MOBI

MODBUS Interface

INSTRUCTION MANUAL

Revision 5.2

Table of contents

Introduction

The need to be able to interact (or at the minimum collect data) with a flow meter from a remote location is becoming a very important issue. The already available dialup modem is for very remote locations. Most end users have PC's which are equipped with different interfaces. There are flow meters with are equipped with an RS232 interface. When using one flow meter, there isn't a problem. The user connects the flow meter to the PC and has access to information from the meter. The only problem is the distance between the PC and the flow meter.

It becomes a different story when the user has many flow meters. The RS232 interface only supports point to point communication. One could equip the PC with multiple RS232 interfaces but that would be costly and it will require tons of cables (each flow meter each own cable). Beside the maximum distance for RS232 is 20 meters.

An option would be RS485. This type of interface enables users to hook up 247 devices in parallel (the maximum numbers of devices will depend on the line conditions). There are ready made converters available which are able to convert RS232 into RS485. Two problems emerge:

1. RS232/485 converters are costly
2. Sierra flow meters equipped with an RS232 interface only support point to point communication.

There are many protocols available which can handle multiple devices sharing the same bus. A widely used protocol is MODBUS which has proven itself in the field. Unfortunate the Sierra flow meters only support one protocol and can't be modified.

MOBI

Sierra has developed an interface which acts as a gateway between two protocols. The MOBI (**MO**d**B**us **I**nterface) translates the Sierra protocol to MODBUS and visa versa. All available data from the Sierra units is stored in holding registers and can be accessed through MODBUS. Some registers can be modified.

The interface will fit into the existing flow meters (both EN and E housing).

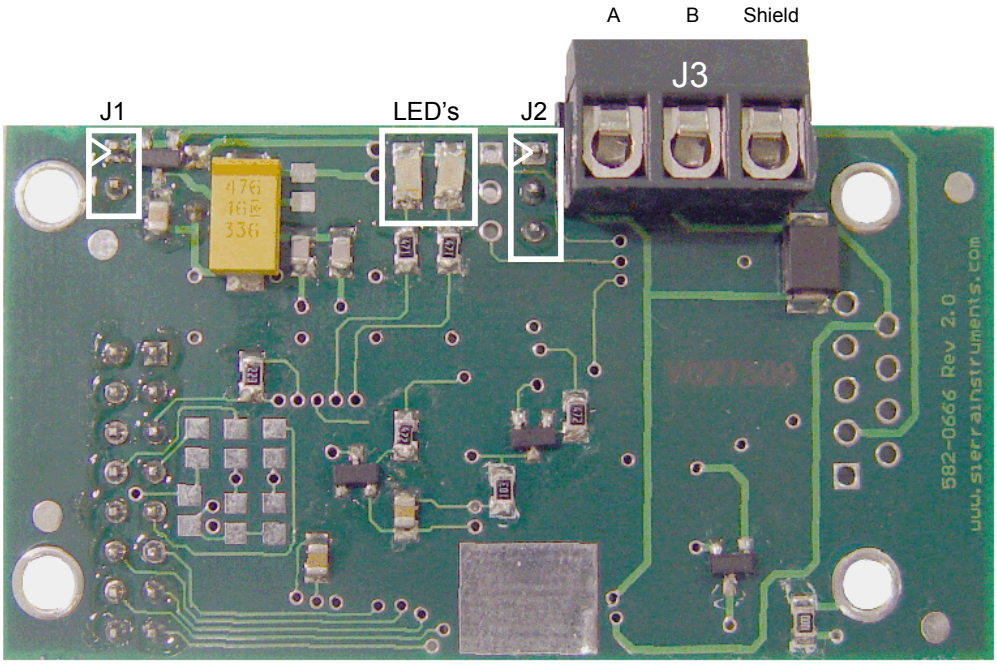
Chapter 1 - The interface board

MOBI is build around a small PCB measuring 35 x 66 mm. Communication with the MODBUS network is done through an optical isolated RS485 driver.

Two LED's show the activities of the interface:

Red LED	Green LED	State
Off	On	Interface powered an active
Flashing	On	Processing message
On	Off	Initializing
Slowly flashing	Off	Error has occurred

Board overview



(Bottom view)

J1 – Power supply

Pin	Function
1	Power in (8 ~ 30 VDC)
2	Ground

J2 – RS232 port

Pin	Function
1	Transmit (output)
2	Ground
3	Receive (input)

Standard RS232 interface which connects to the flow meter or PC (when using the set up tool).

J3 – Isolated RS485

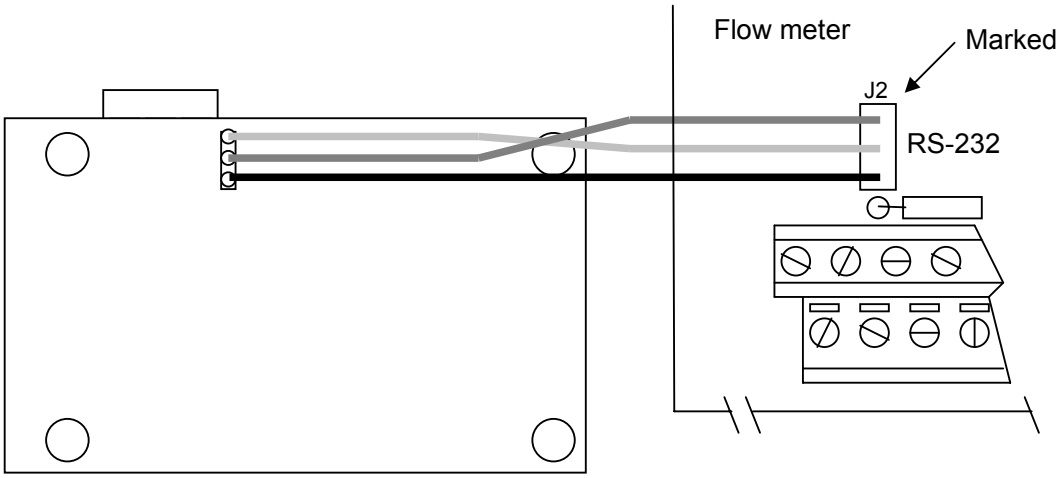
Pin	Function
1	Shield (Optional)
2	B - Inverting output / input
3	A - Non-inverting output / input

No external power is required for the RS485 interface. The shield can be connected to the ground/shielding of the network cable. Don't connect shield with ground when electrical isolation is required.

Connecting the board

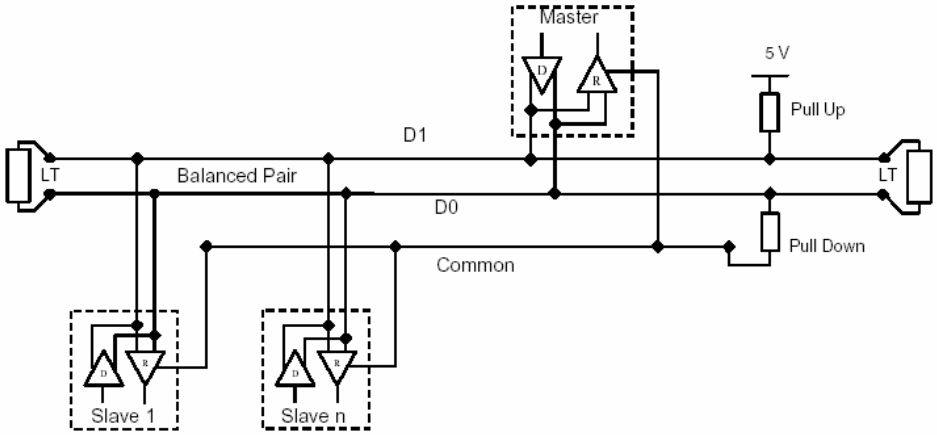
RS232 connection

The board is connected with the flow meter using a special cable. The diagram below shows how the board can be connected with a 640 flow meter:



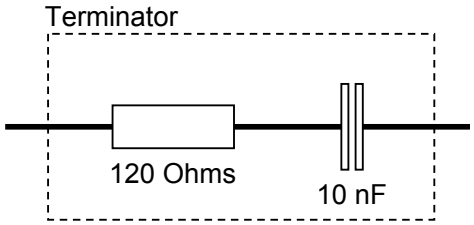
RS485 connection

The interface can only be connected to a 2 wire RS485 network as shown below:



D0 = A D1 = B Common = shield

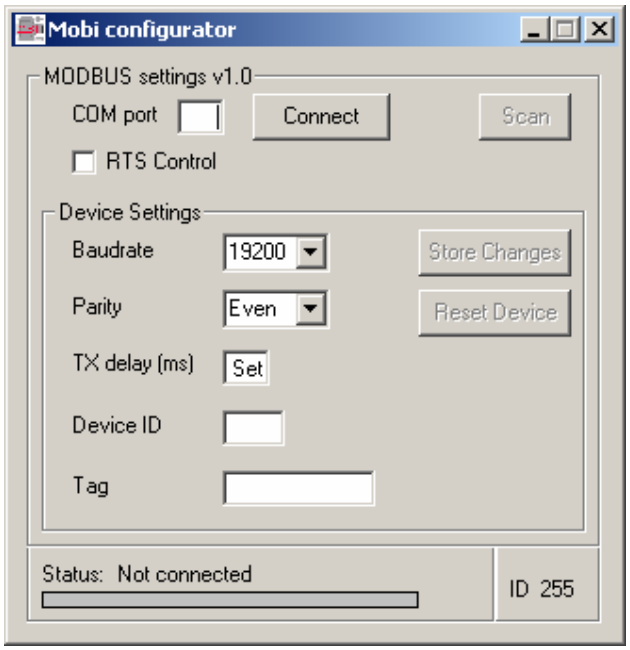
If the interface is the last device on the network then a terminator consisting of a serial capacitor (10 nF, 16 volts) with resistor (120R, 0.25W) has to be connected between terminal A and B.



Chapter 2 - Interface setup tool

The setup tool, “Mobi configurator.exe”, is used to configure MOBI. In order to use this tool the PC needs to be equipped with an RS485 interface or RS232 port with an external converter.

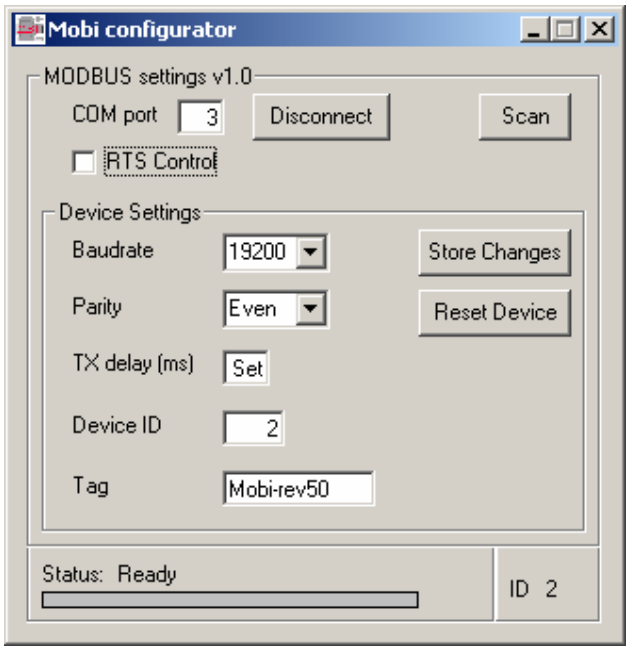
When the tool starts the following screen is presented:



Enter the COM port number to which the interface is connected and press the “Connect” button. If the COM port isn’t available a warning message will be displayed.

The program will first try to connect to device ID=255. If that fails a message will be shown:

If the device isn’t found then press the “scan” button. The program will search for the device and upon detecting it, it will retrieve the tag string.



Description of the buttons and fields

Disconnect

Pressing the “Disconnect” button will close the COM port and the start screen will be shown again.

Tag

Tag string assigned to the unit. The maximum numbers of characters is 10 and any character may be used.

ID

Identification number of the interface and used by the MODBUS system to address this device. This number has to be unique! The ID number can be changed to any value between 1 and 247.

Baud rate

Currently selected baud rate. There are two speeds available, 9600 & 19200. Select one which is appropriate for the network.

Parity

Currently selected parity mode. There are three settings available: none, odd & even. Select one which is appropriate for the network.

TX delay

Set the delay time between switching from receive mode to transmit mode. A value between 0 – 255 can be entered.

Scan

Press the scan button to search for a device with an unknown ID. The scan can be started and stopped at any time. The scan function is able to detect any MODBUS device on the network.

RTS

Check RTS when an RS232/RS485 converter is used which switches the transmitter on the RTS line

Store changes

Store the settings in the device. The following data will be stored:

- Baud rate
- Parity
- Delay time between receive and transmit
- Device ID
- Tag string

Reset Device

Restart the device to activate the changes. This function will reset any MODBUS device which supports function 0x08, sub function 0x01

Chapter 3 - MODBUS commands

The implemented commands are all according to the MODBUS protocol as described in document “MODBUS Application Protocol Specification V1.1” available from the MODBUS organization (www.modbus.org). The commands can be tested using software tools like MODBUS Poll from Wittecom (www.wittecom.com).

Implemented commands

The following commands are implemented:

Function	Sub function	Description
0x03	N/A	Read holding registers
0x06	N/A	Write single holding register
0x08		Diagnostics
	0x00	Return query data
	0x01	Restart communications option
	0x04	Force listen only mode
	0x0A	Clear counters
	0x0B	Return bus message count
	0x0C	Return bus communication error count
	0x0D	Return bus exception error count
	0x0E	Return slave message count
	0x0F	Return slave no response count
	0x10	Return slave NAK count
	0x11	Return slave busy count
	0x12	Return bus character overrun count
	0x14	Clear overrun counter and flag

Holding registers overview table

Register	Read	Write	Type	No. registers
40001	Actual flow - low word		32 bit float	2
40002	Actual flow - high word			
40003	Totalizer - low word		32 bits int	2
40004	Totalizer - high word			
40005	User full scale	User full scale	16 bits int	1
40006	Factory full scale		16 bits int	1
40007	K factor	K factor	16 bits int	1
40008	Dummy (reads $0001)	Reset totalizer	16 bits int	1
40009	Calibration - high word *		32 bits Int	2
40010	Calibration - low word *			
40011	Flow unit - char 1,2		String	3
40012	Flow unit - char 3,4			
40013	Flow unit - char 5,6			
40014	Totalizer unit- char 1,2		String	2
40015	Totalizer unit- char 3,4			
40016	Serial number – char 1,2		String	6
40017	Serial number – char 3,4			
40018	Serial number – char 5,6			
40019	Serial number – char 7,8			
40020	Serial number – char 9,10			
40021	Serial number – char 11,12			
40022	Tag number - char 1,2		String	5
40023	Tag number - char 3,4			
40024	Tag number - char 5,6			
40025	Tag number - char 7,8			
40026	Tag number - char 9,10			
40027	Decimal point – flow/totalizer		16 bits int	1
40028	Analog CH0 (10 bit res.)**		16 bits int	1
40029	Analog CH1 (10 bit res.)**		16 bits int	1

** Format = mmddyyyy (decimal)*

*** Only available in special cases*

Read holding register overview

Each register holds a specific type of data. Sometimes more registers are required to obtain the desired information.

40001 – 40002: Actual flow

The actual flow as displayed on the LCD of the unit (if available). The flow is IEEE-754 encoded.

Example: $44C34599 = 1562.175

40003 - 40004: Totalizer value

The totalizer value as displayed on the LCD of the unit (if available). The value isn't scaled and might need correction. Read register 40027 to determine the location of the decimal point or scale the value in the OPC/HMI software

Example: $293F0D = 2703117

Reading register 40027 returns $0002 ⇒ totalizer decimal point = $02 ⇒ #.##
The value of the totalizer becomes: 27031.17

40005: User full scale

The returned word contains the full scale of the unit as set by the user.

Returned: User full scale hexadecimal encoded

Example: $4E20 = 20000

40006: Factory full scale

Returned word contains the full scale of the unit as set by the manufacturer.

Returned: Factory full scale hexadecimal encoded

Example: $5DC0 = 24000

40007: K factor

The returned word contains the K factor of the used gas as set by the user.

Returned: K factor hexadecimal encoded

Example: $03E8= 1000 ⇒ the value needs to be divided by 1000 to get the correct factor ⇒ 1.000

40008: Dummy

Reading this address will return fixed data ($0001).

40009 – 40010: Calibration date

The returned data contains the calibration date of the unit.

Returned: calibration date hexadecimal encoded

Example: Reading 0x8D2CA3 which equals 9252003 in decimal. This equals to 9 25 2003 = September 25, 2003

40011 – 40013: Flow unit

Each register contains two characters of the flow unit. The returned word is encoded in ASCII.

Returned: Characters

Example: $534C ⇒ “SL”

40014 - 40015: Totalizer unit

Each register contains two characters of the totalizer unit. The returned word is encoded in ASCII.

Returned: Characters

Example: $534C ⇒ “SL”

40016 - 40021: Serial number

Each register contains two characters of the serial number. The returned word is encoded in ASCII.

Returned: Characters

Example: $5339 ⇒ “S9”

The serial number is 12 characters long. It always starts with “SN:”

40022 – 40026: Tag number

Each register contains two characters of the tag number. The returned word is encoded in ASCII.

Returned: Characters

Example: $5330 ⇒ “S0”

The tag number is set through the configuration software

40027: Decimal point of the flow/totalizer

Location of the decimal point in the actual flow/totalizer

Returned: 2 bytes (high byte = flow, low byte = totalizer)

Example: $0201 ⇒ decimal point flow = $02, decimal point totalizer = $01

Data	Point location	Divide by
0	00000000	0
1	0000000.0	10
2	000000.00	100
3	00000.000	1000
Etc.		

Write holding register overview

40005: User Full scale

The sent word contains the full scale of the unit as set by the user.

Example: Writing $5DC0 will set the scale to 24000

40007: K factor

Set the K factor of the unit.

Example: Writing $4B0 will set the K factor to 1.2 (1200)

40008: Reset totalizer

Reset the totalizer by writing the value $0001.

Note: Sometimes it needs two write attempts to get the value written to the unit