ELEKTRONIK GMBH

GET IN touch WITH SENSITIVE TESTING

Manual

Toolmonitor SerIO

Softline

Modline

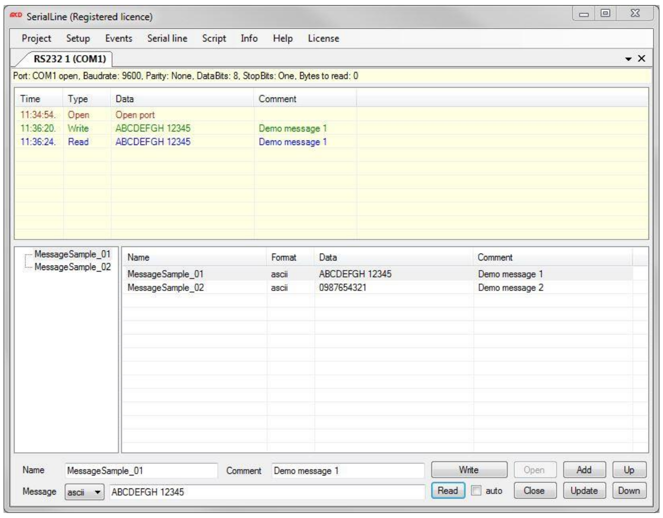
Conline

Boardline

Avidline

Pixline

Application


SerialLine (Registered licence)

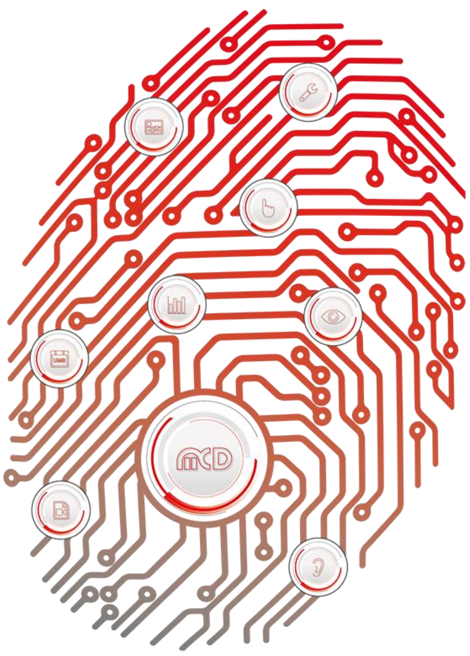
Port: COM1 open, Baudrate: 9600, Parity: None, DataBits: 8, StopBits: One, Bytes to read: 0

Time	Type	Data	Comment
11:34:54.	Open	Open port	
11:36:20.	Write	ABCDEFGH 12345	Demo message 1
11:36:24.	Read	ABCDEFGH 12345	Demo message 1

MessageSample_01
MessageSample_02

Name	Format	Data	Comment
MessageSample_01	ascii	ABCDEFGH 12345	Demo message 1
MessageSample_02	ascii	0987654321	Demo message 2

Name: MessageSample_01
Comment: Demo message 1
Message: ascii ABCDEFGH 12345



MCD Elektronik GmbH
Hoheneichstr. 52
75217 Birkenfeld
Tel. +49 (0) 72 31/78 405-0
Fax +49 (0) 72 31/78 405-10
info@mcd-elektronik.de
www.mcd-elektronik.com
HQ: Birkenfeld
Managing CEO: Bruno Hörter
Register Court Mannheim
HRB 505692

Template version: 5.4 / 2018-04-09
V1.1 2018-09-04 TI (TS)

Table of Contents

1. General

This document's primary sections are “Features” and “Programming”. The descriptions of the individual software modules will be found in the “Features” section. The interfaces for users of this software will be listed in the form of tables in the “Programming” section. The functions for accessing Toolmonitor will be documented in the Interface section. The valid parameters for these functions will be indicated in the respective sections about the modules.

Toolmonitor SerIO is based on the MCD Framework for Toolmonitors. The framework's basic features, such as logging, general settings and licensing, will also be described here.

Additional information about Toolmonitor's framework will be found in the “General Help” or “Description” section of the MCD Framework. In particular, this includes the description of the “Script Engine”, the “Virtual Interface” and the user - specific “Visualization”.

Order number: # 150419

2. Installation of Software

2.1. Requirements

- Operating system: Windows XP® - Windows 8.1®
- Architecture: 32 bit or 64 bit
- .Net framework: version 3.0

To install the MCD Toolmonitor SerIO, it is sufficient simply to copy “SerIOMonitor.exe” into any directory on the target system.

Alternatively, the installer provided (“SerIOInstall.msi”) may be executed.

2.2. License

To protect the software from unauthorized use, it is required to license the Toolmonitor after installation.

For purposes of demonstrations and testing the Toolmonitor can also be operated for 30 minutes at a time without a license. Some program functions are disabled. A 24 - hour temporary license can also be activated while waiting for permanent activation (for example on a weekend).

To activate the Toolmonitor, please open the “License administration” dialog from the menu item
License → Register.

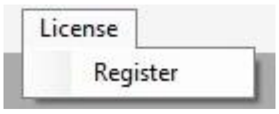
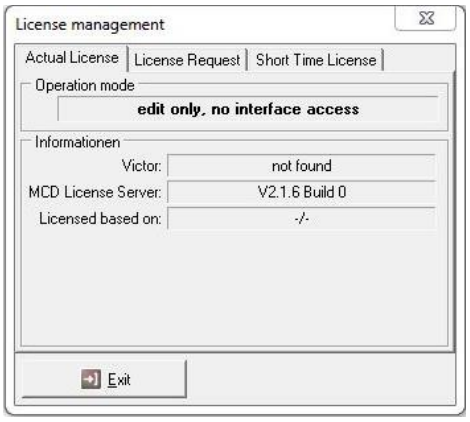


Figure 1: Calling Up the Registration Dialog

1.) The “Current licensing” dialog shows the status of your current license:



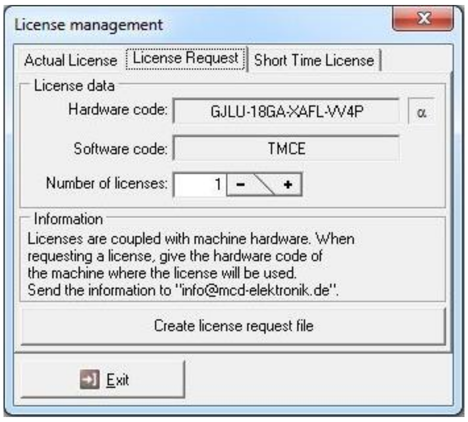
Operation mode: edit only, no interface access

Informationen
Victor: not found
MCD License Server: V2.1.6 Build 0
Licensed based on: -/-

Figure 2: Retrieving License Status

2.) To request a permanent license for your software, please proceed as follows:

- Select the “Request license” dialog
- Specify the number of licenses needed (for your PC) in the “Number of licenses” field
- Click the “Generate request file” button
- Another window then opens to ask you to save the “MCD Licenser Request” file (*.mlr)
- Please save this file and send it by email to info@mcd-elektronik.de. Please specify an order number or project number to simplify allocation
- You will then receive an email from MCD Elektronik with your license file (“MCD License Key *.mlk”) attached
- Finally, save this file either under “C:\Windows” or into the folder where the “.exe” file for your software was installed
- After the next start of your software, it will then be available with its full functional scope



Hardware code: GJLU-18GA-XAFL-VV4P

Software code: TMCE

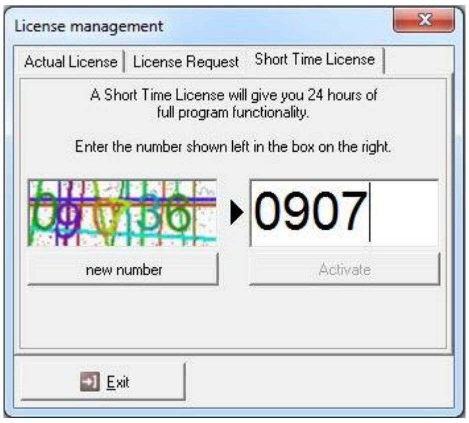
Number of licenses: 1

Information

Licenses are coupled with machine hardware. When requesting a license, give the hardware code of the machine where the license will be used. Send the information to "info@mcd-elektronik.de".

Figure 3: Requesting a Permanent License

3.) To activate a temporary license (24 hours), please select the “Temporary license” tab. Then enter the sequences of digits from the left window into the right window. If you cannot interpret the numbers, please click the “New number” button to receive a new number. Once you have correctly entered the number, you can activate the temporary license using the “Activate license” button. Please note that the temporary license will expire as soon as you stop the software. However, you can activate the temporary license as often as you like.



A Short Time License will give you 24 hours of full program functionality.

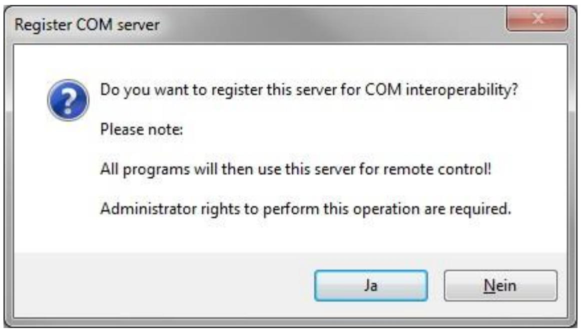
Enter the number shown left in the box on the right.

0907

Figure 4: Requesting a Temporary License

2.3. Register COM Server

This command registers the Toolmonitor as a COM Server. This is required if the Toolmonitor will be remote controlled by other programs, such as the MCD TestManager.



Do you want to register this server for COM interoperability?

Please note:

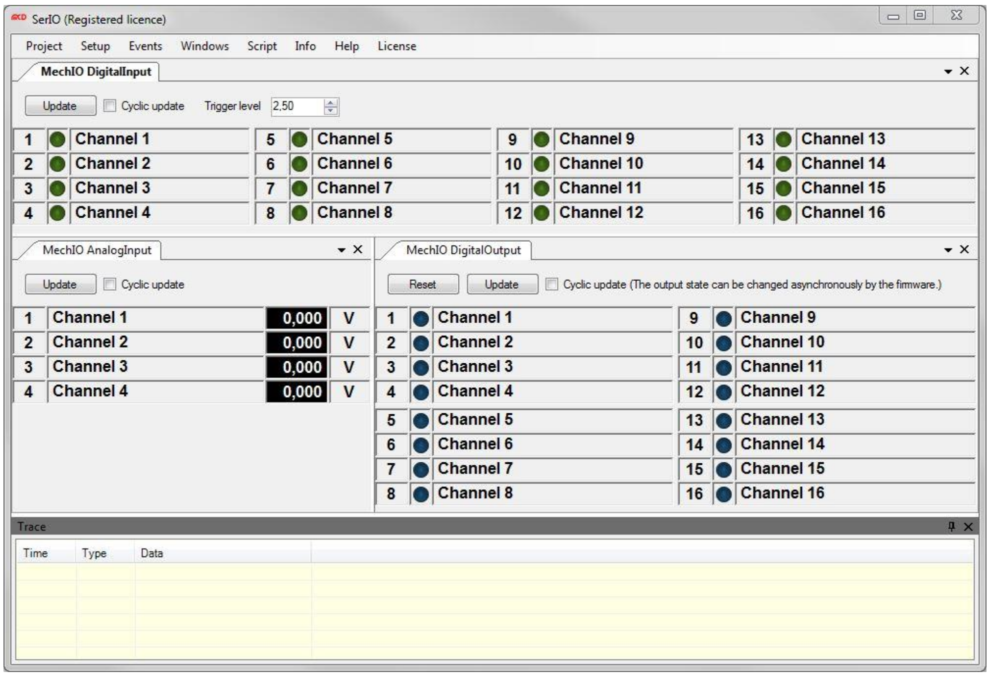
All programs will then use this server for remote control!

Administrator rights to perform this operation are required.

Figure 5: Register COM Server

3. Introduction to Toolmointor SerIO

Toolmonitor SerIO serves to control and visualize MCD products and those devices that are controlled by the standardized SerIO protocol used internally by MCD.



MechIO DigitalInput

1	●	Channel 1	5	●	Channel 5	9	●	Channel 9	13	●	Channel 13
2	●	Channel 2	6	●	Channel 6	10	●	Channel 10	14	●	Channel 14
3	●	Channel 3	7	●	Channel 7	11	●	Channel 11	15	●	Channel 15
4	●	Channel 4	8	●	Channel 8	12	●	Channel 12	16	●	Channel 16

MechIO AnalogInput

1	Channel 1	0,000	V
2	Channel 2	0,000	V
3	Channel 3	0,000	V
4	Channel 4	0,000	V

MechIO DigitalOutput

1	●	Channel 1	9	●	Channel 9
2	●	Channel 2	10	●	Channel 10
3	●	Channel 3	11	●	Channel 11
4	●	Channel 4	12	●	Channel 12
5	●	Channel 5	13	●	Channel 13
6	●	Channel 6	14	●	Channel 14
7	●	Channel 7	15	●	Channel 15
8	●	Channel 8	16	●	Channel 16

Figure 6: Toolmonitor SerIO

The SerIO protocol supports the following MCD products:

- Unknown (core functions for non - specific customer requests)
- MechIO 8
- MechIO 16
- Adapter CPU
- Pass / Fail display

The following devices are supported:

- Unknown (core functions for non - specific customer requests)
- PCF8574 | PIO 8 channels per port
- PCF8574A | PIO 8 channels per port
- PCA9554 | PIO 8 channels per port
- PCA9554A | PIO 8 channels per port
- PIO 3 x PCA 9554 | B.140.010.110
- PIO 3 x PCA 9554A | B.140.010.210
- PIO 3 x PCF 8574 | B.140.010.310
- PIO 3 x PCF 8574A | B.140.010.410
- RelMUX16 | Relay multiplexer 16 channels 7442
- PST30M | Power supply for tuner applications master B.170.030.112
- PST30S | Power supply for tuner applications slave B.170.030.212
- PST30L | Power supply for tuner applications linear slave B.170.030.412

Freely definable telegrams or those stored in a library may be sent and received in a simple manner. Digital or analog inputs and output can be queried or set.

The program interface can largely be designed with an open hand and customized to the users' requirements. Once created, configuration can be saved in “Project files” and loaded as needed. All telegrams can be sent and received automatically with the help of an integrated “Script Engine”. Asynchronous processes can be stored in the Toolmonitor. Using third - party software, the Toolmonitor can be entirely remote controlled.

COM / DCOM or a .Net assembly may be used as the interface for this. Thereby, the Toolmonitor can be integrated into a number of applications, including Microsoft Visual Studio® (C#, C++ and Visual Basic), Microsoft Office® (such as Excel®), Open Office®, LabVIEW® and MCD TestManager CE. Additional information about this can be found in the “Programming” section and in the “General Help” about the Toolmonitors.

4. Windows

The “Windows” menu item on Toolmonitor SerIO‘s main menu provides access to all configured communications channels, SerIO devices and I2C modules.

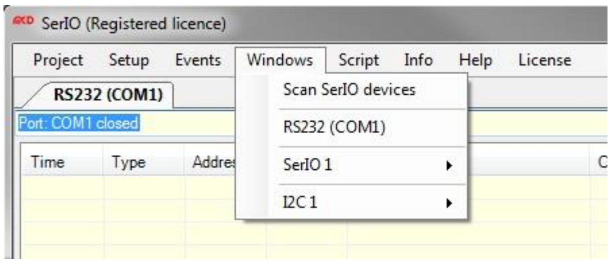


Figure 7: Windows Menu

4.1. Scan SerIO Devices

The “Scan SerIO Devices” menu item on Toolmonitor SerIO‘s “Windows” menu makes automated scanning for connected SerIO devices possible. The scanning process will first close all serial interfaces and then check them in turn for SerIO devices using various baud rates and addresses.

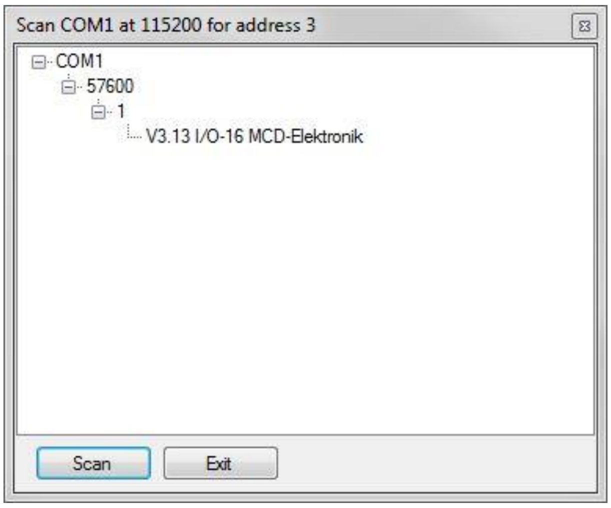
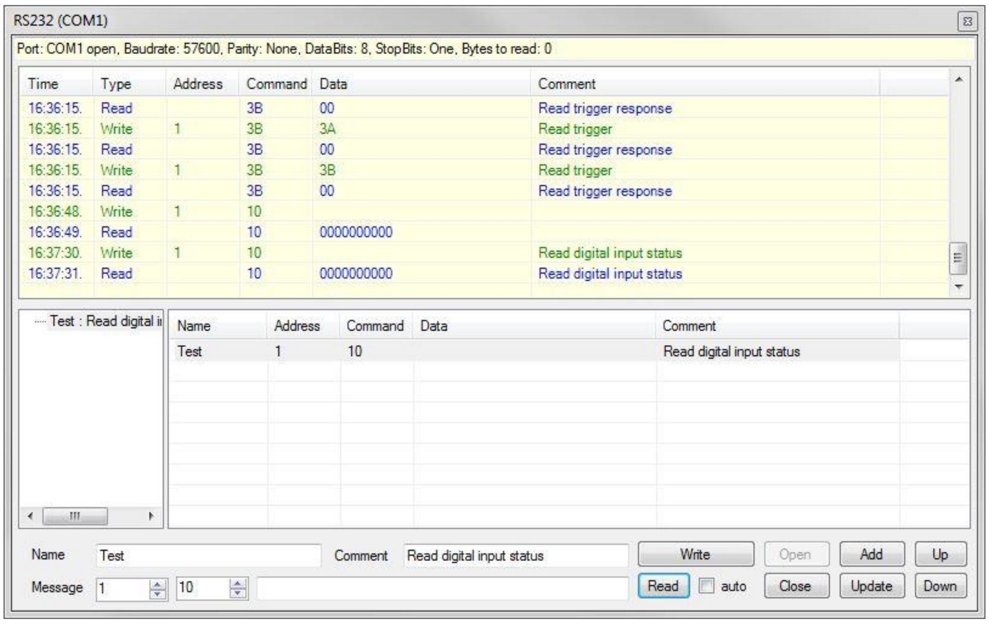

Scan COM1 at 115200 for address 3
COM1
57600
1
V3.13 I/O-16 MCD-Elektronik
Scan
Exit

Figure 8: Scan SerIO Devices

4.2. RS232 Communications

A menu item will be added to Toolmonitor SerIO‘s “Windows” menu for each RS232 communications channel. A window for low level analysis and communications control through these RS232 channels can be opened using the corresponding items. All open serial communications can be listened to or the user‘s own messages can be sent with the help of this window. The received data can be logged. Furthermore, messages, or sequences of messages, created by the user can be saved and loaded.



Port: COM1 open, Baudrate: 57600, Parity: None, DataBits: 8, StopBits: One, Bytes to read: 0

Time	Type	Address	Command	Data	Comment
16:36:15.	Read		3B	00	Read trigger response
16:36:15.	Write	1	3B	3A	Read trigger
16:36:15.	Read		3B	00	Read trigger response
16:36:15.	Write	1	3B	3B	Read trigger
16:36:15.	Read		3B	00	Read trigger response
16:36:48.	Write	1	10		
16:36:49.	Read		10	0000000000	
16:37:30.	Write	1	10		Read digital input status
16:37:31.	Read		10	0000000000	Read digital input status

Name	Address	Command	Data	Comment
Test	1	10		Read digital input status

Name: Test Comment: Read digital input status
Message: 1 10

Figure 9: RS232 Communications

Clicking the “Open” button will start communications. Clicking this button will automatically make settings to the channel based on the “Options”.

User messages can be created using an entry dialog. The message type and data content can be entered for this purpose.

The “SerIO address”, the “SerIO command” and the associated data can be entered using the “Message” area.

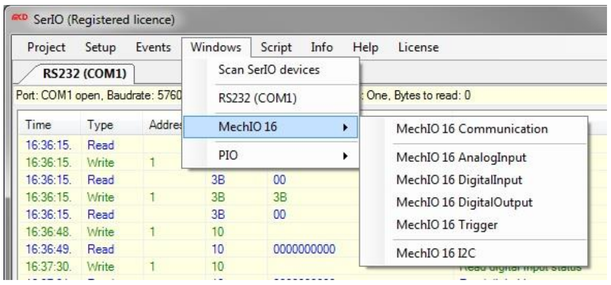
Clicking the “Write” button will send the message. Messages, if any exist, can be received with the help of the “Read” button. If the “Auto” option has been enabled, incoming messages will be received and displayed automatically.

Additional information about the messages can be stored in the “Name” and “Comment” entries, which can be saved by clicking the “Add” button.

Clicking the “Close” button will terminate communications.

4.3. SerIO Devices

A menu item will be added on Toolmonitor SerIO‘s “Windows” menu for each SerIO device. A variety of windows for controlling the SerIO device can be opened using these menu items.



Port: COM1 open, Baudrate: 5760 … One, Bytes to read: 0

Time	Type	Addres		
16:36:15.	Read			
16:36:15.	Write	1		
16:36:15.	Read		3B	00
16:36:15.	Write	1	3B	3B
16:36:15.	Read		3B	00
16:36:48.	Write	1	10	
16:36:49.	Read		10	0000000000
16:37:30.	Write	1	10	

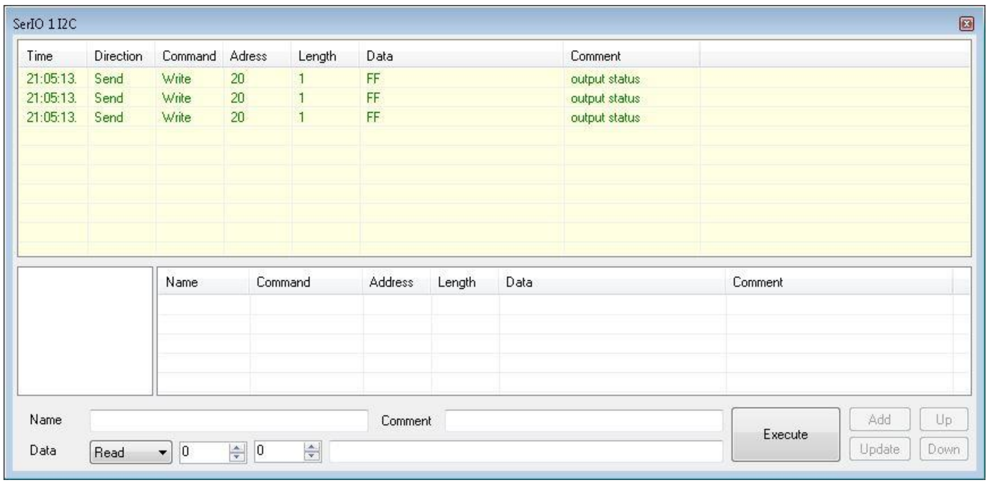
Figure 10: SerIO Devices Sub Menu

4.3.1. MechIO 8

4.3.1.1. I^{2}C Communications

All open I^{2}C communications with the corresponding SerIO device can be listened to or the user's own messages can be sent with the help of this window.

The received data can be logged. Furthermore, messages, or sequences of messages, created by the user can be saved and loaded.



SerIO 1 I2C

Time	Direction	Command	Adress	Length	Data	Comment
21:05:13.	Send	Write	20	1	FF	output status
21:05:13.	Send	Write	20	1	FF	output status
21:05:13.	Send	Write	20	1	FF	output status

Name	Command	Address	Length	Data	Comment

Figure 11: I^{2}C Communications

User messages can be created using an entry dialog. The message type and data content can be entered to this purpose.

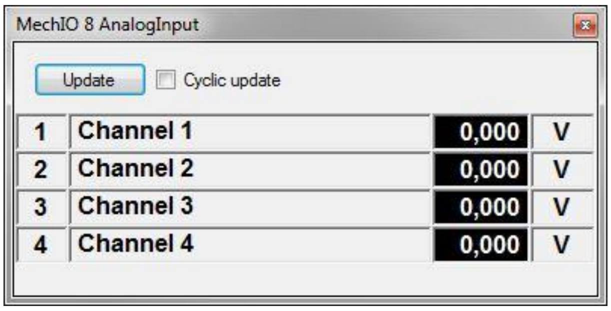
The "I^{2}C command", the "I^{2}C address" and the associated data can be entered using the "Message" area.

Clicking the "Execute" button will send the message.

Additional information about the messages can be stored in the "Name" and "Comment" entries, which can be saved by clicking the "Add" button.

4.3.1.2. MechIO 8 Analog Input

The analog inputs to the mechanical control system are displayed and controlled using this dialog window. Clicking the "Update" button will update the current state of the inputs. Checking the "Cyclic update" checkbox will update the inputs at periodic intervals.



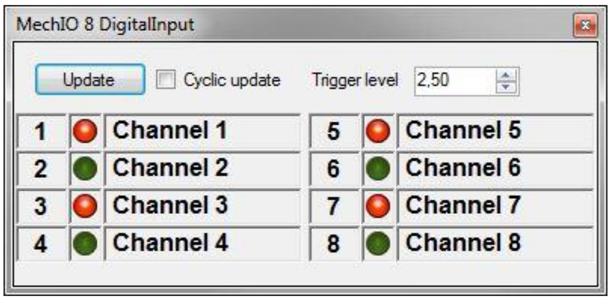
MechIO 8 AnalogInput

1	Channel 1	0,000	V
2	Channel 2	0,000	V
3	Channel 3	0,000	V
4	Channel 4	0,000	V

Figure 12: Analog Inputs

4.3.1.3. MechIO 8 Digital Input

The digital inputs to the mechanical control system are displayed and controlled using this dialog window. Clicking the “Update” button will update the current state of the inputs. Checking the “Cyclic update” checkbox will update the inputs at periodic intervals. The “Trigger level” entry will set the trigger thresholds (in volts).


MechIO 8 DigitalInput

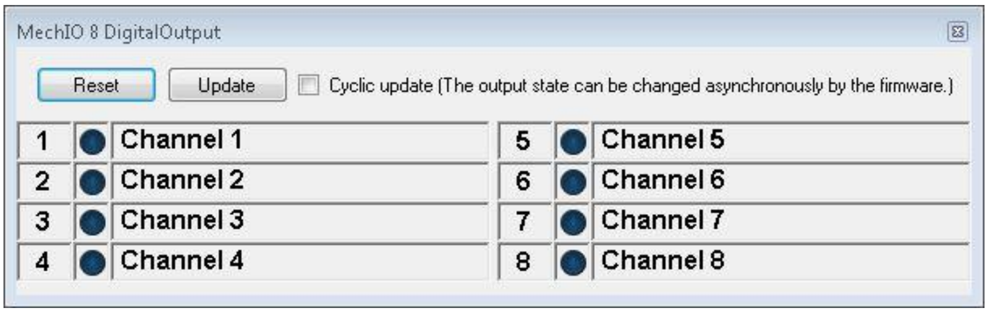
Update ☐ Cyclic update Trigger level 2,50

1	🔴	Channel 1	5	🔴	Channel 5
2	🟢	Channel 2	6	🟢	Channel 6
3	🔴	Channel 3	7	🔴	Channel 7
4	🟢	Channel 4	8	🟢	Channel 8

Figure 13: Digital Inputs

4.3.1.4. MechIO 8 Digital Output

The digital outputs from the mechanical control system can be displayed and controlled using this dialog window. Clicking the “Update” button will update the current state of the inputs. Checking the “Cyclic update” checkbox will update the outputs at periodic intervals.


MechIO 8 DigitalOutput

Reset Update ☐ Cyclic update (The output state can be changed asynchronously by the firmware.)

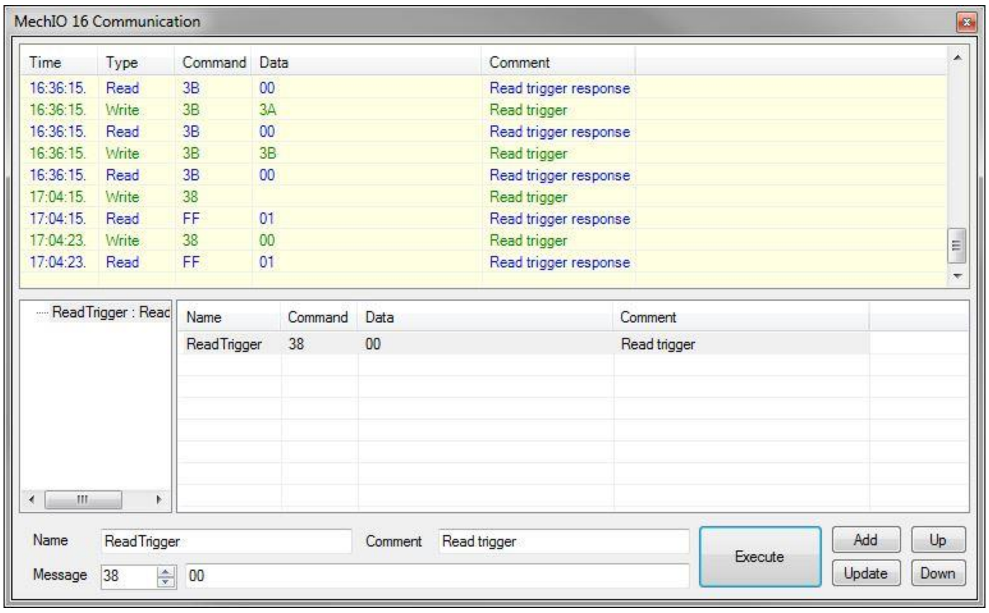
1	🔵	Channel 1	5	🔵	Channel 5
2	🔵	Channel 2	6	🔵	Channel 6
3	🔵	Channel 3	7	🔵	Channel 7
4	🔵	Channel 4	8	🔵	Channel 8

Figure 14: Digital Outputs

4.3.1.5. SerIO Communications

All open SerIO communications with the associated device can be listened to or the user‘s own messages can be sent with the help of this window.

The received data can be logged. Furthermore, messages, or sequences of messages, created by the user can be saved and loaded.



Time	Type	Command	Data	Comment
16:36:15.	Read	3B	00	Read trigger response
16:36:15.	Write	3B	3A	Read trigger
16:36:15.	Read	3B	00	Read trigger response
16:36:15.	Write	3B	3B	Read trigger
16:36:15.	Read	3B	00	Read trigger response
17:04:15.	Write	38		Read trigger
17:04:15.	Read	FF	01	Read trigger response
17:04:23.	Write	38	00	Read trigger
17:04:23.	Read	FF	01	Read trigger response

Name	Command	Data	Comment
ReadTrigger	38	00	Read trigger

Figure 15: SerIO Communication

User messages can be created using an entry dialog. The message type and data content can be entered to this purpose.

The “SerIO command” and the associated data can be entered using the “Message” area.

Clicking the “Execute” button will send the message.

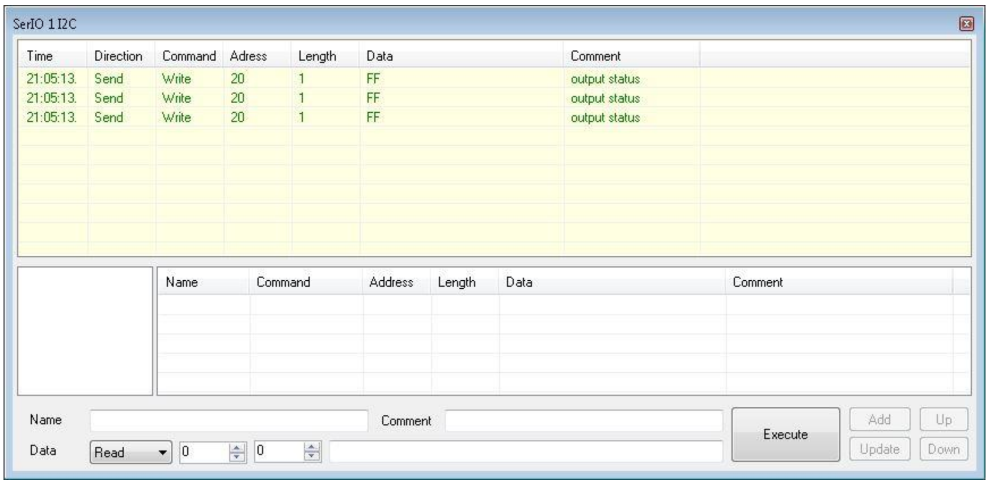
Additional information about the messages can be stored in the “Name” and “Comment” entries, which can be saved by clicking the “Add” button.

4.3.2. MechIO 16

4.3.2.1. I^{2}C Communications

All open I^{2}C communications with the corresponding SerIO device can be listened to or the user‘s own messages can be sent with the help of this window.

The received data can be logged. Furthermore, messages, or sequences of messages, created by the user can be saved and loaded.


SerIO 1 I2C

Time	Direction	Command	Adress	Length	Data	Comment
21:05:13.	Send	Write	20	1	FF	output status
21:05:13.	Send	Write	20	1	FF	output status
21:05:13.	Send	Write	20	1	FF	output status

Name	Command	Address	Length	Data	Comment

Name
Comment
Data Read 0 0

Figure 16: I^{2}C Communication

User messages can be created using an entry dialog. The message type and data content can be entered to this purpose.

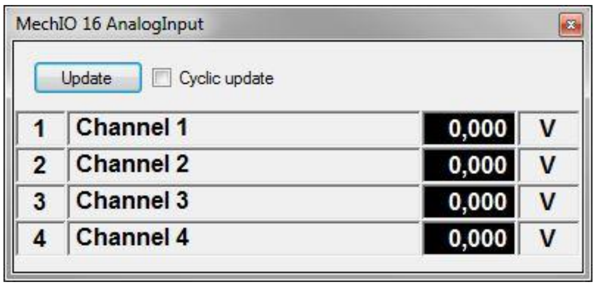
The “I^{2}C command”, the “I^{2}C address” and the associated data can be entered using the “Message” area.

Clicking the “Execute” button will send the message.

Additional information about the messages can be stored in the “Name” and “Comment” entries, which can be saved by clicking the “Add” button.

4.3.2.2. MechIO Analog Input

The analog inputs to the mechanical control system are displayed and controlled using this dialog window. Clicking the “Update” button will update the current state of the inputs. Checking the “Cyclic Update” checkbox will update the inputs at periodic intervals.

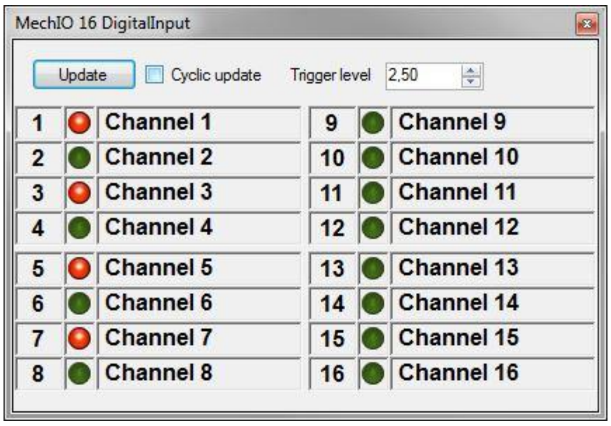

MechIO 16 AnalogInput

1	Channel 1	0,000	V
2	Channel 2	0,000	V
3	Channel 3	0,000	V
4	Channel 4	0,000	V

Figure 17: Analog Inputs

4.3.2.3. MechIO 16 Digital Input

The digital inputs to the mechanical control system are displayed and controlled using this dialog window. Clicking the “Update” button will update the current state of the inputs. Checking the “Cyclic Update” checkbox will update the inputs at periodic intervals. The “Trigger level” entry will set the trigger thresholds (in volts).


MechIO 16 DigitalInput

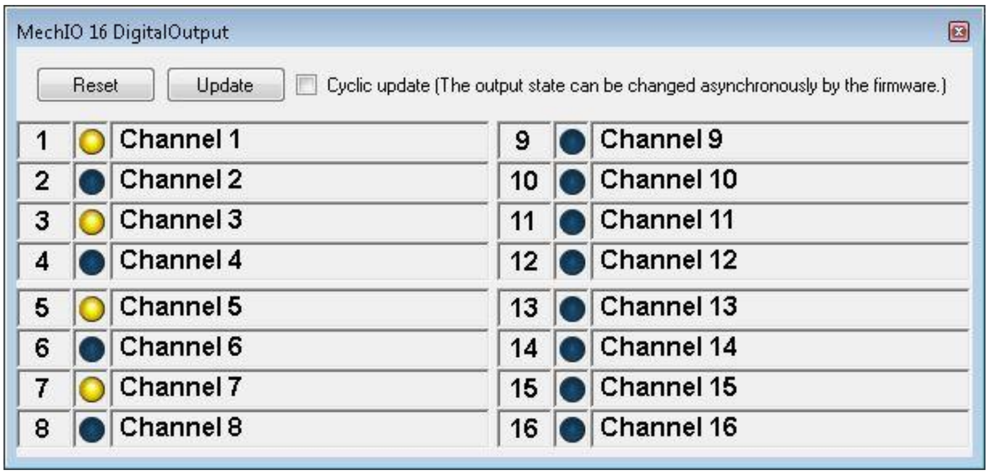
Update ☐ Cyclic update Trigger level 2,50

1	🔴	Channel 1	9	🟢	Channel 9
2	🟢	Channel 2	10	🟢	Channel 10
3	🔴	Channel 3	11	🟢	Channel 11
4	🟢	Channel 4	12	🟢	Channel 12
5	🔴	Channel 5	13	🟢	Channel 13
6	🟢	Channel 6	14	🟢	Channel 14
7	🔴	Channel 7	15	🟢	Channel 15
8	🟢	Channel 8	16	🟢	Channel 16

Figure 18: Digital Inputs

4.3.2.4. MechIO 16 Digital Output

The digital outputs from the mechanical control system can be displayed and controlled using this dialog window. Clicking the “Update” button will update the current state of the outputs. Checking the “Cyclic update” checkbox will update the outputs at periodic intervals.


MechIO 16 DigitalOutput

Reset Update ☐ Cyclic update (The output state can be changed asynchronously by the firmware.)

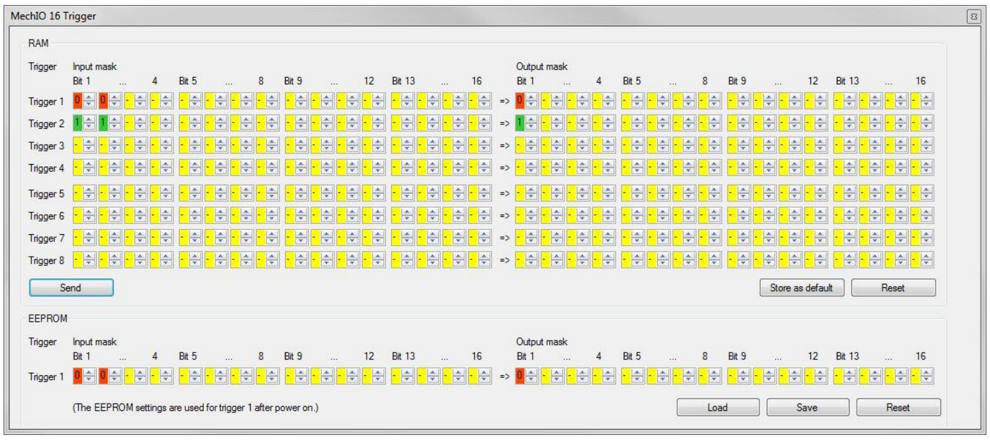
1	🟡	Channel 1	9	🔵	Channel 9
2	🔵	Channel 2	10	🔵	Channel 10
3	🟡	Channel 3	11	🔵	Channel 11
4	🔵	Channel 4	12	🔵	Channel 12
5	🟡	Channel 5	13	🔵	Channel 13
6	🔵	Channel 6	14	🔵	Channel 14
7	🟡	Channel 7	15	🔵	Channel 15
8	🔵	Channel 8	16	🔵	Channel 16

Figure 19: Digital Outputs

4.3.2.5. MechIO 16 Trigger

The triggers for the mechanical control system can be set using this window.

Up to a maximum of eight internal triggers can be defined. The mechanical control system will periodically check the inputs and set both the corresponding output states and the corresponding “Flag” according to the trigger condition. The level of precedence decreases as the “Index” number increases. This means that “Index 0” always has priority over the output state of, for example, “Index 1”. “Trigger Index 0” will be loaded from the EEPROM after “Power Up” and has the highest level of precedence.



RAM

Trigger	Input mask Bit 1			4	Bit 5			8	Bit 9			12	Bit 13			16		Output mask Bit 1			4	Bit 5			8	Bit 9			12	Bit 13			16
Trigger 1	0	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	=>	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Trigger 2	1	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	=>	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Trigger 3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	=>	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Trigger 4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	=>	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Trigger 5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	=>	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Trigger 6	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	=>	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Trigger 7	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	=>	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Trigger 8	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	=>	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

EEPROM

Trigger	Input mask Bit 1			4	Bit 5			8	Bit 9			12	Bit 13			16		Output mask Bit 1			4	Bit 5			8	Bit 9			12	Bit 13			16
Trigger 1	0	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	=>	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

(The EEPROM settings are used for trigger 1 after power on.)

Figure 20: Trigger Interface

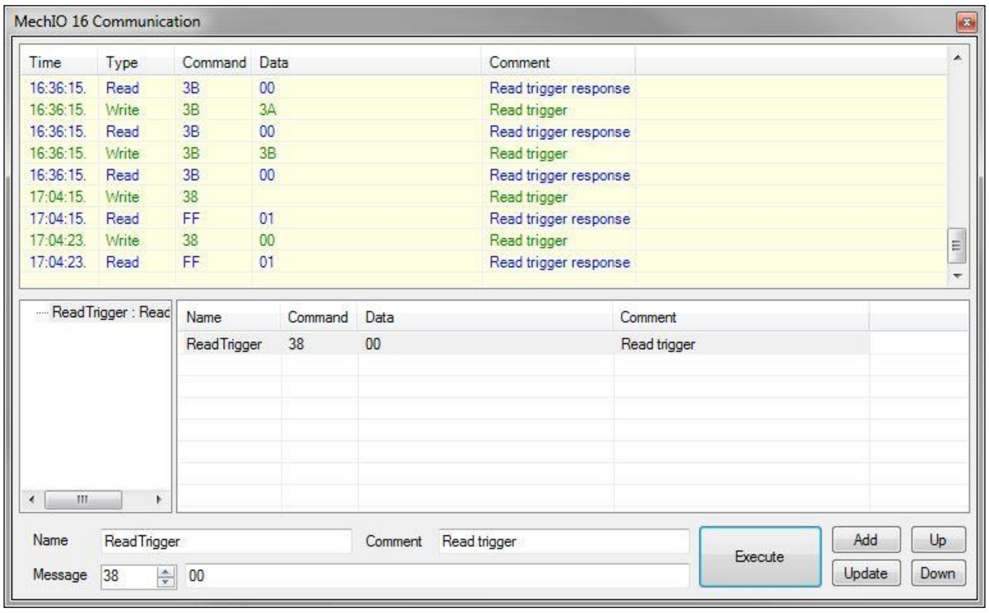
The Trigger interface consists of the following areas:

- Input mask: The input mask to be used for the corresponding trigger is set from here. A minus sign “-” means that the corresponding bit is not relevant for the trigger, a “0” means that the corresponding bit must be “Low” and a “1” means that the corresponding bit must be “High”
- Output mask: The output mask to be used for the corresponding trigger is set from here. A minus sign “-” means that the corresponding bit is not relevant for the trigger, a “0” means that the corresponding bit must be set to “Low” and a “1” means that the corresponding bit must be set to “High”
- Send: This button will transfer the trigger configuration to the mechanical control system
- Store as default: This button will store the current trigger configuration to Toolmonitor‘s setup
- Reset: This button will reset the trigger configuration
- EEPROM: The configuration for “Trigger 1” can be permanently saved in the mechanical control system‘s EEPROM. The settings are made possible from this area of the triggers window

4.3.2.6. SerIO Communications

All open SerIO communications with the associated device can be listened to or the user‘s own messages can be sent with the help of this window.

The received data can be logged. Furthermore, messages, or sequences of messages, created by the user can be saved and loaded.



Time	Type	Command	Data	Comment
16:36:15.	Read	3B	00	Read trigger response
16:36:15.	Write	3B	3A	Read trigger
16:36:15.	Read	3B	00	Read trigger response
16:36:15.	Write	3B	3B	Read trigger
16:36:15.	Read	3B	00	Read trigger response
17:04:15.	Write	38		Read trigger
17:04:15.	Read	FF	01	Read trigger response
17:04:23.	Write	38	00	Read trigger
17:04:23.	Read	FF	01	Read trigger response

ReadTrigger : Reac

Name	Command	Data	Comment
ReadTrigger	38	00	Read trigger

Name: ReadTrigger Comment: Read trigger
Message: 38 00

Figure 21: SerIO Communications

User messages can be created using an entry dialog. The message type and data content can be entered to this purpose.

The “SerIO command” and the associated data can be entered using the “Message” area.

Clicking the “Execute” button will send the message.

Additional information about the messages can be stored in the “Name” and “Comment” entries, which can be saved by clicking the “Add” button.

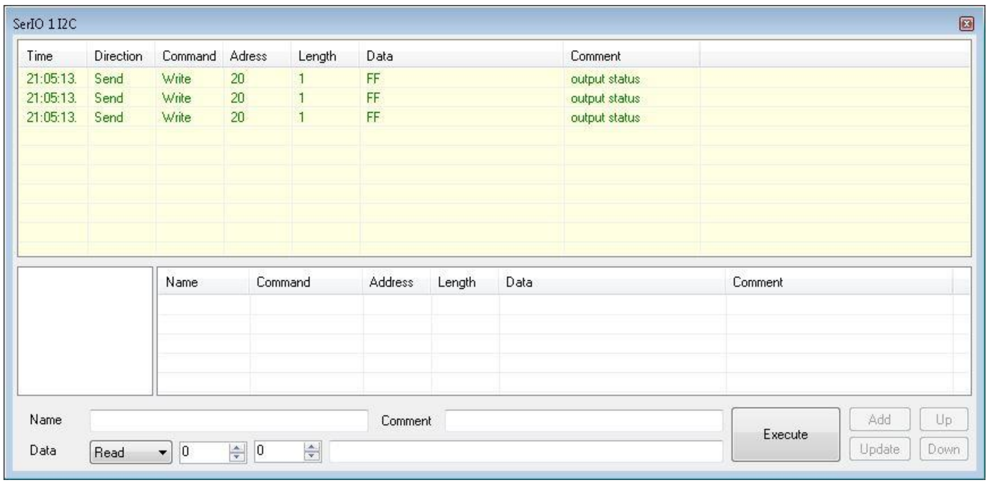
4.3.3. Adapter CPU

This SerIO device type represents the MCD mechanical control system “Adapter CPU”.

4.3.3.1. I^{2}C Communications

All open I^{2}C communications with the corresponding SerIO device can be listened to or the user‘s own messages can be sent with the help of this window.

The received data can be logged. Furthermore, messages, or sequences of messages, created by the user can be saved and loaded.


SerIO 1 I2C

Time	Direction	Command	Adress	Length	Data	Comment
21:05:13.	Send	Write	20	1	FF	output status
21:05:13.	Send	Write	20	1	FF	output status
21:05:13.	Send	Write	20	1	FF	output status

Name	Command	Address	Length	Data	Comment

Name: | Comment: | Data: Read, 0, 0

Figure 22: I^{2}C Communications

User messages can be created using an entry dialog. The message type and data content can be entered to this purpose.

The “I^{2}C command”, the “I^{2}C address” and the associated data can be entered using the “Message” area.

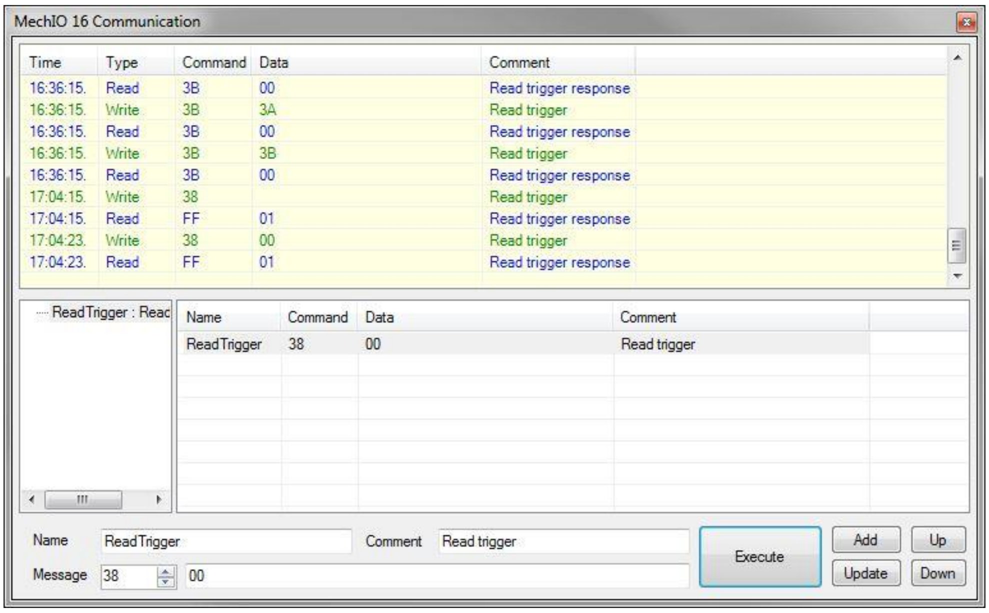
Clicking the “Execute” button will send the message.

Additional information about the messages can be stored in the “Name” and “Comment” entries, which can be saved by clicking the “Add” button.

4.3.3.2. SerIO Communications

All open SerIO communications with the associated device can be listened to or the user‘s own messages can be sent with the help of this window.

The received data can be logged. Furthermore, messages, or sequences of messages, created by the user can be saved and loaded.


MechIO 16 Communication

Time	Type	Command	Data	Comment
16:36:15.	Read	3B	00	Read trigger response
16:36:15.	Write	3B	3A	Read trigger
16:36:15.	Read	3B	00	Read trigger response
16:36:15.	Write	3B	3B	Read trigger
16:36:15.	Read	3B	00	Read trigger response
17:04:15.	Write	38		Read trigger
17:04:15.	Read	FF	01	Read trigger response
17:04:23.	Write	38	00	Read trigger
17:04:23.	Read	FF	01	Read trigger response

ReadTrigger : Reac

Name	Command	Data	Comment
ReadTrigger	38	00	Read trigger

Name: ReadTrigger | Comment: Read trigger | Message: 38, 00

Figure 23: SerIO Communications

User messages can be created using an entry dialog. The message type and data content can be entered to this purpose.

The “SerIO command” and the associated data can be entered using the “Message” area.

Clicking the “Execute” button will send the message.

Additional information about the messages can be stored in the “Name” and “Comment” entries, which can be saved by clicking the “Add” button.

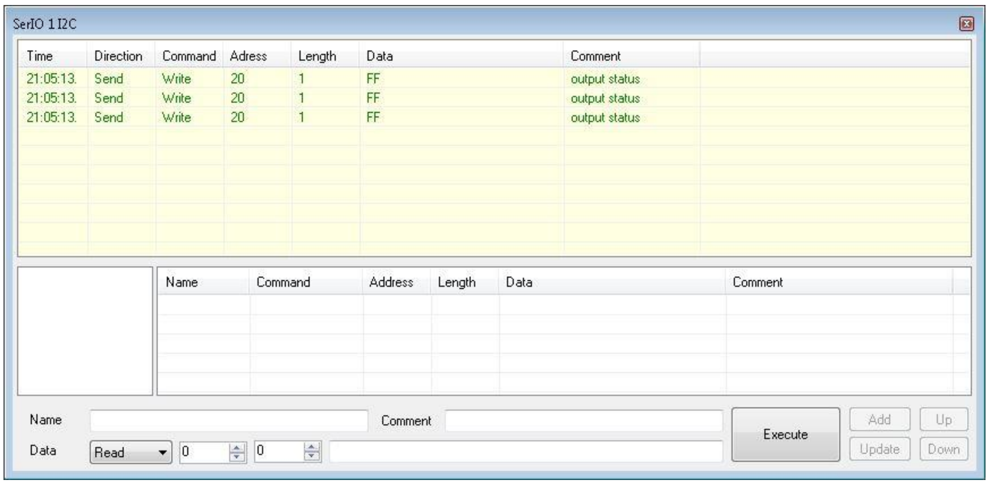
4.3.4. Pass / Fail Display

This SerIO device type represents the MCD “Pass / Fail display” system.

4.3.4.1. I^{2}C Communications

All open I^{2}C communications with the corresponding SerIO device can be listened to or the user‘s own messages can be sent with the help of this window.

The received data can be logged. Furthermore, messages, or sequences of messages, created by the user can be saved and loaded.



Time	Direction	Command	Adress	Length	Data	Comment
21:05:13.	Send	Write	20	1	FF	output status
21:05:13.	Send	Write	20	1	FF	output status
21:05:13.	Send	Write	20	1	FF	output status

Name	Command	Address	Length	Data	Comment

Figure 24: I^{2}C Communications

User messages can be created using an entry dialog. The message type and data content can be entered to this purpose.

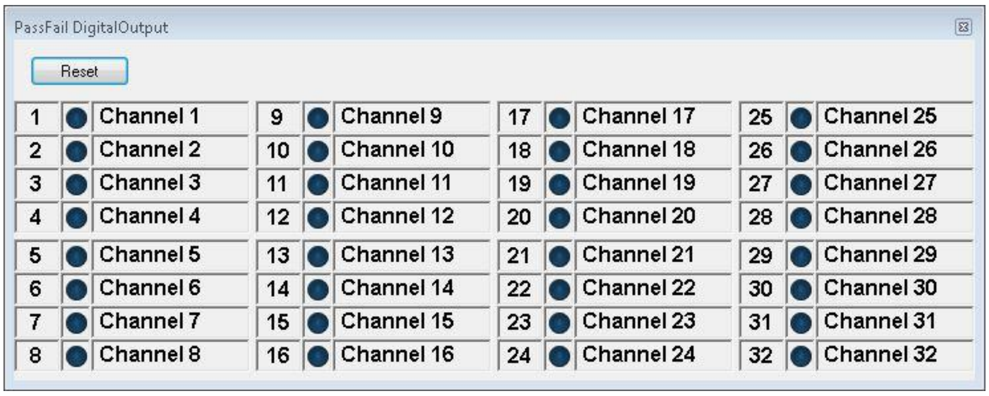
The “I^{2}C command”, the “I^{2}C address” and the associated data can be entered using the “Message” area.

Clicking the “Execute” button will send the message.

Additional information about the messages can be stored in the “Name” and “Comment” entries, which can be saved by clicking the “Add” button.

4.3.4.2. Pass / Fail Digital Output

The digital outputs from the “Pass / Fail display” system can be displayed and controlled using this dialog window. Clicking the “Reset” button will reset all outputs.



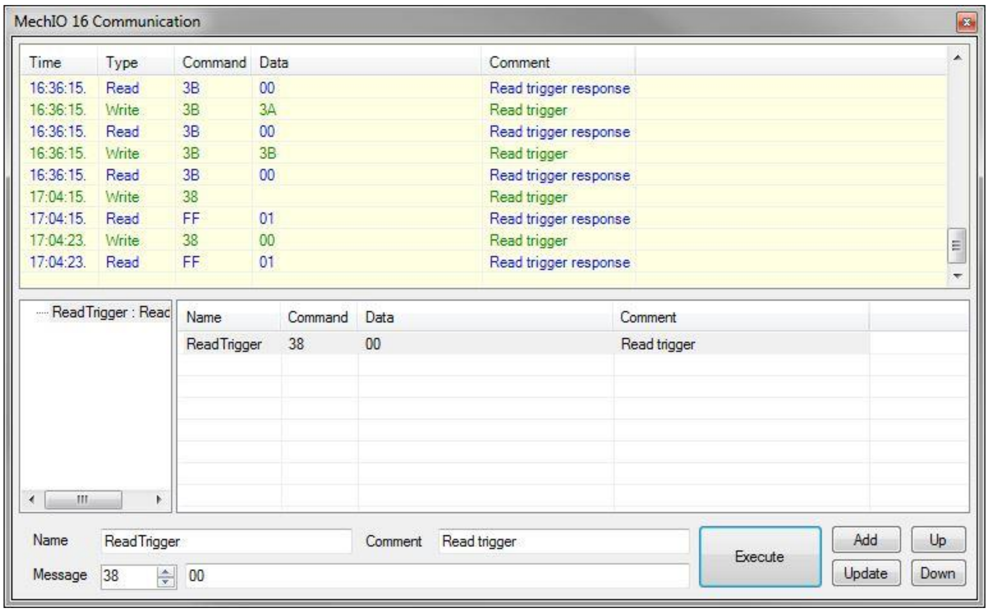
1	●	Channel 1	9	●	Channel 9	17	●	Channel 17	25	●	Channel 25
2	●	Channel 2	10	●	Channel 10	18	●	Channel 18	26	●	Channel 26
3	●	Channel 3	11	●	Channel 11	19	●	Channel 19	27	●	Channel 27
4	●	Channel 4	12	●	Channel 12	20	●	Channel 20	28	●	Channel 28
5	●	Channel 5	13	●	Channel 13	21	●	Channel 21	29	●	Channel 29
6	●	Channel 6	14	●	Channel 14	22	●	Channel 22	30	●	Channel 30
7	●	Channel 7	15	●	Channel 15	23	●	Channel 23	31	●	Channel 31
8	●	Channel 8	16	●	Channel 16	24	●	Channel 24	32	●	Channel 32

Figure 25: Digital Outputs

4.3.4.3. SerIO Communications

All open SerIO communications with the associated device can be listened to or the user‘s own messages can be sent with the help of this window.

The received data can be logged. Furthermore, messages, or sequences of messages, created by the user can be saved and loaded.



Time	Type	Command	Data	Comment
16:36:15.	Read	3B	00	Read trigger response
16:36:15.	Write	3B	3A	Read trigger
16:36:15.	Read	3B	00	Read trigger response
16:36:15.	Write	3B	3B	Read trigger
16:36:15.	Read	3B	00	Read trigger response
17:04:15.	Write	38		Read trigger
17:04:15.	Read	FF	01	Read trigger response
17:04:23.	Write	38	00	Read trigger
17:04:23.	Read	FF	01	Read trigger response

Name	Command	Data	Comment
ReadTrigger	38	00	Read trigger

Name: ReadTrigger
Comment: Read trigger
Message: 38 00

Figure 26: SerIO Communications

User messages can be created using an entry dialog. The message type and data content can be entered to this purpose.

The “SerIO command” and the associated data can be entered using the “Message” area.

Clicking the “Execute” button will send the message.

Additional information about the messages can be stored in the “Name” and “Comment” entries, which can be saved by clicking the “Add” button

4.4. I²C Devices

A menu item will be added to Toolmonitor SerIO‘s “Windows” menu for each I²C device. A variety of windows for controlling the I²C device can be opened using this item.

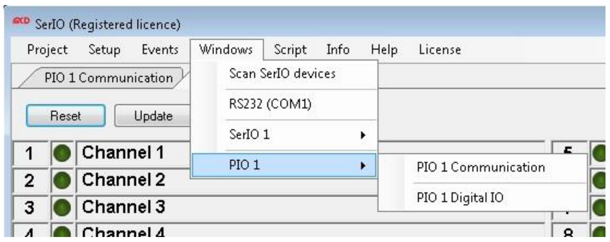


Figure 27: I²C Devices Sub Menu

4.4.1. I²C Digitale IO Devices (PIO)

A number of I²C devices serve digital signals to input or output. They are based on one or more I²C PIO systems and are configured in the same manner.

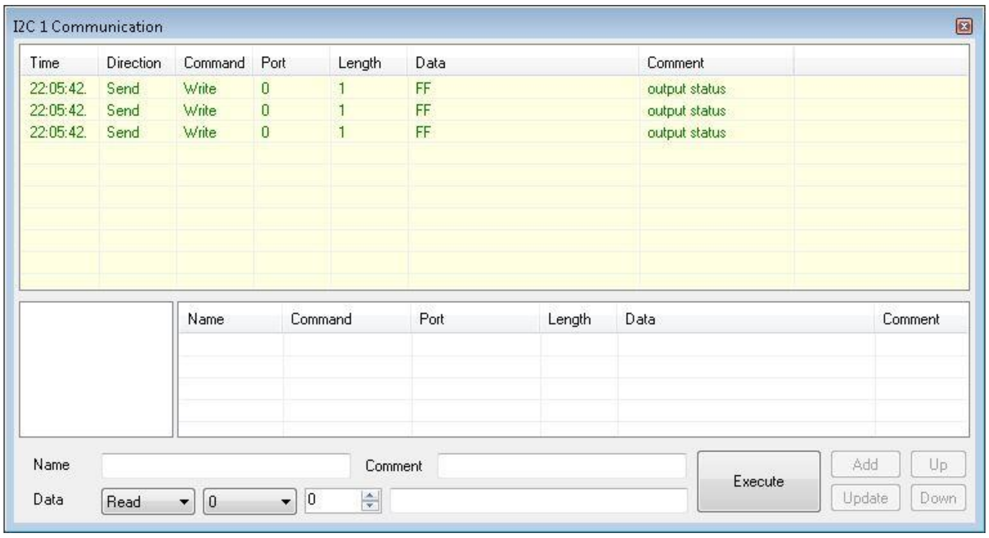
This applies to the following devices:

- PCF8574 | PIO 8 channels per port
- PCF8574A | PIO 8 channels per port
- PCA9554 | PIO 8 channels per port
- PCA9554A | PIO 8 channels per port
- PIO 3 x PCA 9554 | B.140.010.110
- PIO 3 x PCA 9554A | B.140.010.210
- PIO 3 x PCF 8574 | B.140.010.310
- PIO 3 x PCF 8574A | B.140.010.410
- RelMUX16 | Relay multiplexer 16 channels 7442

4.4.1.1. I^{2}C Communications

Open I^{2}C communications with the corresponding I^{2}C modules can be listened to or the user‘s own messages can be sent with the help of this window.

The received data can be logged. Furthermore, messages, or sequences of messages, created by the user can be saved and loaded.



I2C 1 Communication

Time	Direction	Command	Port	Length	Data	Comment
22:05:42	Send	Write	0	1	FF	output status
22:05:42	Send	Write	0	1	FF	output status
22:05:42	Send	Write	0	1	FF	output status

Name	Command	Port	Length	Data	Comment

Name: Comment: Data: Read 0 0

Figure 28: I^{2}C Communications

User messages can be created using an entry dialog. The message type and data content can be entered to this purpose.

The “I^{2}C command”, the “I^{2}C bank” and the associated data can be entered using the “Message” area.

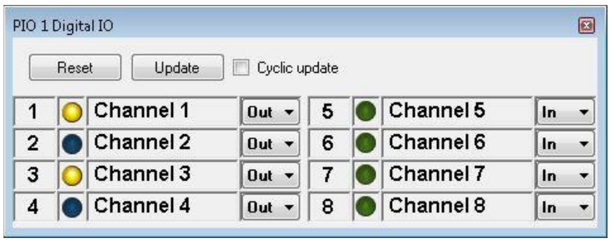
Clicking the “Execute” button will send the message.

Additional information about the messages can be stored in the “Name” and “Comment” entries, which can be saved by clicking the “Add” button.

4.4.1.2. PIO Digital IO

The “I^{2}C PIO modules” digital inputs and outputs can be displayed and controlled using this dialog window. Clicking the “Reset” button will reset all outputs. Clicking the “Update” button will update the current state of the inputs and outputs. Checking the “Cyclic update” checkbox will update the inputs and outputs at periodic intervals.

The operating mode (input or output) can be switched online depending on the configuration of the PIO module. This option is not available for all PIO modules and depends on the configuration.



PIO 1 Digital IO

1	●	Channel 1	Out	5	●	Channel 5	In
2	●	Channel 2	Out	6	●	Channel 6	In
3	●	Channel 3	Out	7	●	Channel 7	In
4	●	Channel 4	Out	8	●	Channel 8	In

Figure 29: Digital Inputs and Outputs

4.4.2. Power Supply Modules

A number of I^2C devices serve to provide power through special power supply modules and the regulating component can adjust the frequency.

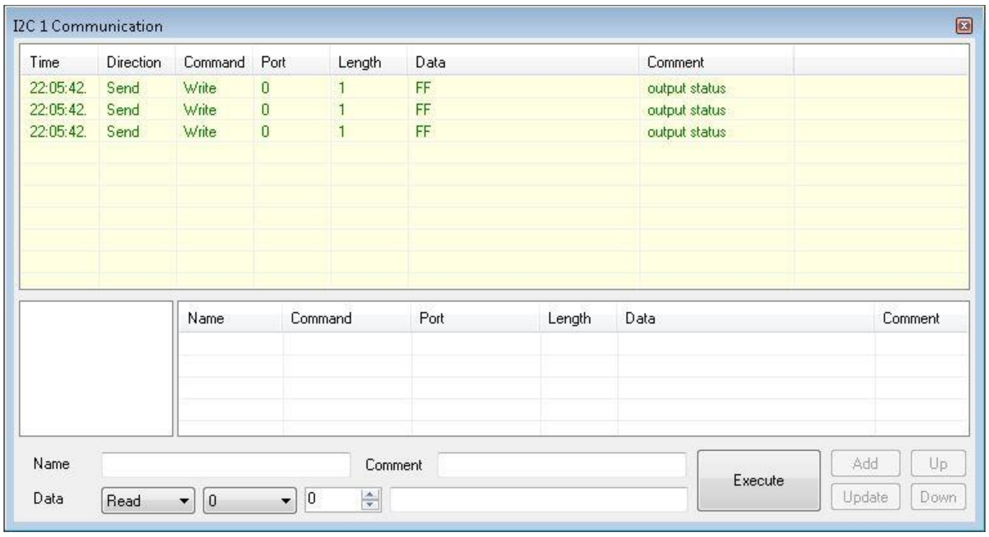
This applies to the following devices:

- PST30M | Power supply for tuner applications master B.170.030.112
- PST30S | Power supply for tuner applications slave B.170.030.212
- PST30L | Power supply for tuner applications linear slave B.170.030.412

4.4.2.1. I^2C Communications

Open I^2C communications with the corresponding I^2C modules can be listened to or the user‘s own messages can be sent with the help of this window.

The received data can be logged. Furthermore, messages, or sequences of messages, created by the user can be saved and loaded.



Time	Direction	Command	Port	Length	Data	Comment
22:05:42.	Send	Write	0	1	FF	output status
22:05:42.	Send	Write	0	1	FF	output status
22:05:42.	Send	Write	0	1	FF	output status

Name	Command	Port	Length	Data	Comment

Figure 30: I^2C Communications

User messages can be created using an entry dialog. The message type and data content can be entered to this purpose.

The “I^2C command”, the “I^2C bank” and the associated data can be entered using the “Message” area.

Clicking the “Execute” button will send the message.

Additional information about the messages can be stored in the “Name” and “Comment” entries, which can be saved by clicking the “Add” button.

4.4.2.2. Analog Power Supply Voltage Meter

The measured voltage can be displayed using this dialog window. The display range can be set using a context menu.

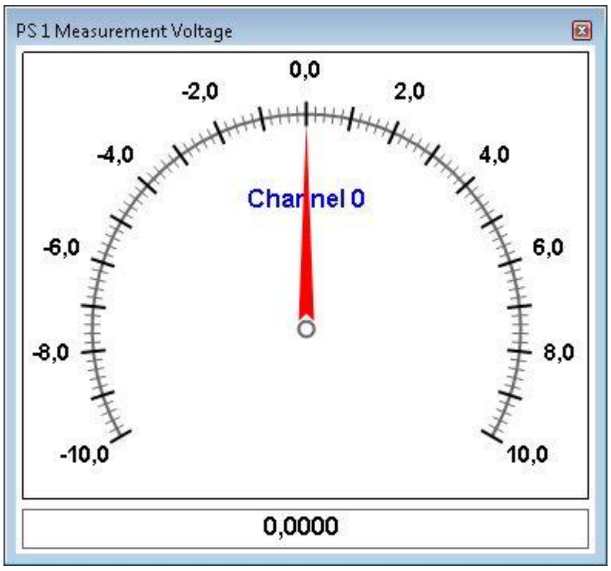


Figure 31: Analog Power Supply Voltage Meter

4.4.2.3. Analog Power Supply Current Meter

The measured current can be displayed using this dialog window. The display range can be set using a context menu.

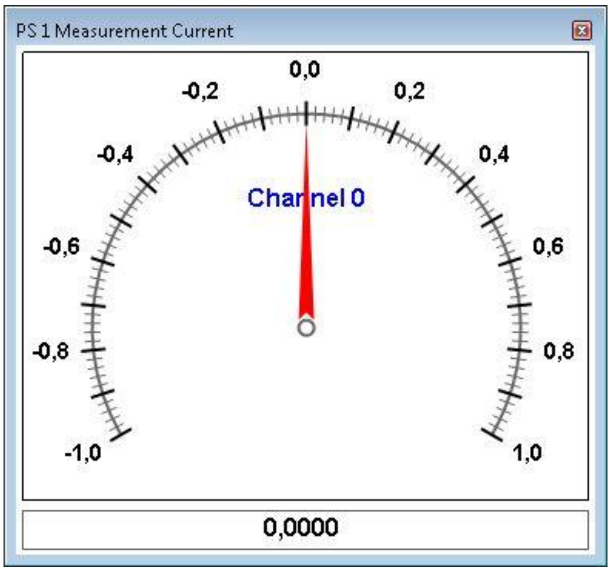


Figure 32: Analog Power Supply Current Meter

4.4.2.4. Power Supply Settings

The associated power supply can be controlled using this dialog window. The power and the “Sense” feature can be activated and deactivated from here.

In addition, the frequency of the controller module can be set for a “Master” module. This option is not available for all modules.

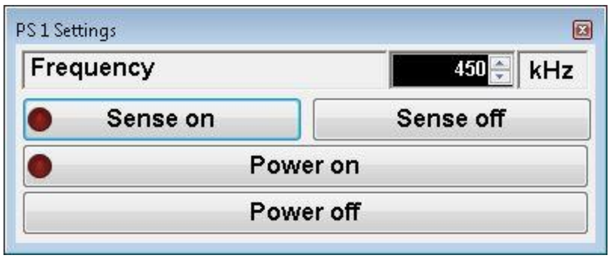


Figure 33: Power Supply Settings

4.4.2.5. Power Supply Measurement Dialog

The voltage and current for the associated power supply module can be scanned and displayed using this dialog window.

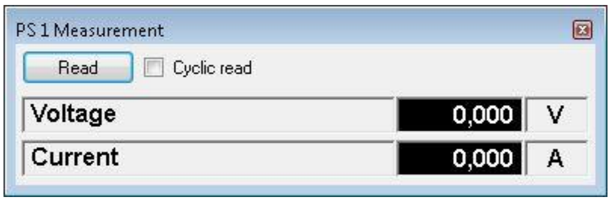


Figure 34: Power Supply Measurement Dialog

5. Example Configuration (Quick Start Guide)

The section provides a step - by - step description as an example for the configuration of the Toolmonitor SerIO. In this example, it is assumed that an MCD mechanical control system MechIO16 has been connected to the PC‘s “COM1” port. In addition, a relay multiplexer (such as the RelMUX16 | Relay multiplexer 16 channels 7442) should be connected to the mechanical control system‘s I^{2}C connection

5.1. Load the Sample Preset

To reset the configuration, load the “Default” preset from the “Presets” menu item on the “Project” menu

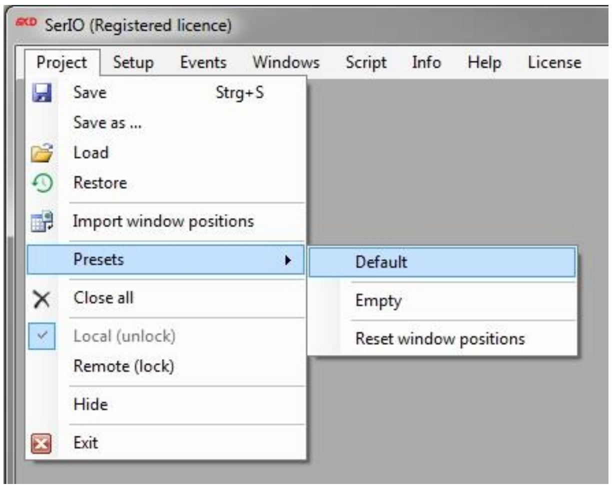


Figure 35: Load the Sample Preset

5.2. Search for Connected Devices

To find the connected devices, select the “Scan SerIO Devices” menu item on the “Windows” menu.

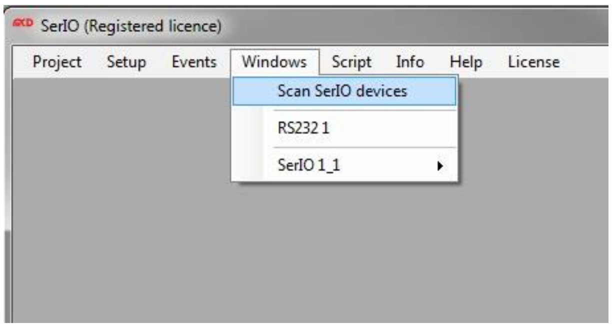


Figure 36: Scan Connected Devices

This will trigger a search for connected devices, which will then be displayed:

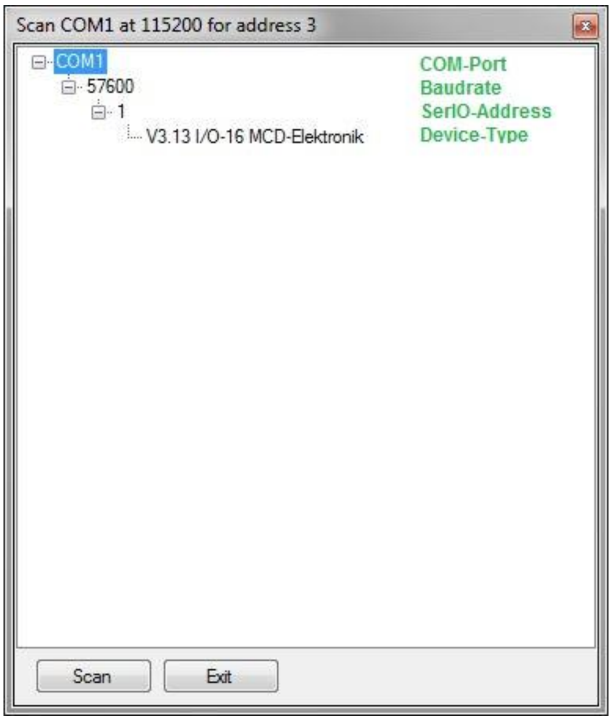
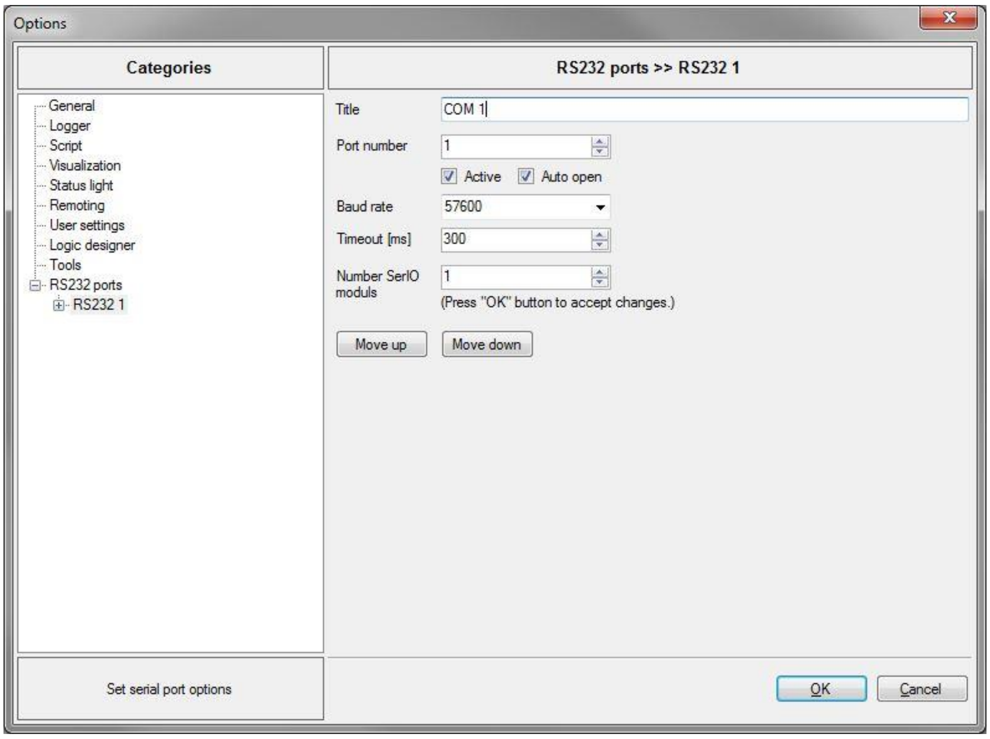

Scan COM1 at 115200 for address 3
COM1
57600
1
V3.13 I/O-16 MCD-Elektronik
COM-Port
Baudrate
SerIO-Address
Device-Type
Scan
Exit

Figure 37: Connected Devices Dialog

5.3. Configuring RS232 Port

The “Options” dialog can be opened from the “Setup” menu and settings made for the RS232 port. Click the “OK” button to close the “Options” dialog.

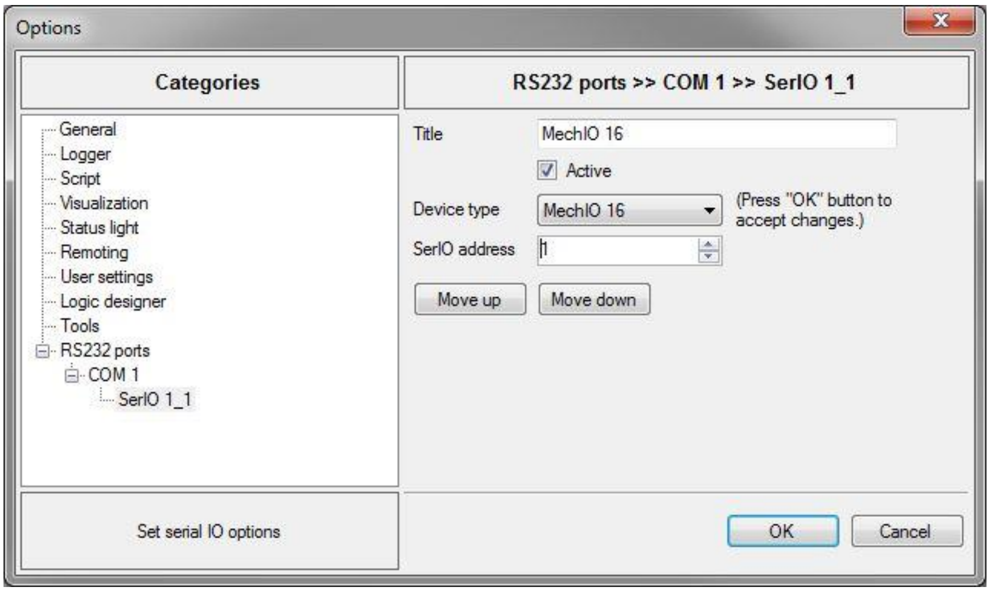


Title: COM 1
Port number: 1
[✓ Active] [✓ Auto open]
Baud rate: 57600
Timeout [ms]: 300
Number SerIO moduls: 1
(Press "OK" button to accept changes.)

Figure 38: Configuring RS232 Port

5.4. Configuring SerIO Devices

Open the “Options” dialog again and make the settings for the SerIO device. Click the “OK” button to close the “Options” dialog.



RS232 ports >> COM 1 >> SerIO 1_1

Title: MechIO 16

☑ Active

Device type: MechIO 16

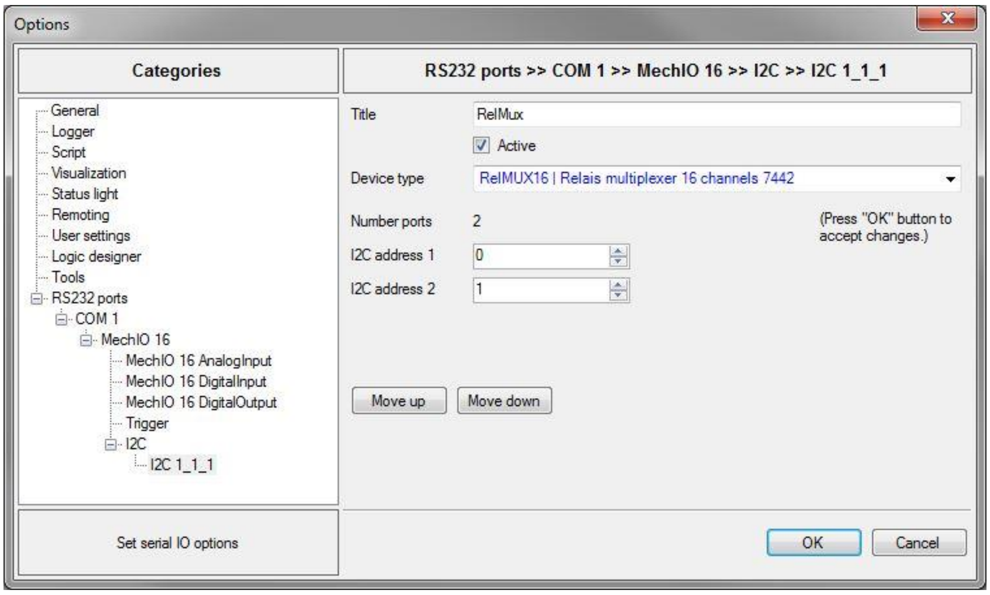
(Press "OK" button to accept changes.)

SerIO address: 1

Figure 39: Configuring SerIO Devices

5.5. Configuring I^{2}C Device (RelMUX16)

Open the “Options” dialog again and make the settings for the I^{2}C device (RelMUX16). Click the “OK” button to close the “Options” dialog.



RS232 ports >> COM 1 >> MechIO 16 >> I2C >> I2C 1_1_1

Title: RelMux

☑ Active

Device type: RelMUX16 | Relais multiplexer 16 channels 7442

Number ports: 2

(Press "OK" button to accept changes.)

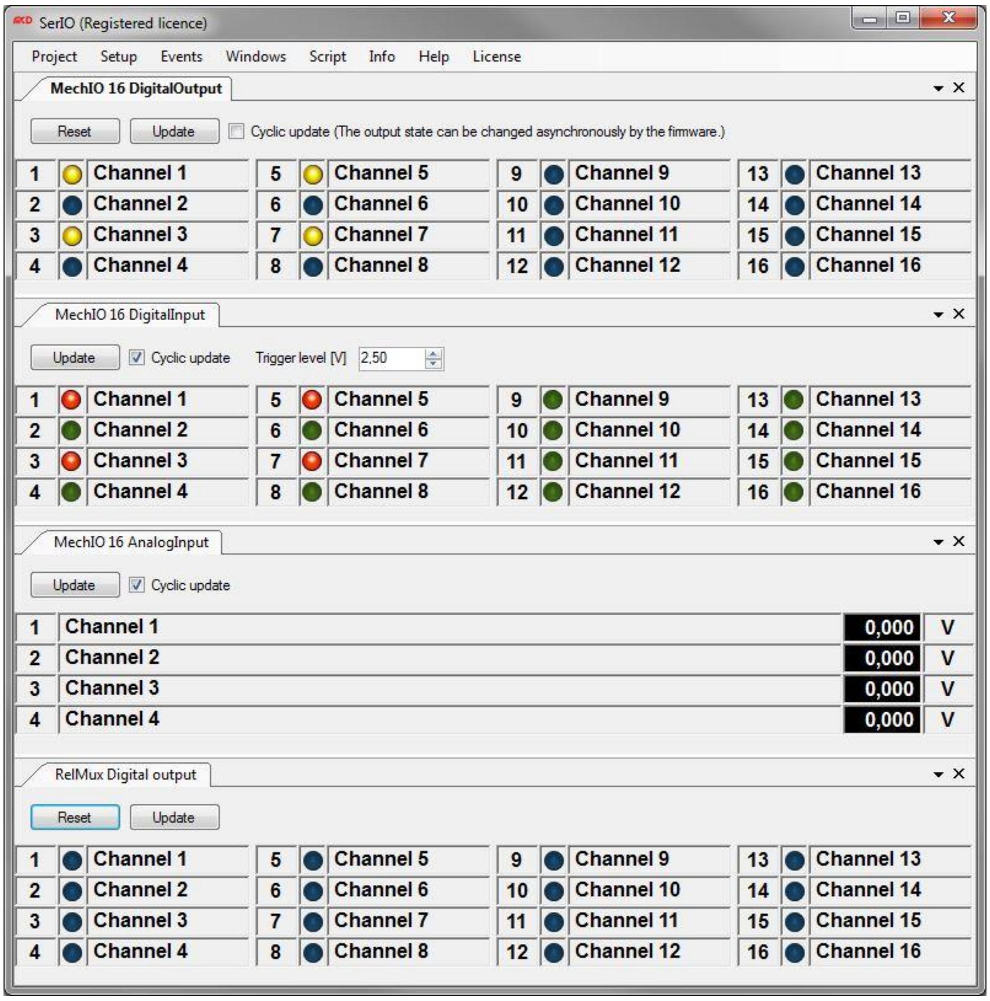
I2C address 1: 0

I2C address 2: 1

Figure 40: Configuring I^{2}C device (RelMUX16)

5.6. Displaying the Digital and Analog Inputs and Outputs

The individual tab pages for the digital and analog inputs and outputs can then be selected and displayed using the menu items under the “Windows” menu. The forms can then be dragged to any position on the screen.



MechIO 16 DigitalOutput

Reset | Update | ☐ Cyclic update (The output state can be changed asynchronously by the firmware.)

1	🟡	Channel 1	5	🟡	Channel 5	9	🔵	Channel 9	13	🔵	Channel 13
2	🔵	Channel 2	6	🔵	Channel 6	10	🔵	Channel 10	14	🔵	Channel 14
3	🟡	Channel 3	7	🟡	Channel 7	11	🔵	Channel 11	15	🔵	Channel 15
4	🔵	Channel 4	8	🔵	Channel 8	12	🔵	Channel 12	16	🔵	Channel 16

MechIO 16 DigitalInput

Update | ☑ Cyclic update | Trigger level [V] 2,50

1	🔴	Channel 1	5	🔴	Channel 5	9	🟢	Channel 9	13	🟢	Channel 13
2	🟢	Channel 2	6	🟢	Channel 6	10	🟢	Channel 10	14	🟢	Channel 14
3	🔴	Channel 3	7	🔴	Channel 7	11	🟢	Channel 11	15	🟢	Channel 15
4	🟢	Channel 4	8	🟢	Channel 8	12	🟢	Channel 12	16	🟢	Channel 16

MechIO 16 AnalogInput

Update | ☑ Cyclic update

1	Channel 1	0,000	V
2	Channel 2	0,000	V
3	Channel 3	0,000	V
4	Channel 4	0,000	V

RelMux Digital output

Reset | Update

1	🔵	Channel 1	5	🔵	Channel 5	9	🔵	Channel 9	13	🔵	Channel 13
2	🔵	Channel 2	6	🔵	Channel 6	10	🔵	Channel 10	14	🔵	Channel 14
3	🔵	Channel 3	7	🔵	Channel 7	11	🔵	Channel 11	15	🔵	Channel 15
4	🔵	Channel 4	8	🔵	Channel 8	12	🔵	Channel 12	16	🔵	Channel 16

Figure 41: Displaying the Digital and Analog Inputs and Outputs

6. Project Management

The current settings and the Toolmonitor layout can be saved and loaded from the menu options under the “Project” menu. All windows can be positioned freely and adjusted according to the user‘s own needs.

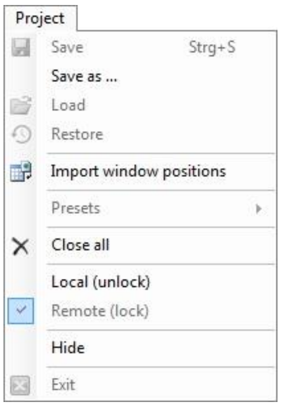


Figure 42: Project Menu

The “Project” menu consists of the following areas:

- Save / Save as: All current settings can be saved in a project file by clicking the “Save” menu item. Even the current window positions will be stored in the file
- Load: Previously saved settings can be re - loaded by clicking the “Load” menu item. Even the original window positions will be recreated
- Import window positions: The “Import Window Positions” menu item allows the window positions to be imported from a saved setup file. All original settings will not be affected by this
- Presets: Predefined settings can be set using this menu item
- Close all: This menu item will close all open forms. The Toolmonitor will continue to run
- Local: If the Toolmonitor is remote controlled or the “Setup” has been protected with a password, most user actions are protected against direct entry. Activating “Local” mode removes this protection and all operating controls will be accessible again. If the setup process included the entry of a password, it must be entered to release the Toolmonitor
- Remote: Clicking this menu item will return Toolmonitor to protected mode
- Hide: Clicking this menu item will hide the Toolmonitor, but keep it running. If it is not controlled remotely, Toolmonitor can be re - activated using an icon on the taskbar
- Exit: Clicking this menu item will terminate the Toolmonitor (item will be disabled when Toolmonitor is controlled remotely)

7. Events

The logging and trace message dialogs can be accessed from this menu.

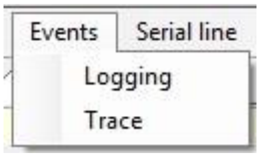
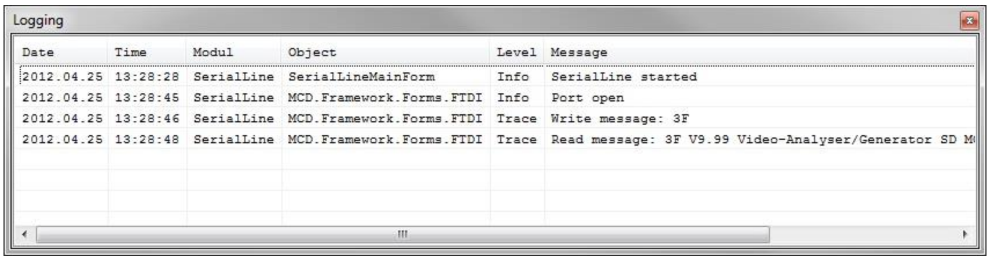


Figure 43: Events Menu

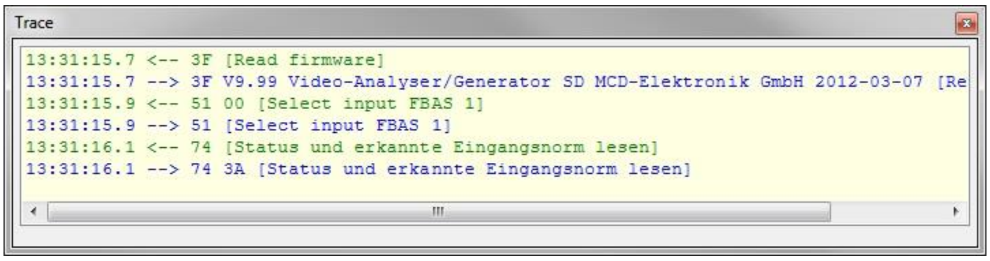
The Events menu consists of the following areas:

- Logging: The logging messages for general events, warnings, errors and so forth will be displayed using this menu option
- Trace: The trace messages (sent or received messages) will be displayed using this menu option



Date	Time	Modul	Object	Level	Message
2012.04.25	13:28:28	SerialLine	SerialLineMainForm	Info	SerialLine started
2012.04.25	13:28:45	SerialLine	MCD.Framework.Forms.FTDI	Info	Port open
2012.04.25	13:28:46	SerialLine	MCD.Framework.Forms.FTDI	Trace	Write message: 3F
2012.04.25	13:28:48	SerialLine	MCD.Framework.Forms.FTDI	Trace	Read message: 3F V9.99 Video-Analyser/Generator SD M

Figure 44: Log Monitor



```
13:31:15.7 <-- 3F [Read firmware]
13:31:15.7 --> 3F V9.99 Video-Analyser/Generator SD MCD-Elektronik GmbH 2012-03-07 [Re
13:31:15.9 <-- 51 00 [Select input FBAS 1]
13:31:15.9 --> 51 [Select input FBAS 1]
13:31:16.1 <-- 74 [Status und erkannte Eingangsnorm lesen]
13:31:16.1 --> 74 3A [Status und erkannte Eingangsnorm lesen]
```

Figure 45: Trace Monitor

8. Setup

The Setup menu provides access to the project - specific options and registration of the Toolmonitor as a COM server.

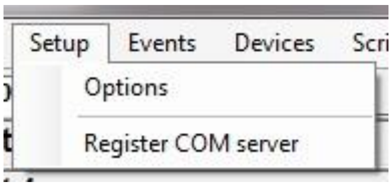


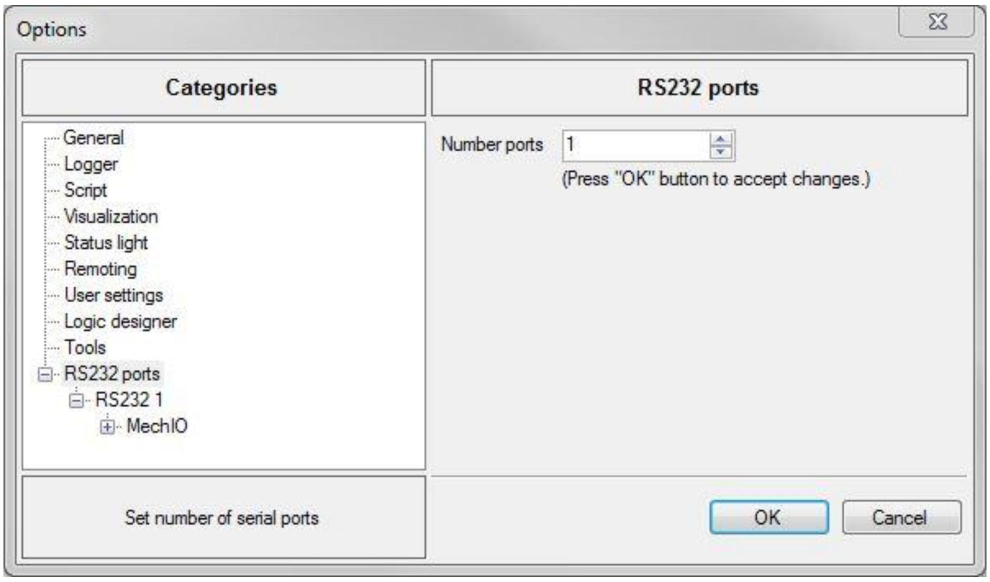
Figure 46: Setup Menu

8.1. RS232 Ports

The serial interfaces are configured from here.

8.1.1. Number of RS232 Ports

The number of RS232 interfaces to be used by the Toolmonitor SerIO will be determined from here.



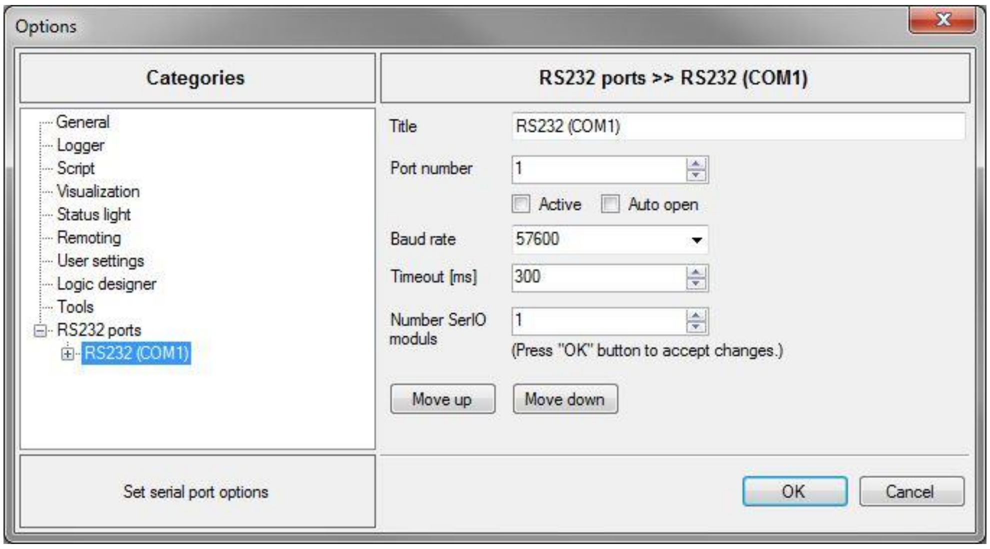
Number ports: 1

(Press "OK" button to accept changes.)

Figure 47: Setting the Number of RS232 Ports

8.1.2. Communications Parameters

The communications parameters can be entered from here for each RS232 port to be used.



Title: RS232 (COM1)
Port number: 1
Active
Auto open
Baud rate: 57600
Timeout [ms]: 300
Number SerIO moduls: 1
(Press "OK" button to accept changes.)

Figure 48: Configuring a Serial Port

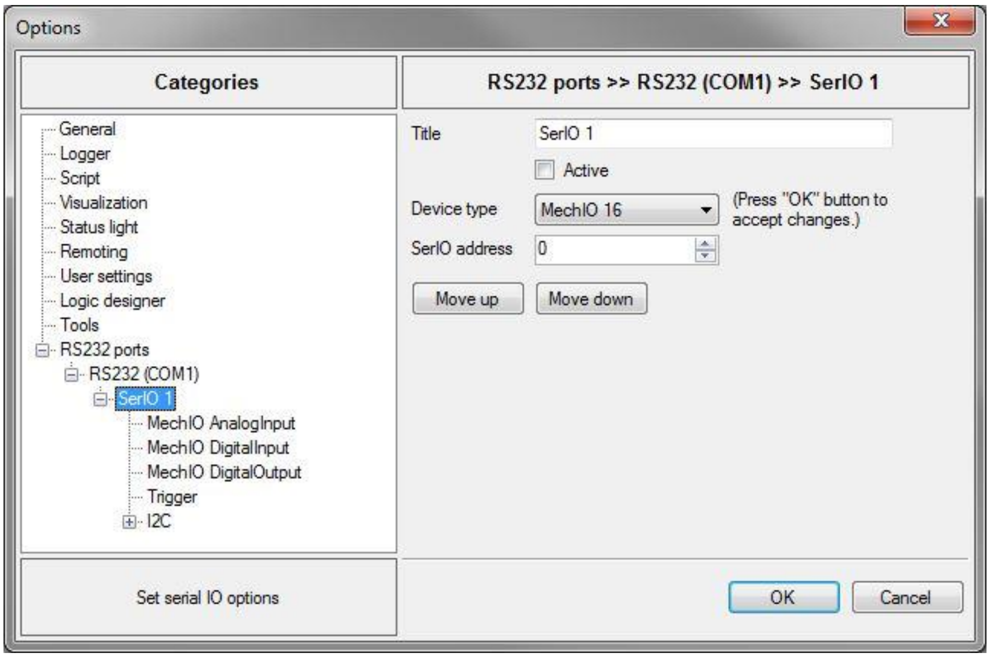
The following areas are to be configured:

- Title: A name used for the port for display and control purposes
- Port number: The COM port to be used
- Active: A “Flag” indicating that the port can be used
- Auto open: A “Flag” indicating that the port should be opened automatically when starting the Toolmonitor. (Note: the port can be opened even when this “Flag” has been disabled, if a supporting device requires the port to be opened.)
- Timeout: The maximum time frame (in milliseconds) before a sending or receiving process will time out and the process must be closed
- Number SerIO moduls: The number of SerIO modules that have been connected to this RS232 port
- Move up / Move down: The selected RS232 port can be moved up or down in the list of RS232 ports by clicking these buttons

8.2. SerIO Devices

The type and the SerIO address can be chosen from here for each SerIO device to be used. The following types are available:

- Unknown (core functions for non - specific customer requests)
- MechIO 8
- MechIO 16
- Adapter CPU
- Pass / Fail display



Title: SerIO 1

Active

Device type: MechIO 16

(Press "OK" button to accept changes.)

SerIO address: 0

Figure 49: Configuring a SerIO Device

The following areas are to be configured:

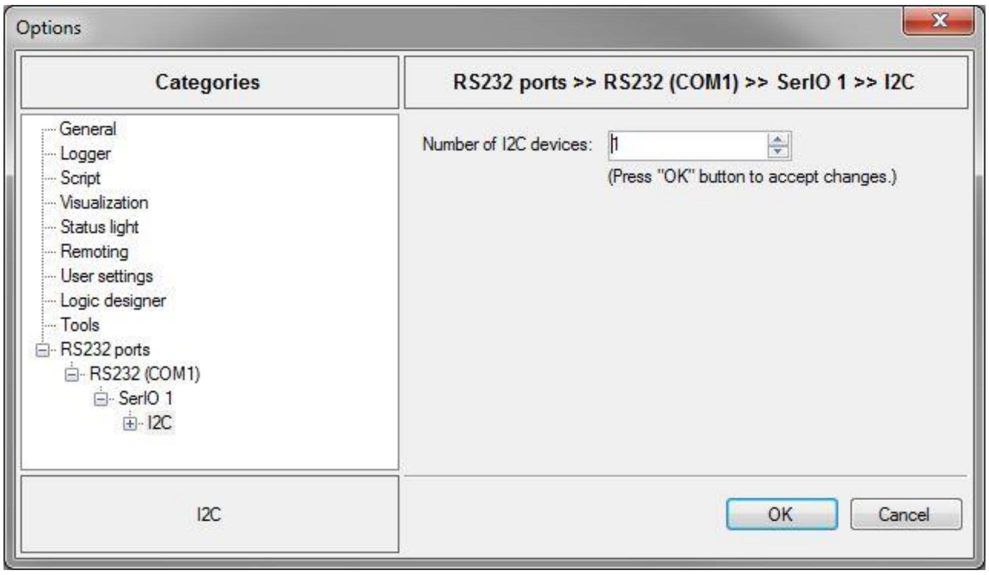
- Title: A name used for a SerIO device for display and control purposes
- Active: A “Flag” indicating that the port can be used
- Device type: The SerIO device type (see above)
- SerIO address: The SerIO address used for the SerIO device (each SerIO device connected to a shared RS232 port must be assigned a unique SerIO address)
- Move up / Move down: The selected SerIO device can be moved up or down in the list of SerIO devices by clicking these buttons

8.2.1. MechIO 8

This SerIO device type represents the MCD mechanical control system “MechIO 8”.

8.2.1.1. I^2C

The number of I^2C ports to be controlled by the MechIO 8 will be determined by this setting



RS232 ports >> RS232 (COM1) >> SerIO 1 >> I2C

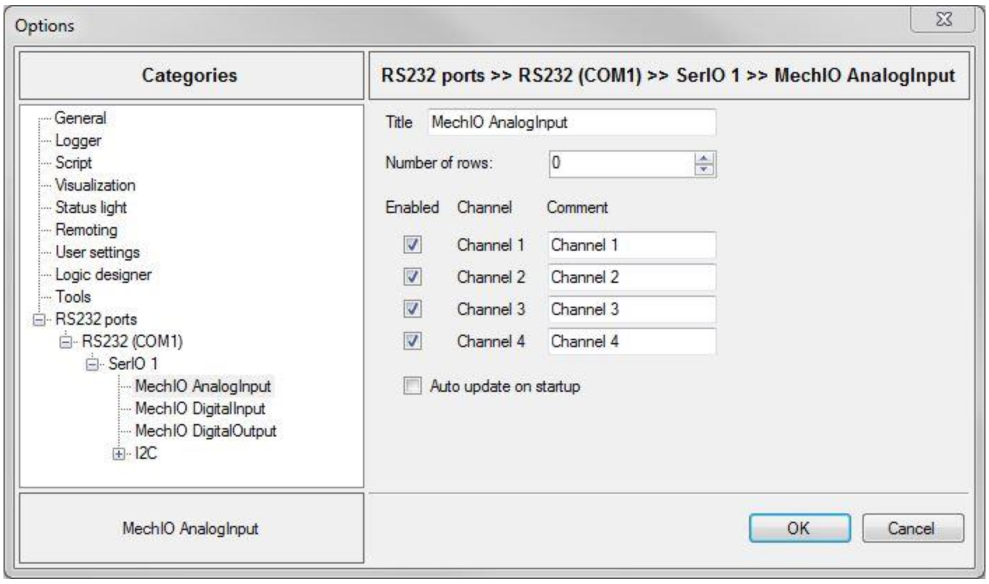
Number of I2C devices: 1

(Press "OK" button to accept changes.)

Figure 50: Setting the Number of I^2C Ports

8.2.1.2. MechIO 8 Analog Input

The analog inputs to the mechanical control system are configured using this dialog window.



RS232 ports >> RS232 (COM1) >> SerIO 1 >> MechIO AnalogInput

Title: MechIO AnalogInput

Number of rows: 0

Enabled	Channel	Comment
✓	Channel 1	Channel 1
✓	Channel 2	Channel 2
✓	Channel 3	Channel 3
✓	Channel 4	Channel 4

☐ Auto update on startup

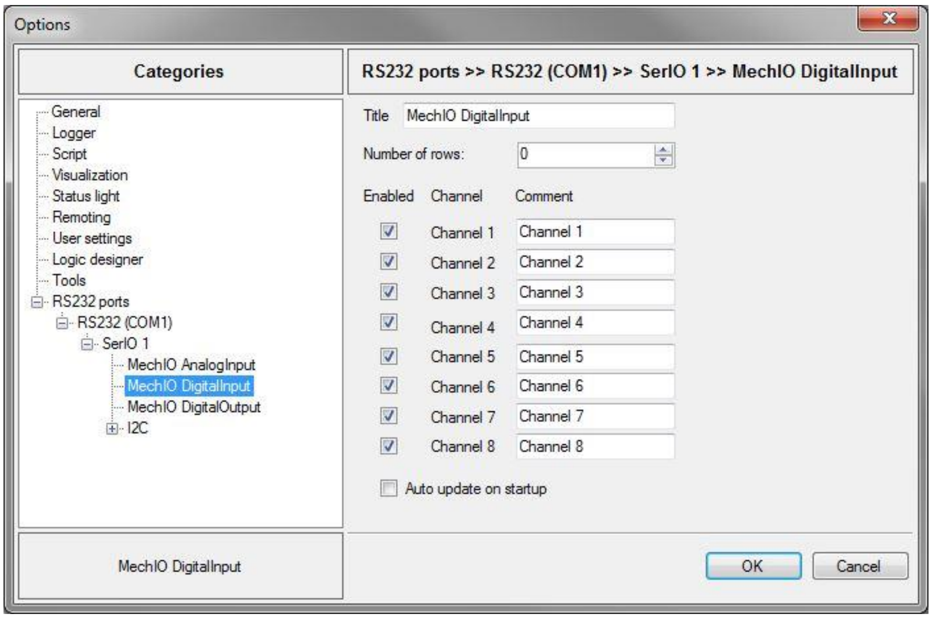
Figure 51: Configuring the Analog Inputs

The following areas are to be configured:

- Title: A name used for an analog input for display and control purposes
- Number of rows: The number of rows used to display the input channels. If the value is set to “0”, the number of rows and columns will be determined automatically in order to save as much display space as possible
- Enabled: This setting indicates whether or not the indicated channel should be displayed (channels that are not in use are hidden)
- Comment: A channel name, which will be used for display and control purposes, can be assigned to each channel
- Auto update on startup: If this “Flag” has been enabled, the indicated communications channel will be opened automatically when the Toolmonitor starts, and the status for the input channels will be scanned and displayed once

8.2.1.3. MechIO 8 Digital Input

The digital inputs to the mechanical control system are configured using this dialog window.



RS232 ports >> RS232 (COM1) >> SerIO 1 >> MechIO DigitalInput

Title: MechIO DigitalInput

Number of rows: 0

Enabled	Channel	Comment
✓	Channel 1	Channel 1
✓	Channel 2	Channel 2
✓	Channel 3	Channel 3
✓	Channel 4	Channel 4
✓	Channel 5	Channel 5
✓	Channel 6	Channel 6
✓	Channel 7	Channel 7
✓	Channel 8	Channel 8

☐ Auto update on startup

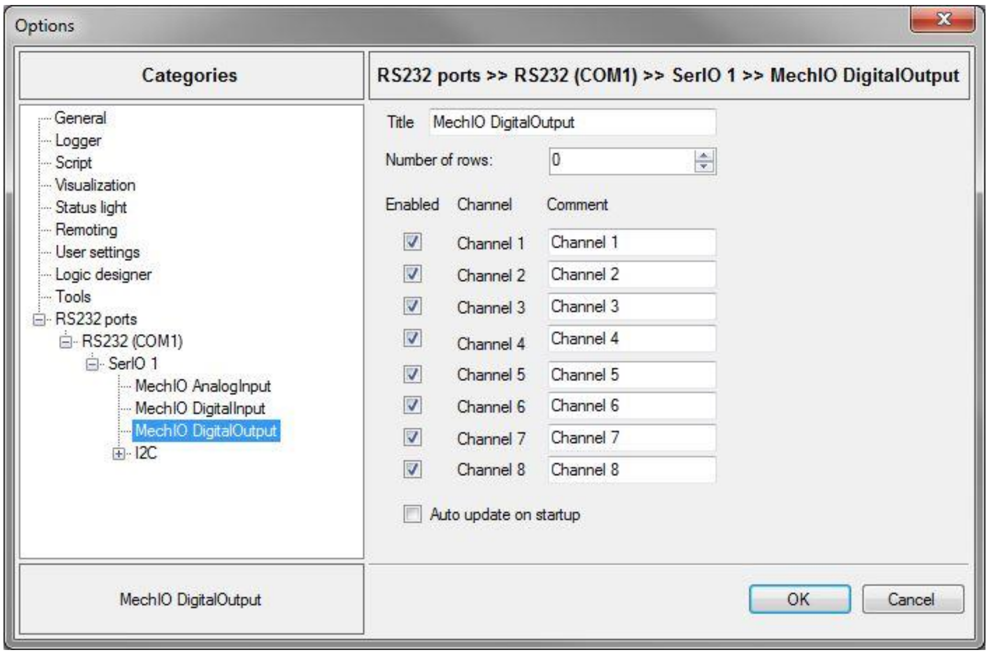
Figure 52: Configuring the Digital Inputs

The following areas are to be configured:

- Title: A name used for a digital input for display and control purposes
- Number of rows: The number of rows used to display the input channels. If the value is set to “0”, the number of rows and columns will be determined automatically in order to save as much display space as possible
- Enabled: This setting indicates whether or not the indicated channel should be displayed (channels that are not in use are hidden)
- Comment: A channel name, which will be used for display and control purposes, can be assigned to each channel
- Auto update on startup: If this “Flag” has been enabled, the indicated communications channel will be opened automatically when the Toolmonitor starts, and the status for the input channels will be scanned and displayed once

8.2.1.4. MechIO 8 Digital Ouput

The digital outputs from the mechanical control system are configured using this dialog window.



Title: MechIO DigitalOutput

Number of rows: 0

Enabled	Channel	Comment
✓	Channel 1	Channel 1
✓	Channel 2	Channel 2
✓	Channel 3	Channel 3
✓	Channel 4	Channel 4
✓	Channel 5	Channel 5
✓	Channel 6	Channel 6
✓	Channel 7	Channel 7
✓	Channel 8	Channel 8

☐ Auto update on startup

Figure 53: Configuring the Digital Outputs

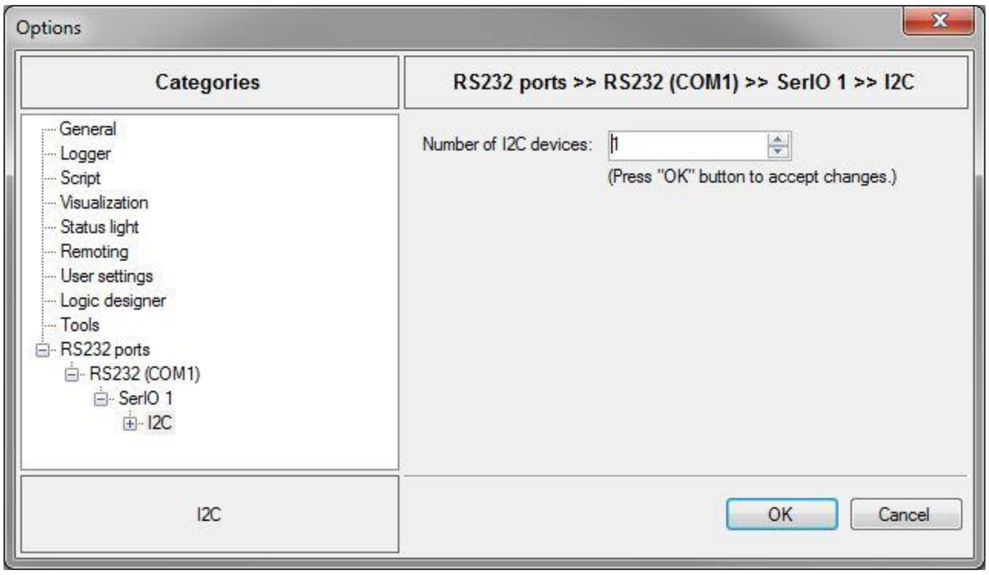
The following areas are to be configured:

- Title: A name used for a digital output for display and control purposes
- Number of rows: The number of rows used to display the output channels. If the value is set to “0”, the number of rows and columns will be determined automatically in order to save as much display space as possible
- Enabled: This setting indicates whether or not the indicated channel should be displayed (channels that are not in use are hidden)
- Comment: A channel name, which will be used for display and control purposes, can be assigned to each channel
- Auto update on startup: If this “Flag” has been enabled, the indicated communications channel will be opened automatically when the Toolmonitor starts, and the status for the output channels will be scanned and displayed once

8.2.2. MechIO 16

8.2.2.1. I^{2}C

The number of I^{2}C ports to be controlled by the MechIO 16 will be determined by this setting.



Options

RS232 ports >> RS232 (COM1) >> SerIO 1 >> I2C

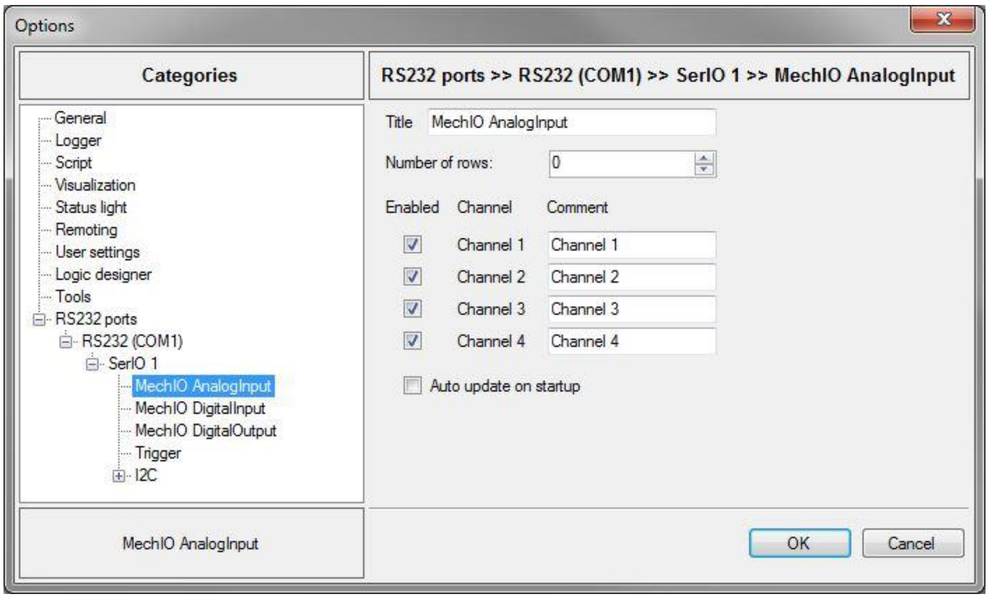
Number of I2C devices: 1

(Press "OK" button to accept changes.)

Figure 54: Setting the Number of I^{2}C Ports

8.2.2.2. Mech IO 16 Analog Input

The analog inputs to the mechanical control system are configured using this dialog window.



Options

RS232 ports >> RS232 (COM1) >> SerIO 1 >> MechIO AnalogInput

Title: MechIO AnalogInput

Number of rows: 0

Enabled	Channel	Comment
✓	Channel 1	Channel 1
✓	Channel 2	Channel 2
✓	Channel 3	Channel 3
✓	Channel 4	Channel 4

☐ Auto update on startup

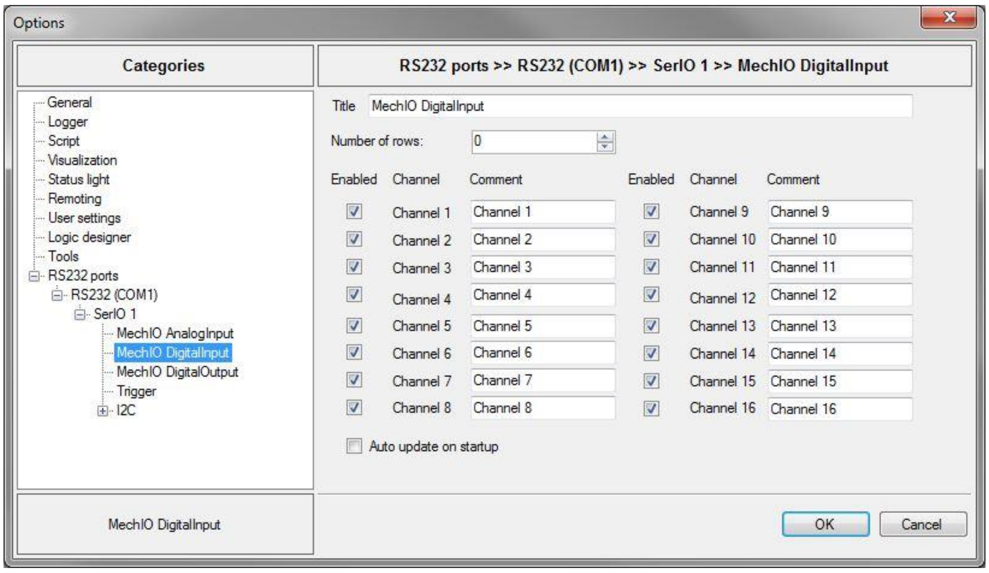
Figure 55: Configuring the Analog Inputs

The following areas are to be configured:

- Title: A name used for an analog input for display and control purposes
- Number of rows: The number of rows used to display the input channels. If the value is set to “0”, the number of rows and columns will be determined automatically in order to save as much display space as possible
- Enabled: This setting indicates whether or not the indicated channel should be displayed (channels that are not in use are hidden)
- Comment: A channel name, which will be used for display and control purposes, can be assigned to each channel
- Auto update on startup: If this “Flag” has been enabled, the indicated communications channel will be opened automatically when the Toolmonitor starts, and the status for the input channels will be scanned and displayed once

8.2.2.3. MechIO 16 Digital Input

The digital inputs to the mechanical control system are configured using this dialog window



RS232 ports >> RS232 (COM1) >> SerIO 1 >> MechIO DigitalInput

Title: MechIO DigitalInput

Number of rows: 0

Enabled	Channel	Comment	Enabled	Channel	Comment
☑	Channel 1	Channel 1	☑	Channel 9	Channel 9
☑	Channel 2	Channel 2	☑	Channel 10	Channel 10
☑	Channel 3	Channel 3	☑	Channel 11	Channel 11
☑	Channel 4	Channel 4	☑	Channel 12	Channel 12
☑	Channel 5	Channel 5	☑	Channel 13	Channel 13
☑	Channel 6	Channel 6	☑	Channel 14	Channel 14
☑	Channel 7	Channel 7	☑	Channel 15	Channel 15
☑	Channel 8	Channel 8	☑	Channel 16	Channel 16

☐ Auto update on startup

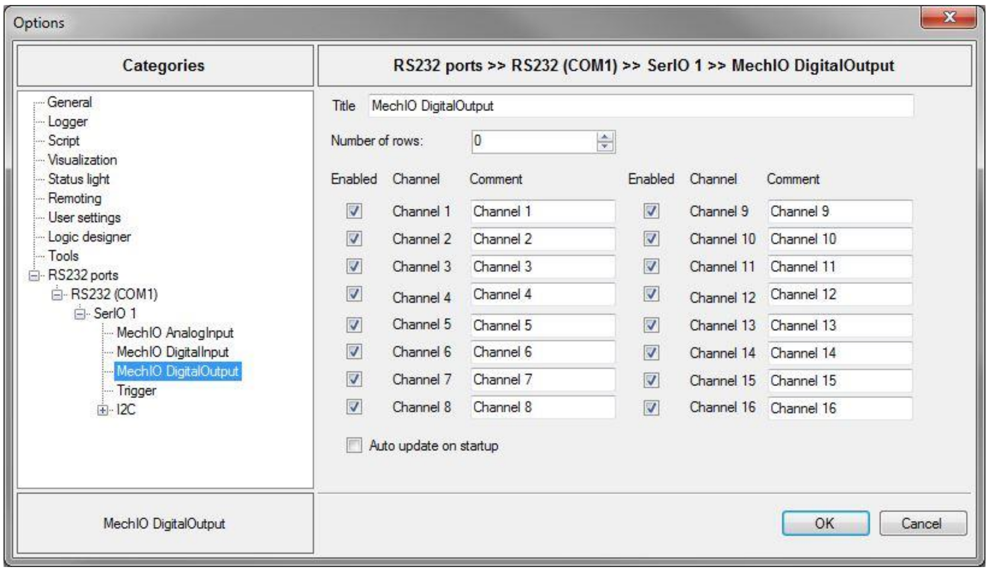
Figure 56: Configuring the Digital Inputs

The following areas are to be configured:

- Title: A name used for a digital input for display and control purposes
- Number of rows: The number of rows used to display the input channels. If the value is set to “0”, the number of rows and columns will be determined automatically in order to save as much display space as possible
- Enabled: This setting indicates whether or not the indicated channel should be displayed (channels that are not in use are hidden)
- Comment: A channel name, which will be used for display and control purposes, can be assigned to each channel
- Auto update on startup: If this “Flag” has been enabled, the indicated communications channel will be opened automatically when the Toolmonitor starts, and the status for the input channels will be scanned and displayed once

8.2.2.4. MechIO 16 Digital Output

The digital outputs from the mechanical control system are configured using this dialog window.



RS232 ports >> RS232 (COM1) >> SerIO 1 >> MechIO DigitalOutput

Title: MechIO DigitalOutput

Number of rows: 0

Enabled	Channel	Comment	Enabled	Channel	Comment
✓	Channel 1	Channel 1	✓	Channel 9	Channel 9
✓	Channel 2	Channel 2	✓	Channel 10	Channel 10
✓	Channel 3	Channel 3	✓	Channel 11	Channel 11
✓	Channel 4	Channel 4	✓	Channel 12	Channel 12
✓	Channel 5	Channel 5	✓	Channel 13	Channel 13
✓	Channel 6	Channel 6	✓	Channel 14	Channel 14
✓	Channel 7	Channel 7	✓	Channel 15	Channel 15
✓	Channel 8	Channel 8	✓	Channel 16	Channel 16

☐ Auto update on startup

Figure 57: Configuring the Digital Outputs

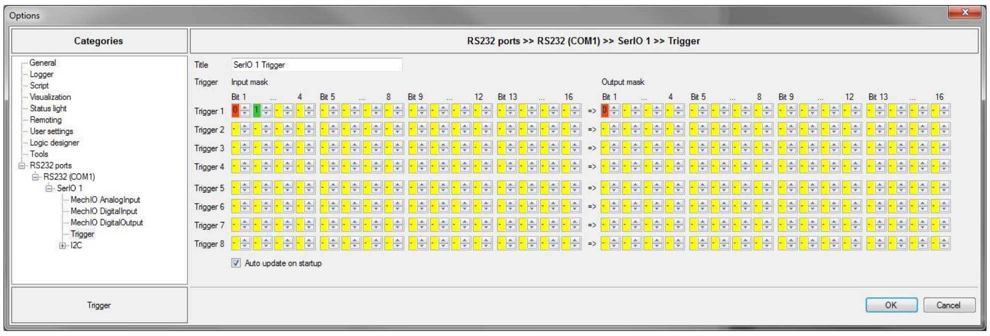
The following areas are to be configured:

- Title: A name used for a digital output for display and control purposes
- Number of rows: The number of rows used to display the output channels. If the value is set to “0”, the number of rows and columns will be determined automatically in order to save as much display space as possible
- Enabled: This setting indicates whether or not the indicated channel should be displayed (channels that are not in use are hidden)
- Comment: A channel name, which will be used for display and control purposes, can be assigned to each channel
- Auto update on startup: If this “Flag” has been enabled, the indicated communications channel will be opened automatically when the Toolmonitor starts, and the status for the output channels will be scanned and displayed once

8.2.2.5. MechIO 16 Trigger

The internal triggers for the mechanical control system are configured using this dialog window.

Up to a maximum of eight internal triggers can be defined. The mechanical control system will periodically check the inputs and set both the corresponding output states and the corresponding “Flag” according to the trigger condition. The level of precedence decreases as the “Index” number increases. This means that “Index 0” always has priority over the output state of, for example, “Index 1”. “Trigger Index 0” will be loaded from the EEPROM after “Power Up” and has the highest level of precedence.



RS232 ports >> RS232 (COM1) >> SerIO 1 >> Trigger

Title: SerIO 1 Trigger

Trigger	Input mask Bit 1 … 16		Output mask Bit 1 … 16
Trigger 1	0 1 - - - - - - - - - - - - - -	=>	0 - - - - - - - - - - - - - - -
Trigger 2	- - - - - - - - - - - - - - - -	=>	- - - - - - - - - - - - - - - -
Trigger 3	- - - - - - - - - - - - - - - -	=>	- - - - - - - - - - - - - - - -
Trigger 4	- - - - - - - - - - - - - - - -	=>	- - - - - - - - - - - - - - - -
Trigger 5	- - - - - - - - - - - - - - - -	=>	- - - - - - - - - - - - - - - -
Trigger 6	- - - - - - - - - - - - - - - -	=>	- - - - - - - - - - - - - - - -
Trigger 7	- - - - - - - - - - - - - - - -	=>	- - - - - - - - - - - - - - - -
Trigger 8	- - - - - - - - - - - - - - - -	=>	- - - - - - - - - - - - - - - -

☑ Auto update on startup

Figure 58: Configuring of the MechIO 16 Internal Trigger

The trigger interface consists of the following areas:

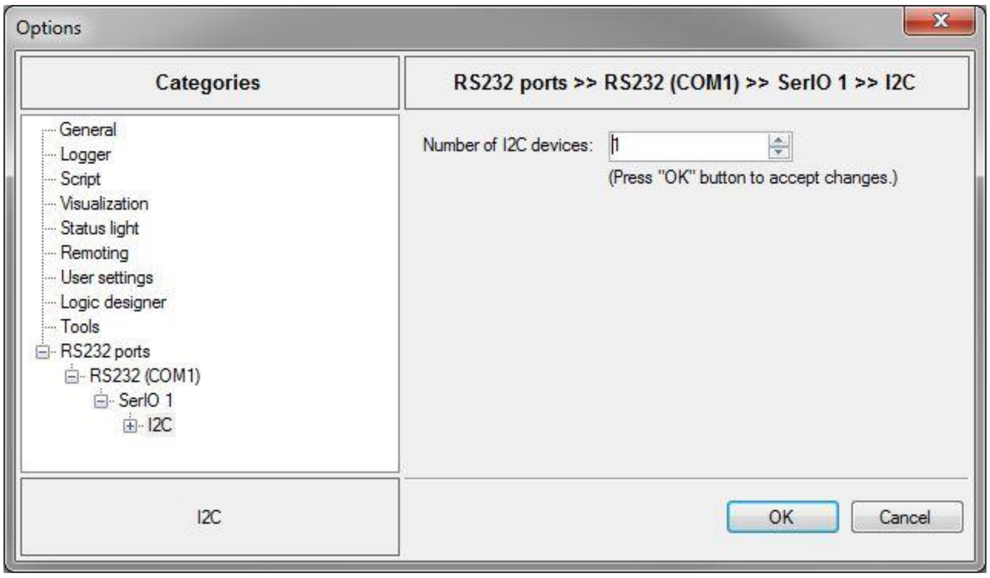
- Title: A name used for the internal trigger for display and control
- Input mask: The input mask to be used for the corresponding trigger is set from here. A minus sign "-" means that the corresponding bit is not relevant for the trigger, a "0" means that the corresponding bit must be "Low" and a "1" means that the corresponding bit must be "High"
- Output mask: The output mask to be used for the corresponding trigger is set from here. A minus sign "-" means that the corresponding bit is not relevant for the trigger, a "0" means that the corresponding bit must be set to "Low" and a "1" means that the corresponding bit must be set to "High"
- Auto update on startup: If this "Flag" has been enabled, the associated communications channel will be opened automatically when Toolmonitor starts and the trigger configuration will be sent to the mechanical control system. If this "Flag" has not been enabled, the trigger settings will not be transferred automatically to the mechanical control system. The trigger configuration can then only be triggered manually or set through the *virtual interface*

8.2.3. Adapter CPU

This SerIO device type represents the MCD mechanical control system "Adapter CPU"

8.2.3.1. I^{2}C

The number of I^{2}C ports to be controlled by the "Adapter CPU" will be determined by this setting.



RS232 ports >> RS232 (COM1) >> SerIO 1 >> I2C

Number of I2C devices: 1

(Press "OK" button to accept changes.)

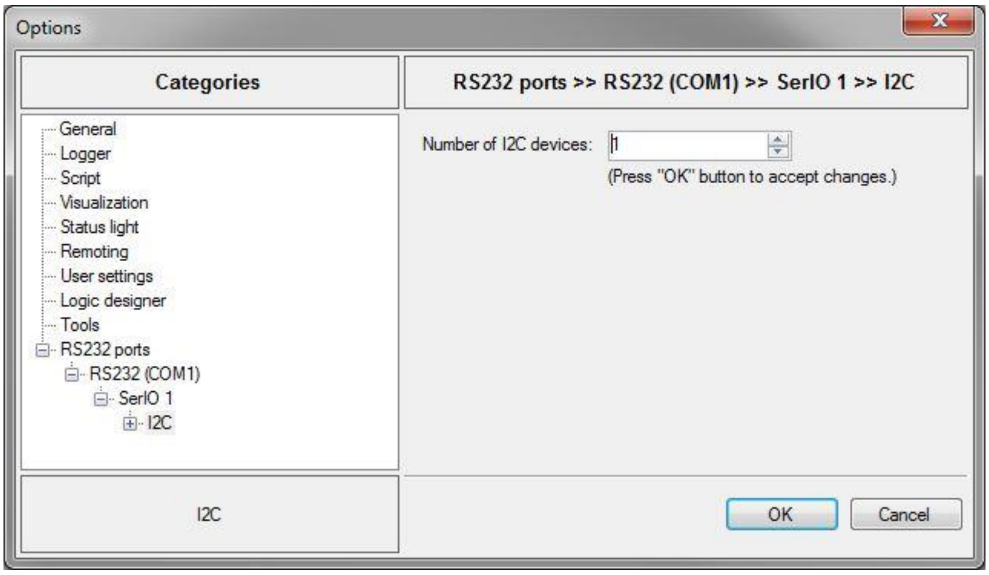
Figure 59: Setting the Number of I^{2}C Ports

8.2.4. Pass / Fail Display

This SerIO device type represents the MCD Pass / Fail display system.

8.2.4.1. I^{2}C

The number of I^{2}C ports to be controlled by the Pass / Fail display will be determined by this setting.



Options

Categories

- General
- Logger
- Script
- Visualization
- Status light
- Remoting
- User settings
- Logic designer
- Tools
- RS232 ports
  - RS232 (COM1)
    - SerIO 1
      - I2C

I2C

RS232 ports >> RS232 (COM1) >> SerIO 1 >> I2C

Number of I2C devices: 1

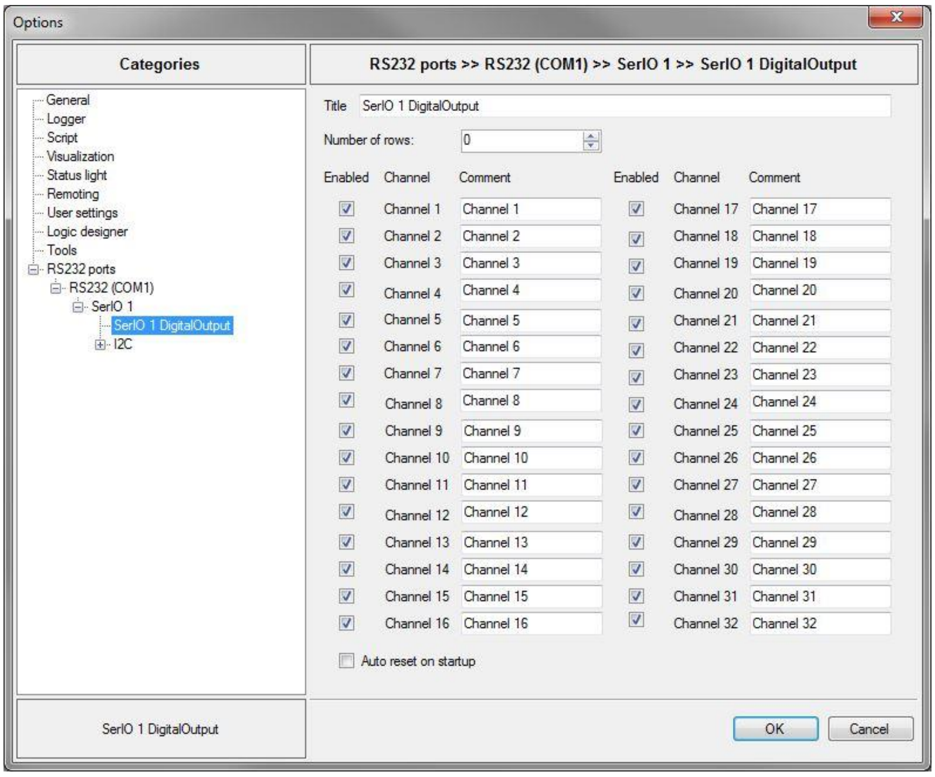
(Press "OK" button to accept changes.)

OK Cancel

Figure 60: Setting the Number of I^{2}C Ports

8.2.4.2. Pass / Fail Digital Output

The digital outputs from the “Pass / Fail display” system can be controlled using this dialog window.



Options

Categories

- General
- Logger
- Script
- Visualization
- Status light
- Remoting
- User settings
- Logic designer
- Tools
- RS232 ports
  - RS232 (COM1)
    - SerIO 1
      - SerIO 1 DigitalOutput
      - I2C

SerIO 1 DigitalOutput

RS232 ports >> RS232 (COM1) >> SerIO 1 >> SerIO 1 DigitalOutput

Title: SerIO 1 DigitalOutput

Number of rows: 0

Enabled	Channel	Comment	Enabled	Channel	Comment
✓	Channel 1	Channel 1	✓	Channel 17	Channel 17
✓	Channel 2	Channel 2	✓	Channel 18	Channel 18
✓	Channel 3	Channel 3	✓	Channel 19	Channel 19
✓	Channel 4	Channel 4	✓	Channel 20	Channel 20
✓	Channel 5	Channel 5	✓	Channel 21	Channel 21
✓	Channel 6	Channel 6	✓	Channel 22	Channel 22
✓	Channel 7	Channel 7	✓	Channel 23	Channel 23
✓	Channel 8	Channel 8	✓	Channel 24	Channel 24
✓	Channel 9	Channel 9	✓	Channel 25	Channel 25
✓	Channel 10	Channel 10	✓	Channel 26	Channel 26
✓	Channel 11	Channel 11	✓	Channel 27	Channel 27
✓	Channel 12	Channel 12	✓	Channel 28	Channel 28
✓	Channel 13	Channel 13	✓	Channel 29	Channel 29
✓	Channel 14	Channel 14	✓	Channel 30	Channel 30
✓	Channel 15	Channel 15	✓	Channel 31	Channel 31
✓	Channel 16	Channel 16	✓	Channel 32	Channel 32

☐ Auto reset on startup

OK Cancel

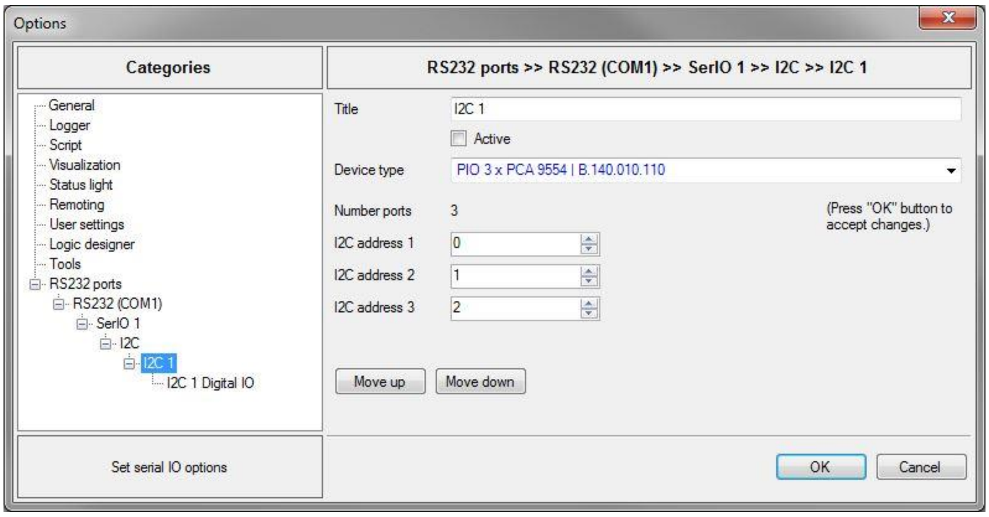
Figure 61: Configuring the Digital Outputs

The following areas are to be configured:

- Title: A name used for a digital output for display and control purposes
- Number of rows: The number of rows used to display the output channels. If the value is set to “0”, the number of rows and columns will be determined automatically in order to save as much display space as possible
- Enabled: This setting indicates whether or not the indicated channel should be displayed (channels that are not in use are hidden)
- Comment: A channel name, which will be used for display and control purposes, can be assigned to each channel
- Auto update on startup: If this “Flag” has been enabled, the associated communications channel will be opened automatically when the Toolmonitor starts and the status for the output channels will be reset once

8.3. I^{2}C

The type and the I^{2}C address(es) for each I^{2}C device to be used can be set using this dialog window.



RS232 ports >> RS232 (COM1) >> SerIO 1 >> I2C >> I2C 1

Title: I2C 1
Active
Device type: PIO 3 x PCA 9554 | B.140.010.110
Number ports: 3
(Press "OK" button to accept changes.)
I2C address 1: 0
I2C address 2: 1
I2C address 3: 2

Figure 62: Configuring an I^{2}C Device

The following types are available:

- Unknown (core functions for non - specific customer requests)
- PCF8574 | PIO 8 channels per port
- PCF8574A | PIO 8 channels per port
- PCA9554 | PIO 8 channels per port
- PCA9554A | PIO 8 channels per port
- PIO 3 x PCA 9554 | B.140.010.110
- PIO 3 x PCA 9554A | B.140.010.210
- PIO 3 x PCF 8574 | B.140.010.310
- PIO 3 x PCF 8574A | B.140.010.410
- RelMUX16 | Relay multiplexer 16 channels 7442
- PST30M | Power supply for tuner applications master B.170.030.112
- PST30S | Power supply for tuner applications slave B.170.030.212
- PST30L | Power supply for tuner applications linear slave B.170.030.412

The following areas are to be configured:

- Title: A name used for an I^{2}C device for display and control purposes
- Active: A “Flag” indicating that the deivce is active
- Device type: The I^{2}C device type (see above)
- I^{2}C address 1 .. n: The I^{2}C address(es) for the I^{2}C device (each I^{2}C device connected to a shared I^{2}C port must be assigned one or more unique I^{2}C address(es), number of I^{2}C addresses required will depend on the type of I^{2}C device)
- Move up / Move down: The selected I^{2}C device can be moved up or down in the list of I^{2}C devices by clicking these buttons

8.3.1. I^{2}C Digitale IO Devices (PIO)

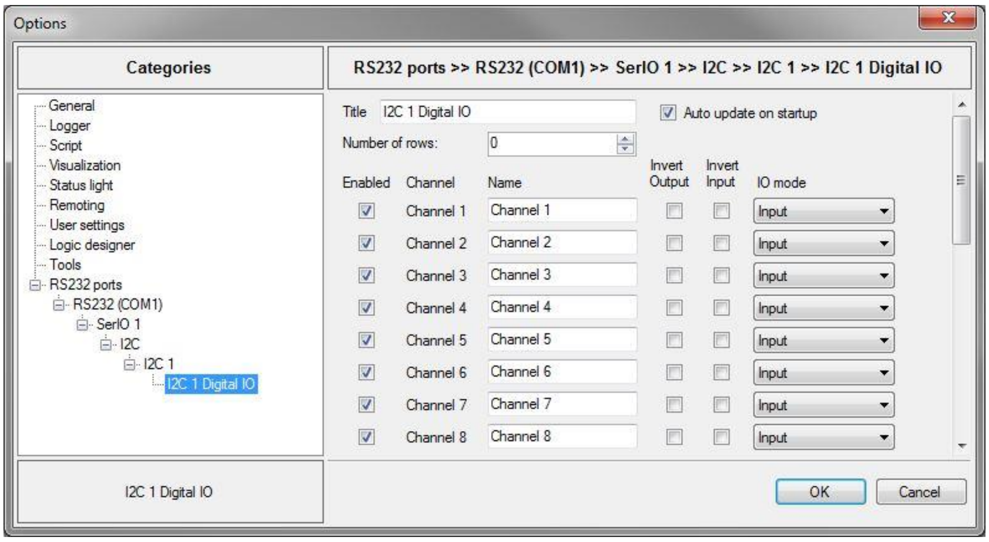
A number of I^{2}C devices serve to input or output digital signals. They are based on one or more I^{2}C PIO systems and are configured in the same manner.

The following types are available:

- PCF8574 | PIO 8 channels per port
- PCF8574A | PIO 8 channels per port
- PCA9554 | PIO 8 channels per port
- PCA9554A | PIO 8 channels per port
- PIO 3 x PCA 9554 | B.140.010.110
- PIO 3 x PCA 9554A | B.140.010.210
- PIO 3 x PCF 8574 | B.140.010.310
- PIO 3 x PCF 8574A | B.140.010.410
- RelMUX16 | Relay multiplexer 16 channels 7442

8.3.1.1. PIO Digital IO

The digital inputs and outputs to be controlled by the configured I^{2}C device can be configured using this dialog window.



RS232 ports >> RS232 (COM1) >> SerIO 1 >> I2C >> I2C 1 >> I2C 1 Digital IO

Title: I2C 1 Digital IO

☑ Auto update on startup

Number of rows: 0

Enabled	Channel	Name	Invert Output	Invert Input	IO mode
☑	Channel 1	Channel 1	☐	☐	Input
☑	Channel 2	Channel 2	☐	☐	Input
☑	Channel 3	Channel 3	☐	☐	Input
☑	Channel 4	Channel 4	☐	☐	Input
☑	Channel 5	Channel 5	☐	☐	Input
☑	Channel 6	Channel 6	☐	☐	Input
☑	Channel 7	Channel 7	☐	☐	Input
☑	Channel 8	Channel 8	☐	☐	Input

Figure 63: Configuring the Digital Inputs and Outputs

The following areas are to be configured:

- Title: A name used for the digital inputs and outputs for display and control purposes
- Auto update on startup: If this “Flag” has been enabled, the associated communications channel will be opened automatically when the Toolmonitor starts and the status for the input and output channels will be scanned and displayed once
- Number of rows: The number of rows used to display the input and output channels. If the value is set to “0”, the number of rows and columns will be determined automatically in order to save as much display space as possible
- Enabled: This setting indicates whether or not the indicated channel should be displayed (channels that are not in use are hidden)
- Name: A name can be assigned for each channel, which will be used for display and control purposes
- Invert Output: This setting specifies whether the input should be inverted or not for each channel. This option is not available for all I^{2}C devices
- Invert Input: This setting specifies whether the output should be inverted or not for each channel. This option is not available for all I^{2}C devices
- IO mode: The setting specifies if the channel should be set up as an input, an output or a bi - directional port (with corresponding “Default” values, option is not available for all I^{2}C devices)

8.3.2. I^{2}C Modules for Power Supplies

A number of I^{2}C devices serve to provide power through special power supply modules. The regulating component can adjust the frequency.

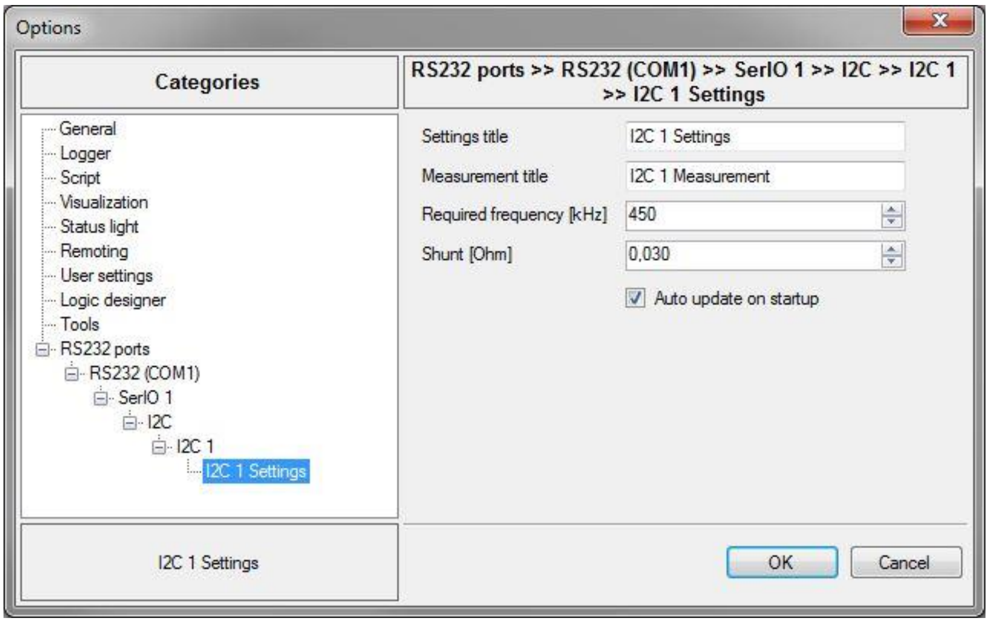
This applies to the following devices:

- PST30M | Power supply for tuner applications master B.170.030.112
- PST30S | Power supply for tuner applications slave B.170.030.212
- PST30L | Power supply for tuner applications linear slave B.170.030.412

8.3.2.1. Power Supply for Tuner Applications Master

This I^{2}C device represents a power supply with a controller module that can adjust the frequency.

These settings control the names used to display and control the “Settings” themselves, the “Measurements” and the frequency of the controller module, as well as the “Shunt” for measuring current.



RS232 ports >> RS232 (COM1) >> SerIO 1 >> I2C >> I2C 1 >> I2C 1 Settings

Settings title: I2C 1 Settings
Measurement title: I2C 1 Measurement
Required frequency [kHz]: 450
Shunt [Ohm]: 0,030
[✓] Auto update on startup

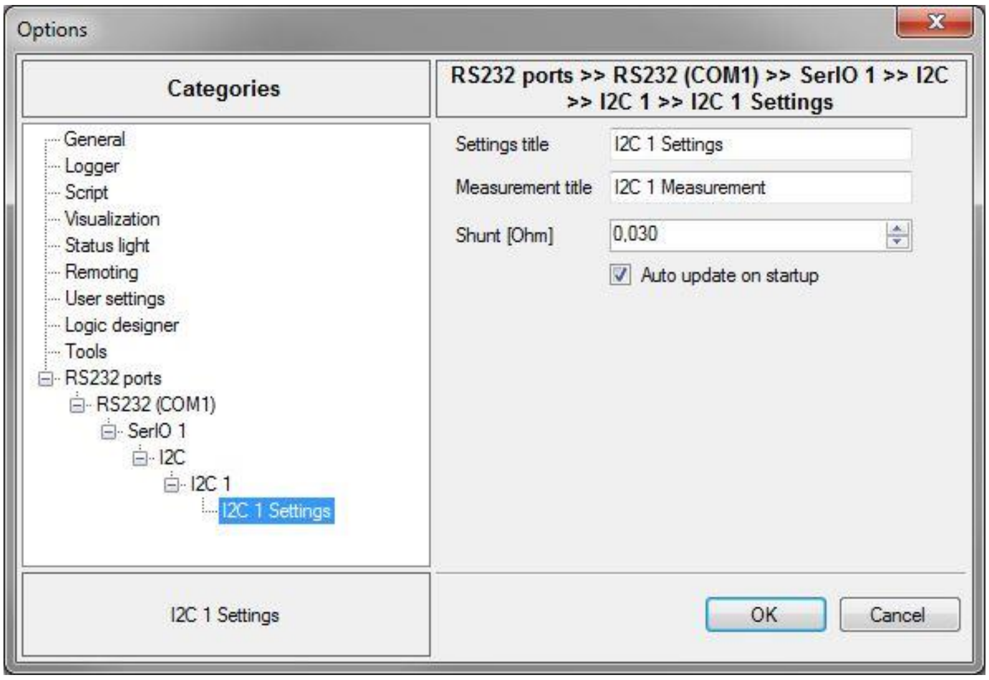
I2C 1 Settings

Figure 64: Configuring the Power Supply for Tuner Applications Master

8.3.2.2. Power Supply for Tuner Applications Slave

This I^{2}C device represents a “Slave” module for a power supply with a controller module that can adjust the frequency.

These settings control the names used to display and control the “Settings” themselves, the “Measurements” and the frequency of the controller module, as well as the “Shunt” for measuring current.



RS232 ports >> RS232 (COM1) >> SerIO 1 >> I2C >> I2C 1 >> I2C 1 Settings

Settings title: I2C 1 Settings
Measurement title: I2C 1 Measurement
Shunt [Ohm]: 0,030
[✓] Auto update on startup

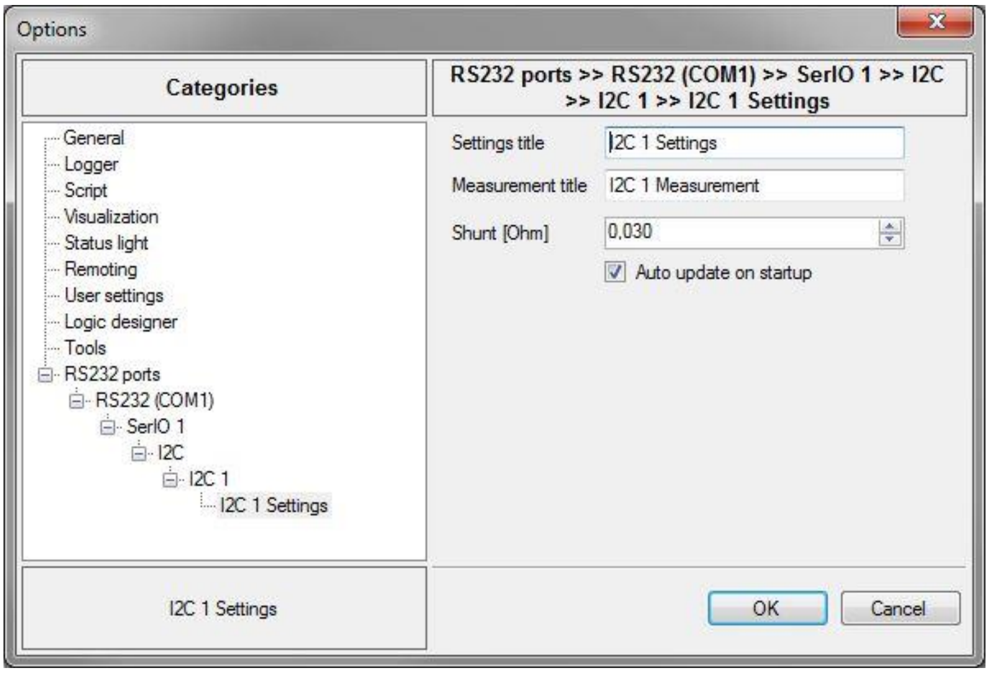
I2C 1 Settings

Figure 65: Configuring the Power Supply for Tuner Applications Slaves

8.3.2.3. Power Supply for Tuner Applications Linear Slave

This I^{2}C device represents a “Slave” module with linear controls for a power supply with a controller module that can adjust the frequency.

These settings control the names used to display and control the “Settings” themselves, the “Measurements” and the frequency of the controller module, as well as the “Shunt” for measuring current.



RS232 ports >> RS232 (COM1) >> SerIO 1 >> I2C >> I2C 1 >> I2C 1 Settings

Settings title: I2C 1 Settings

Measurement title: I2C 1 Measurement

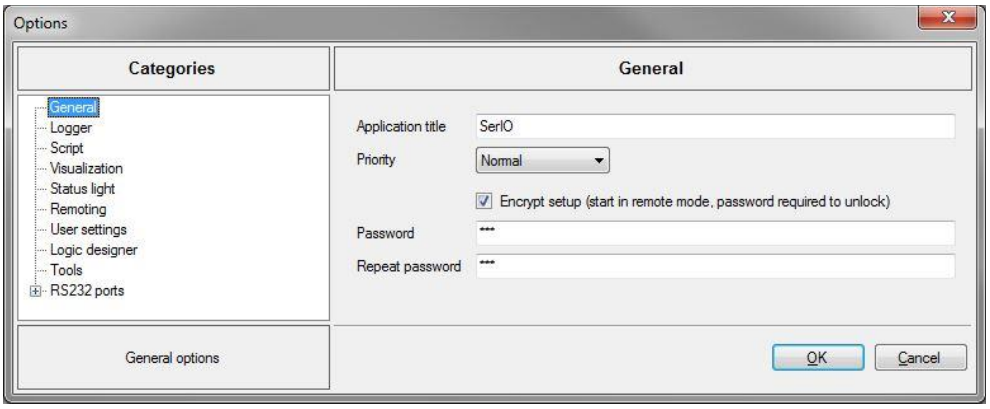
Shunt [Ohm]: 0,030

[✓] Auto update on startup

Figure 66: Configuring the Power Supply for Tuner Applications Linear Slaves

8.4. General

General Toolmonitor settings can be made from this dialog.



General

Application title: SerIO

Priority: Normal

[✓] Encrypt setup (start in remote mode, password required to unlock)

Password: ***

Repeat password: ***

Figure 67: General Settings

The following areas are to be configured:

- Applications title: Toolmonitor's window title can be adjusted.
- Priority: Toolmonitor's base priority can be set using this entry. This value should only be changed when there is a real need to do
- Encrypt setup: Whether the setup should be stored in an encrypted format or not can be determined using this setting. In that case, the Toolmonitor will be started in remote mode. In addition, a password must be set, which can be used to access the Toolmonitor. As a control measure, this password must be confirmed. If the password is lost, it will no longer be possible to change the associated setup (if the Toolmonitor is password - protected, the local and remote modes can no longer be controlled)

8.5. Logger

Toolmonitor's settings for the logging system can be made from this dialog.

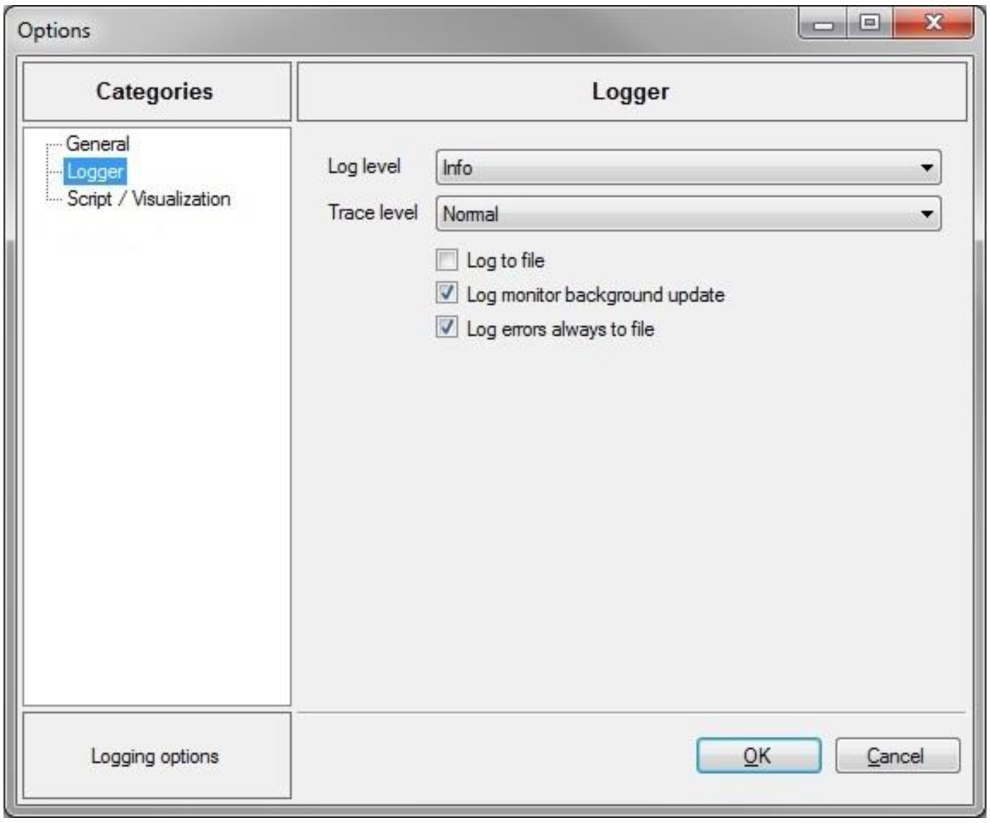

Log level: Info
Trace level: Normal
[ ] Log to file
[✓] Log monitor background update
[✓] Log errors always to file

Figure 68: Logger

8.5.1. Log Level

This setting determines if general event messages, warnings, errors and so forth will be logged.

The options include:

- ExtendedDebug
- Debug
- Trace
- Info
- Warning
- ErrorTrace
- CriticalWarning
- Error

8.5.2. Trace Level

This setting determines the trace level for logging sent or received messages.

The options include:

- Streaming
- Cyclic
- Normal

8.5.3. Log to File

If this option is activated, all event messages will be saved in a file. The name of the file will consist of Toolmonitor‘s name and a timestamp. The file will be stored in the same folder as the Toolmonitor executable file.

8.5.4. Log Monitor Background Update

Normally, event messages are always generated, even when displaying event messages has not been enabled in the monitor. If that is not desired, this feature can be deactivated using this option.

8.5.5. Log Errors Always to File

Normally, error messages will always be written to an additional log file. If that is not desired, this feature can be deactivated using this option. The name of the file will consist of Toolmonitor‘s name and the literal “Exceptions”. The file will be stored in the same folder as the Toolmonitor executable file.

9. Programming

9.1. AnalogInput

This is the base class for analog inputs of SerIO devices. This class provides the functions for controlling the analog input channels of the SerIO devices.

9.1.1. GetValueNames

Names for querying the values for the commands are defined here. Using these names, the *virtual interface's* "GetValue" function for the analog inputs will be triggered.

Example:

```
GetValue("Channel 1")
GetValue("Channel 1.HasAutoRange")
GetValue("IsValidMeasurementType.VoltDC")
```

List of names:

HasAutoRange	Queries if the named channel supports the "AutoRange" feature (always "0" in this case)
MaxRangeIndex	Queries the maximum index for the measurement range (always "0" in this case)
MaxAverages	Queries the maximum number of averages for the measurement range (always "1" in this case)
ChannelName	Queries the voltage of the analog input
IsValidMeasurementType	Queries if the transferred attribute is a valid measurement range

9.1.2. SetValueNames

Names for setting the values for the commands are defined here. Using these names, the *virtual interface's* "SetValue" function for the analog inputs will be triggered.

Example:

```
SetValue("Channel 1.Averages", 0)
```

List of names:

Averages	Number of averages (always "1" in this case)
MeasurementRange	Switches between measurement ranges (value must lie between "0" and "5")
RangeIndex	Index to the measurement range (always "0" in this case)
MeasurementType	Type of measurement (always "VoltDC" in this case)
Speed	Speed of measurement (permissible values are "Fast", "Slow" or "Normal", which are not relevant here)

9.2. DigitalInput

This is the base class for digital inputs of SerIO devices. This class provides the functions for controlling the digital input channels of the SerIO devices.

9.2.1. GetValuesNames

Names for querying the values for the commands are defined here. Using these names, the *virtual interface's* "GetValue" function for the digital inputs will be triggered.

Example:

```
            GetValue("Channel 1")
            GetValue("Ports")
            GetValue("Channel 1.HasAutoRange")
            GetValue("IsValidMeasurementType.VoltDC")
```

List of names:

HasAutoRange	Queries if the named channel supports the "AutoRange" feature (always "0" in this case)
MaxRangeIndex	Queries the maximum index for the measurement range (always "0" in this case)
MaxAverages	Queries the maximum number of averages for the measurement range (always "1" in this case)
ChannelName	Queries the state of the analog input
IsValidMeasurementType	Queries if the transferred attribute is a valid measurement range
TriggerLevel	Queries the set trigger level
Ports	Queries the state of the digital input channels (encoded bit - wise)

9.2.2. SetValueNames

Names for setting the values for the commands are defined here. Using these names, the *virtual interface's* "SetValue" function for the analog inputs will be triggered.

Example:

```
            SetValue("Channel 1.TriggerLevel", 3.5)
```

List of names:

Averages	Number of averages (always "1" in this case)
MeasurementRange	Switches between measurement ranges (value must lie between "0" and "5")
RangeIndex	Index to the measurement range (always "0" in this case)
MeasurementType	Type of measurement (always "VoltDC" in this case)
Speed	Speed of measurement (permissible values are "Fast", "Slow" or "Normal", which are not relevant here)
TriggerLevel	Sets the trigger level

9.3. DigitalOutput

This is the base class for digital outputs of SerIO devices. This class provides the functions for controlling the digital output channels of the SerIO devices.

9.3.1. GetValueNames

Names for querying the values for the commands are defined here. Using these names, the *virtual interface's* "GetValue" function for the digital outputs will be triggered.

Example:

```
GetValue("Channel 1")
GetValue("Ports")
```

List of names:

ChannelName	Queries the state of the digital output ("0" or "1")
Ports	Queries the state of the digital output channels (encoded bit - wise)

9.3.2. SetEventNames

Names for events are defined here. Using these names, the virtual interface's "SetEvent" function for the digital outputs will be triggered.

List of names:

Reset	Sets all outputs to "0" or "Low")

9.3.3. SetValueNames

Names for setting the values for the commands are defined here. Using these names, the *virtual interface's* "SetValue" function for the digital outputs will be triggered.

Example:

```
SetValue("Channel 1", 0)
```

List of names:

ChannelName	Sets the state of the digital output ("0" or "1")
Ports	Sets the state of the digital output channels (encoded bit - wise)

9.4. I^{2}CBaseLine

This is the I^{2}C base communication class. This class implements a communications channel for all I^{2}C SerIO device communications.

9.5. I²CDigitalIO

This is the base class for digital inputs and outputs of SerIO devices.

9.5.1. GetValueNames

Names for querying the values for the commands are defined here. Using these names, the *virtual interface's* "GetValue" function for the digital inputs and outputs will be triggered.

Examples:

```
GetValue("Channel 1")
GetValue("Ports")
GetValue("Channel 1.HasAutoRange")
GetValue("IsValidMeasurementType.VoltDC")
```

List of names:

HasAutoRange	Queries if the named channel supports the "AutoRange" feature (always "0" in this case)
MaxRangeIndex	Queries the maximum index for the measurement range (always "0" in this case)
MaxAverages	Queries the maximum number of averages for the measurement range (always "1" in this case)
ChannelName	Queries the state of the digital input ("0" or "1")
IsValidMeasurementType	Queries if the transferred attribute is a valid measurement range
Port0	Queries the state of all port 0 digital input channels (8 channels, encoded bit - wise)
Port1	Queries the state of all port 1 digital input channels (8 channels, encoded bit - wise)
Port2	Queries the state of all port 2 digital input channels (8 channels, encoded bit - wise)
Port3	Queries the state of all port 3 digital input channels (8 channels, encoded bit - wise)
Ports	Queries the state of the digital input channels (encoded bit - wise)

9.5.2. SetEventNames

Names for events are defined here. Using these names, the *virtual interface's* "SetEvent" function for the digital inputs and outputs will be triggered.

List of names:

Reset	Sets all digital inputs and outputs to the corresponding value to the default settings. Outputs will be set to "0" or "Low"

9.5.3. SetValueNames

Names for setting the values for the commands are defined here. Using these names, the *virtual interface‘s* “SetValue” function for the digital inputs and outputs will be triggered.

Example:

```
            SetValue("Channel 1", 1)
            SetValue("Channel 1.Mode", 1)
            SetValue("Ports0Mode", 0)
```

List of names:

Averages	Number of averages (always “1” in this case)
MeasurementRange	Switches between measurement ranges (value must lie between “0” and “5”)
RangeIndex	Index to the measurement range (always “0” in this case)
MeasurementType	Type of measurement (always “VoltDC” in this case)
Speed	Speed of measurement (permissible values are “Fast”, “Slow” or “Normal”, which are not relevant here)
Mode	Sets the channel to input or output mode. This parameter is only allowed if the channel supports this and switching the channel is permitted in the default settings. (mode = 0: input, mode = 1: output)
PortsMode	Sets all channels to input or output mode. This parameter is only allowed if the module supports this and switching is permitted in the default settings. (mode = 0: input, mode = 1: output)
Port0Mode	Sets all port 0 channels to input or output mode. This parameter is only allowed if the module supports this and switching is permitted in the default settings. (mode = 0: input, mode = 1: output)
Port1Mode	Sets all port 1 channels to input or output mode. This parameter is only allowed if the module supports this and switching is permitted in the default settings. (mode = 0: input, mode = 1: output)
Port2Mode	Sets all port 2 channels to input or output mode. This parameter is only allowed if the module supports this and switching is permitted in the default settings. (mode = 0: input, mode = 1: output)
Port3Mode	Sets all port 3 channels to input or output mode. This parameter is only allowed if the module supports this and switching is permitted in the default settings. (mode = 0: input, mode = 1: output)
Port0	Sets the state of all port 0 digital output channels (8 channels, encoded bit - wise)
Port1	Sets the state of all port 1 digital output channels (8 channels, encoded bit - wise)
Port2	Sets the state of all port 2 digital output channels (8 channels, encoded bit - wise)
Port3	Sets the state of all port 3 digital output channels (8 channels, encoded bit - wise)
Ports	Sets the state of the digital output channels (encoded bit - wise)

9.6. I^{2}CLine

This is the I^{2}C base communication class. This class implements a communications channel for all I^{2}C communications with an I^{2}C module.

9.7. Line

This is the base class for serial communications using a variety of communications paths. This includes I^{2}CBaseLine, I^{2}CLine, SerIOLine and RS232.

9.7.1. VirtualInterfaceCommands

Names for the commands for the *virtual interface* of the communications channel are defined here. Issuing these commands “SetEvent”, “SetValue”, “GetValue” or “GetString”, the *virtual interface‘s* “SetValue” will trigger events in the communications channel‘s *virtual interface*.

List of names:

Open	Opens the communications channel
Close	Closes the communications channel
Clear	Deletes the data in the communications channel‘s send and receive buffers
IsActive	Indicates if the communications channel in the basic settings is active
IsOpen	Indicates if the communications channel has been opened
DataAvailable	Indicates if data is ready for receipt (not supported by all communications channels)
BytesAvailable	Returns the number of bytes ready for receipt (not supported by all communications channels)
Write	Sends the indicated data through the communications channel
WriteMessage	Sends the message stored under the indicated name through the communications channel
WriteSequence	Sends the string stored under the indicated name through the communications channel
Read	Reads the data from the communications channel
Execute	Sends the indicated command through the communications channel for execution and saves the associated response under “Response” (command is only intended for communications channels without using separated read / write functions)
ExecuteMessage	Sends the command stored under the indicated name through the communications channel for execution and saves the associated response under “Response” (command is only intended for communications channels without using separated read / write functions)
ExecuteSequence	Sends the string stored under the indicated name through the communications channel for execution and saves the associated response under “Response” (command is only intended for communications channels without using separated read / write functions)
Response	Returns the last message received about an “Execute” command

TraceLevel	Changes the trace level for sending and receiving through the *virtual interface* (allowed values: Streaming = “0”, Cycling = “1”, Normal = “2”, None = “3”)
Timeout	Changes the timeout for sending and receiving through the *virtual interface*
Count	Changes the byte count for sending and receiving through the *virtual interface*
Comment	Changes the comment for sending and receiving through the *virtual interface*
EndOfLine	Changes the EndOfLine string for sending and receiving through the *virtual interface*
ThrowTimeoutException	Activates or deactivates the timeout exception when sending or receiving through the *virtual interface*
GetMessage	Returns the message stored under the indicated name

9.8. RS232

This is the RS232 base communication class. This class implements a communications channel through an RS232 interface. It is inherited from the “Line” base class and is itself an inheritance for the “SerIOBaseLine” base class.

9.9. SerIOBaseLine

This is the base class for SerIO communications via RS232. This class implements a communications channel through an RS232 interface.

9.10. SerIOLine

This is the SerIO base communication class. This is the SerIO base communication class. This class implements a communications channel for communications to a SerIO device.

9.11. Trigger

This is the base class for controlling the MechIO 16 mechanical control trigger. This class provides the functions for controlling the MechIO 16 mechanical control trigger.

9.11.1. SetEventNames

Names for events are defined here. Using these names, the *virtual interface‘s* “SetEvent” function for the triggers will be triggered.

Example:

```
            SetEvent("Reset");
            SetEvent("WriteAll");
            SetEvent("Trigger1.Write");
```

List of names:

Reset	Reset all triggers
WriteAll	Sends the trigger configuration for all triggers to the mechanical control system
Write	Write the configuration for the associated trigger to the mechanical control system (respective trigger must be pre - defined for this, see example)
TriggerN	Names for the pre - defined triggers for the "Write" parameters (N = 1..16)

9.11.2. SetValueNames

Names for setting the values for the commands are defined here. Using these names, the *virtual interface's* "SetValue" function for the triggers will be triggered.

Example:

```
SetValue("Trigger 1.InputMask", 3)
SetValue("Trigger 1.InputValue", 2)
SetValue("Trigger 1.OutputMask", 255)
SetValue("Trigger 1.OutputValue", 0)
SetValue("Trigger 1.Input.Bit1", -1)
SetValue("Trigger 1.Input.Bit1", 0)
SetValue("Trigger 1.Input.Bit1", 1)
SetValue("Trigger 1.Output.Bit1", -1)
SetValue("Trigger 1.Output.Bit1", 0)
SetValue("Trigger 1.Output.Bit1", 1)
```

List of names:

InputMask	Sets the input mask for the corresponding trigger what involves bit - wise encoding the inputs that are relevant for the trigger (respective trigger must be pre - defined for this, see example)
InputValue	Sets the input condition for the corresponding trigger what involves bit - wise encoding the input values that are relevant for the trigger (respective trigger must be pre - defined for this, see example)
OutputMask	Sets the output mask for the corresponding trigger what involves bit - wise encoding the outputs that are relevant for the trigger (respective trigger must be pre - defined for this, see example)
OutputValue	Sets the output condition for the corresponding trigger what involves bit - wise encoding the output values that are relevant for the trigger (respective trigger must be pre - defined for this, see example)
Input	Direct configuration of the input bits (see example)
Output	Direct configuration of the output bits (see example)
BitN	The individual bits for the input and output configurations can be set specifically in order to configure the triggers (N = 1..16, the following values are allowed: "-1" = bit is not relevant, "0" = bit is "Low" or must be "Low", "1" = bit is "High" or must be "High")