PROTEUS INSTRUMENTS

Proteus Integrated SDI-12 and MODBUS Output Operating Manual

V1.1 December 2021

Proteus Instruments Ltd, Canalside, Harris Business Park, Hanbury Road, Stoke Prior, Bromsgrove, B60 4DJ, United Kingdom
www.proteus-instruments.com | info@proteus-instruments.com | +44 1527 433221

Patented GB2553218 | Version 1.1

1) Introduction

In March of 2020, Proteus introduced integrated SDI-12 and RS-422 MODBUS data outputs for the Proteus water-quality multiprobe product line. This optional feature replaces the external converter units previously required for those output formats.
The new feature is called "Multi-Protocol Interface Board", or MIB. One adapter cable provides MODBUS output, and a different adapter cable provides SDI-12 output.
The MIB is normally built into the Proteus at the time of the unit's manufacture; it cannot be seen from outside the instrument, and does not alter the Proteus' size or appearance. The photo below shows a Proteus, Underwater Cable, and at the top of the photo, the short SDI-12 Adapter Cable. The SDI-12 master is attached to the three bare wires on one end of the Adapter Cable. The MODBUS Adapter Cable works the same way.

If you wish to retrofit a Proteus with the MIB option, please contact us. If you wish to build your own MODBUS or SDI-12 converter cable, or convert your Underwater Cable into a converter cable, use the wiring diagrams in Appendices One and Two.
Note that Proteus Data Cables (the short cables used for tasks such as calibration) will communicate with a PC or laptop, but do not support MODBUS or SDI-12 operation.

MIB-equipped Proteus can communicate with a PC and other RS-232 devices as usual (this is the "transparent" mode). Most MIB-equipped Proteus' can be operated with USB power as usual. However, some of the larger Proteus' and/or longer cables (>20m cable or P35/P40 with >250mA consumption - Use Battery life Calculator to check) may require the USB Converter's 12V adapter to communicate with that Proteus. The photo at right shows a "wall wart" that provides 12 volts to the USB Adapter that connects an Underwater Cable or Data Cable to the USB port on a PC or laptop. You can use a 12-volt battery as well.
Note also that early Proteus Data Cables will work only in transparent mode; the Modbus and SDI-12 features work only with an Underwater Cable.

2) MODBUS Operation

a) How It Works

For MODBUS communication, simply connect the MODBUS Adapter Cable to the nine-pin connector on the Underwater Cable attached to your MIB-equipped Proteus. The Adapter Cable gives you the wires you need to connect the Proteus to a MODBUS device, and is wired to inform the Proteus that you wish to communicate in the MODBUS format instead of the usual RS-232.
Note that the MODBUS interface can use either half-duplex differential RS-485 or full-duplex RS-232 (separate transmit and receive).

b) The MODBUS Technical Details

A MIB-equipped Proteus uses MODBUS protocol over RS-485 or RS-232 to read the parameters processed by the unit. The upstream communication functions either as a full-duplex RS-232 standard interface, or as a half-duplex, RS-485 standard interface. Data format is 8-bits with no parity, one stop bit. Baud rate is 19,200.
The MODBUS interface provides measurement values, beginning at holding register 40001 (see Table 1), for all enabled Proteus parameters, with each measurement value occupying 2 MODBUS registers. Values are formatted in IEEE-754 32-bit floating point representation.

Table 1: Mapping MODBUS Parameter Measurement Values

MODBUS Holding Register	Bus Address	Read Value	Format
40001	0	Parameter 1 MSW	IEEE32
40002	1	Parameter 1 MSW	
40003	2	Parameter 2 MSW	
40004	3	Parameter 1 MSW	IEEE32
..	..		
40035	34	Parameter 18 MSW	IEEE32
40036	35	Parameter 1 MSW	

In RS-485 operation, two communication lines to the MIB are used for the differential lines Data+ and Data- (see Appendix 1). In RS-232 operation, the Data+ line is connected to the Proteus Rx line and the Data- line is connected to Proteus Tx line. A negative voltage on the Proteus Rx signals the MIB that the Tx line from an RS-232 host is connected, so that the MIB operates in MODBUS/RS-232 format; otherwise, RS-485 format is assumed. The data format is 8-bits with no parity and one stop bit. The MIB normally operates at 19,200 baud. If you wish to change that rate, please see Table 2.

The MIB provides store/update read-only registers and read/write settings for communicating with common SCADA systems, PLC interfaces, or other data collection platforms. A built-in MODBUS map provides aggregated sensor readings and other equipment information.
The MODBUS interface provides measurement values, beginning at holding register 40001 (see Table 3), for all enabled Proteus parameters, with each measurement value occupying two MODBUS registers. Values are formatted in IEEE-754 32-bit floating-point representation.
The MIB's address is register-programmable (the default value is 1).
The MIB will always respond to MODBUS address 0 if you don't know the real address.

Table 2: MODBUS Baud Rate Indicies

Index	BaudRate
0	9600
1	19200 (default)
2	38400
3	57600
4	115200

Table 3: MODBUS Control Register Mapping			
MODBUS Holding Register	Bus Address	Read/Write Value	Format
40201	200	Baud Rate - Upstream	Fixed at 19, 200 baud
40202	201	MODBUS Device Address	Integer 1-250 Default=1
40203	202	Baud Rate- Downstream	Integer index, 0-4
40204	203	SDI-12 Address	Integer 0-9, A-Z, a-z
40205	204	Power Switch Delay	Integer 0-60
40206	205	Proteus Wipe Interval	Integer 0-1440 (minutes)
40207	206	Proteus Wipe Freeze Time	Integer 0-60 (seconds)

c) MIB Commands for MODBUS

A MIB-equipped Proteus can be connected to a host PC or laptop to send commands directly to the Proteus CPU, as well as special commands (see Table 4) to the MIB itself. This mode of communication - using the Proteus' normal RS-232 output and not MODBUS - is called the "transparent mode".
When a terminal emulator, such as TeraTerm or Hyperterminal, is used to talk to the Proteus in this transparent mode, the MIB recognizes and responds to certain ASCII commands to allow the pro-gramming/verifying of some parameters, as shown below. The format of these MIB command is "$ccxxx<cr>", where:

'$' indicates a MIB command
cc is a two-character MIB command identifier
xxx is a parameter values specific to the command

Table 4: Special MIB Commands			
Command	Description	Parameters	Response
$AMxxx<cr>	Set MODBUS Address	xxx ; 001 to 250	OK<cr>
$AM?<cr>	Read MODBUS Address	none; default= 1	xxx<cr> ; 001 to 250
$WPxxxx<cr>	Write Proteus wiper interval	xxxx ; 0000 to 1440 minutes, default= 0	OK<cr>
$WP?<cr>	Read Proteus wiper interval	none	xxxx<cr> ; 0000 to 1440 minutes
$WFxx<cr>	Write wipe data freeze time	xx ; 0 to 60 seconds, default=15	OK<cr>
$WF?<cr>	Read Proteus wipe data freeze time	None	xx<cr> ; 0 to 60 seconds
$FV?<cr>	Read IB firmware revision	None	IB Firmware revision

d) MODBUS Automatic Wiper Operation

Some Proteus models include a sensor-cleaning wiper built into the turbidity sensor. The wiper clears debris, foulants, and bubbles from the sensors' active faces when the Proteus is first powered-up, and when a WIPE command is sent to the Proteus. If your Proteus is continuously powered during MODBUS operation, you may wish to periodically initiate wipe cycles using MIB commands (see Table 4). The Wipe Interval is the number of minutes between wipe cycles. Note that setting the Wipe Interval to 0 disables automatic wiping.
Some parameter values are invalid during normal wiper cycles because of the movement of the wiper. When the cycle ends, the data resume their real-time format. But if your MODBUS controller may create an alarm on account of the invalid data during the wipe cycle, you can use MIB WIPE commands (see Table 4) to "freeze" all sensor data while the wiper is cycling. That means that all data coming from the Proteus during the wiper cycle is the same data sent in the last data transmission before the wipe cycle started, i.e. the readings stay the same during the wiper cycle. This programmable freeze time sets the number of seconds (default is 15 seconds) that the data are frozen after the Proteus is given a WIPE command. The data resume their real-time format after that number of seconds has passed.

3) Using the MIB for SDI-12 Communication

a) How It Works

For SDI-12 communication, simply connect the SDI-12 Adapter Cable to the nine-pin connector on the Data Cable or Underwater Cable attached to your MIB-equipped Proteus. The Adapter Cable gives you the wires you need to connect the Proteus to an SDI-12 device, and is wired to inform the Proteus that you wish to communicate in the SDI-12 format instead of the usual RS-232 format (i.e. transparent mode).
Appendix One shows the wire assignments by color.

b) The SDI-12 Technical Details

The host computer-to-Proteus communication adheres to requirements of the SDI-12 Support Group, Version 1.3. Table 5 summarizes the implemented SDI-12 commands. If you are not familiar with the SDI-12 Protocol, the SDI-12 Support Group website (**www.sdi-12.org**) provides more detail.

Table 5: MIB SDI-12 Commands (a = SDI-12 address)

a!	Empty Command	
aA!	Change address	
aC!	Request a concurrent measurement	returns up to 20 values
aM!	Request a measurement	returns up to 9 values
aM1!	Request an additional measurement	returns up to 9 additional values
aM2!	Request an additional measurement	returns up to 2 additional values
aCC!	Request a concurrent measurement with CRC	
aMC!	Request a measurement with CRC	
aDn!	Read measurement results data	n=0..2
aI!	Request device identification string	

c) Special MIB Commands for SDI-12

A MIB-equipped Proteus can be connected to a host PC or laptop to send commands directly to the Proteus CPU, as well as special commands to the MIB itself. Using the Proteus' normal RS-232 output and not SDI-12 output is called the "transparent mode".
When a terminal emulator, such as TeraTerm or Hyperterminal, is used to talk to the Proteus in this transparent mode, the MIB recognizes and responds to certain ASCII commands (see Table 6) to allow the programming/verifying some parameters, as shown below. The format of these MIB command is "$ccxxx<cr>", where:

'$' indicates an MIB command
cc is a two-character MIB command identifier
xxx is a parameter values specific to the command

Table 6: MIB Transparent-Mode Commands

Command	Description	Parameter(s)	Response
$ASx<cr>	Set SDI-12 Address	x= 0-9, A-Z, a-z; default= 0	OK<cr>
$AS?<cr>	Read SDI-12 Address	None	x<cr> ; x= 0 to 9, A-Z, and a-z
$PDxx<cr>	Set power-off delay (extend Proteus+ power ON-time from the last measure command)	xxx=) to 60 seconds; default= 30 seconds	OK<cr>
$PD?<cr>	Read power-off delay	None	xxx<cr> ; x= 0 to 60 seconds
$FV?<cr>	Read IB-firmware revision	None	IB Fireware revision

Table 7 shows example SDI-12 commands and responses for a Proteus for which 10 parameters have been selected for SDI-12 monitoring.

Table 7: Sample SDI-12 Commands and Responses for a Proteus with 10 Parameters Selected

Command	Response
0!	0<CR><LF>
0I!	013 PROTEUS 711SN10162469<CR><LF>
0V!	00000<CF><LF>
0M!	00169<CR><LF>
0D0!	0+0+408.6999+4938.999+489.3999<CR><LF>
0D1!	0+4494.399+132.6000+3651.699+131.2000<CR><LF>
0D2!	0+2269.900<CR><LF>
0M1!	00031<CF><LF>
0D0!	0+11.70000<CR><LF>
0C!	000310<CR><LF>
0D0!	0+0+1.800000+2.100000+489.6999<CR><LF>
0D1!	0+4523.299+133.1000+3591.099+132.2000<CR><LF>
0D2!	0+2243.600+11.72000<CR><LF>
0MC!	00039<CR><LF>
0D0!	0+0+1.900000+2.100000+488.999AD<CR><LF>
0D1!	0+4538.699+133.0000+3557.699+132.4000@Zy<CR><LF>
0D2!	0+2224.000NWS<CR><LF>
0MC1!	00031<CR><LF>
0D0!	0+11.68000BS_<CR><LF>
0CC!	000310<CR><LF>
0D0!	0+0+1.900000+2.000000+489.0999EHG<CR><LF>
0D1!	0+4546.699+133.100.3540.199+132.6000O]X<CR><LF>
0D2!	0+2214.500+11.70000CSh<CR><LF>

<CR> deontes an ASCII carriage return; <LF>denotes an ASCII line feed

In the return string of the "0I!" command, "13" is the SDI-12 Version number (1.3), "711" is the Proteus CPU Firmware version (7.11), and the string following "SN", "10162469" is the Proteus Serial Number.

Appendix 1 - MODBUS and SDI-12 Adapter Cable Wiring Assignments

MODBUS

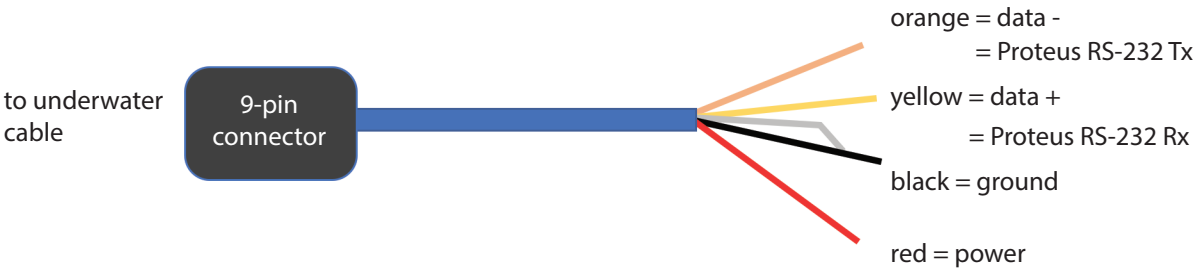
pin	colour
1	white
2	orange
3	yellow
4	n/c
5	black
6	n/c
7	n/c
8	blue
9	red

SDI-12

pin	colour
1	white
2	orange
3	yellow
4	n/c
5	black
6	n/c
7	n/c
8	blue
9	red

Appendix Two - Making Your Own MODBUS and SDI-12 Adapter Cables

MODBUS - connect the white wire to the black wire



SDI-12 - connect the orange wire to the yellow wire