MODBUS/SERIAL INTEGRATION KIT FOR PAC PROJECT

Form 1697-111010—October 2011

OPTO 22

43044 Business Park Drive • Temecula • CA 92590-3614
Phone: 800-321-OPTO (6786) or 951-695-3000
Fax: 800-832-OPTO (6786) or 951-695-2712
www.opto22.com

Product Support Services
800-TEK-OPTO (835-6786) or 951-695-3080
Fax: 951-695-3017
Email: support@opto22.com
Web: support.opto22.com

Modbus/Serial Integration Kit for PAC Project
Form 1697-111010—October 2011

The information in this manual has been checked carefully and is believed to be accurate; however, Opto 22 assumes no responsibility for possible inaccuracies or omissions. Specifications are subject to change without notice.

Opto 22 warrants all of its products to be free from defects in material or workmanship for 30 months from the manufacturing date code. This warranty is limited to the original cost of the unit only and does not cover installation, labor, or any other contingent costs. Opto 22 I/O modules and solid-state relays with date codes of 1/96 or later are guaranteed for life. This lifetime warranty excludes reed relay, SNAP serial communication modules, SNAP PID modules, and modules that contain mechanical contacts or switches. Opto 22 does not warrant any product, components, or parts not manufactured by Opto 22; for these items, the warranty from the original manufacturer applies. These products include, but are not limited to, OptoTerminal-G70, OptoTerminal-G75, and Sony Ericsson GT-48; see the product data sheet for specific warranty information. Refer to Opto 22 form number 1042 for complete warranty information.

Wired+Wireless controllers and brains and N-TRON wireless access points are licensed under one or more of the following patents: U.S. Patent No(s). 5282222, RE37802, 6963617; Canadian Patent No. 2064975; European Patent No. 1142245; French Patent No. 1142245; British Patent No. 1142245; Japanese Patent No. 2002535925A; German Patent No. 60011224.

Table of Contents

1: Getting Started

The Modbus/Serial Integration Kit for PAC Project™ (Part # PAC-INT-MBSER) allows Opto 22 SNAP PAC controllers, using PAC Control™, to connect via RS-232 or RS-485 to a Modbus device and communicate using the Modbus RTU or ASCII protocol.

The Integration Kit contains:

- A set of PAC Control master subroutines that are added to a strategy to enable an Opto 22 SNAP PAC controller to communicate as a Modbus master
- An example Modbus slave strategy containing the slave chart MB_Slave_Serial that is imported into a strategy to enable an Opto 22 controller to communicate as a Modbus slave

Both the master subroutines and the slave chart transmit message strings as specified in the *Modbus Application Protocol Specification v1.1a* and *Modbus over Serial Line Specification & Implementation Guide v1.0*. Both guides are available on the web at http://Modbus-IDA.org.

The master subroutines and slave strategy transmit and receive messages using Modbus standard register, input and coil numbers. The data in the PAC controller is stored in numeric tables.

This manual assumes that you understand how to use PAC Control, Modbus/Serial, and the Modbus device to be used.

This chapter includes the following topics:

Understanding Modbus Serial Protocol

This toolkit assumes that you are knowledgeable about Modbus serial protocol addressing, and register, coil, and input numbering. Even for those who are experienced, we highly recommend reading the *Modicon Modbus Protocol Reference Guide*, which is available at this link:

http://www.modbus.org/docs/PI_MBUS_300.pdf

We especially recommend the following two sections of the Modbus guide:

- *Chapter 2: Data and Control Functions* of the Modicon guide explains the key subtleties of register, coil, and input numbering/naming as opposed to register, coil, and input addressing. This helps eliminate a common point of confusion, even for those who are experienced with Modbus serial protocol.
- *Appendix A: Exception Responses* of the Modicon guide discusses the possible exception codes a Modbus device can reply with and what the codes mean. This helps when diagnosing communication problems.

What is Required

Before including the subroutines in your strategy, you will need a PC with PAC Project 8.1a (Basic or Pro) or newer.

Protocols Supported

The following protocols are supported. They are set by the Comm Mode subroutine parameter.

Comm Mode	Modbus Serial Protocol
0	RTU
1	ASCII

Modbus/Serial Functions Supported

The following Modbus/Serial function codes are supported by the master subroutines. These function codes are also supported by the slave chart.

Modbus/Serial Function Code	Name	PAC Control Subroutine
01	Read Coils	MBMaster Read Coil Status
02	Read Discrete Inputs	MBMaster Read Input Status
03	Read Holding Registers	MBMaster Read Holding Registers
04	Read Input Registers	MBMaster Read Input Registers
05	Write Single Coil	MBMaster Force Single Coil

Modbus/Serial Function Code	Name	PAC Control Subroutine
06	Write Single Register	MBMaster Preset Single Register
08	Diagnostics	MBMaster Diagnostics
15	Write Multiple Coils	MBMaster Force Multiple Coils
16	Write Multiple Registers	MBMaster Preset Multiple Registers
22	Mask Write 4x Register	MBMaster Mask_Write_4x Register
23	Read/Write Multiple Registers	MBMaster Read Write Holding Registers

Data Types Supported in the Input and Holding Registers

The following data types are supported for input and holding registers. These data types are set by the Data Type subroutine parameter in the master subroutines and the nMB_Data_Type variable used in the slave chart:

Value	Data Type
0	16-bit unsigned (Modbus standard and default)
1	16-bit signed
2	Floating point (Uses two registers)
3	Floating point (Swapped. Uses two registers.)
4	32-bit signed (Uses 2 registers)
5	32-bit signed (Swapped. Uses 2 registers.)

NOTE: Most Modbus devices store 32-bit data values in two consecutive 16-bit registers. However, Opto 22 SNAP PAC controllers store 32-bit data values in individual table elements because tables support full 32-bit data values.

When accessing 32-bit data in most Modbus protocol devices, the data is stored in two consecutive 16-bit registers. Data that is 16 bits is sometimes referred to as a *word*, just as 8-bit data is referred to as a *byte*.

Modbus serial protocol messages treat each 16-bit register, or word, as 2 bytes with the high order byte coming before the low order byte. However, when device manufactures started supporting 32-bit data in standard Modbus serial messages, there was not a standard regarding the order of 16-bit registers (words) in the message when it comes to 32-bit data. Because of this, some Modbus protocol devices put the bytes of the high order register (word) first, and the low order register (word) second. Other devices do just the opposite.

In order to provide flexibility when communicating with both types of devices, the PAC Modbus Serial toolkit supports *word swapping* when using 32-bit data types. We normally send the high-order word before the low order word in the message. If the data is not correct, it may be because the word order is backwards compared to your Modbus device. If this is the case, you can simply change the word order by selecting the appropriate swapped data type.

For example, if you are dealing with floating point data using Data Type 2, you could try Data Type 3 in order to swap the order of the 16-bit words in the message.

Important Note for Users Upgrading from Version 8.1 (or Earlier) of this Toolkit

In prior versions of the toolkit (versions 8.1 and earlier), all Register data was handled with either a float table (in the case of Read Holding Registers, Read Input Registers, Preset Multiple Registers, and Read Write Holding Registers) or a float variable (in the case of Preset Single Register) regardless of the value of the Data Type parameter.

Starting with version R8.2a of the toolkit, the table and variable data types now correctly match the Data Type parameter. This impacts how the subroutines are called by the strategy and how the strategy interacts with the data tables.

When calling the subroutine, you will now need to pass both float and integer tables (or float and integer variables). The subroutine will know which to use based on the value of the Data Type parameter. For example, if using integer data, you still have to pass the float table (or variable) even though it won't be used (and vice versa). The simplest thing to do is to just configure the extra table as having a length of 1 so it does not take up too much room in the controller.

You will also need to make sure that your strategy interacts with the correct data tables. When using Data Types 2 or 3, which are both float data types, your strategy will need to interact with the appropriate float table. When using Data Types 0, 1, 4, or 5, which are all integer data types, your strategy will need to interact with the appropriate integer table.

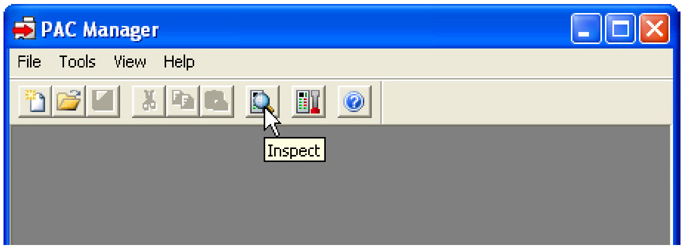
Installing the Integration Kit

To install the integration kit on your computer, unzip the ModbusSerialPACv8.1d.zip file to your C: drive. The expanded files will be placed automatically in C:\ModbusSerialPAC.

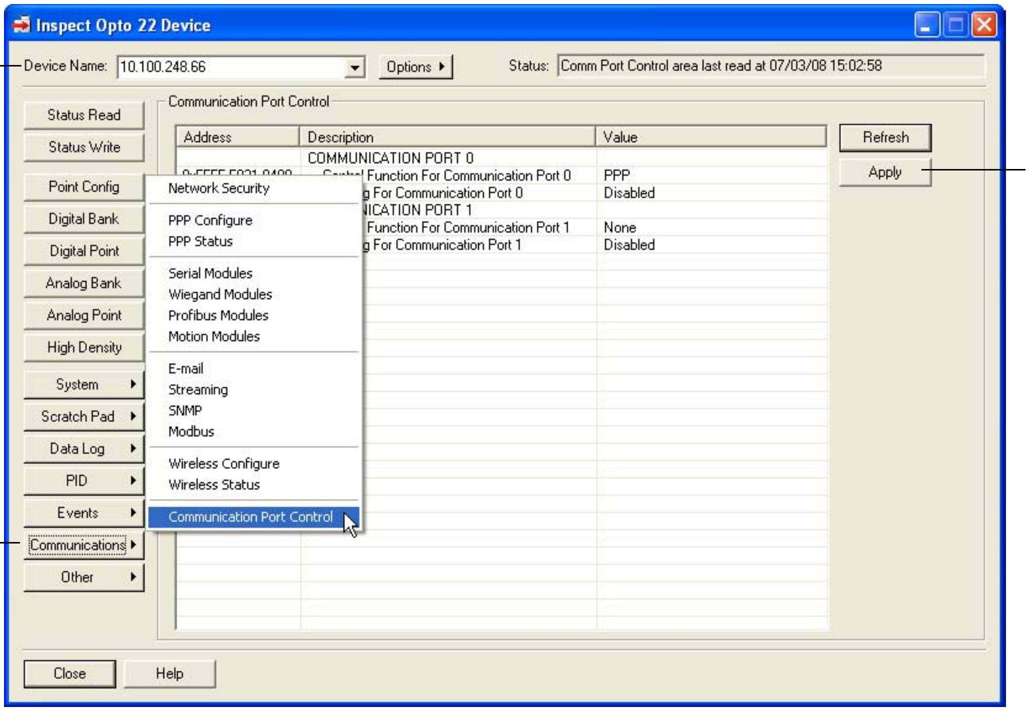
Using the Controller's Serial Port

If you are using the controller's built-in serial port with the Modbus/Serial Integration Kit, rather than using a Serial Communication Module (SCM), you must open PAC Manager to configure the control function for that port, and then store the configuration to flash.

1. On the PC, choose Start > Programs > Opto 22 > PAC Project > PAC Manager.
2. Click the Inspect button, or choose Tools > Inspect to open the Inspect dialog box.



3. Choose the controller.
4. Click the Communications button and choose Communication Port Control from the pop-up menu.



Controller

Device Name: 10.100.248.66

Status: Comm Port Control area last read at 07/03/08 15:02:58

Communication Port Control

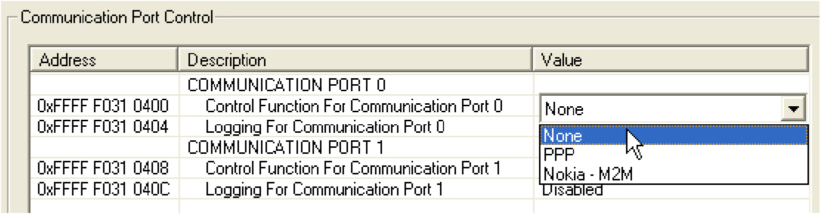
Address	Description	Value
	COMMUNICATION PORT 0	
0xFFFF F031 0400	Control Function For Communication Port 0	PPP
	g For Communication Port 0	Disabled
	ICATION PORT 1	
	Function For Communication Port 1	None
	g For Communication Port 1	Disabled

Apply button

Communications button

The Inspect window shows the possible ports and settings.

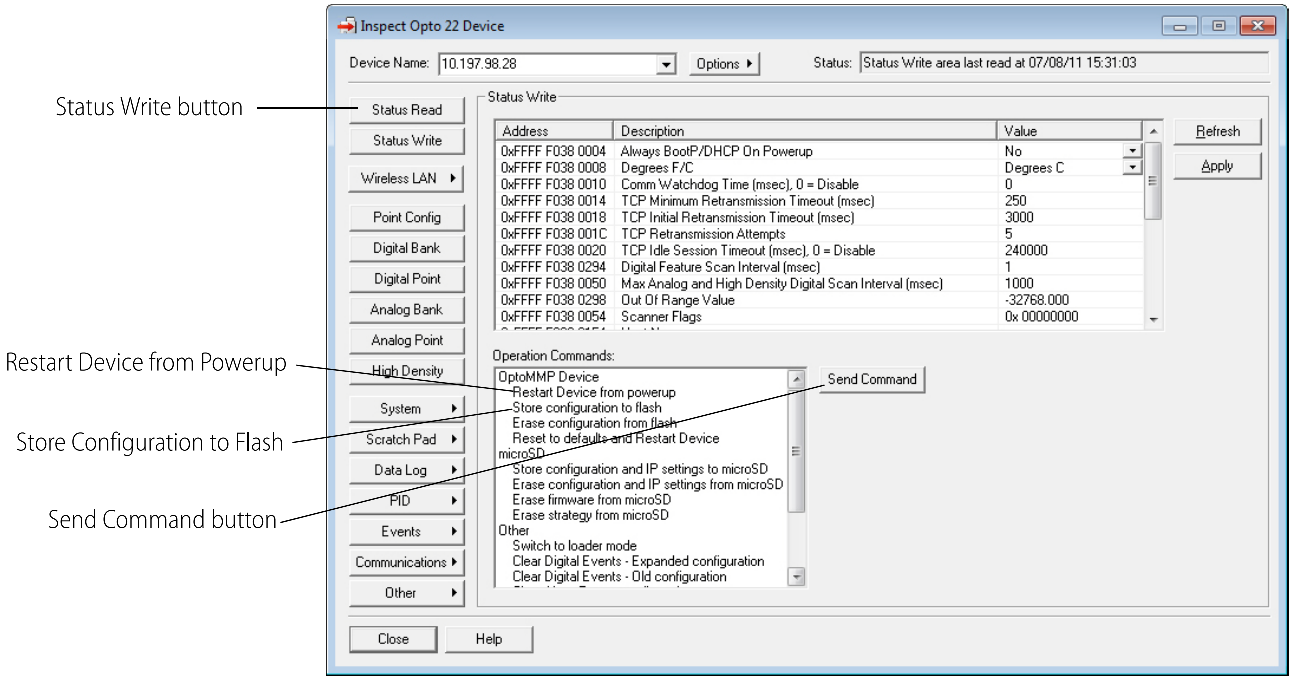
5. In the Value field for Control Function for the serial port you will be using with the toolkit, choose None from the drop-down list.


Communication Port Control

Address	Description	Value
	COMMUNICATION PORT 0	
0xFFFF F031 0400	Control Function For Communication Port 0	None
0xFFFF F031 0404	Logging For Communication Port 0	
	COMMUNICATION PORT 1	
0xFFFF F031 0408	Control Function For Communication Port 1	
0xFFFF F031 040C	Logging For Communication Port 1	Disabled

6. If you are using the SNAP-PAC-S2 controller, in the Value field for Mode for Communication Port, choose the appropriate RS232/RS485 configuration from the drop-down list.
7. Click the Apply button.

8. Click the Status Write button.



Status Write button

Restart Device from Powerup

Store Configuration to Flash

Send Command button

Inspect Opto 22 Device

Device Name: 10.197.98.28

Status: Status Write area last read at 07/08/11 15:31:03

Status Write

Address	Description	Value
0xFFFF F038 0004	Always BootP/DHCP On Powerup	No
0xFFFF F038 0008	Degrees F/C	Degrees C
0xFFFF F038 0010	Comm Watchdog Time (msec), 0 = Disable	0
0xFFFF F038 0014	TCP Minimum Retransmission Timeout (msec)	250
0xFFFF F038 0018	TCP Initial Retransmission Timeout (msec)	3000
0xFFFF F038 001C	TCP Retransmission Attempts	5
0xFFFF F038 0020	TCP Idle Session Timeout (msec), 0 = Disable	240000
0xFFFF F038 0294	Digital Feature Scan Interval (msec)	1
0xFFFF F038 0050	Max Analog and High Density Digital Scan Interval (msec)	1000
0xFFFF F038 0298	Out Of Range Value	-32768.000
0xFFFF F038 0054	Scanner Flags	0x 00000000

Options
Status Read
Status Write
Wireless LAN
Point Config
Digital Bank
Digital Point
Analog Bank
Analog Point
High Density
System
Scratch Pad
Data Log
PID
Events
Communications
Other
Refresh
Apply
Operation Commands:
OptoMMP Device
Restart Device from powerup
Store configuration to flash
Erase configuration from flash
Reset to defaults and Restart Device
microSD
Store configuration and IP settings to microSD
Erase configuration and IP settings from microSD
Erase firmware from microSD
Erase strategy from microSD
Other
Switch to loader mode
Clear Digital Events - Expanded configuration
Clear Digital Events - Old configuration
Send Command
Close
Help

9. Select Store Configuration to Flash.
10. Click the Send Command button.
11. After the command completes, click the OK button on the pop-up window.
12. Select Restart Device from Powerup.
13. Click the Send Command button.

For more information, see the *PAC Manager User's Guide*, form 1704.

Running the Example Strategies

The toolkit includes example strategies to demonstrate how to use the master subroutines and the slave chart in a PAC Control strategy. Before including the subroutines in your own strategy, we recommend that you first run the example strategy and pay special attention to the strategy logic and the configuration of variables.

To run the example master strategy, start PAC Control, and then open the strategy file named MBMasterSerial. To run the example slave strategy, open the strategy named MBSlaveSerial.

Using Communication Handles

Be sure to use separate communication handles for each of the controller's physical serial ports used with the subroutines. It is best to have only one communication handle per port. This prevents two charts from trying to access the same communication port at the same time.

To configure a communication handle for a serial (SNAP-SCM) module or a controller serial port, use the following information:

- The name of the communication handle variable in the example strategies is chModbus
- Set the communication handle parameters in the string variable named sMB_Port_Configuration_String
- Search the example strategy for chModbus to find out where this is used.

Communication Handle Syntax

Controller Serial Port

When using a controller serial port, the communication handle syntax is:

```
ser:<port number>,<baud rate>,<parity>,<data bits>,<stop bits>
```

Example for a controller port:

```
ser:0,115200,n,8,1
```

Serial Module

When using a serial module, the communication handle syntax is:

```
tcp:<IP address>:<port>
```

Example for a serial module:

```
tcp:10.192.255.185:22500
```

For more examples and details on communication handle parameters, see chapter 10 of the *PAC Control User's Guide*, form 1700.

Using Data Addresses in Modbus

When used as Modbus terminology, the term *address* can be confusing. In Modbus, *addresses* are always zero-based, which means that the first address is 0, not 1.

For example:

- The coil addresssed as coil 0000 in the data address field of a Modbus message is called *coil 1* in a programmable controller.
- Address 0000 in the data address field of the message is holding register 40001. Since the function code field already specifies a *holding register* operation, the *4XXXX* reference is implicit.

PAC Display Example

Two PAC Display™ example projects are included in the ZIP file to show what can be done using PAC Display. While not necessary components of the toolkit, you can use the ModbusSerialMaster and ModbusSerialSlave projects to check PAC Control master and slave connections. For more information on using PAC Display, see the *PAC Display User's Guide*, form 1702.

2: Using the Master Subroutines

This chapter includes the following topics:

Adding Master Subroutines

The Modbus master subroutines allow an Opto 22 controller to function as a Modbus master device. Each master subroutine in the integration kit supports one Modbus function code and can function independently of the other subroutines. Therefore, you need only use the subroutines for the Modbus functions that you require. For more information about subroutines, see the *PAC Control User's Guide*.

When you decide which subroutines you need, include them in your strategy as follows:

1. Start PAC Control in Configure Mode and open the strategy that you intend to use with the integration kit.
2. Select Configure > Subroutines Included to open the Subroutine Files dialog.
3. Click the Add button and use the browser to select each subroutine file (.ISB extension) you wish to include in your strategy from the folder C:\ModbusSerialPAC\Subs.
4. Click OK.

   The subroutines appear in the Subroutines Included folder and are ready to be used in your strategy.

Data Tables Used By Master Subroutines

These are the names used for Modbus data tables in the example Master strategy:

- ftMB_Holding_Registers_4X (type is Float table)
- ftMB_Input_Registers_3X (type is Float table)
- ntMB_Coils_0X (type is Integer 32 table)
- ntMB_Holding_Registers_4X (type is Float table)
- ntMB_Input_Registers_3X (type is Float table)
- ntMB_Inputs_1X (type is Integer 32 table)

You can name the tables however you want because the names of the tables are passed to the subroutines.

You may need to adjust the lengths of these tables to accommodate the amount of Modbus data and the register, coil, and input numbers expected to be accessed by the Modbus master device.

Use strategy logic to populate data in or retrieve data from these tables from the most recent Modbus message received.

See "Important Note for Users Upgrading from Version 8.1 (or Earlier) of this Toolkit" on page 4.

Operation Mode Details for Master Subroutines

Version 8.1d of the Modbus Serial PAC Control toolkit added a new feature called the Operation Mode. In the Master subroutines, this is implemented by an additional passed parameter called Master Register.

If you want to use the same register, coil, or input numbers in the master and slave, use a value of -1 for the Master Register parameter. This is how the original versions of this toolkit worked prior to adding the Operation Mode feature.

When the Master Register parameter is greater than or equal to 0, it designates the starting table index used in the Opto 22 SNAP PAC controller (the Master).

All subroutines that are passed a table to read or write Modbus data use the new parameter. The following subroutines do not support the new Operation Mode feature:

- MBMaster Force Single Coil
- MBMaster Mask Write 4X Register
- MBMaster Preset Single Register

If you want to use different register, coil, or input numbers in the master and slave, then use the specific starting table index in the master as the Master Register parameter and specify the slave starting register, coil, or input number in the Slave Register parameter.

This method is useful when the data in the slave is offset to a high register, coil, or input number. In this case, it allows using much smaller tables in the master. This method can also be used when accessing data in multiple slaves and consolidating it into one set of tables in the master.

NOTE: The parameter formerly named Starting_Address has been renamed to Slave_Register.

Examples Using the Master Register Parameter

Case 1

Subroutines Related to Coils and Inputs

For Coils and Inputs, there is always a one-to-one correlation between the number or coils or inputs in the slave and the number of table elements used for the data in the master.

The Master Register parameter only affects the starting coil (or input) number used in the master data tables. This corresponds to the data table index number.

Example 1A: When the Master Register parameter is -1, the subroutine uses the value of the Start Coil (or Start Input) parameter as the starting Coil (or Input) number in the slave and also as the starting table index in the master.

Parameter	Value
Master Register	-1
Slave Register	19
Qty of Coils	3

Master Table Index		Slave Coil Number
19	←	19
20	←	20
21	←	21

Example 1B: When the Master Register parameter is greater than or equal to 0, the subroutine uses the value of the Start Coil (or Start Input) parameter as the starting Coil (or Input) number in the Slave and it uses the value of the Master Register parameter as the starting table index in the master.

Parameter	Value
Master Register	1
Slave Register	101
Qty of Inputs	3

Master Table Index		Slave input Number
1	←	101
2	←	102
3	←	103

Case 2

Subroutines related to Input Registers and Holding Registers when the Data Type Parameter is 0 or 1 (both 16-bit data types)

For Input Registers and Holding Registers, when using 16-bit data types, there is always a one-to-one correlation between the number or registers in the slave and the number of table elements used for the data in the master.

The Master Register parameter only affects the starting register number used in the master data tables. This corresponds to the data table index number.

Example 2A: When the Master Register parameter is -1, the subroutine uses the value of the Start Register parameter as the starting register number in the slave and also as the starting table index in the master.

Parameter	Value	Comment
Master Register	-1	
Slave Register	7001	
Qty of H Registers	3	
Data Type	0 or 1	data type is 16-bit

Master Table Index		Slave Register Number
7001	→	7001
7002	→	7002
7003	→	7003

Example 2B: When the Master Register parameter is greater than or equal to 0, the subroutine uses the value of the Start Register parameter as the starting register number in the Slave and it uses the value of the Master Register parameter as the starting table index in the master.

Parameter	Value	Comment
Master Register	99	
Slave Register	7001	
Qty of I Registers	3	
Data Type	0 or 1	data type is 16-bit

Master Table Index		Slave Register Number
99	←	7001
100	←	7002
101	←	7003

Case 3

Subroutines related to Input Registers and Holding Registers when the Data Type parameter is 2, 3, 4, or 5 (all 32-bit data types)

NOTE: Opto 22 SNAP- PAC tables support 32-bit data; tables start with index 0. A table with length of 10 has indexes 0 through 9.

For Input Registers and Holding Registers, when using 32-bit data types, there is always a *two*-to-one correlation between the number of (16-bit) registers in the slave and the number of (32-bit) table elements used for the data in the master. This is because most slave devices store data in 16-bit registers. Consequently, 32-bit data in these devices are stored in two consecutive 16-bit registers.

For 32-bit data, the Master Register parameter affects not only the starting register number used in the master data tables, but also the quantity of registers accessed in the slave.

Example 3A: When the Master Register parameter is -1, the subroutine uses the value of the Start Register parameter as the starting register number in the slave and also as the starting table index in the master.

In addition, the Qty of H Registers (or Qty of I Registers) parameter determines the quantity of 16-bit registers read from or written to the slave. The number of 32-bit registers in the master will be half the quantity of this parameter.

The Qty of H Registers (or Qty of I Registers) parameter *must be an even number.*

The Start Register parameter *must be an odd number.*

The Start Register value will determine the first table index used for the 32-bit data in the master. Additional 32-bit values will be put into subsequent *odd indexes* of the table so that the table indexes in the master will match the first register number for each set of two consecutive 16-bit registers in the slave. All even table indexes are unused.

Parameter	Value	Comment
Master Register	-1	
Slave Register	7001	must be an odd number
Qty of I Registers	4	must be an even number
Data Type	2,3,4 or 5	data type is 32-bit

Master Table Index		Slave Register Number
7001	←	7001
		7002
7003	←	7003
		7004

Example 3B: When the Master Register parameter is greater than or equal to 0, the subroutine uses the value of the Start Register parameter as the starting register number in the Slave and it uses the value of the Master Register parameter as the starting table index in the master.

In addition, the Qty of H Registers (or Qty of I Registers) parameter determines the quantity of 32-bit data values you want to read from or write to the slave and the number of 32-bit table elements used in the master. However, the Modbus serial protocol has no mechanism for identifying 32-bit data in the messages, so the subroutine requests twice the quantity of registers because the data in the slave is assumed to be 16-bit.

The Master Register value will determine the first table index used for the 32-bit data in the master. Additional 32-bit values will be put into consecutive indexes of the table so that there will not be any gaps in the table.

Parameter	Value	Comment
Master Register	8	
Slave Register	7001	must be an odd number
Qty of H Registers	3	Qty of 32-bit values
Data Type	2,3,4 or 5	data type is 32-bit

Master Table Index		Slave Register Number
8	→	7001
		7002
9	→	7003
		7004
10	→	7005
		7006

Configuration of Subroutines

The tables on the following pages list the parameters for each function code and describe the type of data for each parameter:

Function Code	Page
MBMaster Diagnostics	16
MBMaster Force Multiple Coils	18
MBMaster Force Single Coil	19
MBMaster Mask Write 4x Register	20
MBMaster Preset Multiple Registers	21
MBMaster Preset Single Register	23
MBMaster Read Coil Status	24
MBMaster Read Holding Registers	25
MBMaster Read Input Registers	27
MBMaster Read Input Status	29
MBMaster Read Write Holding Registers	30

Starting Register, Coil, and Input *numbers* are from 1 to 65536. However, Modbus *addresses* start at 0. In order to correct for this offset, the subroutines reduce the Starting Register, Coil, and Input number by 1 in the packet sent to the Modbus slave.

Modbus Register 1 = Opto 22 Index 1 = Modbus *Address* 0

Modbus Register 100 = Opto 22 Index 100 = Modbus *Address* 99

MBMaster Diagnostics

Passed Parameter	Value Type	Value	Description
Slave Address	Integer 32 Variable	1–255	Address of Modbus Slave Device
Register	Integer 32 Variable	0–65535	Register number to write to in the slave
Data(Send)	Integer 32 Variable	0–65535	Data required to be sent for some sub-functions
Data(Rec)	Integer 32 Variable	See “MBMaster Diagnostics Sub-function Codes” on page 17	Data returned by some sub-functions
Comm Mode	Integer 32 Variable	0	Modbus RTU Protocol
		1	Modbus ASCII Protocol
Com Handle	Communication Handle	For details, see “Using Communication Handles” on page 6.	Serial port communication parameters on the PAC controller (master).
Wait Time(s)	Float Variable		Timeout value in seconds that the master subroutine will wait for the slave to respond
Return Status	String Variable		Status of executing the logic in the subroutine
		No Port	Not able to open port
		Timeout	No response within time limit
		Too Many Characters	More then 260 characters
		Address Mismatch	Send and receive address do not match
		CRC or LRC Mismatch	Checksum error
		Sub-Function Reserved	Not supported code
		Response Mismatch	Send and Receive packet mismatch
		Function and Exception code	Error from PDU
		OK	Success
Put Status In	Integer 32 Variable		Status of calling the subroutine
		0	Success
		-67	Out of memory
		-69	Null object error

MBMaster Diagnostics Sub-function Codes

Sub-Function	Name	Data (Send)	Data(Rec)
0	Return Query Data	Any	0
1	Restart Communication Option	0 or 1 = Clear Log	0
2	Return Diagnostic Register	0	Register data
3	Change ASCII Input Delimiter	Decimal Value of Character	0
4	Force Listen Only Mode	0	0
5	Reserved		
6	Reserved		
7	Reserved		
8	Reserved		
9	Reserved		
10	Clear Counters and Diagnostic Register	0	0
11	Return Bus Message Count	0	Message Ct.
12	Return Bus Communication Error Count	0	Error Ct.
13	Return Bus Exception Error Count	0	Error Ct.
14	Return Slave Message Count	0	Message Ct.
15	Return Slave No Response Count	0	No Response Ct.
16	Return Slave NAK Count	0	NAK Ct.
17	Return Slave Busy Count	0	Busy Ct.
18	Return Bus Character Overrun Count	0	Overrun Ct.
19	Reserved		
20	Clear Overrun Counter And Flag	0	0
21	Reserved	0	

MBMaster Force Multiple Coils

Passed Parameter	Value Type	Value	Description
Slave Address	Integer 32 Variable	1 - 255	Address of Modbus Slave Device
Start Coil	Integer 32 Variable	1 - 65536	First coil number to force in the slave
Quantity of Coil	Integer 32 Variable	1 - 1968	The quantity of coils to force in the slave
MB Coils 0X	Integer 32 Table		PAC Control strategy table containing the coil data
Comm Mode	Integer 32 Variable	0	Modbus RTU Protocol
		1	Modbus ASCII Protocol
Com Handle	Communication Handle	For details, see "Using Communication Handles" on page 6.	Serial port communication parameters on the PAC controller (master).
Wait Time(s)	Float Variable		Timeout value in seconds that the master subroutine will wait for the slave to respond
Master Register	Integer 32 Variable	-1	When Master Register is -1, it uses the value of Start Coil parameter as the starting Coil number in the slave and also as the starting table index in the PAC (master). See "Examples Using the Master Register Parameter" on page 11.
		>= 0	When Master Register is greater than or equal to 0, it uses the value of Start Coil as the starting Coil number in the Slave and it uses the value of Master Register as the starting table index in the PAC (master). See "Examples Using the Master Register Parameter" on page 11.
Return Status	String Variable		Status of executing the logic in the subroutine
		No Port	Not able to open Port
		Timeout	No response within time limit (Wait Time parameter)
		Too Many Characters	More then 260 characters
		Address Mismatch	Send and receive address do not match
		CRC or LRC Mismatch	Checksum error
		Function and Exception code	Error returned by the slave device. For details, see Chapter 4 on page 39.
		Invalid Table Length	Used an index greater than the last index of the data table.
		OK	Success
Put Status In	Integer 32 Variable		Status of calling the subroutine
		0	Success
		-67	Out of memory
		-69	Null object error

MBMaster Force Single Coil

Passed Parameter	Value Type	Value	Description
Slave Address	Integer 32 Variable	1 - 255	Address of Modbus Slave Device
Coil	Integer 32 Variable	1 - 65536	Coil number to force in the slave
Coil State	Integer 32 Variable	0	Turns coil OFF
		1	Turns coil ON
Comm Mode	Integer 32 Variable	0	Modbus RTU Protocol
		1	Modbus ASCII Protocol
Com Handle	Communication Handle	For details, see "Using Communication Handles" on page 6.	Serial port communication parameters on the PAC controller (master).
Wait Time(s)	Float Variable		Timeout value in seconds that the master subroutine will wait for the slave to respond
Return Status	String Variable		Status of executing the logic in the subroutine
		No Port	Not able to open Port
		Timeout	No response within time limit (Wait Time parameter)
		Too Many Characters	More then 260 characters
		Address Mismatch	Send and receive address do not match
		CRC or LRC Mismatch	Checksum error
		Function and Exception code	Error returned by the slave device. For details, see Chapter 4 on page 39.
		OK	Success
Put Status In	Integer 32 Variable		Status of calling the subroutine
		0	Success
		-67	Out of memory
		-69	Null object error

MBMaster Mask Write 4x Register

NOTE: This subroutine will work with 16-bit integer Holding Registers only (signed or unsigned).

Passed Parameter	Value Type	Value	Description
Slave Address	Integer 32 Variable	1 - 255	Address of Modbus Slave Device
Ref Register	Integer 32 Variable	1 - 65536	Register number to write to in the slave
AND Mask	Integer 32 Variable	0 - 65536	
OR Mask	Integer 32 Variable	0 - 65536	
Comm Mode	Integer 32 Variable	0	Modbus RTU Protocol
		1	Modbus ASCII Protocol
Com Handle	Communication Handle	For details, see "Using Communication Handles" on page 6.	Serial port communication parameters on the PAC controller (master).
Wait Time(s)	Float Variable		Timeout value in seconds that the master subroutine will wait for the slave to respond
Return Status	String Variable		Status of executing the logic in the subroutine
		No Port	Not able to open Port
		Timeout	No response within time limit (Wait Time parameter)
		Too Many Characters	More then 260 characters
		Address Mismatch	Send and receive address do not match
		CRC or LRC Mismatch	Checksum error
		Function and Exception code	Error returned by the slave device. For details, see Chapter 4 on page 39.
		OK	Success
Put Status In	Integer 32 Variable		Status of calling the subroutine
		0	Success
		-67	Out of memory
		-69	Null object error

MBMaster Preset Multiple Registers

Passed Parameter	Value Type	Value	Description
Slave Address	Integer 32 Variable	1 - 255	Address of Modbus Slave Device
Start Register	Integer 32 Variable	1 - 65536	First register number to write to in the slave
Qty of Registers	Integer 32 Variable	1 - 120	The quantity of registers to write to the slave.
		1 - 60	When using Data Type of 2, 3, 4, or 5 (32-bit data) and Master Register value greater than or equal to 0, the limit is 60. See “Examples Using the Master Register Parameter” on page 11.
MB H Reg4X Int	Integer 32 Table		PAC Control strategy table containing the Holding Register data when using integer data (data types 0, 1, 4, 5) See “Important Note for Users Upgrading from Version 8.1 (or Earlier) of this Toolkit” on page 4
MB H Reg4X Float	Float Table		PAC Control strategy table containing the Holding Register data when using float data (data types 2, 3) See “Important Note for Users Upgrading from Version 8.1 (or Earlier) of this Toolkit” on page 4
Comm Mode	Integer 32 Variable	0	Modbus RTU Protocol
		1	Modbus ASCII Protocol
Com Handle	Communication Handle	For details, see “Using Communication Handles” on page 6.	Serial port communication parameters on the PAC controller (master).
Wait Time(s)	Float Variable		Timeout value in seconds that the master subroutine will wait for the slave to respond
Data Type	Integer 32 Variable	0	16-bit unsigned integer (Modbus standard)
		1	16-bit signed integer
		2	32-bit Float
		3	32-bit Float (swapped)
		4	32-bit signed integer
		5	32-bit signed integer (swapped)
Master Register	Integer 32 Variable	-1	When the Master Register parameter is -1 and the Data Type parameter is 0 or 1 (16-bit data), the subroutine uses the value of the Start Register parameter as the starting register number in the slave and also as the starting table index in the master. See “Examples Using the Master Register Parameter” on page 11.
		>= 0	When the Master Register parameter is greater than or equal to 0 and the Data Type parameter is 0 or 1 (16-bit data), the subroutine uses the value of the Start Register parameter as the starting register number in the Slave and it uses the value of the Master Register parameter as the starting table index in the master. See “Examples Using the Master Register Parameter” on page 11.
			When using Data Type 2, 3, 4, or 5 (32-bit data). See “Case 3” on page 13 for details.

MBMaster Preset Multiple Registers (Continued)

Passed Parameter	Value Type	Value	Description
Return Status	String Variable		Status of executing the logic in the subroutine
		No Port	Not able to open Port
		Timeout	No response within time limit (Wait Time parameter)
		Too Many Characters	More then 260 characters
		Address Mismatch	Send and receive address do not match
		CRC or LRC Mismatch	Checksum error
		Function and Exception code	Error returned by the slave device. For details, see Chapter 4 on page 39.
		Invalid Table Length	Used an index greater than the last index of the data table.
		OK	Success
Put Status In	Integer 32 Variable		Status of calling the subroutine
		0	Success
		-67	Out of memory
		-69	Null object error

MBMaster Preset Single Register

Passed Parameter	Value Type	Value	Description
Slave Address	Integer 32 Variable	1 - 255	Address of Modbus Slave Device
Register	Integer 32 Variable	1 - 65536	Register number to write to in the slave
Reg Value Int	Integer 32 Variable		Value to write to the register when using integer data (data types 0, 1, 4, 5) See "Important Note for Users Upgrading from Version 8.1 (or Earlier) of this Toolkit" on page 4
Reg Value Float	Float Variable		Value to write to the register when using float data (data types 2, 3) See "Important Note for Users Upgrading from Version 8.1 (or Earlier) of this Toolkit" on page 4
Comm Mode	Integer 32 Variable	0	Modbus RTU Protocol
		1	Modbus ASCII Protocol
Com Handle	Communication Handle	For details, see "Using Communication Handles" on page 6.	Serial port communication parameters on the PAC controller (master).
Wait Time(s)	Float Variable		Timeout value in seconds that the master subroutine will wait for the slave to respond
Data Type	Integer 32 Variable	0	16-bit unsigned integer (Modbus standard)
		1	16-bit signed integer
		2	32-bit Float
		3	32-bit Float (swapped)
		4	32-bit signed integer
		5	32-bit signed integer (swapped)
Return Status	String Variable		Status of executing the logic in the subroutine
		No Port	Not able to open Port
		Timeout	No response within time limit (Wait Time parameter)
		Too Many Characters	More then 260 characters
		Address Mismatch	Send and receive address do not match
		CRC or LRC Mismatch	Checksum error
		Function and Exception code	Error returned by the slave device. For details, see Chapter 4 on page 39.
		OK	Success
Put Status In	Integer 32 Variable		Status of calling the subroutine
		0	Success
		-67	Out of memory
		-69	Null object error

MBMaster Read Coil Status

Passed Parameter	Value Type	Value	Description
Slave Address	Integer 32 Variable	1 - 255	Address of Modbus Slave Device
Start Coil	Integer 32 Variable	1 - 65536	First coil number to read from the slave
Quantity of Coils	Integer 32 Variable	1 - 2000	The quantity of coils to read from the slave
Comm Mode	Integer 32 Variable	0	Modbus RTU Protocol
		1	Modbus ASCII Protocol
Com Handle	Communication Handle	For details, see "Using Communication Handles" on page 6.	Serial port communication parameters on the PAC controller (master).
Wait Time (s)	Float Variable		Timeout value in seconds that the master subroutine will wait for the slave to respond
MB Coils 0X	Integer 32 Table		PAC Control strategy table used by the Master to store the coil data
Master Register	Integer 32 Variable	-1	When Master Register is -1, it uses the value of Start Coil parameter as the starting Coil number in the slave and also as the starting table index in the PAC (master). See "Examples Using the Master Register Parameter" on page 11.
		>= 0	When Master Register is greater than or equal to 0, it uses the value of Start Coil as the starting Coil number in the Slave and it uses the value of Master Register as the starting table index in the PAC (master). See "Examples Using the Master Register Parameter" on page 11.
Return Status	String Variable		Status of executing the logic in the subroutine
		No Port	Not able to open Port
		Timeout	No response within time limit (Wait Time parameter)
		Too Many Characters	More then 260 characters
		Address Mismatch	Send and receive address do not match
		CRC or LRC Mismatch	Checksum error
		Function and Exception code	Error returned by the slave device. For details, see Chapter 4 on page 39.
		Invalid Table Length	Used an index greater than the last index of the data table.
		OK	Success
Put Status In	Integer 32 Variable		Status of calling the subroutine
		0	Success
		-67	Out of memory
		-69	Null object error

MBMaster Read Holding Registers

Passed Parameter	Value Type	Value	Description
Slave Address	Integer 32 Variable	1 - 125	Address of Modbus Slave Device
Start Register	Integer 32 Variable	1 - 65536	First register number to read from the slave
Qty of H Register	Integer 32 Variable	1 - 125	The quantity of registers to read from the slave.
		1 - 62	When using Data Type of 2, 3, 4, or 5 (32-bit data) and Master Register value greater than or equal to 0, the limit is 62. See "Examples Using the Master Register Parameter" on page 11.
Comm Mode	Integer 32 Variable	0	Modbus RTU Protocol
		1	Modbus ASCII Protocol
Com Handle	Communication Handle	For details, see "Using Communication Handles" on page 6.	Serial port communication parameters on the PAC controller (master).
Wait Time(s)	Float Variable		Timeout value in seconds that the master subroutine will wait for the slave to respond
MB H Reg4X Int	Integer 32 Table		PAC Control strategy table used by the master to store the Holding Register data when using integer data (data types 0, 1, 4, 5). See "Important Note for Users Upgrading from Version 8.1 (or Earlier) of this Toolkit" on page 4
MB H Reg4X Float	Float Table		PAC Control strategy table used by the master to store the Holding Register data when using float data (data types 2, 3) See "Important Note for Users Upgrading from Version 8.1 (or Earlier) of this Toolkit" on page 4
Data Type	Integer 32 Variable	0	16-bit unsigned integer (Modbus standard)
		1	16-bit signed integer
		2	32-bit Float
		3	32-bit Float (swapped)
		4	32-bit signed integer
		5	32-bit signed integer (swapped)

MBMaster Read Holding Registers (Continued)

Passed Parameter	Value Type	Value	Description
Master Register	Integer 32 Variable	-1	When the Master Register parameter is -1 and the Data Type parameter is 0 or 1 (16-bit data), the subroutine uses the value of the Start Register parameter as the starting register number in the slave and also as the starting table index in the master. See "Examples Using the Master Register Parameter" on page 11.
		>= 0	When the Master Register parameter is greater than or equal to 0 and the Data Type parameter is 0 or 1 (16-bit data), the subroutine uses the value of the Start Register parameter as the starting register number in the Slave and it uses the value of the Master Register parameter as the starting table index in the master. See "Examples Using the Master Register Parameter" on page 11.
			When using Data Type 2, 3, 4, or 5 (32-bit data). See "Case 3" on page 13 for details.
Return Status	String Variable		Status of executing the logic in the subroutine
		No Port	Not able to open Port
		Timeout	No response within time limit (Wait Time parameter)
		Too Many Characters	More then 260 characters
		Address Mismatch	Send and receive address do not match
		CRC or LRC Mismatch	Checksum error
		Function and Exception code	Error returned by the slave device. For details, see Chapter 4 on page 39.
		Invalid Table Length	Used an index greater than the last index of the data table.
		OK	Success
Put Status In	Integer 32 Variable		Status of calling the subroutine
		0	Success
		-67	Out of memory
		-69	Null object error

MBMaster Read Input Registers

Passed Parameter	Value Type	Value	Description
Slave Address	Integer 32 Variable	1 - 255	Address of Modbus Slave Device
Start Register	Integer 32 Variable	1 - 65536	First register number to read from the slave
Qty of I Registers	Integer 32 Variable	1 - 125	The quantity of registers to read from the slave.
		1 - 62	When using Data Type of 2, 3, 4, or 5 (32-bit data) and Master Register value greater than or equal to 0, the limit is 62. See "Examples Using the Master Register Parameter" on page 11.
Comm Mode	Integer 32 Variable	0	Modbus RTU Protocol
		1	Modbus ASCII Protocol
Com Handle	Communication Handle	For details, see "Using Communication Handles" on page 6.	Serial port communication parameters on the PAC controller (master).
Wait Time(s)	Float Variable		Timeout value in seconds that the master subroutine will wait for the slave to respond
MB I Reg3X Int	Integer 32 Table		PAC Control strategy table used by the master to store the Input Register data when using integer data (data types 0, 1, 4, 5) See "Important Note for Users Upgrading from Version 8.1 (or Earlier) of this Toolkit" on page 4
MB I Reg3X Float	Float Table		PAC Control strategy table used by the master to store the Input Register data when using float data (data types 2, 3) See "Important Note for Users Upgrading from Version 8.1 (or Earlier) of this Toolkit" on page 4
Data Type	Integer 32 Variable	0	16-bit unsigned integer (Modbus standard)
		1	16-bit signed integer
		2	32-bit Float
		3	32-bit Float (swapped)
		4	32-bit signed integer
		5	32-bit signed integer (swapped)

MBMaster Read Input Registers (Continued)

Passed Parameter	Value Type	Value	Description
Master Register	Integer 32 Variable	-1	When the Master Register parameter is -1 and the Data Type parameter is 0 or 1 (16-bit data), the subroutine uses the value of the Start Register parameter as the starting register number in the slave and also as the starting table index in the master. See "Examples Using the Master Register Parameter" on page 11.
		>= 0	When the Master Register parameter is greater than or equal to 0 and the Data Type parameter is 0 or 1 (16-bit data), the subroutine uses the value of the Start Register parameter as the starting register number in the Slave and it uses the value of the Master Register parameter as the starting table index in the master. See "Examples Using the Master Register Parameter" on page 11.
			When using Data Type 2, 3, 4, or 5 (32-bit data). See "Case 3" on page 13 for details.
Return Status	String Variable		Status of executing the logic in the subroutine
		No Port	Not able to open Port
		Timeout	No response within time limit (Wait Time parameter)
		Too Many Characters	More then 260 characters
		Address Mismatch	Send and receive address do not match
		CRC or LRC Mismatch	Checksum error
		Function and Exception code	Error returned by the slave device. For details, see Chapter 4 on page 39.
		Invalid Table Length	Used an index greater than the last index of the data table.
		OK"	Success
Put Status In	Integer 32 Variable		Status of calling the subroutine
		0	Success
		-67	Out of memory
		-69	Null object error

MBMaster Read Input Status

Passed Parameter	Value Type	Value	Description
Slave Address	Integer 32 Variable	1 - 255	Address of Modbus Slave Device
Start Input	Integer 32 Variable	1 - 65536	First input number to read from the slave
Quantity of Inputs	Integer 32 Variable	1 - 2000	The quantity of inputs to read from the slave
Comm Mode	Integer 32 Variable	0	Modbus RTU Protocol
		1	Modbus ASCII Protocol
Com Handle	Communication Handle	For details, see "Using Communication Handles" on page 6.	Serial port communication parameters on the PAC controller (master).
Wait Time(s)	Float Variable		Timeout value in seconds that the master subroutine will wait for the slave to respond
MB Inputs 1X	Integer 32 Table		PAC Control strategy table used by the master to store the input data
Master Register	Integer 32 Variable	-1	When Master Register is -1, it uses the value of Start Input parameter as the starting Input number in the slave and also as the starting table index in the PAC (master). See "Examples Using the Master Register Parameter" on page 11.
		>= 0	When Master Register is greater than or equal to 0, it uses the value of Start Input as the starting Input number in the Slave and it uses the value of Master Register as the starting table index in the PAC (master). See "Examples Using the Master Register Parameter" on page 11.
Return Status	String Variable		Status of executing the logic in the subroutine
		No Port	Not able to open Port
		Timeout	No response within time limit (Wait Time parameter)
		Too Many Characters	More then 260 characters
		Address Mismatch	Send and receive address do not match
		CRC or LRC Mismatch	Checksum error
		Function and Exception code	Error returned by the slave device. For details, see Chapter 4 on page 39.
		Invalid Table Length	Used an index greater than the last index of the data table.
		OK	Success
Put Status In	Integer 32 Variable		Status of calling the subroutine
		0	Success
		-67	Out of memory
		-69	Null object error

MBMaster Read Write Holding Registers

Passed Parameter	Value Type	Value	Description
Slave Address	Integer 32 Variable	1 - 255	Address of Modbus Slave Device
R Start Register	Integer 32 Variable	1 - 65536	First register number to read from in the slave
R Qt H Registers	Integer 32 Variable	1 - 118	The quantity of registers to read from the slave.
		1 - 59	When using Data Type of 2, 3, 4, or 5 (32-bit data) and Master Register value greater than or equal to 0, the limit is 59. See "Examples Using the Master Register Parameter" on page 11.
W Start Register	Integer 32 Variable	1 - 65536	First register number to write to in the slave
W Qt H Registers	Integer 32 Variable	1 - 118	The quantity of registers to write to the slave.
		1 - 59	When using Data Type of 2, 3, 4, or 5 (32-bit data) and Master Register value greater than or equal to 0, the limit is 59. See "Examples Using the Master Register Parameter" on page 11.
Comm Mode	Integer 32 Variable	0	Modbus RTU Protocol
		1	Modbus ASCII Protocol
Com Handle	Communication Handle	For details, see "Using Communication Handles" on page 6.	Serial port communication parameters on the PAC controller (master).
MB H Reg4X Int	Integer 32 Table		PAC Control strategy table containing the Holding Register data when using integer data (data types 0, 1, 4, 5) See "Important Note for Users Upgrading from Version 8.1 (or Earlier) of this Toolkit" on page 4
MB H Reg4X Float	Float Table		PAC Control strategy table containing the Holding Register data when using float data (data types 2, 3) See "Important Note for Users Upgrading from Version 8.1 (or Earlier) of this Toolkit" on page 4
Data Type	Integer 32 Variable	0	16-bit unsigned integer (Modbus standard)
		1	16-bit signed integer
		2	32-bit Float
		3	32-bit Float (swapped)
		4	32-bit signed integer
		5	32-bit signed integer (swapped)

MBMaster Read Write Holding Registers (Continued)

Passed Parameter	Value Type	Value	Description
Master Register	Integer 32 Variable	-1	When the Master Register parameter is -1 and the Data Type parameter is 0 or 1 (16-bit data), the subroutine uses the value of the Start Register parameter as the starting register number in the slave and also as the starting table index in the master. See "Examples Using the Master Register Parameter" on page 11.
		>= 0	When the Master Register parameter is greater than or equal to 0 and the Data Type parameter is 0 or 1 (16-bit data), the subroutine uses the value of the Start Register parameter as the starting register number in the Slave and it uses the value of the Master Register parameter as the starting table index in the master. See "Examples Using the Master Register Parameter" on page 11.
			When using Data Type 2, 3, 4, or 5 (32-bit data). See "Case 3" on page 13 for details.
Return Status	String Variable		Status of executing the logic in the subroutine
		No Port	Not able to open Port
		Timeout	No response within time limit (Wait Time parameter)
		Too Many Characters	More then 260 characters
		Address Mismatch	Send and receive address do not match
		CRC or LRC Mismatch	Checksum error
		Function and Exception code	Error returned by the slave device. For details, see Chapter 4 on page 39.
		Invalid Table Length	Used an index greater than the last index of the data table.
		OK	Success
Put Status In	Integer 32 Variable		Status of calling the subroutine
		0	Success
		-67	Out of memory
		-69	Null object error

NOTE: This subroutine no longer has the Wait Time(s) passed parameter. Wait time is set in block 1 Init of the subroutine. Default = 4 seconds.

3: Using the Slave Chart

This chapter includes the following topics:

Initialization

Before starting the Slave chart, there are a number of parameters that must be initialized. These include:

nMB_Comm_Mode

0=Modbus RTU protocol
1=Modbus ASCII protocol

nMB_Data_Type

See "Data Types Supported in the Input and Holding Registers" on page 3 for details.

nSlave_Register_Mode

See "Operation Mode Details for Slave Chart" on page 34.

nMB_Slave_Address

This variable is configured for address 1.

chModbus_Slave_Port:

This is set to the value of the string variable named sMB_Port_Configuration_String in Block-213, "Build Com Handle"

See "Using Communication Handles" on page 6 for details.

Data Tables Used By Slave Chart

- ntMB_Coils_0X (integer 32 table)
- ntMB_Inputs_1X (integer 32 table)
- ftMB_Holding_Registers_4X_Float (float table)
- ftMB_Input_Registers_3X_Float (float table)
- ntMB_Holding_Register_4X_Int (integer 32 table)
- ntMB_Input_Register_3X_Int (integer 32 table)

NOTE: An integer table is used for data types 0, 1, 4, and 5. A float table is used for data types 2 and 3.

You may need to adjust the lengths of these tables to accommodate the amount of Modbus data and the register, coil, and input numbers expected to be accessed by the Modbus master device.

Use strategy logic to populate data in or retrieve data from these tables.

Diagnostic Status Variables

See "Communication Problems When Using the Slave Chart" on page 40.

Operation Mode Details for Slave Chart

Version 8.1e of the Modbus Serial PAC Control toolkit added a new feature called the Operation Mode. In the Slave chart, this is implemented by a new variable named Slave_Register_Mode.

The Slave_Register_Mode variable only applies when using 32-bit data types (DataType is 2, 3, 4, or 5) with Input and Holding Registers. When accessing coils or inputs, or when using 16-bit data types, the Slave_Register_Mode variable has no effect; data is accessed in consecutive indexes.

The Slave_Register_Mode variable is treated as a flag that can either be true or false. In PAC controllers, False is defined as the value 0, and True is defined as any non-zero value.

If the value of Slave_Register_Mode is True when accessing Input or Holding Registers using 32-bit data types (DataType is 2, 3, 4, or 5), data will be accessed in consecutive table indexes. However, if it is False, data will be in every other odd index of the data table, which requires that the Master use a starting register number that is odd.

If you want to maintain compatibility with previous versions of the toolkit, before the Operation Mode feature is added, set the Slave_Register_Mode variable to a value of False (0).

Examples

Coils & Inputs.

Example 1A:

Master		Slave Parameter	Value
		Slave_Register_Mode	(any value)
Starting Coil	19		
Qty of Coils	3		

Master Coil Number		Slave Table Index
19	→	19
20	→	20
21	→	21

Example 1B:

Master		Slave Parameter	Value
		Slave_Register_Mode	(any value)
Starting Input	102		
Qty of Inputs	3		

Master Input Number		Slave Table Index
102	←	102
103	←	103
104	←	104

Input & Holding Registers with 16-bit Data Types

Example 2A:

Master		Slave Parameter	Value
		Slave_Register_Mode	(any value)
Starting Holding Register	7001		
Qty of Registers	3		
		Data Type	0 or 1

Master Register Number		Slave Table Index
7001	→	7001
7002	→	7002
7003	→	7003

Example 2B:

Master		Slave Parameter	Value
		Slave_Register_Mode	(any value)
Starting Holding Register	7001		
Qty of Registers	3		
		Data Type	0 or 1

Example 2B (continued)

Master Register Number		Slave Table Index
7001	←	7001
7002	←	7002
7003	←	7003

Input & Holding Registers with 32-bit Data Types

Example 3A:

Master		Slave Parameter	Value
		Slave_Register_Mode	False (0)
Starting Input Register	7001	*must be an odd number*	
Qty of Registers	4	*must be an even number*	
		Data Type	2, 3, 4, or 5

Master Register Number		Slave Table Index
7001	←	7001
7002		
7003	←	7003
7004		

Example 3B:

Master		Slave Parameter	Value
		Slave_Register_Mode	True (1)
Starting Holding Register	8		
Qty of Registers	6		
		Data Type	2, 3, 4, or 5

Master Register Number		Slave Table Index
8	→	8
9		
10	→	9
11		
12	→	10
13		

Importing the Slave Chart

The Modbus/Serial slave chart allows an Opto 22 controller to function as a Modbus slave device. Unlike the subroutines used in master strategies, which are run as needed, the MB_Slave_Serial chart is started in the Powerup chart and must run all the time. After the chart is started, it continuously monitors the serial port for Modbus traffic.

To copy the Modbus Slave chart to your strategy, you must export the chart MB_Slave_Serial as a PAC Control chart export file (.cxf file) and then import it into your strategy. For more information, see Chapter 8 of form 1700, the *PAC Control User's Guide*.

Start the Modbus Slave chart in the Powerup chart of your strategy.

4: Troubleshooting

This chapter includes the following topics:

Addressing Problems

Controller and Modbus device will not connect

If the Master Subroutines are being used:

Verify the status of the Open Outgoing Communication command by inspecting the nModbus_Port_Status variable located in the Modbus command subroutine being executed. If it was successful, the value should be 0. Any other value indicates a problem. Please see form 1701, the *PAC Control Command Reference* for the definitions of error codes for the Open Outgoing Communication command.

If the Slave Chart is being used:

Verify the status of the Accept Incoming Communication Command by inspecting the nModbus_Port_Status variable. If it was successful the value should be 0. Any other value indicates a problem. Please see form 1701, the *PAC Control Command Reference* for the definitions of error codes for the command.

The controller is receiving data from the Modbus device but the data is not correct

- Verify that the Data Type configured in the controller matches the Modbus device.
- If the Data Type for the controller and Modbus device match but you are still getting incorrect data, try assigning other Data Types on the controller to resolve the issue. For example, when using 32-bit data types, it may be necessary to use the appropriate "swapped" version of that data type.

Using a Serial Module

Use PAC Manager to configure the baud rate, parity, data bits, and stop bits for the module's communication port. See form 1704, the *PAC Manager User's Guide* for detailed instructions. Be sure to store the configuration to flash using the PAC Manager's Status Write page.

Configuring the Communication Handle

Be sure to use separate communication handles for each of the controller's physical serial ports used with the subroutines. It is best to have only one communication handle per port. This prevents two charts from trying to access the same communication port at the same time. For more information, see "Using Communication Handles" on page 6.

Communication Problems When Using the Slave Chart

Use the following variables to help diagnose problems with Modbus communication when using the slave chart:

nChart_Status

This is the status of starting the slave chart.

nException_Code

This is the exception code to the Master device returned by the slave. A value of 0 indicates no exception code.

nModbus_Slave_Port_Status

This indicates the status of the command to open the communication handle to be used for Modbus slave chart communication. If the value is 0, it indicates the communication handle was opened successfully. If the value is negative, then it is the error code. See the command named Open Outgoing Communication in form 1701, the *PAC Control Command Reference* for details.

sLast_Return_Status

This is the status of the last transmitted Modbus packet from the slave.

It will return "OK" if the transmit was successful.

ntReceive_Table

This table contains the characters that were received by the slave chart from the most recent Modbus message received.

sReturn_Status

This indicates the status of the received and transmitted Modbus packets.

It will return "OK" or several error messages. Examples of the error messages are CRC Mismatch and Wrong Slave Address.

Problems When Using the Master Subroutines

Use the following guidelines to help diagnose problems when using the master subroutines:

- Review the status of the communication handle variable to verify that it is open.
- Review the value of the variable used for the Return Status parameter. See "Configuration of Subroutines" on page 15.
- Review the value of the variable used for the Put Status In parameter.

Modbus Exception Codes

For a list of the Modbus exception codes and their meanings, see *Modicon Modbus Protocol Reference Guide*, which is available at this link:

http://www.modbus.org/docs/PI_MBUS_300.pdf