  
*Conversion Function:* NOTHING\_FP

## Driver Parameters

The driver is configured using a separate section [CTVANTAGE]: In this section it is possible to include declaration items:

- creating a log file,
- the log file size,
- log of telegrams,

The names of the items related with the log file refer to the convention used in other ASMEN drivers.

**Section name:** CTVANTAGE

**Option name:** LOG\_FILE

**Option value:** *file\_name*

Meaning: the text log file to which driver status messages are stored is used for diagnostic purposes.

The default value: By default, the log file is not created.

**Section name:** CTVANTAGE

**Option name:** LOG\_FILE\_SIZE

**Option value:** *number*

Meaning: this item is used to determine the size of the log file defined using the LOG\_FILE item.

The default value: by default the log file size is 10 MB.

Parameter:

*number* - the size of the log file in MB.

**Section name:** CTVANTAGE

**Option name:** LOG\_OF\_TELEGRAMS

**Option value:** YES/NO

Meaning: The item allows writing the contents of messages communicated between the driver and the controllers to the log file (declared using the LOG\_FILE item) The item in question should only be used during the ASIX system start-up.

The default value: NO.

## Channel Parameters

The parameters specific for a transmission channel can be defined using an individual section having the name of a given channel. In this section it is possible to include declaration items:

- creating a log file,
- the log file size,
- log of telegrams,
- response character timeout.

**Section name:** *<channel\_name>*

**Option name:** LOG\_FILE

**Option value:** *file\_name*

Meaning: The text log file to which the messages related to transmission in a given channel are stored is used for diagnostic purposes.

The default value: By default, the messages related to a channel are stored to the driver log file.

**Section name:** *<channel\_name>*

**Option name:** LOG\_FILE\_SIZE

**Option value:** *number*

Meaning: this item is used to determine the size of the log file defined using the LOG\_FILE item for the channel.

The default value: by default the log file size is 10 MB.

Parameter:

*number* - the size of the log file in MB.

**Section name:** *<channel\_name>*

**Option name:** LOG\_OF\_TELEGRAMS

**Option value:** YES/NO

Meaning: the option LOG\_TELEGRAMOW allows writing the contents of messages communicated in this channel to the log file (declared using the LOG\_FILE item for the channel). The item value overrides. The item in question should only be used during the ASIX system start-up.

The default value: NO.

**Section name:** *<channel\_name>*

**Option name:** CHAR\_TIMEOUT

**Option value:** YES/NO

Meaning: the option CHAR\_TIMEOUT specifies the maximum time (in milliseconds) between response characters from the device.

The default value: the default response character timeout value is 75 milliseconds.

Parameter:

*number* - time out in milliseconds.

### EXAMPLE

Driver parameters:

```
[CTVANTAGE]
LOG_FILE=d:\tmp\CtVantage\stacja.log
LOG_FILE_SIZE=30
```

Channel parameters:

```
[CHANNEL1]
```

LOG\_FILE=d:\tmp\CtVantage\kanal\_1.log  
LOG\_FILE\_SIZE=20  
LOG\_OF\_TELEGRAMS=YES

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value

- NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value

- by default, no time for data validity is set.

## 1.112 Wago - Wago Controllers Protocol Driver

### Driver Use

Wago protocol driver is used to exchange data between the Asix system and Wago controllers. Data exchange is via Ethernet in UDP mode based on the network variables concept described in the "Network Functionality in CoDeSys V2.3" - Document Version 1.7.

The driver design is based on the following assumptions:

a/ a separate set of network variables is created for each controller to communicate with the CtWago driver. The set has the following attributes:

- 'Pack variables'
- 'Read'
- 'Write', 'Answer bootup requests'
- 'Transmit on change'

b/ a set of 'network variables' is stored in a file whose name is a parameter in the CtWago driver transmission channel declaration,

c/ a separate CtWago driver transmission channel is defined for each controller,

d/ driver updates the process variables values by capturing the 'StateFree' frames transmitted by the controllers due to a change in the value of network variables, or as a result of controller activation by the driver using the 'BootUpRequest' frames.

e/ driver performs write operations by transmitting the 'StateFree' frame to the controller. It is assumed that the controller responds by transmitting a 'StateFree' frame, which is the result of setting the "Transmit on change" parameter.

### Transmission Channel Declaration

Declaration of the transmission channel operating according to the CtWago driver protocol requires adding a channel with the following parameters to the *Current data* module:

**Standard** tab:

*Name:* logical name of the transmission channel

*Driver:* CtWago

**Wago/Channel parameters** tab:

**IP port number in the controller** - IP port number in the controller. Default 1202,  
**SYM\_XML file name generated for the project CoDeSys** - SYM\_XML file name generated for CoDeSys project implemented by the controller. The driver imports from this file the definitions of all types used in the CoDeSys project. It is important that the SYM\_XML file is generated taking into account the network variables, otherwise the types used only in the network variables file will not be accessible the SYM\_XML file;

**The name of the file with network variable declarations of the controller** - the name of the file holding the controller network variable declarations (in the CoDeSys project implemented by the driver);

**Identifier (COB-ID) assigned to network variables** - identifier (COB-ID) assigned to network variables (in the CoDeSys project implemented by the controller);

***IP Address to which Bootup Request commands are sent (address IP of the controller);***

***Alarm Number of the lack of communication with the controller*** - alarm number in the absence of communication with the controller.

***Wago/Channel parameters 2*** tab:

Options related to time synchronization with the controller:

***Time synchronization*** - the variable name in the controller, to which time frame will be sent,

***Synchronization interval*** - interval (in seconds) at which time frame will be sent to the controller. The default is 60 seconds,

***UTC Time Sync Mode*** - if this item is set to YES, time is synchronized with UTC time. If the item is set to NO, it is time synchronized with local time. By default, the items is set to NO.

Other parameters:

***Computer network card address*** - this option is used to specify the network adapter address used by the driver to communicate with the controller. If the item is not declared, then the driver will use the network adapter provided by the system, which can prevent communication with the controller. NIC address in IPv4 notation.

***Log file*** - the text log file is for diagnostic purposes; the log records the datagrams sent and received from the controller.

***Log file size*** - this option is used to determine the size of the log file defined using the *Log file* option

**EXAMPLE**

*Channel/Name: WAGO211*

*The log file: d:\logi\wago211.log*

*Log file size: 20*

*The computer's network card address: 10.10.105.2*

Time frame structure transmitted to the PLC has the following format:

```
struct dateTime
{
  word wYear;
  word wMonth;
  word wDayOfWeek;
  word wDay;
  word wHour;
  word wMinute;
  word wSecond;
  word wMilliseconds;
  word wSynchr;
};
```

where:

wYear - year 1970 ~ 2106  
 wMonth - Month 1 ~ 12 (January = 1, February = 2, etc.)  
 wDayOfWeek - day of the week 0 ~ 6 (Sunday = 0, Monday = 1, etc.)  
 wDay - day 1 ~ 31;  
 wHour - time 0 ~ 23;  
 wMinute - minute 0 ~ 59;  
 wSecond - second 0 ~ 59;  
 wMilliseconds - millisecond 0 ~ 999;  
 wSynchr - synchronization, 1 stored when time frame is transmitted to the controller

## EXAMPLE

Example of transmission channel declaration:

*Channel/Name: WAGO211*  
*IP port number in the controller: 1202*  
*address IP of the controller: 255.255.255.255*  
*SYM\_XML file name generated for the project CoDeSys: d:\Proj211.sym\_xml*  
*The name of the file with network variable declarations of the controller: d:\Zm211.exp*  
*Identifier (COB-ID) assigned to network variables: 211*  
*IP Address to which Bootup Request commands are sent (address IP of the controller):*  
 10.10.105.211  
*Alarm Number of the lack of communication with the controller: 1211*  
*Time synchronization: Glob211ArrDt[1]*  
*Synchronization interval: 15*  
*UTC Time Sync Mode: YES*

## Declaration of Variables

The driver accepts the process variables, whose symbolic addresses are consistent with the symbolic address of CoDeSys project variables read from the network variable definition file (the 'File' parameter in the transmission channel declaration). The symbolic address may refer to variables that are simple types, elements of structures or arrays.

Note:

The network variable definition file:

- there can be only one VAR\_GLOBAL ... VAR\_END section,
- sections other than VAR\_GLOBAL ... VAR\_END are not allowed,
- Comments may be included only as single, separate lines.

The VAR\_GLOBAL opening section can include the PERSISTENT and RETAIN options.

The driver accepts CoDeSys simple types with the following names:

BYTE  
 WORD  
 DWORD  
 UINT16  
 SINT  
 USINT  
 INT  
 INT16  
 DINT

REAL  
LREAL  
STRING  
STRING()  
TIME  
TOD  
DATE  
DT

**Note:**

The driver returns the values of TIME, TOD, DATE and DT type variables as DWORD.

The driver imports the definitions of all types used in the CoDeSys project from the SYM\_XML file. This file should be created in the CoDeSys project using the 'Project >Options->SymbolConfiguration->DumpXMLsymbolTable' menu item. The SYM\_XML file name must be specified in the transmission channel declaration as the "Sym" parameter.

**EXAMPLE**

Examples of process variables declarations based on simple types:

```
JJ_1, GlobalWord, Glob211Word, WAGO211, 1, 1, NOTHING
JJ_2, GlobalDw, Glob211Dw, WAGO211, 1, 1, NOTHING _DW
JJ_3, GlobalReal, Glob211Real, WAGO211, 1, 1, NOTHING _FP
JJ_4, GlobalInt, Glob211Int, WAGO211, 1, 1, NOTHING _INT
JJ_9, GlobalByte, Glob211Byte, WAGO211, 1, 1, NOTHING _BYTE
JJ_10, GlobalString, Glob211Str, WAGO211, 17, 1, NOTHING _TEXT
JJ_11, GlobalStringArr2, Glob211StrArr[2], WAGO211, 21, 1, NOTHING _TEXT
JJ_12, GlobalBool, Glob211Bool, WAGO211, 1, 1, NOTHING _BYTE
# Array used to synchronize the time - 9 elements, the WORD type
JJ_13, GlobalDtArray, Glob211ArrDt[1], WAGO211, 9, 1, NOTHING
# The first element of the array used for time synchronization
JJ_14, GlobalDtArray0, Glob211ArrDt[1], WAGO211, 1, 1, NOTHING
```

Network variable declarations file content, which defines the translation process variables:

```
VAR_GLOBAL
Glob211Bool: BOOL;
Glob211Word: WORD;
Glob211Str: STRING(16):='ala ma kota';
Glob211StrArr: ARRAY [1..4] OF STRING(20);
Glob211Cnt: INT;
Glob211Byte: BYTE;
Glob211Dw: DWORD;
Glob211Int: INT;
Glob211Real: REAL;
Glob211ArrDt: ARRAY [1..9] OF WORD;
END_VAR
```

**EXAMPLE**

Examples of process variables declarations for the structures:

```
TYPE stLevel3:
STRUCT
parL3_1: INT;
parL3_2: ARRAY [1..5] OF WORD;
END_STRUCT
END_TYPE

TYPE stLevel2:
STRUCT
parL2_1: INT;
parL2_2: ARRAY [1..5] OF stLevel3;
END_STRUCT
```

```
END_TYPE
```

```
TYPE stLevel1:
STRUCT
parL1_1: INT;
parL1_2: ARRAY [1..5] OF stLevel2;
END_STRUCT
END_TYPE
```

```
JJ_1, , testLevel[1].parL1_1, KANAL, 1, 1, NOTHING
JJ_2, , testLevel[2].parL1_2[2].parL2_1, KANAL, 1, 1, NOTHING
JJ_3, , testLevel[2].parL1_2[2].parL2_2[3].parL3_1, KANAL, 1, 1, NOTHING
JJ_4, , testLevel[5].parL1_2[5].parL2_2[5].parL3_2[5], KANAL, 1, 1, NOTHING
```

Network variable declarations file content, which defines the translation process variables relating to structures:

```
VAR_GLOBAL
testLevel: ARRAY [1..5] OF stLevel1;
END_VAR
```

## Driver Parameters

CtWago driver parameters are declared in the [Driver parameters / Driver parameters 2](#) tab.

### **Period of controllers connection test**

Meaning - this option is used to determine the maximum allowable time between two successive messages received from the controller. After this time, the driver sends a 'Bootup request' command to the controller to initiate a response from the controller. If there is no response, loss of connection with the controller is indicated.

Parameter:  
*number* - Max. acceptable time expressed in seconds between two consecutive messages received from the controller.

The default value - The default value is 10 (seconds).

### **Response timeout after control**

Meaning - This option is used to specify the maximum time to receive the 'Write on change' type message in response to the control signal transmitted to the controller. After this time, a 'Bootup request' message is transmitted to force a message from the controller. If there is no response, a runtime error and loss of connection with the controller are indicated.

Parameter:  
*number* - Max. acceptable time expressed in milliseconds between sending a control signal and receiving a 'Write on change' message from the controller.

The default value - The default value is 500 (milliseconds).

### **Response timeout after a request to refresh**

Meaning - This option is used to specify the maximum time to receive the response to a 'Bootup request' message sent by the driver to the controller. If there is no response, loss of connection with the controller is indicated.

Parameter:  
*number* - Max. acceptable time expressed in milliseconds between sending a 'Bootup request' message and receiving a response from a controller.

The default value - The default value is 500 (milliseconds).

**Timeout between commands Bootup Request**

Meaning - This option is used to specify the delay between successive 'Bootup request' messages sent to controllers during driver start-up. The value of this parameter determines the duration of the application start-up.

Parameter:  
*number* - time delay expressed in milliseconds between successive 'Bootup request' messages.

The default value - The default value is 20 (milliseconds).

**Log file**

Meaning - the text log file to which driver status messages are stored is used for diagnostic purposes.

The default value - by default, the log file is not created.

Parameter:  
*file name*

**Log file size**

Meaning - This option is used to determine the log file size defined using the *Log file* option.

Parameter:  
*number* - the size of the log file in MB.

The default value - By default, the log file size is 10 MB.

**Log of telegrams**

Meaning - this option allows writing the contents of messages communicated between the driver and the controllers to the log file (declared using the *Log File* item) The option in question should only be used during the Asix system start-up.

The default value - By default, this option is disabled.

**EXAMPLE**

Sample driver parameterization.

*Driver: Wago*

*Log file: d:\tmp\CtWago\muel.log*

*Log file size: 20*

*Log of telegrams: YES*

*Period of controllers connection test: 20*

*Response timeout after control: 700*

*Response timeout after a request to refresh: 600*

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

**Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the

names of network computers which will be permitted to exercise remote control.

Option value:  
 YES / NO  
*computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - NO - by default, only local computers can modify variable values.

**Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:  
*computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value - by default, all computers can read variables.

**Redundancy**

Purpose - with this option searches a redundant channel when connection with a controller is broken in local channel.

Option value:  
*computer name* - list including the names of network computers which will be permitted to search a redundant channel.

The default value - by default, when option is enabled, the channel is searched on all computers.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

**Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:  
 YES / NO  
 The default value - NO.

**Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:  
*number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value - by default, no time for data validity is set.

## 1.113 ZdarzenieZmienna Driver

### Driver Use

The ZdarzenieZmienna driver is used to generate process variables of WORD types (16-bit word) on the basis of current values of alarm events in the Asix system.

### Declaration of Transmission Channel

The ZdarzenieZmienna driver is a dynamic library DLL with an interface meeting requirements of the ASMEN module. ASMEN activates the driver after having found in the application configuration file, a channel definition calling to the ZdarzenieZmienna driver.

Declaration of the transmission channel using the ZdarzenieZmienna driver requires a channel with the following parameters to be added to the Current data module:

**Standard tab:**

*Name:* logical name of the transmission channel

*Driver:* ZdarzenieZmienna

After starting the Asix system, the ZdarzenieZmienna driver receives from ASMEN one after the other an algorithm, taken from the definition file, calculating the value of each process variable. This algorithm is executed at each reading the variable and its result becomes a value of the process variable.

Variables supplied by the driver are variables only for reading.

### Calculating the Asix Process Variables

The alarm module of the Asix system generates values of individual alarm events, which are identified by its number in the file of alarm definitions.

Two ways of determining values of alarm event exist:

- alarm event assumes a value of 1 when the alarm is active, 0 when it is inactive.
- alarm event assumes a value of 1 when the alarm is active and not acknowledged, 0 when it is active and acknowledged or inactive.

Alarm events are used to calculate Asix process variables identified by means of their name in the whole system.

Definition of ASMEN variables (in database of variables) contains the way of calculating values of an individual variable of the Asix system.

The declaration of each variable has the following form:

```
variable_name,variable_description,algorithm for calculating
values,channel,1,frequency,NOTHING
```

A detailed description of components of this line may be found in chapter Declaring process variables.

The algorithm for calculating values is specific for this described driver. There are three algorithms of operations on alarm events that generate a value of the process variable:

- composition of alarm events,
- logical sum of alarm events,
- logical product of alarm events.

An alarm event is treated as a number of alarm in the alarm file with a suffix N. The N suffix signifies that an alarm event assumes a value of 1 only when the alarm with a given number is active and not acknowledged.

#### Composition of Alarm Events

The algorithm form:

*Composition: <Alarm event >; <Alarm event >; ...*

or

*MERGE: <Alarm event >; <Alarm event >; ...*

The interpretation result of this algorithm is a number, the bit no. 0 of which has a value equal to the first alarm variable, bit no. 1 equal to the second alarm variable etc. The algorithm may contain maximally 16 alarm variables.

#### **EXAMPLE**

MERGE: 11; 12; 13N

In the algorithm, the alarm variables may be omitted and then the state of bit corresponding with omitted alarm variable is always equal to 0.

#### **EXAMPLE**

MERGE: 11; ; 13; ;15; 16

The bits no. 1 and 3 have always a value of zero.

#### Logical Sum of Alarm Events

The algorithm form:

*Logical\_sum: <Alarm event>; <Alarm event>; ...*

or

*OR: <Alarm event >; <Alarm event >; ...*

The interpretation result of this algorithm is a value of 1 if at least one of the alarm variables has a value of 1, 0 - otherwise. The algorithm may contain maximally 16 alarm variables.

**EXAMPLE**

OR: 11N; 12; 13

Logical Product of Alarm Events

The algorithm form:

*LogicalProduct: <Alarm event>; <Alarm event>; ...*

or

*AND: <Alarm event>; <Alarm event>; ...*

The interpretation result of this algorithm is a value of 1 if all alarm variables have a value of 1, 0 - otherwise. The algorithm may contain maximally 16 alarm variables.

**EXAMPLE**

AND: 11; 12N; 13N

Subjection of Values of Alarm Events to Value Status

Three algorithm above may also use alarm events, the value of which, taken from the alarm module, is still corrected by the state of the other alarm event treated as a status. You should use then the writing: alarm event/status e.g.101/102. When the status is equal to 0, then it does not change the value of alarm event. Otherwise a measure error for all Asix process variables is generated. It is calculated on the basis of this status.

**EXAMPLE**

AND: 11/110; 12; 13

or

OR: 11; 12/120; 13/130

## **EXAMPLE OF USING THE DRIVER**

Our task is to cause a text on synoptic diagram to blink while simultaneous exceeding limits for two temperatures T1 and T2. Two-state signals about exceeding appropriate temperatures we obtain from the object. We assume that alarms in the system are already activated and a file of alarms definitions exists.

## Declarations in the Configuration Application File

In the configuration file for the application we place a declaration initiating the driver ZdarzenieZmienna for the channel named EVENT:

Channel name: EVENT  
Driver: ZdarzenieZmienna

## Complement of File of Alarms Definitions

In the alarm file you should add two alarms, which signals exceeding the temperatures T1 and T2. For example, we place them under numbers: 101 for exceeding the temperature T1 and 102 for exceeding the temperature T2:

101,AL,Exceeding of temperature T1  
102,AL,Exceeding of temperature T2

## Creation of File Defining the Asmen Variables

You should declare new variable by using *VariableBase Manager*.

The process variable should assume a value of 1 when both alarms 101 and 102 are active (they assume a value of 1), otherwise the value of process variable should be equal 0. Let's name it T1\_T2\_MAX.

The file of the variable definition should contain the following line:

T1\_T2\_MAX,Process variable of exceeding T1 and  
T2,AND: 101; 102,EVENT,1,1,NOTHING

## Location of a Text Object on the Application Synoptic Diagram

After having opened the mask under *Constructor* you should place on it the TEXT object and parameterize it on our variable T1\_T2\_MAX. The object must have two states. For the value of a monitored variable 0 the text "Temperatures T1 and T2" are displayed. For the value 1 a text "Exceeding of temperatures T1 and T2" with a blink attribute.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:  
YES / NO

## Communication Drivers

- computer name* - list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

- Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

- computer name* - list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).
- The default value - by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

- Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

- YES / NO
- The default value - NO.

### **Data Validity Period**

- Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

- number* - time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.
- The default value - by default, no time for data validity is set.

## 1.114 ZxD400 - Driver of Protocol of Landys & Gyr ZxD400 Electric Energy Counters

### Driver Use

The ZxD400 driver is used for data exchange between Asix and electric energy counters of ZxD400 type manufactured by Landys & Gyr, via the RS-485 interface. The driver is not adapted for data exchange through an optical connection, because it demands the protocol with initial negotiations of transmission speed to be used.

The driver allows readout of register statement of a counter as well as registration of data (read by commands *Read Log Book* or *Read Load Profile*) in files.

Parameterization of ZxD400 driver is performed with the use of Architect module.

### Declaration of Transmission Channel

Declaration of the transmission channel using the ZxD400 driver requires a channel with the following parameters to be added to the *Current data* module:

**Standard** tab:

*Name*: logical name of the transmission channel

*Driver*: ZxD400

**ZxD400/Channel partameters** tab:

<i>Port</i>	- number of the COM serial port;
<i>Boud rate</i>	- speed of transmission between computer and device; the following speeds are acceptable: 1200,2400, 4800, 9600, 19200, 38400 Bd; a default value is 2400 Bd;
<i>Period</i>	- timeout (in seconds) between successive readouts of counter registers. A default value is 10 seconds.

### Declaring the Process Variables

The syntax of the symbolic address of the process variable is as follows:

"[*counter\_name*]/*register\_code*"

where:

<i>counter_name</i>	- (option); defines the controller unique name used in multipoint installations to identify particular controllers;
<i>counter_name</i>	- it corresponds to <i>Device address</i> according to PN-EN 61107; <i>counter_name</i> may be omitted in point-point connection;

*register\_code* - code and index of the counter register compatible with a readout list - that is loaded into the counter by the manufacturer in the parameterization stage.

**NOTICE** All the variable values are of *FLOAT* type.

### EXAMPLE

```
/* C.1.0      - identification number of the counter */
JJ_01, identification number of the counter, "/C.1.0",CHANNEL, 1, 1, NOTHING_FP

/* 1.8.0      - register of consumed active energy*/
JJ_03, register of consumed active energy, "/1.8.0",CHANNEL, 1, 1, NOTHING_FP

/* C.8.0      - total worktime */
JJ_03, total worktime, "/C.8.0",CHANNEL, 1, 1, NOTHING_FP
```

## Driver Parameterization

The driver configuration is defined in the *Current Data* module, in the channel operating according to Zx400 driver.

**Log file**

Meaning - allows to define a file to which all the diagnostic messages of the driver will be written.  
 Default value - by default, the log file is not created.

**Log file size**

Meaning - this item is used to define the size of the log file defined with use of the LOG\_FILE item.  
 Default value - by default, the log file size is 1 MB.  
 Parameter:  
     *number* - log file size in MB.

**Log of telegrams**

Meaning - this item allows contents of telegrams transferred between driver and controllers to be written into the log file (declared with use of the LOG\_FILE item). The referred item should only be used in the Asix system start-up.  
 Default value - by default, value of this item is set to NO.

**History log file**

Meaning - allows to define history log file.  
 Default value - by default, the log file is not created.

**Suspended Readout Period**

Purpose: The option specified the time during which the counter readout is suspended around midnight (i.e. readout disabled - *seconds* - midnight - *seconds* - readout enabled). When determining the value

of the option, allow for some offset during the disabled readout time, because the disable time may occur during current data transmission (a dozen or so seconds ?). For other data categories the offset is of several seconds. During the suspended readout time, the values of current variables are given status OPC\_QUALITY\_UNCERTAIN.

Option value:

*seconds*

- time measured in seconds.

The default value:

The default value is 0 (i.e. the readout is not suspended).

**Suspended Readout – Every Day**

Purpose: This option is used to suspend the counter readout on a daily basis. It suspends the data readout at midnight every day.

The default value: By default value of the position is NO (readout is not suspended).

**Suspended Readout – Every Decade**

Purpose: This option is used to suspend the counter readout on a decade basis. It suspends the data readout at midnight when the decade changes.

The default value: By default value of the position is NO (readout is not suspended).

**Suspended Readout – Every Month**

Purpose: This option is used to suspend the counter readout on a monthly basis. It suspends the data readout at midnight when the month changes.

The default value: By default value of the position is NO (readout is not suspended).

## Parameterization of Particular Counters

The driver allows the set of individual parameters concerning service of particular counters to be transferred in separate sections. Such the parameterization is set on **Zx400 / Channel parameters / Counters** tab. The parameters differs depending on counters definition (**Add new counter** or **Add point-to-point**).

**Counter name**

- address name of the counter (the name used in a variable address); counter\_name may be empty, if point-point installation is used.

A parameterization of the counter may be performed by the following items:

**Option name: Time maker**

**Option value: register\_code [,register\_code]**

Meaning

- it allows to define a register (or two registers), that includes the data and time stamp transmitted from the counter. It is assumed that, if one register is declared, it contains data and time in the 'YY-MM-DD hh:mm:ss' format. If the couple of registers is declared, it is assumed that the first register contains data in the 'YY-MM-DD' format, and the second one contains time in the 'hh:mm:ss' format.

## Communication Drivers

Default value - by default, it is assumed that the PC's data and time stamp from the moment of the end of data receiving will be assigned to variables transmitted from the counter.

**Option name: Maximum buffer length**

**Option value: number**

Meaning - this item is used to define maximum length of the buffer of the receiver.

Default value - 100000.

Parameter:  
*number* - capacity in bytes

**Suspended Readout Period**

Purpose: The option specified the time during which the counter readout is suspended around midnight (i.e. readout disabled - *seconds* - midnight - *seconds* - readout enabled). When determining the value of the option, allow for some offset during the disabled readout time, because the disable time may occur during current data transmission (a dozen or so seconds ?). For other data categories the offset is of several seconds. During the suspended readout time, the values of current variables are given status OPC\_QUALITY\_UNCERTAIN.

Option value:  
*seconds* - time measured in seconds.

The default value: The default value is 0 (i.e. the readout is not suspended).

**Suspended Readout – Every Day**

Purpose: This option is used to suspend the counter readout on a daily basis. It suspends the data readout at midnight every day.

The default value: By default value of the position is NO (readout is not suspended).

**Suspended Readout – Every Decade**

Purpose: This option is used to suspend the counter readout on a decade basis. It suspends the data readout at midnight when the decade changes.

The default value: By default value of the position is NO (readout is not suspended).

**Suspended Readout – Every Month**

Purpose: This option is used to suspend the counter readout on a monthly basis. It suspends the data readout at midnight when the month changes.

The default value: By default value of the position is NO (readout is not suspended).

**Log file**

Meaning - allows to define a file to which all the diagnostic messages of the driver will be written.

Default value - by default, the log file is not created.

**Log file size**

Meaning - this item is used to define the size of the log file defined with use of the LOG\_FILE item.

Default value - by default, the log file size is 1 MB.

Parameter:

*number* - log file size in MB.

**Log of telegrams**

Meaning - this item allows contents of telegrams transferred between driver and controllers to be written into the log file (declared with use of the LOG\_FILE item). The referred item should only be used in the Asix system start-up.

Default value - by default, value of this item is set to NO.

**Events log**

Meaning - allows to define a file to which all the vents of the counter will be written.

Default value - by default, the log file is not created.

**Events log file size**

Meaning - this item is used to define the size of the log file defined with use of the *Events file* item.

Default value - by default, the log file size is 10 MB.

Parameter:  
*number* - log file size in MB.

**Without events log**

Meaning - this item allows to set the option not reading event form the counter and not writing them into the log file.

Default value - the option is not set.

**Events log read period**

Meaning - this item allows to set the period of read of events log from the counter.

Default value - 1.

Parameter:  
*number* - time in hours.

**Power profiles log**

Meaning - Power profiles are read from the counter and saved to a text file in the format 'csv'. Each power profile is written in a separate line. Power profiles are ordered by increasing timestamps. An exemplary form of a singular power profile is as follows (the profile is divided into 2 text lines for the better clarity):

```
P.01; 1; 2005-05-10 12:45:00; 0008; 15; 6; 1.5.0; kW; 5.5.0; kvar; 8.5.0; kvar;
32.7; V; 52.7; V; 72.7; V; 5.123000; 24.2438900; 123.654320; ---.; ---.; ---.-
```

The option allows to define the full path to the file where profile log will be written.

Default value - The log file is not created.

**Power profiles log file size**

Meaning - The item allows to specify the size of log file defined by the option: *Power profiles log*.

Default value - 10.

Option value:

## Communication Drivers

*number* - log file size in MB.

**Without power profiles log**

Meaning - The option switches off reading power profiles.  
Default value - By default, the driver reads the power profile log.

**Power profiles log read period**

Meaning - The option determines the cycle of reading the power profile log from the counter.  
Default value - Every 1 hour.

Option value:

*number* - time in hours.

**Power profiles log history scope**

Meaning - The option declares how far into the past history of events the driver will reach, reading power profiles log.  
Default value - By default, the driver goes back 32 days since the current 24 hour, reading the history of power profiles.

**Option name: Log book data**

**Option value: YES/NO**

Meaning - it allows to declare whether detailed description of parsing of particular event log lines should be entered to the driver log. The log file should be used only while the Asix system start-up.  
Default value - by default, event parsing details are not written to the driver log.

**Option name: Log profile data**

**Option value: YES/NO**

Meaning - it allows to declare whether detailed description of parsing of particular profile log lines should be entered to the driver log. The log file should be used only while the Asix system start-up.  
Default value - by default, event parsing details are not written to the driver log.

**Option name: Meter data**

**Option value: YES/NO**

Meaning - it allows to declare whether detailed description of parsing of particular data (read from the counter) lines should be entered to the driver log. The log file should be used only while the Asix system start-up.  
Default value - by default, readout data parsing details are not written to the driver log.

## Advanced Channel Parameters

The parameters of a channel are declared in the *Advanced* tab:

### **Remote Write Access**

Purpose - this option allows all computers to exercise remote control in the channel (by default, only locally installed applications can modify variable values); it is also possible to declare a list including the names of network computers which will be permitted to exercise remote control.

Option value:

YES / NO

*computer name*

- list including the names of network computers which will be permitted to exercise remote control (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- NO - by default, only local computers can modify variable values.

### **Remote Read Limitation**

Purpose - with this option deployed, the number of computers authorised to remote readout of data will be limited to the computers with specific network names.

Option value:

*computer name*

- list including the names of network computers which will be permitted to readout data (network names can consist of maximum 15 characters; the names are separated by commas).

The default value

- by default, all computers can read variables.

\*\*\*

The parameters of a channel are declared in the *Advanced 2* tab:

### **Local Write Denied**

Purpose - with this option deployed, local control in a channel will be disabled.

Option value:

YES / NO

The default value

- NO.

### **Data Validity Period**

Purpose - this option specifies the time throughout which variable values in a channel are valid; this option should only be used when temporary lack of data is not treated as an error.

Option value:

*number*

- time in seconds; a value (much) higher than the longest period of data refresh in a channel should be specified.

The default value

- by default, no time for data validity is set.

