# Sending Messages From a PLC to ViewMarq



### In this Chapter...

Introduction	7-4
ASCII	7-5
Modbus	7-5
Reading the ASCII Reply when using Modbus	7-6
String Length Limitations with PLCs	7-6
Sending Strings from multiple PLCs	7-11
Modbus with Multiple Displays or other Slave Devices	7-12
CLICK PLC by AutomationDirect	7-13
RS-232	7-13
RS-485	7-13
ASCII	7-14
Modbus	7-17
Optional Error Checking	7-21
Productivity3000 PAC by AutomationDirect	7-25
Ethernet	7-25
RS-232	7-25
RS-485	7-26
Embedding the String into PAC memory	7-27
Sending Strings from Productivity3000 to the ViewMarq display by ASCII ov	er Serial 7-28/
Sending Strings from Productivity3000 to the ViewMarq display via Modbu	s7-31
Modbus Serial	7-34
Optional Error Checking	7-38
Ethernet Modbus TCP	7-42
Optional Error Checking	7-44

# Sending Messages From a PLC to ViewMarq



Do-More PLC by AutomationDirect7-47
RS-232
RS-4857-47
Ethernet
ASCII
Modbus
Modbus TCP7-63
DirectLogic PLC by AutomationDirect7-66
Sending Strings from DirectLogic to the ViewMarq display by ASCII over Serial7-66
Using the DirectLogic PRINT instruction
Using PRINTV instruction (D2-260 or D0-06 only)7-68
Embedding the String into PLC memory7-68
Sending Strings from DirectLogic to the ViewMarq display by Modbus over Serial7-70
Optional Error Checking7-73
Sending Strings from DirectLogic to the ViewMarq display by Modbus over Ethernet
PLC Cabling Chart7-83
Allen Bradley MicroLogix and SLC PLCs7-84
RS-232
ASCII
Modbus
Allen Bradley CompactLogix and ControlLogix PLCs7-103
RS-232
ASCII

## Introduction



As you create a message in the ViewMarq software, the software creates an ASCII string of commands, called a "Command String" in the ViewMarq software. This string may be pasted into your PLC's instruction or memory, then logic in the PLC will send the message to the ViewMarq.

Click the Copy button to place the command string on to the clipboard. Then paste the string into your PLC instruction.

The ViewMarq LED message display can receive an ASCII Command String by:

Protocol	Connection
ASCII	RS-232 (Port 1), RS-485 (Port 2)
Modbus RTU	RS-232 (Port 1), RS-485 (Port 2)
Modbus TCP	Ethernet



#### ASCII

When the ViewMarq display serial port and the PLC serial port are both set for ASCII, the PLC may send the Command String directly to the display's port. If the Command String is addressed to a single display, for example **<ID 1>**, and the display is configured with ASCII Reply turned on, the display will reply with an ASCII string on the same port. If the PLC can receive ASCII strings as well as send them, then you may read the ASCII Reply with the PLC. For more information about ASCII Reply see Chapter 6 – Configuring the ViewMarq LED Display.





#### Modbus

When using the Modbus RTU protocol with the ViewMarq, the same ASCII Command Strings are used but they are embedded within the data portion of the Modbus message (placed into the Modbus registers). The same ASCII Reply is also embedded within the Modbus data section of a message and can be read from a separate set of Modbus registers.

Command Strings are written to the ViewMarq Command String buffer starting at Modbus Registers 411000 (up to 256 words).

ASCII Replies are read from the ViewMarq Status Buffer starting at Modbus Registers 411500 (up to 256 words).



**NOTE:** The carriage return termination (0x0d) is still required in the Command String that is embedded with the Modbus message.



Attention!: Command Strings should be sent at least 100ms apart.

#### Reading the ASCII Reply when using Modbus

#### To accurately read the ASCII reply from the ViewMarq display, follow the steps below:

- 1) Write the Command String to the display Modbus register block starting at 411000
- 2) Monitor the value in the first Modbus register 411000
- 3)When the value in register changes to 0 (zero), this indicates the message has been processed and the ASCII Reply buffer has been updated
- 4) Read the updated ASCII Reply from the register block starting at 411500



#### String Length Limitations with PLCs

Depending on the PLC, instruction and protocol you may not be able to send an ASCII command string that is 500 characters long in one PLC ASCII or Modbus instruction. For example, the AutomationDirect PLCs have the following limitations.

PLC	Instruction	Protocol	Maximum Characters
CLICK	Send	ASCII	128
CLICK	Send	Modbus	246
P3000	ASCII Out	ASCII	128
P3000	P3000 MWX - String Modbus		128
P3000	MWX - Integers Modbus		240
Do-more	Do-more STREAMOUT		1023
Do-more	MWX	Modbus	246
DirectLogic	PRINT	ASCII	128
DirectLogic	DirectLogic VPRINT		128
DirectLogic	MWX	Modbus	250

In order to send a String greater than the limit of the PLC instruction, the string will need to be sent in multiple parts. The ViewMarq Display is looking for an <ID n> and a Termination Character \$0D (Carriage Return) before it processes its buffer. Therefore a long command string may be sent like this:

PLC Instruction 1 <ID n> Command String Part 1

PLC Instruction 2 Command String Part 2 \$0D

SAGE DISFLAT User Manual, 2nd Ed. Rev. A–MD-USEM-M

#### Sending a Command String in Multiple Parts using a CLICK PLC

#### **CLICK ASCII Send Example**

For example, the following Command String is 181 characters and is too long for a CLICK ASCII Send Instruction.

<ID 0><CLR>><WIN 0 0 287 31><POS 0 0><CJ><BL N><CS 0><GRN>
<T>AutomationDirect</T><POS 0 8><CJ><RED><T>#1 in</T><POS
0 16><CJ><T>Service</T><POS 0 24><CJ><AMB><T>12 Years in a
row</T>

It needs to be sent in two parts:

#### **Command String Part 1**

<ID 0><CLR><WIN 0 0 287 31><POS 0 0><CJ><BL N><CS 0><GRN>
<T>AutomationDirect</T>

#### **Command String Part 2**

## <POS 0 8><CJ><RED><T>#1 in</T><POS 0 16><CJ><T>Service</T> <POS 0 24><CJ><AMB><T>12 Years in a row</T>

It does not matter where the Command String is broken apart because it will not be processed by the ViewMarq Display until the Termination Character at the end of the Command String is received.

Command String Part 1

#### **Command String Part 2**

Send	Send
Com Port:       Port2         Protocol:       ASCII         COM Port Setup         Sending Data Setup	Com Port:       Port2         Protocol:       ASCII         COM Port Setup         Sending Data Setup <ul> <li>Static Text Message (MAX: 128 characters)</li> <li>Possible Message Length = 101</li> <li><pos 0="" 8=""><ci><ped><t>\$1 in   <td< th=""></td<></t></ped></ci></pos></li></ul>
Dynamic Text Message (MAX: 128 characters)         Start Address:         TXT1         Number of Bytes:         6	Dynamic Text Message (MAX: 128 characters)     Start Address:     Number of Bytes:
Termination Code (ASCII HEX code)          ⓐ 1 Character 1: \$00         2:           2 Characters ASCII Table          Byte Swap          ⓐ All           All but null	Image: Construction Code (ASCII HEX code)       Image: Construction Code (ASCII HEX code)       Image: Construction Code (ASCII Table)         Image: Byte Swap       Image: All Construction Code (Ascin to the code)       Image: All Code (Ascin to the code)       Image: All Code (Ascin to the code)
Status Flags Sending Success C2	Status Flags Sending V Success V C3
OK Cancel Help	OK Cancel Help



#### **CLICK Modbus Send Example**

When using Modbus, ASCII strings must be an even number of bytes in length because Modbus registers are 16 bits (2 bytes) long.

Since the Modbus Write instructions are limited depending on the PLC, longer ASCII strings must be sent by using successive Modbus Write instructions. This example uses the CLICK PLC which limits the Modbus Write to 246 characters.

For example, the ASCII string below is 274 characters long including the Termination Character (\$0D).

<ID 0><CLR><WIN 0 0 287 31><POS 0 0><CJ><BL N><CS
9><GRN><T>A</T> <CS 3><AMB><T>utomation</T><CS 5><GRN><T>D<//
T><CS 3><AMB><T> irect</T><POS 0 16><CJ><CS 0><GRN><T> #1
in Service</T><POS 0 24><RJ><AMB><T>12 </T><RED><T>Years<//
T><GRN><T> in</T><AMB><T> a </T><RED><T>row</T>

To send it to the ViewMarq using a Modbus Write, the ASCII string will need to be sent in two parts. The first is sent without a Termination Character to Modbus address 411000.

#### Command String Part 1 - 228 Characters

<ID 0><CLR><WIN 0 0 287 31><POS 0 0><CJ><BL N><CS
9><GRN><T>A</T> <CS 3><AMB><T>utomation</T><CS 5><GRN><T>D<//
T><CS 3><AMB><T>irect </T><POS 0 16><CJ><CS 0><GRN><T> #1
in Service</T><POS 0 24> <RJ><AMB><T>12 </T><RED><T>Years<//
T>

The second is sent with a Termination Character (\$0D) to Modbus address 411114 = [411000 + 228 characters / 2 characters per byte)].

```
Command String Part 2 - 44 characters + termination character ($0D) = 45 characters
<GRN><T> in</T> <AMB><T> a </T><RED><T>row</T>
```

Notice the Termination Character is only added to the last string. This increases the length to 45 characters. As you will see, because this length is an odd number of characters, it makes it necessary to add one to the length to keep the number even in the Modbus Write instruction. It is not shown in the string above, because it is not added by ViewMarq, but added in the PLC instruction

#### Dividing up the Command String

Each Command string may be broken at any location that creates an even length string. Consecutive strings must be sent to the very next Modbus register after the previous string. The last string may be even or odd because the length in the Modbus Write instruction can be rounded up. This extra character is allowed in the last command string because ViewMarq will only process the string up to the Termination Character (Carriage Return). Any characters following this will be ignored

If the strings are NOT written to the correct address, then they may:

- 1) Overlap causing a syntax error
- 2) Leave gaps between the parts of the ASCII string that contain unexpected characters or NULL's.
  - Unexpected characters may cause a syntax error. If a NULL is encountered by the ViewMarq, it will stop processing the string at the NULL and wait until the NULL is replaced

In the example below, the Command String has already been copied into the CLICK memory location TXT1 – TXT272 and the Termination Character (\$0D) has been copied into TXT273. (See the following Section "CLICK PLC by AutomationDirect – Modbus" for details about copying the Command String into memory.)

**Command String Part 1** 

**Command String Part 2** 

Send	Send
Com Port: Port2  Protocol: MODBUS COM Port Setup Sending Data Setup	Com Port: Port2  Protocol: MODBUS COM Port Setup Sending Data Setup
Slave ID (0-247): 2 Modbus Function Code: 16 - Write Multiple Registers Addressing Type: Modbus 984 Addressing Starting Slave Address: 41100 (400001 to 465535) Starting Master Address: 7171 Number of Master Addresses: 228 Word Swap: 0FF Status Flags Sending Success Error Exception Response (Error Code) Cital	Slave ID (0-247):       2         Modbus Function Code:       16 - Write Multiple Registers         Addressing Type:       Modbus 984 Addressing         Starting Slave Address:       411114         Starting Master Addresse:       411114         Number of Master Addresse:       16 - Write Multiple Registers         Word Swap:       DTT229         Word Swap:       OFF         Status Flags       C13         Success       C13         Error       C14         Exception Response (Error Code)
OK Cancel Help	OK Cancel Help
Send Part 1 of the Command String to the ViewMarq Command String buffer starting at Modbus address 411000	Send (Port:2) MODBUS Slave ID 2 Modbus Function Code 16 Slave Addr 411000 NO. of Master Addresses 228 Word Swap OFF Master III TXT1
Command String buffer starting at Modbus address 411114 [411000 + (228 characters / 2 characters per byte)] IBI C11	Send (Port:2) MODBUS Slave ID 2 Modbus Function Code 16 Slave Addr 411114 NO. of Master Addresses 46 Word Swap OFF Master II TXT229

7-10 VIEW User Manual, 2nd Ed. Rev. A-MD-USEM-M

#### Sending Strings from multiple PLCs

The ViewMarq Message LED message display is a slave device. If more than one PLC, master in this case, is sending strings to the display, the PLC logic must be written in such a way as to prevent the two master PLCs from interfering with each other.

Care must be taken so that one PLC has completed sending its command string before another PLC sends a command string. Once a complete command string (<ID n> and \$0D) has been received by the LED Display, time must be given to process it. A delay between complete command strings of 100ms is required

**PLC #1** 



#### Modbus with Multiple Displays or other Slave Devices.

The ViewMarq is a Standard Modbus Slave. On a multi-drop Modbus RTU network there may be multiple Modbus slave devices including one or more ViewMarq displays. If Modbus requests are being made of alternating slave devices, the ViewMarq displays require a 55ms delay between these packets. If the ViewMarq Display is polled during this time it will not respond and a timeout error will occur in the Master device.

Whenever possible, the poll rate of the master device should be set to 55ms or longer to create this delay between packets. As an example, this can be accomplished with the AutomationDirect Productivity3000 PAC by setting the "Response / Request Delay" to 55ms or higher for the RS-485 port.

For masters devices without a port delay setting the user will need to create delays between communication instructions in their PLC code.



## **CLICK PLC by AutomationDirect**

The CLICK PLC has two communication networks available, RS-232 or RS-485. Either may be used with ViewMarq.

#### **RS-232**

Connect the ViewMarq Port 1 RJ12 to the Click PLC Port 2 RJ12 using a D0-CBL serial cable.



#### **RS-485**

Connect the ViewMarq Port 2 terminals to the Click PLC Port 3 terminals using a 3-conductor RS-485 Cable, AutomationDirect Part No. L19773.



The Click PLC Send instruction may utilize ASCII or Modbus depending on how the port is configured. Both methods are shown.

#### ASCII

- 1) Using the CLICK Programming Software configure the CLICK Port 2 or Port 3 for ASCII protocol.
- 2) Set the port on the CLICK PLC Com Port to match the settings for the ViewMarq display.

Node Address (1-247): Baud Rate (bps): Parity: Stop Bit:	1 38400 • Odd •	Port2 RS-232C (Non isolation)
Parity:	Odd 👻	Caria Gaugala anadulan
Stop Bit:		(RJ12 phone jack)
Communication Data (bit):	1 • 8 •	OV +5V
Advanced Configuration		
Time-out Setting;	500 ms 🔍 🔻	
Character Time-out (2-1000ms):	2 🔺	
RTS ON Delay (0-5000ms):	0	
RTS OFF Delay (0-5000ms):	0	
Response Delay Time (0-5000ms):	0	

3) In the ViewMarq Software, type and configure a message. Copy the string in the Command String window by selecting Copy on the Command String Toolbar.

Command String	77 / 500	
<id 0=""><clr><win 0="" 287="" 31=""><pos 0=""><lj><bl n=""><cs 0=""><grn><t>Hello World</t></grn></cs></bl></lj></pos></win></clr></id>		
CLR NUM		

Take note of <ID 0> in the string above. This identifies which ViewMarq LED display is intended to display the message on a multiple display network such as RS-485. Refer to Other Tools, LED Display ID in Chapter 5. The default ID for all ViewMarq LED message displays is 1 and is set using the LED Display Configuration Dialog which is covered in Chapter 6. 4) In the CLICK Programming software use a SEND instruction as shown below.

Com Parts David	—(a)
Com Port: Port 2	
Protocol: ASCII	COM Port Setup
Sending Data Setup	
O Static Text Messa	ge (MAX: 128 characters)
<t>Hello World<td>&gt;" 00</td></t>	>" 00
Embed ASCIL C	ade Embed Memory Address Simulate
Embed Aberr e	
Dynamic Text Mes	ssage (MAX: 128 characters)
Start Address:	
Number of Byte	s;
Termination Code	
(ASCII HEX code)	1: SOD (0) 2: ASCILTable
Byte Swap	All     All but null
Status Flags	
Sending	
Success	✓ C1 (1)

- a) Select Port 2 for RS-232 (or 3 for RS-485) for the Com Port.
- b) Select the Static Text Message radio button.
- c) Click in the Text Box and press Ctrl-V on your keyboard to paste the String into the instruction.
- d) Add quotes to the beginning and end of the string.
- e) Select the Termination Code checkbox.
- f) Select 1 Character radio button.
- g) Enter \$0D or \$0A to embed a Carriage Return at the end of the string.
- h) Select an address for the success bit. For example C1.
- i) Select OK.



#### Example CLICK PLC code for sending ASCII string



**NOTE:** To prevent the string from being sent with every scan of the PLC use an EDGE triggered (or One Shot) instruction.



7

Attention!: Command Strings should be sent at least 100ms apart.



#### Modbus

- 1) Using the CLICK Programming Software set the Port on the CLICK PLC Com Port to Modbus.
- 2) Set the port on the CLICK PLC Com Port to match the settings for the ViewMarq display.

Com Port Setup Details		X
Port: Port2 Protocol: M Basic Configuration	Iodbus 🔹	Wiring Details
Node Address (1-247):	1	Port2 RS-232C (Non isolation)
Baud Rate (bps):	38400 🔻	
Parity:	Odd 👻	6 pin female modular. (RJ12 phone jack)
Stop Bit:	1 •	
Communication Data (bit);	8 🔻	
Advanced Configuration	]	TX RTS
Time-out Setting:	500 ms 🔻	V <sub>OV</sub>
Character Time-out (2-1000ms):	2	
RTS ON Delay (0-5000ms):	0	
RTS OFF Delay (0-5000ms):	0	
Response Delay Time (0-5000ms):	0	
	ОК	Cancel Help

3) In the ViewMarq Software, type and configure a message. Copy the string in the Command String window by selecting Copy on the Command String Toolbar.

Command String	77 / 500	
<id 0=""><clr><win 0="" 287="" 31=""><pos 0=""><lj><bl n=""><cs 0=""><grn><t>Hello World</t></grn></cs></bl></lj></pos></win></clr></id>		
CLR NUM		

Take note of <ID 0> in the string above. This identifies which ViewMarq LED display is intended to display the message on a multiple display network such as RS-485. Refer to Other Tools, LED Display ID in Chapter 5. The default ID for all ViewMarq LED message displays is 1 and is set using the LED Display Configuration dialog which is covered in Chapter 6.

Сору		<b>X</b>
Copy Type Single Copy Block Copy Fill	Copy Setting Source:	Sc41 : Address Error
		OK Cancel Help

4) Paste the string into a Copy command as shown.



**NOTE:** Quotation marks must be placed at the beginning and end of the String that was pasted into the "Source" field of this instruction. Enter in the beginning TXT address of the block where the String will reside. Note the ending address of the destination. This will be used in the next command.

5) Add a carriage return to the end of the string using another Copy command.



The carriage return character (entered as \$0D in the Source field), should be placed into the next TXT address after the end of the block used in the previous COPY command. In this example, the end TXT address from the block used in the previous COPY command was TXT75, so TXT76 is used in this COPY command Destination address.

Care Danta	and a	
Com Port:	Port 2 🕜 👻	
Protocol:	MODBUS	COM Port Setup
Receiving D	ata Setup	
Slave ID	(1-247):	1 🐌 🛋
Modbus	Function Code:	16 - Write Multiple Registers 🔹
Addressi	ng Type:	Modbus 984 Addressing 👻
Starting	Slave Address:	✓ 411000 (400001 to 465534)
Starting	Master Address:	/ TXT1
Number	of Master Addresses:	246 (d) (2 to 246 : Even
Word Sw	iap:	OFF v
Status Flags	e	
Sending	Ŭ	✓
Success		✓ C11
Error		✓ C12
Exceptio	n Response (Error Coo	ie) 🗸 🛄

6) In the CLICK Programming software use a SEND instruction as shown below.

- a) Select Port 2 for RS-232 (or 3 for RS-485) for the Com Port.
- b) Select the Slave ID (Node Address) for the connected ViewMarq.
- c) The Command String is written to ViewMarq Command String Buffer starting at Modbus address 411000.
- d) The number of Master Addresses may be set to the maximum possible value for the instruction.
- e) Configure any Status Flags desired for the program control.
- f) Select OK



**NOTE:** To prevent the string from being sent with every scan of the PLC use an EDGE triggered (or One Shot) instruction.



#### Example CLICK PLC code for sending ASCII string over Modbus

Copy the ViewMarq String to to TXT memory location	Conv
	Copy Single
	Src " <id 0=""><clr><win 0="" 2<="" td=""></win></clr></id>
	Des 🔟 TXT1 🔟 TXT75
Copy a Carraige Return to the next TXT memory location	
	Copy Single
	Option Suppress
	Src 📓 \$0D
Send the String to the ViewMarq Command String buffer	Des 🔟 TXT76
starting at Modbus address 411000	
IBIC1	Send (Port:2) MODBUS
└───ĨŤ└────	Modbus Function Code 16
	Slave Addr 411000 success
	Word Swap OFF
	Master 🗖 TXT1

7-20 VIEW User Manual, 2nd Ed. Rev. A-MD-USEM-M

#### **Optional Error Checking**

You may choose to read the ViewMarq status to make sure the ASCII string was received with no errors.

1) In the Click Programming software, use a Receive instruction as shown below to read the Command Block address until it equals 0. This indicates that the ViewMarq display has finished processing the command and the status is ready to be read.

Com Port: Port 2 🕘 👻	
Protocol: MODBUS	COM Port Setup
Receiving Data Setup	
Slave ID (1-247):	
Modbus Function Code:	03 - Read Holding Registers 🔹
Addressing Type:	Modbus 984 Addressing 🔹
Starting Slave Address: 🗸 🗸	411000 (400001 to 465535)
Starting Master Address: 🛛 🗸	DS1 ()
Number of Master Addresses:	1 (1 to 125)
Word Swap:	OFF 💌
Charcter Order:	Char1, Char2 💌
Status Flags 0	
Receiving	✓ ····
Error	✓ C22
Exception Response (Error Code)	· · · · · · · · · · · · · · · · · · ·

- a) Select Port 2 for RS-232 (or 3 for RS-485) for the Com Port that you previous set to Modbus.
- b) Select the Slave ID (Node Address) for the connected ViewMarq.
- c) Read the first address of the Command String Buffer, Modbus address 411000.
- d) Choose a Master Address that is an unused, Integer address (such as the DS data type) that can be compared to 0. Once this register is equal to 0, the Status Block can be read.
- e) Configure any Status Flags desired for program control.
- f) Select OK

2) Once the Command Block is equal to 0, use a Receive instruction to read the Status Block to verify that the Command String written was accepted by the ViewMarq display.

Receive	×
Com Port: Port 2 (a) -	
Protocol: MODBUS	COM Port Setup
Receiving Data Setup	
Slave ID (1-247):	1 (b)
Modbus Function Code:	16 - Write Multiple Registers 🔻
Addressing Type:	Modbus 984 Addressing 👻
Starting Slave Address: 🗸	411500 C (400001 to 465534)
Starting Master Address: 🗸	TXT1 (d)
Number of Master Addresses:	128 (2 to 250 : Even)
Word Swap:	OFF 👻
Charcter Order:	Char1, Char2 👻
Status Flags (e)	
Receiving	✓
Success	✓ C31
Error	✓ C32
Exception Response (Error Code)	· · · · · · · · · · · · · · · · · · ·
	OK 🕖 Cancel Help

- a) Select Port 2 for RS-232 (or 3 for RS-485) for the Com Port that you previous set to Modbus.
- b) Select the Slave ID (Node Address) for the connected ViewMarq.
- c) The Status String can be read from the ViewMarq display starting at Modbus address 411500.
- d) Choose an unused, available block of 128 TXT addresses.
- e) Configure any Status Flags desired for program control.
- f) Select OK.

7-22 VIEW User Manual, 2nd Ed. Rev. A-MD-USEM-M

3) Once the string in the Status Block has been read, check the value of the string for the text "OK" using the Search instruction.

Search	22
Condition Symbol Search: ☐ ✓ "OK" ≠	(a) Value
Range Starting Address: <b>(b)</b> ✓ TXT301 to	Ending Address: V TXT428
Result	Result Flag
This data register stores the address of the first data register that has the data to find.	ON: Found OFF: Not Found
Continuous Search	
One Shot (Execute one time)	Cancel Help

- a) Enter search text "OK".
- b) Enter the starting and ending addresses of the block of TXT addresses in the previous Receive instruction.
- c) Choose an available Integer address for the Result.
- d) Choose an available C address for the Result Flag.
- e) Select OK.

#### Example CLICK PLC code for checking the Viewmarq Status Block



7-24 VIEW User Manual, 2nd Ed. Rev. A-MD-USEM-M

## **Productivity3000 PAC by AutomationDirect**

The Productivity3000 PAC has three communication networks available, Ethernet, RS-232 or RS-485. Either of the three may be used with ViewMarq.

#### Ethernet

Connect the ViewMarq RJ45 Ethernet port to the Productivity3000 RJ45 Ethernet port using a Cat5e Ethernet crossover cable.



#### **RS-232**

Connect the ViewMarq RJ12 Port 1 to the Productivity3000 RJ12 RS-232 port using a D0-CBL serial cable.



#### **RS-485**

Connect the ViewMarq Port 2 terminals to the Productivity3000 RS-485 terminals using a 3 conductor RS-485 Cable, AutomationDirect Part No. L19773.



The Productivity3000 PAC may communicate with the ViewMarq LED display by ASCII, Modbus RTU or Modbus TCP.

This section discusses:

- Sending a Command String by ASCII over Serial
- Sending a Command String by Modbus over Serial
- Sending a Command String by Modbus TCP over Ethernet

#### Embedding the String into PAC memory

1) In the ViewMarq software, type and configure a message. Copy the string in the Command String window by selecting Copy on the Command String Toolbar.



2) In the Productivity Suite Programming Software Paste (Ctrl-V) the Command String into the Copy Data (CPD) Instruction as shown.

y Data (CPD)		<b>-</b> X
Source	Dectination	
<pre>&gt;cld 0&gt;<clr><win 0<="" pre=""></win></clr></pre>	0 2 ViewMarq_SW_String	•
		-
Copy Value     Cop     Copy Value     Cop     Show Instruction Con	py Binary mment	
Monitor	CK Cancel	Help



**NOTE:** Quotation marks must be placed around the message that has been pasted into the CPD instruction. ViewMarq\_SW\_String must be a String Data type in the Productivity3000 PAC.



# Sending Strings from Productivity3000 to the ViewMarq display by ASCII over Serial

The PAC port must be configured for "ASCII / Custom Protocol" in order to allow ASCII strings to be sent.

1) Go to Setup>Hardware Configuration and double-click on the P3-550 box in the Center window:

Read Configuration	hocal Base Group # 00	Hardware Components
-PAC Base Groups		Base Chassis
.ocal Base Group	Base	P3-03B
Peripheral Devices		P3-05B
		P3-08B
		P3-11B
		Discrete Input
		P3-08ND3S
		P3-16ND3
		P3-32ND3
		P3-64ND3
		P3-08NAS
		P3-16NA
		P3-16SIM
		Discrete Output
		P3-08TD1S
		P3-08TD2S
	F	P3-16TD1
		P3-16TD2
		P3-16TD3P
		P3-32TD1
	Ц	P3-32TD2
		P3-64TD1
		P3-64TD2
		DO ODTAC

2) Click the Serial Ports tab.

CPU	RS-232 (RJ-12)	RS-485 (TB Style)
Port Name	CPU-232	CPU-485
Port Security	Read/Write	Read/Write
2 Protocol	ASCII / Custom Protocol	Modbus RTU
Baud Rate	38.4K	38.4K
Node Address		1
Parity	Odd	Odd
Data Bits	8	8
Stop Bits	1	1
RTS Off Delay Time (0-5,000 msec)		
RTS On Delay Time (0-5,000 msec)		
Timeout between query a response (100-30,000 ms	and 5 x100 mse sec)	ec 5 x100 ms
Modbus Character Timeou (0-10,000 msec)	ut	
Response/Request Delay (1-5,000 msec)		
Comm Heartbeat Value (2-1,000 sec)		

- 3) Match the Baud Rate, Data Bits and Stop Bits to the serial port settings of the ViewMarq display serial port.
- 4) Choose ASCII / Custom Protocol on the Protocol selection.

5) Now that the port has been configured correctly, use the AOUT (ASCII Out) instruction as shown below to choose the String tag created previously, and to send out the serial port.

ASCII Out (AC	(דטכ			X
Serial Port	CPU-232 🗸	In Progress	VM_Send_In_Progress	•
String	ViewMarq_SW_String	Complete	VM_Send_Complete	•
	No Termination			
	I Character			
	2 Characters			
	Termination Code 1 0x d			
	Termination Code 2 0x 0			
	Byte Swap Option 💿 No Byte Swap			
	All Characters			
	All But Null Char	acters		
	Convert 0x 20 to NU	ш		
Show I	nstruction Comment			
Monito		ОК	Cancel	Help

Remember to add the one character termination code for a carriage return, 0x0d.

**NOTE:** The AOUT instruction is Edge-triggered so the String will be sent only once when the enable leg goes from low to high.

The "ASCII Reply" option in the ViewMarq should be disabled when sending ASCII strings with the AOUT instruction in Productivity3000. See Chapter 6 - Configuring the ViewMarq LED Display for more information. If the application requires more reliable error detection and handshaking consider using Modbus communications instead.

#### Example P3000 code for sending an ASCII string out the serial port

Embedd the String into he PAC memory			
	Enable	COPY DATA *CID 1> <clr><win 0="" 287="" 31=""><p05 0=""><l3><bl n=""><c5 0=""><grn><t>Hello World</t>" Type: Copy Value</grn></c5></bl></l3></p05></win></clr>	Destination ViewMarq_SW_String
Send te ASCII String out the Serial Port			
	1	ASCII OUT	
		Serial Port CPU-232 In Progress VM_Send_In_Progress	
	Enable	String ViewMarq_SW_String Complete VM_Send_Complete	
		Termination Code 1 0xD	
		Byte Swap Option No Byte Swap	
		No Null Conversion	



Attention!: Command Strings should be sent at least 100ms apart.





#### Sending Strings from Productivity3000 to the ViewMarq display via Modbus

To send a string by Modbus, a couple of steps need to be taken.

1) String length must be calculated.

2) Termination Codes must be added to the end of the string.

#### **Calculating String Length**

Add a String Length (SLEN) Instruction to move the length of the string into a Signed Integer 32 tag to be used later.

String Length (SLEN)	x
Input ViewMarq_SW_String	Length ViewMarq_SW_StringL 💌
Show Instruction Comment	
Monitor	OK Cancel Help

#### **Adding Termination Codes**

You will need to add a termination character ("\$0d") to the end of the message string. In Productivity3000, non-printable characters cannot be directly inserted into a string tag. Here are the steps to insert the characters at the end of the string:

1) Create an Unsigned Int 8 array and use a CPD instruction copy 0x0D to this array.

Source	Destination	
0x0D	CR_IntArray(1)	•
		E
		•
Copy Value O C	opy Binary	
Show Instruction C	omment	

2) Use the CPC (Copy Character) instruction to create a string of 1 character to move 1 byte into.

Copy Character (CPC)	×
Copy Type	
<ul> <li>Integer Array to String</li> </ul>	
Source CR_IntArray	Destination CR_String
Show Instruction Comment	
Monitor	OK Cancel Help

3) Combine the two strings together into a string ready to send out the port using the PKS (Pack String) instruction.

ack String (PKS)						×
Source	Length	Digits After Decimal		Destination	ViewMarq_SendString	•
ViewMarq_SW_String CR_IntArray(1)	ViewMarq_SW_StringLength		E E			
Show Instruction Comme	nt					
Monitor				ОК	Cancel	Help

4) Finally, set the destination tag string length to 128.

ne Tags	peed to be defined						
Description	Tag Name	Data Type	Retentive	String Length	Rows	Columns	Initial Value
Destination	ViewMarg_Send.	. String		12	В		
Length - Row: 1	ViewMarq_SW	. Integer, 32 Bit		· · · · · · · · · · · · · · · · · · ·			
						ок Са	ancel Help



#### Example P3000 code for sending an ASCII string out the serial port



#### **Modbus Serial**



NOTE: See previous section for details about preparing the string to send via Modbus.

The PAC port must be configured for "Modbus RTU" in order to allow raw ASCII strings encapsulated within a Modbus serial packet to be sent.

1) Go to Setup > Hardware Configuration and double-click on the P3-550 box in the Center window.

Read Configuration	bocal Base Group # 00	Hardware Componen	ts
		Base Chassis	-
Local Base Group	Base	P3-03B	
• Peripheral Devices		P3-05B	
	99. 99.	P3-08B	
		P3-11B	
		Discrete Input	
		P3-08ND3S	-
		P3-16ND3	
		P3-32ND3	
		P3-64ND3	
		P3-08NAS	
	H	P3-16NA	
		P3-16SIM	
		Discrete Output	
		P3-08TD 1S	
		P3-08TD2S	
		P3-16TD1	
		P3-16TD2	
		P3-16TD3P	
		P3-32TD1	
	Ц	P3-32TD2	
		P3-64TD1	
		P3-64TD2	
		D2-00TAC	

2) Click the Serial Ports tab.

	RS-232 (RJ-12)	RS-485 (TB Style)
Port Name	CPU-232	CPU-485
Port Security	Read/Write	Read/Write
Protocol	ASCII / Custom Protocol	Modbus RTU
Baud Rate	38.4K	38.4K
Node Address	1	1
Parity	Odd	Odd
Data Bits	8	8
Stop Bits	1	1
RTS Off Delay Time (0-5,000 msec)		
RTS On Delay Time (0-5,000 msec)		
Timeout between query and response (100-30,000 msec)	5 x100 msec	5 x100 mse
Modbus Character Timeout (0-10,000 msec)		
Response/Request Delay (1-5,000 msec)		
Comm Heartbeat Value (2-1,000 sec)		

- 3) Match the Baud rate, Parity, Data bits and Stop bits to the serial port settings of the ViewMarq display serial port.
- 4) Choose Modbus RTU on the Protocol selection.

Now that the port has been configured correctly, use a Modbus Write (MWX) Instruction to write the data over as shown.

Modbu	us Write	(MWX)										x
						_	In Proc	ress	VM Write InPr	ogress	<b>•</b>	1
© E	thernet P	ort CPU	ETH-Ex	t		<b>-</b>	Com	plete	VM Write Com	nlete		
	IP	Address					Com	piece		- La la		
	TCP Por	t Number	502	]			Suc	cess	VM_VVrite_Com	piete	• •	
S	lave Node	e Number		255 (1	Default=3	255 <b>)</b>	I	Error	VM_Write_Erro	r	• III	
@ S	erial Port	CPU-23	2			•	Tim	eout	VM_Write_Erro	r	-	
s	lave Nod	e Number		1 (	Default=	1) F	Except Response S	tring	VM_WriteExcep	otionRe	▼	
<b>A</b>	utomatic	Polling	eve	ery	100 mse	ec.	poll offse	t	0 msec			
S	kip execu	ition if buf	fer is gre	eater tha	n	75 % fi	11					
V 1	Word Swap Slave Modbus Starting Address 11000 + 400000											
M	lap 32 bit	data to 1	5 bit		0	1odbus D	ecimal Add	Iressin	g			
					© 2	ero Base	ed Modbus	Addre	ssing			
Mode	ous Funct	ion Code	16: Wri	te Multipl	e R •	•						
0	Non-Arra	зу		Nu	mber of 1	Tags 📄	15 🔷					
	Tag I	Name Map	ping		Tag							
				1				<u> </u>				
				3			-					
				4								
				5				Ŧ				
C	Array	Array Na	ne			•	Start	ting In	dex 1	End Inc	lex	1
	String	String Na	me Viev	vMarq_S	endString		Num	ber of	Characters	128		
	🗸 Byte	e Swap					(E	ven N	umber Only)			
S	how Instr	ruction Co	nment									
M	onitor	]							ок Са	ancel	He	lp

- 1) Choose the Serial Port option and select which CPU the message will be sent from.
  - a) The Slave Node Number should match the node in the LED Display Configuration.
  - b) Word Swap and Byte Swap should be checked assuming the selections are Off in the ViewMarq display.
  - c) Slave Modbus Memory Starting Address is the location of the Command Block within the ViewMarq display (400000+11000 = 411000).
  - d) The Function Code should be set to 16 Write Multiple Registers.


**NOTE:** The Productivity3000 MWX instruction uses the Modicon style addressing where the highest digit (4 in this case) is the Modbus data type indicator (Holding Registers).

- e) Choose the "String" option at the bottom and select the String tag previously created in the Pack String Instruction.
- f) The Number of Characters should be set at least as high as the Character Count in the ViewMarq Software Command String Viewer.

In this example, 77 would be sufficient, but using the maximum of 128 will also work correctly. ViewMarq ignores any data after the 0D.

g) Create Tags for the "In Progress", "Complete", "Success", "Error", "Timeout" and "Exception Response String" fields to ensure that the Modbus message was configured correctly.

# **Optional Error Checking**

After the Modbus Write is successful, the ViewMarq display will process the message. When the display has finished processing the message it will clear the Command Block (411000).

1) The next step of the logic should be to read the first element of Command Block using a Modbus Read (MRX) Instruction until it reads 0.

Modbus Read (MRX)			X
Ethernet Port     CPU-ETH-Ext	<b></b>	In Progress	VM_ChkCMD_InProgre 🔻 🛄
IP Address		Complete	VM_ChkCMD_Complet
TCP Port Number 502		Success	VM ChkCMD Success 👻
Slave Node Number 2	55 (Default=255)	[	
Serial Port     CPU-232	•	Error	
Slave Node Number	1 (Default=1)	Timeout	VM_Chk_Timeout
	R	Exception esponse String	VM_ChkCMD_ExResp 🔻
Automatic Polling ever	y 100 msec	ooll offset	0 msec
Skip execution if buffer is grea	ater than 75 % full		
✓ Word Swap Map 16 bit data to 32 bit	Slave Modbus Sta	rting Address imal Addressing	11000 + 400000
Modbus Function Code 3: Read I	Holding Reg	L5 🔶	ng
Tag Name Mapping	Tag		
	1 CMD_BLK_Reg1 2 3 4 5		
🔘 Array Array Name	•	Starting Inde	x 1 End Index 1
String String Name	•	Number of Ch	aracters 2
Byte Swap		(Even Numb	er Only)
Show Instruction Comment			
Monitor		ОК	Cancel Help

- a) Slave Modbus Starting Address is the location of the Command Block within the ViewMarq display (400000+11000 = 411000).
- b) The Function Code should be set to 3 Read Holding Registers.



**NOTE:** The Productivity3000 MRX instruction uses the Modicon style addressing where the highest digit (4 in this case) is the Modbus data type indicator (Holding Registers).

c) Choose the "Non-array" option at the bottom and create an Unsigned Integer 16 tag to read the first register of the Command Block into.

- d) Keep executing this MRX command until the first register of the Command Block reads 0.
- 2) After the first register of the Command Block returns a 0, the Reply Status Block should be read using another MRX instruction.

Modbus Read (MRX)			×
Ethernet Port     CPU-ETH-Ext	T	In Progress	M_RdStat_InProgress 🔻
IP Address 192.168.10	0.14	Complete	VM_RdStat_Complete
Slave Node Number 2	55 (Default=255)	Success	VM_RdStat_Success
Serial Port CPU-232	▼	Error	VM_RdStat_Error
Slave Node Number	1 (Default=1)	Timeout	VM_RdStat_Timeout
	F	esponse String	VM_RdStat_ExRespon ▼
Automatic Polling every	y 100 msec	poll offset	0 msec
Skip execution if buffer is grea	ter than 75 % full		
Vord Swap	Slave Modbus Sta	rting Address	11500 + 400000
Map 16 bit data to 32 bit	Modbus Dec	imal Addressing	
	C Zero Based	Modbus Address	sing
Modbus Function Code 3: Read H	Holding Reg 👻		
Tag Name Mapping	Number of Tags	15	]
	1	*	
	2		
	4		
l	5	*	
🔘 Array Array Name	•	Starting Inde	ex 1 End Index 1
String String Name ViewMa	arq_StatusStrin; 💌	Number of Ch	haracters 128
📝 Byte Swap		(Even Num	ber Only)
Show Instruction Comment			
Monitor		ОК	Cancel Help

- a) Slave Modbus Starting Address is the location of the Reply Status Block within the ViewMarq display. (400000+11500 = 411500).
- b) Word Swap and Byte Swap should be checked assuming the selections are Off (default) in the ViewMarq display.
- c) The Function Code should be set to 3 Read Holding Registers.



**NOTE:** The Productivity3000 MRX instruction uses the Modicon style addressing where the highest digit (4 in this case) is the Modbus data type indicator (Holding Registers).

- d) Choose the "String" option at the bottom and create a String tag to read the Status info into. Choose 128 characters as the length.
- 3) Once the string in the Reply Status Block has been read, check the value of the string for the text "OK" using the Find instruction.

Find String (FNDS)	×
Source ViewMarQ_StatusStrin 💌	Found Index ViewMarq_StatusOK_I 👻
From Index 0 🔹 📖	Success ViewMarq_StatusOK 🔻
To Index 127 🔹	
Instance 🔹 🐨	
Search For Value	
String     OK	
🔿 Tag Value 🔍 🗸	
─ Hex Data 0x (1 or 2 bytes)	
Show Instruction Comment	
Monitor	OK Cancel Help

a) Select the Source String where the Status Reply was stored.

b) Enter a Found Index Tag; this tag is required, but not important in this case.

- c) Enter the search Range from 0 to 128.
- d) Enter a Tag for the Success bit. This tag will be on if "OK" is found.
- e) Enter search text "OK".
- f) Select OK.



#### Example P3000 Code for writing an ASCII string to ViewMarq over Modbus Serial



#### **Ethernet Modbus TCP**

Sending messages from the Productivity3000 to the ViewMarq display via Modbus TCP is the same as sending messages over Modbus RTU with one change to the MRX and MWX instructions.

Instead of choosing the "Serial Port" option in the MWX and MRX instructions, choose the "Ethernet Port" option and enter in the IP address of the ViewMarq display. Set the TCP Port Number to the port you have selected on your ViewMarq (502 by default) and leave the Slave Node Number as 255.

1) With the Ethernet port properly configured in the Productivity3000 PAC, use a Modbus Write (MWX) Instruction to write the data over as shown.

Modbus Write (MWX)				X	
Ethernet Port CPU-ETH-Ext	-	In Progress	VM_Write_InProgress	•	
IP Address 192.168	8.10.14	Complete	VM_Write_Complete		
TCP Port Number 502		Success	VM_Write_Complete		
Slave Node Number	255 (Default=255)	Error	VM_Write_Error		
Serial Port CPU-232		Timeout	VM_Write_Error	•	
Slave Node Number	1 (Default=1) Re	Exception esponse String	VM_WriteExceptionRe	•	
Automatic Polling even	ry 100 msec	poll offset	0 msec		
Skip execution if buffer is gre	ater than 75 % ful				
Word Swap	Slave Modbus St	arting Address	11000 + 400000	)	
	Modbus De C Zero Based	d Modbus Addressir	ig Issing		
Madhua Eurotian Codo 16: Write	a Multipla D				
Non-Array	Number of Tags 1	5			
Tag Name Mapping	Tag				
	1	*			
	3				
	4				
	5	+			
🔘 Array Array Name	▼	Starting In	dex 1 End Inde	ex 1	
String String Name View	O String String Name ViewMarq_SendString      ✓ Number of Characters 128				
👿 Byte Swap	✓ Byte Swap (Even Number Only)				
Show Instruction Comment					
Monitor			OK Cancel	Help	

- a) Enter in the IP address of the ViewMarq display.
- b) Set the TCP Port Number to the port you have selected on your ViewMarq (502 by default) and leave the Slave Node Number as 255.

- c) Word Swap and Byte Swap should be checked assuming the selections are Off (default) in the ViewMarq display.
- d) Slave Modbus Memory Starting Address is the location of the Command Block within the ViewMarq display (400000+11000 = 411000).
- e) The Function Code should be set to 16 Write Multiple Registers.



**NOTE:** The Productivity3000 MRX instruction uses the Modicon style addressing where the highest digit (4 in this case) is the Modbus data type indicator (Holding Registers).

- f) Choose the "String" option at the bottom and select the String tag previously created in the Pack String instruction.
- g) Number of Characters will be 128 to capture the entire String Length in this case.



**NOTE:** Be sure to select the "Byte Swap" checkbox option when writing String data to the ViewMarq display from the Productivity3000.

h) Create Tags for the "In Progress", "Complete", "Success", "Error", "Timeout" and "Exception Response String" fields to ensure that the Modbus message was configured correctly.

# **Optional Error Checking**

After the Modbus Write is successful, the ViewMarq display will process the message. When the display has finished processing the Message it will clear the Command Block (411000).

1) The next step of the logic should be to read the first element of Command Block using a Modbus Read (MRX) Instruction until it reads 0.

Modbus Read (MRX)				x
Ethernet Port CPU-ETH-Ext	•	In Progress	VM_ChkCMD_InProgre 🔻	
IP Address 192.168.10	0.14	Complete	VM_ChkCMD_Complet -	
TCP Port Number 502 Slave Node Number 2	55 (Default=255)	Success	VM_ChkCMD_Success 🔻 .	••
Certial Death     COUL 222		Error	VM_ChkCMD_Error 👻 🗔	••
Slave Node Number	1 (Default=1)	Timeout	VM_Chk_Timeout	••
		Exception Response String	VM_ChkCMD_ExResp 👻 🗔	••
Automatic Polling even	y 100 msec	poll offset	0 msec	
Skip execution if buffer is grea	ater than 75	% full		
Word Swap	Slave Modb	us Starting Address	11000 + 400000	
Map 16 bit data to 32 bit	Modbe	us Decimal Addressing		
	🔘 Zero B	Based Modbus Address	sing	_
Modbus Function Code 3: Read H	Holding Reg 👻			
Non-Array	Number of Tags	15 🗢		
Tag Name Mapping	Tag		]	
1 CMD_BLK_Reg1				
	2			
	4			
	5	÷		
🔘 Array Array Name	-	Starting Inde	ex 1 End Index	1
String String Name	-	Number of C	haracters 2	
Byte Swap		(Even Num	ber Only)	
Show Instruction Comment				
Monitor		ОК	Cancel Help	

- a) Slave Modbus Starting Address is the location of the Command Block within the ViewMarq display (400000+11000 = 411000).
- b) The Function Code should be set to 3 Read Holding Registers.



**NOTE:** The Productivity3000 MRX instruction uses the Modicon style addressing where the highest digit (4 in this case) is the Modbus data type indicator (Holding Registers).

- c) Choose the "Non-array" option at the bottom and create an Unsigned Integer 16 tag to read the first register of the Command Block into.
- d) Keep executing this MRX command until the first register of the Command Block reads 0.

2) After the first register of the Command Block returns a 0, the Reply Status Block should be read using another MRX instruction.

Modbus Read (MRX)			X
Ethernet Port CPU-ETH-Ext	•]	In Progress	M_RdStat_InProgress 🔻
IP Address 192, 168, 10	0.14	Complete	VM_RdStat_Complete
Slave Node Number 2	55 (Default=255)	Success	VM_RdStat_Success
Serial Port CPU-232		Error	VM_RdStat_Error
Slave Node Number	1 (Default=1)	Timeout	VM_RdStat_Timeout 💌
		Exception Response String	VM_RdStat_ExRespon 💌
Automatic Polling ever	y 100 msec	poll offset	0 msec
Skip execution if buffer is grea	ter than 75 % fi	1	
Word Swap	Slave Modbus S Modbus D	tarting Address ecimal Addressing	11500 + 400000
	Zero Base	d Modbus Address	sing
Modbus Function Code 3: Read H	lolding Reg 👻		
Non-Array	Number of Tags	15 🗢	
Tag Name Mapping	Tag 1 2 3 4 5		
O Array Array Name	▼	Starting Ind	ex 1 End Index 1
String String Name ViewMa	arq_StatusStrin; 💌 🗔	Number of C	haracters 128
📝 Byte Swap		(Even Num	ber Only)
Show Instruction Comment			
Monitor		ОК	Cancel Help

- a) Slave Modbus Starting Address is the location of the Command Block within the ViewMarq display (400000+11500 = 411500).
- b) Word Swap and Byte Swap should be checked assuming the selections are Off in the ViewMarq display.
- c) The Function Code should be set to 3 Read Holding Registers.



**NOTE:** The Productivity3000 MRX instruction uses the Modicon style addressing where the highest digit (4 in this case) is the Modbus data type indicator (Holding Registers).

d) Choose the "String" option at the bottom and create a String tag to read the Status info into. Choose 128 characters as the length.



7-46 VIEW MESSAGE DISPLAY User Manual, 2nd Ed. Rev. A-MD-USEM-M

#### Example P3000 code for writing an ASCII string to ViewMarq over Modbus Ethernet

# **Do-More PLC by AutomationDirect**

The Do-More PLC has three communication networks available; RS-232, RS-485 and Ethernet. Any of the three may be used with the ViewMarq LED display.

# **RS-232**

Connect the ViewMarq RJ12 Port 1 to the Do-more H2-DM1E CPU RJ12 port or H2-SERIO-4 RJ12 port using a D0-CBL serial cable.



# **RS-485**

Connect the ViewMarq Port 2 terminals to the Do-more H2-SERIO-4 RS-485 terminals using a 3 conductor RS-485 Cable, AutomationDirect Part No. L19773





#### Ethernet

Connect the ViewMarq RJ45 Ethernet port to the Do-more H2-DM1E Ethernet port using a Cat5e Ethernet cable.



# ASCII

1) Use the Do-More Programming Software to configure the PLC port for "General Purpose" in order to allow ASCII strings to be sent. Go to PLC>System Configuration and open the CPU Configuration dialog shown below and select "General Purpose".



2) Click on the "Device Settings" in this window to match the Baud Rate, Data Bits, Stop Bits, and Parity to the settings of the serial port in the ViewMarq display:

Edit Serial Port Settings				
Device Name:	IntSerial			
Port Settings				
Baud Rate:	38400 💌			
Data Bits:	8 💌			
Stop Bits:	1 💌			
Parity:	Odd 💌			
Transmit Control:	Unconditional 💌			
RTS Control:	Follows Transmitter 💌			
ОК	Cancel			

#### Sending messages to the ViewMarq Display with the Do-More PLC

1) In the ViewMarq software, type and configure a message. Copy the string in the Command String window by selecting Copy on the Command String Toolbar.



- 2) In the Do-More Programming Software use the STRPRINT command to embed the Command String into Do-More PLC memory.
- 3) Paste the Command into a STRPRINT instruction.

✓X № ?	0
STRPRINT	Print to String
Print to	SL0 °
Append to String	and offer each term
Print Script	ace aller each term
" <id 1=""><clr><win 0="" 20<br="">0&gt;<lj><bl n=""><cs 0=""><gi World\$0D"</gi </cs></bl></lj></win></clr></id>	87 31> <pos 0="" a<br="" •="">RN&gt;<t>Hello</t></pos>
	$\overline{\mathbf{v}}$



Attention!: Quotation marks must be placed around the message that has been pasted into the STRPRINT instruction. You will also need to add a termination character ("\$0d") to the end of the message string. Use the "\$L" memory type as it allows for more characters (256).





**NOTE:** The STRPRINT instruction is an "Edge Triggered" instruction (as indicated by the Gray arrow) so if any changes within the text of the instruction (such as a dynamic variable), it will need to be re-triggered. If the data is changing often, consider using a transitioning bit within the contact such as the ST5 (100ms toggle) bit.

## Sending the String to the ViewMarq

Now that the port has been configured correctly and the Command String is embedded into the memory, use the STREAMOUT instruction to choose the String to send out the serial port. Ensure that the "Device" selected is for the serial port that was configured earlier.

	0
STREAMOUT	- Stream Out Data to Device
Device	@IntSerial • 🗸
Data Source	
<ul> <li>String Structure</li> </ul>	SL0
O Numeric Data Block	Create Byte Buffer
Buffer Start	D0
Number of Bytes to Output	64
Endian Settings Swap Byte Swap Word	
Flush INPUT device first	
On Success: . Set bit . JMP to Stage	e C10 °
On Error: 📀 Set bit 🔿 JMP to Stage	e C11 •



**NOTE:** The STREAMOUT instruction is an "Edge Triggered" instruction so that the enable leg logic must transition from OFF to ON for every message being sent to the display.

## Reading the ASCII Reply from the ViewMarq (Optional)

If the "ASCII Reply" option is enabled in the "LED Display Configuration" settings AND you are sending to a single ID (example <ID 1>), you may use a STREAMIN instruction to receive the reply from the display and load into a String as shown.



**NOTE:** An ID of 0 causes the Viewmarq Display to not reply.

✓×≈?		0
STREAMIN	Stream in Data from I	Device
Device	@IntSerial	• •
Complete when	,	
✓ Length is 128	bytes OR	
Delimiter(s) received OR		
1 - OxOD • © Exact sequ	ence limiter(e)	
	iter(e) from Output Otring	
	iter(s) from Output String	
Network Timeout 100	ms	
Advanced		
Data Destination		
<ul> <li>String Structure</li> </ul>	SL1	<u> </u>
C Numeric Data Block	Create Byte Buffe	
StartAddress	D0	
Buffer Size in Bytes	64	<b>-</b>
Number of Bytes Read	D1	
Endian Settings		
On Success: (• Set bit () JMP to S	stage C13	
On Error: . Set bit . JMP to S	stage C14	
✓X¤?	0	
STRCMP		
First String	GL1 °	
Case Sensitive		
<ul> <li>Yes, upper/lower cas</li> </ul>	se are different	
C No, upper/lower cas	e are equal	
Set If Equal	C15 °	
Set If Less Than	02	
Set If Greater Than	02	
Second String or Text		
"OK"		
	~	

Use the String Compare instruction to check the string for the text "OK" and set the discrete flag.

**NOTE:** The STREAMIN instruction is an "Edge Triggered" instruction so that the enable leg logic must transition from OFF to ON for every message being received into the serial port.



Attention!: Command Strings should be sent at least 100ms apart.

#### Example Do-more logic for writing an ASCII string to ViewMarq over ASCII



7

# Modbus

# Sending Strings from Do-More to the ViewMarq display via Modbus Serial

1) Use the Do-More Programming Software to configure the PLC port for Modbus RTU client in order to send Strings encapsulated within a Modbus RTU message to the ViewMarq display. Go to PLC>System Configuration and open the CPU Configuration dialog shown below and select "Modbus RTU Client".



2) Click on Device Settings to configure the Baud Rate, Parity, Data bits and Stop bits to the settings that match the serial port on the ViewMarq display.

Edit Modbus RTU Client	Settings X
Device Name:	IntSerModbusClient
Modbus Protocol Setting	gs
Timeout:	1000 ms
Retries:	2
Inter-packet Delay:	1750 us
Port Settings	
Baud Rate: 3	8400 🔹
Data Bits: 8	▼
Stop Bits: 1	<b>•</b>
Parity:	bbd
Transmit Control:	Inconditional 💌
RTS Control:	ollows Transmitter 💌
ОК	Cancel

#### To send messages to the ViewMarq display with the Do-More PLC:

#### Embed the String in PLC Memory

1) In the ViewMarq software, type and configure a message. Copy the string in the Command String window by selecting Copy on the Command String Toolbar.

Command String	77 / 500	
<id 0=""><clr><win 0="" 287="" 31=""><pos 0=""><lj><bl n=""><cs 0=""><grn><t>Hello World</t></grn></cs></bl></lj></pos></win></clr></id>		
CLR NUM		

- 2) In the Do-More Programming Software use the STRPRINT command to embed the Command String into Do-More PLC memory.
- 3) Paste the Command into a STRPRINT instruction.





Attention!: Quotation marks must be placed around the message that has been pasted into the STRPRINT instruction. You will also need to add a termination character ("\$0d") to the end of the message string. Use the "SL" memory type as it allows for more characters (256).





**NOTE:** The STRPRINT instruction is an "Edge Triggered" instruction (as indicated by the Gray arrow) so if any changes within the text of the instruction (such as a dynamic variable), it will need to be re-triggered. If the data is changing often, consider using a transitioning bit within the contact such as the ST5 (100ms toggle) bit.

#### Move the String to V-Memory

1) In the Do-More PLC, Strings cannot be directly accessed using Modbus commands so the data must be moved into integer memory first. To do this, use a STRGETB command.



- 2) This instruction will move the characters from the String and put them into V0. We use the .Length member of the String in the length field so that if our String size changes, it will still move the correct amount over.
- 3) Next adjust the value to convert it from bytes to words since the V-memory type is 16 bit and that is what is required to be sent on Modbus. So divide the byte count by 2 (2 bytes per word) and then add 1 in case the String length works out to an odd number of bytes so that 1 character of the String doesn't get chopped off.



## Sending the String to the ViewMarq

1) Now the data is prepared to be written over Modbus and with the calculated request size. Use a MWX instruction to write the data over.

✓X <sup>∞</sup> ?	0	
MWX	Modbus Network Write	
Device	@IntSerModbusClient 💌	
Modbus/TCP addressing		
IP Address	127 . 0 . 0 . 1	
TCP Port Number	502	
Unit ID	1 *	
Function Code	16 - Write Multiple Registers 💌	
Modbus Address 4 + offset value entered below		
To Modbus Offset Address	11000 *	
Number of Modbus Registers	V1000 °	
From Do-more Memory Address or C	Constant V0	
Continuous on Power Flow at Interval		
🖸 Constant 00 h	00 m 00 s 000 ms	
C Variable DO	ms	
On Success: 💿 Set bit 🔿 JMP to	o Stage C3 °	
On Error: 🙃 Set bit C JMP to	o Stage C4 *	
Exception Response	D0	

- a) The Unit ID should match the node address setting of the serial port configuration in the ViewMarq LED Display Configuration.
- b) The Function Code should be set to 16 Write Multiple Registers.
- c) The Modbus Offset Address is the location of the Command Block within the ViewMarq display and is 11000.
- d) Number of Modbus Registers should contain the V-memory location that was the result of our calculated Modbus request size from above.
- e) From Do-More Memory Address should be the resulting memory location of the STRGETB instruction that contains the String data converted to Integer.



**NOTE:** The enable options for the Modbus Network Write instruction. It is recommended to use the "Once on Leading Edge" option with this instruction when writing to the ViewMarq display. Subsequent actions (explained below) should be taken before sending another message.

7-56 VIEW MERSAGE DISPLAY User Manual, 2nd Ed. Rev. A-MD-USEM-M



#### Reading the ASCII Reply from the ViewMarq (Optional)

After the Modbus Write is successful, the ViewMarq display will process the message. When the display has finished processing the Message it will clear the Command Block.

1) The next step of the logic should be to read the first element of Command Block until it reads 0.

/ <mark>X</mark> ¤?	(
MRX	Modbus Network Read
Device	@IntSerModbusClient 🔹
Modbus/TCP addressing	
IP Address	127 . 0 . 0 . 1
TCP Port Number	502
Unit ID	1
Function Code 3	- Read Holding Registers 🛛 🔻
Modbus Address	4 + offset value entered below
From Modbus Offset Address	11000
Number of Modbus Registers	1
To Do-more Memory Address	V400
Enable     Once on Leading Edge     Continuous on Power Flow at	Interval
Constant 00 h 00	) m 00 s 000 ms
C Variable D0	ms
On Success:      Set bit      JMP to	Stage C6
On Error: 💿 Set bit 🔘 JMP to S	Stage C7
Exception Response	D0

- a) Unit ID should be the same as the previous MWX instruction.
- b) Function Code should be 3 Read Holding Registers.
- c) From Modbus Offset Address is still 11000 which is the Command Block of the display.
- d) Number of Modbus Registers should be 1.
- e) The Do-More Memory Address should be an available, unused memory location. This location should be checked after every read until it goes to 0.

7-58 VIEW MESSAGE DIEPLAW User Manual, 2nd Ed. Rev. A-MD-USEM-M

2) After the first register of the Command Block returns a 0, the Status Block should be read.

✓X <sup>∞</sup> ?	(
MRX	Modbus Network Read
Device	@IntSerModbusClient 💌
Modbus/TCP addressing	,
IP Address	127 . 0 . 0 . 1
TCP Port Number	502
Unit ID	1 •
Function Code	3 - Read Holding Registers 💌
Modbus Addre	ess 4 + offset value entered below
From Modbus Offset Address	11500 *
Number of Modbus Registers	64 <b>°</b>
To Do-more Memory Address	V200 °
Enable Once on Leading Edge C Continuous on Power Flow a	at Interval
Constant 00 h	00 m 00 s 000 ms
C Variable D0	ms
On Success:  Set bit  JMP to On Error:  Set bit  JMP to Exception Response	o Stage C1 ° o Stage C2 ° D0

- a) Unit ID should be the same as the previous MWX and MRX instructions.
- b) Function Code should be 3 Read Holding Registers.
- c) From Modbus Offset Address should be 11500 which is the location of the Status Block in the ViewMarq display.
- d) Number of Modbus Registers should be 64 (128 bytes) which is the size of the Status Block area.
- e) To Do-More Memory Address should be the starting location of 64 consecutive, available V-memory locations.
- 3) The final step to make the Status reply more readable is to convert the integer block location of the MRX into a String using the STRPUTB instruction.

✓X <sup>∞</sup> ?	0
STRPUTB	Put Bytes Into a String
Put Into String	SL1 °
Start at Index	0
Length in Bytes	128 °
From Starting Element	V200 •

After this instruction has been enabled, the Status Reply should be readable in text form in the String location that was entered.

3) Once the string in the Status String has been read and converted, check the value of the string for the text "OK" using the String Compare instruction.

✓X № ?	0
STRCMP	String Compare
First String	SL1 °
Case Sensitive	
Yes, upper/lower	case are different
O No, upper/lower c	ase are equal
Set If Equal	C10 •
🔲 Set If Less Than	C11
🗌 Set If Greater Than	C11
Second String or Text	
"OK"	•
	~

- a) Select the Source String where the Status String was stored.
- b) Select for Case sensitive.
- c) Select "Set if Equal".
- d) Enter an available discrete tag for the flag.

#### Example Do-more PLC code for sending an ASCII string over serial Modbus



Logic continued next page.



7-62 VIEW MERSAGE DISPLAY User Manual, 2nd Ed. Rev. A-MD-USEM-M

# **Modbus TCP**

The same exact steps used above for sending Modbus Serial Strings should be used for sending Strings over Modbus TCP with only one difference in configuration of the MWX and MRX instructions.

✓X <sup>∞</sup> ?	
MWX	Modbus Network Write
Device	@IntModTCPClient
Modbus/TCP addressing	
IP Address	192 . 168 . 1 . 109
TCP Port Number	502 °
Unit ID	255 °
Function Code	16 - Write Multiple Registers 💌
Modbus Addre	ss 4 + offset value entered below
To Modbus Offset Address	11000 °
Number of Modbus Registers	V1000 °
From Do-more Memory Address o	r Constant V0
Enable	
Once on Leading Edge     Once on Leading Edge	
Continuous on Power Flow	at Interval
Constant 00 h	00 m 00 s 000 ms
C Variable D0	ms
On Success: <ul> <li>Set bit</li> <li>JMP t</li> </ul>	o Stage C3 •
On Error: <ul> <li>Set bit</li> <li>JMP t</li> </ul>	o Stage C4 •
Exception Response	D0

Choose the "@IntModTCPClient" Device and enter in the IP address of the ViewMarq display. Leave the TCP Port Number the same (default is 502) and the Unit ID as 255. Everything else should be the same as the configuration for Modbus Serial.



Example Do-more PLC code for sending an ASCII string over Ethernet Modbus

Logic continued next page.

7



# **DirectLogic PLC by AutomationDirect**

The Direct Logic PLC may communicate with the ViewMarq LED display by ASCII, Modbus RTU or Modbus TCP.

This section discusses: Sending a Command String by ASCII over Serial Sending a Command String by Modbus over Serial Sending a Command String by Modbus TCP over Ethernet

#### Sending Strings from DirectLogic to the ViewMarq display by ASCII over Serial

- 1) There are two methods for sending ASCII strings to the ViewMarq over serial.
  - a. PRINT
  - b. PRINTV
- First the PLC port must be configured for "Non-Seq(ASCII)" in order to allow ASCII strings to be sent.

a. Go to PLC>Setup>Setup Secondary Comm Port:

b. Set the Protocol to Non-Seq(ASCII)

Setup Communication Ports			×
Port: Port : Protor K. Di	2	Base Timeout: 800 ms 800 ms 500 ms	Close
Time-out: Base RTS on delay time: 0 ms	on-Seq[ASUI] emote I/O Timeout	Characters (U	FF flow control
RTS off delay time: 0 ms Data bits: 8 Baud rate: 3840		Echo Supp	ression
Stop bits: 1 Parity: Odd Memory Address: V110		© RS-422 RS-232 © RS-485	2/485 (4-wire) 2C (2-wire) 5 (2-wire)
Port 2: 15 Pin			

c. Match the Data bits, Baud rate, Stop bits and Parity to the serial port settings of the ViewMarq display serial port. The Memory Address V-memory location should be an available, unused block of registers

# Using the DirectLogic PRINT instruction

1) In the ViewMarq software, type and configure a message. Copy the string in the Command String window by selecting Copy on the Command String Toolbar.



2) In the DirectSoft Programming Software paste (Ctrl-V) the ASCII String into the PRINT instruction as shown below.

<b>く</b> X 刻		C
PRINT	Port: K2	•
Message:		
"~EID 0> <clr><win 0="" 287<br="">0&gt;<lj><bl n=""><cs 0=""><red World\$0d"</red </cs></bl></lj></win></clr>	31> <pos 0<br="">&gt;<t>Hello</t></pos>	• •
		-



**NOTE:** Quotation marks must be placed around the message that has been pasted into the PRINT instruction. You will also need to add termination character ("\$0d") to the end of the message string.

#### Example of DirectLogic PRINT instruction to send ASCII



**NOTE:** Use a Store Positive Differential (STRPD) instruction when enabling a PRINTV instruction in order to send only 1 String of data out the port.

#### Print the ASCII String directly to Port 2 of the PLC

C1	PRINT	K2
	" <id 0=""><clr>&lt; 0&gt;<lj><bl n="">&lt; World\$0d"</bl></lj></clr></id>	WIN 0 0 287 31> <pos 0<br=""><cs 0=""><red><t>Hello</t></red></cs></pos>

The "ASCII Reply" option should be disabled when sending ASCII strings with the D2-260 or D0-06. Those PLCs only support one-way ASCII communications. See Chapter 6 - Configuring the ViewMarq LED Display for further details. If the application requires more reliable error detection and handshaking consider using Modbus communications instead.

# Using PRINTV instruction (D2-260 or D0-06 only)

You may also choose to use the PRINTV instruction to send the ASCII string to the ViewMarq. In order to use the PRINTV instruction, the ASCII string must first be embedded into the V-memory of the PLC.

# Embedding the String into PLC memory

1) In the ViewMarq software, type and configure a message. Copy the string in the Command String window by selecting Copy on the Command String Toolbar.

Ĭ	Command String	77 / 500	
	<id 0=""><clr><win 0="" 287="" 31=""><pos 0=""><l< td=""><td>J&gt;<bl n=""><cs 0=""><grn><t>Hello World</t></grn></cs></bl></td><td></td></l<></pos></win></clr></id>	J> <bl n=""><cs 0=""><grn><t>Hello World</t></grn></cs></bl>	
		CLR NUM	

2) In the DirectSoft Programming Software Paste (Ctrl-V) the VPRINT Instruction as shown.

N N N N N N N N N N N N N N N N N N N	0
VPRINT	
Byte Swap :	
• None	
Print to starting V-memory address : V2000	•
* Starting V-memory address = Byte count * Starting V-memory address + 1 = Start of data	
Message:	
" <id 1=""><clr><win 0="" 287="" 31=""><pos 0<="" td=""><td>• •</td></pos></win></clr></id>	• •
World""\$0d"	
	-



Attention!: Quotation marks must be placed around the message that has been pasted into the VPRINT instruction. You will also need to add a termination character ("\$0d") to the end of the message string.

NOTE: For reference later, the V-memory location entered will be where the number of the characters entered into the Message field will be located. The actual ASCII data will start at the 2nd location of the V-memory block. For the "Print to starting V-memory address:" use any available unused block of V-memory addresses.

✓X <sup>×</sup>	•
PRINTV CPU/DCM : C CPU C DCM	Slot Number : K0 Port Number : K2
Start Address :	V2001 *
Number of Bytes :	V2000 *
Append characte	r(s) :
<ul> <li>None</li> <li>1 Character</li> <li>2 Characters</li> </ul>	
Character Code 1	: 00 hexadecimal
Character Code 2	: 00 hexadecimal
Byte Swap : None C All C All but Null	Busy : C20 • Complete : C21 •

3) Use the PRINTV instruction as shown below to choose the block of ASCII characters to send out the serial port.

- a) Port Number should be set to K2 to indicate the HD-15 pin port (Port 2).
- b) Start Address should be the second V-memory location of the V-memory block entered in the VPRINT instruction from earlier.
- c) Number of Bytes should be the first V-memory location specified in the V-memory block entered in the VPRINT instruction earlier.



**NOTE:** Use a Store Positive Differential (STRPD) instruction when enabling a PRINTV instruction in order to send only 1 String of data out the port.

[]	PRINTV CPU/DCM Slot : Port Number : Start Address : Number of Bytes : Append : Byte Swap : Busy : Complete :	CPU K2 V2001 V2000 None None C20 C21
----	--	---



Attention!: Command Strings should be sent at least 100ms apart.

The "ASCII Reply" option should be disabled when sending ASCII strings with the D2-260 or D0-06. Those PLCs only support one-way ASCII communications. See Chapter 6 - Configuring the ViewMarq LED Display for further details. If the application requires more reliable error detection and handshaking consider using Modbus communications instead.

VIEW User Manual, 2nd Ed. Rev. A-MD-USEM-M

7-69

# Sending Strings from DirectLogic to the ViewMarq display by Modbus over Serial

1) The serial port must be configured for Modbus in order to send Strings encapsulated within a Modbus RTU message to the display.

Port	Port 2	Base Timeout: 800 ms 800 ms 500 ms 3 Characters	Close
Time-out:	Base Timeout × 1	-	
RTS on delay time:	0 ms 💌		
RTS off delay time:	0 ms 💌		
Station Number:	1		
Baud rate:	38400 💌	Echo Supp	
Stop bits:	1 💌	• RS-422 RS-232	27485 (4-wire) 2C (2-wire)
Parity:	Odd 💌	C RS-48	5 (2-wire)
Port 2: 15 Pin			

a) Go to PLC>Setup>Setup Secondary Comm Port:

b) Match the Data bits, Baud rate, Stop bits and Parity to the serial port settings of the ViewMarq display serial port.

#### Calculate the Word Count

The ASCII string data was placed in V-memory as Byte Data earlier using the VPRINT instruction. In order to send it over Modbus the Word Count is needed.

1) Divide the byte count by 2 (2 bytes per word) and then add 1 in case the String length works out to an odd number of bytes so that we don't chop off 1 character of the String.

✓X 湾	0	
Math - BCD		
MATHBCD	IB-521	
Result (WORD) V3000	•	
Expression		
(V2000 / K2) + K1	• *	
	$\nabla$	

Continued on Next Page.

2) Now the data is prepared to be transported over Modbus and we have the calculated request size. Use a MWX instruction to write the data out the serial port.

<u>/ X X</u>		0
MWX CPU/DCM :	Slot Number : Port Number :	K0 K2 •
Slave Address : Function	· Preset Multiple	K1 •
Start Slave Memory Address :		K411000 •
Start Master Memory Address : Number of Elements : Modbus Data Format		V3000 •
© 584/984 r C 484 mode	node	
Exception Respo	nse Buffer :	V400 •

- a) The Port Number should be K2 to specify the secondary comm. port (Port 2) of the D2-260 or D0-06.
- b) The Slave Address should match the Node Address setting of the serial port configuration in the LED Display Configuration.
- c) The Function Code should be set to 16 Preset Multiple Registers.
- d) Start Slave Memory Address is the location of the Command Block within the ViewMarq display.



**NOTE:** The DirectLogic MWX instruction uses the Modicon style addressing where the highest digit (4 in this case) is the Modbus data type indicator (Holding Registers).

- e) Start Master Memory Address should be the resulting memory location of the VPRINT instruction + 1 that contains the String data converted to Integer.
- f) Number of Elements should contain the V-memory location that was the result of our calculated Modbus request size from above.
- g) Modbus Data Format should be 584/984 mode to match the example addressing shown above.
- h) Exception Response Buffer should be an available, unused V-memory address. Note that this field uses 3 consecutive V-memory addresses.



#### Example DirectLogic PLC code for sending a string to ViewMarq over serial Modbus

7-72 VIEW User Manual, 2nd Ed. Rev. A-MD-USEM-M
## **Optional Error Checking**

 After the Modbus Write is successful, the ViewMarq display will process the message. When the display has finished processing the Message it will clear the Command Block. Use the MRX instruction to read the first element of Command Block until it reads 0.

√X̤		C
MRX		
CPU/DCM :	Slot Number :	K0
O DCM	Port Number :	K2 •
Slave Address :	1	К1 •
Function Code : 03	- Read Holding I	Registers 💌
Start Slave Mem	ory Address :	K411000 •
Start Master Mer	nory Address :	V4000 •
Number of Eleme	ents :	К1 •
Modbus Data F	Format -	
<ul> <li>584/984</li> <li>484 mode</li> </ul>	mode :	
Exception Respo	onse Buffer :	V405 •

- a) The Port Number should be K2 to specify the secondary comm. port (Port 2) of the D2-260 or the D0-06.
- b) The Slave Address should match the Node Address setting of the serial port configuration in the LED Display Configuration.
- c) The Function Code should be set to 03 Read Holding Registers.
- d) Start Slave Memory Address is the location of the Command Block within the ViewMarq display.



**NOTE:** The DirectLogic MWX instruction uses the Modicon style addressing where the highest digit (4 in this case) is the Modbus data type indicator (Holding Registers).

- e) Start Master Memory Address should be an available, unused V-memory location that can be used to compare to 0. The PLC should keep reading until this V-memory location indicates a 0.
- f) Number of Elements should be 1 as we are reading only the first register of the Command Block.
- g) Modbus Data Format should be 584/984 mode to match the example addressing shown above.
- h) Exception Response Buffer should be an available, unused V-memory address. Note that this field uses 3 consecutive V-memory addresses.

2) After the first register of the Command Block returns a 0, use the MRX instruction again to read the Status Block.

✓X [20]	
MRX CPU/DCM : Slot Number : CPU C DCM Port Number :	К0 К2
Slave Address : Function	K1 •
Start Slave Memory Address :	K411500 •
Start Master Memory Address : Number of Elements :	K125
Modbus Data Format • 584/984 mode • 484 mode	
Exception Response Buffer :	V410 •

- a) The Port Number should be K2 to specify the secondary comm. Port of the D2-260 or the D0-06.
- b) The Slave Address should match the Node Address setting of the serial port configuration in the LED Display Configuration.
- c) The Function Code should be set to 03 Read Holding Registers.
- d) Start Slave Memory Address is the location of the Status Block within the ViewMarq display.



**NOTE:** The DirectLogic MWX instruction uses the Modicon style addressing where the highest digit (4 in this case) is the Modbus data type indicator (Holding Registers).

- e) Start Master Memory Address should be an available, unused block of V-memory locations.
- f) Number of Elements should be 125.
- g) Modbus Data Format should be 584/984 mode to match the example addressing shown above.
- h) Exception Response Buffer should be an available, unused V-memory address. Note that this field uses 3 consecutive V-memory addresses.

To view the Status string in clear text, enter in the starting V-memory location of the last MRX instruction and change the view to "Text" and the size to 40 in Data View.



7-74

**NOTE:** If the response was an error and the error text is longer than 40 characters, you will have to enter in a V-memory location further into the block and change the view for that location to text as well.

VIEW MARGINE User Manual, 2nd Ed. Rev. A–MD-USEM-M



### Example DirectLogic PLC code for reading the Status Block in the ViewMarq

# Sending Strings from DirectLogic to the ViewMarq display by Modbus over Ethernet

Sending messages from the PLC over Modbus TCP requires that an H2-ECOM100 module is used with the D2-260 and a H0-ECOM100 module with the D0-06.

The same steps used above for sending Modbus Serial Strings should be used for sending Strings over Modbus TCP but the Modbus instructions will be quite different.

## Configuring the ECOM100 Module

Before the instructions are entered, the ECOM100 module must first be configured for a Modbus TCP message. The NetEdit tool is required for this and can be downloaded for free from www.automationdirect.com.

1) Start the NetEdit software after it has been downloaded and installed on the PC. It will automatically scan the network attached to the PC and bring up the devices in a window that looks like the window below.

🛓 NetEdit 3										<b></b>		x
File Network View	N H	elp										
IPX ТСРИР S	ican twork		0									
Ethernet Address	F	3   C	Module T	/ A	IP Add	ress	ID	Name			Description	
00 E0 62 20 43 8F			H2-ECOM	100	(172.3	0.52.91)	0	H2-EC	DM100		H2-ECOM1	00 Eth
•				111								÷.
Module Info ECOM	I Sett	ngs	ECOM Help	Gene	ral Help							
General Info						Ethernet	Stats			Reset S	tats	
Firmware Rev: 4.	0.1776	;	PWB Rev:	2	۵,	Missed	Fr:	132	Bad Pkts:	0		
Booter Rev: 4	.0.165		PLD Rev:	1/	Δ,	TX Coll:		0	Unknown:	0		
Switch Setting: 00	00000	0	CPU Rev:	1.0	0.1	Lost Pk	is:	0	TX Errors:	0		
Ready											NUM	

2) Click on the "ECOM Settings" tab at the bottom and then click on the "Peer to Peer Config..." button.

🔏 NetEdit 3					Ŧ	
File Network View	Help					
IPX TCP/IP Sc Netv	an vork	2				
Ethernet Address	FBC	Module Ty 🛆	IP Address	ID	Name	Description
00 E0 62 20 43 8F		H2-ECOM100	(172.30.52.91)	0	H2-ECOM100	H2-ECOM100 Eth
•						+
Module Info ECOM	Settings	ECOM Help   Gene	ral Help			
Configuration	-		Utils		Firmware	
General		EMail Config	Test CF	PU Access!	Update Firmware.	
Advanced			Test	EMail	Update Booter	
Peer to Peer Config	g				Restore Factory Settin	ngs
Ready						NUM

3) In the Peer to Peer Configuration window, click on the "Add" button to the right.

Device	Protocol	Configuration	Add
			E dit
			Delete
			Delete Al
			OK
			Cancel

4) Fill in the Add Device Address window as shown below but using the IP address of your ViewMarq display.

1	1			-
Π	-	_	_	-
/	-	_	_	-
	-			- 1

NOTE: The RX/WX Device Number will be needed in your Modbus TCP instruction setup.

Add Device Address	X
RX/WX Device Number:	1
Device Configuration	
О ЕСОМ	Find Hx-ECOM
IP Address:	
Ethernet Address:	「
Modbus/TCP	
IP Address:	172.30.52.53
Port:	502
Unit ID:	255
OK	Cancel

5) Click the Ok button to send this configuration to the ECOM100 module.

7-78 VIEW User Manual, 2nd Ed. Rev. A-MD-USEM-M

## Calculating the Byte Count

To send a Modbus TCP message, you must use the RX and WX instructions. It is not just one instruction but a series of 4 instructions that work together. Here we focus primarily on how these instructions are used with the ViewMarq display. To find more details on the usage of the RX and WX instruction when doing Modbus TCP, consult the ECOM100 manual.

To send a Modbus TCP write message, we must first adjust the size we are writing. The WX instruction uses a byte count for sending data but it must be an even count.

1) The first thing to do is to divide the byte count (V2000 in this case) by 2 and check to see if there is a remainder. The DIV instruction will place the remainder (if there is one) into the second stack location. In order to retrieve that value, we must use a POP instruction and then an OUT to move the value to a V-memory location.



2) Now that we know where the remainder is, we can do a simple comparison to see if there is a remainder or not. If there is no remainder we simply move the count to the V-memory location that will be our Modbus TCP write size. If there is a remainder, we simply add 1 to the original count to make it even and place the result in the V-memory location that will be our Modbus TCP write size.



### Sending the String

1) Now that the byte count is adjusted, send the string to the ViewMarq using the WX instruction. The logic for sending the first Modbus TCP write is as follows:



- a) The first LD instruction specifies the slot number that the ECOM100 is located in and the RX/WX Device number that was configured earlier in NetEdit. The upper byte specifies the Slot (2 in the example above). The lower byte specifies the RX/ WX Device number (01 in the example above).
- b) The second LD instruction specifies the size, in bytes, that will be sent. V3000 is the result of the calculation explained above.
- c) The LDA instruction specifies where the data will be written from. In this case, V2001 is where the String data resides from the VPRINT instruction configured earlier.
- d) The WX instruction specifies the location of the Command Block within the ViewMarq display.

When doing Modbus TCP, the V-memory location specified is an alias of the Modbus address and has to be converted. V-memory locations in the DirectLogic PLCs are in octal format. To convert the Modbus address to the alias address in DirectLogic, you take the

desired Modbus address in 6 digit format (example 411000), subtract 400001 and then convert the value to Octal. So 411000 – 400001 = 10999 Convert to Octal = 25367. (See http://support.automationdirect.com/technotes.html for the Modbus Conversion Utility AN-MISC-010).

2) After the Modbus Write is successful, the ViewMarq display will process the message. When the display has finished processing the Message it will clear the Command Block. Use the RX instruction to read the first element of Command Block until it reads 0.



- a) The first LD instruction specifies the slot number that the ECOM100 is located in and the RX/WX Device number that was configured earlier in NetEdit. The upper byte specifies the Slot (2 in the example above). The lower byte specifies the RX/WX Device number.
- b) The second LD instruction specifies the size, in bytes, that will be read. K2 specifies 1 Modbus element, which is the first register of the Command Block.
- c) The LDA instruction specifies where the data will be written into. In this case, we chose an available V-memory location that we can use to check for 0 data.
- e) The instruction specifies the location of the Command Block within the ViewMarq display.

When doing Modbus TCP, the V-memory location specified is an alias of the Modbus address and has to be converted. V-memory locations in the DirectLogic PLCs are in octal format. To convert the Modbus address to the alias address in DirectLogic, you take the desired Modbus address in 6 digit format (example 411000), subtract 400001 and then convert the value to Octal. So 411000 – 400001 = 10999 Convert to Octal = 25367.



3) After the first register of the Command Block returns a 0, use the RX instruction again read the Status Block.



- a) The first LD instruction specifies the slot number that the ECOM100 is located in and the RX/WX Device number that was configured earlier in NetEdit. The upper byte specifies the Slot (2 in the example above). The lower byte specifies the RX/WX Device number.
- b) The second LD instruction specifies the size, in bytes, that will be read. K128 specifies 64 Modbus elements.
- c) The LDA instruction specifies where the data will be written into. In this case, we chose an available V-memory block that we can place the Status data into.
- d) The RX instruction specifies the location of the Status Block within the ViewMarq display.

When doing Modbus TCP, the V-memory location specified is an alias of the Modbus address and has to be converted. V-memory locations in the DirectLogic PLCs are in octal format. To convert the Modbus address to the alias address in DirectLogic, you take the desired Modbus address in 6 digit format (example 411500 which is where the Status data is located), subtract 400001 and then convert the value to Octal. So 411500 – 400001 = 11499 Convert to Octal = 26353.

## **PLC Cabling Chart**

ViewMarq LED message displays can communicate with any master serial ASCII device, serial Modbus device or Modbus TCP controller. This chart below is a reference for communicating from AutomationDirect PLCs to the ViewMarq LED message displays.

	AutomationDirect PLC to ViewMarq Cabling							
Oshla	Maturali	Autom	ationDirect Cor	ntroller	Marin Davit			
Cable	Network	Family	Model	Port	viewiwiard Port			
		Productivity3000	P3-550		ĺ			
<u> </u>		,	H2-DM1E	Ethernet (Modbus TCP)	RJ45			
Cat 5e	Ethernet	Do-more	H2-ECOM-100					
		DirectLogic	H2-ECOM-100					
				1				
			C0-00					
		CLICK	C0-01					
			C0-02					
		Draductivity (2000	P3-530					
		Productivity5000	P3-550					
		Domoro	H2-DM1					
		Do-more	H2-DM1E					
D0-CBL	RS-232		DL05	RJ12	Port 1			
			DL06	Senal Port	NJ 12			
		DirectLogic	DL105					
			DL205					
			H2-SERIO					
			H2-SERIO-4					
			D3-350					
			D4-450					
				1				
			D0-DCM	D-SUB 15HD Serial Port				
DO-CBL	DC 333	Directl ogic	DL06		Port 1			
+ FA-HD15	NJ-232	DirectLogic	D2-250-1		RJ12			
			D2-260					
				1	1			
			D0-DCM	-				
D2-DSCBL-2	RS-485	DirectLogic	DL06	D-SUB 15HD	Port 1			
			D2-250-1	Serial Port	I erminai			
			D2-260					
					Dort 1			
+ FA-CABKIT	RS-232	DirectLogic	DL-405	Serial Port	RJ12			
				Contain one	1.0.2			
		CLICK	C0-02					
		Draductivity 2000	P3-530	]				
		Productivity3000	P3-550					
Cable L19773	RS-485	Demen	H2-SERIO	Terminal Block	Port 2			
		Do-more	H2-SERIO-4					
		Divert	H2-SERIO	1				
		DirectLogic	H2-SERIO-4	1				

## Allen Bradley MicroLogix and SLC PLCs

Depending on the PLC model\*, the Allen Bradley MicroLogix and SLC series PLCs can utilize RS-232 with either ASCII or Modbus RTU to send ASCII strings to the ViewMarq LED Display.

## **RS-232**

Connect the ViewMarq Port 1 terminal block to the Allen Bradley 8-pin Mini Din port or the 9-pin D-Sub port. The cables for this will need to be constructed by the user.



and / or Modbus instructions.

7-84

VIEW User Manual, 2nd Ed. Rev. A-MD-USEM-M

## ASCII

In the LED Display Configuration window configure the ViewMarq RS-232 port to communicate using ASCII.

LED Display Configuration	×
LED Display Selection	Configuration File
Display Name: ViewMarq	Import from File
Link Name: Ethernet	Export to File
Change Link Blink	Global Settings
Read Configuration from ViewMarq	Name: ViewMarg
	LED Display ID: 1
Ethernet RS-232 RS-485	Heartbeat
Basic Configuration	Timeout 0 seconds
Protocol: ASCII -	
Node-Address (1-247):	ASCII

Configure the Allen Bradley PLC port to communicate ASCII with the ViewMarq LED Display.

1) In RSLogix500, open the Channel Configuration for the PLC...

Channel Configuration	×
General Channel 0 Channel 1 Channel 2	
Driver ASCII	
Baud 38.4K -	
Parity ODD -	
Stop Bits 1 💌	
Data Bits 8	
Termination Characters	
Termination 1 Vd Termination 2 Vff	
Protocol Control	
Control Line No Handshaking	
Delete Mode Ignore	
I Echo	
X0N/X0FF	
	_

- 2) For the Channel being used on the PLC (Channel 0 shown above)...
  - a. Set the Driver to ASCII.
  - b. Match the Baud Rate, Parity, Stop Bits and Data Bits to match those of Port 1 of the ViewMarq LED Display.
  - c. If the ASCII Reply from the ViewMarq will be read, set the Termination Characters as shown.

i. \d is a Carriage Return

ii. \ff means not used

3) On the General Tab, for the Channel configured in Step 2...

a. Set the AWA Append Characters as shown below.

i. \d is a Carriage Return

ii. \ff means not used

anner configuration	
ieneral Channel 0 Channel 1 Channel 2	
Channel 0 Driver: ASCII	Comms Servicing Selection
Memory Module Over-Write Protected	Message Servicing Selection
Edit Resource/Owner Timeout (x 1sec) 60	1st AWA Append Character \d 2nd AWA Append Character \ff

In order for the PLC to send the String it must be in the PLC Memory as a String Data Type.

1) Copy the ASCII String from the ViewMarq Software.

	77 / 50	0 🗵
<pre><id 1=""><clr>WIN 0 0 287 31&gt;<pos 0=""><lj><bl n=""><cs 0=""><grn><t>Hello World</t></grn></cs></bl></lj></pos></clr></id></pre>		
CLR N	NUM	

2) In RSLogix500, open the String data file you are using and paste the ASCII String into the file.

🖉 Data F	ile ST9			
Offset	LEN String	Text (Symbol) I	Description	
ST9:0	75 <id 1=""></id>	<clr><win 0="" 287<="" td=""><td>7 31&gt;<pos 0=""><lj><bl< td=""><td>N&gt;<cs 0=""><amb><t>Hello World</t> 🔺</amb></cs></td></bl<></lj></pos></td></win></clr>	7 31> <pos 0=""><lj><bl< td=""><td>N&gt;<cs 0=""><amb><t>Hello World</t> 🔺</amb></cs></td></bl<></lj></pos>	N> <cs 0=""><amb><t>Hello World</t> 🔺</amb></cs>
ST9:1	0			
ST9:2	0			×
•				• • • • • • • • • • • • • • • • • • •
ST	19:4			Radix.
Symbol:				Columns: 2 💌
Desc:				
ST9		Properties	<u>U</u> sage	Help

Now that the Port has been configured and the ASCII String is in PLC memory, use an AWA (ASCII Write Append) instruction to send the string out the serial port.

AWA	
ASCII Write Apper	ıd
Channel	0
Source	ST9:0
Control	R6:0
String Length	82<
Characters Sent	76
Error	00h

- 1) Select the Channel number that was configured earlier and that is connected to the ViewMarq.
- 2) For the Source select the String Data Address the ASCII String was Pasted into.
- 3) Choose an unused Control Address.
- 4) Enter 82 for the String Length Maximum. For strings longer than 82 characters, see the next section, "Long ASCII Strings".



**NOTE:** The AWA instruction is a "Level-enabled" instruction so be sure to put a One Shot (ONS) instruction for the enable.

Send a string less than 83 characters long

Use a ONS (One Shot) to enable the AWA (ASCII Write Append) i The termination characters (/d - carriage return) defined in the Cha	nstruction to send String ST9:0 out Channel 0. annel Configuration will be appended to the string.	
Взл Взл ] ] [[ОМS]] 0 0	ASCII Write Append Channel 0 Source ST9:0 Control R6:0 String Length 82< Characters Sent 76 Error 00h	SN)

## **Reading the ASCII Reply**

If the "ASCII Reply" option is enabled in the "LED Display Configuration" settings AND you are sending to an ID other than 0, your PLC may Read the Reply Response String that is sent by the ViewMarq LED Display.

Use an ABL (ASCII Test For Line) instruction to check for the Termination characters entered in the Channel configuration and then a ARL (ASCII Read Line) instruction to read in the Status reply response String.

- 1) For the ABL instruction, select the Channel number that was configured earlier and that is connected to the ViewMarq.
- 2) Choose an unused Control Address.
- 3) Trigger the ABL instruction with a normally closed contact using the Control Address Enable to trigger the instruction each PLC scan.
- 4) For the ARL instruction, select the Channel number that was configured earlier and that is connected to the ViewMarq.
- 5) Choose an unused Control Address.
- 6) Trigger the ARL instruction with the Found (FD) bit of the ABL instruction Control Address.

#### Read the ASCII Reply

Enable an ABL(Test Buffer for Line) instruction every scan to check the input buffer.



If any characters are found (FD), read the input buffer with an ARL (ASCII Read Line) instruction.

R6:2 FD	ARL ASCII Read Line Channel 0 Dest ST9:1 Control R6:3 String Length 82< Characters Read 29 Error 00h	-(EN)- -(DN)- -(ER)-
	LHor OOR	

If the syntax sent to the ViewMarq display was correct, you will see an "OK" in your Response String. If there is a problem, you will get a Syntax error and the position where the error occurred.

🖉 Data F	ile ST9								
Offset	LEN S	String Text	: (Symbol)	Description	1				
ST9:0	75 <	ID 1> <clr:< td=""><td><win 0="" 28<="" td=""><td>7 31&gt;<pos 0<="" td=""><td>0&gt;<lj><bl< td=""><td>N&gt;<cs< td=""><td>0&gt;<amb><t>Hello</t></amb></td><td>World</td><td>^</td></cs<></td></bl<></lj></td></pos></td></win></td></clr:<>	<win 0="" 28<="" td=""><td>7 31&gt;<pos 0<="" td=""><td>0&gt;<lj><bl< td=""><td>N&gt;<cs< td=""><td>0&gt;<amb><t>Hello</t></amb></td><td>World</td><td>^</td></cs<></td></bl<></lj></td></pos></td></win>	7 31> <pos 0<="" td=""><td>0&gt;<lj><bl< td=""><td>N&gt;<cs< td=""><td>0&gt;<amb><t>Hello</t></amb></td><td>World</td><td>^</td></cs<></td></bl<></lj></td></pos>	0> <lj><bl< td=""><td>N&gt;<cs< td=""><td>0&gt;<amb><t>Hello</t></amb></td><td>World</td><td>^</td></cs<></td></bl<></lj>	N> <cs< td=""><td>0&gt;<amb><t>Hello</t></amb></td><td>World</td><td>^</td></cs<>	0> <amb><t>Hello</t></amb>	World	^
ST9:1	4 0	)K^M^J							-
ST9:2	0								~
•									· 🖃
S	T9:6						Radix:		-
Symbol:								Columns: 2	-
Desc:									
ST9 ·		1	Properties		<u>U</u> sage		Help		

#### **Example of a good String**

🖀 Data	File ST	9										
Offset	LEN	String	Text	(Symbol	) Descrip	otion						
ST9:0	74	<id 1="">&lt;</id>	CLR> <w< td=""><td>IN O O</td><td>287 31&gt;&lt;1</td><td>POS 0</td><td>0&gt;<lj><bl< td=""><td>N&gt;<cs< td=""><td>0&gt;<ame< td=""><td>&gt;<t>Hello</t></td><td>World<t></t></td><td>^</td></ame<></td></cs<></td></bl<></lj></td></w<>	IN O O	287 31><1	POS 0	0> <lj><bl< td=""><td>N&gt;<cs< td=""><td>0&gt;<ame< td=""><td>&gt;<t>Hello</t></td><td>World<t></t></td><td>^</td></ame<></td></cs<></td></bl<></lj>	N> <cs< td=""><td>0&gt;<ame< td=""><td>&gt;<t>Hello</t></td><td>World<t></t></td><td>^</td></ame<></td></cs<>	0> <ame< td=""><td>&gt;<t>Hello</t></td><td>World<t></t></td><td>^</td></ame<>	> <t>Hello</t>	World <t></t>	^
ST9:1	24	Syntax	error	at pos	54^M^J							-
ST9:2	0											~
												• 🗖
[	ST9:0									Radix:		Ŧ
Symbol:											Columns: 2	-
Desc:											,	
ST9	- -		Prop	perties			<u>U</u> sage			<u>H</u> elp		

Example of a syntax error String

## Sending Long ASCII Strings

If the ASCII String is longer than 82 characters, multiple String Files and multiple instructions need to be used.

1) Copy the ASCII String from the ViewMarq software.



- 2) Paste up to 82 characters of the String into the first String File.
- Paste the up to 82 of the next characters into the next available String File and so on until the entire string is contained in memory.



**NOTE:** It does not matter where the string is broken apart. It will be assembled again in the ViewMarq display when a Carriage Return character is received.

🗃 Data F	ile ST	9	
Offset	LEN	String Text (Symbol) Description	
ST9:0	75	<pre><id 1=""><clr><win 0="" 287="" 31=""><pos 0=""><lj><bl n=""><cs 0=""><grn><t>Hello World</t> _</grn></cs></bl></lj></pos></win></clr></id></pre>	
ST9:1	28	<pos 0="" 8=""><lj><t>ViewMarq</t></lj></pos>	
ST9:2	0		
ST9:3	0		
ST9:4	0		
ST9:5	0		
ST9:6	0		
ST9:7	0		
ST9:8	0		
ST9:9	0		
			• <b>-</b>
5	/9:0	Radix:	
Symbol:		Gor	umns: 2 🖉 💌
Desc:			
ST9		Properties Usage Help	

The AWA instruction will always "append" the termination character to the end of the string. To get around this, use the AWT (ASCII Write) Instruction as shown below. The AWT instruction sends the ASCII String out the Serial port WITHOUT the termination character.

Use as many instructions necessary to send each String File triggering the instruction with the Done (DN) bit of the previous instruction. The LAST instruction must be an AWA instruction to send the termination character.

Send a string longer than 82 characters

Use a ONS (One Shot) to enable the AWT (ASCII Write) instruction to send String ST9:0 out Channel 0. No termonation characters are appended using this instruction.

B3:0 B3:0	AWT	
— ¬ Г Г (омз ¬	ASCII Write	-CEN )-
	Channel 0	2.1
	Source ST9:0 -	-(DN)-
	Control R6:0	
	String Length 82< -	-(ER)
	Characters Sent 75	
	Error 00h	

When the AWT instruction is "Done" use a ONS (One Shot) to enable the AWA (ASCII Write Append) instruction to send String ST9:1 out Channel 0. The termination characters (/d - carriage return) defined in the Channel Configuration will be appended to the string.

R6:0	B3:0	AWA		1
F	-Fons 7-	ASCII Write Append		(EN)
DN	2	Channel	0	
		Source	ST9:1	-CND)-
		Control	R6:1	
		String Length	82<	-(ER)-
		Characters Sent	28	
		Error	00h	

## Modbus

## Writing the ASCII String to the Command Block

In the LED Display Configuration window configure the ViewMarq RS-232 port.

- 1) Set the Protocol to Modbus.
- 2) Set the LED Display ID to the desired Modbus Node number.
- 3) Select Byte Swap because Allen Bradley Modbus is the opposite Byte order from ViewMarq Modbus.

LED Display Configuration	×
LED Display Selection	Configuration File
Display Name: ViewMarq AT Link Name: Ethernet	Import from File Export to File
Change Link Blink	Global Settings
Bood Configuration from VisueMara	Identification
Read Configuration from ViewMarq	Name: ViewMarg
	LED Display ID: 1
Ethernet RS-232 RS-485 Discrete Inputs	🕼 Display Startup Banner
Basic Configuration	
Protocol: Modbus 🔻	Heartbeat
Node Address (1 - 247): 1	Timeout 0 seconds
Baud Rate (bps): 38400 🔻	Discretes
Parity: Odd 🗸	Cnabled
Stop Bit: 1	
Data Bits: 8 🗸	MODBUS ASCII
	Byte Swap ASCII Reply
Advanced Configuration	Word Swap

Configure the Allen Bradley PLC port to communicate Modbus with the ViewMarq LED display.

1) In RSLogix500, open the Channel Configuration for the PLC.

Channel Configuration	×
General Channel 0 Channel 1 Channel 2	
Driver Modbus RTU Master 👤 (a)	
Baud 38.4K	
Parity ODD -	
Stop Bits 1 💽	
Data Bits 8 💌 🔳	
~	

- 2) For the Channel being used on the PLC (Channel 0 shown above):
  - a. Set the Driver to Modbus.
  - b. Match the Baud Rate, Parity, Stop Bits and Data Bits to match those of Port 1 of the ViewMarq LED display.

In order for the PLC to send the String it must be in the PLC Memory as an Integer Data Type.

3) Copy the ASCII String from the ViewMarq Software.

Command String	Сору	103 / 500	
<id 0=""><clr><win 0<="" th=""><th>0 287 31&gt;<pos 0=""><lj><bl n=""><cs 0=""><grn><t>Hello World</t><pos 0="" 8="">&lt;</pos></grn></cs></bl></lj></pos></th><th>(LJ&gt;<t>ViewMarq</t></th><th></th></win></clr></id>	0 287 31> <pos 0=""><lj><bl n=""><cs 0=""><grn><t>Hello World</t><pos 0="" 8="">&lt;</pos></grn></cs></bl></lj></pos>	(LJ> <t>ViewMarq</t>	
		NUM	]



4) In RSLogix500 open the Integer data file you are using and paste the ASCII String into the file. Open the Integer data file and change the Radix to "ASCII".

🖉 Data Fi	ile N7 (as		INTEGER							_	
Offset	0	1	2	3	4	5	6	7	8	9	
N7:0	\00\00	\00\00	\00\00	\00\00	/00/00	\00\00	\00\00	/00/00	\00\00	\00\00	^
N7:10	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00/	
N7:20	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00/	
N7:30	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00/	
N7:40	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00/	
N7:50	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00/	
N7:60	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00/	
N7:70	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	
N7:80	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00/	
N7:90	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	
N7:100	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00/	~
•											
N7	M0								Rada	ASCIL	
Sumbol								=	naub	Columns	
Deser										Columna	110
				1			1			1	
N/ 🖃		E	roperties			Usage			Help		

5) Double-click on the first Integer file location and Paste the data in and hit Enter. It should look something like the window below.

🖉 Data	File N7 (a	scii)	INTEGER	2							
Offset	; 0	1	2	3	4	5	6	7	8	9	
N7:0	<1	D	1>	<c< td=""><td>LR</td><td><math>\sim</math></td><td>WI</td><td>N</td><td>0</td><td>0</td><td>^</td></c<>	LR	$\sim$	WI	N	0	0	^
N7:10	28	7	31	~	PO	s	0	0>	<l< td=""><td>J&gt;</td><td></td></l<>	J>	
N7:20	<b< td=""><td>L</td><td>N&gt;</td><td><c< td=""><td>s</td><td>0&gt;</td><td><g< td=""><td>RN</td><td>~</td><td>T&gt;</td><td>=</td></g<></td></c<></td></b<>	L	N>	<c< td=""><td>s</td><td>0&gt;</td><td><g< td=""><td>RN</td><td>~</td><td>T&gt;</td><td>=</td></g<></td></c<>	s	0>	<g< td=""><td>RN</td><td>~</td><td>T&gt;</td><td>=</td></g<>	RN	~	T>	=
N7:30	He	11	0	¥o	rl	d<	/T	$\sim$	PO	S	
N7:40	0	8>	<l< td=""><td>J&gt;</td><td><t< td=""><td>&gt;V</td><td>ie</td><td>ωM</td><td>ar</td><td>q&lt;</td><td></td></t<></td></l<>	J>	<t< td=""><td>&gt;V</td><td>ie</td><td>ωM</td><td>ar</td><td>q&lt;</td><td></td></t<>	>V	ie	ωM	ar	q<	
N7:50	/T	>	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	
N7:60	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	
N7:70	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	
N7:80	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	
N7:90	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	~
•										•	Ξ
	N7:109							] F	Radix ASC	21	•
Symbol:									Co	lumns: 10	•
Desc:											
N7	•	Er	operties			<u>U</u> sage			Help		

6) An additional byte (termination) must be added to the end of the data. If the String data is an odd number of bytes, the two termination characters will have to be separated between 2 different Integer addresses. Double-click on the first field and add "\d" after the ">".

🖀 Data F	ile N7 (as	scii)	INTEGER	2						l	
Offset	0	1	2	3	- 4	5	6	7	8	9	
N7:0	<1	D	1>	<c< th=""><th>LR</th><th><math>\times</math></th><th>MI</th><th>N</th><th>0</th><th>0</th><th>^</th></c<>	LR	$\times$	MI	N	0	0	^
N7:10	28	7	31	$\sim$	PO	s	0	0>	<l< th=""><th>J&gt;</th><th></th></l<>	J>	
N7:20	<b< th=""><th>L</th><th>N&gt;</th><th><c< th=""><th>s</th><th>0&gt;</th><th><a< th=""><th>MB</th><th><math>\times</math></th><th>T&gt;</th><th></th></a<></th></c<></th></b<>	L	N>	<c< th=""><th>s</th><th>0&gt;</th><th><a< th=""><th>MB</th><th><math>\times</math></th><th>T&gt;</th><th></th></a<></th></c<>	s	0>	<a< th=""><th>MB</th><th><math>\times</math></th><th>T&gt;</th><th></th></a<>	MB	$\times$	T>	
N7:30	He	11	0	Wo	rl	d<	/1	>\0d	~J0	0	=
N7:40	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	100/00	00/00	\00\00	
N7:50	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	/00/00	\00\00	\00\00	_

If the String data is an even number of bytes, enter in "0d000" in the next Integer address after the String data.



7-94

NOTE: Please note that when you hit enter, the display changes the \0d to ^M.

🖀 Data Fi	ile N7 (as	cii)	INTEGER	2							
Offset	0	1	2	3	4	5	6	7	8	9	
N7:0	<1	D	1>	<c< td=""><td>LR</td><td><math>\sim</math></td><td>WI</td><td>N</td><td>0</td><td>0</td><td>^</td></c<>	LR	$\sim$	WI	N	0	0	^
N7:10	28	7	31	><	PO	ន	0	0>	<l< td=""><td>J&gt;</td><td></td></l<>	J>	
N7:20	<b< td=""><td>L</td><td>N&gt;</td><td><c< td=""><td>S</td><td>0&gt;</td><td><g< td=""><td>RN</td><td>&gt;&lt;</td><td>T&gt;</td><td></td></g<></td></c<></td></b<>	L	N>	<c< td=""><td>S</td><td>0&gt;</td><td><g< td=""><td>RN</td><td>&gt;&lt;</td><td>T&gt;</td><td></td></g<></td></c<>	S	0>	<g< td=""><td>RN</td><td>&gt;&lt;</td><td>T&gt;</td><td></td></g<>	RN	><	T>	
N7:30	He	11	o	Wo	rl	d<	/T	><	PO	s	
N7:40	0	8>	<l< td=""><td>J&gt;</td><td><t< td=""><td>&gt;V</td><td>ie</td><td>ωM</td><td>ar</td><td>q&lt;</td><td></td></t<></td></l<>	J>	<t< td=""><td>&gt;V</td><td>ie</td><td>ωM</td><td>ar</td><td>q&lt;</td><td></td></t<>	>V	ie	ωM	ar	q<	
N7:50	/Т	>^M	100100	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	
N7:60	\00\00	\00\00	100/00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	
N7:70	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	

Now that the data is prepared, use a MSG (Read/Write Message) instruction to send the data out the port to the ViewMarq Display.

MSG		
Read/Write	Message	
MSG File	MG10:0	
Setup Se	creen <	

1) Choose an available, unused MG address.

2) Click on the "Setup Screen" option to configure the instruction.

🔀 MSG - MG10:0 : (1 Elements)	
General         This Controller         Channet:       0 [Integral]         Modbus Command:       16 Write Multiple Registers (4xxxx)         Data Table Address:       N7:0         Size in Elements:       120         Target Device       Message Timeout :         MB Data Address (1-65536):       11000         Slave Node Address:       411000	Control Bits Ignore if timed out (TO): ① Awaiting Execution (EW): ① Error (ER): ① Message done (DN): ① Message Transmitting (ST): ① Message Enabled (EN): ① Error

- 3) Select the Channel number that was configured earlier and that is connected to the ViewMarq.
- 4) Select Modbus Command: 16 Write Multiple Registers (4xxxx).
- 5) In the Data Table Address field, enter the starting address of the Integer data file that the copy and pasted data was placed into.
- 6) The Size in Elements field should be sufficient in size to include all of the populated Integer addresses. For Modbus, this can be up to 120 registers in length. For strings longer than 240 characters, see the section, "Sending Long ASCII Strings over Modbus".
- 7) Choose "16 Bit" for the Data field.
- 8) Leave the Message Timeout at 2. This may need to be increased if the cable quality is suspect or if the environmental conditions are less than ideal.
- 9) The MB Data Address should be 11000 for the Command Block. This equates to 411000 in Modicon Address style.
- 10) The Slave Node Address should match the LED Display ID (Node Address) configured in the LED Display for the port you are connecting to.

Write the ASCII String stored starting at N7:0 to the ViewMarq ASCII Command Block (Modbus address 411000)



### **Reading the Status Reply Buffer**

After the Modbus Write is successful, the ViewMarq display will process the message. When the display has finished processing the message it will clear the Command Block. The next step of the logic should be to read the 1st element of Command Block until it is 0.

Use another MSG (Read/Write Message) instruction to read the Command Block in the ViewMarq Display.



1) Choose an available, unused MG address.

2) Click on the "Setup Screen" option to configure the instruction.

- 3) Select the Channel number that was configured earlier and that is connected to the ViewMarq.
- 4) Select Modbus Command: 03 Read Multiple Registers (4xxxxx).
- 5) In the Data Table Address field, enter an available, unused Integer address that can be compared to 0.
- 6) The Size in Elements field should be 1 to read the first word of the Command Block.
- 7) Choose "16 Bit" for the Data field.
- 8) Leave the Message Timeout at 2. This may need to be increased if the cable quality is suspect or if the environmental conditions are less than ideal.
- 9) The MB Data Address should be 11000 for the Command Block. This equates to 411000 in Modicon Address style.
- 10) The Slave Node Address should match the LED Display ID (Node Address) configured in the LED Display for the port you are connecting to.

11) Keep executing the Read MSG of the Command Block until the 1st element is 0. Once the value equals 0, the Status can be read.



Use another MSG (Read/Write Message) instruction to read the Status Reply Buffer in the ViewMarq Display.



- 1) Choose an available, unused MG address.
- 2) Click on the "Setup Screen" option to configure the instruction.

🖹 MSG - MG10:2 : (1 Elements)	
General         This Controller         Channet 0 (Integral)         Modbus Command:         03 Read Holding Registers (4xxxxx)         4         5         Data Table Address:         120         Data:         16         Data:         16         Bata:         7         Target Device         Message Timeout:         2         9         MB Data Address (1-65535):         11500         Slave Node Address (dec):         1         Modbus Address:         411500	Control Bits Ignore if timed out (TO): 0 Awaiting Execution (EW): 0 Error (ER): 0 Message done (DN): 0 Message Transmitting (ST): 0 Message Enabled (EN): 0 Error Error Code(Hex): 0

- 3) Select the Channel number that was configured earlier and that is connected to the ViewMarq.
- 4) Select Modbus Command: 03 Read Multiple Registers (4xxxxx).
- 5) In the Data Table Address field, enter in the starting Integer address of an available, unused Data Block that the Status data can be read into.
- 6) The Size in Elements field should be 120 to read the first word of the Status Block.
- 7) Choose "16 Bit" for the Data field.
- 8) Leave the Message Timeout at 2. This may need to be increased if the cable quality is suspect or if the environmental conditions are less than ideal.
- 9) The MB Data Address should be 11500 for the Command Block. This equates to 411500 in Modicon Address style.
- 10) The Slave Node Address should match the LED Display ID (Node Address) configured in the LED Display for the port you are connecting to.

Read the ViewMarq Status Block (Modbus address 411500) and put the value in N12:0



7-98 VIEW User Manual, 2nd Ed. Rev. A-MD-USEM-M

After this instruction has been enabled and is successful, the Status Reply can be read by opening the Data File where the Status data was read into and changing the Radix to ASCII.

If the syntax sent to the ViewMarq Display was correct, you will see an "OK" in your Response String. If there is a problem, you will get a Syntax error and the position where the error occurred.

🖀 Data Fi	le N12 (	ascii)									
Offset	0	1	2	3	4	5	6	7	8	9	
N12:0	OK	00/00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	^
N12:10	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	
N12:20	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	
N12:30	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	=
N12:40	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	
N12:50	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00/	
N12:60	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00/	
N12:70	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	
N12:80	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	
N12:90	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	
N12:100	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	~
										Þ	
N1:	2:0							] F	adix: ASC	CII	-
Symbol:									Col	lumns: 10	•
Desc:										,	
N12 ·		Pr	operties			<u>U</u> sage			Help		_

## Sending Long ASCII Strings over Modbus

If the ASCII String is longer than 240 characters, multiple files and multiple instructions need to be used.

1) Copy the ASCII String from the ViewMarq Software.



2) Double-click on the first Integer file location and Paste the data in and hit Enter. It should look something like the following window.

🖀 Data F	ile N7 (a	scii)	INTEGER	ł						l	
Offset	0	1	2	3	4	5	6	7	8	9	
N7:0	<1	D	1>	<c< td=""><td>LR</td><td>&gt;&lt;</td><td>WI</td><td>N</td><td>0</td><td>0</td><td>^</td></c<>	LR	><	WI	N	0	0	^
N7:10	28	7	31	~	PO	S	0	0>	<l< td=""><td>J&gt;</td><td></td></l<>	J>	
N7:20	<b< td=""><td>L</td><td>N&gt;</td><td><c< td=""><td>S</td><td>10</td><td>&gt;&lt;</td><td>GR</td><td>N&gt;</td><td><t< td=""><td></td></t<></td></c<></td></b<>	L	N>	<c< td=""><td>S</td><td>10</td><td>&gt;&lt;</td><td>GR</td><td>N&gt;</td><td><t< td=""><td></td></t<></td></c<>	S	10	><	GR	N>	<t< td=""><td></td></t<>	
N7:30	>V	</td <td>T&gt;</td> <td><c< td=""><td>S</td><td>4&gt;</td><td><t< td=""><td>&gt;i</td><td>ew</td><td>Ma</td><td></td></t<></td></c<></td>	T>	<c< td=""><td>S</td><td>4&gt;</td><td><t< td=""><td>&gt;i</td><td>ew</td><td>Ma</td><td></td></t<></td></c<>	S	4>	<t< td=""><td>&gt;i</td><td>ew</td><td>Ma</td><td></td></t<>	>i	ew	Ma	
N7:40	rq	</td <td>T&gt;</td> <td><c< td=""><td>S</td><td>1&gt;</td><td><t< td=""><td>&gt;</td><td><!--</td--><td>T&gt;</td><td></td></td></t<></td></c<></td>	T>	<c< td=""><td>S</td><td>1&gt;</td><td><t< td=""><td>&gt;</td><td><!--</td--><td>T&gt;</td><td></td></td></t<></td></c<>	S	1>	<t< td=""><td>&gt;</td><td><!--</td--><td>T&gt;</td><td></td></td></t<>	>	</td <td>T&gt;</td> <td></td>	T>	
N7:50	<c< td=""><td>ន</td><td>0&gt;</td><td><t< td=""><td>&gt;</td><td>NE</td><td>MA</td><td><!--</td--><td>T&gt;</td><td><p< td=""><td></td></p<></td></td></t<></td></c<>	ន	0>	<t< td=""><td>&gt;</td><td>NE</td><td>MA</td><td><!--</td--><td>T&gt;</td><td><p< td=""><td></td></p<></td></td></t<>	>	NE	MA	</td <td>T&gt;</td> <td><p< td=""><td></td></p<></td>	T>	<p< td=""><td></td></p<>	
N7:60	05	0	8	$\sim$	LJ	><	T>				
N7:70								4/	12	</td <td></td>	
N7:80	T>	< P	05	0	1	6>	<l< td=""><td>J&gt;</td><td><r< td=""><td>ED</td><td></td></r<></td></l<>	J>	<r< td=""><td>ED</td><td></td></r<>	ED	
N7:90	$\sim$	T>			by	<	/T	$\sim$	GR	N>	
N7:100	<t< td=""><td>&gt;A</td><td>ut</td><td>om</td><td>at</td><td>io</td><td>nD</td><td>ir</td><td>ec</td><td>t&lt;</td><td></td></t<>	>A	ut	om	at	io	nD	ir	ec	t<	
N7:110	/T	><	PO	s	0	24	><	LJ	~	T>	
N7:120				</td <td>T&gt;</td> <td><a></a></td> <td>MB</td> <td><math>\sim</math></td> <td>T&gt;</td> <td>#1</td> <td></td>	T>	<a></a>	MB	$\sim$	T>	#1	
N7:130	i	n	Su	pp	or	t<	/T	>	\00\00	\00\00/	
N7:140	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	
N7:150	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	
N7:160	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	
N7:170	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	
hun 100	100100					100100					
<u> </u>										Lean	<u> </u>
N	7:0								Radi	ASCII	
Symbol:										Column	s: 10 💌
Desc:							1			1	
N7 🕂		E	properties			<u>U</u> sage			<u>H</u> elp		

3) An additional byte (termination) must be added to the end of the data. If the String data is an odd number of bytes, the two termination characters will have to be separated between 2 different Integer addresses. Double-click on the first field and add "\d\00".

<t< th=""><th>&gt;A</th><th>ut</th><th>OL</th><th>at</th><th>10</th><th>nD</th><th>ir</th><th>ec</th><th>t&lt;</th><th></th></t<>	>A	ut	OL	at	10	nD	ir	ec	t<	
/T	$\sim$	PO	s	0	24	$\sim$	LJ	~	T>	
			</th <th>T&gt;</th> <th><a< th=""><th>MB</th><th><math>\sim</math></th><th>T&gt;</th><th>#1</th><th></th></a<></th>	T>	<a< th=""><th>MB</th><th><math>\sim</math></th><th>T&gt;</th><th>#1</th><th></th></a<>	MB	$\sim$	T>	#1	
i	n	Su	pp	or	t<	/T	>	\0d\00/	00/00	
\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	100100	100100	\00\00	\00\00	
\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	100100	\00\00	\00\00	\00\00/	N7:149
\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00/	
\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00/	-
										<u>) – </u>
138								Radi	ASCII	-
									Column	s: 10 💌
	E	roperties			<u>U</u> sage			<u>H</u> elp		
	<t /T i \00\00 \00\00 \00\00 \00\00</t 	<t>A /T &gt;&lt; i n \00\00 \00\00 \00\00 \00\00 \00\00 \00\00 \00\00 \00\00 \00\00 \00\00 \00\00</t>	<t< td="">         &gt;A         ut           /T         &gt;&lt;</t<>	<t< td="">         &gt;A         ut         om           /T         &gt;&lt;</t<>	<t< td="">     &gt;A     ut     om     at       /T     &gt;&lt;</t<>	<t< td="">         &gt;A         ut         om         at         io           /T         &gt;&lt;</t<>	<t< td="">       &gt;A       ut       om       at       10       nD         /T       &gt;&lt;</t<>	<t< td="">       &gt;A       ut       om       at       io       nD       ir         /T       &gt;&lt;</t<>	<t< td="">       &gt;A       ut       on       at       io       nD       ir       ec         /T       &gt;&lt;</t<>	<t< td="">       &gt;A       ut       om       at       io       nD       ir       ec       tc         /T       &gt;&lt;</t<>

7-100 VIEW User Manual, 2nd Ed. Rev. A-MD-USEM-M

If the String data is an even number of bytes, enter in "\0d\00" in the next Integer address after the String data.

A	<del>33333</del>
	=
/	=
4	

NOTE: Please note that when you hit enter, the display changes the \0d to ^M.

a7.00	10	<b>N</b>	0.0		*	~	~ "	0.2	~1~	60	
N7:90	$\sim$	T>			by	<	/T	$\sim$	GR	N>	
N7:100	<t< td=""><td>&gt;A</td><td>ut</td><td>on</td><td>at</td><td>io</td><td>nD</td><td>ir</td><td>ec</td><td>t&lt;</td><td></td></t<>	>A	ut	on	at	io	nD	ir	ec	t<	
N7:110	/T	$\sim$	PO	s	0	24	$\sim$	LJ	><	T>	
N7:120				</td <td>T&gt;</td> <td><a< td=""><td>MB</td><td><math>\sim</math></td><td>T&gt;</td><td>#1</td><td></td></a<></td>	T>	<a< td=""><td>MB</td><td><math>\sim</math></td><td>T&gt;</td><td>#1</td><td></td></a<>	MB	$\sim$	T>	#1	
N7:130	i	n	Su	pp	or	t<	/T	>	^M\00	100/00	_
N7:140	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	
N7:150	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00/	
N7:160	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00/	
N7:170	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00	\00\00/	
100	100100										
-											<u>) – </u>
	N7:138								Radi	ASCII	-
Symbol:										Column	s: 10 💌
Desc:											
N7		E	roperties			<u>U</u> sage			Help		

Now that the data is prepared, use multiple MSG (Read/Write Message) instructions to send the data out the port to the ViewMarq Display.

🕿 MSG - MG10:0 : (1 Elements)	
General         This Controller         Channet       [[Integral]]         Modbus Command:       15 Write Multiple Registers (4xxxxx)         Data Table Address:       N7:0         2       Size in Elements:         120       Data:         Target Device       Message Timeout :         MB Data Address (1-65536);       [11000         Slave Node Address:       [1:000         Modbus Address:       [411000	Control Bits Ignore if timed out (TO): ① Awaiting Execution (EW): ① Error (ER): ① Message done (DN): ① Message Transmitting (ST): ① Message Enabled (EN): ①

- 1) In the Data Table Address field, enter the starting address of the Integer data file that the copy and pasted data was placed into.
- 2) The Size in Elements field should be 120 registers in length for Modbus.
- 3) Choose "16 Bit" for the Data field.
- 4) The MB Data Address should be 11000 for the Command Block.

Trigger the next MSG instruction with the DN (Done) bit of the previous MSG instruction. The second MSG instruction will pick up where the first instruction left off.

🕿 MSG - MG10:1 : (1 Elements)	
General         This Controller         Channet [] [Integral]         Modbus Command:         1 Data Table Address:         N7:120         Data Table Address:         N7:120         Data:         1 Data Table Address:         N7:120         Data:         1 Data Table Address:         N7:120         Data:         1 Data Table Address:         N:120         Data:         If BB:         3         Target Device         Message Timeout:         MB Data Address (1-65536);         11120         Modbus Address:         411120	Control Bits Ignore if timed out (TO): ① Awaiting Execution (EW): ① Error (ER): ① Message done (DN): ① Message Transmitting (ST): ① Message Enabled (EN): ① Error Error Code(Hex): ①

- 1) In the Data Table Address field, enter the next value of the Integer data file where the last instruction left off (N7:0 + 120 = N7:120).
- 2) The Size in Elements field should be 120 registers in length for Modbus.
- 3) Choose "16 Bit" for the Data field.
- 4) The MB Data Address should be 11000+120 = 11120 for the Command Block.

Write the first 120 words of the ASCII String stored starting at N7:0 to the ViewMarq ASCII Command Block (Modbus address 411000)

B3:0 B3:1 [ONS] 0 0	MSG - Read/Write Message MSG File MG10:0 Setup Screen < ER
Write the next 120 words of the ASCII String stored starting at	N7:120 to the ViewMarq ASCII Command Block (Modbus address
411120)	MSG
MG10:0 B3:1	Read/Write Message
ONS	MSG File MG10:1
DN 1	Setup Screen <

7

7-102

VIEW LED MEDSAGE DISPLAN User Manual, 2nd Ed. Rev. A-MD-USEM-M

## Allen Bradley CompactLogix and ControlLogix PLCs

The Allen Bradley ControlLogix and CompactLogix PLCs both utilize RS-232 and the ASCII protocol to send ASCII strings to the ViewMarq LED display.

## **RS-232**

Connect the ViewMarq Port 1 terminal block to the Allen Bradley 9-pin D-Sub port. The cable for this will need to be constructed by the user.



## ASCII

In the LED Display Configuration window configure the ViewMarq RS-232 port to communicate using ASCII.

LED Display Configuration	×			
LED Display Selection	Configuration File			
Display Name: ViewMarq	Import from File			
Link Name: Ethernet	Export to File			
Change Link Blink	Global Settings			
Read Configuration from ViewMarg	Identification			
	Name: ViewMarg			
	LED Display ID: 1 🚖			
Ethernet HS-232 RS-485	Heartbeat			
Basic Configuration	Timeout 0 seconds			
Protocol: ASCII -				
Node-Address [1-247]:	ASCII			

Configure the Allen Bradley PLC port to communicate ASCII with the ViewMarq LED Display.

1) In RSLogix5000, open the Controller Properties for the PLC.

Date/Time	Advanced	SFC Execution F	ile Nonvolatile	e Memory	Memory Security
General	Serial Port	System Protocol	User Protocol	Major Faul	Its Minor Faults
Mode:	a) User 🗸		Show Offline	Values	
Baud Rate:	38400 🗸	5			
Data Bits:	8 🗸				
Parity:	Odd 🔽				
Stop Bits:	1 💌	J			
Control Line:	No Handshake	•			
	Continuous	Carrier			

- 2) On the Serial Port tab:
  - a. Set the Mode to User.
  - b. Match the Baud Rate, Parity, Stop Bits and Data Bits to match those of Port 1 of the ViewMarq LED Display.

Date/Time	Advance	ed SI	FC Execution	File	Nonvolati	e Memory	Memory	Security
General	Serial P	ort	System Proto	col	User Protocol	Major Fa	aults I	Minor Faults
Protocol:		ASCII	<b>a</b>	*				
Read/Write Bu	ffer Size:	500	C (Bytes)					
Termination Ch	aracter 1:	'\$1'	C) 2	2 '\$FF'				
Append Chara	oter 1:	21,	i 🛈 2	'\$FF'				
XON/XOFF			-					
Echo Mode								
Delete Mode								
<ul> <li>Ignore</li> </ul>								
OCRT								

3) On the User Protocol tab:

- a. Set the Protocol to ASCII.
- b. Set the Buffer size to 500 Byte. This is the maximum String the ViewMarq can accept.

7-104 VIEW LED MESSAGE DISPLAY User Manual, 2nd Ed. Rev. A-MD-USEM-M

- c. If the ASCII Reply from the ViewMarq will be read, set the Termination Characters as shown.
  - Termination Character 1 '\$r' is a Carriage Return
  - Termination Character 2 '\$FF' means not used
- d. Set the Append Characters as shown. These will be appended to the end of the String that is sent.
  - Append Character 1 '\$r' is a Carriage Return
  - Append Character 2 '\$FF' means not used

In order for the PLC to send the String it must be in the PLC Memory. A String data type 500 characters long is needed.

- 1) In RSLogix5000 create a new String Data Type.
- 2) Set Maximum Characters to 500.

Name:	VM_String_Data_Type								
Description:									
Maximum Characters:	Maximum Characters: 500								
Members:	Members: Data Type Size: 504 byte(s)								
Name	Data Type	Style	Description	External Access					
LEN	DINT	Decimal		Read/Write					
DATA	SINT[500]	ASCII		Read/Write					

Use this new String Data Type when creating new strings to send to the ViewMarq.

New Tag		×
Name:	VM_Message1	Create 🗸 🗸
Description:		Cancel Help
Usage:	<normal></normal>	
Туре:	Base Connection	
Alias For:	~	
Data Type:	VM_String_Data_Type	
Scope:	🗓 Lab_CompactLogix_L35E 🛛 👻	
External Access:	Read/Write	
Style:	×	
Constant		
🗌 Open Confi	guration	

7-106 VIEW User Manual, 2nd Ed. Rev. A-MD-USEM-M

7) Create a New String Tag using the Data Type just created.

New Tag		X
Name:	VM_Msg1_Write_Ctrl	Create 🛛 🔻
Description:		Cancel
		Help
	<ul> <li>X</li> <li>X</li> </ul>	
Usage:	<normal></normal>	
Туре:	Base Connection	
Alias For:	×	
Data Type:	SERIAL_PORT_CONTROL	
Scope:	🗓 Lab_CompactLogix_L35E 🛛 👻	
External Access:	Read/Write	
Style:		
Constant		
🗌 Open Conl	figuration	

8) Create a New Tag using the predefined SERIAL\_PORT\_CONTROL Data type.

9) Copy the ASCII String from the ViewMarq Software.

Ĭ.	Command String	Сору 📋					~~~~	103 / 500	
	<id 0=""><clr><win 0<="" th=""><th>0 287 31:</th><th>&gt;<pos 0=""><l< th=""><th>J&gt;<bl n=""><cs< th=""><th>0&gt;<grn><t>Hello</t></grn></th><th>World<pos 0<="" th=""><th>) 8&gt;<lj><t>ViewMa</t></lj></th><th>arq</th><th></th></pos></th></cs<></bl></th></l<></pos></th></win></clr></id>	0 287 31:	> <pos 0=""><l< th=""><th>J&gt;<bl n=""><cs< th=""><th>0&gt;<grn><t>Hello</t></grn></th><th>World<pos 0<="" th=""><th>) 8&gt;<lj><t>ViewMa</t></lj></th><th>arq</th><th></th></pos></th></cs<></bl></th></l<></pos>	J> <bl n=""><cs< th=""><th>0&gt;<grn><t>Hello</t></grn></th><th>World<pos 0<="" th=""><th>) 8&gt;<lj><t>ViewMa</t></lj></th><th>arq</th><th></th></pos></th></cs<></bl>	0> <grn><t>Hello</t></grn>	World <pos 0<="" th=""><th>) 8&gt;<lj><t>ViewMa</t></lj></th><th>arq</th><th></th></pos>	) 8> <lj><t>ViewMa</t></lj>	arq	
							NUM		

10) In RSLogix5000 open the String data file just created and paste the ASCII String into the file.

-VM_Message1			VM_String_Data_Type
+ VM_Message1.LEN	0	Decimal	DINT
+ VM_Message1.DATA	{}	ASCII	SINT[500]

👪 String Browser - VM_Message1*	×
	\$\$
<grn><t>Hello World</t></grn>	\$1
	۶L
	\$N
	\$P
	\$R
	\$T
OK Cancel Apply Help	
🅐 🥸 0 Error(s) 0 🛛 INS 👘 75 of 500	

Now that the Port has been configured and the ASCII String is in the PLC Tag, use an AWA (ASCII Write Append) instruction to send the string out the serial port.

AWA		
ASCII Write Append		
Channel	0	
Source VM_Messa	ge1	
' <id 0=""><clr><win '="" 0="" td="" ←<=""></win></clr></id>		
SerialPort Control VM_Msg1_W	ite_	
	Ctrl	
SerialPort Control Length	0 🗲	
Characters Sent	0 🗲	

- 1) Select the Channel number that was configured earlier and that is connected to the ViewMarq.
- 2) For the Source select the String Data File the ASCII String was Pasted into.
- For the Serial Port Control choose the SERIAL\_PORT\_CONTROL tag created above.

```
Send a string to the ViewMarg
      Enable the AVVA (ASCII Write Append) instruction to send String VM_Message1 of type VM_String out Channel 0.
           Termonation characters configured in the Serial Port configuration are appended to the End of the string.
VM_Msg1_Write
                                                                                             -ΑνγΑ
                                                                              ASCII Write Append
                                                                                                                      (EN)
                                                                              Channel
                                                                                                                Π
                                                                              Source
                                                                                                   VM_Message1
                                                                                           '<ID 0><CLR><WIN 0 0 ' •
                                                                              SerialPort Control VM_Msg1_Write_
                                                                                                                      (ER)
                                                                                                              Ctrl
                                                                              SerialPort Control Length
                                                                                                                0 <
                                                                              Characters Sent
                                                                                                                0 🗧
```

7

7-108

## VIEW LED MESSAGE DISPLANT User Manual, 2nd Ed. Rev. A – MD-USEM-M
## **Reading the ASCII Reply**

If the "ASCII Reply" option is enabled in the "LED Display Configuration" settings AND you are sending to an ID other than 0, your PLC may Read the Reply Response String that is sent by the ViewMarq LED Display.

1) Create a New String Tag using the new String Data Type created.

General	rties - VM_Message1_Reply	
Name:	VM_Message1_Reply	
Description:		
Туре:	Base Connection	
Alias For:	~	
Data Type:	VM_String_Data_Type	
Scope:	🗗 Lab_CompactLogix_L35E	
External Access: Style:	Read/Write	
Constant		
[	OK Cancel Apply	Help

7-110 VIEW User Manual, 2nd Ed. Rev. A-MD-USEM-M

👸 Tag Prope	rties - VM_Msg1_Read_Reply_Ctrl 💦 🔲 🔀
General	
Nama	VM Must Deed Deels Of
ivanie.	
Description:	<u>^</u>
Type:	Page II Connection
Alias For:	~
Data Type:	SERIAL PORT CONTROL
Carrow	
Scope:	Ba Lab_CompactLogix_L35E
External Access:	Read/Write
Style:	~
Constant	
Constant	
	OK Cancel Apply Help

2) Create a New Tag using the predefined SERIAL\_PORT\_CONTROL Data type.

Use an ABL (ASCII Test For Line) instruction to check for the Termination characters entered in the Channel configuration and then an ARL (ASCII Read Line) instruction to read in the Status reply response String.

- 1) For the ABL instruction, select the Channel number that was configured earlier and that is connected to the ViewMarq.
- 2) For the Serial Port Control choose the SERIAL\_PORT\_CONTROL tag created above.
- 3) Trigger the ABL instruction with a normally closed contact using the Control File Enable to trigger the instruction each PLC scan.
- 4) For the ARL instruction, , select the Channel number that was configured earlier and that is connected to the ViewMarq.
- 5) For the Serial Port Control choose the SERIAL\_PORT\_CONTROL tag created for Read Reply above.
- 6) Trigger the ARL instruction with the Found (FD) bit of the ABL instruction Control File.

Read the ASCII Reply

Enable an ABL(Test Buffer for Line) instruction every scan to check the input buffer.

VM_Msg1_Read_C	
	ASCILTest For Buffer Line
	SerialPort Control VM_Msg1_Read_C -(ER)-
	Character Count 0 🗧
If any characters are found (FD), read the inp VM_Mod1_Read_C	put buffer VM_Msg1_Reply with an ARL (ASCII Read Line) instruction.
If any characters are found (FD), read the inp VM_Msg1_Read_C trl.FD	put buffer VM_Msg1_Reply with an ARL (ASCII Read Line) instruction.
If any characters are found (FD), read the ing VM_Msg1_Read_C trl.FD ]	put buffer VM_Msg1_Reply with an ARL (ASCII Read Line) instruction.
If any characters are found (FD), read the inp /M_Msg1_Read_C trl.FD ]	put buffer VM_Msg1_Reply with an ARL (ASCII Read Line) instruction.  ARL  ASCII Read Line  Channel  Destination VM_Message1_Re  CDN)—
If any characters are found (FD), read the inp VM_Msg1_Read_C trl.FD ]	put buffer VM_Msg1_Reply with an ARL (ASCII Read Line) instruction. ARL ASCII Read Line Channel Destination VM_Message1_Re ply
If any characters are found (FD), read the ing VM_Msg1_Read_C trl.FD ]	put buffer VM_Msg1_Reply with an ARL (ASCII Read Line) instruction.  ARL  ASCII Read Line  Channel  Destination VM_Message1_Re  ply  (\$IOK\$r' ←(ER)—
If any characters are found (FD), read the ing VM_Msg1_Read_C trl.FD ][]	put buffer VM_Msg1_Reply with an ARL (ASCII Read Line) instruction. ARL ASCII Read Line Channel Destination VM_Message1_Re ply "\$IOK\$r' ← (ER) SerialPort Control VM_Msg1_Read_R eply_Ctrl
If any characters are found (FD), read the ing VM_Msg1_Read_C trl.FD ] []	put buffer VM_Msg1_Reply with an ARL (ASCII Read Line) instruction. ARL ASCII Read Line Channel Destination VM_Message1_Re ply "\$IOK\$r' ← (ER) SerialPort Control VM_Msg1_Read_R eply_Ctrl SerialPort Control Length 499 ←

If the syntax sent to the ViewMarq Display was correct, you will see an "OK" in the String Tag . If there is a problem, you will get a Syntax error and the position where the error occurred.