

AutomationDesk

Accessing RS232

For AutomationDesk 6.5

Release 2021-A – May 2021

How to Contact dSPACE

Mail:	dSPACE GmbH Rathenaustraße 26 33102 Paderborn Germany
Tel.:	+49 5251 1638-0
Fax:	+49 5251 16198-0
E-mail:	info@dspace.de
Web:	http://www.dspace.com

How to Contact dSPACE Support

If you encounter a problem when using dSPACE products, contact your local dSPACE representative:

- Local dSPACE companies and distributors: <http://www.dspace.com/go/locations>
- For countries not listed, contact dSPACE GmbH in Paderborn, Germany.
Tel.: +49 5251 1638-941 or e-mail: support@dspace.de

You can also use the support request form: <http://www.dspace.com/go/supportrequest>. If you are logged on to mydSPACE, you are automatically identified and do not need to add your contact details manually.

If possible, always provide the relevant dSPACE License ID or the serial number of the CmContainer in your support request.

Software Updates and Patches

dSPACE strongly recommends that you download and install the most recent patches for your current dSPACE installation. Visit <http://www.dspace.com/go/patches> for software updates and patches.

Important Notice

This publication contains proprietary information that is protected by copyright. All rights are reserved. The publication may be printed for personal or internal use provided all the proprietary markings are retained on all printed copies. In all other cases, the publication must not be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form, in whole or in part, without the prior written consent of dSPACE GmbH.

© 2017 - 2021 by:
dSPACE GmbH
Rathenaustraße 26
33102 Paderborn
Germany

This publication and the contents hereof are subject to change without notice.

AUTERA, ConfigurationDesk, ControlDesk, MicroAutoBox, MicroLabBox, SCALEXIO, SIMPHERA, SYNECT, SystemDesk, TargetLink and VEOS are registered trademarks of dSPACE GmbH in the United States or other countries, or both. Other brand names or product names are trademarks or registered trademarks of their respective companies or organizations.

Contents

About This Document	5
Basics and Instructions	7
Overview of the RS232 Library Elements.....	7
Example of an RS232 Sequence.....	9
How to Configure the Serial Interface.....	11
How to Send Data via the Serial Interface.....	12
How to Receive Data via the Serial Interface.....	14
Reference Information	17
Automation Blocks.....	18
GetNumInBytes.....	18
Read.....	19
RS232Configuration.....	20
SetReadTimeout.....	22
Write.....	23
WriteString.....	23
Commands And Dialogs.....	25
Edit (RS232Configuration).....	25
Insert (RS232 Elements).....	26
Automation	29
Basics on Automating the Access to RS232.....	29
Limitations	31
Limitations When Using the RS232 Library.....	31
Index	33

About This Document





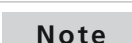


Content This document gives you information on how to access an external device via serial interface using AutomationDesk.


Required knowledge Working with AutomationDesk requires:

- Basic knowledge in handling the PC and the Microsoft Windows operating system.
- Basic knowledge in developing applications or tests.
- Basic knowledge in handling the external device, which you control remotely via AutomationDesk.

dSPACE provides trainings for AutomationDesk. For more information, refer to <https://www.dspace.com/go/trainings>.

Symbols dSPACE user documentation uses the following symbols:

Symbol	Description
	Indicates a hazardous situation that, if not avoided, will result in death or serious injury.
	Indicates a hazardous situation that, if not avoided, could result in death or serious injury.
	Indicates a hazardous situation that, if not avoided, could result in minor or moderate injury.
	Indicates a hazard that, if not avoided, could result in property damage.
	Indicates important information that you should take into account to avoid malfunctions.
	Indicates tips that can make your work easier.
	Indicates a link that refers to a definition in the glossary, which you can find at the end of the document unless stated otherwise.

Symbol	Description
	Precedes the document title in a link that refers to another document.

Naming conventions

dSPACE user documentation uses the following naming conventions:

%name% Names enclosed in percent signs refer to environment variables for file and path names.

< > Angle brackets contain wildcard characters or placeholders for variable file and path names, etc.

Special folders

Some software products use the following special folders:

Common Program Data folder A standard folder for application-specific configuration data that is used by all users.

%PROGRAMDATA%\dSPACE\<InstallationGUID>\<ProductName>

or

%PROGRAMDATA%\dSPACE\<ProductName>\<VersionNumber>

Documents folder A standard folder for user-specific documents.

%USERPROFILE%\Documents\dSPACE\<ProductName>\<VersionNumber>

Local Program Data folder A standard folder for application-specific configuration data that is used by the current, non-roaming user.

%USERPROFILE%\AppData\Local\dSPACE\<InstallationGUID>\<ProductName>

Accessing dSPACE Help and PDF Files


After you install and decrypt dSPACE software, the documentation for the installed products is available in dSPACE Help and as PDF files.

dSPACE Help (local) You can open your local installation of dSPACE Help:

- On its home page via Windows Start Menu
- On specific content using context-sensitive help via **F1**

dSPACE Help (Web) You can access the Web version of dSPACE Help at www.dspace.com/go/help.

To access the Web version, you must have a *mydSPACE* account.

PDF files You can access PDF files via the  icon in dSPACE Help. The PDF opens on the first page.

Basics and Instructions

Where to go from here

Information in this section

Overview of the RS232 Library Elements.....	7
Provides basic information on AutomationDesk's library elements that you can use for accessing the serial interface.	
Example of an RS232 Sequence.....	9
Provides an example of automating access to the serial interface.	
How to Configure the Serial Interface.....	11
Instruction how to configure its parameters to use the serial interface of your host PC.	
How to Send Data via the Serial Interface.....	12
Instruction how to use a Write or WriteString automation block to send data to the communication partner.	
How to Receive Data via the Serial Interface.....	14
Instruction how to use the Read automation block to receive data from the communication partner.	

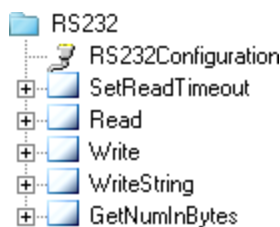
Overview of the RS232 Library Elements

Library overview

AutomationDesk provides library elements for realizing communication via the serial interface of the host PC. After configuring the COM port, you can send and receive data.

Note

- The serial interface of your host PC must be connected to the serial interface of the remote device via a null-modem cable. This could be another serial interface of your host PC. On the remote device, you have to start a terminal program for control of its serial interface.
- The RS232 library is included in AutomationDesk 6.5 for the last time. For later versions of AutomationDesk, a custom library that provides the same functionality will be available on demand. Refer to <https://www.dspace.com/go/AUD-RS232-CustLib>.



RS232Configuration

The RS232Configuration library element is a data object for configuring the serial interface. You can specify the port, baud rate, data bits, stop bits, parity and buffer sizes for the serial port's input and output buffers. For further information, refer to [RS232Configuration](#) on page 20.

SetReadTimeout

The SetReadTimeout automation block specifies the time for waiting for new data in the serial port's input buffer. For further information, refer to [SetReadTimeout](#) on page 22.

Read

The Read automation block reads the specified number of bytes from the serial port's input buffer. For further information, refer to [Read](#) on page 19.

Write

The Write automation block writes the specified ASCII character, which is represented by its integer value, to the serial port's output buffer. For further information, refer to [Write](#) on page 23.

WriteString

The WriteString automation block writes the specified string to the serial port's output buffer. For further information, refer to [WriteString](#) on page 23.

GetNumInBytes

The GetNumInBytes automation block provides the number of bytes in the serial port's input buffer. For further information, refer to [GetNumInBytes](#) on page 18.

Related topics

References

[Reference Information..... 17](#)

Example of an RS232 Sequence

Introduction

The following automation sequence shows you a very simple program for realizing access to the serial interface. It automates only:

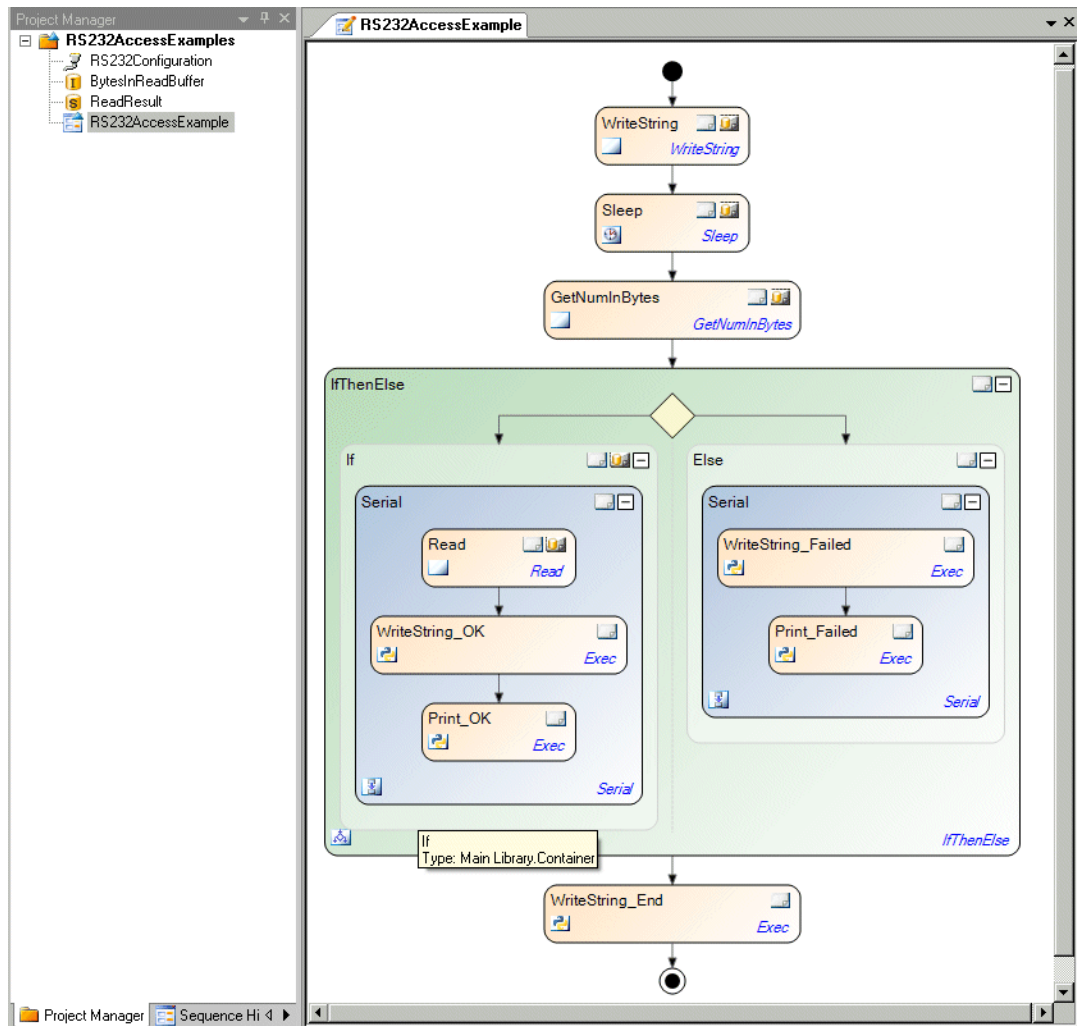
- Writing a string to the serial port's output buffer
- Getting the number of bytes in the serial port's input buffer
- Reading the bytes from the serial port's input buffer

The sequence does not contain a Write and a SetReadTimeout automation block.

Preconditions

You have to create some project-specific data objects in the Project Manager:

- RS232Configuration data object to specify the serial interface on the host PC
- Integer data object to store the return value of the GetNumInBytes automation block
- String data object to store the return value of the Read automation block



Result


When you execute the sequence, you can write a predefined number of characters to the serial port's output buffer. Before getting the number of bytes in the input buffer, the execution waits for a while, for example, 5 seconds. The condition of the IF construction is the comparison of the predefined number of characters and the number of characters found in the input buffer. If they are equal, the string is read and a message for successful transmission is written to the serial port and to the standard output. Otherwise, a message for a failed transmission is written to the serial port and to the standard output.

Related topics

Basics

[Overview of the RS232 Library Elements..... 7](#)

How to Configure the Serial Interface

Objective	The default COM port for serial communication might already be used for another device. To specify a free COM port for your automation task, you can use the Configuration block from the RS232 library.
Configuration parameters	<p>Communication via the serial interface is specified by:</p> <ul style="list-style-type: none"> ▪ Port ▪ Baud rate ▪ Number of data bits ▪ Parity scheme ▪ Number of stop bits ▪ Sizes of the input and output buffers
Configuration referencing	<p>The RS232 automation blocks need an RS232Configuration data object as an input parameter:</p> <ul style="list-style-type: none"> ▪ If you have not specified an RS232 configuration in the Project Manager, the default configuration will be used. ▪ If you have specified one RS232 configuration in the Project Manager, this data object is referenced automatically from the RS232 automation blocks. ▪ If you have specified more than one RS232 configuration in the Project Manager, the RS232 automation blocks use the first configuration data object of their hierarchy level or the explicitly referenced one. This behavior is the same as for any other project-specific data object. For further information, refer to Scope of Data Object References (AutomationDesk Basic Practices ).
Restrictions	<ul style="list-style-type: none"> ▪ Each serial port can be set only once in your project. ▪ Do not specify the following combinations of data bit and stop bit numbers: <ul style="list-style-type: none"> ▪ Number of data bits = 5 and number of stop bits = 2 ▪ Number of data bits = 6, 7, or 8 and number of stop bits = 1.5 These combinations lead to invalid transmissions. ▪ The input and output buffer sizes must be even values.
Preconditions	<ul style="list-style-type: none"> ▪ To access a serial port with the RS232 library, the serial device driver of the operating system (serial.sys) must have been started. ▪ Make sure that the serial ports you want to use are not used by other applications at the same time.
Method	<p>To configure the serial interface</p> <ol style="list-style-type: none"> 1 Drag an RS232Configuration block from the Library Browser to the Project Manager.

- 2 Double-click the block to open the RS232 Configuration dialog.
- 3 Specify the port, bits per second, data bits, parity, stop bits and sizes of the input and output buffer.
- 4 Close the dialog.

Result You have created a project-specific configuration data object of the serial interface. This configuration will be used by the other RS232 automation blocks of your communication task.

Next step When you have configured the serial interface, you should proceed with [How to Send Data via the Serial Interface](#) on page 12.

Related topics

References

Edit (RS232Configuration).....	25
RS232Configuration.....	20

How to Send Data via the Serial Interface

Objective If you want to use the serial interface for remote control of an external application, you have to send data to it.

Write data buffering Sending data via the serial interface means that the data to be sent is first written to the specified serial port's output buffer on your host PC. Then the serial device driver of your operating system starts the transmission to the connected communication partner.

Preconditions Before you can send data via the serial interface, you have to configure it by using the RS232Configuration data object. Refer to [How to Configure the Serial Interface](#) on page 11.

Possible methods AutomationDesk provides two different blocks for sending data:

- If you want to send a single ASCII value, you can use the Write automation block. Refer to [Method 1](#) on page 13.
- If you want to send a string, you can use the WriteString automation block. Refer to [Method 2](#) on page 13.

Method 1**To send a single ASCII value via the serial interface**

- 1** Drag a **Write** automation block from the Library Browser to the sequence.
- 2** In the Sequence Builder, choose **View Data Objects** from the automation block's context menu.
- 3** Type the integer representation of the ASCII value to be transmitted in the **Value** column of the **ByteToWrite** data object.

Method 2**To send a string via the serial interface**

- 1** Drag a **WriteString** automation block from the Library Browser to the sequence.
- 2** In the Sequence Builder, choose **View Data Objects** from the automation block's context menu.
- 3** Type the string to be transmitted in the **Value** column of the **StringToWrite** data object.

Result

When you execute your communication task, the specified data is written to the serial port's output buffer and then transmitted to the connected communication partner. The transmission of a string can take some time. Thus, it is important that the communication partner waits for the complete transmission if you use the **WriteString** automation block.

Example

If you want to send an 'A' to the communication partner, you must:


- Specify the integer value 65 using the **Write** automation block
- Specify the string 'A' using the **WriteString** automation block

Next step

When you have sent data to the communication partner, you should proceed with [How to Receive Data via the Serial Interface](#) on page 14.

Related topics**HowTos**

[How to Configure the Serial Interface..... 11](#)

How to Receive Data via the Serial Interface.....	14
Examples	
Example of an RS232 Sequence.....	9
References	
View Data Objects (AutomationDesk Basic Practices )	
Write.....	23
WriteString.....	23

How to Receive Data via the Serial Interface

Objective	Communication via the serial interface is mostly in bidirectional mode. So it is also necessary to know how to receive data from the communication partner.
Read data buffering	Receiving data via the serial interface means reading the contents of the specified serial port's input buffer. Because the transmission depends on the communication partner's send parameters, you have to check the input buffer's content before reading.
Preconditions	<ul style="list-style-type: none"> ▪ Before you can send data via the serial interface, you have to configure it using the RS232Configuration data object. Refer to How to Configure the Serial Interface on page 11. ▪ To limit the time the Read block waits for data to arrive in the serial port's input buffer, you should specify a read timeout using the SetReadTimeout automation block. ▪ To avoid problems by reading data from the serial port's input buffer, you should get the actual number of bytes contained in the buffer before executing the Read block. You can use the GetNumInBytes automation block for this purpose.
Method	<p>To receive data from the serial interface</p> <ol style="list-style-type: none"> 1 Drag a Read automation block from the Library Browser to the sequence. 2 In the Sequence Builder, choose View Data Objects from the automation block's context menu. 3 Specify the number of bytes to be read from the serial port's input buffer. If you use the GetNumInBytes automation block, you can set a reference from the BytesInReadBuffer data object to the BytesToRead data object so that the number of bytes to be read is always set to the correct value.

Result

When you execute your communication task, the Read block reads the specified number of bytes from the serial port's input buffer and stores it as a string in the ReadResult data object. If there is no new byte sent within the specified timeout, the Read block aborts with an exception.

Related topics


HowTos

How to Configure the Serial Interface.....	11
How to Send Data via the Serial Interface.....	12

Examples

Example of an RS232 Sequence.....	9
-----------------------------------	---

References

GetNumInBytes.....	18
Read.....	19
SetReadTimeout.....	22
View Data Objects (AutomationDesk Basic Practices )	

Reference Information

Where to go from here

Information in this section

Automation Blocks.....	18
Commands And Dialogs.....	25

Automation Blocks

Introduction

The RS232 library provides automation blocks for realizing communication via the serial interface of the host PC.

Note

The RS232 library is included in AutomationDesk 6.5 for the last time. For later versions of AutomationDesk, a custom library that provides the same functionality will be available on demand. Refer to <https://www.dspace.com/go/AUD-RS232-CustLib>.

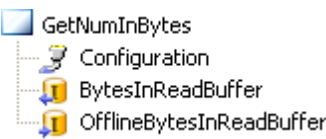
Where to go from here

Information in this section

GetNumInBytes.....	18
To get the number of bytes in the serial port's input buffer.	
Read.....	19
To read a number of values from the serial port's input buffer.	
RS232Configuration.....	20
To specify the configuration of the serial interface.	
SetReadTimeout.....	22
To specify the read timeout.	
Write.....	23
To write a byte to the serial port.	
WriteString.....	23
To write a string to the serial port.	

GetNumInBytes

Graphical representation



Purpose

To get the number of bytes in the serial port's input buffer.

Description

When the Read block is executed, a runtime error can occur if the number of bytes to be read is greater than the number of bytes contained in the buffer. You can avoid these problems by executing the GetNumInBytes block before executing the Read block.

Data objects

This automation block provides the following data objects:

Name	In / Out	Data Type	Default Value	Description
Configuration	In	RS232Configuration	-	Contains the configuration of the RS232 interface.
BytesInReadBuffer	Out	Int	0	Contains the number of bytes in the input buffer.
OfflineBytesInReadBuffer	In	Int	0	Lets you specify the value to be used in offline operation mode.

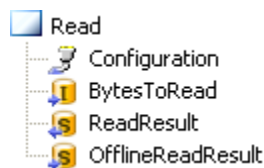
Related topics**Basics**

[Executing Sequences Using Different Operation Modes \(AutomationDesk Basic Practices !\[\]\(3e2231b1ad3ca8da8658228c00dd08e0_img.jpg\)\)](#)
[Overview of the RS232 Library Elements..... 7](#)

References

[Read..... 19](#)

Read

Graphical representation**Purpose**

To read a number of values from the serial port's input buffer.

Description

The Read block provides byte-oriented reading of data from the input buffer. Thus, the number of bytes to be read must be specified. By default, one byte is read.

Note

When the Read block is executed, one of the following cases occurs:

- There are at least as many bytes in the input buffer as specified for reading: The block returns the read bytes as a string.
- One byte is to be read and there is no data in the input buffer: The block is finished as soon as a byte is transferred into the input buffer within the specified timeout. If the timeout is reached without a new byte being transferred into the input buffer, the Read block is aborted with an exception.
- More than one byte is to be read and no values are stored in the input buffer when executing the Read block: The block is finished with an exception as soon as a byte is transferred into the input buffer or the timeout is reached.

Data objects

This automation block provides the following data objects:

Name	In / Out	Data Type	Default Value	Description
Configuration	In	RS232Configuration	None	Contains the configuration of the RS232 interface.
BytesToRead	In	Int	1	Contains the number of bytes to be read.
ReadResult	Out	String	" "	Contains a string read from the PC port.
OfflineReadResult	In	String	" "	Lets you specify the string to be used in offline operation mode.

Related topics**Basics**

[Executing Sequences Using Different Operation Modes \(AutomationDesk Basic Practices !\[\]\(aa53ad6fea213b8b2226d3077e30533a_img.jpg\)\)](#)
[Overview of the RS232 Library Elements..... 7](#)

References

[GetNumInBytes..... 18](#)
[SetReadTimeout..... 22](#)
[Write..... 23](#)

RS232Configuration

Graphical representation

Purpose

To specify the configuration of the serial interface.

Description

The RS232Configuration data object contains the configuration of a serial interface. Several RS232Configuration data objects can be used in your project, each specifying a different serial interface. The following parameters are specified:

Parameter	Description
Port	The PC's serial port to be used. The predefined values are COM1 ... COM4, but you can also enter another port, for example, COM5. Each serial port can be set only once. By default, COM1 is used.
Bits per second	The number of bits per second transmitted via the serial interface. Valid values are: 110, 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 38400, 56000, 57600, 115200. By default, 9600 is set.
Data bits	The number of data bits to be used. Valid values are 5 ... 8. By default, 8 data bits are used.
Parity	The parity scheme to be used. Valid values are "NO", "ODD", "EVEN", "MARK", or "SPACE". By default, "NO" is set.
Stop bits	The number of stop bits to be used. Valid values are 1, 1.5, or 2. By default, 1 stop bit is used.
In-Buffer size	The input buffer size. You can enter any even value. By default, 5120 bytes is set.
Out-Buffer size	The output buffer size. You can enter any even value. By default, 5120 bytes is set.

Note

- Do not specify the following combinations of data bit and stop bit numbers:
 - Number of data bits = 5 and number of stop bits = 2
 - Number of data bits = 6, 7, or 8 and number of stop bits = 1.5
 These combinations lead to invalid transmissions.
- The input and output buffer sizes must be even values.

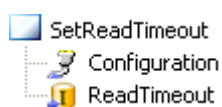
Related topics**Basics**

[Overview of the RS232 Library Elements..... 7](#)

References

[Edit \(RS232Configuration\)..... 25](#)
[GetNumInBytes..... 18](#)
[Read..... 19](#)
[SetReadTimeout..... 22](#)
[Write..... 23](#)
[WriteString..... 23](#)

SetReadTimeout

Graphical representation**Purpose**

To specify the read timeout.

Description

The read timeout specifies how long the Read block waits for data to arrive at the serial port's input buffer before reading is aborted and an exception is raised.

Data objects

This automation block provides the following data objects:

Name	In / Out	Data Type	Default Value	Description
Configuration	In	RS232Configuration	-	Contains the configuration of the RS232 interface.
ReadTimeout	In	Int	0	Contains the timeout value for reading in ms.

Related topics**Basics**

[Overview of the RS232 Library Elements..... 7](#)

References

[Read..... 19](#)

Write

Graphical representation



Purpose

To write a byte to the serial port.

Description

The ASCII value of the byte is written to the serial port's output buffer. The ASCII value must be specified as integer in the Edit dialog.

Data objects

This automation block provides the following data objects:

Name	In / Out	Data Type	Default Value	Description
Configuration	In	RS232Configuration	-	Contains the configuration of the RS232 interface.
ByteToWrite	In	Int	0	Contains the ASCII value of the byte to be written.

Related topics

Basics

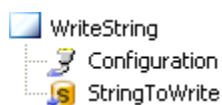
[Overview of the RS232 Library Elements..... 7](#)

References

[Read..... 19](#)
[WriteString..... 23](#)

WriteString

Graphical representation



Purpose

To write a string to the serial port.

Description

You can enter an arbitrary string to be written using the Edit dialog. The string is written to the serial port's output buffer.

Note

There is no guarantee that these strings are immediately transferred to the connected communication partner. Depending on the configuration of the serial interface, they can remain in the output buffer for a specified time. Thus, it is important that the serial interface of the communication partner is configured in the same way.

Data objects

This automation block provides the following data objects:

Name	In / Out	Data Type	Default Value	Description
Configuration	In	RS232Configuration	None	Contains the configuration of the RS232 interface.
StringToWrite	In	String	" "	Contains the string to be written.

Related topics**Basics**

[Overview of the RS232 Library Elements..... 7](#)

References

[Write..... 23](#)

Commands And Dialogs

Where to go from here	Information in this section
	Edit (RS232Configuration)..... 25
	To configure the serial interface of the host PC.
	Insert (RS232 Elements)..... 26
	To insert data objects of the RS232 library using the Home ribbon.

Edit (RS232Configuration)

Access	You can access this command via:	
	Ribbon	None
	Context menu of	<ul style="list-style-type: none">▪ Project Manager with a RS232Configuration data object selected or double-click▪ Data object column of the Data Object Editor
	Shortcut key	None
	Icon	None

Purpose	To configure the serial interface of the host PC.
---------	---

Result	The properties are assigned to the RS232Configuration data object.
--------	--

Description	The RS232Configuration data object is an element of the RS232 library. The RS232 configuration contains the serial port, transfer rate, number of data, start and stop bits, the sizes of the serial port's input and output buffers and the parity scheme.
-------------	---

RS232 Configuration dialog	<p>This dialog is opened when you choose Edit for a RS232Configuration data object or double-click the RS232Configuration data object.</p> <p>Port Lets you select the name of the PC's serial port. Valid values are COM1 ... COM4.</p> <p>Bits per second Lets you select the baud rate of the serial interface. All valid baud rates are displayed in the list.</p>
----------------------------	--

Data bits Lets you select the number of data bits to be used. The valid numbers are 5 ... 8. The default value is 8 bits.

Parity Lets you select the parity scheme to be used. The following schemes are available:

- No (default value of parity scheme)
- Odd
- Even
- Mark
- Space

Stop bits Lets you select the number of stop bits to be used. The valid numbers are 1, 1.5, or 2. The default value is 1 stop bit.

Note

Do not specify the following combinations of data bit and stop bit numbers:

- Number of bits = 5 and number of stop bits = 2
- Number of bits = 6, 7, or 8 and number of stop bits = 1.5

In-buffer size Lets you select the input buffer size from the list. At present, only the default input buffer size of 5120 bytes is available.

Out-buffer size Lets you select the output buffer size from the list. At present, only the default output buffer size of 5120 bytes is available.

Note

The value of In-buffer size and Out-buffer size must be an even number.

Related topics

Basics

[Overview of the RS232 Library Elements..... 7](#)

References

[RS232Configuration..... 20](#)

Insert (RS232 Elements)

Access

You can access the data objects of the RS232 library via:

Ribbon	Home - Insert - Data Objects
Context menu of	None

Shortcut key	None
Icon	 RS232 Configuration

Purpose To insert data objects of the RS232 library using the Home ribbon.

Description The ribbon commands are enabled or disabled according to context.
You can add data objects to your project in the Project Manager and to specific automation blocks in the Sequence Builder, such as Exec automation blocks. The data object is added to the selected element.

Related topics	Basics Overview of the RS232 Library Elements..... 7
-----------------------	--

Automation

Basics on Automating the Access to RS232

Introduction

AutomationDesk provides a COM-based API to automate the handling of AutomationDesk.

Related information

The AutomationDesk COM API provides the following object for configuring the access to a tool via serial interface using the RS232 library:

- [RS232Configuration](#) ([AutomationDesk Automation](#) )

For basic information and instructions, refer to [Basics and Instructions](#) on page 7.

Limitations

Limitations When Using the RS232 Library

Using the WriteString block

The WriteString block contains a String data object to parameterize the string to be written via RS232 interface. Since AutomationDesk 2.0, strings are handled with unicode characters. This conflicts with the string representation in the RS232 interface.

To avoid this problem, you have to use the methods from the `rs232lib2` in Exec blocks.

C

- Common Program Data folder 6
- Configuration RS232 dialog 25
- configuring serial interface 11

D

- Documents folder 6

E

- edit
 - RS232Configuration 25
- Edit command (RS232Configuration) 25

G

- GetNumInBytes 18

I

- inserting AutomationDesk library elements
 - RS232 26

L

- limitations
 - RS232 31
- Local Program Data folder 6

R

- Read (RS232) 19
- receiving data
 - RS232 14
- RS232
 - configuring 11
 - receiving data 14
 - sending data 12
- RS232 library
 - example 9
 - overview 7
- RS232Configuration 20

S

- sending data
 - RS232 12
- serial interface
 - configuring 11
- SetReadTimeout 22

W

- Write (RS232) 23
- WriteString 23

