wireflow

AC0120-008, rev B

WF 2144
User Manual

Contents

AC0120-008AC0120-008, rev B

Support information

Technical support and Product information

www.wireflow.se

WireFlow headquarters

WireFlow AB
Theres Svenssons gata 10
SE-417 55 Göteborg

Please see appendix “Technical support and Professional services” for more information.

Important information

Copyright

High risk activities

The software and hardware is not designed, manufactured or intended for use or resale as on-line control equipment in hazardous environments requiring fail-safe performance, such as in (but not limited to) the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, direct life support machines, or weapons systems, in which the failure of the Software could lead directly to death, personal injury, or severe physical or environmental damage ("High Risk Activities"). WireFlow and its suppliers specifically disclaim any express or implied warranty of fitness for High Risk Activities.

Compliance

CE - European Union EMC and Safety Compliance CE

This product meets the essential requirements of applicable European Directives, as follows:

- 2014/35/EU; Low-Voltage Directive (safety)
- Electromagnetic Compatibility (EMC) Directive 2014/30/EU
- RoHS Directive 2015/863/EU

Please contact WireFlow to get a copy of the Declaration of Conformity for the WF 2144 module

Device information

Introduction

The WF 2144 from WireFlow is a 4-channel USB programmable resistor. It can emulate resistor values between 16ohm and 160kohm which make it suitable for simulating resistive sensors such as PT100 and others.

LabVIEW and python drivers are included as well as a documented Modbus API which enables PC, Mac and Linux users to use the device. Power and communication are transmitted over a single USB 2.0-port where the device acts as a Modbus slave on a virtual COM port.

The unit is designed for use in labs, test equipment and Hardware-In-the-Loop systems.

Features

- Four independent, galvanically isolated channels
- Entirely solid-state simulation
- High resolution with non-linear scaling
- Wide resistance range
- Enhanced accuracy mode
- On-board calibration memory
- Open Modbus API
- LabVIEW driver included
- Python driver included
- Combines permutations of real resistors to achieve desired value

Operation

The WF 2144 works by connecting up to 16 resistors in parallel to achieve the desired resistance. The exact permutation of resistors for each channel is calculated by an algorithm and depends on the calibration values for the individual unit.

Resistance setpoints for the four channels are set via USB individually or simultaneously. Resistance values are reset to infinite (open circuit) when the device is powered on or power-cycled.

Enhanced mode

It is possible to increase the accuracy by combining two channels into enhanced mode. The user writes the desired combined resistance to a separate register and the WF 2144 uses this to calculate and set the resistance on both channels.

The channels must be physically connected in series and the possible combinations are channel 0 & 1, and/or channel 2 & 3. Enhanced mode and normal mode can be used intermittent.

AC0120-008AC0120-008, rev B

Specifications

Resistor Characteristics		
Number of channels	4	
Max Voltage (over resistor terminals)	150 V	
Max Power (through resistor terminals)	100 mW/channel	
Range	16 Ω – 160 kΩ	
Max error in Normal mode	R < 100 Ω	0.03%
	R < 1 kΩ	0.1%
	R < 10 kΩ	1%
	R < 160 kΩ	10%
Max error in Enhanced mode	32 Ω < R < 160 kΩ	0,25%
Max update rate	200 Hz	

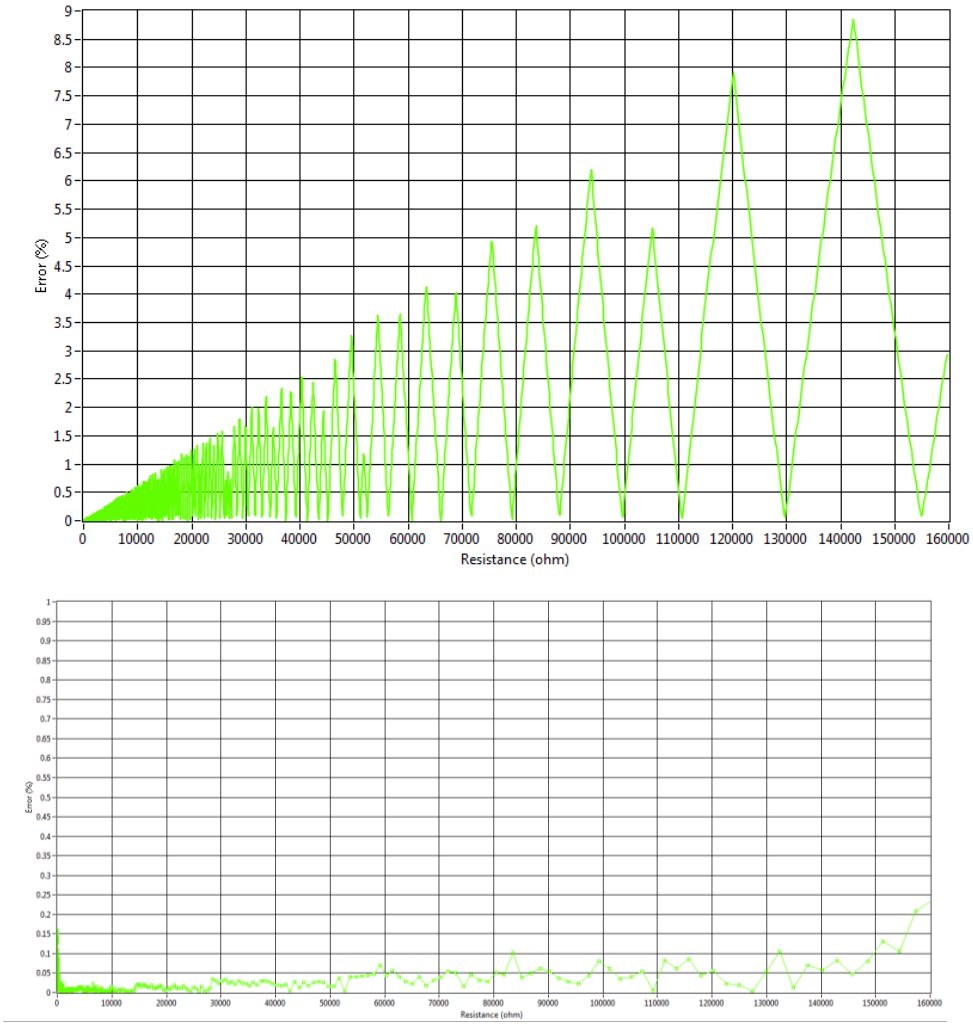
Isolation Voltages (rated working voltage)	
Channel-to-channel, continuous	320 V_{DC} (354V_{RMS})
Channel-to-earth ground, continuous	500 V_{DC} (354V_{RMS})

Power Requirements	
Current consumption (via USB)	5V / 300 mA

Environmental	
Operating temperature	0 °C to 70 °C
Storage temperature	-20 °C to 85 °C
Pollution	Degree 2
Maximum altitude	3000 m
Indoor / Outdoor	Indoor use only

Accuracy

Accuracy is better at lower resistance values and follows a logarithmic scale (see specifications).



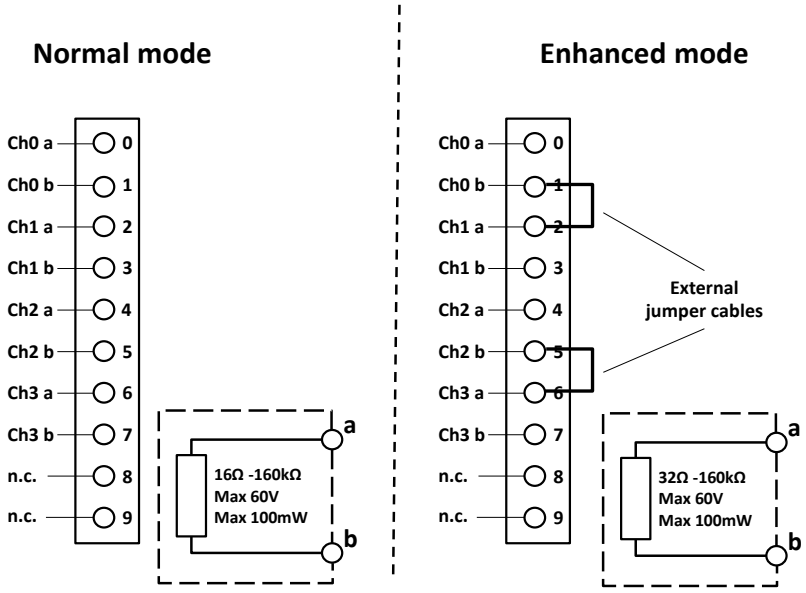
Users can estimate the accuracy for a given setpoint by reading out the actual resistance set by the device via USB (see *ReadResistanceOutput* in the software chapter). This value is the best permutation of the resistors that the device could find.

Calibration

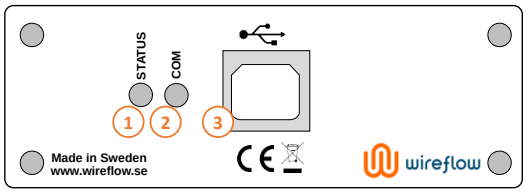
Each module is delivered calibrated. Modules that no longer stay within the Max Error range given in the specifications section can be re-calibrated by WireFlow.

For questions regarding calibration, please contact WireFlow's support, see chapter "Technical Support and Professional Services".

Pinout



External wiring and indicators



The device has the following interfaces

1) Status LED (green) - Solid when device is operational
2) Communication LED (orange) - Blinks during data transfer
3) USB Type B connector (USB 2.0)

Mounting

The unit is designed for desktop usage. The package includes rubber feet for mounting in either horizontal or vertical arrangement.

Software

The WF2144 is supplied with LabVIEW and Python drivers. They provide easy installation and integration into multiple programming environments. The drivers are used for setting resistor values and reading device information.

The device uses the Modbus RTU protocol for communication between host and devices. The registers are documented in this manual which allows the user to easily use the device on many platforms and programming languages.

Supported Platforms

	LabVIEW Driver	Python Driver	Modbus API
PC (Windows)	YES	YES	YES
PC (Linux)		YES	YES
Mac		YES	YES
NI cRIO / sbRIO	YES	YES	YES
NI PXI	YES	YES	YES
NI Industrial Controllers	YES	YES	YES
Other		YES	YES

Requirements

	LabVIEW Driver	Python Driver	Modbus API
Operating system	Win 7, 10 or LabVIEW RT	Any*	Any**
LabVIEW Version	2014+		
VIPM Version	2016+		
NI VISA	17.5+		
NI Modbus Library	1.1.5.39+		
Python Version		3.6.7+	
MinimalModbus		2.0.1+	
PySerial		3.0+	
USB host	2.0	2.0	2.0

* OS must support Python

** OS must support installing or programming a Modbus RTU driver using USB virtual COM ports (CDC class devices)

LabVIEW Driver

Installation

The LabVIEW driver is supplied as a *.vip package and requires the user to have VIPM(Vi Package Manager) installed, which can be obtained from www.ni.com.

To install the package, follow these steps:

1) Double click the *.vip package
2) Follow the instructions in VIPM to select LabVIEW version
3) Restart LabVIEW

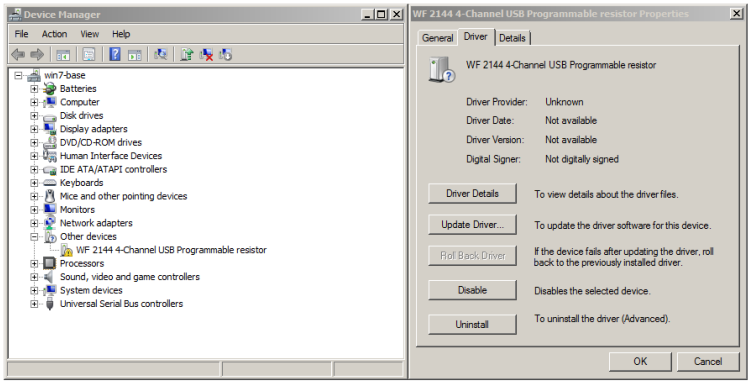
Win 7 computers need an additional *.inf file to be installed (not needed on win 10).

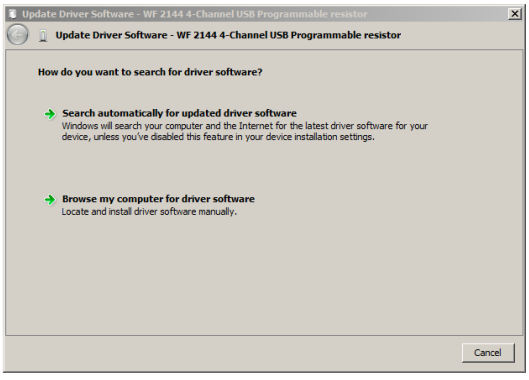
After the *.vip installation, a windows explorer window should open with at the location of the *.inf file. If not shown, the file can be located here:

C:\Program Files (x86)\National Instruments\<LabVIEW XXXX>\vi.lib\addons\WireFlow_WF USB Programmable Resistor LabVIEW driver\Windows device driver

To install the .inf-file follow these steps:

1) Plug in the device in a USB-port
2) Open Device Manager
3) Double-click the WF2144
4) Select Update driver... and browse to the directory of the *.inf-file





API

Init.vi

Initialize a connection to a USB programmable resistor via a virtual COM port.

Clear.vi

Clears the connection to a USB programmable resistor.

ReadCalibrationDate.vi

Reads the calibration date from the USB programmable resistor.
Date format is <year>-<month>-<day>.

ReadDeviceType.vi

Reads the device type of the USB programmable resistor.
1 = the device is a WF 2144

ReadFirmwareVersion.vi

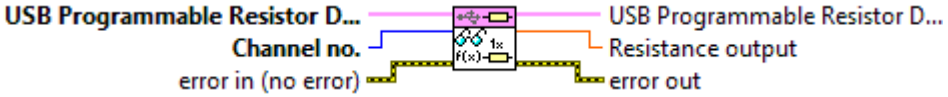
Reads the firmware version from the USB Programmable resistor.
Firmware format is <major>.<minor>.<fix>.<build>.

ReadResistanceOutput_FourChannels.vi

Reads the calculated output resistance for four channels.

ReadResistanceOutput_OneChannel.vi

Reads the calculated output resistance for one channel.

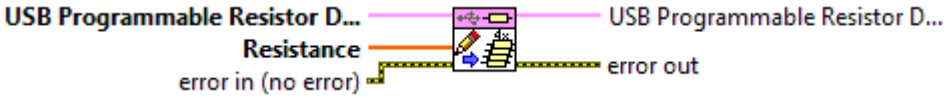


ReadSerialNumber.vi

Reads the serial number from the USB programmable resistor.

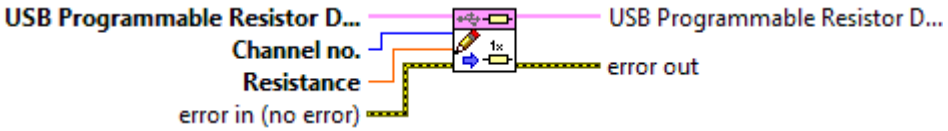
WriteResistance_FourChannels.vi

Sets the resistance values for all four channels at the same time. The input array must be four elements long.



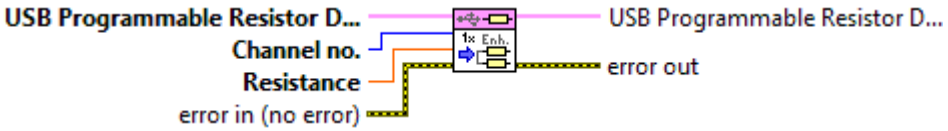
WriteResistance_OneChannel.vi

Sets the resistance for one channel.



WriteResistance_OneChannel_Enhanced.vi

Sets the resistance for two channels that are connected in series in enhanced mode. The total resistance will have a higher accuracy.



Examples

An example file can be opened from either the NI example finder, or from VIPM.

Python Driver

Installation

The Python driver is supplied as a zip file. Unzip it and follow the instructions in readme.txt.

API

The Python API function calls are the same as the LabVIEW implementation. Below is a list of the available functions.

```
def __init__(self, port):
def clear(self):
def get_serial_number(self):
def get_device_type(self):
def get_device_type(self):
def get_calibration_date(self):
def get_firmware_version(self):
def get_resistance_output_one_chan(self,channel):
def get_resistance_output_four_chan(self):
def set_resistance_one_chan(self,channel,value):
def set_resistance_one_chan_enhanced(self,channelgroup,value):
def set_resistance_four_chan(self,values):
```

Examples

An example file (`example.py`) is attached to the zip file.

Modbus API

The Modbus protocol is a request-response based protocol where a master device (typically a PC) sends a *request message* to a slave device (the WF 2144), which replies with a *response message*.

This way, the master can read or write data from/to the slave. Data is organized in 16-bit registers which are used for control, measurement and configuration of the slave.

The Modbus protocol specification is available for download at https://modbus.org/specs.php under *MODBUS protocol specification.*

The WF 2144 uses the Modbus RTU implementation which is the serial line version of the Modbus protocol. This implementation adds a *Unit ID* and *error check* to each message.

The Modbus over serial line specification is available for download at https://modbus.org/specs.php under *Modbus Serial Line Protocol and Implementation Guide V1.02.*

Unit ID

The WF 2144 uses a virtual COM port for communication, hence it will be the only slave on that COM port. The unit ID is therefore fixed and set to 1.

Serial Settings

The WF 2144 uses the following serial settings:

Parameter	Value
Baudrate	115200
Stop bits	1
Parity	None
Data bits	8

Function Codes

The WF 2144 support the following function codes:

Function	Code
Read Holding Registers	0x03
Read Input registers	0x04
Write Multiple Registers	0x10

Error Handling

The WF 2144 applies two extra requirements on Modbus requests in addition to the standards:

a) LSW and MSW registers must be written in the same request.
b) The combined value of LSW and MSW must be supported by the channel (16Ω -160kΩs for normal mode and 32Ω-160kΩ for enhanced mode).

A request failing to meet a) will result in exception code 0x02 (ILLEGAL DATA ADDRESS)

A request failing to meet b) will result in exception code 0x03 (ILLEGAL DATA VALUE)

Registers

Name	Address	Data Type	Description
Input registers			
Firmware version major	0	U16	
Firmware version minor	1	U16	
Firmware version fix	2	U16	
Firmware version build	3	U16	
Serial number	4	U16	
Calibration year	5	U16	
Calibration month	6	U16	
Calibration day	7	U16	
Device type	8	U16	1=4-channel model
Resistor 0 output MSW	9	Float.MSW	
Resistor 0 output LSW	10	Float.LSW	
Resistor 1 output MSW	11	Float.MSW	
Resistor 1 output LSW	12	Float.LSW	
Resistor 2 output MSW	13	Float.MSW	
Resistor 2 output LSW	14	Float.LSW	
Resistor 3 output MSW	15	Float.MSW	
Resistor 3 output LSW	16	Float.LSW	
Holding registers			
Resistor 0 MSW	0	Float.MSW	
Resistor 0 LSW	1	Float.LSW	
Resistor 1 MSW	2	Float.MSW	
Resistor 1 LSW	3	Float.LSW	
Resistor 2 MSW	4	Float.MSW	
Resistor 2 LSW	5	Float.LSW	
Resistor 3 MSW	6	Float.MSW	
Resistor 3 LSW	7	Float.LSW	
Resistor 0-1 MSW	8	Float.MSW	For enhanced mode
Resistor 0-1 LSW	9	Float.LSW	For enhanced mode
Resistor 2-3 MSW	10	Float.MSW	For enhanced mode
Resistor 2-3 LSW	11	Float.LSW	For enhanced mode

Example – Read input registers

Below is an example of the bytes transmitted on the serial bus when reading the Firmware version. The firmware version consists of four U16 registers that are read with the *read input registers function*.

The read back version in this example is 0.4.0.130.

Request message

Field Name	Value		
Slave Address	0x01	Modbus ADU	Slave ID
Function Code	0x04		Modbus PDU
Starting Address Hi	0x00		
Startinf Address Lo	0x00		
Quantity of Input Registers Hi	0x00		
Quantity of Input Registers Lo	0x04		
CRC Hi	0xF1		Error check
CRC Lo	0xC9		

Response message

Field Name	Value		
Slave Address	0x01	Modbus ADU	Slave ID
Function Code	0x04		Modbus PDU
Byte Count	0x08		
Firmware version build Hi	0x00		
Firmware version build Lo	0x00		
Firmware version fix Hi	0x00		
Firmware version fix Lo	0x04		
Firmware version minor Hi	0x00		
Firmware version minor Lo	0x00		
Firmware version major Hi	0x00		
Firmware version major Lo	0x82		
CRC Lo	0x55		Error check
CRC Hi	0xAC		

Example – Write multiple registers (holding)

Below is an example of the bytes transmitted on the serial bus when writing the resistance setpoint for channel 0. The resistance setpoint consists of two U16 registers that together make up a 32-bit IEEE 754 single precision float value. They are written with the *write multiple registers* function.

The written setpoint in this example is 100ohm.

Request message

Field Name	Value		
Slave Address	0x01	Modbus ADU	Slave ID
Function Code	0x10		Modbus PDU
Starting Address Hi	0x00		
Starting Address Lo	0x00		
Quantity of Input Registers Hi	0x00		
Quantity of Input Registers Lo	0x02		
Byte Count	0x04		
Resistor 0 LSW Hi	0x42		
Resistor 0 LSW Lo	0xC8		
Resistor 0 MSW Hi	0x00		
Resistor 0 MSW Lo	0x00		
CRC Hi	0x66		Error check
CRC Lo	0x29		

Response message

Field Name	Value		
Slave Address	0x01	Modbus ADU	Slave ID
Function Code	0x10		Modbus PDU
Starting Address Hi	0x00		
Starting Address Lo	0x00		
Quantity of Input Registers Hi	0x00		
Quantity of Input Registers Lo	0x02		
CRC Lo	0x41		Error check
CRC Hi	0xC8		

Troubleshooting

This chapter should list the most common problems that a user might encounter

Technical support and Professional services

WireFlow Product Take-Back and Recycle Program

In support of the global requirements to dispose of electrical waste within environmentally acceptable specifications, WireFlow offers customers the take-back and recycle process to properly dispose of surplus and end-of-life products.

Equipment that is returned to WireFlow through this program is disposed of in an environmentally safe manner using processes that comply with the WEEE (EU Directive on Waste Electrical and Electronic Equipment) regulations. All WireFlow-branded products are accepted under the program

Ordering information

Article no	Product	Description
AE0029	AC0120, WF 2144	4-chn USB programmable resistor