NOTE Some protocols on the Digi One IAP product family that are described in this document are being moved to legacy status because the protocols described are no longer actively supported. The products themselves continue to be active products.

Rockwell ControlLogix polling ASCII devices as PCCC slaves

March 2020

90000642

Contents

1 Introduction

Abstract

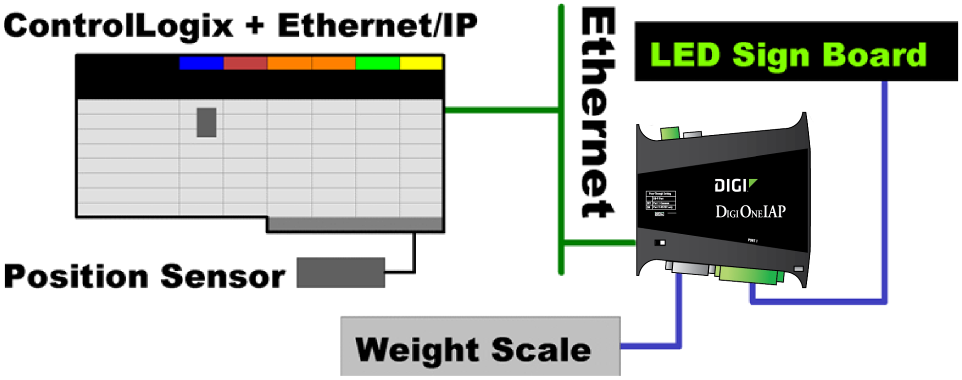
This application note explains how to use a MSG block within a ControlLogix PLC program to remotely read or write ASCII data via the Digi One IAP. The Digi One IAP makes the ASCII device (such as a bar-code scanner, weight scale, or LED marquee sign) appear as a PCCC/DF1 device. So the ControlLogix can just use SLC5 typed read and writes to access it.

1.1 Example application

The example design below shows two distinct uses for the Digi One IAP's ASCII_Import function and uses both serial ports.

A ControlLogix system uses digital input from a position sensor to sense the presence of a product on a moving conveyor. It uses an ASCII command to poll the weight from a weight scale under the conveyor. So it must send a fixed ASCII command to receive a variable ASCII response.

The second application is the ControlLogix formats and it sends ASCII strings to be displayed on a marquee or LED signboard. It must only send ASCII strings and expect no response.



1.2 Overview of the ASCII_Import function

The ControlLogix (like most Ethernet-enabled PLC) only supports specific protocols on its Ethernet port. There is no way to open a raw socket to access a remote ASCII device by TCP/IP tunneling.

The Digi One IAP ASCII_Import function overcomes this by making the ASCII serial device appear as a small Rockwell PLC. It accepts specific **Read** and **Write** commands, and responds with the expected protocol responses. It allows Rockwell PLC to use MSG blocks to send SLC Typed **Read** and **Write** commands. ASCII data is packed into the data registers read or written.

1.2.1 ASCII_Import as receive_only

Suppose you have a device that just sends out messages, such as printer output from a vibration monitor or a bar-code scanner. The Digi One IAP considers this “Receive Only” or “duplex=rcv.” Your PLC should do a protocol read of the Digi One IAP, and the

response will be either all zero (no waiting data) or the oldest ASCII message waiting in queue. Setting the slavetimeout=0 causes old messages to be saved forever to be read, while setting slavetimeout=0 from any value from 1 to 65,535 milliseconds (about one minute) will cause old messages to be flushed after this timeout.

1.2.2 ASCII_Import as transmit_only

Suppose you have a device that just receives messages. An example could be a serial printer or an LED Marquee display. The Digi One IAP considers this “Transmit Only” or “duplex=xmt.” Your PLC creates an ASCII string and does a protocol WRITE to the Digi One IAP, which unpacks the ASCII data and sends it out the serial port.

1.2.3 ASCII_Import as half_duplex

Suppose you have a device that must be polled for a response. An example could be a weight scale with various gross, net, and tare weights. The Digi One IAP considers this half-duplex or duplex=half. Your PLC creates an ASCII string that represents the poll and does a protocol WRITE to the Digi One IAP, which unpacks the ASCII data and remembers it.

The WRITE is not sent out at this time. The reason is simple. A protocol WRITE does not expect ASCII data within the response. Instead—when your PLC does a protocol read of the Digi One IAP—it sends the remembered POLL, and returns the ASCII response packed into the protocol data response.

The Digi One IAP maintains four separate buffer pairs per serial port. These can be used by four remote masters to avoid conflict, or a single master can use them to remember four distinct polls—such as the poll for read_gross, read_net, read_tare, and reset_tare for a weight scale.

2 Setting up the Digi One IAP

This application guide assumes you understand the basics of setting up and accessing your Digi product by Ethernet and TCP/IP. If you need help, see other guides at Digi's support site at www.digi.com/products/serial-servers/digioneiapfamily.

2.1 Overview

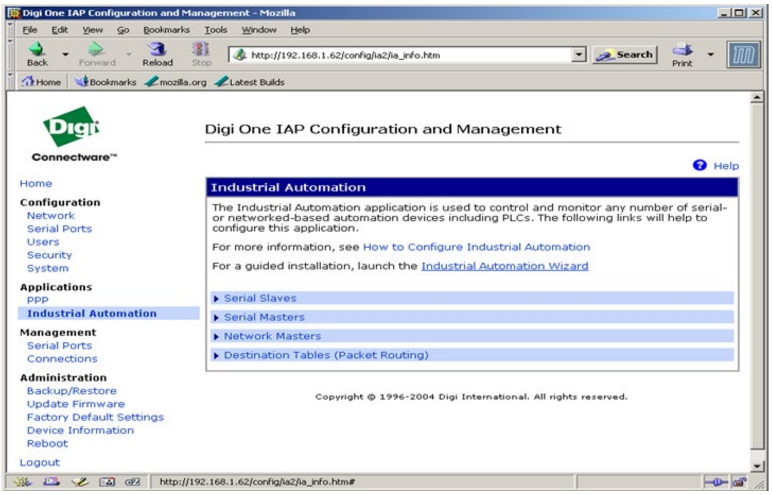
To configure the Digi One IAP:

- Configure to accept incoming Ethernet/IP masters (a message source)
- Configure for an ASCII device on port 1 that we poll for response
- Configure for an ASCII device on port 2 that we just write to
- Configure EIP messages to DST=1 to go to port 1
- Configure EIP messages to DST=2 to go to port 2

2.2 How to set up by web wizard

2.2.1 Launch the Industrial Automation Wizard

1. Click the **Industrial Automation** link under the **Applications** section in the left column.
2. Select the **Launch the Industrial Automation Wizard** link. This walks you through the process of defining where messages come from and where they go.



http://192.168.1.62/config/ia2/ia_info.htm

Digi One IAP Configuration and Management

Industrial Automation

The Industrial Automation application is used to control and monitor any number of serial- or networked-based automation devices including PLCs. The following links will help to configure this application.

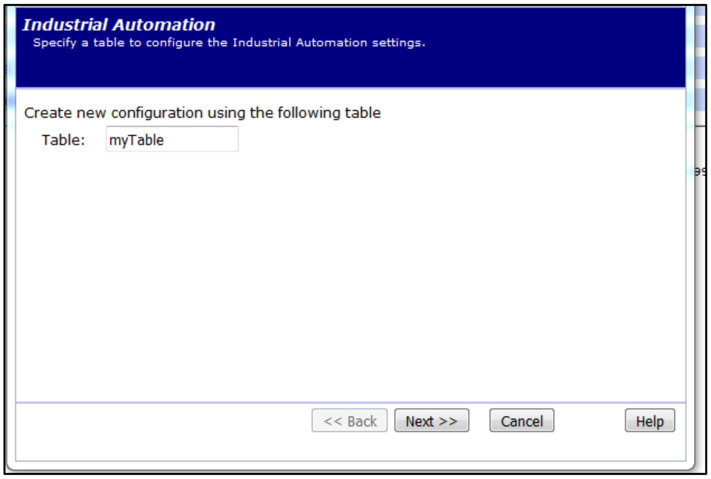
For more information, see How to Configure Industrial Automation

For a guided installation, launch the Industrial Automation Wizard

Copyright © 1996-2004 Digi International. All rights reserved.

http://192.168.1.62/config/ia2/ia_info.htm#

2.2.2 Create a new configuration



Industrial Automation

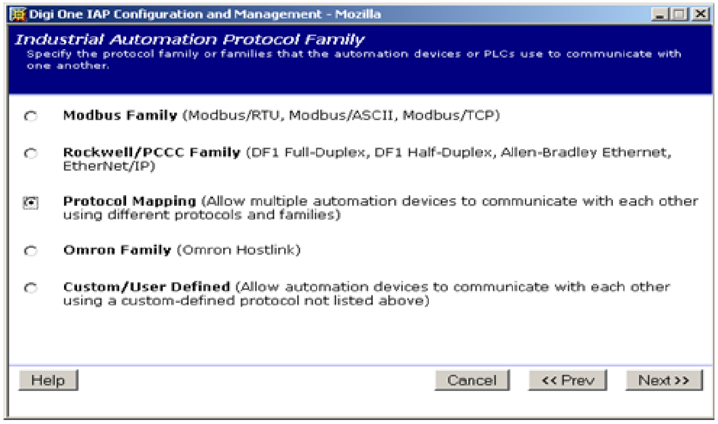
Specify a table to configure the Industrial Automation settings.

Create new configuration using the following table

Table: myTable

2.2.3 Limit configuration to any protocol that supports “Mapping”

1. Create a **Protocol Mapping** group to bridge Rockwell protocols to ASCII. This enables most protocols to be selected.
2. Click the **Next** button.



Digi One IAP Configuration and Management - Mozilla

Industrial Automation Protocol Family

Specify the protocol family or families that the automation devices or PLCs use to communicate with one another.

○ **Modbus Family** (Modbus/RTU, Modbus/ASCII, Modbus/TCP)

○ **Rockwell/PCCC Family** (DF1 Full-Duplex, DF1 Half-Duplex, Allen-Bradley Ethernet, EtherNet/IP)

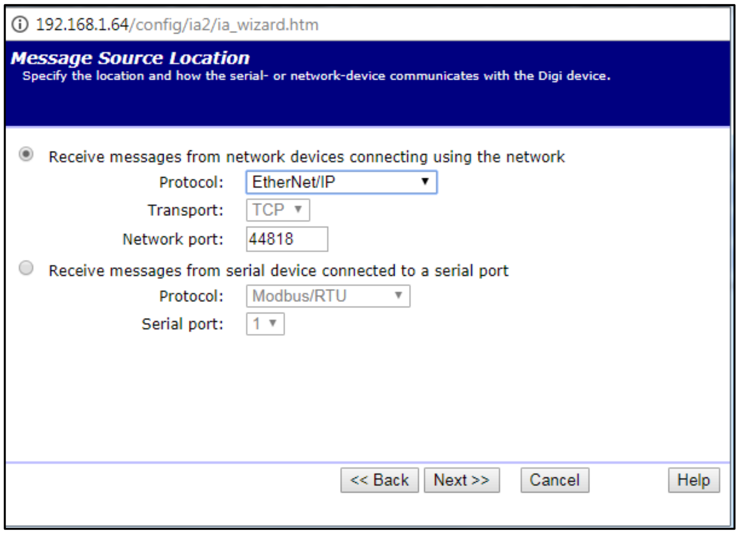
● **Protocol Mapping** (Allow multiple automation devices to communicate with each other using different protocols and families)

○ **Omron Family** (Omron Hostlink)

○ **Custom/User Defined** (Allow automation devices to communicate with each other using a custom-defined protocol not listed above)

2.2.4 Define Ethernet/IP as the master or message source.

1. Click the Receive messages from network devices connecting using the network radio button.
2. Click the **Protocol** drop-down list and select **Protocol Ethernet/IP**. The Network port automatically sets to 44818.
3. Click the **Next** button.



192.168.1.64/config/ia2/ia_wizard.htm

Message Source Location

Specify the location and how the serial- or network-device communicates with the Digi device.

● Receive messages from network devices connecting using the network

Protocol: EtherNet/IP

Transport: TCP

Network port: 44818

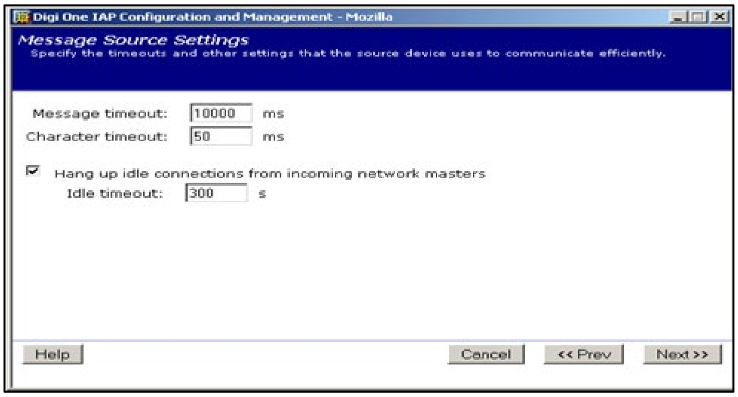
○ Receive messages from serial device connected to a serial port

Protocol: Modbus/RTU

Serial port: 1

2.2.5 Define timeouts for the incoming Ethernet/IP connection

1. Change the **Message Timeout** from 2500 ms to 10,000 ms, since ControlLogix usually waits up to 30 seconds for a response. Use the other two default settings.
2. Click the **Next** button.



Digi One IAP Configuration and Management - Mozilla

Message Source Settings

Specify the timeouts and other settings that the source device uses to communicate efficiently.

Message timeout: 10000 ms

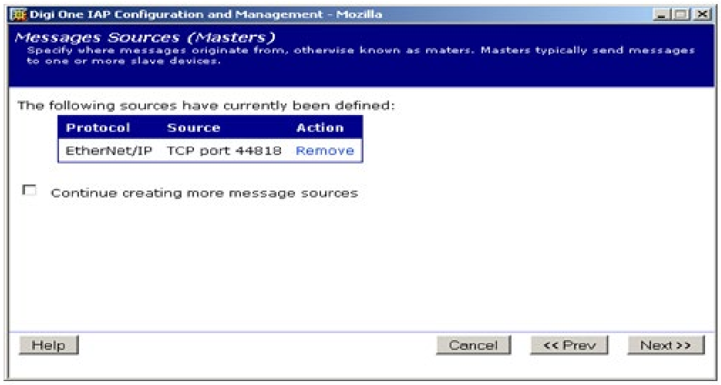
Character timeout: 50 ms

☑ Hang up idle connections from incoming network masters

Idle timeout: 300 s

2.2.6 Master summary—incoming Ethernet/IP all set

1. Click the **Continue creating more message sources** checkbox to define more masters. In this example you don't need any more, but you could loop and enable Allen-Bradley/Ethernet. This is the older CSP protocol used by pre-ControlLogix PLC5E and SLC5/05.
2. Click the **Next** button.



Digi One IAP Configuration and Management - Mozilla

Messages Sources (Masters)

Specify where messages originate from, otherwise known as maters. Masters typically send messages to one or more slave devices.

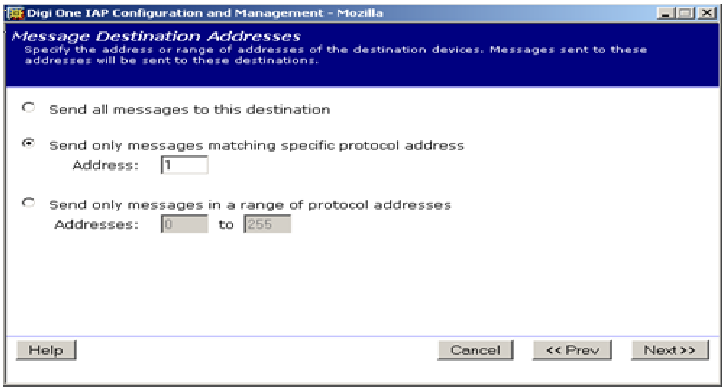
The following sources have currently been defined:

Protocol	Source	Action
EtherNet/IP	TCP port 44818	Remove

□ Continue creating more message sources

2.2.7 Define first message destination—our serial port 1 (weight bridge)

1. Click the Send only messages matching specific address radio button.
2. Enter the **1** in the **Address** textbox. This is the DST or Destination Node number you'll be setting in your ControlLogix MSG block.
3. Click the **Next** button.



Digi One IAP Configuration and Management - Mozilla

Message Destination Addresses

Specify the address or range of addresses of the destination devices. Messages sent to these addresses will be sent to these destinations.

○ Send all messages to this destination

● Send only messages matching specific protocol address

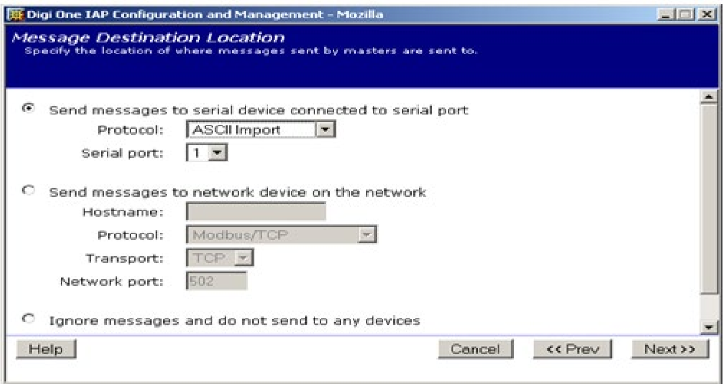
Address: 1

○ Send only messages in a range of protocol addresses

Addresses: 0 to 255

2.2.8 Set the protocol and port number

1. Click the Send messages to serial device connected to serial port radio button.
2. Click the **Protocol** drop-down list and select **Serial Port 1**. Note that address = 1 and serial port = 1 are independent. You can also send address 1 to serial port 2 if you like. You can also select a Destination Node like 23 instead of 1 within the ControlLogix MSG block.
3. Click the **Next** button.


Digi One IAP Configuration and Management - Mozilla

Message Destination Location

Specify the location of where messages sent by masters are sent to.

● Send messages to serial device connected to serial port

Protocol: ASCII Import

Serial port: 1

○ Send messages to network device on the network

Hostname:

Protocol: Modbus/TCP

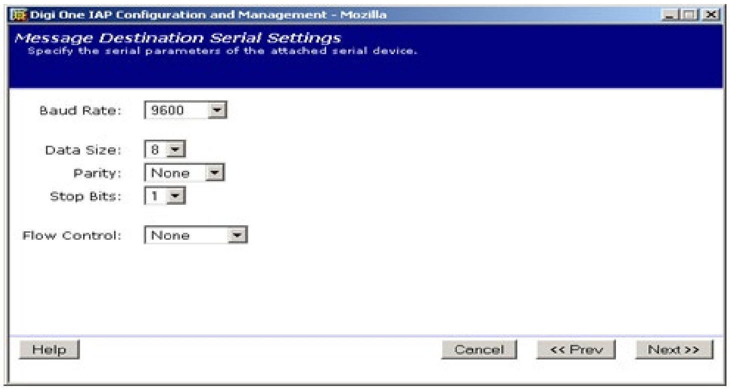
Transport: TCP

Network port: 502

○ Ignore messages and do not send to any devices

2.2.9 Set the serial port settings

1. Use the **Message Destination Serial Settings** page to enter your settings. Flow control must be **None** for most IA protocols.
2. Click the **Next** button.


Digi One IAP Configuration and Management - Mozilla

Message Destination Serial Settings

Specify the serial parameters of the attached serial device.

Baud Rate: 9600

Data Size: 8

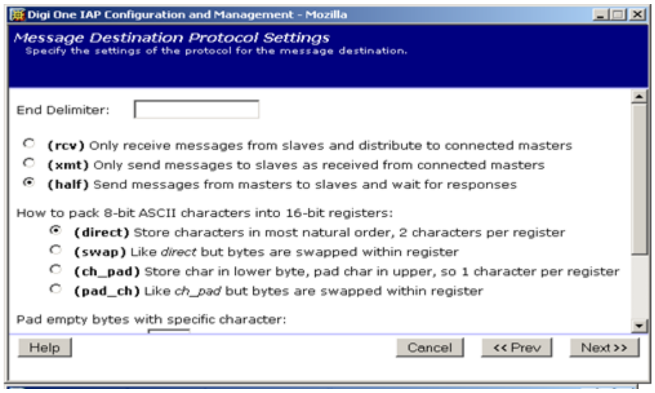
Parity: None

Stop Bits: 1

Flow Control: None

2.2.10 Set the ASCII import-specific protocol settings

1. Scroll down in this dialog, if necessary.
2. Click the **Help** button to see detailed help.
3. For this example, click the **(half) Send messages from masters to slaves** radio button.
4. Click the **wait for responses** radio button.
5. Click the **(direct) Store characters in most natural order, 2 characters per register** radio button. Use the default (**0**) for the character (ASCII numeric code).
6. Enter a preload poll string for the weight-bridge if it is fixed. The example below shows WG?\r\n, which reads the gross weight on some scales.



Message Destination Protocol Settings

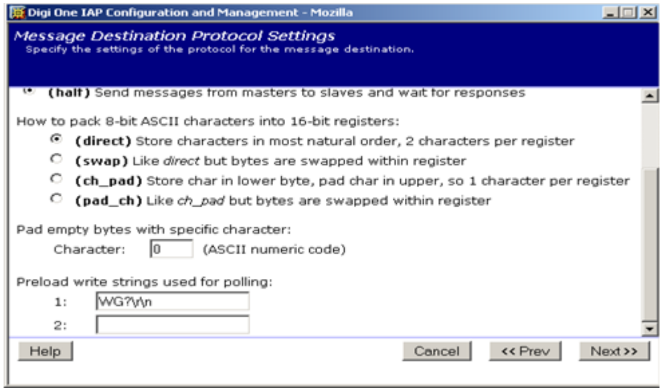
Specify the settings of the protocol for the message destination.

End Delimiter:

○ **(rcv)** Only receive messages from slaves and distribute to connected masters
○ **(xmt)** Only send messages to slaves as received from connected masters
● **(half)** Send messages from masters to slaves and wait for responses

How to pack 8-bit ASCII characters into 16-bit registers:
● **(direct)** Store characters in most natural order, 2 characters per register
○ **(swap)** Like *direct* but bytes are swapped within register
○ **(ch_pad)** Store char in lower byte, pad char in upper, so 1 character per register
○ **(pad_ch)** Like *ch_pad* but bytes are swapped within register

Pad empty bytes with specific character:



Message Destination Protocol Settings

Specify the settings of the protocol for the message destination.

● **(half)** Send messages from masters to slaves and wait for responses

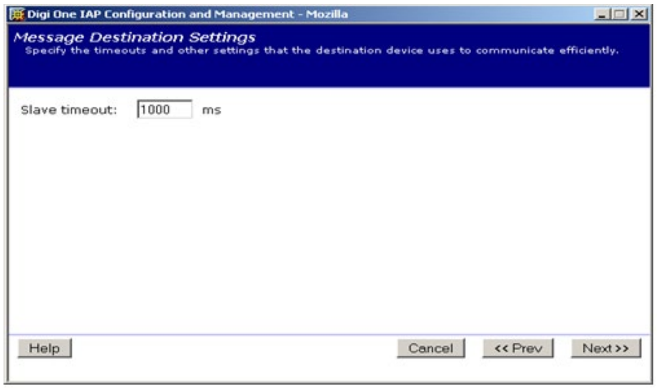
How to pack 8-bit ASCII characters into 16-bit registers:
● **(direct)** Store characters in most natural order, 2 characters per register
○ **(swap)** Like *direct* but bytes are swapped within register
○ **(ch_pad)** Store char in lower byte, pad char in upper, so 1 character per register
○ **(pad_ch)** Like *ch_pad* but bytes are swapped within register

Pad empty bytes with specific character:
Character: 0 (ASCII numeric code)

Preload write strings used for polling:
1: WG?\r\n
2:

2.2.11 Set the Slave timeout

The **Slave timeout** applies only to half-duplex or receive-only ASCII_Import modes. It is ignored in transmit-only mode. In half-duplex mode, this is the maximum time the Digi One IAP waits for the start of a slave response. Receive-only mode is used as an age-limit for received ASCII messages. If it is set to 0 there is no age-limit, or else this is the age in milliseconds the received ASCII message can wait in queue to be read. Click the **Next** button.



Message Destination Settings

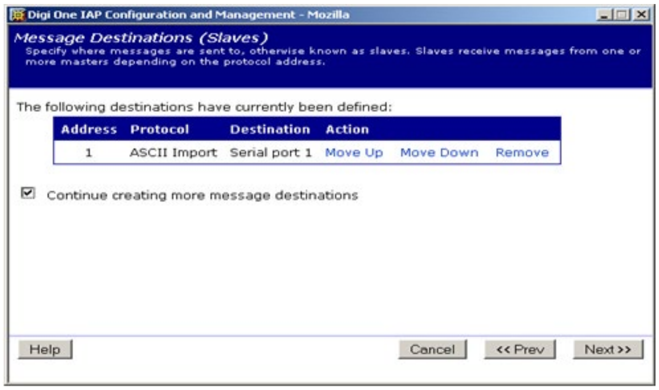
Specify the timeouts and other settings that the destination device uses to communicate efficiently.

Slave timeout: 1000 ms

2.2.12 Loop—point, repeat to set up port 2 (LED Sign)

1. Check the Continue creating more message destination checkbox.

2. Click the **Next** button. You'll set up the second port for ASCII Import and address 2. This one uses an XMT or transmit/send-only mode.



Message Destinations (Slaves)

Specify where messages are sent to, otherwise known as slaves. Slaves receive messages from one or more masters depending on the protocol address.

The following destinations have currently been defined:

Address	Protocol	Destination	Action		
1	ASCII Import	Serial port 1	Move Up	Move Down	Remove

☑ Continue creating more message destinations

2.2.13 Almost finished

1. After setting up port 2, click the **Next** button. You'll see a summary of the master and Destinations that have been set up.
2. Click the **Finish** button.

2.3 Reboot the Digi One IAP……

You can make minor changes to the Digi One IAP configuration. When changing the number/type of masters (message sources) or number/type of slaves (message destinations), it is safest to reboot. This is because these changes affect the number and type of tasks running in the Digi One IAP RTOS and, occasionally, these changes fail to take effect correctly without a reboot.

2.4 By telnet

Use HyperTerminal or telnet to log into your Digi One IAP. You must enter the IP address of your DS and the telnet port 23. Below is a text script you can cut and paste into an editor like Notepad.exe. Edit it as needed, then cut and paste again into HyperTerminal using the "Edit | Paste to Host" menu option.

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

# setup port 1 (screw term) as ascii_import (baud = 9600,8,N,1)
set port ra=1 dev=ia
set line ra=1 baud=9600 csize=8 parity=N stopb=1
set ia serial=1 protocol=ascii_import type=slave table=1 chartimeout=50ms
set ia serial=1 slavetimeout=5sec duplex=half packing=direct
prewrite1="hi?\r"

# setup port 2 (db9) as ascii_import (baud = 2400,7,E,2)
set port ra=2 dev=ia
set line ra=2 baud=2400 csize=7 parity=E stopb=2
set ia serial=2 protocol=ascii_import type=slave table=1 chartimeout=50ms
set ia serial=2 slavetimeout=5sec duplex=xmt packing=direct

# setup network for Ethernet/IP PCCC-encap incoming
set ia master=1 active=on protocol=ethernetip transport=tcp ipport=44818 table=1
```

```
# setup destination table
set ia table=1 name=table1
set ia table=1 addroute=1 active=on protocol=ascii_import
set ia table=1 route=1 protaddr=1 type=serial port=1
set ia table=1 addroute=2 active=on protocol=ascii_import
set ia table=1 route=2 protaddr=2 type=serial port=2
```

The following information is optional. Adding the third and fourth routes to your table demonstrates how ASCII devices can be located remotely attached to the Digi One SP or Digi One IA. ASCII handling is done within the local Digi One IAP, so the remote units can be a non-IAP device server. You must change the defined IP address. The Digi One IAP supports up to 64 in/out-going TCP sockets, so you should be able to connect to about 60 remote device servers and still allow the ControlLogix to connect as incoming.

```
# setup a remote DigiOne-SP or DigiOne-IA for receive-only ASCII_IMPORT
# via MSG block Destination Node = 3 & 4
set ia table=1 addroute=3 active=on protocol=ascii_import
set ia table=1 route=3 protaddr=3 type=ip ipaddress=192.168.1.60
ipport=2101
set ia table=1 route=3 connect=active duplex=rcv slvtout=0ms
set ia table=1 addroute=4 active=on protocol=ascii_import
set ia table=1 route=4 protaddr=4 type=ip ipaddress=192.168.1.61
ipport=2101
set ia table=1 route=4 connect=active duplex=xmt
```

3 Setting up the ControlLogix MSG block

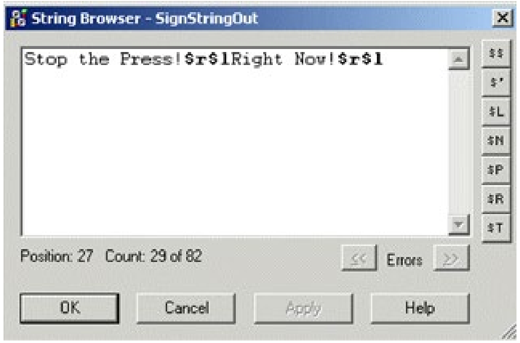
This application guide assumes you understand the basics of programming and accessing your ControlLogix PLC.

3.1 Overview

Here is what you need to do to configure the ControlLogix:

- Define tag and MSG block to load poll string for weight scale on port 1.
- Define tag and MSG block to fetch poll response for weight scale on port 1.
- Define tag and MSG block to send strings to LED Sign on port 2.

3.2 Comment on string data tags in ControlLogix



String Browser - SignStringOut

Stop the Press!rlRight Now!rl

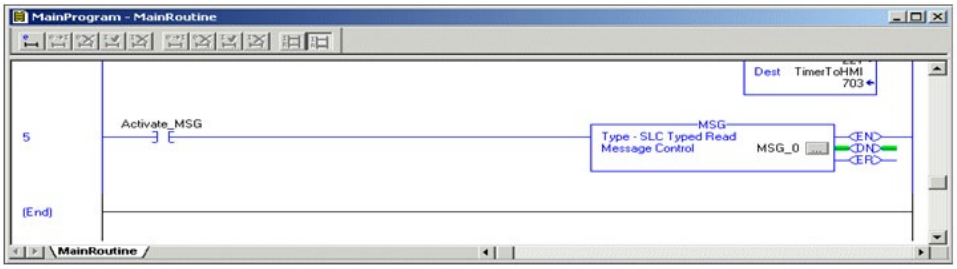
Position: 27 Count: 29 of 82

You may find it handy to use tags of type string in your ControlLogix program. The ControlLogix considers a string to consist of a length up to 82 bytes or characters of data. When entering the data, RSLogix considers the “$r” to be a <CR> or carriage return character, and considers a “$l” (lower case “L””) to be a <NL> or new-line character.

When issuing an **SLC Typed Write**, you must set the element count to **1**. You cannot write 2-string elements within the same message. This is a limitation imposed because the string data is sent as an integer file write, not as a string file write. However, you CAN embed extra rl within a single string element to output multiple short lines in transmit-only mode. However, this may produce unpredictable results in half-duplex mode.

3.3 Create your MSG block rung in RSLogix 5000

Create a MSG block using whatever trigger is appropriate. Create the tags you’ll want to read to or write from. These can be either integer or string tags.



3.4 Operation of MSG blocks for xmt-only to LED sign on port 2

To operate, your program needs to load a string into the tag used as **Source Element** in the MSG block, then trigger the message block. The MSG block creates a SLC Typed Write and uses a PCCC-encapsulation CIP class via Ethernet/IP to the Digi One IAP. Because port 2 is set for Transmit Only (xmt), the ASCII_Import driver just sends the N-file register data out the serial port.

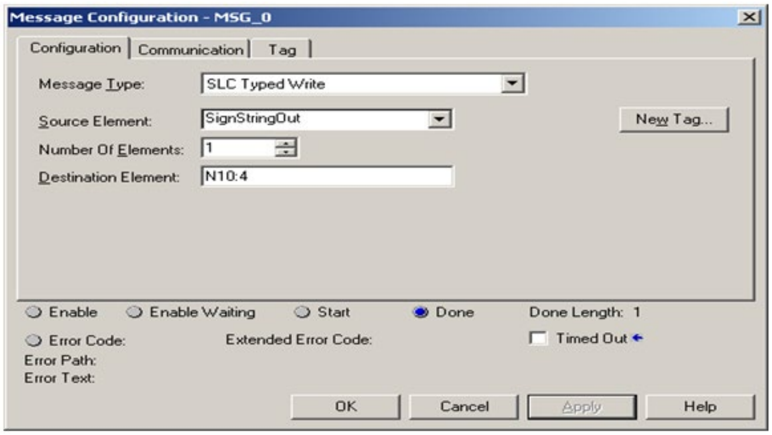
No response is expected and any response is discarded.

3.5 Configure the MSG for xmt-only to LED sign on port 2

Set the **Message Type** to **SLC Typed Write**.

Set the **Source Element** to your ControlLogix tag. You can only write one string element within each message.

Set the **Destination Element** to **N10:4**. Because this data sends immediately out of the serial port and any response is ignored, there is no need to select the second to fourth buffers. The data is not remembered or buffered at port 2. You cannot set this to an ST file; it must be an N file. You must set the starting element to **4** to confirm to Digi One IAP that this is a string data type.



Message Configuration - MSG_0

Message Type: SLC Typed Write

Source Element: SignStringOut

Number Of Elements: 1

Destination Element: N10:4

○ Enable ○ Enable Waiting ○ Start ● Done Done Length: 1

○ Error Code: Extended Error Code: □ Timed Out

Error Path:

Error Text:

3.6 Operation of MSG blocks for half-duplex to weight-scale on port 1

The half-duplex (poll-response) used for the weight-scale is a bit more complex. The **SLC Typed Write** command sends the poll, but it does NOT return any data. So an **SLC Typed Read** is required to fetch the response.

When in half-duplex mode it sees an **SLC Typed Write**, it does NOT send the data out the serial port. Instead it just remembers the data in the appropriate RAM buffer and returns an **SLC Typed Write** response. This needs to be done (and redone) any time the Digi One IAP is reset. The first two buffers can be preloaded from NVRAM in the parameters “prewrite1” and “prewrite2.” However, these two prewrites are limited to 12 characters each. Dynamically loaded polls can contain 255 characters.

To operate then, your program triggers the message block, which creates a **SLC Typed Read** and uses a PCCC-encapsulation CIP class to send it via Ethernet/IP to the Digi One IAP. Because port 1 is set for half-duplex (half), the ASCII_Import driver first sends out the appropriate buffer, then waits for a response. The **SLC Typed Read** response will be any ASCII data packed into the N-file response. If no response is received, the data will be all zeros.

3.7 Configure the “poll” MSG block for half-duplex to weight-scale on port 1

Set the **Message Type** to **SLC Typed Write**.

Set the **Destination Element** to **N10:0**, **N11:0**, **N12:0** or **N13:0** to load the first, second, third, or fourth poll buffer within the Digi One IAP. An **SLC Typed Read** to the same location returns the current contents of the buffer.

Set the **Source Element** to your ControlLogix tag (see dialog box above).

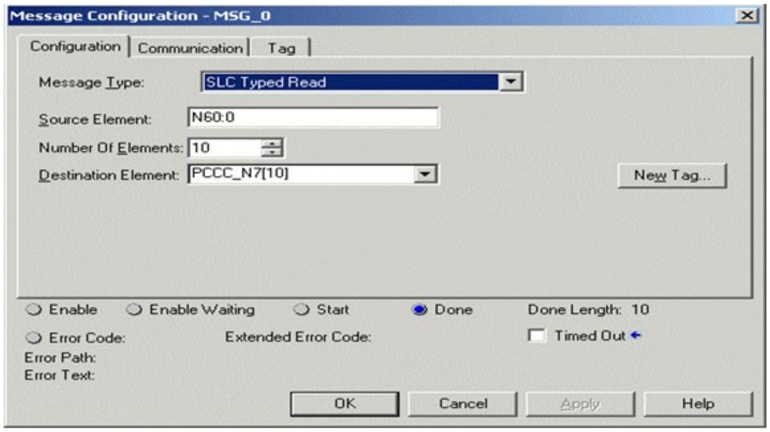
3.8 Configure the basic MSG (poll and read a response)

Set the **Message Type** to **SLC Typed Read**.

Set the **Destination Element** to the local ControlLogix tag you want to read into. In this case, it is the 11th word of an array of words called PCCC_N7.

Number Of Elements depends on data type; in this case, 10 words or 20 bytes.

Set the **Source Element** to either **N60:0**, **N61:0**, **N62:0** or **N63:0** to use the first, second, third, or fourth poll buffer within the Digi One IAP. Remember, the Digi One IAP ASCII Import function sends out one of four ASCII messages it has buffered and returns to you the response. Since this is pure integer data, set the starting element to 0.

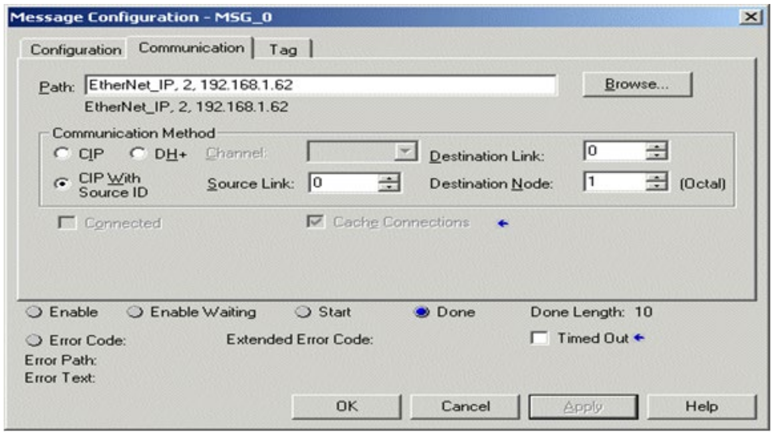


Message Type: SLC Typed Read
Source Element: N60:0
Number Of Elements: 10
Destination Element: PCCC_N7[10]

○ Enable ○ Enable Waiting ○ Start ● Done Done Length: 10
○ Error Code: Extended Error Code: □ Timed Out
Error Path:
Error Text:

Set the **Communication Path** as shown, using the Digi One IAP IP address.

Select **CIP With Source Id** and set the Destination Node to **1**. This allows the Digi One IAP to understand that this data is for port 2. Don't forget this step. It must be set to **1** or **2** to select ports on the Digi One IAP.



Path: EtherNet_IP, 2, 192.168.1.62
EtherNet_IP, 2, 192.168.1.62

Communication Method
○ CIP ○ DH+ Channel: Destination Link: 0
● CIP With Source ID Source Link: 0 Destination Node: 1 (Octal)
□ Connected ☑ Cache Connections

○ Enable ○ Enable Waiting ○ Start ● Done Done Length: 10
○ Error Code: Extended Error Code: □ Timed Out
Error Path:
Error Text:

4 How to test this application

4.1 Testing the xmt-only on port 2

Connect a computer running HyperTerminal or some other comms package to port 2 with the appropriate baud rate settings. Then, using RSLogix5000, load a string into your source tag and trigger your MSG block. You should see your ASCII message in HyperTerminal. You won't need to change any timing parameters for this test.

Add the "$r$l" (this is lower case "R" and "L") to your data so HyperTerminal shows the carriage return and new line.

4.2 Testing the half-duplex on port 1

This is a bit more complex since you'll need to increase the timeouts to accommodate human typing speed. Connect a computer running HyperTerminal or some other comms package to port 1 with the appropriate settings. Using either the Digi One IAP Web UI or telnet command line, you need to increase the timeouts to something reasonable for your typing speed. For example, a slave timeout parameter of 3000 ms or three seconds, and a char timeout parameter of something like 2000 ms, or two seconds, should be suitable.

You need to load the **poll string**. You can do this by setting the **prewrite1** parameter either by Web UI or telnet command line. To make the output nicer, you should append the <CR><NL> characters to your poll. For **prewrite1**, this could be "Poll\r\n."

Trigger your MSG block and watch HyperTerminal. If all goes well, you'll see your poll message and have three seconds to start typing a response. Each new character has to be typed within two seconds of the last. If this is too fast for you, then increase these timeouts. If all goes well, you'll see your response show up as data within your ControlLogix tag.

5 Troubleshooting and FAQ

5.1 My ASCII is showing up funny; "Stop The Press" becomes "tSpoT ehP erss"

This is controlled by the packing parameter ("packing=direct" in CLI example). The settings are referred to as **direct**, **swap**, **ch_pad**, and **pad_ch**. Because of the nature of Big versus Little-Endian designs, plus the fact that some protocols or even client/master software apps treat text differently, there is no absolute way to define this packing.

If you have "packing=direct" and you see this problem, set "packing=swap." If you have "packing=swap" and you see this problem, set "packing=direct" These two settings are the byte-swap of each other.

If you have "packing=ch_pad" and you see this problem, set "packing=pad_ch." If you have "packing=pad_ch" and you see this problem, set "packing=ch_pad." These two are the byte-swap of each other.

5.2 Can I use PLC5 or CIF style commands?

Since 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.

5.3 I'm confused about the remote file access—why N60 instead of N7? Why are integer reads aimed at N60:0 and string reads at N60:4?

The SLC Typed Read/Write includes five pieces of information—a byte count, file number, file type, starting element number, and starting sub-element number. A read of 10 words from N7:12 is coded as (20,7,'N',12,0), a read of 25 words from N60:0 is coded as (50,60,'N',0,0), and (as coded by ControlLogix MSG) a read of one string from N60:4 is coded as (88,60,'N',4,0). Notice that the read into a ControlLogix string element is not sent as file type "ST", but mimics a word read of 44 words from an “N” file. So some guess work is required for the Digi One IAP to distinguish a ControlLogix read for one string from 44 words.

No AB-style “files” exist within the Digi One IAP. Instead, the Digi One IAP uses these five pieces of information to calculate the response the ControlLogix desires. The file numbers 10-13 and 60-63 are used to select either internal or external buffers. Normally, the starting element number must be zero. The only exception is that the Digi One IAP expects you to use starting element number of **4** when handling string data. This acts as a double-check with the byte count of 88 bytes. In other words, just assuming ALL word requests for 88 bytes in a string is dangerous, so Digi One IAP handles a string request or response only if both the byte-count is 88 and the starting element is 4.

Duplex setting	CLgx data type	Elements	Reads	Writes
rcv/receive-only	Integer	Max 236 bytes	N60:0	Error
rcv/receive-only	String data	1 (is 82 chars)	N60:4	Error
xmt/transmit-only	Integer	Max 234 bytes	Error	N10:0
xmt/transmit-only	String data	1 (is 82 chars)	Error	N10:4
half-duplex (buf #1)	Integer	Max 234 bytes	N60:0	N10:0*
half-duplex (buf #1)	String data	1 (is 82 chars)	N60:4	N10:4*
half-duplex (buf #2)	Integer	Max 234 bytes	N61:0	N11:0*
half-duplex (buf #2)	String data	1 (is 82 chars)	N61:4	N11:4*
half-duplex (buf #3)	Integer	Max 234 bytes	N62:0	N12:0*
half-duplex (buf #3)	String data	1 (is 82 chars)	N62:4	N12:4*
half-duplex (buf #4)	Integer	Max 234 bytes	N63:0	N13:0*
half-duplex (buf #4)	String data	1 (is 82 chars)	N63:4	N13:4*

(*) Half-duplex note: Reading the N10-N13 remote files returns a copy of the “remembered” data waiting to be used as polls for this buffer pair.

5.4 Why is my MSG block returning a bad status?

Assuming your Ethernet is working and the ControlLogix is able to create an Ethernet/IP connection to the Digi One IAP, the things that may 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.”
- The remote file number is NOT one of the predefined: 10–13 or 60–63.
- The starting offset is NOT 0 or 4. Note that you cannot read N60:1 thinking it will skip the first two bytes of the ASCII message. The Digi One IAP does not work that way; you must use starting element 0 for integer rd/wr and 4 for string rd/wr. You will be given the entire ASCII message; any parsing or truncating of the ASCII message must be done completely by your PLC program.
- In receive-only mode, you are trying to write messages.
- In transmit-only mode, you are trying to read messages.
- In half-duplex mode, the “remembered” poll buffer is empty.