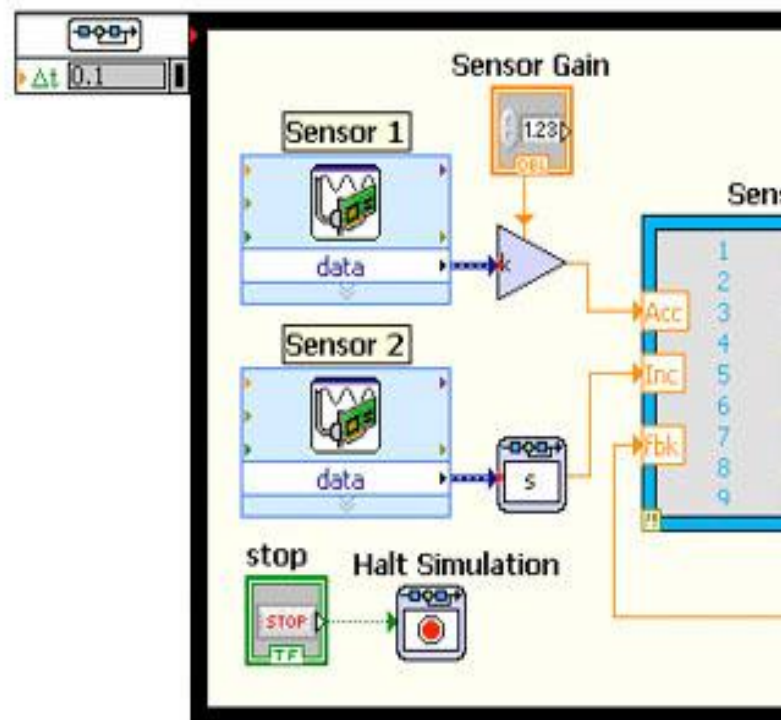
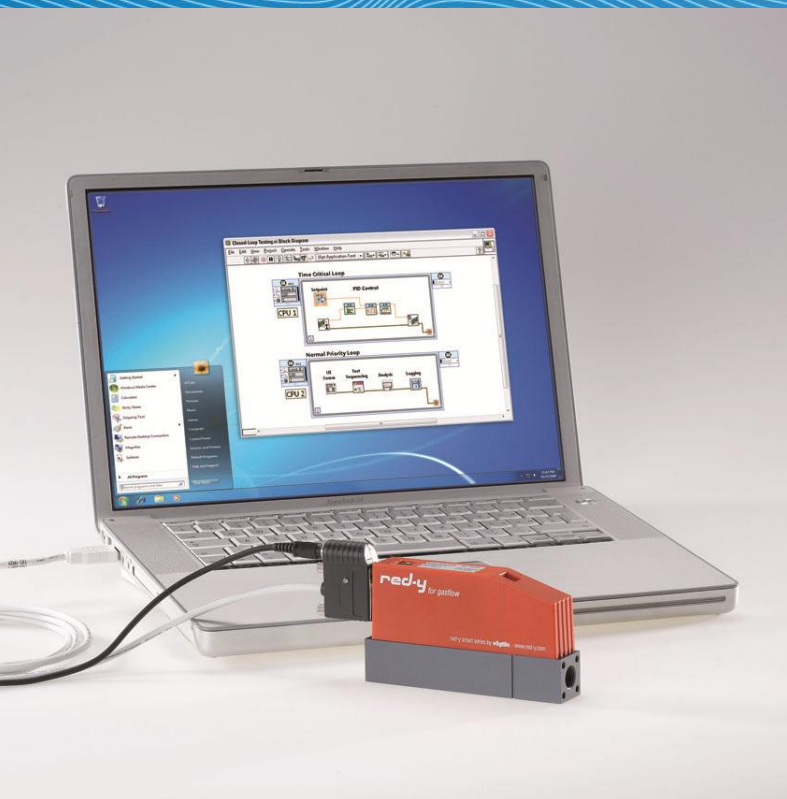


## LabView driver for red-y smart series Operating instructions



## LabView Driver for the thermal mass flow meters & controllers *red-y smart series*

# Operating instructions

LabView driver for the thermal mass flow meters & controllers  
*red-y smart series*

This manual describes the Virtual Instruments for LabView starting at version 8.6.



Version: LabView\_E1\_2

For the latest information on our products, see our website at [www.voegtlin.com](http://www.voegtlin.com)

© 2011 Vögtlin Instruments AG, Switzerland

## Copyright and Liability Disclaimer

All rights reserved. This publication or even just parts of it may not be reproduced without prior written permission of the publisher.

The contents of this instruction manual are intended exclusively for information purposes. Vögtlin Instruments AG assumes no responsibility or liability for any errors or inaccuracies in this publication.



This symbol alerts the user to important operating, maintenance and service information.

### Subject to change

Due to our policy of ongoing product development, we reserve the right to change the information in this manual without prior notice.

# Contents

<b>1. Introduction</b>	<b>5</b>
1.1 System requirements	5
1.2 What we supply	5
<b>2. Installation</b>	<b>6</b>
2.1 Installation of the RS485 interface	6
<b>3. Modbus Driver / Virtual Instruments</b>	<b>7</b>
<b>4. Examples of the communication sequence</b>	<b>10</b>
<b>5. Appendix</b>	<b>12</b>
5.1 Abbreviations	12
<b>6. Change history</b>	<b>13</b>

# 1. Introduction

These National Instruments LabView Drivers (Virtual Instruments, VI) allow a simple read and write access to mass flow meter and controller as well as the pressure controllers of the *red-y smart series*. You can access any register of your red-y device directly from LabView without special hardware or Modbus knowledge. The simplest to complex systems can be effectively created with LabView.

Several devices can be addressed on one interface via a RS485 communications connection. The manual will explain the installation and use of the VIs.

## 1.1 System requirements

Please refer to the manual of National Instruments for the exact definition of the system requirement of the LabView 8.6 software and newer. [www.ni.com](http://www.ni.com)

### Operating system:

- The drivers were created for LabView 8.6 and newer. Windows XP and Windows 7 were used as the operating systems.

### Interfaces:

- The software supports the RS232 and RS485 interfaces. We recommend our interface converter cable USB-> RS485 "PDM-U"

## 1.2 What we supply

The software can be downloaded via our homepage free of charge.

Service and Support: [www.voegtlin.com/downloads](http://www.voegtlin.com/downloads)

You can also find the software on every CD-ROM enclosed with each shipment.

### Minimum accessory for the connection to a PC

1 PDM-U USB-RS485 Adapter	Item No. 328-2169
1 AC adapter plug PSD (8W)	Item No. 328-2234
1 AC adapter plug PSD (53W)	Item No. 328-2233

For additional accessories for operation with more than one device on a bus see „*Cable accessories for red-y smart series*“, datasheet 329-2014.

## 2. Installation

The following components must be available for startup:

- Computer with LabView V8.6 or newer
- Communications cable

*PDM-U USB-RS485 adapter, or RS-485 interface converter cable*

- 24V Power supply unit (min. 8W)
- Instruments or Controllers

*smart 4 and newer*

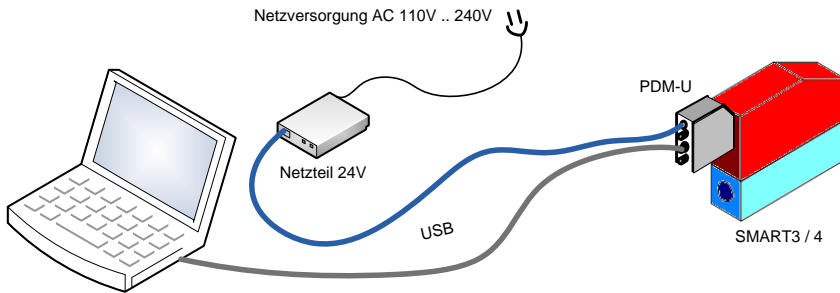


Figure 1: Configuration of the measuring station

### 2.1 Installation of the RS485 interface

The connection to the device can be made via an USB – RS485 adapter.

For the *smart* devices *Vögtlin Instruments AG* offers an adapter, which is geared to meet the requirements. A female connector on the D-Sub plug is used to feed the 24V DC supply voltage.



Before the USB connector can be connected to the computer the driver software for the RS485 product must be installed. In the case of Windows 7 the driver will be automatically installed if an Internet connection exists. The *Vögtlin Instruments AG* installation CD contains one driver. The user must make certain that the driver is compatible with the operating system.

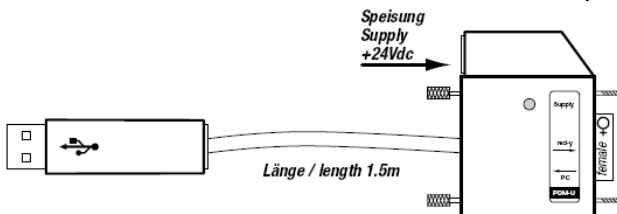


Figure 2: PDM-U USB-RS485 Adapter

### 2.2 Software

The following software must be installed on your computer (VISA and serial drivers are mostly included in a LabView standard installation):

- **LabView 8.6 or newer**
- **NI-VISA driver** (LabView Installations CD or [www.ni.com](http://www.ni.com))
- **NI-Serial driver** (LabView Installations CD or [www.ni.com](http://www.ni.com))

**LabView SMART driver:** The drivers are organized as LabView project library. All files associated with the library are in the folder "Vögtlin NI SMART Modbus Drivers". In order to add the driver library to your project and use it, proceed as follows:

- **Copy the "Vögtlin NI SMART Modbus Drivers" folder to your local hard drive.** Ensure that LabView always has access to this folder. It is best that you copy it directly into your project folder.
- **Copy the "Vögtlin NI SMART Modbus Drivers" folder to your local hard drive.** Ensure that LabView always has access to this folder. It is best that you copy it directly into your project folder.
- **You can add any virtual instrument (VI) from the copied folder to your LabView project tree.** (Right click → Add File or simply perform a Drag&Drop in Windows explorer). The complete library is now automatically added to your project.  
*If you are not using a project, then you can also insert the VIs directly into a block diagram using Drag&Drop (not recommended).*

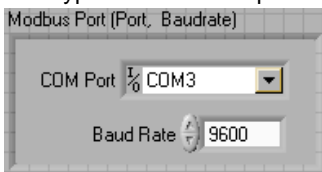
You can only use the VIs in the root folder of the driver library. All VIs in the "Low level Driver" folder are private<sup>1</sup> members of the library and can only be used by library internal VIs.

### 3. Modbus Driver / Virtual Instruments

You are able to use the following data structure and driver VIs in your LabView program.

#### **Port** Port data structure

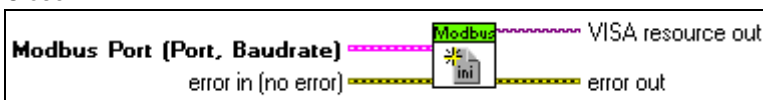
This type definition comprises the three parameters required for a communication configuration.



- I/O** **COM Port** to which the devices are connected
- U32** **Baud Rate** defines the communication speed

#### **Modbus Init VI**

This VI sets up a Modbus communication channel. Once set-up, the parameter of the communication channel can no longer be modified. However, you can have several communication channels open at the same time by selecting Modbus Init several times with different COM Ports. You then must individually close each open channel using Modbus Close.






- Port** **Modbus Port (port, baud rate)** defines the access parameter for the Bus to be initialized.
- I/O** **VISA resource out** is the reference of the open communication channel
- Bool** **error in (no error)** describes the prevailing error status prior to performing the VI
- Bool** **error out** contains the error information

<sup>1</sup> This only applies for the LabView 2010 project folder. LabView 8.6 has known problems with polymorphic VIs and their scope in a library, which is why all library members are public.

## Modbus Close VI

This VI closes the open Modbus communication channel.

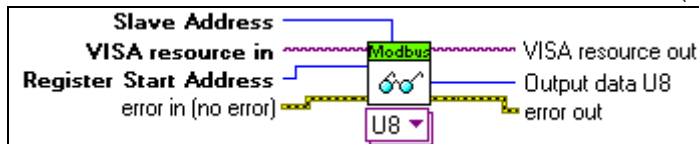









-  **VISA resource in** is the reference of the to be closed communication channel
-  **error in (no error)** describes the prevailing error status prior to performing the VI
-  **error out** contains the error information

## Modbus Read VI

This VI reads the value of a selected data type of the indicated register address. You can select the following data types via the selector: U8, U16, U32, SGL (*float32*), STR8 (*8 character string*), STR50 (*50 character string*).

The device to be addressed is identified via "VISA resource in" (Bus) and "Slave Address" (device address).



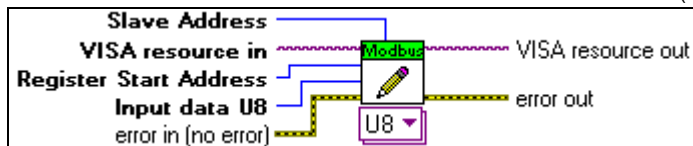
-  **VISA resource in** is the reference to a channel via which the communication must be performed
-  **Slave Address** contains the device address from which reading should be done
-  **Register Start Address** contains the register address from which reading is done
-  **VISA resource out** is the reference to a channel, which is used currently for communication
-  **Output data xx** contains the read value in form of the selected data type
-  **error in (no error)** describes the prevailing error status prior to performing the VI
-  **error out** contains the error information








## Modbus Write VI

This VI writes a value of the selected data type to the indicated register address. You can select the following data types via the selector: U8, U16, U32, SGL (*float32*), STR8 (*8 character string*), STR50 (*50 character string*).

Remember, depending on data type several registers can be written (one register comprises 16bit).

The device to be addressed is identified via "VISA resource in" (Bus) and "Slave Address" (device address).



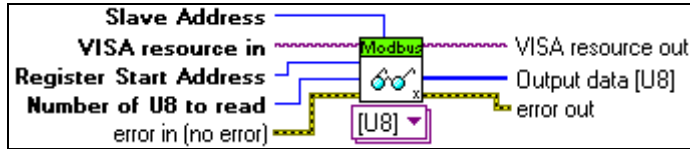
-  **VISA resource in** is the reference to a channel via which the communication must be performed
-  **Slave Address** contains the device address to which writing should be done
-  **Register Start Address** contains the register address to which writing is done
-  **Output data xx** contains the value to be written in form of the selected data type
-  **VISA resource out** is the reference to a channel, which is used currently for communication
-  **error in (no error)** describes the prevailing error status prior to performing the VI
-  **error out** contains the error information











## Modbus Multi-Read VI

This VI reads several values of the selected data type from the indicated register address (and following registers). You can select the following data types via the selector: [U8], [U16], [U32], [SGL] (*float32*), STR (*String with any amount of characters*).

The device to be addressed is identified via "VISA resource in" (Bus) and "Slave Address" (device address).

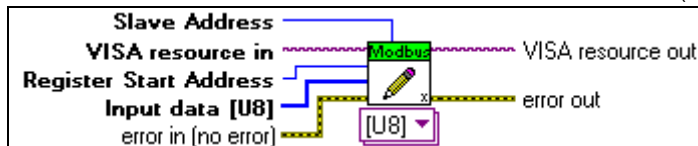









-  **VISA resource in** is the reference to a channel via which the communication must be performed
-  **Slave Address** contains the device address from which reading should be done
-  **Register Start Address** contains the register address from which reading is done
-  **Number of xx to read** indicates how many values are read from the selected data type
-  **VISA resource out** is the reference to a channel, which is used currently for communication
-  **Output data xx (array)** contains the read values in form of an array of the selected data type
-  **error in (no error)** describes the prevailing error status prior to performing the VI
-  **error out** contains the error information

## Modbus Multi-Write VI

This VI writes several values of the selected data type from the indicated register address (and following registers). You can select the following data types via the selector: [U8], [U16], [U32], [SGL] (*float32*), STR (*String with any amount of characters*).

The device to be addressed is identified via "VISA resource in" (Bus) and "Slave Address" (device address).



-  **VISA resource in** is the reference to a channel via which the communication must be performed
-  **Slave Address** contains the device address to which writing should be done
-  **Register Start Address** contains the register address from which writing is done
-  **Input data xx (array)** contains the to be written data in form of an array of the selected data type
-  **VISA resource out** is the reference to a channel, which is used currently for communication
-  **error in (no error)** describes the prevailing error status prior to performing the VI
-  **error out** contains the error information

## 4. Examples of the communication sequence

Each communication sequence, whether it is for reading or writing, consists of three vital steps:

- **Initialization:** The communication channel is established, required resources are reserved.
- **Communication:** Here, you can perform any read or write operations.
- **Close:** The communication channel is closed, bound resources are released.

In the following a short example program for writing and reading a register is created:

### Initialization:

1. Place the Modbus Init VI onto an empty block diagram.
2. Create a constant for the "Modbus Port" by right clicking on the icon on the port and selecting "Create → Constant". An instance of the "Port" type is created on your block diagram (refer to Chapter 3).
3. Configure this constant with the COM port and Baud rate parameter. All devices connected to the same bus must communicate with the same Baud rate.

**Writing the control parameter N in register 0x6208 from device 1:** (if required, check the register address in your device manual):

4. Place the Modbus Write VI onto your block diagram.
5. Connect the error cluster of the Init VI with the "error in" input and guide the "VISA resource" reference from Init VI to Write VI.
6. Select the data format U16 and create one constant each for the "Slave Address" input, "Register Start Address" input and the "Input data U16 (array)".
7. Set the slave address of the device to be controlled.
8. Select the value to be written and configure the start address. **Caution, the register address might has to be entered as decimal value.** Make the selected representation format of the constant visible (right-click → Visible Items → Radix) to ensure that the entered value is correctly interpreted. You can also change the representation format (right-click → Properties → Display Format).

**Reading the control parameter N from register 0x6208 from device 1:**

9. Place the Modbus Read VI onto your block diagram and select data type U16.
10. Connect the error cluster and "Visa resource" reference.
11. Connect the "Slave Address" and the "Register Start Address" input in the same manner as Write VI.
12. Create an indicator for output "Output data U16".

**Close:**

13. Place a Modbus Close VI onto your block diagram and connect the Error Cluster and VISA Reference.
14. Create a "Simple Error Handler" for the "error out" output (in the function range under Dialog & User Interface) to display possible errors.

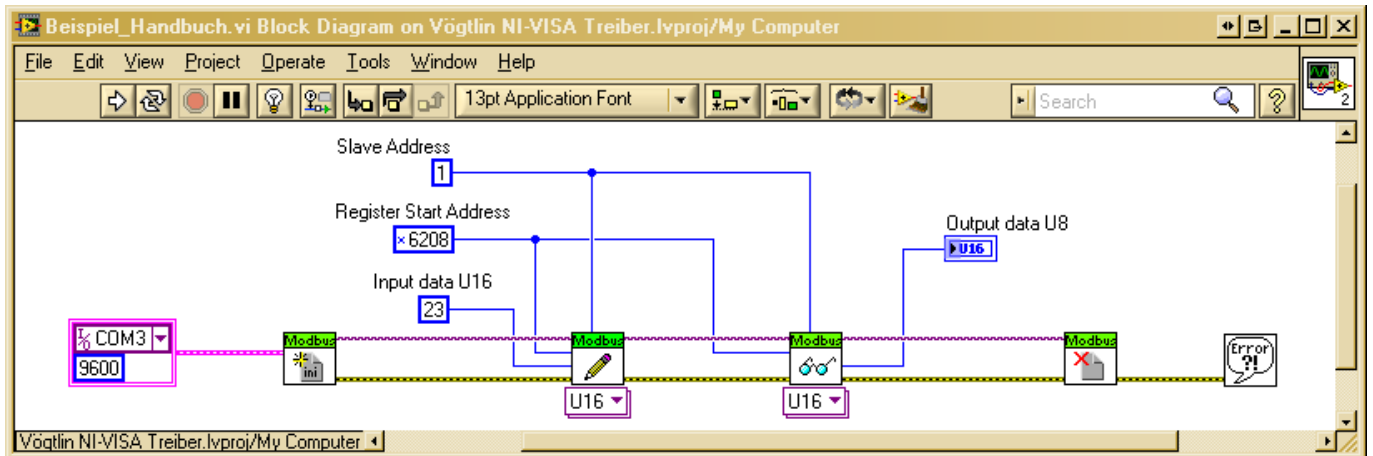


Figure 3: LabView schema view

Now you can test the program. If you did this correctly, then the control parameter N is written, read again and indicated on your front panel.

You are able to initialize several communication channels at the same time and therefore communicate on several channels. For this, just open the Modbus Init VI several times on different COM ports. You can also operate and access several devices on the same bus via the Slave Address.

## 5. Appendix

### 5.1 Abbreviations

<b>GSC</b>	red-y smart controller, Mass flow controller for gases
<b>GSM</b>	red-y smart meter, Mass flow meter for gases
<b>MFC</b>	Mass Flow Controller for gases
<b>MFM</b>	Mass flow meter for gases
<b>red-y smart series / smart</b>	Mass flow meter and controller from Vögtlin Instruments AG
<b>red-y compact series / compact</b>	Mass flow meter from Vögtlin Instruments AG with integrated display
<b>smart 3</b>	MFC/MFM red-y smart of the third generation (up to SN 109999)
<b>smart 4</b>	MFC/MFM red-y smart of the fourth generation (beginning with SN 110000)
<b>VIAG</b>	Vögtlin Instruments AG
<b>DUT</b>	Device Under Test
<b>GUI</b>	Graphical User Interface
<b>VI</b>	Virtual Instruments
<b>Calibrating</b>	Compare measured values between device under test and reference
<b>Adjustment</b>	Synchronizing or matching of a device under test to the reference
<b>Verifying</b>	Checking a device under test in comparison to the reference after the adjustment

Figure 1: Configuration of the measuring station .....	6
Figure 2: PDM-U USB-RS485 Adapter .....	6
Figure 3: LabView schema view .....	11

## 6. Change history

Date	Version	Replaces	Author	Note
25.03.2011	LabView_D1_0	<b>new</b>	MRZ	new
09.05.2011	LabView_D1_2	LabView_D1_0	HIE	Final corrigenda