MOBI
MODBUS Interface

Instruction Manual

Part Number: IM-MOBI, Rev. A 3/08

COPORATE HEADQUARTERS
5 Harris Court, Building L, Monterey, CA 93940
PH: (831) 373-0200   (800) 866-0200 FX: (831) 373-4402
www.sierrainstruments.com

EUROPE HEADQUARTERS
Bijlmansweid 2, 1934RE Egmond aan den Hoef, The Netherlands
PH: +31 72 5071400 FX: +31 72 5071401

ASIA HEADQUARTERS
Room 618, Tomson Centre, Bldg. A
188 Zhang Yang Road
Pu Dong New District, Shanghai, P.R. China
PH: +8621 5879 8521/22 FX: /8621 5879 8586

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.

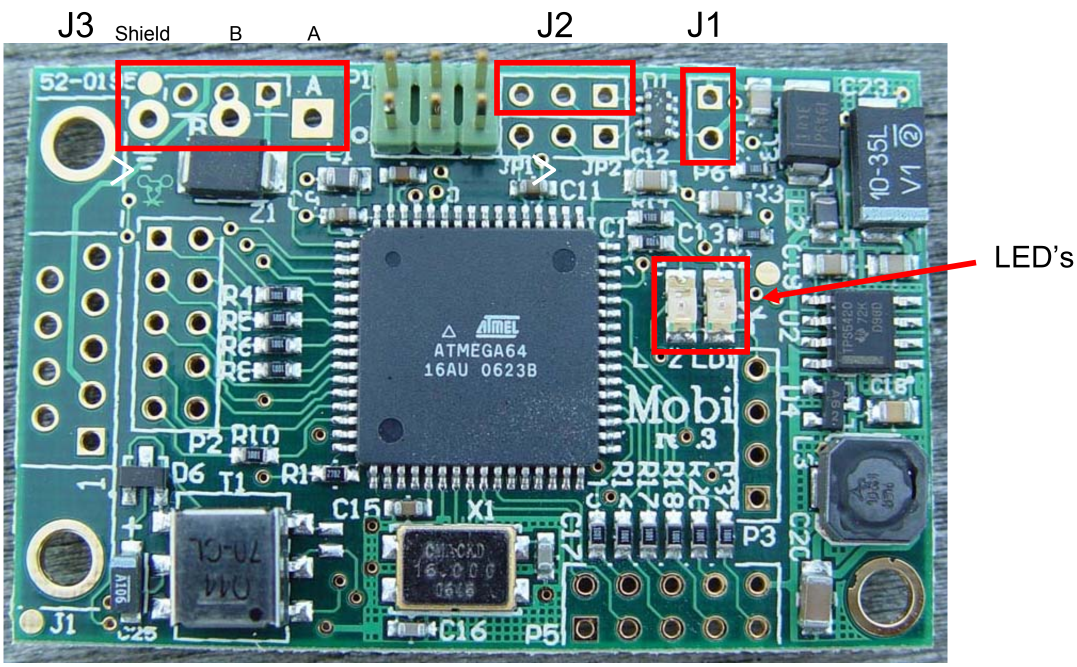
The interface will fit into the existing flow meters (both EN and E housing).

Chapter 1 - The Interface Board

MOBI is build around a small PCB measuring 35 x 55 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	Flashing	Processing data from the smart unit
Flashing	On	Processing MODBUS message
On	Off	Initializing
Slowly flashing	Off	Error has occurred

Board overview



(Top view)

J1 – Power supply

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

J2 – RS232 port

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

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

J3 – Isolated RS485

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

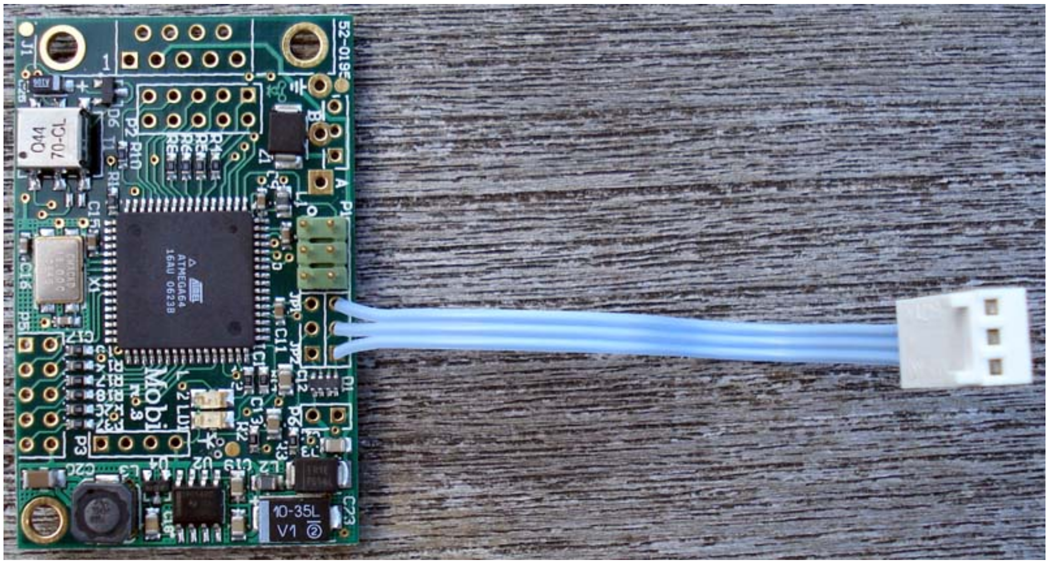
No external power is required for the RS485 interface. The shield can be connected to the ground/shielding of the network cable. Don't connect shield with ground when electrical isolation is required.

Note: Pin 1 on the PCA is identified by a square pad (☐)

Connecting the Board

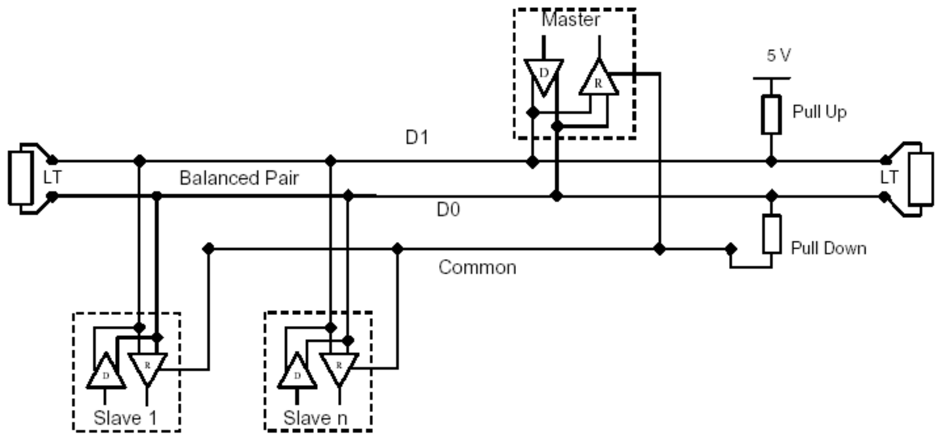
RS232 connection

The board is connected with the flow meter using a special cable (or any cable suitable for the application). The picture 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 has to be connected between terminal A and B. The terminator consists of a resistor with a value of 150 ohm (0.5W)

Chapter 2 - Interface Setup

Mobi can be setup using the boot loader. The boot loader can be accessed using a terminal program. In order to setup Mobi a PC needs to be equipped with an RS485 interface or an external converter connected to the RS232 port.

In this example 'HyperTerminal' from Microsoft will be used.

Start HyperTerminal from windows and select the com port to which the RS485 interface is connected. Use the following settings:

Baud rate:	9600
Number of bits:	8
Parity:	N
Stop bits:	1

The boot loader will only be active during the first 2 seconds after a power-up or reset. During power-up (or reset) the green LED will blink twice before executing the application. To enter the boot loader, follow the next steps:

1. Power up the unit
2. In HyperTerminal press the enter key within 2 seconds of power up (any other key will terminate the boot loader and will start the application)

The following menu should be presented on the screen:

```
Boot 1.0
1)Load Firmware
2)Setup Firmware
3)Quit

>
```

Press the '2' key to select the 'Setup firmware'. The following item is presented:

```
Boot 1.0
1)Load Firmware
2)Setup Firmware
3)Quit

>2
Firmware:v1.0
ID (1-247)=255
(C)hange (N)ext
```

The current version of the firmware is shown as well as the first setting. Press the 'C' (or 'c') button to change the ID code or press 'N' (or 'n') to skip.

Depending on the firmware, several settings are presented which can be changed. After the last settings the menu will be presented again.

Now press the '3' key (Quit) to run the application or perform a power cycle to start the application.

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
40030	Analog CH2 (10 bit res.)**		16 bits int	1
400341	Analog setup	Analog setup	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.		

40028 – 40030: Analog channel

Data from the analog input channels. The analog to digital converter has a resolution of 10 bits.

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

40031: Analog settings

The analog inputs can be configured to perform the following functions:

Low byte bits	Function	CH0	CH1	CH2
0	Input multiplier, 0 = 1x, 1 = 0.5x	●		
1	Input mode, 0 = voltage, 1 = current	●		
2	Input multiplier, 0 = 1x, 1 = 0.5x		●	
3	Input mode, 0 = voltage, 1 = current		●	
4	Input multiplier, 0 = 1x, 1 = 0.5x			●
5	Input mode, 0 = voltage, 1 = current			●
6	-	-	-	-
7	Reference, 0 = 2.56V, 1 = 5V	●	●	●

Note: maximum input range voltage: 5V