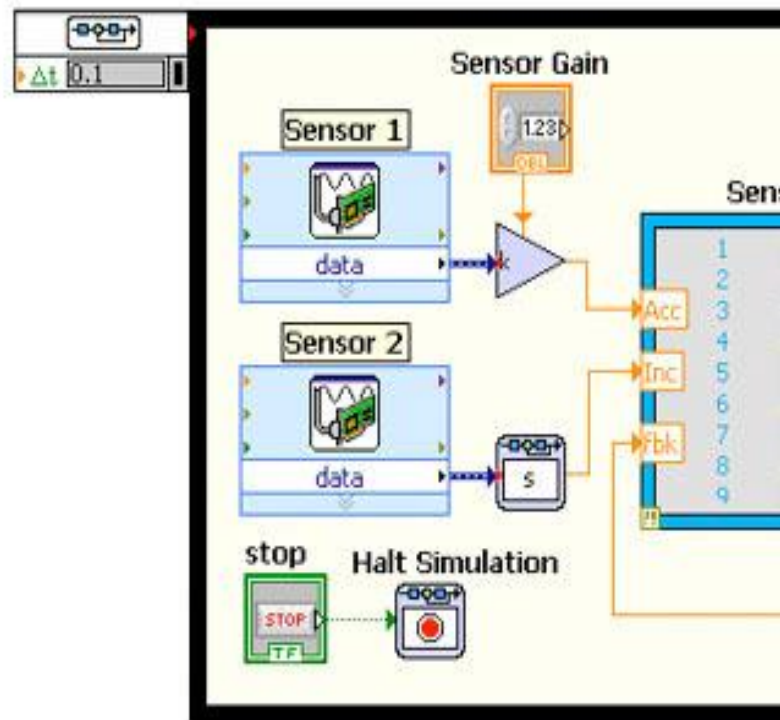
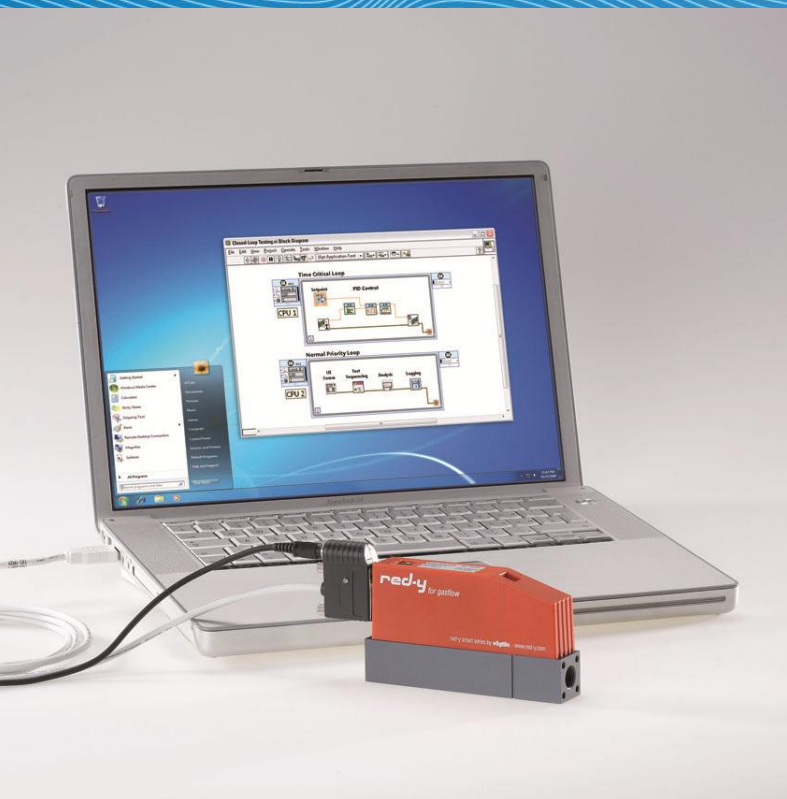


## LabView Treiber für red-y smart series Bedienungsanleitung



## LabView Treiber für die thermischen Massedurchflussmesser und Regler red-y smart series

# Bedienungsanleitung

LabView Treiber für die thermischen Massedurchflussmesser und Regler  
*red-y smart series*.

Diese Anleitung beschreibt die Virtuellen Instrumente für LabView  
ab Version 8.6.



Version: get\_redy\_D1\_2

Aktuelle Informationen zu unseren Produkten finden Sie im Internet unter [www.voegtlin.com](http://www.voegtlin.com)

© 2011 Vögtlin Instruments AG, Switzerland

## Urheberrecht und Haftungsausschluss

Alle Rechte vorbehalten. Diese Publikation oder auch nur Teile davon dürfen nicht ohne vorherige, schriftliche Genehmigung des Herausgebers reproduziert werden.

Der Inhalt dieser Bedienungsanleitung dient ausschließlich Informationszwecken. Vögtlin Instruments AG übernimmt keine Verantwortung oder Haftung für allfällige Fehler oder Ungenauigkeiten in dieser Publikation.



Dieses Symbol weist den Anwender auf wichtige Bedienungs-, Wartungs- und Serviceinformationen hin.

## Änderungsvorbehalt

Aufgrund der kontinuierlichen Weiterentwicklung unserer Produkte behalten wir uns vor, die Angaben in diesem Handbuch ohne vorherige Ankündigung zu ändern.

# Inhaltsverzeichnis

<b>1. Einleitung</b>	<b>5</b>
1.1 Systemanforderungen	5
1.2 Lieferumfang	5
<b>2. Installation</b>	<b>6</b>
2.1 Installation der RS485 Schnittstelle	6
2.2 Software	6
<b>3. Modbus Treiber / Virtuelle Instrumente</b>	<b>7</b>
<b>4. Beispiel einer Kommunikationssequenz</b>	<b>10</b>
<b>5. Anhang</b>	<b>12</b>
5.1 Abkürzungen	12
<b>6. Änderungsverzeichnis</b>	<b>13</b>

# 1. Einleitung

Die vorliegenden National Instruments LabView Treiber (Virtuelle Instrumente, VI) erlauben einen sehr einfachen Lese- und Schreibzugriff auf die Massedurchflussmesser- und Regler sowie auf die Druckregler *red-y smart series*. Damit können Sie ohne spezielle Hardware oder Modbus-Kenntnisse auf beliebige Register Ihrer red-y Geräte direkt aus LabView zugreifen. Einfachste bis komplexe Systemen können mit LabView sehr effektiv erstellt werden.

Über eine RS485 Kommunikationsverbindung lassen sich mehrere Geräte auf einer Schnittstelle ansprechen.

Diese Anleitung erklärt die Installation und die Verwendung der VIs.

## 1.1 Systemanforderungen

Für die exakte Definition der Systemvoraussetzung der Software LabView 8.6 und neuer, konsultieren sie bitte das Handbuch von National Instruments: [www.ni.com](http://www.ni.com)

### Betriebssystem:

- Die Treiber wurden für LabView 8.6 und neuer erstellt. Hierbei wurden die Betriebssysteme Windows XP und Windows 7 benutzt.

### Schnittstellen:

- Die Software unterstützt die Schnittstellen RS232 und RS485. Wir empfehlen unser Schnittstellenkonverter Kabel USB-> RS485 „PDM-U“

## 1.2 Lieferumfang

Das Software kann auf unserer Homepage kostenfrei heruntergeladen werden.

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

Zusätzlich finden Sie die Software auf jeder CD-Rom, die pro Sendung mitgeliefert wird.

### Minimales Zubehör für die Verbindung auf einen PC

1 Stück PDM-U USB-RS485 Adapter	Artikel Nr. 328-2169
1 Stück Steckernetzteil PSD (8W)	Artikel Nr. 328-2234
1 Stück Steckernetzteil PSD (53W)	Artikel Nr. 328-2233

Weiteres Zubehör für den Betrieb mit mehreren Geräten auf einem Bus Siehe Datenblatt „*Kabelzubehör zu red-y smart series*“, 329-2014.

## 2. Installation

Für die Inbetriebnahme müssen folgende Komponenten zur Verfügung stehen:

- Computer mit LabView V8.6 oder neuer
- Kommunikationskabel *PDM-U USB-RS485 Adapter, oder RS-485 Schnittstellen Konvertierkabel*
- 24V Netzteil (mind. 8W)
- Mess- oder Regelgeräte *smart 4 und neuer*

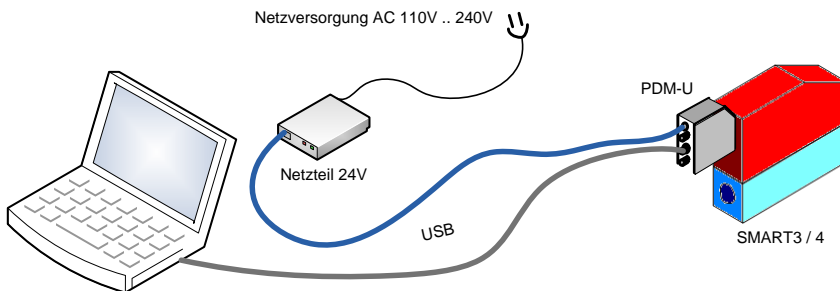


Abbildung 1: Aufbau des Messplatzes

### 2.1 Installation der RS485 Schnittstelle

Die Verbindung zum Gerät kann über einen USB – RS485 Adapter aufgebaut werden.

Für die *smart* Geräte bietet *Vögtlin Instruments AG* einen Adapter an, welcher auf die Anforderungen abgestimmt ist. Eine Buchse am D-Sub Stecker dient für die Einspeisung der 24V DC Versorgungsspannung.



Bevor der USB Stecker mit dem Computer verbunden wird, muss die Treibersoftware für das RS485 Produkt installiert werden. Bei Windows 7 wird der Treiber automatisch installiert, wenn eine Internetverbindung besteht. Auf der Installations-CD von *Vögtlin Instruments AG* ist ein Treiber vorhanden. Der Anwender sollte sicherstellen, dass der Treiber mit dem Betriebssystem kompatibel ist.

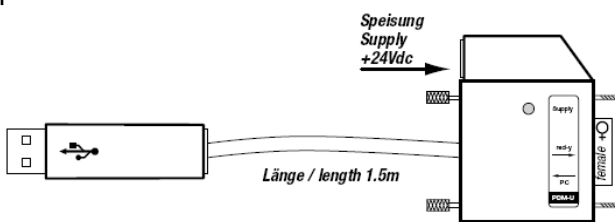


Abbildung 2: PDM-U USB-RS485 Adapter

### 2.2 Software

Folgende Software muss auf Ihrem Computer installiert sein (VISA- und Serial-Treiber sind bei einer LabView Standardinstallation meist schon dabei):

- **LabView 8.6 oder neuer**
- **NI-VISA Treiber** (LabView Installations-CD oder [www.ni.com](http://www.ni.com))
- **NI-Serial Treiber** (LabView Installations-CD oder [www.ni.com](http://www.ni.com))

**LabView SMART Treiber:** Die Treiber sind als LabView Projektbibliothek organisiert. Alle zur Bibliothek gehörenden Dateien befinden sich im Ordner "Vögtlin NI SMART Modbus Drivers". Um die Treiberbibliothek Ihrem Projekt zuzufügen und zu benutzen gehen Sie wie folgt vor:

- **Kopieren Sie den Ordner "Vögtlin NI SMART Modbus Drivers" auf Ihre lokale Festplatte.** Stellen Sie sicher, dass LabView immer Zugriff auf diesen Ordner hat. Am besten kopieren Sie ihn direkt in Ihren Projektordner.
- **Kopieren Sie den Ordner "Vögtlin NI SMART Modbus Drivers" auf Ihre lokale Festplatte.** Stellen Sie sicher, dass LabView immer Zugriff auf diesen Ordner hat. Am besten kopieren Sie ihn direkt in Ihren Projektordner.
- **Fügen Sie ein beliebiges Virtuelles Instruments (VI) aus dem kopierten Ordner Ihrem LabView-Projektbaum hinzu.** (Rechtsklick → Add File oder ganz einfach durch Drag&Drop aus dem Windowexplorer). Die ganze Bibliothek wird nun automatisch Ihrem Projekt hinzugefügt.  
*Wenn Sie kein Projekt verwenden, können Sie die VIs auch direkt per Drag&Drop in ein Blockdiagramm einfügen (nicht empfohlen).*

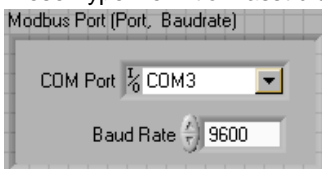
Sie können nur die VIs im Stammordner der Treiberbibliothek verwenden. Alle VIs im Ordner "Low Level Driver" sind private<sup>1</sup> Member der Bibliothek und können nur von bibliothekinternen VIs verwendet werden.

### 3. Modbus Treiber / Virtuelle Instrumente

Die folgende Datenstruktur und Treiber VIs können Sie in Ihrem LabView Programm verwenden.

#### **Port** Port Datenstruktur

Diese Type Definition fasst die drei für einen Kommunikationsaufbau notwendigen Parameter zusammen.

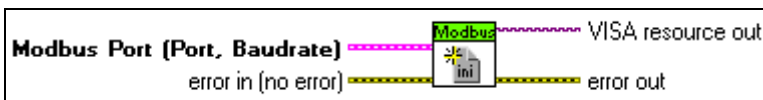


**I/O** COM Port an welchem die Geräte angeschlossen sind

**U32** Baud Rate definiert Kommunikationsgeschwindigkeit

#### **Modbus Init VI**

Dieses VI baut einen Modbus Kommunikationskanal auf. Einmal aufgebaut, können die Parameter des Kommunikationskanals nicht mehr geändert werden. Sie können aber mehrere Kommunikationskanäle gleichzeitig geöffnet haben, indem Sie Modbus Init mehrfach mit verschiedenen COM-Ports aufrufen. Sie müssen jedoch jeden geöffneten Kanal einzeln auch wieder mittels Modbus Close schliessen.




**Port** Modbus Port (Port, Baudrate) definiert die Zugriffsparameter für den zu initialisierenden Bus

**I/O** VISA resource out ist die Referenz auf den geöffneten Kommunikationskanal

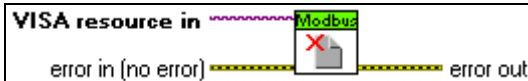
**U32** error in (no error) beschreibt den vorherrschenden Fehlerzustand bevor das VI ausgeführt wird


<sup>1</sup> Dies gilt nur für den Projektordner für LabView 2010. In LabView 8.6 gibt es bekannte Probleme mit polymorphen VIs und deren Scope in einer Library, weshalb alle Library-Members public sind.


 **error out** enthält die Fehlerinformation


### Modbus Close VI

Dieses VI schliesst einen offenen Modbus Kommunikationskanal.



 **VISA resource in** ist die Referenz auf den zu schliessenden Kommunikationskanal

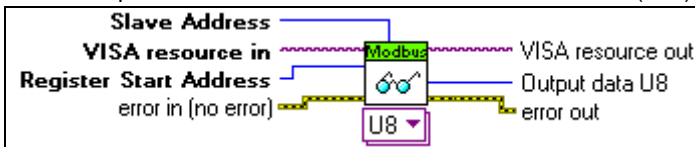
 **error in (no error)** beschreibt den vorherrschenden Fehlerzustand bevor das VI ausgeführt wird


 **error out** enthält die Fehlerinformation

### Modbus Read VI

Dieses VI liest einen Wert des gewählten Datentyps an der angegebenen Registeradresse aus. Über den Selector können Sie aus folgenden Datentypen wählen: U8, U16, U32, SGL (*float32*), STR8 (*8 Zeichen String*), STR50 (*50 Zeichen String*).


Das anzusprechende Gerät wird mittels "VISA resource in" (Bus) und "Slave Address" (Geräteadresse) identifiziert.




 **VISA resource in** ist die Referenz auf den Kanal, über welchen die Kommunikation ausgeführt werden soll

 **Slave Address** enthält die Adresse des Gerätes von welchem gelesen werden soll

 **Register Start Address** enthält die Registeradresse, ab welcher ausgelesen wird

 **VISA resource out** ist die Referenz auf den Kanal, über welchen gerade kommuniziert wurde

 **Output data xx** enthält den ausgelesenen Wert in Form des gewählten Datentyps

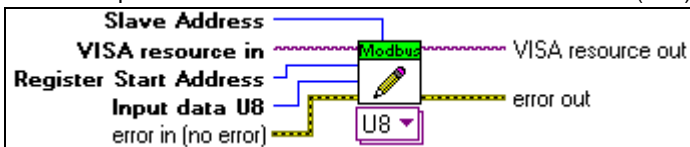
 **error in (no error)** beschreibt den vorherrschenden Fehlerzustand bevor das VI ausgeführt wird


 **error out** enthält die Fehlerinformation

### Modbus Write VI

Dieses VI schreibt einen Wert des gewählten Datentyps an die angegebene Registeradresse. Über den Selector können Sie aus folgenden Datentypen wählen: U8, U16, U32, SGL (*float32*), STR8 (*8 Zeichen String*), STR50 (*50 Zeichen String*). Bedenken Sie, dass je nach Datentyp mehrere Register geschrieben werden (Ein Register umfasst 16bit).

Das anzusprechende Gerät wird mittels "VISA resource in" (Bus) und "Slave Address" (Geräteadresse) identifiziert.



 **VISA resource in** ist die Referenz auf den Kanal, über welchen die Kommunikation ausgeführt werden soll

 **Slave Address** enthält die Adresse des Gerätes auf welches geschrieben werden soll


 **Register Start Address** enthält die Registeradresse, an welche geschrieben wird

 **Input data xx** enthält den zu schreibenden Wert in Form des gewählten Datentyps

 **VISA resource out** ist die Referenz auf den Kanal, über welchen gerade kommuniziert wurde

 **error in (no error)** beschreibt den vorherrschenden Fehlerzustand bevor das VI ausgeführt wird

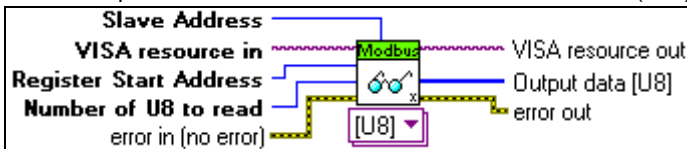



 **error out** enthält die Fehlerinformation

### Modbus Multi-Read VI

Dieses VI liest mehrere Werte des gewählten Datentyps ab der angegebenen Registeradresse (und folgende Register) aus. Über den Selector können Sie aus folgenden Datentypen wählen: [U8], [U16], [U32], [SGL] (*float32*), STR (*String mit beliebiger Anzahl Zeichen*).


Das anzusprechende Gerät wird mittels "VISA resource in" (Bus) und "Slave Address" (Geräteadresse) identifiziert.




 **VISA resource in** ist die Referenz auf den Kanal, über welchen die Kommunikation ausgeführt werden soll

 **Slave Address** enthält die Adresse des Gerätes von welchem gelesen werden soll


 **Register Start Address** enthält die Registeradresse, ab welcher ausgelesen wird

 **Number of xx to read** gibt an, wie viele Werte vom gewählten Datentyp gelesen werden

 **VISA resource out** ist die Referenz auf den Kanal, über welchen gerade kommuniziert wurde

 **Output data xx (array)** enthält die ausgelesenen Werte in Form eines Arrays des gewählten Datentyps

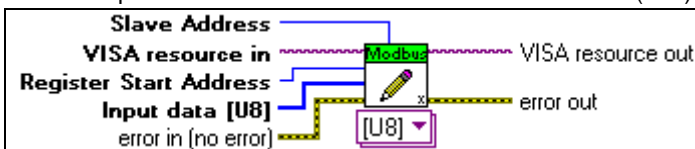
 **error in (no error)** beschreibt den vorherrschenden Fehlerzustand bevor das VI ausgeführt wird


 **error out** enthält die Fehlerinformation

### Modbus Multi-Write VI

Dieses VI schreibt mehrere Werte des gewählten Datentyps ab der angegebenen Registeradresse (und folgende Register). Über den Selector können Sie aus folgenden Datentypen wählen: [U8], [U16], [U32], [SGL] (*float32*), STR (*String mit beliebiger Anzahl Zeichen*).

Das anzusprechende Gerät wird mittels "VISA resource in" (Bus) und "Slave Address" (Geräteadresse) identifiziert.





 **VISA resource in** ist die Referenz auf den Kanal, über welchen die Kommunikation ausgeführt werden soll

 **Slave Address** enthält die Adresse des Gerätes auf welches geschrieben werden soll

 **Register Start Address** enthält die Registeradresse, ab welcher geschrieben wird

 **Input data xx (array)** enthält die zu schreibenden Daten in Form eines Arrays des gewählten Datentyps

 **VISA resource out** ist die Referenz auf den Kanal, über welchen gerade kommuniziert wurde

 **error in (no error)** beschreibt den vorherrschenden Fehlerzustand bevor das VI ausgeführt wird

 **error out** enthält die Fehlerinformation

## 4. Beispiel einer Kommunikationssequenz

Jede Kommunikationssequenz, sei es Lesen oder Schreiben, besteht zwingend aus drei Schritten:

- **Initialisierung:** Der Kommunikationskanal wird aufgebaut, benötigte Ressourcen werden reserviert.
- **Kommunikation:** Hier können Sie beliebige Lese- und Schreiboperationen ausführen.
- **Schliessen:** Der Kommunikationskanal wird geschlossen, gebundene Ressourcen wieder freigegeben.

Im Folgenden soll ein kurzes Beispielprogramm zum Schreiben und Lesen eines Registers erstellt werden:

### Initialisierung:

1. Platzieren Sie ein Modbus Init VI auf ein leeres Blockdiagramm.
2. Erstellen Sie für den Eingang "Modbus Port" eine Konstante, indem Sie auf den Port am Icon rechtsklicken und "Create → Constant" wählen. Auf Ihrem Blockdiagramm wird eine Instanz des Typs "Port" (siehe Kapitel 3) erstellt.
3. Konfigurieren Sie diese Konstante mit den Parametern COM-Port und Baud Rate. Alle am gleichen Bus angeschlossenen Geräte müssen mit der gleichen Baud Rate kommunizieren.

**Schreiben des Regelparameters N in Register 0x6208 von Gerät 1:** (überprüfen Sie die Registeradresse gegebenenfalls in Ihrem Gerätehandbuch):

4. Platzieren Sie ein Modbus Write VI auf Ihrem Blockdiagramm.
5. Verbinden Sie den Error-Cluster des Init VI mit dem "error in" Eingang und führen Sie die "VISA resource" Referenz vom Init VI zum Write VI.
6. Wählen Sie das Datenformat U16 und erstellen Sie jeweils eine Konstante für die Eingänge "Slave Address", "Register Start Address" und "Input data U16 (array)".
7. Setzen Sie die Slave Adresse des Gerätes, welches angesteuert werden soll.
8. Wählen Sie den zu schreibenden Wert und konfigurieren Sie die Startadresse. **Achtung, die Registeradresse muss eventuell als Dezimalwert eingegeben werden.** Machen Sie das gewählte Darstellungsformat der Konstante sichtbar (Rechtsklick → Visible Items → Radix) um sicher zu gehen, dass der eingegebene Wert richtig interpretiert wird. Sie können das Darstellungsformat auch ändern (Rechtsklick → Properties → Display Format).

**Lesen des Regelparameters N aus Register 0x6208 von Gerät 1:**

9. Platzieren Sie ein Modbus Read VI auf Ihrem Blockdiagramm und wählen Sie den Datentyp U16.
10. Verbinden Sie wiederum Error-Cluster und "VISA resource" Referenz.
11. Verbinden Sie den "Slave Address" und den "Register Start Address"-Eingang gleich wie beim Write VI.
12. Erstellen Sie einen Indikator für den Ausgang "Output data U16".

**Schliessen:**

13. Platzieren Sie ein Modbus Close VI auf Ihrem Blockdiagramm und verbinden Sie Error Cluster und VISA Referenz.
14. Erstellen Sie für den "error out" Ausgang einen "Simple Error Handler" (in der Funktionspalette unter Dialog & User Interface) um gegebenenfalls Fehler anzeigen zu lassen.

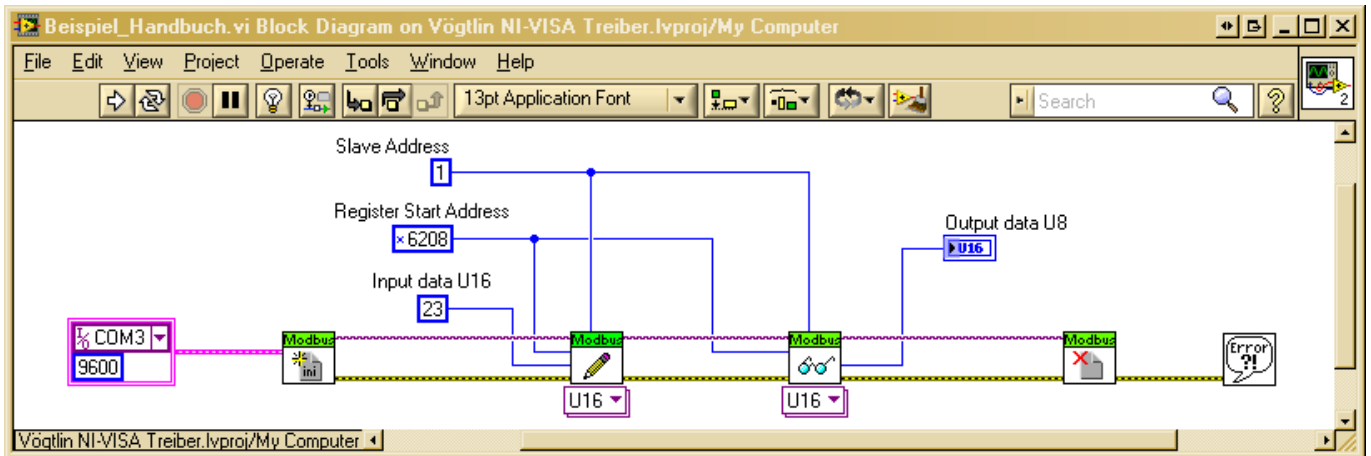


Abbildung 3 LabView Schemaansicht

Sie können das Programm jetzt testen. Wenn Sie alles richtig gemacht haben, wird der Regelparameter N geschrieben, wieder ausgelesen und auf Ihrem Front Panel angezeigt.

Sie können natürlich zur gleichen Zeit auch mehrere Kommunikationskanäle initialisieren und somit auf mehreren Kanälen kommunizieren. Rufen Sie dazu das Modbus Init VI einfach mehrfach mit verschiedenen COM-Ports auf. Ebenfalls können Sie über die Slave Adresse mehrere Geräte am selben Bus betreiben und ansprechen.

## 5. Anhang

### 5.1 Abkürzungen

<b>GSC</b>	red-y smart controller, Massedurchflussregler für Gase
<b>GSM</b>	red-y smart meter, Massedurchflussmesser für Gase
<b>MFC</b>	Mass Flow Controller, Massedurchflussregler für Gase
<b>MFM</b>	Mass Flow Meter, Massedurchflussmesser für Gase
<b>red-y smart series / smart</b>	Massedurchflussmesser und Regler von Vögtlin Instruments AG
<b>red-y compact series / compact</b>	Massedurchflussmesser von Vögtlin Instruments AG mit integrierter Anzeige
<b>smart 3</b>	MFC/MFM red-y smart der dritten Generation (bis SN 109999)
<b>smart 4</b>	MFC/MFM red-y smart der vierten Generation (ab SN 110000)
<b>VIAG</b>	Vögtlin Instruments AG
<b>DUT</b>	Device Under Test, Prüfling
<b>GUI</b>	Graphical User Interface, Grafische Benutzeroberfläche
<b>VI</b>	Virtuelle Instrumente
<b>Kalibrieren</b>	Messwerte zwischen Prüfling und Referenz vergleichen
<b>Justieren</b>	Abgleichen bzw. Angleichen eines Prüflings an die Referenz
<b>Verifizieren</b>	Überprüfen eines Prüflings im Vergleich zur Referenz nach der Justierung

Abbildung 1: Aufbau des Messplatzes .....	6
Abbildung 2: PDM-U USB-RS485 Adapter.....	6
Abbildung 3 LabView Schemaansicht .....	11

## 6. Änderungsverzeichnis

Datum	Version	Ersetzt	Autor	Notiz
25.03.2011	LabView_D1_0	neu	MRZ	Neu
09.05.2011	LabView_D1_2	LabView_D1_0	HIE	Finale Korrigenda