COMMUNICATIONS

TABLE OF CONTENTS

SR55 Communications Overview
Modbus Serial Communications
Modbus RTU Communications Interface
Modbus RTU Connections
Modbus Communications Configuration
Transmission Modes
Message Structure For RTU Mode
Supported Functions
Метогу Мар
Message Timing
Network Communications – EtherNet/IP and Modbus TCP
Communication Module Overview
Module Installation – SR55-CM-ENETIP and SR55-CM-MODTCP
SR55 Configuration
IP Address Configuration
Communication Module Front Panel Indicator Lights
Modbus TCP Network Communications
EtherNet/IP Network Communications.
EtherNet/IP Control
EDS File
Using the IP Configuration Tool (IPconfig)
Connecting to the SR55-CM-ENET Module through I/O (Implicit Messaging) 5–14
Connecting to the SR55-CM-ENET Module through Explicit Message:
Explicit Message Instruction Examples (from Productivity Series CPU)

CHAPTER

5

SR55 COMMUNICATIONS OVERVIEW

MODBUS SERIAL COMMUNICATIONS OVERVIEW

All SR55 soft starters have a built-in RJ12 serial port that can be used to configure and to control one SR55 from one RS-485 master controller, with no additional communications components required (other than RS-485 cabling). Multiple SR55 soft starters can be controlled from a single RS-485 master controller with the use of one optional serial Modbus communication splitter (SR55-SPLT) per soft starter. An RJ45 female to RJ12 male cable adapter (SR55-RJ45-RJ12) is available for easier cable connection. Examples of Modbus masters are SR55-KPD-REM, PLC, or HMI.

MODBUS TCP NETWORK COMMUNICATIONS OVERVIEW

Multiple SR55 soft starters can be networked for configuration and control from a single Modbus TCP master controller. This type of network control requires one optional Modbus TCP communication module (SR55-CM-MODTCP) per SR55 soft starter.

ETHERNET/IP NETWORK COMMUNICATIONS OVERVIEW

Multiple SR55 soft starters can be networked for configuration and control from a single EtherNet/IP master controller. This type of network control requires one optional EtherNet/IP communication module (SR55-CM-ENETIP) per SR55 soft starter.



For communications, ensure that the SR55 "Timeout" parameter setting is > 0. Otherwise, the SR55 will fault as soon as communication is enabled. (Home \rightarrow Device \rightarrow Networks \rightarrow Timeout ms)

IF USING MODBUS RTU / MODBUS TCP / ETHERNET/IP COMMUNICATION FOR <u>CONTROL</u>, THE DIGITAL INPUTS ARE DISABLED. THE DIGITAL OUTPUTS WILL STILL FUNCTION AS CONFIGURED.

IF USING MODBUS RTU / MODBUS TCP / ETHERNET/IP COMMUNICATION FOR <u>MONITORING ONLY</u>, THEN THE DIGITAL INPUTS AND OUTPUTS WILL FUNCTION AS CONFIGURED IF THE CONTROL METHOD IS SET TO USER PROGRAMMABLE, TWO WIRE, OR THREE WIRE CONTROL.



IF USING MODBUS RTU (RJ12 PORT) FOR COMMUNICATIONS, AN SR55-CM-XXXX MODULE MUST NOT BE INSTALLED IN THE BOTTOM COMMUNICATION PORT. THE PRESENCE OF A COMMUNICATIONS MODULE WILL CAUSE INTERFERENCE WITH THE MODBUS RTU COMMUNICATIONS.

MODBUS SERIAL COMMUNICATIONS

MODBUS RTU COMMUNICATIONS INTERFACE

All SR55 soft starters support serial Modbus RTU protocol (slave) communications. The serial RS-485 communications are accessible from the built-in RJ12 port, as shown below.



SERIAL MODBUS COMMUNICATION SPLITTER

Information for the optional SR55-SPLT serial Modbus communication splitter is found in "Accessories" Chapter 6. (An SR55-RJ45-RJ12 adapter can be used to simplify cabling between the splitter's RJ45 ports and the SR55's RJ12 port.)

MODBUS SERIAL COMMUNICATIONS (CONTINUED)

MODBUS COMMUNICATIONS CONFIGURATION

The Modbus communication settings are accessible from the Device menu:

- Device >> Networks >> Modbus Network Settings >> Address (1 32)
- Device >> Networks >> Modbus Network Settings >> Baud (9600 115200)
- Device >> Networks >> Modbus Network Settings >> Parity (Odd / Even)
- (Data bits = 8, Stop bits = 1)

The communication parameters should be set before connecting the Modbus master.

TRANSMISSION MODES

ASCII and RTU transmission modes are defined in the Modbus protocol specification. SR55 uses *only the RTU mode* for the message transmission.

MESSAGE STRUCTURE FOR RTU MODE

The Modbus RTU structure uses a master-slave system for message exchange. In the case of the SR55 system, it allows up to 32 slaves, and one master. Every message begins with the master making a request to a slave, which responds to the master in a defined structure. In both messages (request and answer), the used structure is the same:

• Address, Function Code, Data and CRC.

MASTER (REQUEST MESSAGE):

Address	Address Function		CRC	
(1 byte)	(1 byte)	(n bytes)	(2 bytes)	

SLAVE (RESPONSE MESSAGE):

Address	Function	Response Data	CRC
(1 byte)	(1 byte)	(n bytes)	(2 bytes)

ADDRESS

The master initiates the communication by sending a byte with the address of the destination slave. When responding, the slave also initiates the message with its own address. Broadcast to address 0 (zero) is not supported.

FUNCTION CODE

This field contains a single byte, where the master specifies the type of service or function requested to the slave (reading, writing, etc.). According to the protocol, each function is used to access a specific type of data.

DATA FIELD

The format and contents of this field depend on the function used and the transmitted value.

CRC

The used method is the CRC-16 (Cyclic Redundancy Check). This field is formed by two bytes; where first the least significant byte is transmitted (CRC-), and then the most significant (CRC+). The CRC calculation form is described in the Modbus RTU protocol specification.

MODBUS SERIAL COMMUNICATIONS (CONTINUED)

SUPPORTED FUNCTIONS

Modbus RTU specification defines the functions used to access different types of data.

- SR55 parameters are defined as *holding type registers*.
- For Modbus RTU/TCP Client devices that use Modicon style addressing, place a 4 as the high digit followed by the Modbus address defined in the parameter mapping table. Note that SR55 Modbus addressing starts at zero; not 1 as some devices do.
- SR55 32-bit parameters are High Word / Low Word in Modbus format.

The following services are available:

READ HOLDING REGISTERS

Description: reading register blocks of holding register type (block R/W limited to 8 registers).

• Function code: 03

Modbus Function 03 Transaction Table							
Query		Response					
Field	Hex Byte	Field	Hex Byte				
Slave address	01	Slave address	01				
Function	03	Function	03				
Start address Hi	00	Byte count	02				
Start address Lo	01	Data Hi	01				
No of registers Hi	00	Data Lo	2C				
No of registers Lo	01	CRC Lo	B8				
CRC Lo	D5	CRC Hi	09				
CRC Hi	СА						

Before writing to the SR55, initiate several Modbus Reads to ensure that the master's addressing and configuration are correct.

WRITE SINGLE REGISTER

Description: writing in a single register of the holding type.

• Function code: 06

Modbus Function 06 Transaction Table						
Query		Response				
Field	Hex Byte	Field	Hex Byte			
Slave address	01	Slave address	01			
Function	06	Function	06			
Address Hi	00	Address Hi	02			
Address Lo	0C	Address Lo	0C			
Force data Hi	00	Force data Hi	00			
Force data Lo	09	Force data Lo	09			
CRC Lo	48	CRC Lo	88			
CRC Hi	0C	CRC Hi	77			

MODBUS SERIAL COMMUNICATIONS (CONTINUED)

WRITE MULTIPLE REGISTERS

Description: writing register blocks of holding register type (block R/W limited to 8 registers).

• Function code: 16

Modbus Function 16 Transaction Table							
Query		Response					
Field	Hex Byte	Field	Hex Byte				
Slave address	01	Slave address	01				
Function	16	Function	16				
Address Hi	00	Address Hi	02				
Address Lo	0C	Address Lo	0C				
Force data Hi	00	Force data Hi	00				
Force data Lo	09	Force data Lo	09				
CRC Lo	48	CRC Lo	49				
CRC Hi	0C	CRC Hi	B4				

MEMORY MAP

SR55 Modbus communication is based on reading or writing equipment parameters from or to the holding registers. The data addressing is zero offset, such that the parameter Modbus address corresponds to the register number.

Modbus Address Memory Map							
Parameter	Modbus Data Address						
Modbus Address	Decimal	Hexadecimal					
0000	0	0000h					
0001	1	0001h					
•	•	•					
•	•	•					
•	•	•					
•	•	•					
0128	128	0080h					
•	•	•					
•	•	•					
•	•	•					
•	•	•					

MESSAGE TIMING

In the RTU mode there is no specific start or stop byte that marks the beginning or the end of a message. Indication of when a new message begins or when it ends is achieved by the absence of data transmission for a minimum period of 3.5 times the transmission time of a data byte. Thus, in case a message is transmitted after this minimum time has elapsed; the network elements will assume that the first received character represents the beginning of a new message.



NETWORK COMMUNICATIONS – ETHERNET/IP AND MODBUS TCP

COMMUNICATION MODULE OVERVIEW

Two communication modules are available which allow network communication and control for the SR55 soft starter.

- SR55-CM-ENETIP for EtherNet/IP network communication.
- SR55-CM-MODTCP for Modbus network communication.

Both modules have two RJ45 ports for daisy chaining to multiple starters. These ports act as a switch. It does not matter which port you connect to.

Install the applicable communication module in the SR55 option module slot per the hardware installation instructions in the "Accessories" chapter of this user manual.

⚠

REFER TO THE INSTALLATION INSTRUCTIONS IN THE "ACCESSORIES" CHAPTER OF THIS USER MANUAL BEFORE ATTEMPTING TO INSTALL THE COMMUNICATION MODULES.

MODULE INSTALLATION – SR55-CM-ENETIP AND SR55-CM-MODTCP



Great care must be taken to properly seat the communication modules into the SR55 soft starter without damaging the connection pins. Refer to the detailed instructions in the "Accessories" chapter of this user manual.



SR55-CM-ENETIP

SR55-CM-MODTCP

SR55 CONFIGURATION

The SR55 will automatically configure when the option module is installed.

IP ADDRESS CONFIGURATION

Use the IP address configuration tool available from: http://support.automationdirect.com/downloads.html

NETWORK COMMUNICATIONS – ETHERNET/IP AND MODBUS TCP (CONTINUED)

COMMUNICATION MODULE FRONT PANEL INDICATOR LIGHTS

FRONT PANEL INDICATORS

	Location of	Front	Panel Indicators						
Item Front Panel Diagram					Network Interface LED				
1	Network Status LED		5 ~	- 6		LED Stat	e	Description	
2	Module Status LED			- 2	,	Off		No link, no activity	
3	Network Interface, P	ort 1		-	Green		Link established (100 Mbit/s)		
 4 Network Interface, Port 2 5 Link/Activity Port 1 				Green, flickering		Activity (100 Mbit/s)			
					Yellow		Link established (10 Mbi	t/s)	
6	Link/Activity Port 2		<u> </u>	D		Yellow, fli	ckering	Activity (10 Mbit/s)	
Network Status LED						Mod	ule St	atus LED	
LED State Description			LED State Descri		Descrip	iption			
	0"		10 11		0.0				

LED State	Description
Off	No power or no IP address
Green	Online, connections active
Green, flashing	Online, no connections active
Red	Duplicate IP, fatal error
Red flashing	Connection timeout

Module Status LED					
LED State	Description				
Off	No power				
Green	Controlled, Run state				
Green, flashing	Not configured or idle state				
Red	Major fault				
Red, flashing	Recoverable error(s)				

MODBUS TCP NETWORK COMMUNICATIONS

The SR55-CM-MODTCP Modbus communication module offers the following functionality:

- · Dual switched RJ45 communication ports
- 256 bytes of I/O data in each direction
- 100 Mbps full duplex
- · Supports 4 simultaneous (master) connections

All Modbus functions and addresses that are available in the preceeding "Modbus Serial Communications" section of this chapter are also available via modus TCP.

```
SR55 uses Protocol Addressing (Base O); not PLC Addressing (Base 1). If
you are not using the correct selection, all the addresses will be off
by 1. Recommended test: monitor a non-critical parameter such as Start
Time (address 7104), then manually change the value on the touchscreen
and verify that Modbus master actually sees the correct changes.
```

ETHERNET/IP NETWORK COMMUNICATIONS

The SR55-CM-ENETIP EtherNet/IP communication module offers the following functionality:

- CIP Parameter Object Support
- · Implicit and Explicit messaging
- Dual switched RJ45 communication ports
- 10/100 Mbps full duplex
- 2 Input Words from the network master to the SR55
- 2 Output Words from the SR55 to the network master

ETHERNET/IP CONTROL

	Supported Parameters							
#	Description	Read Only?	Implemented?					
1	Run Forward	N	Y					
2	Run Reverse	N	Ν					
3	Fault Rest	N	Y					
4	Net Control	N	Y					
5	Net Reference	N	N					
6	Speed Reference	N	N					
7	Torque Reference	N	N					
8	Faulted	Y	Y					
9	Warning	Y	Y					
10	Running Forward	Y	Y					
11	Running Reverse	Y	Ν					
12	Ready	Y	Y					
13	Ctrl From Net	Y	Y					
14	Ref From Net	Y	N					
15	At Reference	Y	N					
16	Drive State	Y	Y					

The drive profile used by the interface is currently that provided by the SR55-CM-ENETIP Module and is dictated by the EDS file provided by HMS Industrial Networks.

The EDS describes parameters that can be accessed explicitly in an Acyclic manner. Not all of these parameters are implemented in the SR55 soft starter. See the following table. CIP paths from these parameters are described in the EDS.

The EDS also describes the 25 Implicit Cyclic connections, each of which will set and/or get a combination of the above parameters. The following examples are for connection 6 (Extended Control).

CIP Packet functionality – Extended Control										
O -> T Packet (Control)	Packet (Control) Bit 7 Bit 6 Bit 5 Bit 4 Bit 3 Bit 2 Bit 1 Bit 0									
Byte 0	-	-	#4	-	#3	-	-	#1		
Byte 1	-	-	-	-	-	-	-	-		
T -> O Packet (Status)	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit O		
Byte 0	-	-	#13	#12	-	#10	#9	#8		
Byte 1	#16									

Note: When a cyclic connection is established and Bit4 (Net Control) is set, the network has control of the SR55 soft starter and any other control from the SR55 front touchscreen, switches, or Modbus interface will be overridden.

EDS FILE

An EDS file for the interface is available from the AutomationDirect website:

http://support.automationdirect.com/downloads.html

USING THE IP CONFIGURATION TOOL (IPCONFIG)

The IP address of the SR55 is set using the Anybus IPconfig utility available from: <u>http://support.automationdirect.com/downloads.html</u>.

This section explains how to install the IPconfig utility and how to set the SR55's IP address.

Unzip the file to a temporary folder and run the executable.



Follow the steps through the installation.

Once the installation is complete, run application from the folder that it was installed to (usually the HMS folder in the Start menu). In Windows 8, from the home tile screen, simply type in "ipconfig." Typing any text on this screen will open the Search dialog. The program IPconfig is the configuration tool from HMS.

Search	
ipconfig	P
IPconfig	
ipconfig.exe	

If you use the Desktop most of the time in Windows 8, right click on this file and select "Pin to Taskbar" to always have quick access to the file from the desktop.

The SR55 with the installed SR55-CM-ENETIP module needs to be installed on the same network as the PC running the Ipconfig application.

The messaging uses broadcast and will not go through routers.

```
It is highly recommended to disconnect the PC from any network and have only the SR55 and the PC connected via an Ethernet switch (<u>not a router</u>) or an Ethernet cross-over cable.
```

Use a switch or crossover cable to connect the starter to the SR55.

Start the Ipconfig software. Press the Scan button to have the PC scan for an SR55. The IPconfig utility will automatically find the SR55 units on the network.

If the SR55 is not found, click on the Settings button, then choose "Broadcast from a specific Network Interface Controller." This could be required if there are multiple network cards on the PC. Click OK, then Scan for SR55 units again.

Below shows a screen capture of an SR55-077 and an SR55-242 daisy chained from one to another.

IPconfig						
IP /	SN	GW	DHCP	Version	Туре	MAC
0.0.0 10.11.0.234	0.0.0.0 255.255.0.0	0.0.0.0 0.0.0.0	On Off	1.02.1 1.02.1	Anybus-CC Modbus-TCP (2-Port) Anybus-CC Modbus-TCP (2-Port)	00-30-11-0E-02-09 00-30-11-0E-01-F1
					Settings Scan	Exit

Select the Anybus module in the window and double click on it. This will bring up a window to set the network settings to values appropriate to the network that it will be running on.



```
It is very highly recommended to set DHCP to OFF. Otherwise, the
Ethernet address of the SR55 could be changed by a DHCP server at a
future time. Setting DHCP to OFF will ensure that the SR55 maintains
the same IP address.
```

Ethernet configura	tion							
IP address:	10		11	1	0	1	236	
Subnet mask:	255		255		0		0	C On
								• Off
Default gateway:	0	÷	0	4	0	÷	0	
Primary DNS:	0	÷	0		0		0	
Secondary DNS:	0	÷	0	a.	0		0	
Hostname:		_		_		_		
Password:		_						Change password
New password:		_						

Once the settings have been entered, click on the "Set" button and the Anybus module is now configured and ready to be used. It is not recommended to use DHCP, as the address could be changed. The starter's control power will have to be cycled for the correct settings to show up on the touchscreen.

Networks	Networks	Networks	Anybus Information
Modbus Network Settings	Modbus Network Settings	Modbus Network Settings	Address: 169.254.1.1
Anybus	ModbusTCP	Ethernet IP	Serial Number: 0xa0224493
Timeout mS 5000	Timeout mS 5000	Timeout mS 5000	FW Version: 1.02 Build 1
			Connection: No [0]
BACK STATUS START	BACK STATUS START	BACK STATUS START	BACK STATUS START

TROUBLESHOOTING

If you do not see modules showing up in the IPconfig screen check the following:

- That the SR55-MODTCP or SR55-ENETIP module is inserted correctly, and MS LED is on or flashing green. See module installation instructions in this SR55 user manual.
- That the module appears in the Networks menu under Home >> Device >> Networks. If the module is not recognized the center selection text will read "Anybus" instead of "ModbusTCP or "Ethernet IP."
- On the PC, run "cmd" from the Start Menu (or type "cmd" from the Windows 8 Home tile screen) to get a command prompt. Test the physical connection between the PC and the starter. Type "ping" and the address the SR55 should be set to. Press Enter.
- If the PC can see the starter, valid data will be returned:

Command Prompt	_	×
C:\>ping 10.11.0.236		^
Pinging 10.11.0.236 with 32 bytes of data: Reply from 10.11.0.236: bytes=32 time=2ms TTL=30 Reply from 10.11.0.236: bytes=32 time=1ms TTL=30 Reply from 10.11.0.236: bytes=32 time=1ms TTL=30 Reply from 10.11.0.236: bytes=32 time=1ms TTL=30		
Ping statistics for 10.11.0.236: Packets: Sent = 4, Received = 4, Lost = 0 (0% loss), Approximate round trip times in milli-seconds: Minimum = 1ms, Maximum = 2ms, Average = 1ms		
C:\>		\sim
<		>

• If the PC is set to a different IPv4 network than the SR55, ie., both PC and SR55 are not set to the same first two octets (10.11.xxx.xxx in this example), the following error will be returned:

Command Prompt	_ □	×
C:\>ping 10.11.0.236		^
Pinging 10.11.0.236 with 32 bytes of data: Reply from 172.20.1.7: TTL expired in transit. Reply from 172.20.1.7: TTL expired in transit. Reply from 172.20.1.7: TTL expired in transit. Reply from 172.20.1.7: TTL expired in transit.		
<pre>Ping statistics for 10.11.0.236: Packets: Sent = 4, Received = 4, Lost = 0 (0% loss), C:\>_</pre>		
5		~
N N N N N N N N N N N N N N N N N N N		

• If the IPv4 Ethernet address is incorrect, the following error will be returned (notice that unlike the previous error, this error returns "Destination host unreachable"):



- If Ping from the PC to the SR55 does not work, please recheck that a cross-over cable or an Ethernet switch (NOT a router) is being used to connect the PC to the SR55.
- Also check that the header pins between the comm module and the SR55 were not bent (extreme care must be taken when inserting the module into the starter).

CONNECTING TO THE SR55-CM-ENET MODULE THROUGH I/O (IMPLICIT MESSAGING)

The connection parameters for Connection 6 (Extended Control) are as follows:

- T->O (Input Data) Connection Point Assembly Instance value is 71.
- T->O (Input Data) Size is 4 bytes.
- The Data format for Status is shown in the "Input Data Setup" screen capture.
- O->T (Output Data) Connection Point Assembly Instance value is 21.
- O->T (Output Data) Size is 4 bytes.
- The Data format for Control is shown in the "Output Data Setup" screen capture.
- No Configuration data is required.

To start the SR55, a value of 33 should be placed into Byte 0 of the Control data. 33 equates to Bit 0 (Run Forward) On and Bit 5 (Net Control) On.

To stop the SR55, a value of 32 should be placed into Byte 0 of the Control data. 32 equates to Bit 0 Off and Bit 5 On.

To reset faults on the SR55, a value of 36 should be placed into Byte 0 of the Control data. 36 equates to Bit 2 (Fault Reset) On and Bit 5 (Net Control) On.

The following images are an example setup of I/O (Implicit Messaging) to the SR55 EtherNet/IP adapter from a Productivity Series CPU.

INPUT DATA SETUP

	myDevice		TCP Connected	tcpConn	-		
Ethernet Port	CPU-ETH-Ext	•	Adapter Name	adapNam	ne 👻		
IP Address	10.11.0.236		Vendor ID	venID	•]	
CP Port Number	44818		TCP/IP Error	tcpIPErr	-]	
EXP I	Swap Byte	Order					
Enable	enableIO	•	Connecti	on On <mark>l</mark> ine	connOnlineIO	•	
			Gener	al Status	genStatIO	- [)
📄 Enable Ro	outing Slot N	umber	Gener	al Status	genStatIO extStatIO	• [
Enable Ro	outing Slot N	umber	Gener 0 Extende Status De	al Status ed Status escription	genStatIO extStatIO statDescripIO	• (• (
Enable Ro	outing Slot N (INPUT) O->T Originator (INPLE	(OUTPUT) CO	Gener O Extende Status De	al Status ed Status escription	genStatIO extStatIO statDescripIO	• (• (• (··· ···
T->O (Target To (OUTING Slot N (INPUT) O->T Originator (INPU Delivery Option	(OUTPUT) CO T) Data Multicast	Gener 0 Extends Status De VFIG DATA	al Status ed Status escription	genStatIO extStatIO statDescripIO	• (• (• (
Enable Ro T->O (Target To (R	(INPUT) O->T Originator (INPUT) Delivery Option PI Time (msec)	(OUTPUT) CO T) Data Multicast 250	Gener 0 Extende Status De NFIG DATA	al Status ed Status escription	genStatIO extStatIO statDescripIO	• (• (• (
T->O (Target To C R Cc	(INPUT) O->T Originator (INPU Delivery Option PI Time (msec)	(OUTPUT) CO () Data Multicast 250 71 (Gener 0 Extende Status De VFIG DATA 0x47)	ed Status escription	genStatIO extStatIO statDescripIO	• (• (
Enable Ro T->O (Target To (E R Co	(INPUT) O->T Originator (INPU Delivery Option PI Time (msec) onnection Point Datatype:	(OUTPUT) COT T) Data Multicast 250 71 (Integer, 8 Bit Un	Gener O Extende Status De VFIG DATA Ox47) signed, 1D Array	ed Status escription	genStatIO extStatIO statDescripIO	• (• (• (
T->O (Target To (R Co	OUTING Slot N (INPUT) O->T Originator (INPU Delivery Option SPI Time (msec) Donnection Point Datatype: Data Array	(OUTPUT) COI T) Data Multicast 250 71 (Integer, 8 Bit Un inArrayIO	Gener 0 Extende Status De VFIG DATA 0x47) signed, 1D Array	ed Status escription	genStatIO extStatIO statDescripIO	• (• (• (
T->O (Target To (R Co	OUTING Slot N (INPUT) O->T Originator (INPU Delivery Option PI Time (msec) onnection Point Datatype: Data Array the Size (bytes):	(OUTPUT) COI T) Data Multicast 250 71 (Integer, 8 Bit Un inArrayIO 4	Gener 0 Extende Status De VFIG DATA 0x47) signed, 1D Array	ed Status escription	genStatIO extStatIO statDescripIO	• (• (• (

ETHERNET/IP NETWORK COMMUNICATIONS (CONTINUED)

OUTPUT DATA SETUP

	myDevice	TCP Connected	tcpConn	·	
Ethernet Port	CPU-ETH-Ext	▼ Adapter Name	adapNam	ne 🔹	
IP Address	10.11.0.236	Vendor ID	venID		.]
P Port Number	44818	TCP/IP Error	tcpIPErr	· · ·	-
FXP1	Close unused CIP Se	ession after 30 secs			
Enable	enableIO	▼ Connecti	on Online	connOnlineIO	•
		Gener	al Status	genStatIO	•
	in a start of the		101.1		
Enable Ro	Slot Number	0 Extende	ed Status	extStatio	▼
	Slot Number	0 Extende	ed Status escription	statDescripIO	•
T->0 (0 Extende	ed Status	statDescripIO	•
T->O (Originator 1 R	(INPUT) O->T (OUTPUT) To Target (OUTPUT) Data PI Time (msec)	0 Extends Status De CONFIG DATA 250	ed Status	statDescripIO	• •
T->O (Originator 1 R	(INPUT) O->T (OUTPUT) To Target (OUTPUT) Data PI Time (msec) onnection Point	0 Extends Status De CONFIG DATA 250 21 (0x15)	ed Status	statDescripIO	• •
T->O (Originator T R Cc	(INPUT) O->T (OUTPUT) To Target (OUTPUT) Data PI Time (msec) onnection Point Datatype: Integer,	0 Extends Status De CONFIG DATA 250 21 (0x15) 8 Bit Unsigned, 1D Array	escription	statDescripIO	• •
T->O (Originator 1 R Cc	(INPUT) O->T (OUTPUT) To Target (OUTPUT) Data PI Time (msec) Datatype: Integer, Data Array outArra	0 Extende Status De 250 21 (0x15) 8 Bit Unsigned, 1D Array yIO	escription	statDescripIO	•
T->O (Originator T R Co	(INPUT) O->T (OUTPUT) To Target (OUTPUT) Data PI Time (msec) Datatype: Integer, Data Array outArra e Size (bytes): 4	0 Extende Status De 250 21 (0x15) 8 Bit Unsigned, 1D Array yIO \checkmark	ed status	statDescripIO	•
T->O (Originator 1 R Cc Messag Numb	(INPUT) O->T (OUTPUT) To Target (OUTPUT) Data PI Time (msec) Datatype: Integer, Data Array outArra ie Size (bytes): 4 ber of Elements	0 Extende Status De CONFIG DATA 250 21 (0x15) 8 Bit Unsigned, 1D Array yIO \checkmark	escription	statDescripIO	•

CONFIG DATA SETUP (NO CONFIG DATA)

	myDevice	TCP Connected	tcpConn		
Ethernet Port	CPU-ETH-Ext	✓ Adapter Name	adapNan	ne 🔹 🗸 .	
IP Address	10.11.0.236	Vendor ID	venID	•	
P Port Number	44818	TCP/IP Error	tcpIPErr	•	
EXP I	Swap Byte Order				
Enable	enableIO	▼ Connec	tion On <mark>l</mark> ine	connOnlineIO	•
		Gen	eral Status	genStatIO	▼
🔲 Enable Ro	outing Slot Number	Gen 0 Exten	eral Status ded Status	genStatIO extStatIO	•
Enable Ro	outing Slot Number	Gen 0 Exten Status I	eral Status ded Status Description	genStatIO extStatIO statDescripIO	• •
Enable Ro T->O (Configurati	NPUT) O->T (OUTP	Gen 0 Exten Status I CONFIG DATA	eral Status ded Status Description	genStatIO extStatIO statDescripIO	• • •
T->O (Configurati	INPUT) O->T (OUTP on Data Configuration Data	Gen 0 Exten Status I CONFIG DATA	eral Status ded Status Description	genStatIO extStatIO statDescripIO	• • •
Enable Ro T->O (Configurati Enable	INPUT) O->T (OUTP on Data Configuration Data innection Point	Gen 0 Exten Status I CONFIG DATA 0 (0x0)	eral Status ded Status Description	genStatIO extStatIO statDescripIO	▼ ▼ ▼
Enable Ro T->O (Configurati Enable Co	Duting Slot Number INPUT) O->T (OUTP on Data Configuration Data innection Point Datatype:	Gen 0 Exten Status I CONFIG DATA 0 (0x0)	eral Status ded Status Description	genStatIO extStatIO statDescripIO	• •
Enable Ro T->O (Configurati Enable Co	INPUT) O->T (OUTP on Data Configuration Data nnnection Point Data type: Data Array	Gen 0 Exten Status I 0 (0x0) •	eral Status ded Status Description	genStatIO extStatIO statDescripIO	▼ ▼ ▼
Enable Ro T->O (Configurati Enable Co Messag	INPUT) O->T (OUTP on Data Configuration Data mnection Point Datatype: Data Array e Size (bytes): 0	Gen 0 Exten Status I T CONFIG DATA 0 (0x0)	eral Status ded Status Description	genStatIO extStatIO statDescripIO	 • • • • • •
Enable Ro T->O (Configurati Enable Co Messag Numb	INPUT) O->T (OUTP on Data Configuration Data mnection Point Datatype: Data Array e Size (bytes): 0 er of Elements	Gen 0 Exten Status I 0 (0x0) 0 (0x0) 0	eral Status ded Status Description	genStatIO extStatIO statDescripIO	

CONNECTING TO THE SR55-CM-ENET MODULE THROUGH EXPLICIT MESSAGE:

There are a few different objects that can be read or be written to via Explicit Messaging:

Obje	ects Suppo	rted By Ex	plicit Mess	aging	
Description	Read Only?	Service	Class	Instance	Attribute
Run	No	16 (0x10)	41 (0x29)	1	3
Fault Reset	No	16 (0x10)	41 (0x29)	1	12 (0x0c)
Network Control	No	16 (0x10)	41 (0x29)	1	5
Faulted	Yes	14 (0x0e)	41 (0x29)	1	10 (0x0a)
Warning	Yes	14 (0x0e)	41 (0x29)	1	11 (0x0b)
Running Forward	Yes	14 (0x0e)	41 (0x29)	1	7
Ready	Yes	14 (0x0e)	41 (0x29)	1	9
Control from Network	Yes	14 (0x0e)	41 (0x29)	1	15 (0x0f)
Drive State	Yes	14 (0x0e)	41 (0x29)	1	6

With the exception of the "Drive State" parameter, all of the other parameters either require a value of 1 or 0 for SET (16), and will return a value of 0 or 1 on the GET (14) parameters.

To run the starter, a value of 1 must be set in the "Network Control" parameter first and then a value of 1 can be sent to the "Run" parameter to start the motor and a value of 0 to the same parameter to stop the motor.

If the Communications Trip parameter is enabled, a message must be sent to the starter at a faster rate than what is configured for the Timeout parameter. It is typical in this situation to poll the "Faulted" parameter to view the state of the starter along with keeping the Communications Trip from enabling and stopping the motor.

If communications are interrupted for a long enough period to invoke the Communications Trip fault, the following sequence is required to restart the motor:

- Send a o to the "Run" parameter.
- Send a 1 to the "Fault Reset" parameter.
- Send a o to the "Fault Reset" parameter.
- Now you can restart the motor by sending a 1 to the "Run" parameter.

EXPLICIT MESSAGE INSTRUCTION EXAMPLES (FROM PRODUCTIVITY SERIES CPU)

EXAMPLE INSTRUCTION FOR SETTING THE STARTER TO NETWORK CONTROL (PRODUCTIVITY CPU)

licit Message	(EMSG)				Σ
		Vise Structure	emsg 1	•	
myDevice	·	In Progress	InProgress	•	•••
Unconnected	MSG 👻	Complete	Complete	•	
Generic	•	Success	Success	•	
16	(0x10)	Error	Error	*	
41	(0x29)	Timeout	Timeout	•	
5	(0x5)	Exception Response String	ExcResponse	-	
1	(0x1)				
(bytes): 0 Elements	7				
Л) utput statype: Integ ta Array Net@	ger, 8 Bit Unsig ControlByte	ned, 1D Array			
(bytes): 1					
	myDevice Unconnected I Generic 16 41 5 1) put atatype: ta Array (bytes): 0 Elements (bytes): 0 Elements JT) Jtput atatype: Integ ta Array Net	myDevice Unconnected MSG Generic 16 (0x10) 41 (0x29) 5 (0x5) 1 (0x1)) put statype: ta Array (bytes): 0 Elements 7 JT) utput statype: Integer, 8 Bit Unsig ta Array NetControlByte	myDevice In Progress Unconnected MSG Complete Generic Success 16 (0x10) Error 41 (0x29) Timeout 5 (0x5) Exception nt (0x1) Error 1 (0x1) Exception Exception Response String 1 (0x1) In Progress 1 (0x1) Exception Exception Exception Exception Exception In Progress In Progress In Progress <	myDevice In Progress Inconnected MSG Complete Generic Success 16 (0x10) 16 (0x29) 16 (0x29) 10x1) Exception 2 Exception 1 (0x1) put atatype:	Implexice In Progress In Progress In Progress In Progress In Progress In Progress In Progress In Progress In Progress <td< td=""></td<>

EXAMPLE INSTRUCTION FOR CONTROLLING THE START AND STOP OF THE MOTOR (PRODUCTIVITY CPU)

(value of 1 to start and 0 to stop)

			Vise Structure	emsg1	•	
Device Name	myDevice		In Progress	InProgress	•	
Connection	Unconnected	MSG 👻	Complete	Complete	Ŧ	
Service	Generic	•	Success	Success	•	
Service ID	16	(0x10)	Error	Error	¥	
Class ID	41	(0x29)	Timeout	Timeout	•	
Attribute ID	3	(0x3)	Exception Response String	ExcResponse	÷	
Instance ID	1	(0x1)	responde d'unig			0.0
T->O (INPUT	r)					
Da	atatype:		▼			
Message Size Number	: (bytes): 0 Elements	7				
Message Size Number O->T (OUTP	e (bytes): 0 Elements	7				
Message Size Number O->T (OUTP	: (bytes): 0 Elements UT) Jutput	7				
Message Size Number O->T (OUTP I Enable C	(bytes): 0 Elements UT) Jutput Jatatype: Inter	7 ger, 8 Bit Unsign	ned, 1D Array			
Message Size Number O->T (OUTP I Enable C Da	e (bytes): 0 Elements UT) Dutput Datatype: Inter Sata Array Run	7 ger, 8 Bit Unsig Control	ned, 1D Array			
Message Size Number O->T (OUTP I Enable C D Da Message Size	(bytes): 0 Elements UT) Dutput Datatype: Inter sta Array Run : (bytes): 1	7 ger, 8 Bit Unsign Control	ned, 1D Array			
Message Size Number O->T (OUTP I Enable C D Da Message Size Number	e (bytes): 0 Elements UUT) Jutput Jatatype: Inter ata Array Run e (bytes): 1 Elements	7 ger, 8 Bit Unsig Control	ned, 1D Array			
Message Size Number O->T (OUTP I Enable C D D Message Size Number	(bytes): 0 Elements UT) Jutput Jatatype: Inter ata Array Run (bytes): 1 Elements	7 ger, 8 Bit Unsig Control	ned, 1D Array			

EXAMPLE INSTRUCTION FOR READING BACK THE FAULT STATUS OF THE STARTER (PRODUCTIVITY CPU)

			Vise Structure	emsg1	•
Device Name	myDevice		In Progress	InProgress	• .
Connection	Unconnected MSG		Complete	Complete	• .
Service	Generic		Success	Success	•
Service ID	14 (0xE)	Error	Error	☞ .
Class ID	41 (0x29)	Timeout	Timeout	•
Attribute ID	10 (0xA)	Exception Response String	ExcResponse	·
Instance ID	1 (0x1)	Response buing		
D	atatype: Intege	er, 8 Bit Unsig	ned, 1D Array		
D	atatype: Intege	er, 8 Bit Unsig	ned, 1D Array		
D Da Message Size	atatype: Intege ata Array Faults (bytes): 1	er, 8 Bit Unsig Status	vined, 1D Array		
D Da Message Size Number	atatype: Intege ata Array Faults (bytes): 1 Elements	er, 8 Bit Unsig Status 1	ned, 1D Array		
D Da Message Size Number O->T (OUTP	atatype: Intege ata Array FaultS (bytes): 1 Elements	er, 8 Bit Unsig Status 1	ned, 1D Array		
D Da Message Size Number O->T (OUTP Enable O	atatype: Intege ata Array FaultS (bytes): 1 Elements UT) utput	er, 8 Bit Unsig Status 1	ned, 1D Array		
D Da Message Size Number O->T (OUTP Enable O D	atatype: Intege ata Array FaultS (bytes): 1 Elements UT) utput atatype:	rr, 8 Bit Unsig Status 1	ned, 1D Array		
D Da Message Size Number O->T (OUTP Enable O D Da	atatype: Intege ata Array FaultS (bytes): 1 Elements UT) utput atatype: ata Array	er, 8 Bit Unsig Status 1	med, 1D Array ▼		
D Da Number O->T (OUTP Enable O D Da Message Size	atatype: Intege ata Array FaultS (bytes): 1 Elements UT) utput atatype: ata Array (bytes): 0	r, 8 Bit Unsig Status 1	ned, 1D Array ▼		
D Da Number O->T (OUTP Enable O D Da Message Size Number	atatype: Intege ata Array FaultS (bytes): 1 Elements utput atatype: ata Array (bytes): 0 Elements	r, 8 Bit Unsig Status 1	ned, 1D Array ▼		

EXAMPLE INSTRUCTION TO RESET ANY FAULTS ON THE STARTER (PRODUCTIVITY CPU)

			Vise Structure	emsg1	•
Device Name	myDevice 👻		In Progress	InProgress	•
Connection	Unconnected MSG 🛛 👻		Complete	Complete	•
Service	Generic 👻		Success	Success	-
Service ID	16	(0x10)	Error	Error	*
Class ID	41	(0x29)	Timeout	Timeout	•
Attribute ID	10				
	12	(0xC)	Exception	ExcResponse	+
Instance ID T->O (INPUT Enable Ir D	12 1 nput atatype:	(0xC) (0x1)	Exception Response String	ExcResponse	*
Instance ID T->O (INPUT Enable Ir D Da Message Size Number	12 1 1 1 1 1 1 1 1 1 1 1 1 1	(0xC) (0x1) 7	Exception Response String	ExcResponse	*
Instance ID T->O (INPUT D Enable Ir D Da Message Size Number	12 1 1 1 1 1 1 1 1 1 1 1 1 1	(0xC) (0x1)	Exception Response String	ExcResponse	*
Instance ID T->O (INPUT Enable Ir D Da Message Size Number O->T (OUTP Co->T (OUTP	12 1 1 1 1 1 1 1 1 1 1 1 1 1	(0xC) (0x1)	Exception Response String	ExcResponse	* [
Instance ID T->O (INPUT D Enable Ir D D D Message Size Number O->T (OUTP D Enable O D	12 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	(0xC) (0x1) 7 ger, 8 Bit Unsig	Exception Response String	ExcResponse	*
Instance ID T->O (INPUT D Da Message Size Number O->T (OUTP Co->T (OUTP Da Da Da Da	12 1 1 1 1 1 1 1 1 1 1 1 1 1	(0xC) (0x1) 7 ger, 8 Bit Unsig ItReset	Exception Response String	ExcResponse	*
Instance ID T->O (INPUT Enable Ir D D D Message Size Number 1 O->T (OUTP Enable O D D Z Message Size	12 1 1 1 1 1 1 1 1 1 1 1 1 1	(0xC) (0x1) 7 ger, 8 Bit Unsig	Exception Response String	ExcResponse	*

Drive State					
Byte Value	State Description				
1	Startup				
2	Ready & Stopped				
4	Running				
5	Stopping				
6	Fault Stop				
7	Faulted				

