Chapter 7: EZRack Communication (Modbus, ASCII, etc.)

In this Chapter…

7.1 Supported EZRack Communications

The EZRack PLC supports multiple different options for communication with other devices. These communications originate from the Serial or Ethernet port. Some communications need to be setup in the EZRack Designer Pro and others are always available. Please refer to the lists below for information about setup and use of these communications.

7.1.1 EZRack Serial Communications

The EZRack currently supports the following Serial Communications:

- **AVG EZRack Protocol** – This protocol is used for HMI communication and does not need to be setup. The HMI can immediately talk to EZRack over the EZ-CBL or equivalent cable.
- **Modbus RTU Protocol** – This protocol can be used for any third party communication. The EZRack PLC does need to be setup to be a Modbus Master. No setup is needed for the EZRack to act as a Modbus Slave. To communicate over Modbus RTU a RS422 / RS485 cable is needed, please see *Section 7.2* for more information.
- **ASCII Protocol** – This is the most basic binary commutation where all information is sent over in ASCII format. This communication must be setup in the EZRack Designer Pro. For more information please see *Section 7.3* for more information.

7.1.2 EZRack Ethernet Communications

The EZRack currently supports the following Serial Communications:

- **AVG EZRack TCP/ IP Protocol** – This protocol is used for HMI communication and does not need to be setup. The HMI can immediately talk to EZRack over Ethernet.
- **Modbus TCP/IP Protocol** – This protocol can be used for any third party communication. The EZRack PLC does need to be setup to be a Modbus Master. No setup is needed for the EZRack to act as a Modbus Slave. Please see *Section 7.2* for more information.
- **IIoT/MQTT Protocol** – This protocol is the mainly used for Industrial Internet of Things communication but can be used with any device that supports MQTT protocol. Please see *Chapter 8* for more information.
- **EtherNet/IP Protocol** – This protocol can be used to communicate to any device which uses EtherNet/IP communication. Please see *Chapter 9* for more information on how to use and setup EtherNet/IP.

7.2 Modbus Communications

EZRack PLC provides connectivity to other devices over Modbus RTU and Modbus TCP/IP protocol. You can use EZRack PLC either as a Modbus Master/Client or a Modbus Slave/Server.

In this document we will use Modbus Master and Modbus Client synonymously. Similarly, Modbus Slave and Modbus Server would be used synonymously.

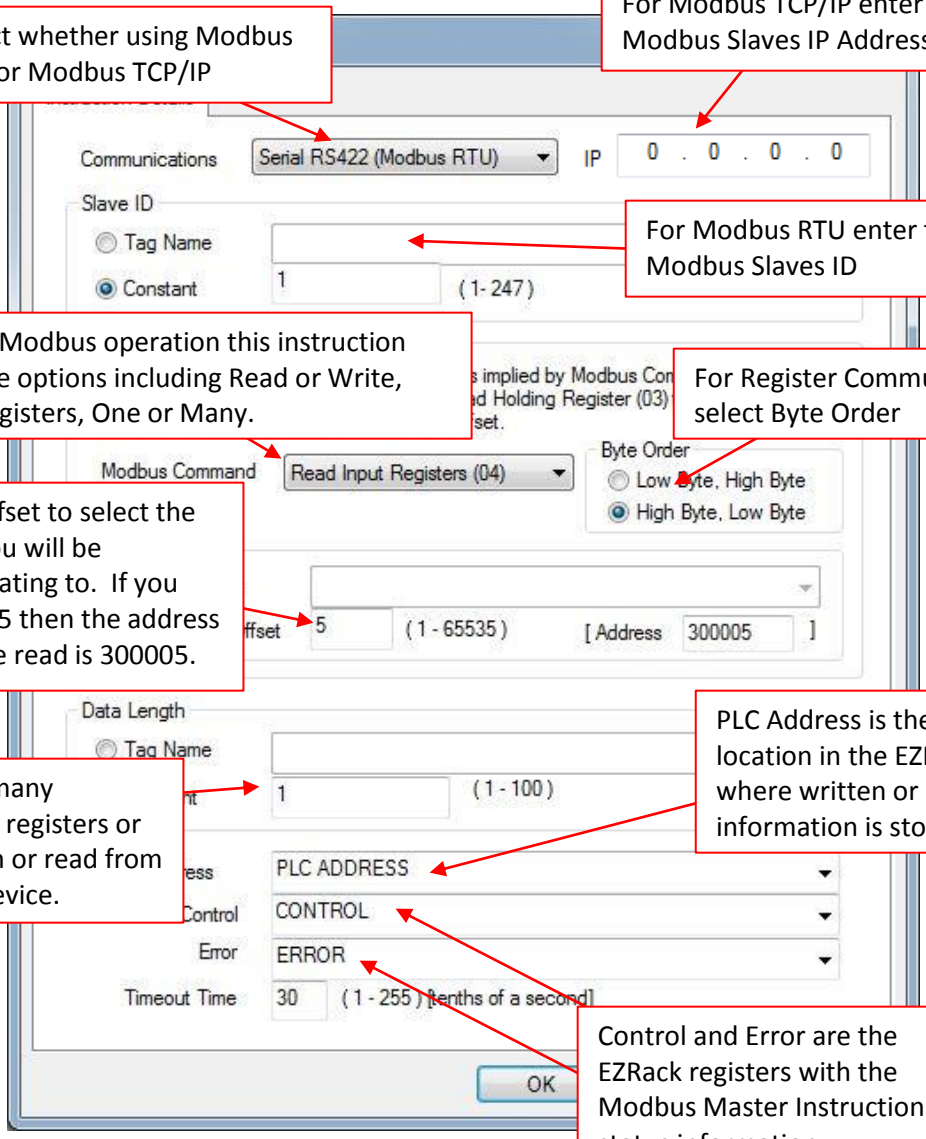
When used as a Modbus Master/Client, EZRack PLC communicates and exchanges data with other Modbus Slaves/Servers. When used as a Modbus Slave, the EZRack PLC can respond to Modbus commands from a Master. The EZRack can be used both as Modbus Master and Slave at the same time if using Modbus TCP/IP. For Modbus RTU only 1 connection can be made a time.

The Ethernet port on the EZRack PLC is used for Modbus TCP/IP connection. Please see section 7.2.1 and 7.2.2 for more information on how to setup the EZRack PLC for Modbus TCP/IP communication.

The Serial port on EZRack PLC is used for the Modbus RTU connection. Please see section 7.2.3 and 7.2.4 for more information on how to setup the EZRack PLC for Modbus RTU communication.

Please see the next page for Modbus Master Instruction Basics.

Modbus Master Instruction Basics



Select whether using Modbus RTU or Modbus TCP/IP

For Modbus TCP/IP enter the Modbus Slaves IP Address

For Modbus RTU enter the Modbus Slaves ID

Select the Modbus operation this instruction will do. The options including Read or Write, Coils or Registers, One or Many.

For Register Communication select Byte Order

Use the offset to select the address you will be communicating to. If you use offset 5 then the address that will be read is 300005.

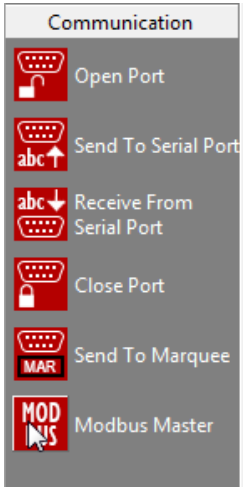
Enter how many consecutive registers or coils written or read from the Slave Device.

PLC Address is the starting location in the EZRack PLC where written or read information is stored.

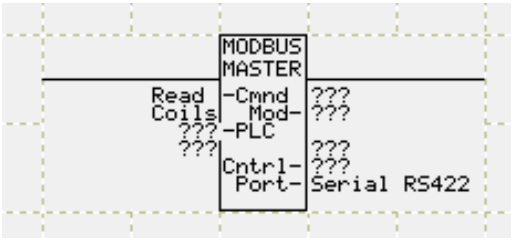
Control and Error are the EZRack registers with the Modbus Master Instruction status information.

7.2.1 Setup EZRack as an Ethernet Modbus Master

The EZRack PLC can act as a Modbus Master to communicate to any third party device that supports Modbus TCP/IP communication. The EZRack currently supports up to 4 simultaneous connections at a time for read / write operations. To setup the EZRack please follow the directions below.

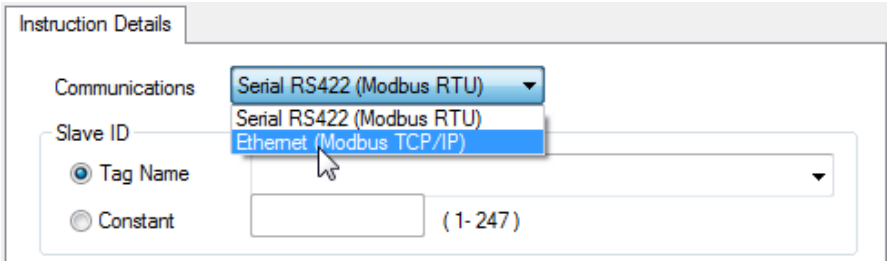


1. In an open project select the Modbus Master instruction from the Instructions Menu or the Operator Bar.

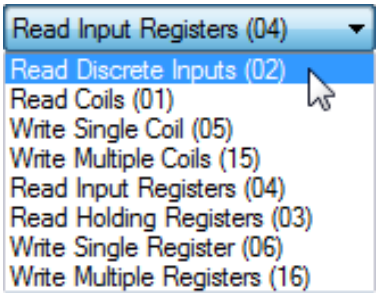


2. Add the instruction to your ladder logic. Then double click on it to open the setup dialogue.

3. In the Instruction Details use the Communications drop down and select Ethernet (Modbus TCP/IP). This will disable the slave ID option and bring up the IP address input area.



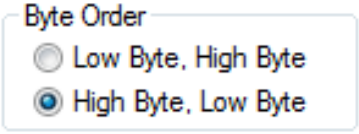
4. In the IP address input please put in the IP address of the Modbus Slave/Server. For example 10.1.200.100.



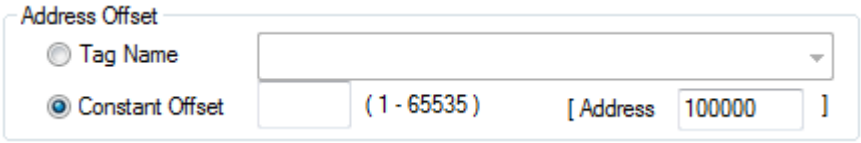
5. Next in the Modbus Command and Address Offset section select the Modbus Command to use. The table below summarizes what address range each command can write to or read from.

Modbus Command	Code	Modbus Address Range*
Read Coils	01	000001 – 065535 (Offset 1 – 65535) No more than 1024 Coils at a time
Read Discrete Inputs	02	100001 – 165535 (Offset 1 – 65535) No more than 1024 Coils at a time
Read Holding Registers	03	400001 – 465535 (Offset 1 – 65535) No more than 100 Holding Registers at a time
Read Input Register	04	300001 – 365535 (Offset 1 – 65535) No more than 100 Holding Registers at a time
Write Singe Coil	05	000001 – 065535 (Offset 1 – 65535) Only one at a time
Write Single Register	06	400001 – 465535 (Offset 1 – 65535) Only one at a time
Write Multiple Coils	15	000001 – 065535 (Offset 1 – 65535) No more than 1024 Coils at a time
Write Multiple Registers	16	400001 – 465535 (Offset 1 – 65535) No more than 100 Holding Registers at a time
**(only Offset entered; type is implied by the command)*		

For registers you further change the Byte Order of the data by using the Byte Order selection.



6. In the Address Offset Input the address you will communicate to.

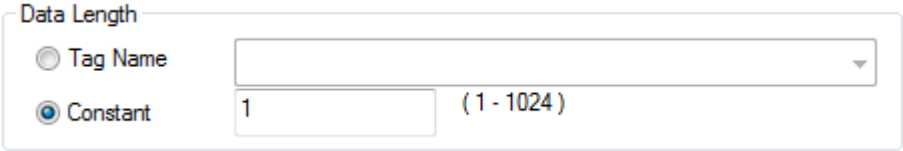


The address offset can be tag based but it always formulated as an offset based on the used command. Below are a few example:

Example 1:
To read one coil at address 200, you will select Modbus Command **Read Coils (01).** Then in the Constant Offset input the value of 200. The Address area will now show 000200.

Example 2:
To write multiple registers at addresses 400005 – 4000020, you will select Modbus Command **Write Multiple Registers (16).** Then in the Constant Offset input the value of 5. The Address area will now show 400005.

7. The Data Length is only available if reading or writing multiple coils/registers. The data length can be tag based or you can put in a constant value. For each Modbus Command the maximum data length will be shown to the side of the Constant Input location.



8. Next the PLC Address must be input. This is the location where the value will either be written (if reading from Slave) or read from (if writing to Slave).

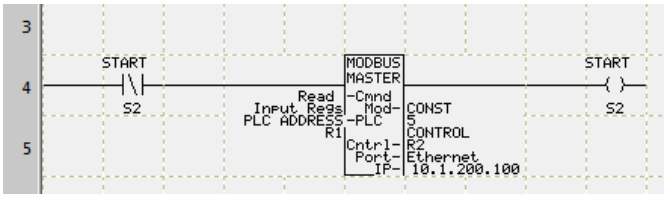
If multiple coils/registers are being written or read then the PLC Address is the starting address. For example if reading 10 registers from the Slave and the PLC Address tag address is R100 then the values will be written to R100, R101, R102… and R109. *Note: These tags will not be auto created, therefore the Auto Addressing will not ignore them and it could be possible they are used in another tag.*

9. Finally for the Modbus Master Instruction enter a register tag for the Control and Error. There is also an option to increase or decrease the time it takes before the communication will timeout. The tables below give values and descriptions for control and error codes.

Control Bit Number	Status when set
B0 (LSB)	Modbus serial Enable
B1	Waiting on reply
B2	Reply processed
B3	Not used
B4	Invalid length for starting address

ERROR CODE	Error	Description
01	Illegal Function	The function code (command code) in the Modbus Master command is not understood by the Slave.
02	Illegal Data Address	The Modbus Master command tried to access an address not available in the Modbus slave device.
03	Illegal Data Value	The Modbus Master Instruction sent a value not acceptable to the slave.
04	Slave Device Failure	An error occurred in slave device, while the slave was trying to perform action requested by Modbus Master.
05	Timeout	A reply was never received from the slave (the communication link Between the Master and the Slave may be disconnected.)
07	Checksum Error	Error in check sum of the reply
08	Slave ID Failure	The slave id in the master command message does not match the slave id Returned in the reply message from the Slave.
09	Port not open error	The Port on EZRack PLC is not opened for Modbus Master Instruction

10. Now that the Modbus Master instruction is created, a contact needs to be placed in front of the instruction. The Modbus Master instruction is only executed once when power is applied. Therefore if you would like to have the instruction constantly repeat, place a normally closed contact in front of the Modbus Master Instruction and a normally open coil after.



7.2.2 Setup EZRack as an Ethernet Modbus Slave

The EZRack PLC is always configured to act as a Modbus Slave. If the EZRack gets a valid Modbus command via TCP/IP it will reply with the requested information. There is no setup needed on the EZRack PLC. Please consult the Modbus Memory Map Table below to know which tags to request for the information you want.

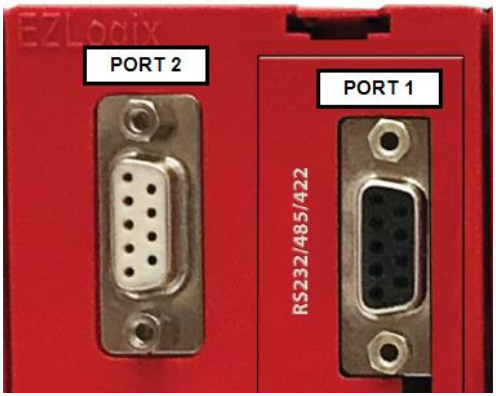
Modbus Memory Map

EZRack PLC Type	Range	Modbus Address	Modbus Type
O – Discrete Outputs	O1 – O128	00001 – 00128	DISCRETE
S – Discrete Internals	S1 – S1024	01001 – 02024	DISCRETE
SD – System Discrete	SD1 – SD16	03001 – 03016	DISCRETE
I – Discrete Inputs	I1 – I128	10001 – 10128	DISCRETE
IR – Input Registers	IR1 – IR64	300001 – 300064	WORD
R – Register Internals	R1 – R16384	400001 – 416384	WORD
OR – Output Registers	OR1 – OR64	450001 – 450064	WORD
SR – System Registers	SR1 – SR20	451001 – 451020	WORD

7.2.3 Setup EZRack as a Serial Modbus Master (Modbus RTU)

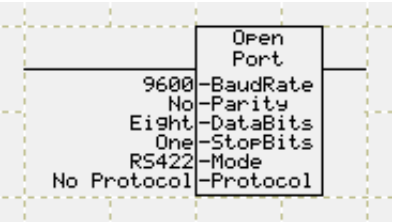
The EZRack PLC can act as a Modbus Master to communicate to any third party device that supports Modbus RTU communication. To setup the EZRack please follow the directions below.

For Modbus RTU communication you will need to use Port 1 of the EZRack PLC and you will need a RS422 or RS485 cable. Please refer to the chart below for pin out information.

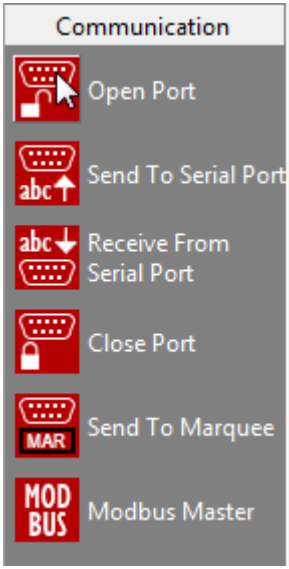


PIN CONFIGURATION	
Pin Number	Function
1	SD -
2	TXD
3	RXD
4	RD -
5	GND
6	SD +
7	CTS
8	RTS
9	RD +

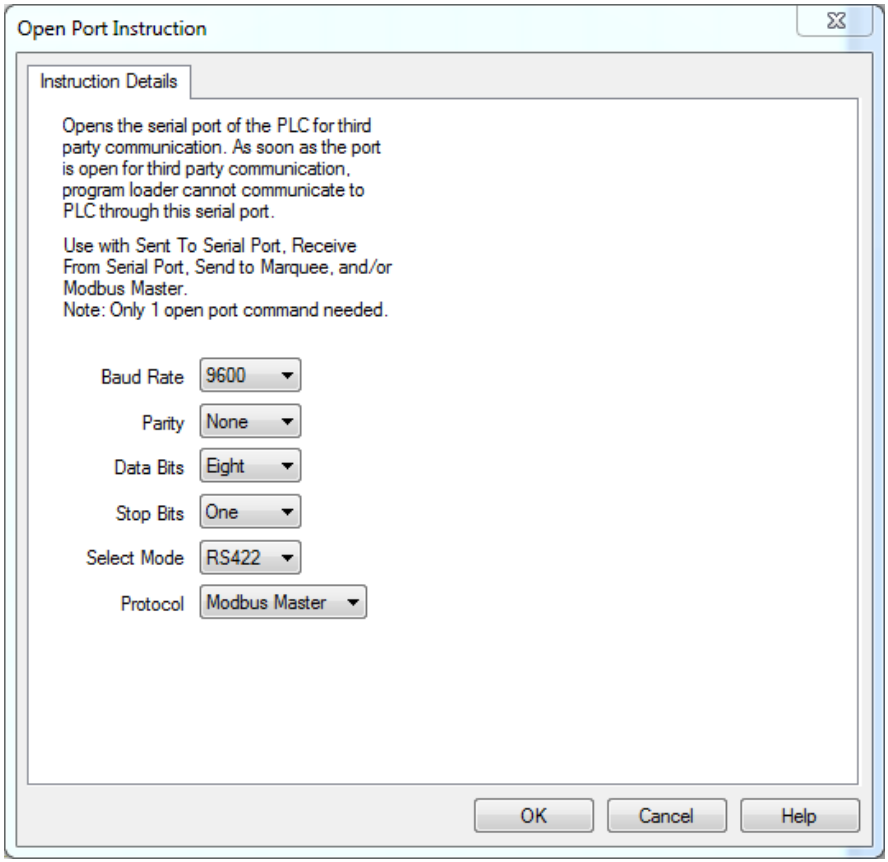
1. In an open project select the Open Port Command from the Instructions Menu or the Operator Bar.



2. Add the instruction to your ladder logic. Then double click on it to open the setup dialogue.



3. In the Open Port Instruction please make sure the Baud Rate, Parity, Data Bits and Stop Bits match the configuration of your other device.

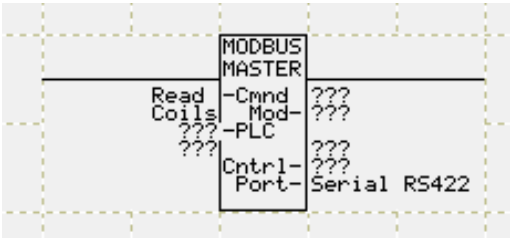


4. Next please select RS422 or RS485 based on which cable you are using. *Note: RS232 does not work for Modbus Master or Modbus Slave.*

5. Finally make sure that the selected protocol is Modbus Master. ***Note: As soon as the Open Port Command is used the PLC will no longer be able to communicate over Port 1 (the primary CPU port).***

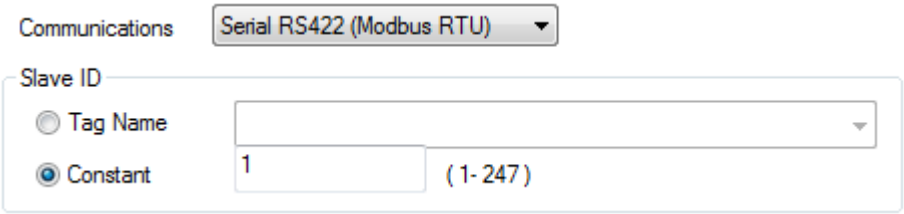
6. Next press OK and your Port 1 is now available to be used to communicate to your Modbus Slave.

7. Next select the Modbus Master instruction from the Instructions Menu or the Operator Bar.

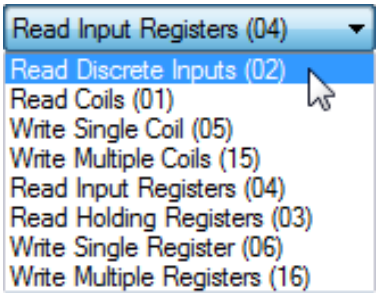


8. Add the instruction to your ladder logic. Then double click on it to open the setup dialogue.

9. In the Instruction Details make sure the Communications option is **Serial RS422 (Modbus RTU).**



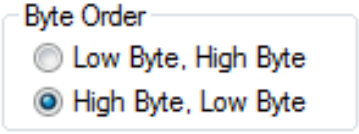
10. Next for the Slave ID input the ID number of the slave you wish to communicate to. This option can also have a tag so you can change the slave you communicate to during operation. *Note: Only 1 Modbus RTU communication can happen at a time. Therefore please wait till the Modbus Communication ends before starting another one.*



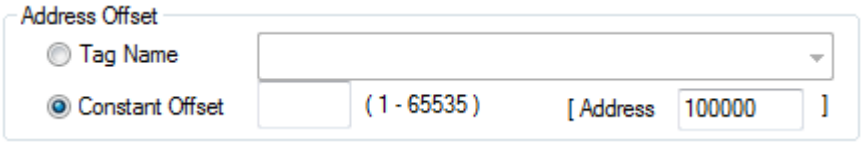
11. Next in the Modbus Command and Address Offset section select the Modbus Command to use. The table below summarizes what address range each command can write to or read from.

Modbus Command	Code	Modbus Address Range*
Read Coils	01	000001 – 065535 (Offset 1 – 65535) No more than 1024 Coils at a time
Read Discrete Inputs	02	100001 – 165535 (Offset 1 – 65535) No more than 1024 Coils at a time
Read Holding Registers	03	400001 – 465535 (Offset 1 – 65535) No more than 100 Holding Registers at a time
Read Input Register	04	300001 – 365535 (Offset 1 – 65535) No more than 100 Holding Registers at a time
Write Singe Coil	05	000001 – 065535 (Offset 1 – 65535) Only one at a time
Write Single Register	06	400001 – 465535 (Offset 1 – 65535) Only one at a time
Write Multiple Coils	15	000001 – 065535 (Offset 1 – 65535) No more than 1024 Coils at a time
Write Multiple Registers	16	400001 – 465535 (Offset 1 – 65535) No more than 100 Holding Registers at a time
**(only Offset entered; type is implied by the command)*		

For registers you further change the Byte Order of the data by using the Byte Order selection.



12. In the Address Offset Input the address you will communicate to.

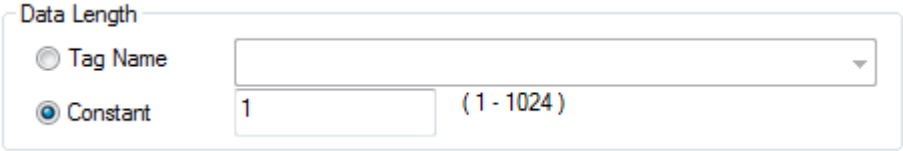


The address offset can be tag based but it always formulated as an offset based on the used command. Below are a few example:

Example 1:
To read one coil at address 200, you will select Modbus Command **Read Coils (01).** Then in the Constant Offset input the value of 200. The Address area will now show 000200.

Example 2:
To write multiple registers at addresses 400005 – 4000020, you will select Modbus Command **Write Multiple Registers (16).** Then in the Constant Offset input the value of 5. The Address area will now show 400005.

13. The Data Length is only available if reading or writing multiple coils/registers. The data length can be tag based or you can put in a constant value. For each Modbus Command the maximum data length will be shown to the side of the Constant Input location.



14. Next the PLC Address must be input. This is the location where the value will either be written (if reading from Slave) or read from (if writing to Slave).

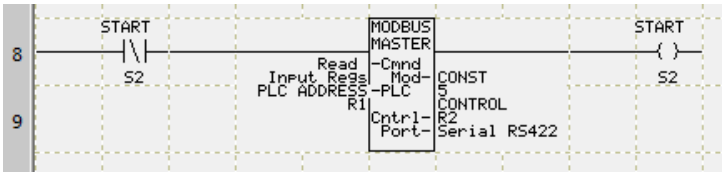
If multiple coils/registers are being written or read then the PLC Address is the starting address. For example if reading 10 registers from the Slave and the PLC Address tag address is R100 then the values will be written to R100, R101, R102… and R109. *Note: These tags will not be auto created, therefore the Auto Addressing will not ignore them and it could be possible they are used in another tag.*

15. Finally for the Modbus Master Instruction enter a register tag for the Control and Error. There is also an option to increase or decrease the time it takes before the communication will timeout. The tables below give values and descriptions for control and error codes.

Control Bit Number	Status when set
B0 (LSB)	Modbus serial Enable
B1	Waiting on reply
B2	Reply processed
B3	Not used
B4	Invalid length for starting address

ERROR CODE	Error	Description
01	Illegal Function	The function code (command code) in the Modbus Master command is not understood by the Slave.
02	Illegal Data Address	The Modbus Master command tried to access an address not available in the Modbus slave device.
03	Illegal Data Value	The Modbus Master Instruction sent a value not acceptable to the slave.
04	Slave Device Failure	An error occurred in slave device, while the slave was trying to perform action requested by Modbus Master.
05	Timeout	A reply was never received from the slave (the communication link Between the Master and the Slave may be disconnected.)
07	Checksum Error	Error in check sum of the reply
08	Slave ID Failure	The slave id in the master command message does not match the slave id Returned in the reply message from the Slave.
09	Port not open error	The Port on EZRack PLC is not opened for Modbus Master Instruction

16. Now that the Modbus Master instruction is created, a contact needs to be placed in front of the instruction. The Modbus Master instruction is only executed once when power is applied. Therefore if you would like to have the instruction constantly repeat, place a normally closed contact in front of the Modbus Master Instruction and a normally open coil after.



7.2.4 Setup EZRack as a Serial Modbus Slave (Modbus RTU)

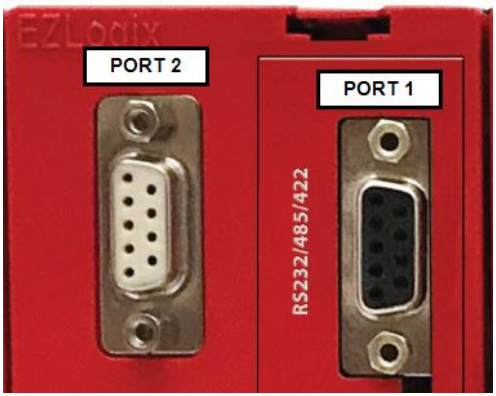
The EZRack PLC is always configured to act as a Modbus Slave but to communicate over Modbus RTU to the EZRack PLC, Serial Port 1 needs to be open and a valid Slave ID needs to be assigned to the EZRack PLC. To open Serial Port 1 follow the directions below. Once the Serial Port is open please consult the Modbus Memory Map Table below to know which tags to request for the information you want.

Modbus Memory Map

EZRack PLC Type	Range	Modbus Address	Modbus Type
O – Discrete Outputs	O1 – O128	00001 – 00128	DISCRETE
S – Discrete Internals	S1 – S1024	01001 – 02024	DISCRETE
SD – System Discrete	SD1 – SD16	03001 – 03016	DISCRETE
I – Discrete Inputs	I1 – I128	10001 – 10128	DISCRETE
IR – Input Registers	IR1 – IR64	300001 – 300064	WORD
R – Register Internals	R1 – R16384	400001 – 416384	WORD
OR – Output Registers	OR1 – OR64	450001 – 450064	WORD
SR – System Registers	SR1 – SR20	451001 – 451020	WORD

Open Serial Port 1

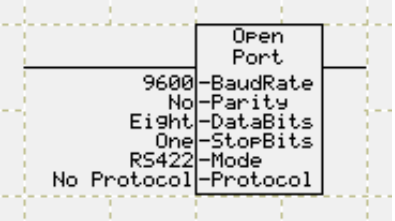
For Modbus RTU communication you will need to use Port 1 of the EZRack PLC and you will need a RS422 or RS485 cable. Please refer to the chart below for pin out information.

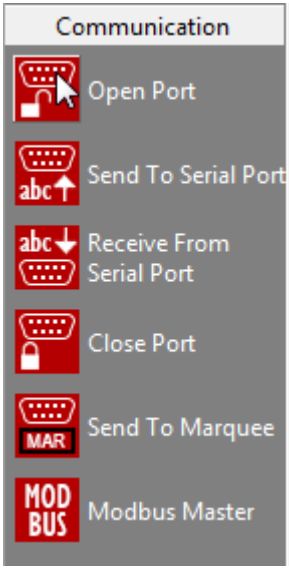


PIN CONFIGURATION	
Pin Number	Function
1	SD -
2	TXD
3	RXD
4	RD -
5	GND
6	SD +
7	CTS
8	RTS
9	RD +

Direction to Open Serial Port 1

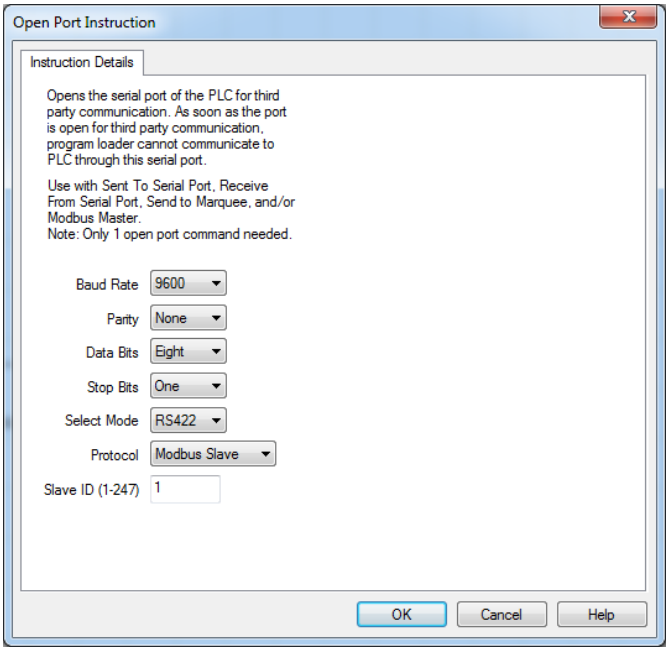
1. In an open project select the Open Port Command from the Instructions Menu or the Operator Bar.





2. Add the instruction to your ladder logic. Then double click on it to open the setup dialogue.

3. In the Open Port Instruction please make sure the Baud Rate, Parity, Data Bits and Stop Bits match the configuration of your other device.



4. Next please select RS422 or RS485 based on which cable you are using. *Note: RS232 does not work for Modbus Master or Modbus Slave.*

5. Next select Modbus Slave for the protocol.

6. Finally enter the Slave ID that will be assigned to the EZRack PLC.

   Note: As soon as the Open Port Command is used the PLC will no longer be able to communicate over Port 1 (the primary CPU port).

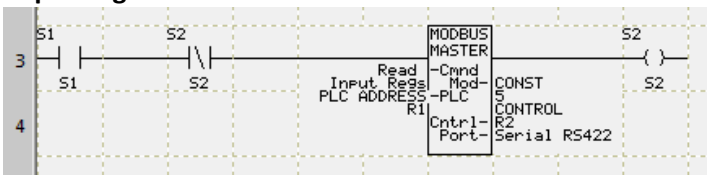
7. Next press OK and your Port 1 is now available to be used to communicate to your Modbus Master.

7.2.5 Modbus Tips and Troubleshooting

When using Modbus RTU or Modbus TCP/IP keep in mind that the Modbus Master instruction will only execute once upon power being applied to it. Also please refer to the table below for basics of how to use multiple instructions.

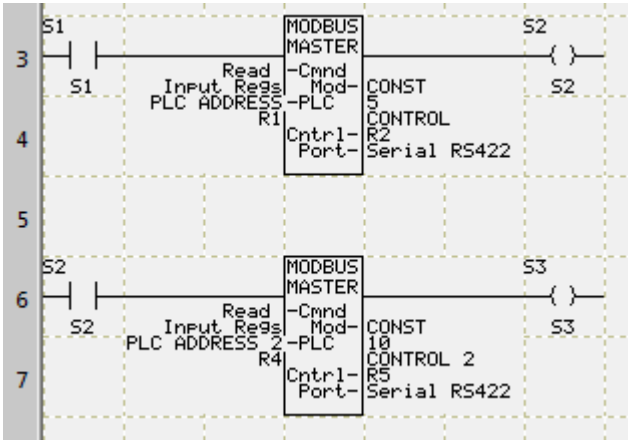
Communication Type	Max Concurrent Running Instructions	Total Max	Connection
Modbus RTU	1	Unlimited*	RS422/RS485
Modbus TCP/IP	4	Unlimited*	Ethernet
**While the number of connections total is unlimited, please make sure instructions are down before another instructions starts. Please refer to examples below.*			

Repeating Modbus Master Instruction:



This example shows a way to repeatedly execute Modbus Master Instruction. S1 will enable the instruction. The Modbus Master instruction is repeatedly executed as long as S1 is true. (Modbus Master Instruction executes once every time the instruction is enabled; to execute it again, the instruction should be disabled and then enabled.)

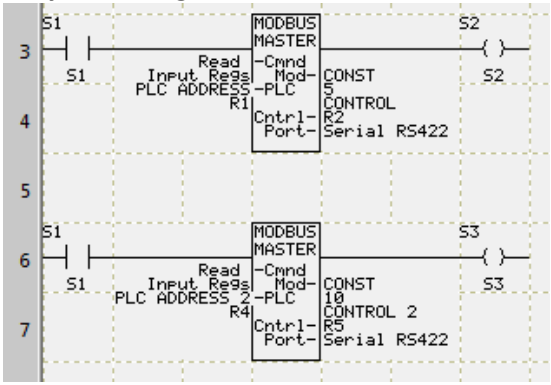
Only 1 Running Modbus Master at 1 Time Example:



This example shows two Modbus instructions. When Normally open contact S1 is true, first instruction gets enabled, and communication to addressed slave starts. S2 will become true when S1 is true AND the Modbus instruction completes its operation.

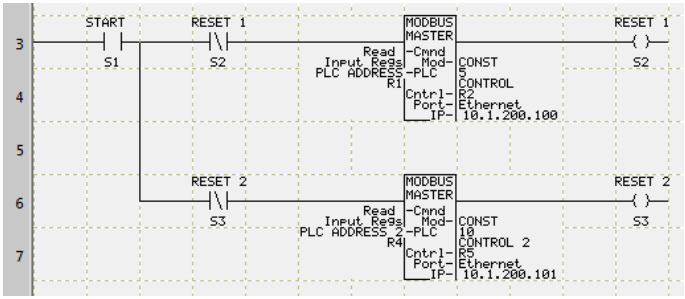
By placing S2 before second Modbus instruction we ensure that the second instruction does not start until the first is completed. In this example S1 should remain on until the second instruction is complete. Otherwise, when S1 turns off, S2 will also turn off, and consequently, second Modbus master instruction may not complete.

Only 1 Running Modbus Master at 1 Time Example (INCORRECT):



This example shows the **INCORRECT** use of the Modbus instructions. Two instructions are enabled simultaneously, resulting in unpredictable behavior for Modbus RTU. This example will function correctly for Modbus TCP/IP.

Constant Modbus TCP/IP Communication Example:



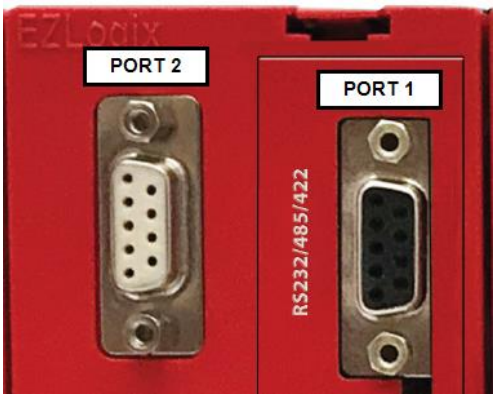
This examples shows how to constantly update Modbus information from multiple slaves when using Modbus TCP/IP. Here both Modbus Master instructions will run once S1 is ON and they will update as fast as they can. *Note: If used for Modbus RTU this will result in unpredictable behavior.*

Troubleshooting Tips

Com Type	EZRack is?	Problem	Solution
RTU & TCP/IP	Master	Illegal Function	Please make sure the Slave supports this function.
RTU & TCP/IP	Master	Illegal Data Address	Please make sure the needed address exist.
RTU & TCP/IP	Master	Illegal Data Value	Please make sure the read value is supported by EZRack PLC
RTU & TCP/IP	Master	Slave Device Failure	Please make sure the Slave is functioning correctly.
RTU	Master or Slave	Timeout	Please make sure the open port setting match those of the other device.
TCP/IP	Master or Slave	Timeout	Please make sure that the correct IP address are set and both devices are connected to Ethernet Hub (or you can use a crossover cable).
RTU	Master or Slave	Checksum Error	Please make sure the open port setting match those of the other device.
RTU	Master or Slave	Slave ID Failure	Please make sure the open port settings match those of the other device. And the correct Slave ID is used for this connection.
RTU	Master	Port not open error	Please make sure you have used an Open Port Instruction before the Modbus Master Instruction.

7.3 ASCII Communication

EZRack PLC provides connectivity to other devices over ASCII protocol. You can send and receive information from the serial port over RS232, RS422 and RS485. To use the Send and Receive from the serial port you need to first open the port using the Open Port Command. ASCII communication uses Serial Port 1 with a RS232, RS422 or RS485 cable, please refer to the table below for pin out information.

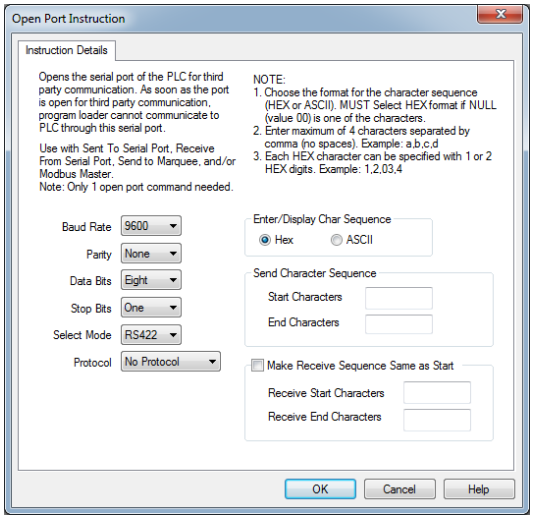


PIN CONFIGURATION	
Pin Number	Function
1	SD -
2	TXD
3	RXD
4	RD -
5	GND
6	SD +
7	CTS
8	RTS
9	RD +

Open Port Command

Open Port command is described in *section 3.3.10* of this manual. Here we repeat this briefly.

Below is the Open Port Instruction dialog box.



The following attributes will need to be set in this dialog box for the Modbus Network you are connecting to.

1. Baud Rate
2. Parity
3. Data bits
4. Stop bits
5. Select Mode “RS232, RS422 or RS485”
6. For Protocol Select “None”

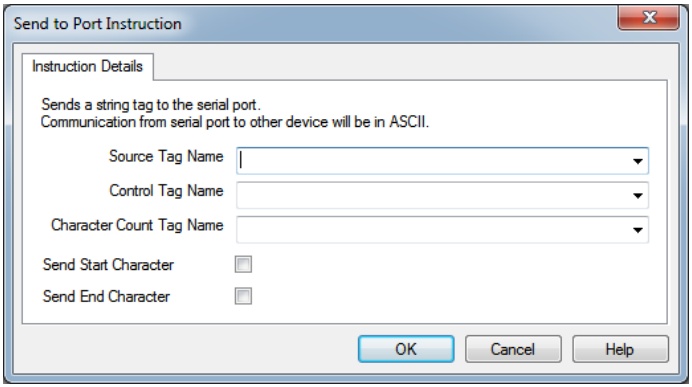
Note: As soon as the Open Port Command is used the PLC will no longer be able to communicate over Port 1 (the primary CPU port).

Enter Optional Parameters:

1. Select how the Char Sequence is inputted (Hex or ASCII).
2. Enter Send Start Characters in the Start Characters field (up to 4 characters).
3. Enter Send End Characters in the End Characters field (up to 4 characters).
4. Enter Receive Start Characters in the Start Characters field (up to 4 characters).
5. Enter Receive End Characters in the End Characters field (up to 4 characters).

Adding Send To and Receive From Port Instructions

To add the Send to Port and Receive From Port instructions, perform the following steps:



1. Select or Add an ASCII tag that contains the string to be sent in the Source Tag field using the drop down list (for a Receive instruction: the String that will receive the characters from the serial port in the Destination Tag field).

2. Select an integer register used by the instruction for status in the Control Register Tag field using the drop down list. The following table describes the control bits in the register:

Bit Number	Function
Bit 0 (lsb)	Enable (0 = Disabled, 1 = Port is Open AND Instruction is Enabled (Power flows to instruction))
Bit 1	Serial transmission done (1= function (transmit or receive) done, 0=not done)

Other bits of the register are used for internal purposes and change state during transmission/receiving.

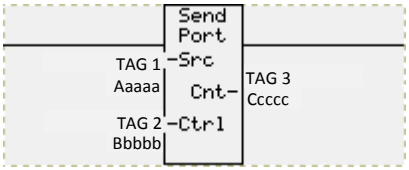
3. Select or Add an integer register that displays the number of characters transferred from the source tag to the serial output buffer in the Character Count Tag field using the drop down list (for a Receive instruction: the Number of characters transferred from the serial port to the destination tag).
4. Check either Send Start Character or Send End Character box if needed.

7.3.1 Setup EZRack to Send Out ASCII Communications

To send ASCII information out you need to use the Send to Serial Port instruction.

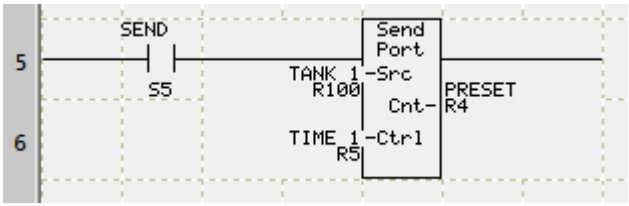
Send to Serial Port:

When power flows through this element, the Send to Serial Port instruction will send an ASCII string present in Src at memory location Aaaaa to the Serial Port. The control and character count used for sending the ASCII string is specified by Cnt at memory location Ccccc and Ctrl at memory location Bbbbb, respectively.



This instruction can only send out the specified ASCII string if the corresponding serial port has been already opened by the Open Port instruction in advance. If the serial port has not been initiated, the Send to Serial Port instruction will not send the ASCII string to the specified port.

Start and End characters can also be sent along with the ASCII string being sent out from the Src register. You can specify Start and/or End characters to be included along with the ASCII string. The starting and ending characters are specified in the Open Serial Port Instruction.

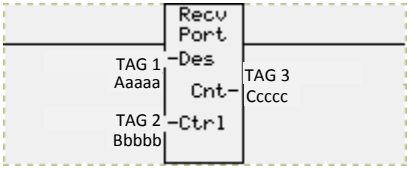


In the example above, if S5 is ON (and the Port is Open), the Send Port command would send the ASCII string as per programmed parameters. If the port is not yet open, the instruction will do nothing, and the Enable Bit in the control register will remain 0, even if the S5 is on.

7.3.2 Setup EZRack to Receive ASCII Communications

To receive ASCII information you need to use the Receive from Serial Port instruction.

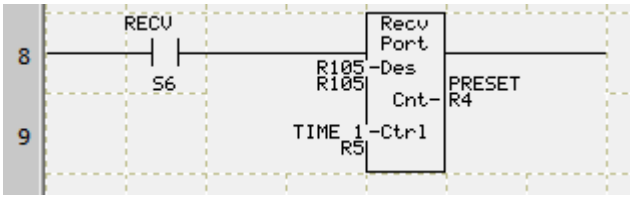
Receive From Serial Port:



When power flows through this instruction, the Receive From Serial Port instruction will receive an ASCII string from the serial port and store it in Dest at memory location Aaaaa. The control and character count used for receiving the ASCII string is specified by Cnt at memory location Ccccc and Ctrl at memory location Bbbbb, respectively.

This instruction can only receive the specified ASCII string if the corresponding serial port has been already opened by the Open Port instruction in advance. If serial port has not been initiated, the Receive from Serial Port instruction will not receive the ASCII string.

Start and End characters can also be received along with the ASCII string being received. You can specify Start and or End characters to be verified when received along with the ASCII string. The starting and ending characters are specified in the Open Serial Port Instruction.



In the example above, if S6 is ON (and the Port is Open), the Send Port command would receive the ASCII string as per programmed parameters. If the port is not yet open, the instruction will do nothing, and the Enable Bit in the control register will remain 0, even if the S6 is on.