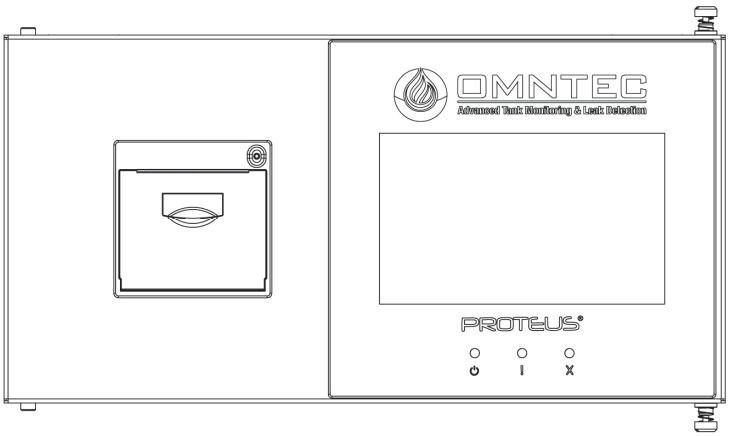
OMNTEC
Advanced Tank Monitoring & Leak Detection

1. Open the camera app
2. Focus the camera on the QR code by gently tapping the code
3. Follow the instructions on the screen to view PDF file

PROTEUS® B/K/X

MODBUS™ COMMUNICATION MANUAL



PROTEUS® Series
TANK GAUGING SYSTEM

Revision 1.3

Document No. DOC00005
Revised to include Gen IV PROTEUS-K and PROTEUS-X

OMNTEC Mfg., Inc. has been certified by DQS Inc. to ISO 9001:2015

OMNTEC® Mfg., Inc. | 2420 Pond Road | Ronkonkoma | NY 11779 | (631) 981-2001 | OMNTEC@OMNTEC.com | www.OMNTEC.com

PROTEUS-X AND PROTEUS-K4/K8 MODBUS COMMUNICATIONS

FIRMWARE RELEASE 2 AND HIGHER (RELEASE 5 FOR MODBUS TCP/IP). SEE NOTE 1 (PAGE 3).

PROTEUS-B4/B8 COMMUNICATIONS

FIRMWARE RELEASE 8 AND HIGHER. SEE NOTE 1 (PAGE 3).

TABLE OF CONTENTS

INDEX OF TABLES

NOTE 1

PROTEUS FIRMWARE RELEASES VERSIONS NEEDED FOR MODBUS:

THE PROTEUS-X/K REQUIRES:

- RELEASE 2 (MCU= E2AC, XB-416=E2AH, AND DSP=E2N VERSIONS) FOR MODBUS RTU (RS-232 AND RS-485) FUNCTIONALITY.
- RELEASE 5 (MCU= E2AF, XB-416=E2AK, AND DSP= E2P VERSIONS) FOR MODBUS TCP/IP FUNCTIONALITY.
- RELEASE 8 (MCU= E2AI, XB-416=E2AN, AND DSP= E2S VERSIONS) FOR B UNIT MODBUS RTU (RS-232) FUNCTIONALITY.
    - MODBUS ADDRESS MAPPING WILL BE DIFFERENT FOR SYSTEMS WITH XB-800 (8-CHANNEL PROBE CARD).

DOCUMENT RELEASES VERSIONS:

1514	UPDATED DOCUMENT FOR INITIAL RELEASE MARCH 31, 2015. .....(BY SLS)
1523	TABLE 2 CHANGED PRODUCT VOLUME TO GROSS VOLUME AND PAGE 10 EXAMPLE. .....(BY SLS)
1617	ADDED REVERSE MODE EXAMPLE ON PAGE 11. .....(BY SLS)
1727	ADDED MODBUS MAP PRINOUT. MODBUS EXCEPTION CODE CLARIFICATIONS.....(BY SLS) XB-416 AND XB-800 PROTEUS-X TABLES AND EXAMPLES. B4/B8 UNITS HAVE MODBUS.
1905	MODIFIED FOR PROTEUS-B, PROTEUS-K, AND PROTEUS-X UNITS IN THE SAME DOCUMENT.
1.2	UPDATE TO INCLUDE GEN 4.0 UNITS. .....(BY BSS)

GENERAL INFORMATION

1. All OEL8000III models (also known as PROTEUS®) support MODBUS. Only PROTEUS-K and PROTEUS-X models support MODBUS via TCP and RTU over Ethernet, as well as RS-232 and RS-485. The PROTEUS-B model supports MODBUS via RS-232 (naturally), and RS-485 via RS-232/485 converter.

2. MODBUS is an optional feature that requires activation with an optional enable code provided by OMNTEC® Mfg., Inc.

3. Set up data format:
   - **DATA BITS** is 8.
   - **PARITY** is None.
   - **STOP BITS** is 1.
   - **BAUD RATE** is 9600 (default).
   - Port 502 is the default port for **MODBUS TCP/IP**.

4. The minimum number of registers per **Read** command is 1.
   The maximum number of registers per **Read** command is 124.

5. The selectable **MODBUS ADDRESS RANGE** is 2 to 99. (default address is 2)

6. The **MODBUS HOLDING REGISTER OFFSET** settings are 0 or 40001 (default offset is 0).

7. Some data types take two MODBUS register places. The first register holds the high bytes, the second register holds the low bytes. (The floating-point data and unsigned long-data types use 2 registers each).

8. When connected to a PLC device, set up the **PLC DEVICE** to the **MASTER**.

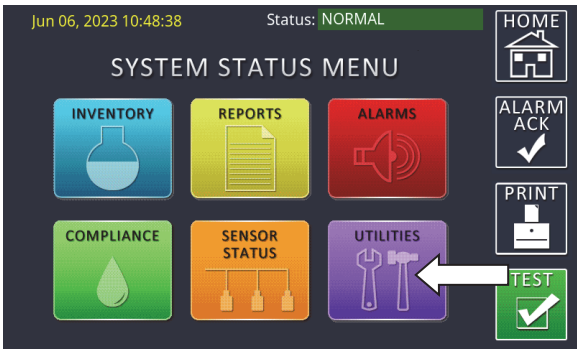
9. The **READ HOLDING REGISTER ACTIVE RANGE** is 1024 to 8188, if the **HOLDING REGISTER OFFSET** is set to 0 (default).

   Note: The tank registers are mapped by the probe board type (4- or 8-channel) and the slot location in a PROTEUS-X, The tank register numbers will not be consecutive for all tank numbers. See the explanation on page 12. The MODBUS registers for each system can be printed or viewed remotely.

PROTEUS-K/X PROGRAMMING TO CONFIGURE MODBUS

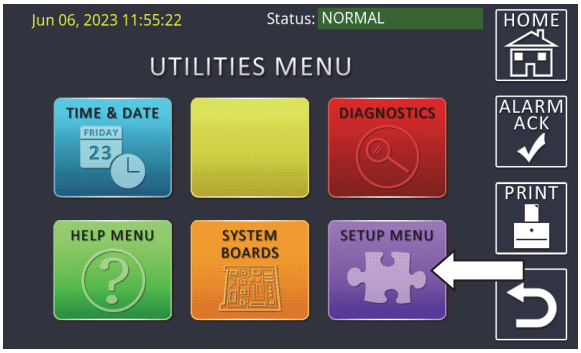
Note: Screen images and steps below are Gen IV and may vary slightly based on software versions.

1



Press the UTILITIES icon.

2

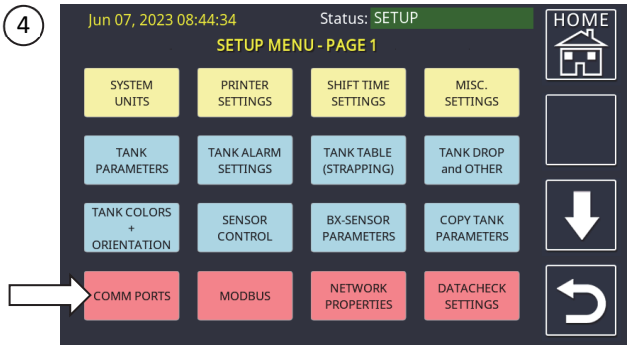


Press the SETUP MENU icon.

3

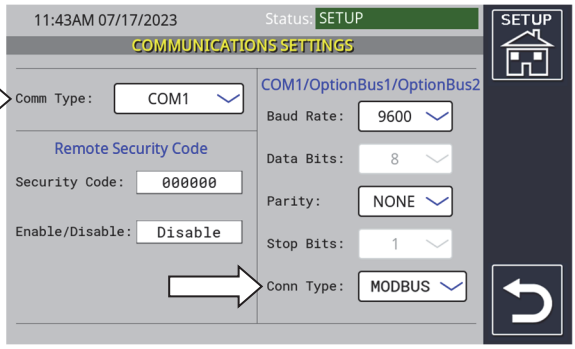
Press **000000** (six zeros) > Press ENTER.

4



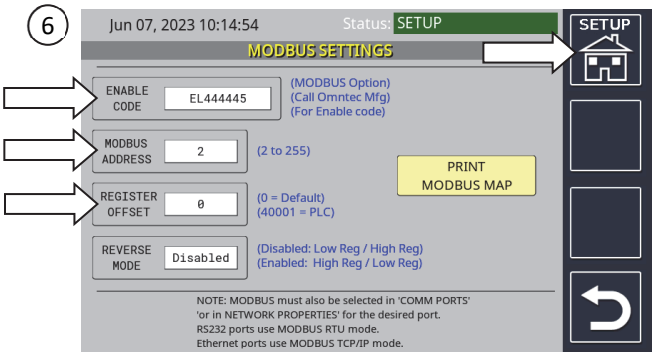
Press the COMM PORTS button.

5



- For RS-232 or RS-485 settings, match the master device.
- COMM TYPE: COM1 (RS-232), OPTION BUS 2 (RS-485).
- Set CONN TYPE to MODBUS.
- If MODBUS TCP/IP is used, see image 6 (below), then 5 or 7 depending on serial or IP connection.

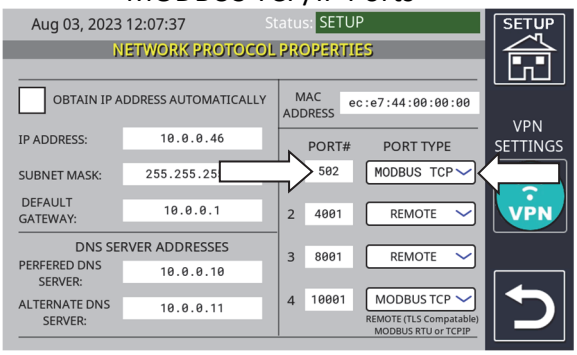
6



- Set the MODBUS ENABLE CODE (required for the MODBUS feature to work). If the optional MODBUS enable code is not set, contact OMNTEC Mfg., Inc.
- Set the MODBUS ADDRESS to a unique identifier for this device.
- Set the REGISTER OFFSET. This value is added to the base address calculated later in this document. 40001 allows the communication to work in the 40001-49999 Holding Register range required by many PLC's.
- Press the SETUP icon, then HOME, to exit SETUP mode.

MODBUS TCP/IP Ports

7



- Port# 502 is the default Port# used for MODBUS. Port numbers other than 502 can be used.
- Set Port Type to MODBUS TCP.

MODBUS WIRING/CONNECTORS

RS-232 Interface DB9 Cable Pinout (Figures 1.0 and 2.0):

- Figure 1.0 and Figure 2.0 show the DB9 connectors from the front.
- Figure 1.0 shows the wiring from a PROTEUS male DB9 connector to a female DB9 connector.
- Figure 2.0 shows the wiring from a PROTEUS male DB9 connector to male DB9 connector.

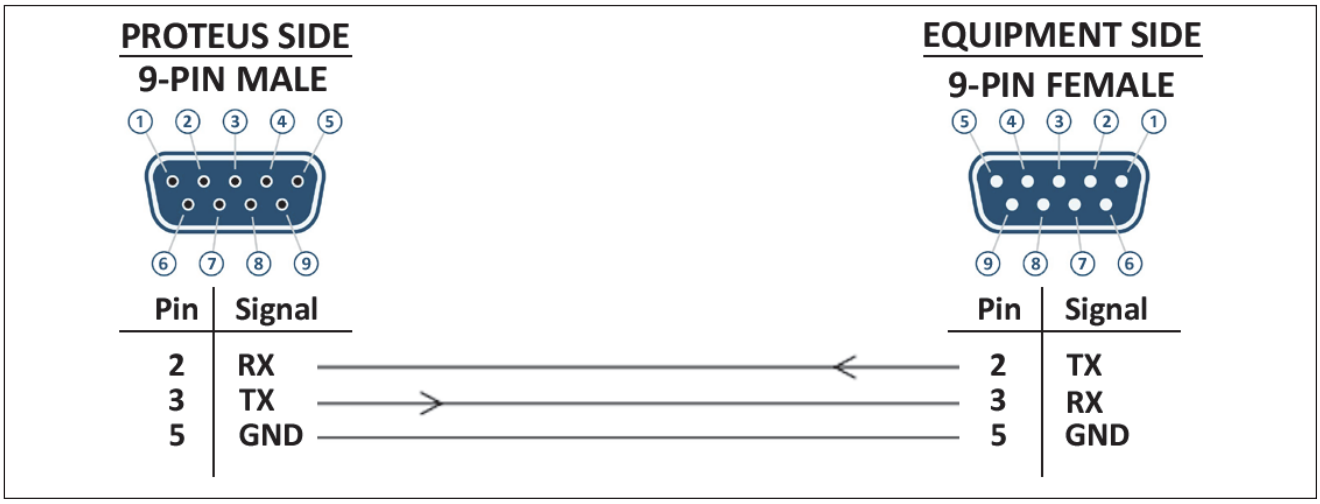


Figure 1.0

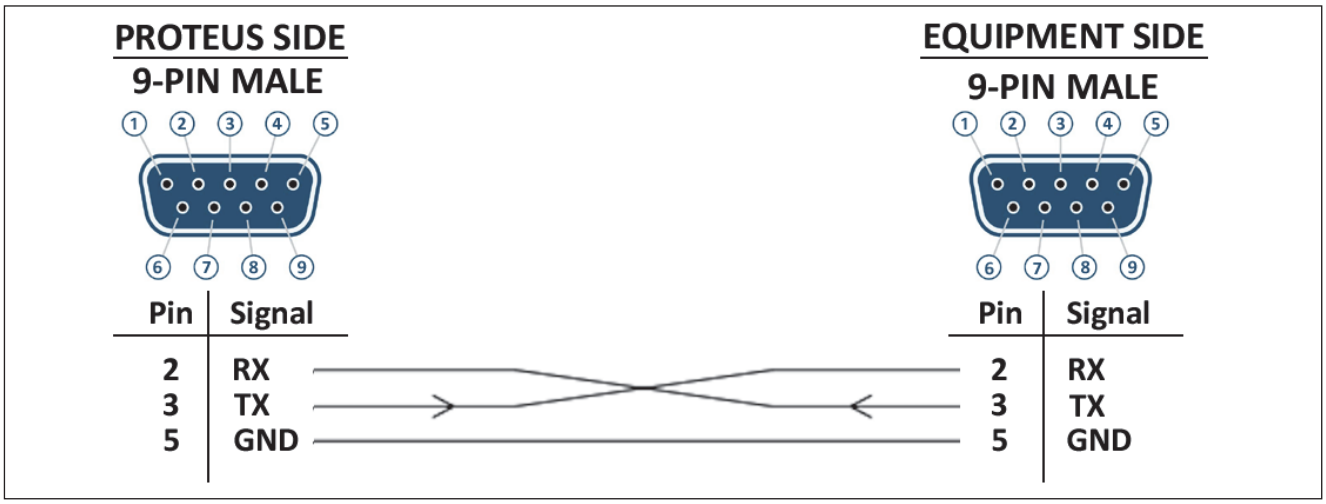


Figure 2.0

RS-485 Interface Pinout:

RS-485 TERMINAL PINOUT	
1	GND
2	NEG (-)
3	POS (+)
4	N.C.

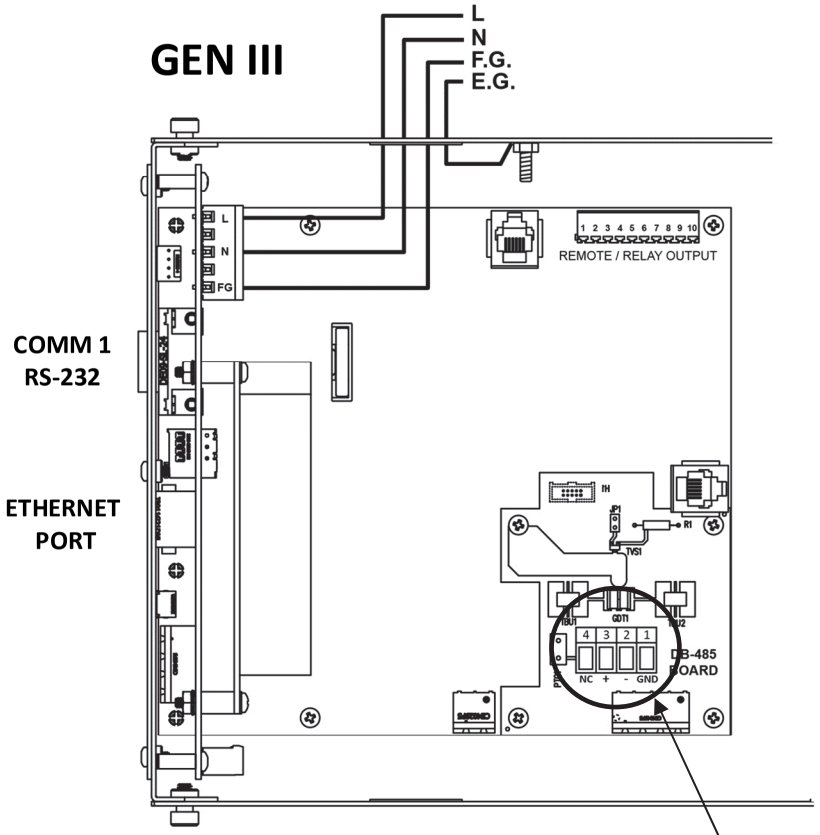
See the next pages for OEL8000IIIK and OEL8000IIIX unit drawings showing RS-485 connections for both Gen III and Gen IV versions.

ETHERNET Pinout (PROTEUS-K and PROTEUS-X):

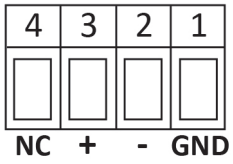
Standard RJ-45 Connector

Port 502 is the default port for MODBUS TCP/IP

PROTEUS OEL8000IIIK WIRING DIAGRAM

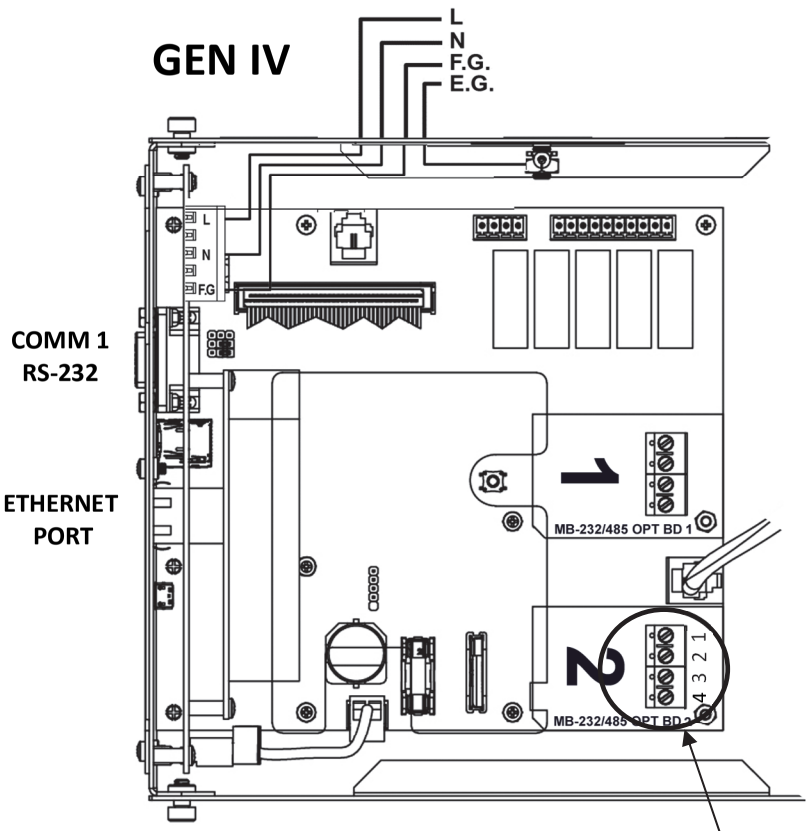


RS-485 Connector Front View

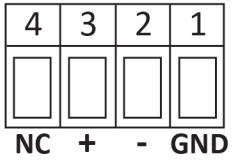


OEL8000IIIK DB-485 WIRING	
1	BLACK (GND)
2	RED (-)
3	WHITE (+)
4	UNUSED

For Gen III MCU firmware versions E2AD or higher.



RS-232/485 Connector Front View

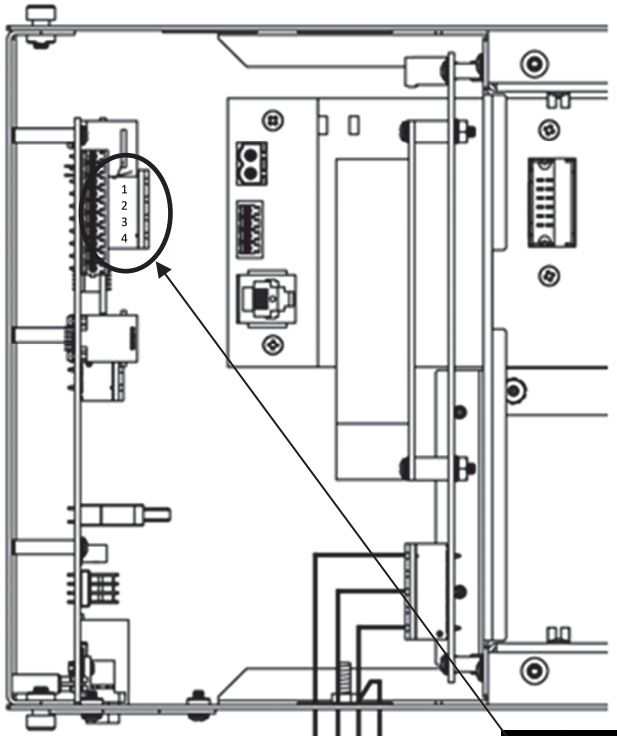


OEL8000IIIK MB-232/485 WIRING	
1	BLACK (GND)
2	RED (-)
3	WHITE (+)
4	UNUSED

Refer to Document DI00012 for more details on MB-232/485 wiring.

PROTEUS OEL8000IIIX WIRING DIAGRAM

GEN III



COMM 1
RS-232
+
ETHERNET
PORT

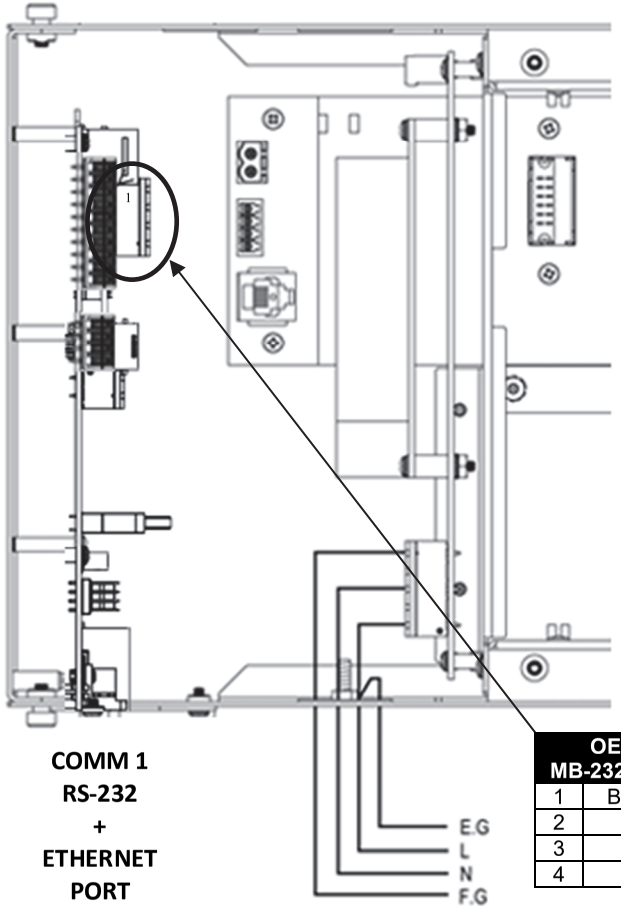
OEL8000IIIX DB-485 WIRING	
1	BLACK (GND)
2	RED (-)
3	WHITE (+)
4	UNUSED

RS-485 Connector Front View

4	3	2	1
NC	+	-	GND

For Gen III MCU firmware versions E2AD or higher.

GEN IV



COMM 1
RS-232
+
ETHERNET
PORT

OEL8000IIIX MB-232/485 WIRING	
1	BLACK (GND)
2	RED (-)
3	WHITE (+)
4	UNUSED

RS-232/485 Connector Front View

4	3	2	1
NC	+	-	GND

Refer to Document DI00012 for more details on MB-232/485 wiring.

MODBUS POLL COMMAND EXAMPLE READING TANK #1 AND SENSOR #1 DATA

Start Address for Tank#1 is 3960. Start Address for Sensor #1 is 4024 (MODBUS offset is set to 0).

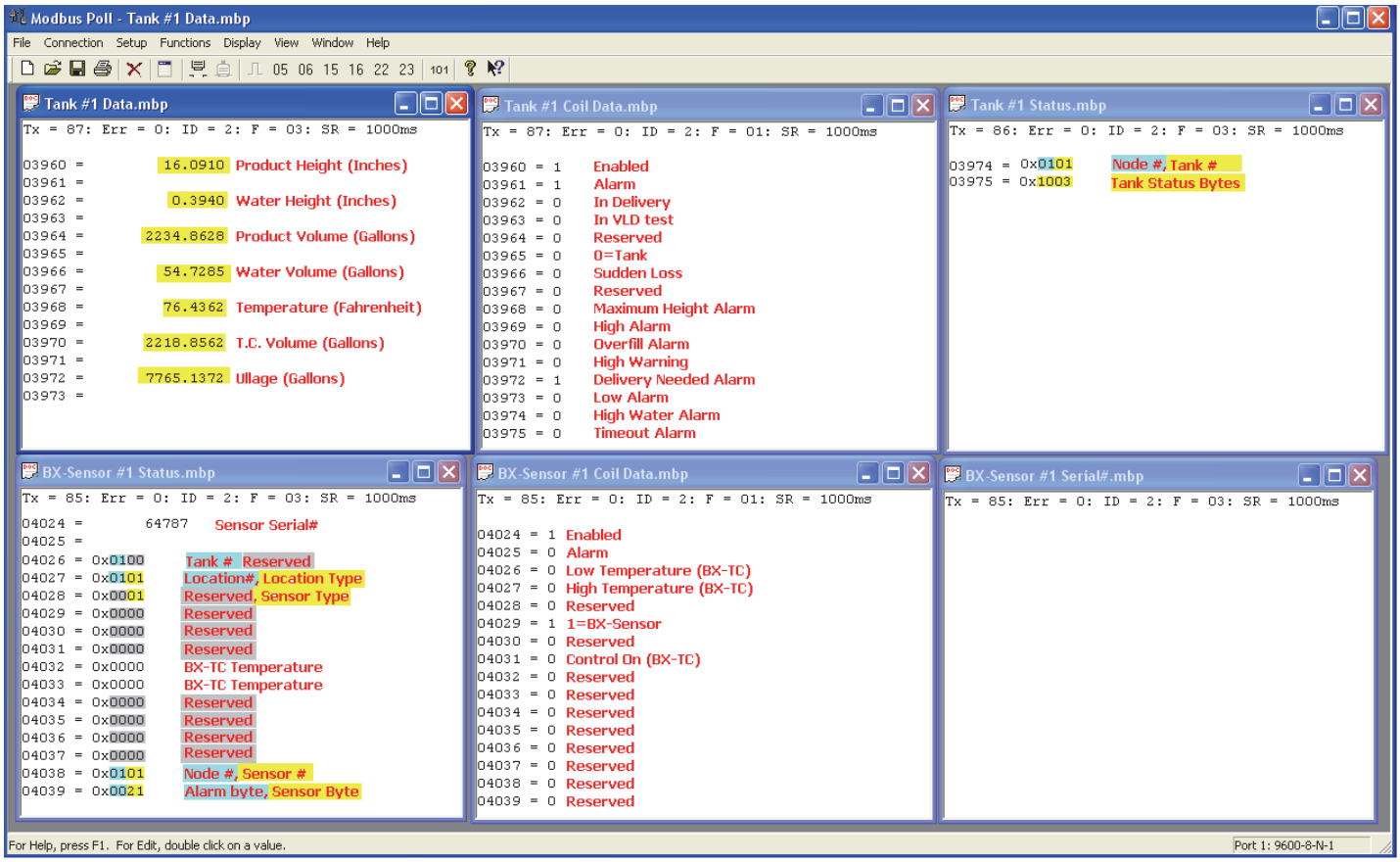


Figure 2: An example of all the register and coil information available for Tank #1 and Sensor #1 when viewed with MODBUS Poll.

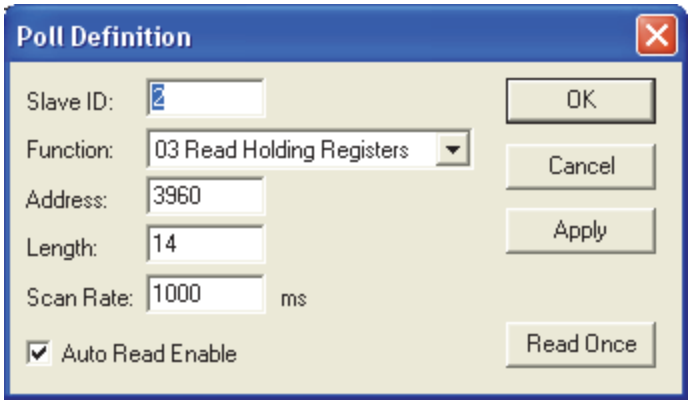


Figure 3: Poll definition for Tank #1 Data (top left window above).

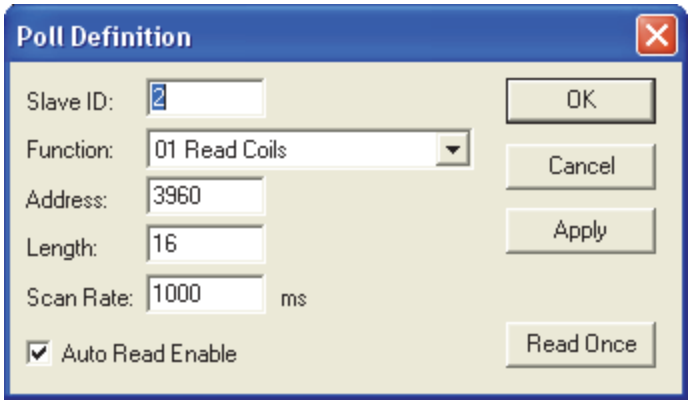


Figure 4: Poll definition for Tank #1 Coil data (top center above).

INTERPRETING COMMUNICATION TRAFFIC

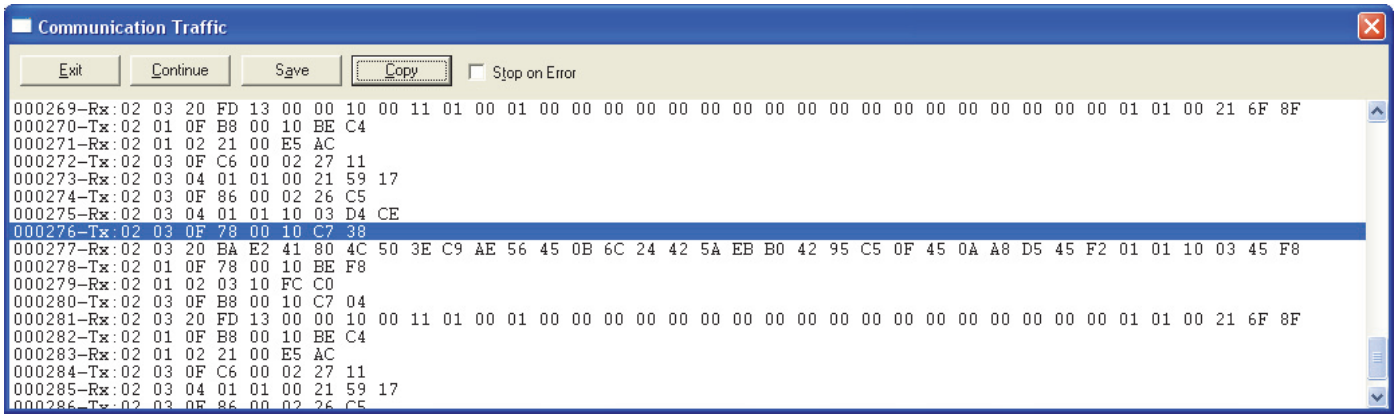


Figure 5: MODBUS Poll Communication Traffic from the example in Figure 2 (data is in hexadecimal format).

000276-TX: 02 03 0F 78 00 10 C7 38 (MODBUS COMMAND)
000277-RX: 02 03 20 BA E2 41 80 4C 50 3E C9 AE 56 45 0B 6C 24 42 5A EB B0 42 95 C5 0F 45 0A A8 D5 45 F2 01 01 10 03 45 F8

EXAMPLE
Read Current Data for Tank 1 (PROTEUS-K4 unit). Read all 16 registers for a tank's inventory. (Note: REVERSE MODE is set to 0)

MODBUS Starting Register Calculation for Tank 1 inventory:

			(See Table 1)		(See Table 2)		
	=	Holding Register Offset	+ Start Register	+	Address Offset.		
Register offset for tank 1 =		0	+ 3960	+	0	=	**3960**

Each tank has 16 registers. Tank 1 = 3960 to 3975 (See Table 2)
3960 (Dec) = 0x0F78 (hex), 16 Registers – 0x0010 (Hex)

Command:
02 03 0F 78 00 10 C7 38

Query from Master:	Description
02	MODBUS Slave address.
03	Read holding register function code.
0F 78	MODBUS holding register, (Tank# 1 starting address = 3960) (0F 78 hex). (Hex)
00 10	Total number of register to read = 16 (10 hex). (Hex)
C7 38	16-bit CRC, low byte first. (Hex)

Response:
02 03 20 BA E2 41 80 4C 50 3E C9 AE 56 45 0B 6C 24 42 5A EB B0 42 95 C5 0F 45 0A A8 D5 45 F2 01 01 10 03 45 F8

Response from Slave:	Description with Number Conversions
02	MODBUS Slave address.
03	Read holding register function code.
20	Total number of bytes to follow = 32. (20 Hex)
BA E2 41 80	Gross Liquid Level = 16.09 inches (Tank #1 first data). (32-bit Float)
4C 50 3E C9	Water Level = 0.39 inches. (32-bit Float)
AE 56 45 0B	Gross Volume = 2234.89 gallons. (32-bit Float)
6C 24 42 5A	Water Volume = 54.68 gallons. (32-bit Float)
EB B0 42 95	Average Temperature = 74.82°F. (32-bit Float)
C5 0F 45 0A	Temperature compensated product volume = 2220.44 gallon. (32-bit Float)
A8 D5 45 F2	Ullage = 7765.10 gallon. (32-bit Float)
01 01	Node Number 1(High Byte), Tank Number 1(Low Byte). (Decimal)
10 03	16 Bits Tank and Alarm Status. (Binary)
45 F8	16 bits CRC, low byte first. (Hex)

EXCEPTION RESPONSE from a slave (see page 33 EXCEPTION CODES)

02 83 09 71 36	**09 = The 'MODBUS Enable' password in the PROTEUS-K is incorrect or not set.**

FLOAT DATA FORMAT - MODBUS REVERSE MODE SETTING

MODBUS Response:
02 03 20 **BA E2 41 80** 4C 50 3E C9 AE 56 45 0B 6C 24 42 5A EB B0 42 95 C5 0F 45 0A A8 D5 45 F2 01 01 10 03 45 F8

Reverse Mode Set to 0 (Default)
BA E2 41 80 = Gross Liquid Level = 16.09 inches (Tank #1 first data). (32-bit Float)

Reverse Mode Set to 1
MODBUS Response:
02 03 20 **41 80 BA E2 ……**

41 80 BA E2 = Gross Liquid Level = 16.09 inches (Tank #1 first data). (32-bit **Inverse** Float)

MODBUS REGISTER MAP COMMAND AND PRINTOUT

The PROTEUS-K and PROTEUS-X units (Release 7 firmware and higher) have a REMOTE COMMAND that displays the MODBUS Registers for each system.
The PROTEUS-K and PROTEUS-X units (Release 8 firmware and higher) have a PRINT MODBUS MAP button on the MODBUS SETUP page.

Command:
MODMAP <enter>

Response:
MODBUS REGISTER MAP

T/S	Start	End
T01	3960	3975
T02	3976	3991
T03	3992	4007
T04	4008	4023
T05	4288	4303
T06	4304	4319
T07	4320	4335
T08	4336	4351
T09	5272	5287
T10	5288	5303
S01	4024	4039
S02	4040	4055
S03	4056	4071
S04	4072	4087
S05	4088	4103

HOW TO CALCULATE A PROTEUS-B/K MODBUS REGISTER ADDRESS

The Read Holding Register active range is 1024 to 8188 (41025 to 48189 with Register OFFSET set to 40001).

The tank registers are mapped by the probe board type (4- or 8-channel).

In a **PROTEUS-K4 (or B4)** the first tank Read Holding Register is 3960. (See Table 1.1)

The first sensor Read Holding Register is 4024. (See Table 1.5)

In a **PROTEUS-K8 (or B8) t**he first tank Read Holding Register is 5272. (See Table 1.2)

The **first tank** Read Holding Registers depends on what type of card it is:

Example 1 On a **PROTEUS-K4 (or B4)** The Read Holding Register for **Tank numbers 1 to 4 are:**
T#1=3960, T#2=3976, T#3=3992, T#4=4008.

Table 2 below will show the breakdown of the data registers for each tank.
Tanks 1's data registers are 3960 to 3975. Tanks 2's data registers are 3976 to 3991, etc..

Example 2 On a **PROTEUS-K8 (or B8)** The Read Holding Register for **Tank numbers 1 to 8 are:**
T#1=5272, T#2=5288 …… T#8=5384.

Table 2 below will show the breakdown of the data registers for each tank.
Tanks 1's data registers are 5272to 5287. Tanks 2's data registers are 5288 to 5303, etc..

NOTE:
The PROTEUS-K unit has a REMOTE COMMAND that displays the MODBUS REGISTERS for each system.
PROTEUS-K Release 8 firmware (and higher) has a PRINT MODBUS MAP button on the MODBUS SETUP page.
See MODBUS REGISTER MAP COMMAND (above).

PROTEUS-B/K MODBUS READ HOLDING REGISTER/COIL LIST NUMBERS

Tank Register Numbering:

TABLE 1.1 PROTEUS-K4 OR PROTEUS-B4 MODBUS TANK READ HOLDING REGISTER/COIL LIST

For PROTEUS systems with <u>only</u> XB-416 probe/sensor cards.

PROTEUS-K/X Data	With MODBUS Offset of 0		Total Registers	With MODBUS Offset of 40001	
	Start Register/Coil	End Register/Coil		Start Register/Coil	End Register/Coil
MTG Probe1 (Tank 1)	3960	3975	16	43961	43976
MTG Probe2 (Tank 2)	3976	3991	16	43977	43992
MTG Probe3 (Tank 3)	3992	4007	16	43993	44008
MTG Probe4 (Tank 4)	4008	4023	16	44009	44024

TABLE 1.2 PROTEUS-K8 OR PROTEUS-B8 MODBUS TANK READ HOLDING REGISTER/COIL LIST

For PROTEUS systems with <u>only</u> XB-800 probe cards.

PROTEUS-K/X Data	With MODBUS Offset of 0		Total Registers	With MODBUS Offset of 40001	
	Start Register/Coil	End Register/Coil		Start Register/Coil	End Register/Coil
MTG Probe1 (Tank 1)	5272	5287	16	45273	45288
MTG Probe2 (Tank 2)	5288	5303	16	45289	45304
MTG Probe3 (Tank 3)	5304	5319	16	45305	45320
MTG Probe4 (Tank 4)	5320	5335	16	45321	45336
MTG Probe1 (Tank 5)	5336	5351	16	45337	45352
MTG Probe2 (Tank 6)	5352	5367	16	45353	45368
MTG Probe3 (Tank 7)	5368	5383	16	45369	45384
MTG Probe4 (Tank 8)	5384	5399	16	45385	45400

HOW TO CALCULATE A PROTEUS-X MODBUS REGISTER ADDRESS

The READ HOLDING REGISTER ACTIVE RANGE is 1024 to 8188 (41025 to 48189 with Register OFFSET set to 40001).

The tank registers are mapped by the probe board type (4- or 8-channel) and the slot location in a PROTEUS-X.

The **PROTEUS-X** unit can have a combination of **XB-416** and **XB-800** boards, in any order.
The tank numbering, as well as the MODBUS addressing, will follow the order of the boards plugged in from slot 1 (right-side of unit, lowest tank number) to slot 7 (highest tank number). See PROTEUS-X diagram.

If the **PROTEUS-X** has only **XB-416** 4-channel probe cards: The first tank Read Holding Register is 3960. (See Table 1.3)
The first sensor Read Holding Register is 4024. (See Table 1.5)

If the **PROTEUS -X** has only **XB-800** 8-channel probe cards: The first tank Read Holding Register is 5792. (See Table 1.4)

If the **PROTEUS -X** has both **XB-416** 4-channel probe cards and **XB-800** 8-channel probe cards: (See Table 1.4)

The **first tank** Read Holding Registers depends on what type of card is in slot #1:

Example 1 — If an **XB-416** board is in slot #1 (**1st XB-416 Bd** probe positions 1 to 4), the Read Holding Register for **Tank numbers 1 to 4:** T#1=3960, T#2=3976, T#3=3992, T#4=4008.

Example 1a — If a **XB-800** board is in slot #1 (**1st XB-800 Bd** probe positions 1 to 8) The Read Holding Register for **Tank numbers 1 to 8:** T#1=5272, T#2=5288 …… T#8=5384.

The 2nd **probe board** in slot #2, the systems register map, again will depend on which type of probe board is plugged into the second slot:

Example 2 — If the first slot board was an **XB-416** board and the second slot board is a **(2nd) XB-416 board** the registers will continue with **Tank numbers 5 to 8:** T#5=4288, T#2=4304, T#3=4320, T#4=4336.

Example 3 — If the first slot board was an **XB-416** board and the second slot board is an **XB-800 board**, the registers will continue with **Tank numbers 5 to 12:** T#5=5272, T#6=5288 …… T#12=5384.

Example 4 — If the first slot board was an **XB-800** and the second slot board is an **XB-416 board**, the registers will continue with **Tank numbers 9 to 12:** T#9=3960, T#10=3976, T#11=3992, T#12=4008. Note: The Register numbers are lower for tanks 9 thru 12 (3960-4008) then for tanks 1 thru 8(5272-5384).

Example 5 — If the first slot board was an **XB-800** and the second slot board is a (second) **XB-800 board**, the registers will continue with **Tank numbers 9 to 16:** T#9=5408, T#10=5424 …… T#16=5520.

The third **probe board** in the system will follow the above format for up to four boards of any one type.
Table 1.4 shows **Registers for PROTEUS systems with both XB-416 and XB-800 probe cards.**

NOTE:
The PROTEUS-K and PROTEUS-X units have a REMOTE COMMAND that displays the MODBUS REGISTERS for each system. PROTUES-K and PROTEUS-X Release 8 firmware (and higher) has a PRINT MODBUS MAP button on the MODBUS SETUP page. See MODBUS REGISTER MAP COMMAND (above).

TABLE 1.3 PROTEUS-X MODBUS TANK READ HOLDING REGISTER/COIL LIST NUMBERS

For PROTEUS systems with only XB-416 probe/sensor cards.

PROTEUS-K/X Data	With MODBUS Offset of 0		Total Registers	With MODBUS Offset of 40001		XB-416 Probe Card Number
	Start Register/Coil	End Register/Coil		Start Register/Coil	End Register/Coil	
1st XB-416 Board **						
MTG Probe1 (Tank 1)	3960	3975	16	43961	43976	1
MTG Probe2 (Tank 2)	3976	3991	16	43977	43992	1
MTG Probe3 (Tank 3)	3992	4007	16	43993	44008	1
MTG Probe4 (Tank 4)	4008	4023	16	44009	44024	1
2rd XB-416 Board **						
MTG Probe1 (Tank 5)	4288	4303	16	44289	44304	2
MTG Probe2 (Tank 6)	4304	4319	16	44305	44320	**2**
MTG Probe3 (Tank 7)	4320	4335	16	44321	44336	**2**
MTG Probe4 (Tank 8)	4336	4351	16	44337	44352	**2**
3rd XB-416 Board **						
MTG Probe1 (Tank 9)	4616	4631	16	44617	44632	**3**
MTG Probe2 (Tank 10)	4632	4647	16	44633	44648	**3**
MTG Probe3 (Tank 11)	4648	4663	16	44649	44664	**3**
MTG Probe4 (Tank 12)	4664	4679	16	44665	44680	**3**
4th XB-416 Board **						
MTG Probe1 (Tank 13)	4944	4959	16	44945	44960	**4**
MTG Probe2 (Tank 14)	4960	4975	16	44961	44976	**4**
MTG Probe3 (Tank 15)	4976	4991	16	44977	44992	**4**
MTG Probe4 (Tank 16)	4992	5007	16	44993	45008	**4**

**** The tanks are numbered in the PROTEUS-X unit from the right-most slot (Tank 1) to the left-most slot (Tank 16).**

TABLE 1.4 PROTEUS-X MODBUS TANK READ HOLDING REGISTER/COIL LIST NUMBERS

For PROTEUS systems with Both XB-416 and XB-800 probe cards.
(Tank numbers will be assigned by the boards position in the PROTEUS-X)

PROTEUS-K/X Data	With MODBUS Offset of 0		Total Registers	With MODBUS Offset of 40001		XB-416 Probe Card Number
	Start Register/Coil	End Register/Coil		Start Register/Coil	End Register/Coil	
1st XB-416 Board **						
MTG Probe Position 1	3960	3975	16	43961	43976	1
MTG Probe Position 2	3976	3991	16	43977	43992	1
MTG Probe Position 3	3992	4007	16	43993	44008	1
MTG Probe Position 4	4008	4023	16	44009	44024	1
2rd XB-416 Board **						
MTG Probe Position 1	4288	4303	16	44289	44304	2
MTG Probe Position 2	4304	4319	16	44305	44320	**2**
MTG Probe Position 3	4320	4335	16	44321	44336	**2**
MTG Probe Position 4	4336	4351	16	44337	44352	**2**
3rd XB-416 Board **						
MTG Probe Position 1	4616	4631	16	44617	44632	**3**
MTG Probe Position 2	4632	4647	16	44633	44648	**3**
MTG Probe Position 3	4648	4663	16	44649	44664	**3**
MTG Probe Position 4	4664	4679	16	44665	44680	**3**
4th XB-416 Board **						
MTG Probe Position 1	4944	4959	16	44945	44960	**4**
MTG Probe Position 2	4960	4975	16	44961	44976	**4**
MTG Probe Position 3	4976	4991	16	44977	44992	**4**
MTG Probe Position 4	4992	5007	16	44993	45008	**4**

**** The tanks are numbered in the PROTEUS-X unit from the right most slot (Tank 1) to the left most slot (Tank # ***).**
***** Tank number depends on how many 4- and 8-channel probe cards are in the system.**

TABLE 1.4 PROTEUS-X MODBUS TANK READ HOLDING REGISTER/COIL LIST NUMBERS (CONT'D)

PROTEUS-K/X Data	With MODBUS Offset of 0		Total Registers	With MODBUS Offset of 40001		XB-800 Probe Card Number
	Start Register/Coil	End Register/Coil		Start Register/Coil	End Register/Coil	
1st XB-800 Board **						
MTG Probe Position 1	5272	5287	16	45273	45288	1
MTG Probe Position 2	5288	5303	16	45289	45304	1
MTG Probe Position 3	5304	5319	16	45305	45320	1
MTG Probe Position 4	5320	5335	16	45321	45336	1
MTG Probe Position 5	5336	5351	16	45337	45352	1
MTG Probe Position 6	5352	5367	16	45353	45368	**1**
MTG Probe Position 7	5368	5383	16	45369	45384	**1**
MTG Probe Position 8	5384	5399	16	45385	45400	**1**
2rd XB-800 Board **						
MTG Probe Position 1	5408	5423	16	45409	45424	**2**
MTG Probe Position2	5424	5439	16	45425	45440	**2**
MTG Probe Position 3	5440	5455	16	45441	45456	**2**
MTG Probe Position 4	5456	5471	16	45457	45472	**2**
MTG Probe Position 5	5472	5487	16	45473	45488	**2**
MTG Probe Position 6	5488	5503	16	45489	45504	**2**
MTG Probe Position 7	5504	5519	16	45505	45520	**2**
MTG Probe Position8	5520	5535	16	45521	45536	**2**

**** The tanks are numbered in the PROTEUS-X unit from the right-most slot (Tank 1) to the left-most slot (Tank # ***).**
***** Tank number depends on how many 4- and 8-channel probe cards are in the system.**

TABLE 1.4 PROTEUS-X MODBUS TANK READ HOLDING REGISTER/COIL LIST NUMBERS (CONT'D)

PROTEUS-K/X Data	With MODBUS Offset of 0		Total Registers	With MODBUS Offset of 40001		XB-800 Probe Card Number
	Start Register/Coil	End Register/Coil		Start Register/Coil	End Register/Coil	
3rd XB-800 Board **						
MTG Probe1 (Tank 17)	5544	5559	16	45545	45560	3
MTG Probe2 (Tank 18)	5560	5575	16	45561	45576	3
MTG Probe3 (Tank 19)	5576	5591	16	45577	45592	3
MTG Probe4 (Tank 20)	5592	5607	16	45593	45608	3
MTG Probe1 (Tank 21)	5608	5623	16	45609	45624	3
MTG Probe2 (Tank 22)	5624	5639	16	45625	45640	**3**
MTG Probe3 (Tank 23)	5640	5655	16	45641	45656	**3**
MTG Probe4 (Tank 24)	5656	5671	16	45657	45672	**3**
4th XB-800 Board **						
MTG Probe1 (Tank 25)	5680	5695	16	45681	45696	**4**
MTG Probe2 (Tank 26)	5696	5711	16	45697	45712	**4**
MTG Probe3 (Tank 27)	5712	5727	16	45713	45728	**4**
MTG Probe4 (Tank 28)	5728	5743	16	45729	45744	**4**
MTG Probe1 (Tank 29)	5744	5759	16	45745	45760	**4**
MTG Probe2 (Tank 30)	5760	5775	16	45761	45776	**4**
MTG Probe3 (Tank 31)	5776	5791	16	45777	45792	**4**
MTG Probe4 (Tank 32)	5792	5807	16	45793	45808	**4**

**** The tanks are numbered in the PROTEUS-X unit from the right most slot (Tank 1) to the left most slot (Tank # ***).**
***** Tank number depends on how many 4- and 8-channel probe cards are in the system.**

SENSORS REGISTER NUMBERING

TABLE 1.5 PROTEUS-B/K/X MODBUS SENSOR READ HOLDING REGISTER/COIL LIST

For all PROTEUS systems with XB-416 probe/sensor cards.

The sensors are numbered in the PROTEUS-X unit from the right-most slot on the XB-416 board (Sensor 1-16) to the left-most slot (Sensor 17-32, 33-48 and 49-64, depending on how many 4-channel probe cards are in the system).

1ST XB-416 BOARD

PROTEUS-X Data	With MODBUS Offset of 0		Total Registers	With MODBUS Offset of 40001		XB-416 Probe Card Number
	Start Register/Coil	End Register/Coil		Start Register/Coil	End Register/Coil	
BX-Sensor 1	4024	4039	16	44025	44040	1
BX-Sensor 2	4040	4055	16	44041	44056	
BX-Sensor 3	4056	4071	16	44057	44072	
BX-Sensor 4	4072	4087	16	44073	44088	
BX-Sensor 5	4088	4103	16	44089	44104	
BX-Sensor 6	4104	4119	16	44105	44120	
BX-Sensor 7	4120	4135	16	44121	44136	
BX-Sensor 8	4136	4151	16	44137	44152	
BX-Sensor 9	4152	4167	16	44153	44168	
BX-Sensor 10	4168	4183	16	44169	44184	
BX-Sensor 11	4184	4199	16	44185	44200	
BX-Sensor 12	4200	4215	16	44201	44216	
BX-Sensor 13	4216	4231	16	44217	44232	
BX-Sensor 14	4232	4247	16	44233	44248	
BX-Sensor 15	4248	4263	16	44249	44264	
BX-Sensor 16	4264	4279	16	44265	44280	

TABLE 1.5 MODBUS SENSOR READ HOLDING REGISTER/COIL LIST FOR PROTEUS-X ONLY (CONT'D)

2ND XB-416 BOARD

PROTEUS-X Data	With MODBUS Offset of 0		Total Registers	With MODBUS Offset of 40001		XB-416 Probe Card Number
	Start Register/Coil	End Register/Coil		Start Register/Coil	End Register/Coil	
BX-Sensor 17	4352	4367	16	44353	44368	2
BX-Sensor 18	4368	4383	16	44369	44384	
BX-Sensor 19	4384	4399	16	44385	44400	
BX-Sensor 20	4400	4415	16	44401	44416	
BX-Sensor 21	4416	4431	16	44417	44432	
BX-Sensor 22	4432	4447	16	44433	44448	
BX-Sensor 23	4448	4463	16	44449	44464	
BX-Sensor 24	4464	4479	16	44465	44480	
BX-Sensor 25	4480	4495	16	44481	44496	
BX-Sensor 26	4496	4511	16	44497	44512	
BX-Sensor 27	4512	4527	16	44513	44528	
BX-Sensor 28	4528	4543	16	44529	44544	
BX-Sensor 29	4544	4559	16	44545	44560	
BX-Sensor 30	4560	4575	16	44561	44576	
BX-Sensor 31	4576	4591	16	44577	44592	
BX-Sensor 32	4592	4607	16	44593	44608	

TABLE 1.5 MODBUS SENSOR READ HOLDING REGISTER/COIL LIST FOR PROTEUS-X ONLY (CONT'D)

3RD XB-416 BOARD

PROTEUS-X Data	With MODBUS Offset of 0		Total Registers	With MODBUS Offset of 40001		XB-416 Probe Card Number
	Start Register/Coil	End Register/Coil		Start Register/Coil	End Register/Coil	
BX-Sensor 33	4680	4695	16	44681	44696	3
BX-Sensor 34	4696	4711	16	44697	44712	
BX-Sensor 35	4712	4727	16	44713	44728	
BX-Sensor 36	4728	4743	16	44729	44744	
BX-Sensor 37	4744	4759	16	44745	44760	
BX-Sensor 38	4760	4775	16	44761	44776	
BX-Sensor 39	4776	4791	16	44777	44792	
BX-Sensor 40	4792	4807	16	44793	44808	
BX-Sensor 41	4808	4823	16	44809	44824	
BX-Sensor 42	4824	4855	16	44825	44840	
BX-Sensor 43	4840	4867	16	44841	44856	
BX-Sensor 44	4856	4871	16	44857	44872	
BX-Sensor 45	4872	4887	16	44873	44888	
BX-Sensor 46	4888	4903	16	44889	44904	
BX-Sensor 47	4904	4919	16	44905	44920	
BX-Sensor 48	4920	4935	16	44921	44936	

TABLE 1.5 MODBUS SENSOR READ HOLDING REGISTER/COIL LIST FOR PROTEUS-X ONLY (CONT'D)

4TH XB-416 BOARD

PROTEUS-X Data	With MODBUS Offset of 0		Total Registers	With MODBUS Offset of 40001		XB-416 Probe Card Number
	Start Register/Coil	End Register/Coil		Start Register/Coil	End Register/Coil	
BX-Sensor 49	5008	5023	16	45009	45024	4
BX-Sensor 50	5024	5039	16	45025	45040	
BX-Sensor 51	5040	5055	16	45041	45056	
BX-Sensor 52	5056	5071	16	45057	45072	
BX-Sensor 53	5072	5087	16	45073	45088	
BX-Sensor 54	5088	5103	16	45089	450104	
BX-Sensor 55	5104	5119	16	45105	45120	
BX-Sensor 56	5120	5135	16	45121	45136	
BX-Sensor 57	5136	5151	16	45137	45152	
BX-Sensor 58	5152	5167	16	45153	45168	
BX-Sensor 59	5168	5183	16	45169	45184	
BX-Sensor 60	5184	5199	16	45185	45200	
BX-Sensor 61	5200	5215	16	45201	45216	
BX-Sensor 62	5216	5231	16	45217	45232	
BX-Sensor 63	5232	5247	16	45233	45248	
BX-Sensor 64	5248	5263	16	45249	45264	

PROTEUS-B/K/X MODBUS TANK REGISTER ADDRESSES

Table 2 (below) lists the tank data registers. Use the formula below to calculate each tank's register/coil locations.

MODBUS Holding Register Calculation for Tank levels and alarm data:

			(See Table 1)		(See Table 2)			
	=	Holding Register Offset	+ Tank	Start Register	+	Address Offset.		Note
Register offset for tank X	**=**	0 or 40001	+	**xxxx**	+	**x**	= **xxxx**	
PROTEUS-K4 unit (or B4 unit)								
Example 1: Tank #1 Alarm Status =		(0)	+	(3960)	+	(15)	= 3975	XB-416 4-Channel Board
Example 2: Tank #3 Water Level =		(0)	+	(3992)	+	(2)	= 3994 + 5	
Example 3: Tank #2 High alarm **Coil** =		(0)	+	(3976)	+	(9) (Coils, see Table 3)	= 3985	
PROTEUS-K8 unit (or B8 unit)								
Example 4: Tank #1 Alarm Status =		(0)	+	(5272)	+	(15)	= 5287	XB-800 8-Channel Board
PROTEUS-X unit								Slot 1 Slot 2
Example 4: Tank #5 Gross Volume =		(0)	+	(5272)	+	(4)	= 5276 + 7	XB-416 and XB-800 Boards
								See Table 1 Example 3 (above pg. 8)

TABLE 2. TANK CURRENT DATA HOLDING REGISTERS

Address Offset	Parameter Name	Dimension[1]	Data Type
0 1	Gross Liquid Level.	Inches	Floating Point
2 3	Water Level.	Inches	Floating Point
4 5	Gross Volume. (Water Volume Included)	Gallons	Floating Point
6 7	Water Volume.	Gallons	Floating Point
8 9	Average Temperature. The average of the submerged thermistor temperature.	°F	Floating Point
10 11	Temperature Compensated Product Volume. The product volume calculated when the product is referenced to 60°F.	Gallons	Floating Point
12 13	Ullage. (Ullage = Capacity – Total Gross)	Gallons	Floating Point
14	High Byte: Node number, Low Byte: Tank number.[2]	N/A	Hex[2] Binary
15	Tank and Alarm Status.[3]		

Note:

1. Dimension will change to metric if units in PROTEUS-K are changed to metric (centimeters/liters/Celsius).
2. See Table 2.1 for binary breakdown of register 14.
3. See Table 2.2 for binary breakdown of register 15.

TABLE 2.1 REGISTER 14 BREAKDOWN

Register 14		Node Number and Tank Number	
	BITS	**Parameter Name**	**Data Type[1]**
High Byte	B15 – B8	Node Number	Hex
Low Byte	B7 – B0	Tank Number	Hex

Note:
1. Each byte must be converted from Hex to Decimal; example 0x11 equals decimal 16.

TABLE 2.2 REGISTER 15 BREAKDOWN

Register 15		Tank Status and Alarms	
	BITS	**Parameter Name**	**Data Type[1]**
High Byte	B15	Timeout Alarm(R)	Binary
	B14	High Water Alarm (R)	Binary
	B13	Low Alarm(R)	Binary
	B12	Delivery Needed Alarm(R)	Binary
	B11	High Warning (R)	Binary
	B10	Overfill Alarm (R)	Binary
	B9	High Alarm(R)	Binary
	B8	Maximum Height Alarm(R)	Binary
Low Byte	B7	CITLD Pass for Month	Binary
	B6	Sudden Loss (R)	Binary
	B5	Reserved (0)	Binary
	B4	Last VLD Test Pass	Binary
	B3	In VLD Test (R)	Binary
	B2	In Delivery (R)	Binary
	B1	Alarm (R)	Binary
	B0	Enabled	Binary

Note: Any alarm bit with an (R), can be a relay card trip point selection.
1. 1= On/Active, 0=Off/Inactive

TABLE 3. COILS: TANK STATUS AND ALARMS

Address Offset	Parameter Name	Data Type[1]
0	Enabled	Binary
1	Alarm (R)	Binary
2	In Delivery (R)	Binary
3	In VLD Test (R)	Binary
4	Reserved	Binary
5	0: Represents Tank	Binary
6	Sudden Loss (R)	Binary
7	Reserved	Binary
8	Maximum Height Alarm(R)	Binary
9	High Alarm(R)	Binary
10	Overfill Alarm (R)	Binary
11	High Warning (R)	Binary
12	Delivery Needed Alarm(R)	Binary
13	Low Alarm(R)	Binary
14	High Water Alarm (R)	Binary
15	Timeout Alarm(R)	Binary

Note: Any alarm bit with an (R), can be a relay card trip point selection.
1. 1= On/Active, 0=Off/Inactive

PROTEUS MODBUS SENSOR REGISTER ADDRESSES

Table 4 (below) lists the BX-Sensor data registers. Use the formula below to calculate each sensor's register/coil locations.

			(See Table 1.2)		(See Table 4)			
	=	Holding Register Offset	+	Sensor Start Register	+	Address Offset.		
Register offset for Sensor 1	=	0	+	4024	+	14	= **4038**	Sensor # (Low Byte)
					+	15	= **4039**	Alarm Byte

TABLE 4. BX-SENSOR CURRENT DATA

Address Offset	Parameter Name	Dimension[1]	Data Type
0 1	Serial Number	N/A	Long
2[2] 3[2]	High Byte: Tank Number, Low Byte: Reserved Byte High Byte: Location Count, Low Byte: Location Type[4]	N/A	Hex
4[2] 5	High Byte: Reserved Byte, Low Byte: Sensor Type[5] High Byte: Reserved Byte, Low Byte: Reserved Byte	N/A	Hex
6 7	Reserved	N/A	N/A
8 9	BX-TC Current Temperature	°F	Floating Point
10 11	Reserved	N/A	N/A
12 13	Reserved	N/A	N/A
14[3] 15[3]	High Byte: Node Number, Low Byte: Sensor Number High Byte: Alarm Byte[6], Low Byte: Sensor Byte	N/A	Hex Hex/Binary

Notes:
1. Dimension will change to metric if units in PROTEUS-K are changed to metric (centimeters/liters/Celsius).
2. Each byte must be converted from Hex to Decimal; example 0x11 equals decimal 16.
3. See Table 4.1 and 4.2 for Registers 14 & 15 breakdown.
4. See Table 6 for Location Type.
5. See Table 7 for Sensor Type.
6. See Table 8 for Alarm Byte.

TABLE 4.1 REGISTER 14 BREAKDOWN

Register 14		Node Number and Sensor Number	
	BITS	**Parameter Name**	**Data Type[1]**
High Byte	B15 – B8	Node Number	Hex
Low Byte	B7 – B0	Sensor Number	Hex

Note:
1. Each byte must be converted from Hex to Decimal; example 0x11 equals decimal 16.

TABLE 4.2 REGISTER 15 BREAKDOWN

Register 15		Alarm Byte and Sensor Byte	
	BITS	**Parameter Name**	**Data Type[2]**
High Byte[1]	B15 – B8	Alarm Byte.[1]	Hex
Low Byte[2]	B7	Reserved	Binary
	B6	Reserved	Binary
	B5	1: Represents Sensor	Binary
	B4	ACK	Binary
	B3	Hi Temp (R)	Binary
	B2	Low Temp (R)	Binary
	B1	Alarm (R)	Binary
	B0	Enabled	Binary

Notes: Any alarm bit with an (R), can be a Relay card trip point selection.
1. High byte must be converted from Hex to decimal, then use Table 8 to determine alarm.
2. 1=On/Active, 0=Off/Inactive

PROTEUS MODBUS SENSOR COIL ADDRESSES

Table 5 (below) lists the BX-Sensor Coils. Use the formula below to calculate each sensor's Coil location.
MODBUS Coil Calculation for sensor alarm data:

				(See Table 1.1)		(See Table 5)		
	=	Holding Register Offset	+	Sensor Start Register	+	Address Offset.		
Coil offset for Sensor X	**=**	0 or 40001	+	**xxxx**	+	**x**	=	**xxxx**
Example 1: Sensor #1 Alarm Status =		(0)	+	(4024)	+	(1)	=	4025
Example 2: Sensor #3 Alarm Status =		(0)	+	(4056)	+	(1)	=	4057

TABLE 5. BX-SENSOR STATUS AND ALARM COILS

Address Offset	Parameter Name	Data Type[1]
0	Enabled	Binary
1	Alarm (R)	Binary
2	Low Temperature (BX-TC) (R)	Binary
3	High Temperature (BX-TC) (R)	Binary
4	Reserved	Binary
5	1: Represents BX-Sensor	Binary
6	Reserved	Binary
7	Control On (BX-TC)	Binary
8	Reserved	Binary
9	Reserved	Binary
10	Reserved	Binary
11	Reserved	Binary
12	Reserved	Binary
13	Reserved	Binary
14	Reserved	Binary
15	Reserved	Binary
Note: Any alarm bit with an (R), can be a relay card trip point selection. 1. 1= On/Active, 0=Off/Inactive		

TABLE 6. BX-SENSOR LOCATION TYPE

Index	Location Type
0	None
1	Sump Sensor
2	Double Wall Sensor
3	Dispenser Sensor
4	Interstitial Sensor
5	Annular Sensor
6	Hi Level Sensor
7	HiHi Level Sensor
8	Caution Level Sensor
9	Reservoir Sensor
10	Brine Sensor
11	Other Sensor
12	Vault Sensor
13	Well Sensor
14	Low Level Sensor
15	Refrig Sensor
16	Freezer Sensor
17	Hi Caution Sensor
18	HiHi Hi Sensor
19	EmHi Hi Sensor
20	Hi Low Sensor
21	HiHi LoLo Sensor
22	EmHi EmLo Sensor
23	Lo LoLo Sensor
24	Lo EmLo Sensor
25	Caution Lo Sensor
26	Pressure Sensor1 (Vertical)
27	Hi Pressure Sensor2
28	Therm Other Sensor
29	Fill
30	PT Tank

TABLE 7. BX-SENSOR TYPE

Index	Sensor Types
1	BX-LS
2	BX-LWF
3	BX-LV
4	BX-LSR
5	BX-PDS
6	BX-PDWF
7	BX-PDWS
8	BX-RES
9	BX-LM2A
10	BX-LM2B
11	BX-LM2C
12	BX-VS
13	BX-UTVR
14	BX-UT1
15	BX-PT
16	BX-TC1
17	BX-PROBE
18	BX-FILCK
19	BX-IPS

TABLE 8. BX-SENSOR ALARM BYTE

Index	Sensor Current Status
0	Normal
1	Setup Data Warning
2	Product Alarm
3	Open Alarm
4	Short Alarm
5	Water Alarm
6	Dry Alarm
7	High Liquid Alarm
8	Low Liquid Alarm
9	Liquid Warning
15	Alarm
16	Liquid Alarm
17	Vapors Detected
18	Level Alarm
19	No Reply
20	Bus No Reply
21	Low Temp Alarm
22	High Temp Alarm
23	High Level Alarm (Group High)
24	High/High Level Alarm
25	Em High Level Alarm
26	High Level Alarm
27	High/High Level Alarm
28	Em High Level Alarm
29	Low Level Alarm
30	Low Level Alarm
31	Caution Level Alarm
32	Caution Level Alarm (Group Low)
33	High Level Alarm
34	High Level Alarm
35	Low Level Alarm
36	Low/Low Level Alarm
37	Em Low Level Alarm
38	Low/Low Level Alarm
39	Em Low Level Alarm
40	Low Level Alarm
41	Caut Failed/Hi L Alarm
42	Hi Failed/HiHi L Alarm
43	Hi Failed/EmHi L Alarm
44	Low Reservoir Alarm (Bx-lsr sensor)

APPENDIX

1. **ASCII Hex Floating-Point Format Conversion**
    a. ASCII Hex Conversion to Decimal, Example in VB6

2. **CRC Checksum Formula Example**
    a. CRC calculation program in VB
    b. CRC calculation program in ‘C’

3. **MODBUS Exception Responses**

4. **Troubleshooting Notes**

5. **MODBUS Tools - MODBUS Poll Example Screens**

1. ASCII HEX FLOATING-POINT FORMAT CONVERSION

Some remote commands use an ASCII Hex format when transmitting or receiving numbers. This procedure will explain how to convert a number from decimal to ASCII Hex and from ASCII Hex to decimal.

Floating-Point Numbers

There are two problems with integers; they cannot express fractions, and the range of the number is limited to the number of bits used. An efficient way of storing fractions is called the floating-point method, which involves splitting the fraction into two parts, an exponent and a mantissa.
The exponent represents a value raised to the power of 2.
The mantissa represents a fractional value between 0 and 1.

Consider the number: 12.50

The number is first converted into the format:

2n * 0.xxxxxx

where **n** represents the exponent and 0.xxxxx is the mantissa.

The computer industry agreed upon a standard for the storage of floating-point numbers. It is called the IEEE 754 standard and uses 32-bits of memory (for single precision), or 64-bits (for double-precision accuracy). The single-precision format looks like:

```
 Bits
  31  30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10  9  8  7  6  5  4  3  2  1  0
 +---+-----------------------+------------------------------------------------------------------+
 | S | Exponent value        |              Mantissa value                                      |
 +---+-----------------------+------------------------------------------------------------------+
   S = Sign Bit
```

The sign bit is 1 for a negative mantissa, and 0 for a positive mantissa.
The exponent uses a bias of 127.
The mantissa is stored as a binary value using an encoding technique.

Working out the FP bit patterns

The number we have is:

12.5

which expressed as fraction to the power of 2 is:

12.5 / 2 = 6.25
6.25 / 2 = 3.125
3.125 / 2 = 1.5625
1.5625 / 2 = 0.78125

NOTE: Keep dividing by 2 till a fraction between 0 and 1 results. The fraction is the mantissa value, the number of divisions is the exponent value.
Thus, our values now are,

0.78125 * (2*2*2*2) (comment: this is 2 to the power of 4) = 12.5

The exponent bit pattern is stored using an excess of 127. This means that this value is added to the exponent when storing (and subtracted when removing).
The exponent bit pattern to store is:

4 + 127 = 131 decimal = '10000011 Hex'

As the mantissa is a positive value, the sign bit is 0.
The mantissa is a little more complicated to work out. Each bit represents 2 to the power of a negative number. It looks like:

1st bit of mantissa = 0.5
2nd = 0.25
3rd = 0.125
4th = 0.0625
5th = 0.03125
etc

The mantissa number value we have is 0.78125, which in binary is:
11001000000000000000000 (0.5 + 0.25 + 0.03125) = 0.78125

However, to make matters even more complicated, the mantissa is normalized by moving the bit patterns to the left (each shift subtracts one from the exponent value) till the first 1 drops off.
The resulting pattern is then stored.
The mantissa now becomes:
10010000000000000000000

and the exponent is adjusted to become:
131 - 1 = 130 = '10000010'

The final assembled format is:
Bits
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
+--+----------+--+
| 0| 1 0 0 0 0 0 1 0| 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 |
+--+----------+--+
S Exponent Mantissa

which equals = 0100 0001 0100 1000 0000 0000 0000 0000 Hex = 41480000 Hexadecimal (= FFFFFFFF used in command)

Now let's convert the following ASCII Hex format back into a decimal number.

Bits
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
+--+----------+--+
| 0| 1 0 0 0 0 0 1 0| 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 |
+--+----------+--+
S Exponent Mantissa

Sign bit is 0 = positive number.

The exponent value of, 10000010 Hex = 2^1 + 2^7 = 130 decimal
130 - 127 = 3

And a mantissa value of, 10010000000000000000000 Hex = 2^22 + 2^19 = 4718592 decimal

1 + (4718592 / 8388608) = 1.5625 OR bit 22 = 0.5, bit 21 = 0.25, etc
1 + (0.5 + 0.0625) = 1.5625

Thus, the number is:
1.5625 * 2^3 = 12.5

a. ASCII Hex Conversion to Decimal, Example in VB6

```
Function convert_gggggg(ch As String)

    A = Mid(ch, 1, 1)
    b = Mid(ch, 2, 1)
    c = Mid(ch, 3, 1)
    d = Mid(ch, 4, 1)
    e = Mid(ch, 5, 1)
    f = Mid(ch, 6, 1)
    g = Mid(ch, 7, 1)
    h = Mid(ch, 8, 1)
   A = convert_num(A)
   b = convert_num(b)
   c = convert_num(c)
   d = convert_num(d)
   e = convert_num(e)
   f = convert_num(f)
   g = convert_num(g)
   h = convert_num(h)
   s = 0
   If (c And 8) Then s = 1
   If (A And 8) Then A = A - 8 'remove sign bit.
   a1 = ((A * 16 + b) * 2) + s
   If (s = 1) Then c = c - 8 'remove 24th bit.
   B1 = ((c * 1048576) + (d * 65536) + (e * 4096) + (f * 256) + (g * 16) + h)
   c1 = 2 ^ (a1 - 127)
   d1 = (B1 / 8388608) + 1
   e1 = Format((c1 * d1), "0.00") 'Round(c1 * d1)

   convert_gggggg = e1

   End Function

Function convert_num(ch)

 If ch = "a" Or ch = "A" Then ch = 10
 If ch = "b" Or ch = "B" Then ch = 11
 If ch = "c" Or ch = "C" Then ch = 12
 If ch = "d" Or ch = "D" Then ch = 13
 If ch = "e" Or ch = "E" Then ch = 14
 If ch = "f" Or ch = "F" Then ch = 15

 convert_num = ch  'return

End Function
```

2. CRC CHECKSUM FORMULA EXAMPLE

a. CRC calculation program in VB

```
Function CRCValue(InData As String) As String
  Dim B1 As Integer
  Dim IDX As Long
  Dim CRC As Long
  Dim ByteVal As Long
  Dim CheckCRC As Long
  Dim XLoop As Integer
  Dim CRCHi As Long
  Dim CRCLo As Long

  CRC = 65535
  For B1 = 1 To Len(InData) Step 2
    ByteVal = Val("&H" + Mid$(InData, B1, 2))

    CRC = (CRC Xor ByteVal) And 65535

    For XLoop = 1 To 8
      CheckCRC = CRC And 1
      CRC = (CRC \ 2) And 65535
      If (CheckCRC = 1) Then
        CRC = (CRC Xor 40961) And 65535
      End If
    Next XLoop

  Next B1

  CRCHi = Fix(CRC / 256) And 255
  CRCLo = CRC And 255
  CRC = CRCLo * 256 + CRCHi

  CRCValue = Right$("0000" + Hex$(CRC And 65535), 4)
End Function
```

b. CRC calculation program in 'C'

```
// Inputs - buf = string to calculate checksum from.  index=starting point in buf[].  len = length of string in buf[].
// Return (temp) is a 2 byte (16 bit) CRC checksum.
unsigned short crc16(unsigned char buf[], unsigned short index, unsigned char len)
{
        unsigned short temp;
        unsigned char i, test;

        temp = 0xffff;

        while (len--)
        {
                temp ^= buf[index++];
                index &= 0x03ff;
                i = 0;
                while (i++ < 8)
                {
                        test = temp & 0x0001;
                        temp >>= 1;
                        if (test)
                                temp ^= 0xa001;
                }
        }
        return temp;
}
```

3. MODBUS EXCEPTION RESPONSES

Except for broadcast messages, when a master device sends a query to a slave device it expects a normal response. One of four possible events can occur from the master's query:

- If the slave device receives the query without a communication error, and can handle the query normally, it returns a normal response.
- If the slave does not receive the query due to a communication error, no response is returned. The master program will eventually process a timeout condition for the query.
- If the slave receives the query but detects a communication error (parity, LRC, or CRC), no response is returned. The master program will eventually process a timeout condition for the query.
- If the slave receives the query without a communication error but cannot handle it (for example, if the request is to read a non–existent coil or register), the slave will return an exception response informing the master of the nature of the error.

The exception response message has two fields that differentiate it from a normal response:

Function Code Field: In a normal response, the slave echoes the function code of the original query in the function code field of the response. All function codes have a Most-Significant Bit (MSB) of 0 (their values are all below 80 hexadecimal). In an exception response, the slave sets the MSB of the function code to 1. This makes the function code value in an exception response exactly 80 hexadecimal higher than the value would be for a normal response.

With the function code's MSB set, the master's application program can recognize the exception response and can examine the data field for the exception code.

Data Field: In a normal response, the slave may return data or statistics in the data field (any information that was requested in the query). In an exception response, the slave returns an exception code in the data field. This defines the slave condition that caused the exception.

TABLE 9. EXCEPTION CODES

Code	Name	Meaning
01	ILLEGAL FUNCTION	The function code received in the query is not an allowable action for the slave. Command Not Supported: (Only Command 03: Read Holding Registers and Command 01: Read single Coil are supported).
02	ILLEGAL DATA ADDRESS	The data address received in the query is not an allowable address for the slave. The starting address is less than the MODBUS Offset Address (0 or 40001). The starting address is less than 1024 or greater than 8188 (not including the Offset Address).
03	ILLEGAL DATA VALUE	A value contained in the query data field is not an allowable value for the slave. Register Quantity Error: Number of Registers to Read is equal to 0 or greater than 124.
04	SLAVE DEVICE FAILURE	An unrecoverable error occurred while the slave was attempting to perform the requested action. Command Length Error: Input Command does not have enough characters.
05	ACKNOWLEDGE	The slave has accepted the request and is processing it, but a long duration of time will be required to do so. This response is returned to prevent a timeout error from occurring in the master. Data Not Ready for one of the systems tanks or sensors. Data is asked for from a MODBUS location that has no tanks or sensors. (Systems with 8-channel tank cards have different addressing then systems with 4-channel cards)
09	MODBUS NOT ACTIVE	The MODBUS option password has not been set on the PROTEUS-K or PROTEUS-X unit.

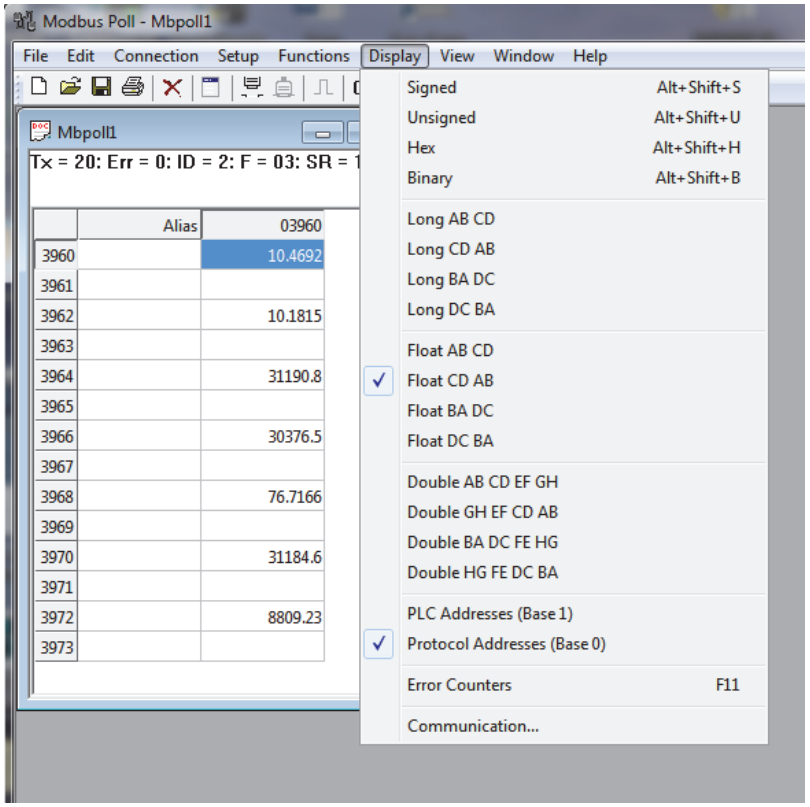
Note: Checksum Errors (MODBUS command has a CRC checksum failure) will not respond with an exception code.

4. TROUBLESHOOTING NOTES

Polling the PROTEUS unit returns 0x0d, 0x0a, 0x03 — Make sure that the Comm ports mode is set for MODBUS not REMOTE mode. (PROTEUS-B)

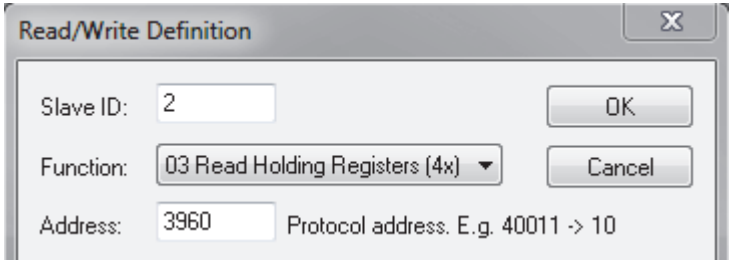
Polling the PROTEUS unit returns 0x02, 0x83, 0x02 — 0x02=MODBUS address, 0x83=exception error on a Read Holding Register command, 02=Exception code. Check 'Exception Responses' (above)

Polling the PROTEUS unit returns data that is close but not exactly the same as the ATG's data readings



Check to see if:

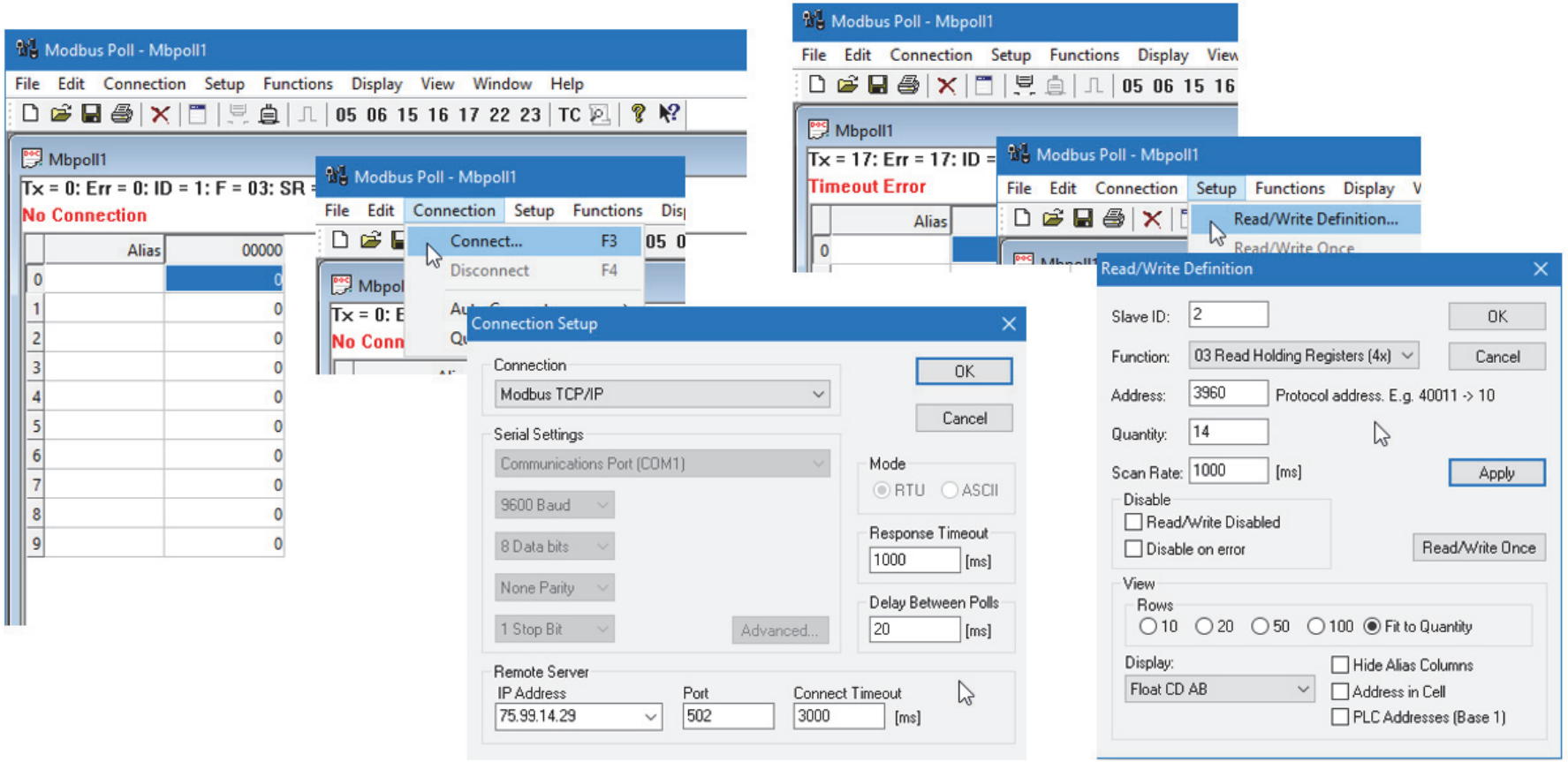
1) Make sure that the controller is sending the correct starting address. On MODBUS Poll example, setting the PLC Address(Base 1) will send the programmed address Plus+1 giving the wrong result.
2) The data's type (Long, Float, Double, Binary,etc.) is correct.
3) The data's byte order is correct (Float AB CD, Float CD AB).
4) Is the MODBUS Password set and correct.



Polling the PROTEUS unit returns no data. — Make sure that the MODBUS address is correct and that the Function is set to '03 Reading Holding Register' Make sure the MODBUS Offset is correct on the starting address (e.g. Offset 0 = 3960, Offset 40001 = 43961)

5. MODBUS TOOLS - MODBUS POLL EXAMPLE SCREENS

MODBUS TOOLS



MODBUS TOOLS