



# Connecting Rockwell ControlLogix to Modbus Slaves by Digi One IAP fw "E"

**Keywords:** IA, DOIAP, DOIARP, DORPIA, Rockwell, ControlLogix, Ethernet/IP, Modbus, mapping

**Abstract:** This document describes how to use a MSG block within a ControlLogix PLC program to remotely read or write Modbus slaves using the Digi One IAP and the Release E firmware. The Digi One IAP makes the Modbus device appear as a SLC5 or MicroLogix DF1 device. The ControlLogix uses SLC5 Typed Read and Write to access the Modbus slaves.

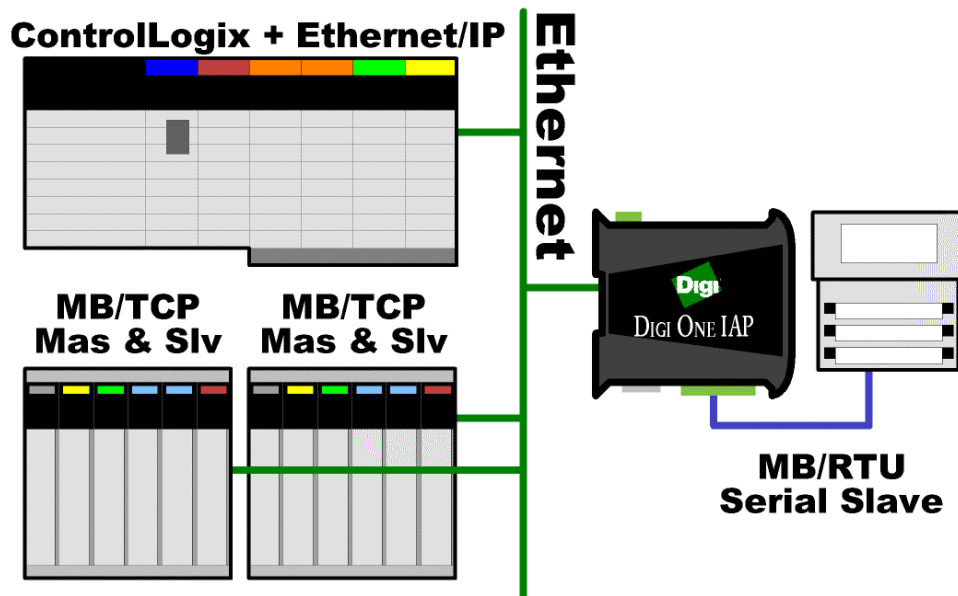
## 1 Introduction

### 1.1 Sample Application

This application note uses a sample design with three Masters and three Slaves:

The ControlLogix can Poll the Modbus/RTU slave via Ethernet/IP

- The ControlLogix can Poll either Modbus/TCP PLC as slave via Ethernet/IP
- The Modbus/TCP PLC can Poll the Modbus/RTU slave via Modbus/TCP



#### *Powerful new Digi One IAP features*

Other vendors make DF1-to-Modbus/RTU converters; adding an ENI allows you to poll the Modbus/RTU slave. However, this approach does *not* help you access the Modbus/TCP slaves, plus this converter pair blocks direct Modbus/TCP access to the Modbus/RTU slave.



The new Digi One IAP approach, however, allows both the ControlLogix and other Modbus/TCP masters to **concurrently** share the Modbus/RTU slave(s).

An alternative is to add a Modbus/TCP cards to your ControlLogix. But these cards are expensive and require a second Ethernet cable, hub/switch port, and IP address. The new Digi One IAP approach allows the existing ControlLogix Ethernet port to (in effect) concurrently handle Rockwell and Modbus data.

## 1.2 Reading Modbus 0x coils

To read data in the Modbus 0x coil area with function 1, use a SLC Typed Read command to a B-file. This command reads full blocks of 16 x 1-bit data packed as words and word aligned. Misaligned bit blocks, partial words, and masked/scattered bit reads are not supported. Reading B10:0 returns the first word of data – Modbus bits 0x00001 to 0x00016 packed as a word. Reading B10:1 returns the second word and so on. Files B0 to B9 are treated like B10.

## 1.3 Writing Modbus 0x coils

To write data in the Modbus 0x coil area with function 15, use a SLC Typed Write command to a B-file. This command writes full blocks of 16 x 1-bit data packed as words and word aligned. Misaligned bit blocks, partial words, and masked/scattered bit writes are not supported.

## 1.4 Writing Single bits (0x coils)

The ControlLogix does not allow using a BOOL or bit data type in the MSG block. In addition, the way Modbus supports single bit read/write is very different from the way DF1/PCCC requires masking bits within words.

To enable writing single bits, the Digi One IAP supports a special use of a string data type written to a B-file. Sending a string such as `"/2=on"` or `"/11=1"` can be used to issue a Modbus function 5 to set or clear a single bit as referenced by the B-file word. Writing `"/2=1"` to B10:0 sets Modbus bit 0x00003 to 1 or on. More detail is given later in this application note.

## 1.5 Reading Modbus 1x and 3x inputs

The ControlLogix does not allow the MSG block to read or write "I" or input files. To enable access to Modbus input data in areas 1x and 3x, the Digi One IAP assumes that AB file numbers 250 to 255 map to these areas.

Read files B250 to B255 to read the Modbus 1x input status area. Misaligned bit blocks, partial words, and masked/scattered bit reads are not supported. B250:0 reads the first word – Modbus bits 1x00001 to 1x00016 packed as a word.

Read files N250 to N255 to read the Modbus 3x input register area. N250:0 reads the first word 3x00001 and so on.

## 1.6 Reading Modbus 4x values

The **SLC Typed Read** to N-file types map to read of Modbus 4x holding registers. This is the algorithm used to map:



- For files N0-N10, use the starting element as the Modbus address. For example, N7:0 and N10:0 map to Modbus 4x00001, while N10:17 maps to Modbus 4x00018. Files N0 to N9 are treated like N10.
- For files N11 to N255, subtract 10 from the file number, multiply this number by 100, and add the starting element. For example, N21:0 is Modbus 4x01101 since  $((21 - 10) * 100) + (0 + 1)$  is 1101. Register N170:17 is Modbus 4x16018 since  $((170 - 10) * 100) + (17 + 1)$  is 16018. The maximum address is thus N255:255 for 4x24757. This may appear complex, but a simple Excel spreadsheet can clearly document your mappings.

If you need access to all 65,535 Modbus registers, use the **PLC2** or **CIF Unprotected Read**. These reads map directly into Modbus registers. A CIF read of offset 0 maps to Modbus 4x00001, offset 100 maps to Modbus 4x00101, and so on. This off-by-one behavior occurs because Modbus register 4x00001 is actually sent on the wire as address 0, not 1.

## 1.7 Writing Modbus 4x values

The **SLC Typed Write** command to an N-file maps to a Modbus function 6 or 16 to Modbus 4x holding registers. Writing one register uses Modbus function 6, while writing more than one register uses function 16.

If you need access to all 65,535 Modbus registers, use the **PLC2** or **CIF Unprotected Write**. These writes map directly into Modbus registers. A CIF write of offset 0 maps to Modbus 4x00001, offset 100 maps to 4x00101, and so on. This off-by-one behavior occurs because Modbus register 4x00001 is actually sent on the wire as address 0, not 1.

## 2 Setting Up the Digi One IAP

The instructions in this document are based on the assumption that you understand the basics of setting up and accessing your Digi product by Ethernet and TCP/IP. If you need help with these procedures, see the related documents on Digi's support Web site.

### 2.1 Overview

Here is a summary of the tasks for configuring the Digi One IAP:

- Configure to accept incoming Ethernet/IP masters (a message source)
  - Configure for Modbus/RTU slaves 0-32 on Port #1
  - Configure for Modbus/TCP slave/unit id 33 at remote IP 192.168.1.xxx
  - Configure for Modbus/TCP slave/unit id 34 at remote IP 192.168.1.yyy
- By default, any EIP messages with DST=35-255 fail (as set up here).

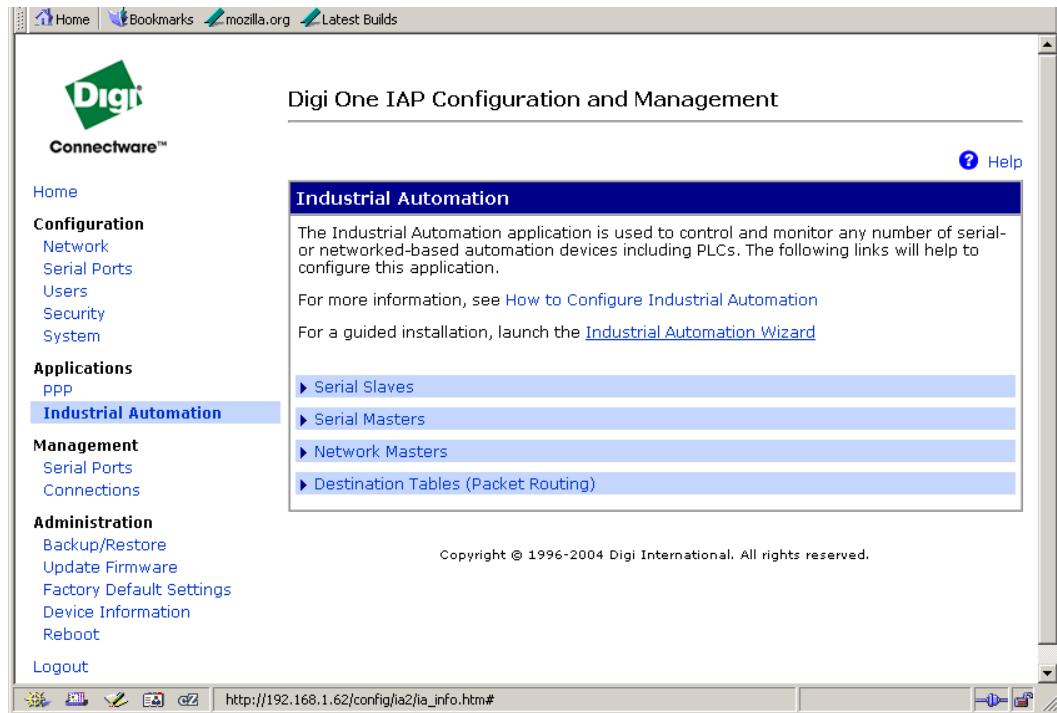


### 3 By Web Wizard, Release “E”

#### 3.1 Select the Industrial Automation Wizard

1. In the left column, under Applications, click **Industrial Automation**
2. Click **Industrial Automation Wizard**.

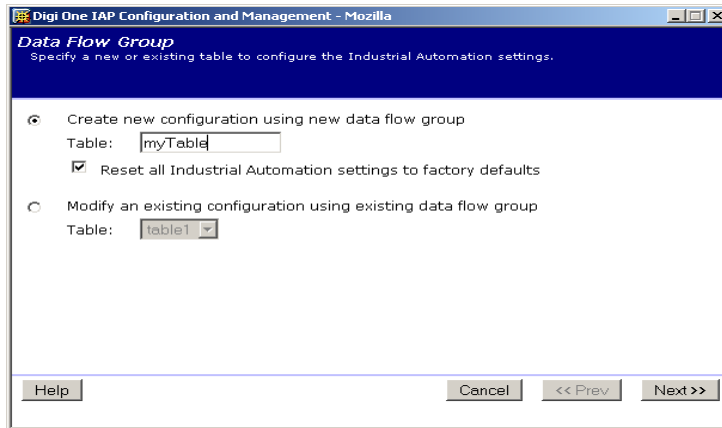
The wizard helps you define where messages come from and where they go.



#### 3.2 Select to Create a new Group, plus Reset all IA settings

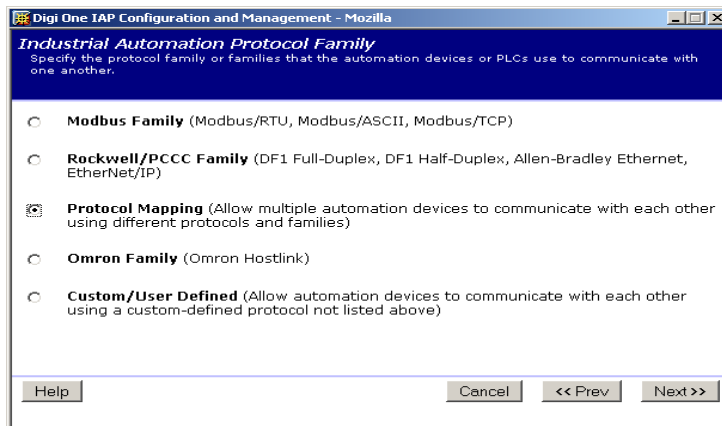
You want to create a new configuration group and clear all existing IA settings.

1. Click **Create new configuration using data flow group**.
2. Check **Reset all Industrial Automation settings to factory defaults**.
3. Enter a name in the **Table** input box, such as table1 or myTable.
4. Click **Next >>**.



### 3.3 Limit this Group to any Protocol that supports Mapping

1. To Map Rockwell protocols to Modbus, click **Protocol Mapping**. – This setting allows most protocols to be selected by you during the wizard.
2. Click **Next >>**.





### 3.4 Define Ethernet/IP as the Master or Message Source

You see some windows about Message Sources that are not covered in this document. Click **Next >>** until you see this window.

1. Click **Receive messages from network devices connecting using the network**.
2. From the **Protocol** input box, select Ethernet/IP.  
The **Network port** automatically sets to 44818.
3. Click **Next>>**.

### 3.5 Define timeouts for the incoming Ethernet/IP connection

1. In the **Message Timeout** input box, enter 10000msec.

This setting makes manual testing with HyperTerminal easier; in addition, ControlLogix usually waits up to 30 seconds for a response. The other two settings are fine as is.

2. Click **Next >>**.



### 3.6 Master Summary – incoming Ethernet/IP is ready

1. To enable incoming Modbus/TCP, check **Continue creating more message sources**.
2. Click **Next >>**.

| Protocol    | Source         | Action |
|-------------|----------------|--------|
| EtherNet/IP | TCP port 44818 | Remove |

Continue creating more message sources

### 3.7 Define Modbus/TCP as the Master or Message Source

1. Click **Receive messages from network devices using the network**.
2. From the **Protocol** input box, select Modbus/TCP.  
The **Network port** automatically sets to 502.
3. Click **Next>>**.

Receive messages from serial device connected to a serial port  
Protocol: Modbus/RTU  
Serial port: 1

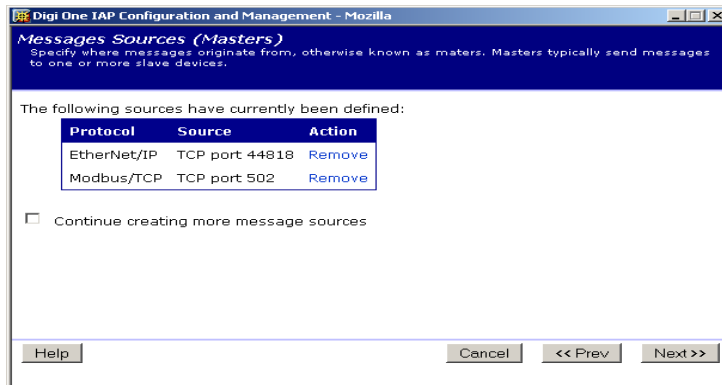
Receive messages from network devices connecting using the network  
Protocol: Modbus/TCP  
Transport: TCP  
Network port: 502



### 3.8 Master Summary – Both incoming Masters all set

At this point you could add another Master such as Allen-Bradley/Ethernet, an informal name for the CSP protocol used by pre-ControlLogix PLC5E and SLC5/05. But for this exercise, these two masters are enough.

Click **Next >>**.



### 3.9 Define Master Priority

You can assign a higher priority to Ethernet/IP or Modbus/TCP Masters. For this exercise, however, accept the default to leave the Masters at same priority; each incoming socket is given fair, round robin access to the serial Modbus slaves.

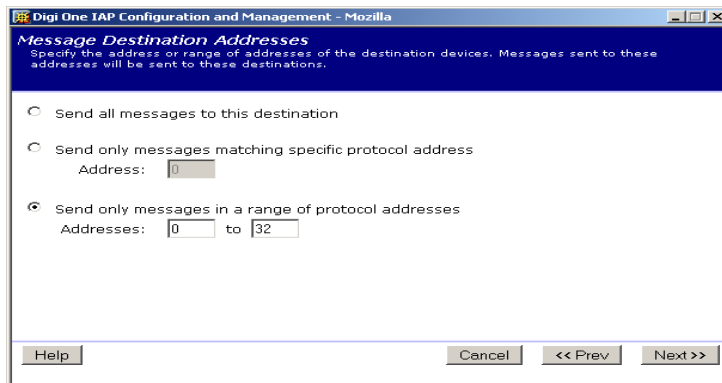
Click **Next >>**.

### 3.10 Define first Message Destination – our Serial Port #1 to PLC(s)

1. You see the Message Destinations window, which is not shown in this document. Click **Next >>** until you see the next window. Click **Send only messages in range of protocol addresses**. enter 0 to 32.

By default, address 0 maps to 1, which is the DST or Destination Node number you will set in the ControlLogix MSG block.

2. Click **Next >>**.





### 3.11 Set the Protocol and Port number

1. From the **Protocol** pull down menu, select Modbus/RTU.  
You'll move through a few windows; the default settings are fine.
2. From the **Serial port** pull down menu, select 1.
3. Click **Next >>**.

### 3.12 First destination complete

1. When you get to this summary, check **Continue creating more message destinations**.
2. Click **Next >>**.

| Address | Protocol   | Destination   | Action                   |
|---------|------------|---------------|--------------------------|
| 0 - 32  | Modbus/RTU | Serial port 1 | Move Up Move Down Remove |



### 3.13 Define second Message Destination – PLC at 192.168.1.xxx

1. Click **Send only messages matching specific protocol address**, and in the **Address** input box, enter 33.
2. Click **Next >>**.

### 3.14 Set the Modbus/TCP Slave details

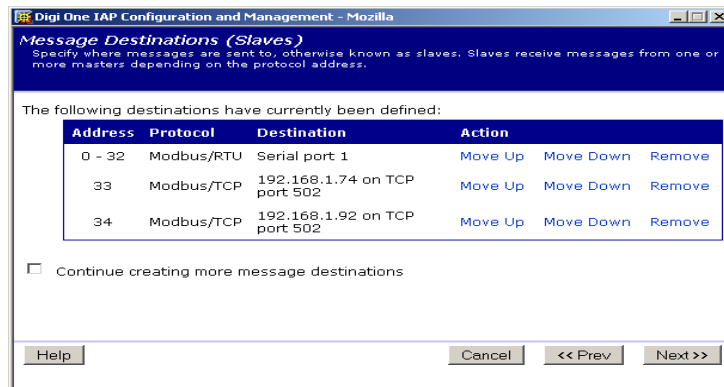
1. Select to use Modbus/TCP to either an IP address or DNS name. You'll move through a few more windows; the default settings are fine.
2. Click **Next >>**.



### 3.15 Define third Message Destination – PLC at 192.168.1.yyy

When you return to the summary, continue and create another destination for protocol address 24 to a Modbus/TCP slave. After you complete the third destination, you return to this summary.

1. Uncheck **Continue creating more message destinations**.
2. Click **Next>>**.

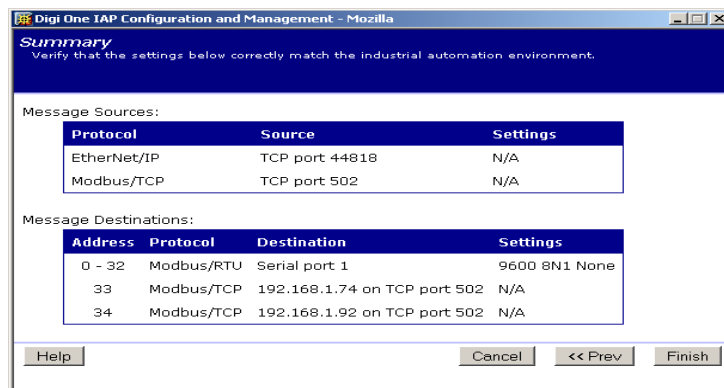


### 3.16 Finished – review the settings

Here is the final summary window. The Digi One IAP accepts messages by Ethernet/IP and Modbus/TCP, and it has three destinations:

- Up to 32 slaves on the serial port
- Two remote Modbus/TCP slaves

Click **Finish**.



### 3.17. Reboot the Digi One IAP

You can make minor changes to the Digi One IAP configuration without rebooting. When you change the number or type of Masters (message sources) or number or type of Slave (message destinations), however, it is safest to reboot. These changes, which affect the number and type of tasks running in the Digi One IAP RTOS, occasionally fail to take effect without a reboot.



## 4 Setup by Telnet, Release “E”

1. To log into your Digi One IAP, use either HyperTerminal or telnet.
2. Enter the IP address of your DS and the well-known telnet port of 23.
3. Next is a text script you can cut and paste into an editor such as WordPad. Edit the script as required, then cut and paste again into HyperTerminal using **Edit | Paste to Host..**

```
# clear all IA config
revert ia=factory

# setup port 1 as Modbus/RTU slave (baud = 9600,8,E,1)
set port ra=1 dev=ia
set line ra=1 baud=9600 csize=8 parity=E stopb=1
set ia serial=1 protocol=mbrtu type=slave chrtout=50ms slvtout=1sec

# setup network for Ethernet/IP PCCC-encap and Modbus/TCP incoming
set ia master=1 active=on protocol=ethernetip transport=tcp ipport=44818 table=1
set ia master=2 active=on protocol=modbustcp transport=tcp ipport=502 table=1

# setup destination table
set ia table=1 name=table1
set ia table=1 addroute=1 active=on protocol=mbrtu
set ia table=1 route=1 protaddr=0-32 type=serial port=1
set ia table=1 addroute=2 active=on protocol=mbtcp protaddr=33
set ia table=1 route=2 type=ip ipaddress=192.168.1.74 ipport=502
set ia table=1 addroute=3 active=on protocol=mbtcp protaddr=34
set ia table=1 route=3 type=ip ipaddress=192.168.1.92 ipport=502

# reboot the Digi One IAP
boot action=reset
```



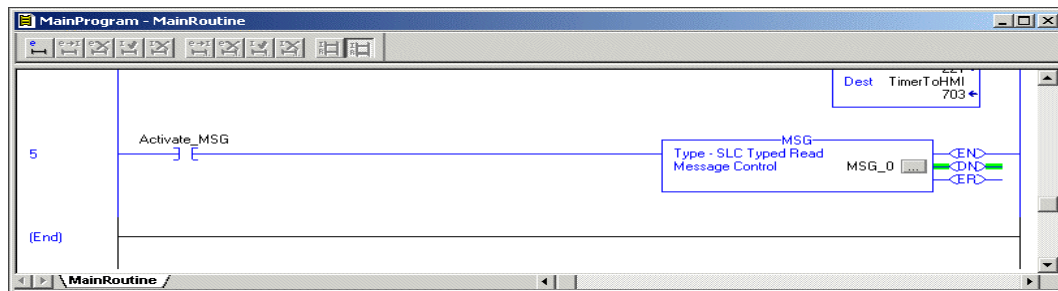
## 5 Setting Up the ControlLogix MSG Block

The instructions in this document are based on the assumption that you understand the basics of programming and accessing your ControlLogix PLC.

### 5.1 Create your MSG Block rung in RSLogix 5000

1. Create a MSG block using whatever trigger is appropriate.
2. Create the tags you want to read to or write from.

The tags can be either integer or string tags.



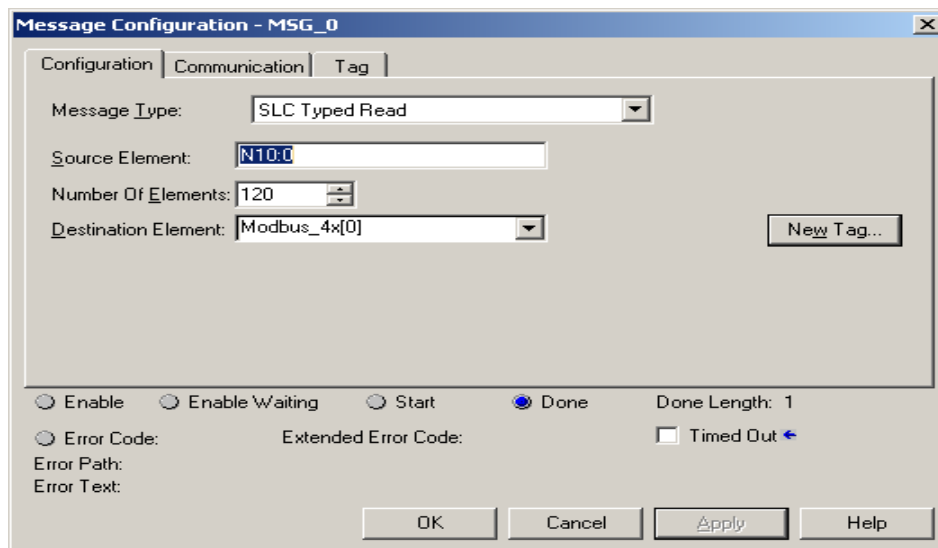


## 5.2 Configure the MSG for reading 0x, 1x, 3, or 4x areas

| Reading these files | Map to                       |
|---------------------|------------------------------|
| B0 to B249          | Modbus function 1 to 0x area |
| B250 to B255        | Modbus function 2 to 1x area |
| N0 to N249          | Modbus function 3 to 4x area |
| N250 to N255        | Modbus function 4 to 3x area |

Select your MSG block in RSLogix and open it up for detailed configuration:

1. Click the **Configuration** tab.
2. From the **Message Type** pull down menu, select **SLC Typed Read**.
3. Set the **Source Element** as required.  
For example, N10:0 starts reading at 4x00001 or  $((10-10)*100)+0+1$
4. In the **Number of Elements** box, select the number of words to read.  
The **SLC Typed Read** is limited to 122 words, which is less than the 125-word limit of Modbus.
4. From the **Destination Element** pull down menu, select your ControlLogix tag – in this case, an integer array named Modbus\_4x that can hold at least 120 words starting at [0].5. Click **OK**.





1. Click the **Communication** tab.
2. Set the **Path** as shown, using the Digi One IAP's IP address.
3. Click **CIP With Source ID**, and set the **Destination Node** to the slave address to index. In our example:
  - Modbus slaves 1-32 are Modbus/RTU serial slaves.
  - Modbus slaves 33 and 34 are remote Modbus/TCP slaves.

You'll need to use the appropriate slave address based on your Digi One IAP configuration.

4. Click **OK**.

The screenshot shows the 'Message Configuration - MSG\_0' dialog box with the 'Communication' tab selected. The 'Path' field contains 'EtherNet\_IP, 2, 192.168.1.20'. Under 'Communication Method', 'CIP With Source ID' is selected. The 'Destination Node' is set to '1 (Octal)'. The 'Done' radio button is selected. The 'Cache Connections' checkbox is checked. At the bottom, the 'Done Length' is set to '1' and the 'Timed Out' checkbox is unchecked. The 'OK' button is highlighted.



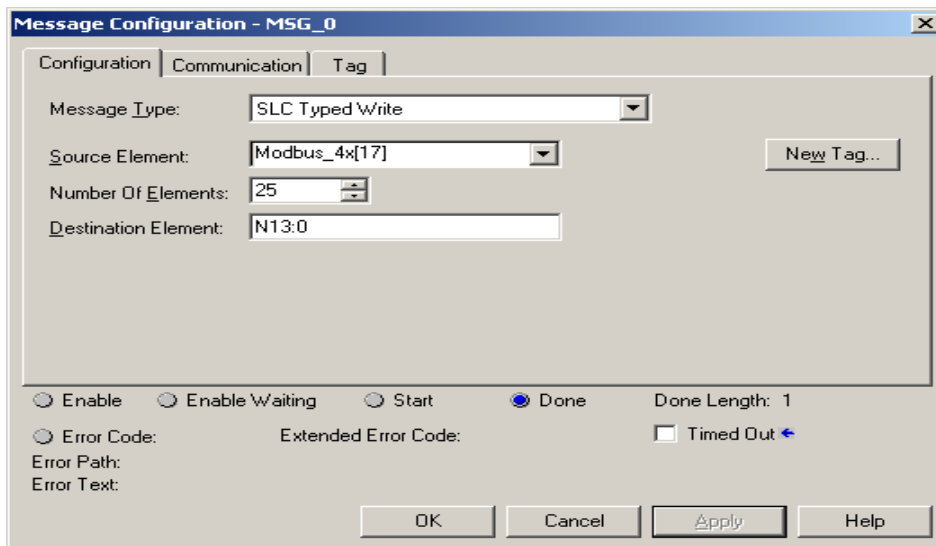
### 5.3 Configure the MSG for writing 0x or 4x areas

| Writing these files | Map to   |
|---------------------|--|
| B0 to B249          | Modbus function 15 to 0x area  |
| N0 to N249          | Modbus function 16 to 4x area<br>or Modbus function 6 if a single register |

Select your MSG block in RSLogix and open it up for detailed configuration:

1. Click the **Configuration** tab.
2. Set **Message Type** to SLC Typed Write
3. Set **Source Element** to your ControlLogix tag – in this case, the eighteenth integer in an array named Modbus\_4x. ControlLogix arrays are zero-based.
4. Set **Number of Elements** to the number of words to write – 25 in this case
5. Set **Destination Element** as required.

N13:0 starts writing at Modbus 4x00301 or  $((13-10)*100)+0+1$ .



Set the Communication Path, CIP With Source Id, and Destination Node on the **Communication Tab** as described above for the SLC Typed Read.

## 5.4 Configure the MSG for writing a single bit to 0x area

Writing single bits to Modbus is more challenging than writing words. Suppose you want to set Modbus bit 0x00036 to 1. The direct mapping to Rockwell file notation would be B10:2/3, but you cannot write this single bit from ControlLogix.

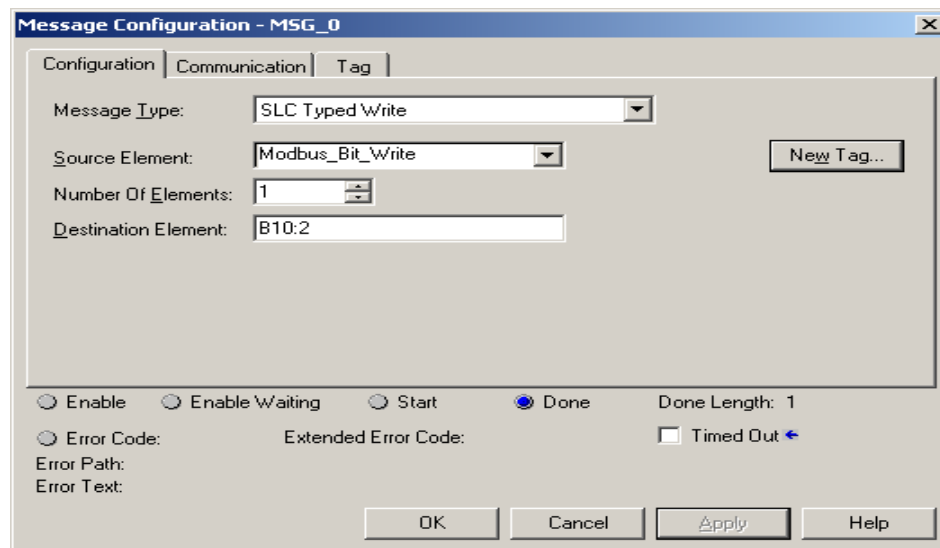
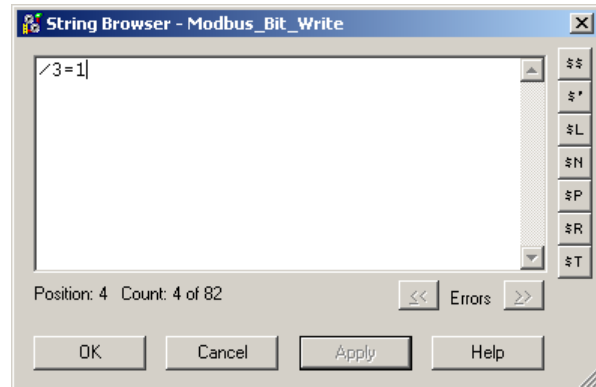
Create a string data tag and fill it with a string that defines:

- The bit to affect
- Whether to set the bit to 0 or 1.

Then you write this string to word B10:2 with a SLC5 Typed Write.

1. Set **Message Type** to SLC Typed Write.
2. Set **Source Element** to your ControlLogix tag of type String.
3. Set **Number of Elements** to 1 element.
4. Enter a **Destination Element** as required.

B10:2 covers Modbus bits 0x00033 to 0x00048, and you are setting bit 3 (/3 is zero-based) or 0x00036.



Set the Communication Path, CIP With Source Id, and Destination Node on the **Communication Tab** as described above for the SLC Typed Read.



## 6 How to test this Application

### 6.1 Testing by Simulator

Protocol mapping is not as easy to test because it introduces many details (Lynn, by "details," do you mean variables, or maybe factors?) that can go wrong (Lynn: that can fail,? Or "that can have problems?").

The easiest way to test may be to use a Modbus slave simulator, which allows you to see the actual polls. One simulator is ModSim32 from [www.win-tech.com](http://www.win-tech.com).

## 7 Trouble Shooting and FAQ

### 7.1 Can I use PLC5 or CIF style commands?

Commands other than CIF/PLC2 Unprotected Reads/Writes and SLC500 style Typed Reads/Writes are processed by the Digi One IAP doing protocol mapping; however the results may not be what you want.

Because the ControlLogix MSG block doesn't have any preference for PCCC commands, you should just use the SLC Typed Read/Write if you want predictable results as documented in this application note. (Lynn, is it a case of should? Or is it something a user must do?)

### 7.2 Why is my MSG block returning a bad status?

If your Ethernet is working, and the ControlLogix is able to create an Ethernet/IP connection to the Digi One IAP, two conditions that can cause an error response include:

- The MSG block is sending something other than SLC Typed reads/writes.
- The MSG block is requesting file types other than "N" or "B."

## 8 How Rockwell polls turn into Modbus Polls

Although your Rockwell master is issuing PCCC polls, the Digi One IAP must turn these into actual Modbus commands for Modbus slaves to be able to answer.

### 8.1 Given Modbus Notation, Determine your Rockwell Notation

Obtain the *Digi document 90000652.xls* from <http://www.digi.com/support/ia>. It is an Excel spreadsheet to convert your existing Modbus notations into the appropriate Rockwell SLC500 polls. With the Digi One IAP, do not think "What does this Rockwell address poll?"; instead think "***I have this known Modbus register, what is the correct SLC500 address to poll it?***".



## 8.2 How to Document

If you supply a drive unit with Modbus/RTU communications, how can you enable Rockwell users to buy and manager your product? You refer them to the Digi One IAP and this application note, but that is only part of the solution.

The Digi One IAP allows customers to poll your driver *concurrently* by:

- DF1 serial in the second RS-232 pass-thru port
- AB/Ethernet (CSP) from older SLC5 and PLC5E
- DF1 serial by remote DF1 encapsulation in TCP/IP
- ControlLogix using Ethernet/IP using SLC Typed Read/Write
- Modbus/RTU by serial or Modbus/TCP using the network

You also need to document your device in the AB file notation to simplify setup. Combining your Modbus/RTU product with a Digi One IAP and a good dual-protocol register map will open up a whole new market for your products with Rockwell Automation / Allen-Bradley users.

Here is a simple partial example:

| AB   | Modbus | Mode  | Desc  | Values   |
|--|--------|-------|---|--|
| N10:0  | 4x0001 | Rd    | Drive Horsepower  | In HP (ie: 15 = 15hp)  |
| N10:1  | 4x0002 | Rd    | Software version  | BCD like 0101  |
| N10:2  | 4x0003 | Rd/Wr | Parameter Lock  | 0/disable, 1/enable  |
| N10:3<br>N10:3/0<br>N10:3/1<br>N10:3/2<br>N10:3/3<br>N10:3/4 | 4x0004 | Rd/Wr | Run Commands<br>bit 1: Stop / Brake<br>bit 2: Forward Run<br>bit 3: Reverse Run<br>bit 4: Fault Reset<br>bit 5: Coast to Stop | Only 1 bit may be set per write, or the write is ignored.<br>Read back shows current status. |
| N10:4  | 4x0005 | Rd    | Current   | By 0.1A  |
| N10:5  | 4x0006 | Rd    | Frequency   | By 0.01 Hz   |
| N10:6  | 4x0007 | Rd    | Voltage   | volts  |
| N10:7  | 4x0008 | Rd    | Output Power  | Watts  |
| N10:8  | 4x0009 | Rd    | Speed   | RPM  |