SCADAPack E ISaGRAF 3 Modbus Communication Interfaces

Schneider Electric

Documentation

Table of Contents

I ISaGRAF 3 Modbus Communication Interfaces

Version: 8.05.4

All pertinent state, regional, and local safety regulations must be observed when installing and using this product. For reasons of safety and to help ensure compliance with documented system data, only the manufacturer should perform repairs to components.

When devices are used for applications with technical safety requirements, the relevant instructions must be followed. Failure to use Schneider Electric software or approved software with our hardware products may result in injury, harm, or improper operating results.

Failure to observe this information can result in injury or equipment damage.

1 Technical Support

Support related to any part of this documentation can be directed to one of the following support centers.

Technical Support: The Americas

Available Monday to Friday 8:00am – 6:30pm Eastern Time

Toll free within North America	1-888-226-6876
Direct Worldwide	+1-613-591-1943
Email	TechnicalSupport@controlmicrosystems.com

Technical Support: Europe

Available Monday to Friday 8:30am – 5:30pm Central European Time

Direct Worldwide	+31 (71) 597-1655
Email	euro-support@controlmicrosystems.com

Technical Support: Asia

Available Monday to Friday 8:00am – 6:30pm Eastern Time (North America)

Direct Worldwide	+1-613-591-1943
Email	TechnicalSupport@controlmicrosystems.com

Technical Support: Australia

Inside Australia	1300 369 233
Email	au.help@schneider-electric.com

2 Safety Information

Read these instructions carefully, and look at the equipment to become familiar with the device before trying to install, operate, or maintain it. The following special messages may appear throughout this documentation or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.

⚡	The addition of this symbol to a Danger or Warning safety label indicates that an electrical hazard exists, which will result in personal injury if the instructions are not followed.

⚠	This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

⚠DANGER

DANGER indicates an imminently hazardous situation which, if not avoided, **will result in** death or serious injury.

⚠WARNING

WARNING indicates a potentially hazardous situation which, if not avoided, **can result** in death or serious injury.

⚠CAUTION

CAUTION indicates a potentially hazardous situation which, if not avoided, **can result** in minor or moderate injury.

CAUTION

CAUTION used without the safety alert symbol, indicates a potentially hazardous situation which, if not avoided, **can result in** equipment damage..

PLEASE NOTE

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

A qualified person is one who has skills and knowledge related to the construction and operation of electrical equipment and the installation, and has received safety training to recognize and avoid the hazards involved.

BEFORE YOU BEGIN

Do not use this product on machinery lacking effective point-of-operation guarding. Lack of effective point-of-operation guarding on a machine can result in serious injury to the operator of that machine.

⚠CAUTION

EQUIPMENT OPERATION HAZARD

- Verify that all installation and set up procedures have been completed.
- Before operational tests are performed, remove all blocks or other temporary holding means used for shipment from all component devices.

• Remove tools, meters, and debris from equipment.
Failure to follow these instructions can result in injury or equipment damage.

Follow all start-up tests recommended in the equipment documentation. Store all equipment documentation for future references.

Software testing must be done in both simulated and real environments.

Verify that the completed system is free from all short circuits and grounds, except those grounds installed according to local regulations (according to the National Electrical Code in the U.S.A, for instance). If high-potential voltage testing is necessary, follow recommendations in equipment documentation to prevent accidental equipment damage.

Before energizing equipment:

- Remove tools, meters, and debris from equipment.
- Close the equipment enclosure door.
- Remove ground from incoming power lines.
- Perform all start-up tests recommended by the manufacturer.

OPERATION AND ADJUSTMENTS

The following precautions are from the NEMA Standards Publication ICS 7.1-1995 (English version prevails):

- Regardless of the care exercised in the design and manufacture of equipment or in the selection and ratings of components, there are hazards that can be encountered if such equipment is improperly operated.
- It is sometimes possible to misadjust the equipment and thus produce unsatisfactory or unsafe operation. Always use the manufacturer’s instructions as a guide for functional adjustments. Personnel who have access to these adjustments should be familiar with the equipment manufacturer’s instructions and the machinery used with the electrical equipment.
- Only those operational adjustments actually required by the operator should be accessible to the operator. Access to other controls should be restricted to prevent unauthorized changes in operating characteristics.

3 Preface

Purpose

The purpose of this document is to describe the SCADAPack E ISaGRAF 3 RTU drivers for Modbus and Open Modbus/TCP and Modbus RTU in TCP protocols..

ISaGRAF is used to interface the RTU with PLC and peripheral devices. The RTU is a Modbus Master /

Modbus Client when used this way. This document details using the driver and communicating with PLC and peripheral devices.

The SCADAPack E also allows the RTU to be a Modbus Slave / Modbus/TCP Server (interfaces using serial Modbus and Open Modbus/TCP communications).

Assumed Knowledge

Familiarity with the Modbus protocol and the ISaGRAF Workbench is recommended.

Target Audience

- Systems Engineers
- Commissioning Engineers
- Maintenance Technicians

References

- *SCADAPack E ISaGRAF Technical Reference* manual
- ICS Triplex ISaGRAF 3 User Manual
- Protocol documentation for various Modbus PLC devices
- Open Modbus/TCP Specification Revision 1.0, March 1999

4 Overview

Master/Slave and Client/Server Terminology

PLC and peripheral devices may communicate with the SCADAPack E RTU using ISaGRAF **Slave** I/O boards.

PLC or peripheral device elements are read and the return values cached in the RTU for access through an ISaGRAF input board. Similarly, ISaGRAF output board data can be transferred to the PLC or peripheral device.

The SCADAPack E RTU's interface with ISaGRAF is detailed in the *SCADAPack E ISaGRAF Technical Reference* manual.

The status (quality) of the data read from the PLC or peripheral device is present in RTU system points that can be accessed from within ISaGRAF or external to the RTU.

When using ISaGRAF Modbus PLC I/O boards for communication with a Modbus peripheral device, or devices, the SCADAPack E RTU is a **Modbus Master**. The peripheral device(s) need to be Modbus Slave(s).

When using ISaGRAF Modbus/TCP I/O boards for communication with Modbus/TCP peripheral devices, the SCADAPack E RTU is an **Open Modbus/TCP Client**. The peripheral device(s) need to be Open Modbus/TCP **Server**(s) (e.g. Ethernet PLC). Open Modbus/TCP protocol has also previously been known as MBAP protocol. It is referred to as Open Modbus/TCP protocol throughout this manual.

When using ISaGRAF Modbus RTU in TCP I/O boards for communication with Modbus RTU in TCP peripheral devices, the SCADAPack E RTU is a **Modbus Master**. The peripheral device(s) must be Modbus Slave(s).

5 Modbus communication to PLC Devices

The following sections detail SCADAPack E RTU communication to PLC devices (where the RTU is a Modbus Master and Modbus/TCP Client).

Details on specific ISaGRAF Modbus I/O boards used for defining the Modbus communication parameters can be found in the *SCADAPack E ISaGRAF I/O Connection Reference* manual.

A maximum of 100 PLC Device I/O Boards Types (total of each PLC type) may be configured in total across both RTU ISaGRAF Applications.

5.1 I/O Board Types & Modbus Addressing Terminology

Modbus Addressing Terminology

The SCADAPack E RTU uses 5-digit Modbus address numbering, where the leading digit generally represents the register data type. In addition, the numbering within each register data type adheres to the classical Modicon PLC numbering convention, commencing at register 1. (the Modbus protocol description implies the register data type from the protocol function code, and uses 4-digit hexadecimal addressing commencing at register 0).

For example:

A register read by the SCADAPack E RTU specifying Modbus register 40010 is represented by Modbus protocol function code 3, protocol register address 0x0009.

See Sections ***Modbus Registers*** 18, ***Modbus/TCP Registers*** 25 and ***Modbus RTU in TCP Registers*** 34 for details on register address ranges and Modbus function codes.

Some Modbus systems use 6-digit addressing, as opposed to the 5-digit Modbus register addressing described above. 6-digit addressing is designed to enable access to additional registers in each register range. A 6 digit address is made up of a single digit numeric prefix and a 5-digit Modbus register number.

For example:

Registers 300001 – 309999 are equivalent to 5-digit Modbus register addresses 30001 – 39999.

However, input registers 310000 – 365536 in a remote PLC device are not addressable with the SCADAPack E RTU's 5-digit register address.

In the 5-digit addressing regime used by the SCADAPack E RTU, *HOLDING REGISTERS* are extended beyond register 49999. RTU Modbus register addresses 50000 – 65535 can be addressed in a remote PLC device, and are the equivalent to 6-digit holding register numbers 450000 – 465535. So the SCADAPack E RTU can access remote PLC device holding registers equivalent to the (6-digit) range 400001-409999 & 450000-465535.

I/O Board Types

Where a SCADAPack E RTU has one or more of its serial ports configured as a '*PLC Device*' and ***mbus..*** or ***mod..*** I/O boards are used within the ISaGRAF application, the RTU communicates using serial "Modbus RTU" protocol. Many of the I/O boards supplied for the SCADAPack E are ***mbus..*** boards.

The RTU does not support "Modbus ASCII" protocol.

The RTU communication port data rate and parity format are used by its Modbus PLC device driver. RS232, RS422 and RS485 communications are supported.

mod.. ISaGRAF I/O boards can not be used when multiple communication ports are configured for PLC peripheral device communications due to the requirement to specify which of these ports connects to the device. Use only ***mbus..*** and/or ***mtcp..*** I/O boards in this situation.

When the *Modbus/TCP(client)* IP service is enabled on the SCADAPack E RTU and ***mtcp..*** I/O boards are used in an ISaGRAF application, the RTU communicates using Open Modbus/TCP communication protocol. The protocol connects TCP socket(s) between the SCADAPack E RTU (Client) and the peripheral device(s) (Servers). TCP/IP over Ethernet and PPP communications from the RTU are supported.

The SCADAPack E RTU supports simultaneous communication using serial Modbus and Open Modbus/ TCP protocols.

i.e. ***mbus..*** I/O boards can communicate with Modbus peripherals on one or more RTU serial ports, and at the same time, ***mtcp..*** I/O boards can communicate with Modbus/TCP peripherals on an SCADAPack E RTU TCP/IP interface (e.g. Ethernet) and ***mrtp..*** I/O boards can communicate with Modbus RTU in TCP peripherals at the same time.

5.2 Serial Modbus I/O Board Interfaces

The ***mbus..*** ISaGRAF boards use a SCADAPack E RTU serial port configured as a '*PLC Device*' to communicate with Modbus peripheral devices (herein described as PLCs).

Details on specific ISaGRAF Modbus I/O boards used for defining the Modbus communication parameters can be found in the *SCADAPack E ISaGRAF I/O Connection Reference* manual.

A maximum of 100 PLC Device I/O Boards Types (total of each PLC type) may be configured in total across both RTU ISaGRAF Applications.

- ***Multiple I/O Boards*** 13
- ***Modbus Input Boards*** 14
- ***Modbus Output Boards*** 16
- ***Modbus Registers and I/O Boards*** 18

5.2.1 Multiple I/O Boards

Multiple I/O boards may be configured within the same ISaGRAF application.

Each I/O board can access different PLC register data within the same PLC device, or in different PLCs. E.g. Multi-drop RS485 permits uniquely addressed Modbus PLCs to be connected to a SCADAPack E RTU serial port.

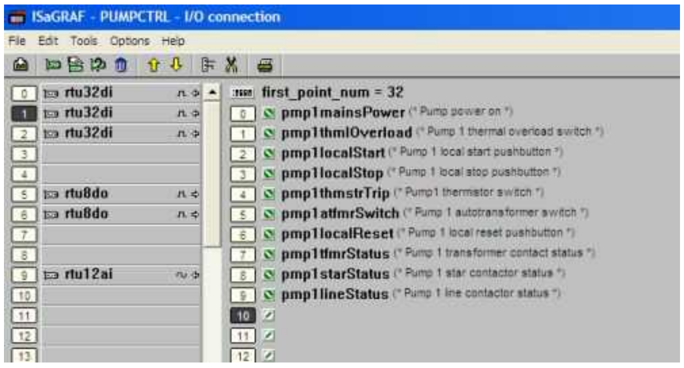
In addition, multiple I/O boards may be configured to use different RTU serial ports configured as a “PLC Device”.

Each of the ISaGRAF application’s PLC I/O boards uses a separate Modbus request to read or write its data. Improved Modbus communication efficiency can be achieved by grouping Modbus registers together and using less I/O boards with a larger number of channels (e.g. ***mbus64ai***), rather than more I/O boards with a smaller number of channels.

A maximum of 100 Slave I/O Boards may be configured in total for communication ports and across both ISaGRAF applications running on the separate kernels. Recall, also that each ISaGRAF application has a total limit of 255 I/O boards for every board type.

Communication status is available on the first 60 I/O boards for ISaGRAF kernel 1, and 14 I/O boards for ISaGRAF kernel 2. See Section ***System Points*** 85 for more information.

ISaGRAF “Complex Equipment” types are comprised of configurations similar to I/O boards. Where a Complex Equipment type includes PLC Device I/O board configurations, each such I/O board configuration within the Complex Equipment type counts towards the limit of 100 slave I/O boards on that communications channel. A corresponding pair of system points relates to each PLC Slave I/O board on the lowest RTU port number, as described in Section ***System Points*** 85.


ISaGRAF - PUMPCTRL - I/O connection
File Edit Tools Options Help

```
0  rtu32di        first_point_num = 32
1  rtu32di        0  pmp1mainsPower (* Pump power on *)
2  rtu32di        1  pmp1thmlOverload (* Pump 1 thermal overload switch *)
3                 2  pmp1localStart (* Pump 1 local start pushbutton *)
4                 3  pmp1localStop (* Pump 1 local stop pushbutton *)
5  rtu8do         4  pmp1thmstrTrip (* Pump1 thermistor switch *)
6  rtu8do         5  pmp1atfmrSwitch (* Pump 1 autotransformer switch *)
7                 6  pmp1localReset (* Pump 1 local reset pushbutton *)
8                 7  pmp1tfmrStatus (* Pump 1 transformer contact status *)
9  rtu12ai        8  pmp1starStatus (* Pump 1 star contactor status *)
10                9  pmp1lineStatus (* Pump 1 line contactor status *)
11                10
12                11
13                12
```

Figure 5.1: ISaGRAF Project Multiple I/O boards

PLC device communications using these I/O boards can be controlled by an ISaGRAF function block: **mbusctrl**.

5.2.2 Modbus Input Boards

Modbus PLC Input Board variables are updated at the start of the ISaGRAF application scan. The value presented to the ISaGRAF variables is the value returned by the PLC to the previous read request. This read may have occurred during previous ISaGRAF application scans. The "data update rate" parameter on the I/O board sets the "scan" rate of the PLC data. The PLC communication status is updated if there is an error returned from the PLC, or no response from the PLC after a data request by the RTU (see Section ***Modbus Status Values*** 62). The status is cleared by the SCADAPack E RTU upon successful communications. To catch transient errors, you can use ISaGRAF code to store non-zero values.

Input Board Parameters

first_register: specifies the Modbus PLC Device data registers to access when reading from PLC data into ISaGRAF variables. The PLC data type accessed is specific to the PLC Device I/O board type and board address (see ***Table 5.1*** 18).

plc_data_type: specifies the Modbus PLC data register type. Various PLC data types are supported for Boolean and Analog boards. See Section ***Data Conversion*** 37 for more information.

data_update_rate: The unit for this parameter is the millisecond (ms), and specifies the rate at which the data for the input board is extracted from the PLC. Individual I/O boards may have different data update rates allowing prioritization of data extracted from a PLC Device. The SCADAPack E RTU may not be able to read requested PLC data within the time set by the data update rate depending on the quantity of data to be read, rate of write requests and PLC communication speed. In this case the update rates will be slower.

plc_device_addr: This parameter specifies the PLC device address. Modbus PLC devices on the same communication channel (e.g. multi-dropped or bridged) need to have unique device addressing in order to be identified. ISaGRAF may access data from multiple PLCs via the same communication interface. In this case a separate I/O board will be required for each PLC device. Values for this parameter are usually in the range 1-254.

timeout: The Modbus PLC device driver provides a parameter for specifying the communications timeout on an individual I/O board (i.e. the timeout applies to communications associated with that board). Where this value is "0", the PLC device driver will use the default timeout (1200ms). Units for this field are the millisecond (ms).

port: this parameter defines which of multiple SCADAPack E RTU ports configured as a "PLC Device" will be used to communicate with the PLC or peripheral device. If only one "PLC Device" port is configured, this field is ignored. ISaGRAF PLC Device I/O boards not including this parameter can only be used when a single "PLC Device" port is configured on the SCADAPack E RTU.

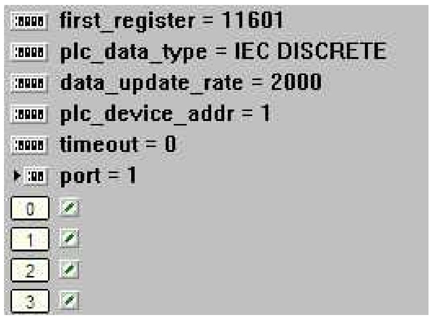

first_register = 11601
plc_data_type = IEC DISCRETE
data_update_rate = 2000
plc_device_addr = 1
timeout = 0
port = 1

Figure 5.2: Modbus Input Board Connection Setup

The figure above illustrates the board parameters for a ***mbus4di*** I/O connection. PLC #1 discrete input registers are read every 2 seconds into the ISaGRAF boolean variables on I/O channels 0 through 3. Discrete input registers 11601 through 11604 are read. A *default PLC timeout* of 1200ms is applied as the “timeout” value is 0.

Controlling PLC Device Communications

PLC device communications using these I/O boards can be controlled by an ISaGRAF function block: **mbusctrl**. The En_RD parameter on the block affects PLC device Input Boards. For more information see *SCADAPack E Function Block Reference* manual for more information.

“OPERATE” on Input Boards

The ISaGRAF “OPERATE” function may be used on a Modbus PLC Input Board variable provided that the PLC Modbus register read by the input board can also be written, i.e. COILS or HOLDING REGISTERS. This permits PLC registers to be inputs into ISaGRAF, but have them “Preset” or initialized in the PLC by the ISaGRAF application.

For more information see *SCADAPack E ISaGRAF Technical Reference* manual.

PLC device communications resulting from the use of the OPERATE functin can be controlled by an ISaGRAF function block: **mbusctrl**. The En_WR parameter on the block affects control operations being sent to the PLC device. For more information see *SCADAPack E Function Block Reference* manual for more information.

5.2.3 Modbus Output Boards

ISaGRAF output board variables are updated at the end of the ISaGRAF application scan. ISaGRAF output variables are sent to the PLC when an ISaGRAF application changes the value of a variable attached to the Modbus PLC Output Board. They are sent to the PLC after this occurs, but the ISaGRAF scan continues executing while the PLC communications are in progress. In order words, communications to the PLC is occurs asynchronously to the program scan.

In addition, output board data is updated to the PLC under the following conditions:

- When the ISaGRAF application starts, output board data is written
- If the PLC does not respond to a control, it is re-sent until it is responded
- Output board data is written at a background “must write rate”.

The device to which Modbus Output commands are sent need to provide Modbus register addresses for each of the channels on the ISaGRAF output board, regardless of whether ISaGRAF variables are attached to the channels, or not.

E.g. for the mbus16do board, 16 contiguous Modbus Coil registers need to be present in the remote device and support external writing.

Output Board Parameters

first_register*:* specifies the PLC Device data registers to access when writing from ISaGRAF variables to PLC data. The PLC data type accessed is specific to the PLC Device I/O board and board address. See ***Table 5.1*** 18.

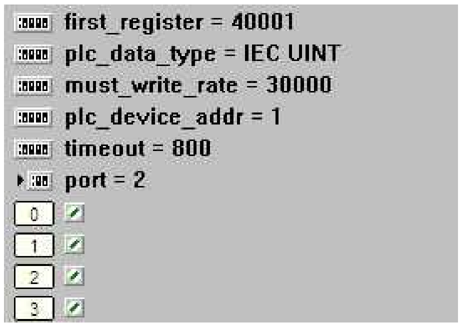
plc_data_type*:* specifies the Modbus PLC data register type. Various PLC data types are supported for Boolean and Analog boards. See Section ***Data Conversion*** 37 for more information.

plc_device_addr*:* This parameter specifies the PLC device address. Modbus PLC devices on the same communication channel (e.g. multi-dropped or bridged) need to have unique device addressing in order to be identified. ISaGRAF may access data from multiple PLCs via the same communication interface. In this case a separate I/O board will be required for each PLC device. Values for this parameter are usually in the range 1-254.

must_write_rate*:* The unit for this parameter is the millisecond (ms), and specifies the rate at which the data for the output board is written to the PLC. Between “*must_write_rate*” periods, data is written to the PLC only when the ISaGRAF output variable values change. Individual I/O boards may have different must write rates allowing prioritization of data sent to a PLC Device. Setting this parameter to 0 disables the time-based writing of output data. Data is written at ISaGRAF application startup (each channels) and thereafter only when individual ISaGRAF output channel variables change. See Section ***PLC Output Board Default Background Update Rate*** 64.

timeout*:* The Modbus PLC device driver provides a parameter for specifying the communications timeout on an individual I/O board (i.e. the timeout applies to communications associated with that board). Where this value is “0”, the PLC device driver will use the default timeout (1200ms). Units for this field are in milliseconds.

port: this parameter defines which of multiple SCADAPack E RTU serial ports configured as a “PLC Device” will be used to communicate with the PLC or peripheral device. If only one “PLC Device” port is configured, this field is ignored. ISaGRAF PLC Device I/O boards not including this parameter can only be used when a single “PLC Device” port is configured on the SCADAPack E RTU.



first_register = 40001
plc_data_type = IEC UINT
must_write_rate = 30000
plc_device_addr = 1
timeout = 800
port = 2

Figure 5.3: Modbus Output Board Connection Setup

In the above ***mbus4ao*** example, PLC #1 holding registers 40001 through 4004 are set from variables on I/O channels 0 through 3. Holding registers are written in IEC UINT (16-bit unsigned integer format) when the ISaGRAF variable on the I/O channel changes, and at a rate of every 30 seconds even if they don't change. The PLC has 800ms to respond to a register write command for up to 4 registers.

PLC device communications using these I/O boards can be controlled by an ISaGRAF function block: **mbusctrl**. The En_WR parameter on the block affects PLC device Output Boards. For more information see *SCADAPack E Function Block Reference Manual* for more information.

5.2.4 Modbus Registers and I/O Boards

The following table describes the relationship between ISaGRAF Modbus I/O board types, the Modbus address ranges and functions used.

Details on specific ISaGRAF Modbus I/O boards used for defining the Modbus communication parameters can be found in the *SCADAPack E ISaGRAF I/O Connection Reference* manual.

Table 5.1: ISaGRAF Modbus PLC Device I/O Board Register Access

ISaGRAF Modbus Slave I/O Board Type	ISaGRAF Modbus Board Address	Type of PLC Register	Uses Modbus Function Code
mbusnnDI	1- 9999	Reads discrete COILS	1
OPERATE on **mbusnnDI** variable	1- 9999	Presets discrete COILS	5
mbusnnDI	10001-19999	Reads discrete INPUTS	2
mbusnnDI	40001-65535	Reads HOLDING Registers as bits	3
OPERATE on **mbusnnDI** variable	40001-65535	Presets HOLDING Register	16
mbusnnDO	1- 9999	Writes discrete COILS	5
mbusnnDO	40001-65535	Writes bits to HOLDING Registers	16
mbusnnAI	30001-39999	Reads INPUT Registers	4
mbusnnAI	40001-65535	Reads HOLDING Registers	3
OPERATE on **mbusnnAI** variable	40001-65535	Presets HOLDING Register	16
modnnAO	40001-65535	Writes single HOLDING Register	6
mbusnnAO	40001-65535	Writes HOLDING Registers	16

Board Types

The following Modbus PLC I/O board types are available:

Board Name	ref+	ISaGRAF Data Type	PLC Data Types Supported
mod4ao	000A	4 Analog Outputs*[1]	IEC UINT to Holding Regs only (single reg write using func code 6)

mod8ao	000A	8 Analog Outputs*1	
mbus16di	000E	16 Boolean Inputs	IEC DISCRETE, 984 DISCRETE from Input Status, Coil Registers, Holding Registers
mbus32di	000E	32 Boolean Inputs	
mbus16do	000F	16 Boolean Outputs	IEC DISCRETE, 984 DISCRETE to Coil Registers, Holding Registers
mbus32do	000F	32 Boolean Outputs	
mbus1ai	0010	1 Analog Input*1	IEC UINT, IEC INT, IEC DINT, IEC REAL, SWAP REAL from Input Registers, Holding Registers
mbus4ai	0010	4 Analog Inputs*1	
mbus8ai	0010	8 Analog Inputs*1	
mbus16ai	0010	16 Analog Inputs*1	
mbus32ai	0010	32 Analog Inputs*1	
mbus64ai	0010	64 Analog Inputs*1	
mbus1ao	0011	1 Analog Output*1	IEC UINT, IEC INT, IEC DINT IEC REAL, SWAP REAL to Holding Registers
mbus4ao	0011	4 Analog Outputs*1	
mbus8ao	0011	8 Analog Outputs *1	
mbus16ao	0011	16 Analog Outputs *1	
mbus32ao	0011	32 Analog Outputs *1	
mbus64ao	0011	64 Analog Outputs *1	
mbus64ao	0011	64 Analog Outputs *1	

+ ref value is in Hexadecimal and is an internal ISaGRAF I/O board field.

*1 Analog input and output board conversion may be used.

The ISaGRAF "Operate" function can also be used to preset input variables on the **mbus*xx*di** and **mbus *xx*ai** I/O boards. For more information see the *SCADAPack E ISaGRAF Technical Reference* manual.

❖ **Information for Advanced ISaGRAF users:**

Other I/O Boards, I/O configurations or Complex Equipment types based on the reference numbers shown in the above table are possible. The following guidelines need to be observed when configuring new I/O interface types:

- There is an upper limit of 32 I/O channels per PLC digital board for the various Modbus board types.

- There is an upper limit of 64 I/O channels per PLC analog board for the various Modbus board types.
- A plc_data_type user parameter is defined for PLC Device I/O boards. The value of this parameter field is a string field describes the data type of data being accessed (See Section ***Data Conversion*** 37 for more information.) – e.g. “IEC UINT”.
- An additional plc_dev_type hidden parameter string field describes the plc type, communication channel type and special controls. The value of this field is driver specific. E.g. “ms” indicates advanced Modbus board (m), serial comms interface (s)

5.3 Modbus TCP I/O Board Interface

The ***mtcp..*** ISaGRAF boards are to be used by the SCADAPack E RTU to communicate via Ethernet or PPP serial interfaces with the Open Modbus/TCP protocol peripheral devices (herein described as PLCs).

Each of the ISaGRAF application's PLC I/O boards use a separate Modbus/TCP request to read or write its data. Improved Modbus communication efficiency can be achieved by grouping Modbus registers together and using less I/O boards with a larger number of channels (e.g. mtcp64ai), rather than more I/O boards with a smaller number of channels.

A maximum of 100 PLC Device I/O Boards Types (total of each PLC type) may be configured in total across both RTU ISaGRAF Applications.

ISaGRAF "Complex Equipment" types are comprised of configurations similar to I/O boards. Where a Complex Equipment type includes PLC Device I/O board configurations, each such I/O board configuration within the Complex Equipment type counts towards the limit of 100 Slave I/O boards.

Details on specific ISaGRAF Modbus I/O boards used for defining the Modbus communication parameters can be found in the *SCADAPack E ISaGRAF I/O Connection Reference* manual.

A corresponding pair of system points relates to each PLC Slave I/O board as described in Section ***System Points*** 85.

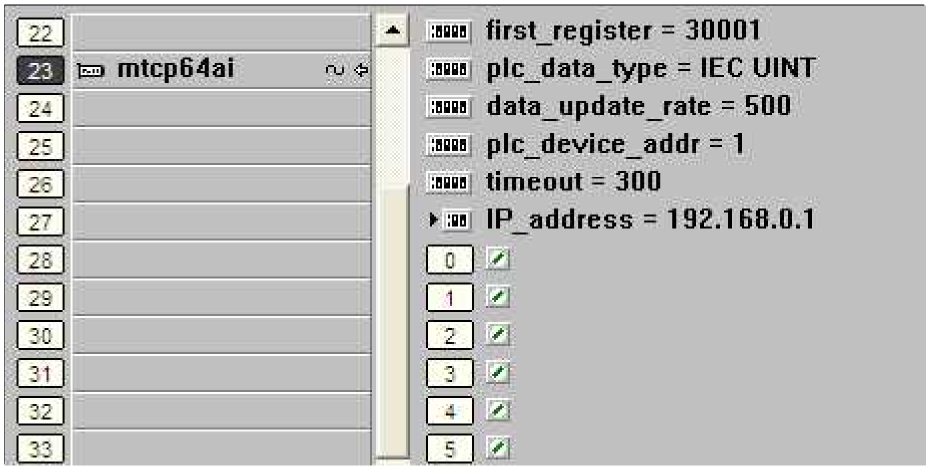
Modbus/TCP boards utilize default IEC data types. Where applicable, the data type may be available for the user to choose.

PLC device communications using these I/O boards can be controlled by an ISaGRAF function block: **mtcpctrl**.

- ***Modbus/TCP Input Boards*** 22
- ***Modbus/TCP Output Boards*** 24
- ***Modbus/TCP Registers*** 25
- ***Modbus/TCP Board Types*** 26
- ***Open Modbus/TCP Conformance Classes*** 28

5.3.1 Modbus/TCP Input Boards

Modbus/TCP PLC Input Board variables are updated at the start of an ISaGRAF application scan. The value presented to the ISaGRAF variables is the value returned by the PLC from the previous read request. This read may have occurred during previous ISaGRAF application scans. The "data update rate" parameter on the I/O board sets the "scan" rate of the PLC data. The PLC communication status is updated if there is an error returned from the PLC, or no response from the PLC after a data request by the RTU. (See Section ***Modbus Status Values*** 62). The status is cleared by the SCADAPack E RTU upon successful communications. To catch transient errors you can use ISaGRAF code to store non-zero values.



first_register = 30001
plc_data_type = IEC UINT
data_update_rate = 500
plc_device_addr = 1
timeout = 300
IP_address = 192.168.0.1

Figure 6.1: ISaGRAF Modbus/TCP board

Input Board Parameters

first_register*:* specifies the Modbus/TCP PLC data registers to access when reading from PLC data into ISaGRAF variables. The PLC data type accessed is the same as Modbus PLC Device I/O boards detailed in Section ***Modbus Registers*** 18.

plc_data_type*:* specifies the Modbus/TCP PLC data register type. Various data types are supported. See Section ***Data Conversion*** 37 for more information.

data_update_rate*:* The unit for this parameter is the millisecond (ms), and specifies the rate at which the data for the Input board is extracted from the PLC. Individual I/O boards may have different data update rates allowing prioritization of data extracted from a PLC Device. The SCADAPack E RTU may not be able to read requested PLC data within the time set by the data update rate depending on the quantity of data to be read, rate of write requests and PLC communication speed. In this case the update rates will be slower.

plc_device_address*:* This parameter specifies the PLC device (unit) address. Modbus PLC devices accessed at the same IP address (e.g. via a Modbus bridge) need to have a unique unit address in order to be identified. ISaGRAF may access data from different units on the same IP address or at different IP addresses. In these cases a separate I/O board will be required for each device.

timeout: The Modbus/TCP PLC device driver provides a parameter for specifying the communications timeout on an individual I/O board (i.e. the timeout applies to communications associated with that board). Where this value is "0", the PLC device driver will use the default timeout (1200ms). Units for this field are in ms.

***IP_address**:* This parameter specifies the IP network address that the SCADAPack E RTU connects to for communication with the PLC for this I/O board. Enter the IP address of the Modbus/TCP PLC, or Modbus bridge if applicable.

Controlling PLC Device Communications

PLC device communications using these I/O boards can be controlled by an ISaGRAF function block: **mtcpctrl**. The En_RD parameter on the block affects PLC device Input Boards. For more information see *SCADAPack E Function Block Reference* manual for more information.

“OPERATE” on Input Boards

The ISaGRAF “OPERATE” function may be used on Modbus/TCP Input Boards where the register read by the input board is also writeable e.g. coils or holding registers. This permits registers to be inputs into ISaGRAF but have them “Preset” or initialized in the PLC by ISaGRAF.

For more information see the *SCADAPack E ISaGRAF Technical Reference* manual.

PLC device communications resulting from the use of the OPERATE functin can be controlled by an ISaGRAF function block: **mtcpctrl**. The En_WR parameter on the block affects control operations being sent to the PLC device. For more information see *SCADAPack E Function Block Reference* manual for more information.

5.3.2 Modbus/TCP Output Boards

Modbus/TCP PLC Output Board data is written to the PLC when an ISaGRAF application changes the value of a variable attached to the Output board. In addition, output board data is written to the PLC under the following conditions:

- When the ISaGRAF application starts, output board data is written
- If the PLC does not respond to a control, it is re-sent until it is responded
- Output board data is rewritten at a background update rate

The device to which Modbus Output commands are sent need to provide Modbus register addresses for each of the channels on the ISaGRAF output board, regardless of whether ISaGRAF variables are attached to the channels, or not.

E.g. for the mbus16do board, 16 contiguous Modbus Coil registers need to be present in the remote device and support external writing.

Output Board Parameters

first_register*:* specifies the Modbus/TCP PLC data registers to access when reading from PLC data into ISaGRAF variables. The PLC data type accessed is the same as Modbus PLC Device I/O boards detailed in Section ***Modbus Registers*** 18. .

plc_data_type*:* specifies the Modbus/TCP PLC data register type. Various data types are supported. See Section ***Data Conversion*** 37 for more information.

plc_device_address*:* This parameter specifies the PLC device (unit) address. Modbus PLC devices accessed at the same IP address (e.g. via a Modbus bridge) need to have a unique unit address in order to be identified. ISaGRAF may access data from different units on the same IP address or at different IP addresses. In these cases a separate I/O board will be required for each device.

must_write_rate*:* The unit for this parameter is milliseconds (ms) and specifies the rate at which the data for the output board is written to the PLC. Between "*must_write_rate*" periods, data is only written to the PLC when the ISaGRAF output variable values change. Individual I/O boards may have different must write rates allowing prioritization of data sent to PLC Device(s). See Section ***PLC Output Board Default Background Update Rate*** 64. Setting this parameter to 0 disables the time-based writing of output data. Data is written at ISaGRAF application startup (each channel) and thereafter only when individual ISaGRAF output channel variables change.

timeout*:* The Modbus/TCP PLC device driver provides a parameter for specifying the communications timeout on an individual I/O board (i.e. the timeout applies to communications associated with that board). Where this value is "0", the PLC device driver will use the default timeout (1200ms). The unit for this field is the millisecond (ms).

IP_address*:* This parameter specifies the IP network address that the SCADAPack E RTU connects to for communication with the PLC for this I/O board. Enter the IP address of the Modbus/TCP PLC, or Modbus gateway or bridge, as applicable.

PLC device communications resulting from the use of the OPERATE functin can be controlled by an ISaGRAF function block: **mtcpctrl**. The En_WR parameter on the block affects control operations being sent to the PLC device. For more information see *SCADAPack E Function Block Reference* manual for more information.

5.3.3 Modbus/TCP Registers

The following table describes the relationship between ISaGRAF Modbus/TCP I/O board types, the Modbus address ranges and functions used.

Table 6.1: ISaGRAF Modbus/TCP PLC Device I/O Board Register Access

ISaGRAF Modbus Slave I/O Board Type	ISaGRAF Modbus Board Address	Type of PLC Register	Uses "Open Modbus/TCP" Function Code
mtcpnnDI	1 - 9999	Reads discrete COILS	1
OPERATE on **mbusnnDI** variable	1- 9999	Presets discrete COILS	5
mtcpnnDI	10001-19999	Reads discrete INPUTS	2
mtcpnnDI	40001-65535	Reads HOLDING Registers as bits	3
OPERATE on **mtcpnnDI** variable	40001-65535	Presets HOLDING Register	16
mtcpnnDO	1- 9999	Writes discrete COILS	5
mtcpnnDO	40001- 65535	Writes bits to HOLDING Registers	16
mtcpnnAI	30001-39999	Reads INPUT Registers	4
mtcpnnAI	40001-65535	Reads HOLDING Registers	3
OPERATE on **mtcpnnAI** variable	40001-65535	Presets HOLDING Register	16
mtcpnnAO	40001-65535	Writes HOLDING Registers	16

5.3.4 Modbus/TCP Board Types

Details on specific ISaGRAF Modbus I/O boards used for defining the Modbus communication parameters can be found in the *SCADAPack E ISaGRAF I/O Connection Reference* manual.

Board Types
The following Modbus/TCP I/O board types are available:

Board Name	ref+	ISaGRAF Data Type	PLC Data Types Supported
mtcp16di	000E	16 Boolean Inputs	IEC DISCRETE, 984 DISCRETE from Input Status, Coil Registers, Holding Registers
mtcp32di	000E	32 Boolean Inputs	
mtcp16do	000F	16 Boolean Outputs	IEC DISCRETE, 984 DISCRETE to Coil Registers, Holding Registers
mtcp32do	000F	32 Boolean Outputs	
mtcp1ai	0010	1 Analog Input*1	IEC UINT, IEC INT, IEC DINT, IEC REAL, SWAP REAL from Input Registers, Holding Registers
mtcp4ai	0010	4 Analog Inputs*1	
mtcp8ai	0010	8 Analog Inputs*1	
mtcp16ai	0010	16 Analog Inputs*1	
mtcp32ai	0010	32 Analog Inputs*1	
mtcp64ai	0010	64 Analog Inputs*1	

mtcp1ao	0011	1 Analog Output*1	IEC UINT, IEC INT, IEC DINT, IEC REAL, SWAP REAL to Holding Registers
mtcp4ao	0011	4 Analog Outputs*1	
mtcp8ao	0011	8 Analog Outputs *1	
mtcp16ao	0011	16 Analog Outputs *1	
mtcp32ao	0011	32 Analog Outputs *1	
mtcp64ao	0011	64 Analog Outputs *1	

+ ref value is in Hexadecimal and is an internal ISaGRAF I/O board field.

*1 Analog input and output board conversion may be used. ISaGRAF “Operate” functions may also be used.

For more information see the *SCADAPack E ISaGRAF Technical Reference*

Information for Advanced ISaGRAF users:

Other I/O Boards, I/O configurations or Complex Equipment types based on the reference numbers shown in the above table are possible. The following guidelines need to be followed when configuring new I/O interface types:

- There is an upper limit of 32 I/O channels per digital board for the various Modbus/TCP board types.
- There is an upper limit of 64 I/O channels per analog board for the various Modbus/TCP board types.
- A plc_data_type user parameter is defined for PLC Device I/O boards. The value of this parameter field is a string field describes the data type of data being accessed (See Section ***Data Conversion*** 37 for more information.) – e.g. “IEC UINT”.
- An additional plc_dev_type hidden parameter string field describes the plc type, communication channel type and special controls. The value of this field is driver specific. E.g. “mtr” indicates advanced Modbus board (m), TCP socket interface (t), optional reset outputs on ISaGRAF application halted (r).

5.3.5 Open Modbus/TCP Conformance Classes

The Open Modbus/TCP standard defines conformance classes for Master & Slave (Client & Server) devices.

When using the ISaGRAF PLC I/O boards in the following way, the SCADAPack E RTU conforms to the requirements for

Open Modbus/TCP **Conformance CLASS 0** devices:

- **mtcp*xx*di** – board address: 40001-65535
  plc data type: IEC DISCRETE
  uses Modbus function code 3 – read multiple registers
- **mtcp*xx*do** – board address: 40001-65535
  plc data type: IEC DISCRETE
  uses Modbus function code 16 – write multiple registers
- **mtcp*xx*ai** – board address: 40001-65535
  plc data type: IEC INT, IEC UINT, IEC DINT, IEC REAL
  uses Modbus function code 3 – read multiple registers
- **mtcp*xx*ao** – board address: 40001-65535
  plc data type: IEC INT, IEC UINT, IEC DINT, IEC REAL
  uses Modbus function code 16 – write multiple registers

Use of the ISaGRAF PLC I/O boards in the following ways requires the PLC slave device (Open Modbus/TCP server) to be at least an

Open Modbus/TCP **Conformance CLASS 1** device:

- **mtcp*xx*di** – board address: 1-9999
  plc data type: IEC DISCRETE
  uses Modbus function code 1 – read coils
- **mtcp*xx*di** – board address: 10001-19999
  plc data type: IEC DISCRETE
  uses Modbus function code 2 – read input discrete (status)
- **mtcp*xx*ai** – board address: 30001-39999
  plc data type: IEC INT, IEC UINT, IEC DINT, IEC REAL
  uses Modbus function code 4 – read input registers
- **mtcp*xx*do** – board address: 1-9999
  plc data type: IEC INT, IEC UINT, IEC DINT, IEC REAL
  uses Modbus function code 5 – write coil

PLC data type options additional to those listed here are available. The above types are a selection of those defined in the Open Modbus/TCP Conformance Classes. Refer to ***Modbus PLC Data Types*** 38 for a complete listing of supported data types.

5.4 Modbus RTU in TCP I/O Board Interface

The ***mrtp*** ISaGRAF boards are used by the SCADAPack E RTU to communicate via Ethernet interfaces with Modbus RTU in TCP protocol peripheral devices (herein described as PLCs).

Each of the ISaGRAF application's PLC I/O boards use a separate Modbus RTU in TCP request to read or write its data. Improved Modbus communication efficiency can be achieved by grouping Modbus registers together and using less I/O boards with a larger number of channels (e.g. mrtp64ai), rather than more I/O boards with a smaller number of channels.

A maximum of 100 PLC Device I/O boards types (total of each type) may be configured in total across both RTU ISaGRAF applications.

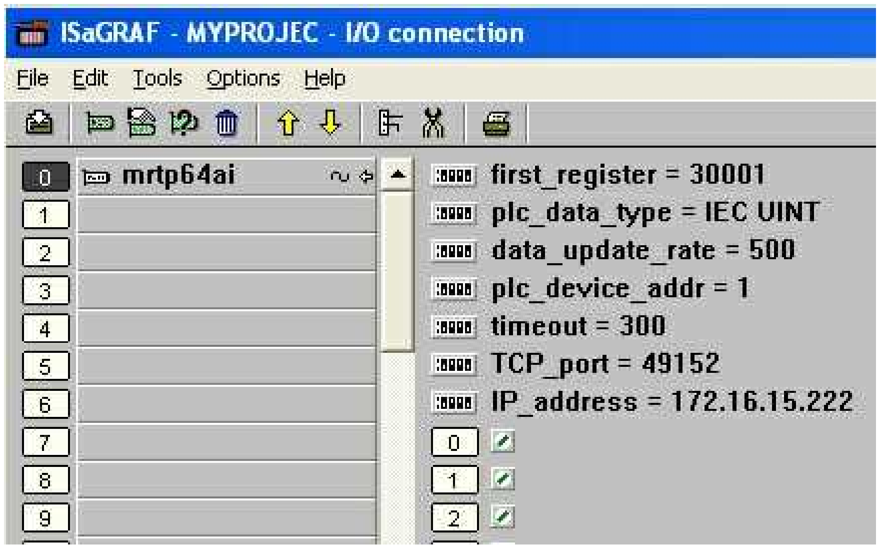
A corresponding pair of system points relates to each PLC Slave I/O board as described in Section ***System Points*** 85.

Modbus RTU in TCP boards utilize default IEC data types. Where applicable, the data type may be available for the user to choose.

PLC device communications using these I/O boards can be controlled by an ISaGRAF function block: **mtcpctrl**.

5.4.1 Modbus RTU in TCP Input Boards

Modbus RTU in TCP PLC Input Board variables are updated at the start of an ISaGRAF application scan. The value presented to the ISaGRAF variables is the value returned by the PLC from the previous read request. This read may have occurred during previous ISaGRAF application scans. The “data update rate” parameter on the I/O board sets the "scan" rate of the PLC data. The PLC communication status is updated if there is a status code returned from the PLC, or no response from the PLC after a data request by the RTU. (See Section ***Modbus Status Values*** 62). The status is cleared by the SCADAPack E RTU upon successful communications. To catch transient status codes you can use ISaGRAF code to store non-zero values.



first_register = 30001
plc_data_type = IEC UINT
data_update_rate = 500
plc_device_addr = 1
timeout = 300
TCP_port = 49152
IP_address = 172.16.15.222

Figure 6.2: ISaGRAF Modbus RTU in TCP board

Input Board Parameters

first_register: specifies the Modbus RTU in TCP PLC data registers to access when reading from PLC data into ISaGRAF variables. The PLC data type accessed is the same as Modbus PLC Device I/O boards detailed in Section ***Modbus Registers*** 18.

plc_data_type: specifies the Modbus RTU in TCP PLC data register type. Various data types are supported. See Section ***Data Conversion*** 37 for more information.

data_update_rate: The unit for this parameter is the millisecond (ms), and specifies the rate at which the data for the Input board is extracted from the PLC. Individual I/O boards may have different data update rates allowing prioritization of data extracted from a PLC Device. The SCADAPack E RTU may not be able to read requested PLC data within the time set by the data update rate depending on the quantity of data to be read, rate of write requests and PLC communication speed. In this case the update rates will be slower.

plc_device_address: This parameter specifies the PLC device (unit) address. Modbus PLC devices accessed at the same IP address (e.g. via a Modbus bridge) need to have a unique unit address in order to be identified. ISaGRAF may access data from different units on the same IP address or at different IP addresses. In these cases a separate I/O board will be required for each device.

timeout: The Modbus RTU in TCP device driver provides a parameter for specifying the communications timeout on an individual I/O board (i.e. the timeout applies to communications associated with that board). Where this value is “0”, the PLC device driver will use the default timeout (1200ms). Units for this field are in ms.

TCP_port: This parameter specifies the port number of the Modbus RTU in TCP server.

IP_address: This parameter specifies the IP network address that the SCADAPack E RTU connects to for communication with the PLC for this I/O board. Enter the IP address of the Modbus RTU in TCP PLC, or Modbus bridge if applicable.

Controlling PLC Device Communications

PLC device communications using these I/O boards can be controlled by an ISaGRAF function block: **mtcpctrl**. The En_RD parameter on the block affects PLC device Input Boards. For more information see *SCADAPack E Function Block Reference* manual for more information.

“OPERATE” on Input Boards

The ISaGRAF “OPERATE” function may be used on Modbus RTU in TCP Input Boards where the register read by the input board is also writeable e.g. coils or holding registers. This permits registers to be inputs into ISaGRAF but have them “Preset” or initialized in the PLC by ISaGRAF.

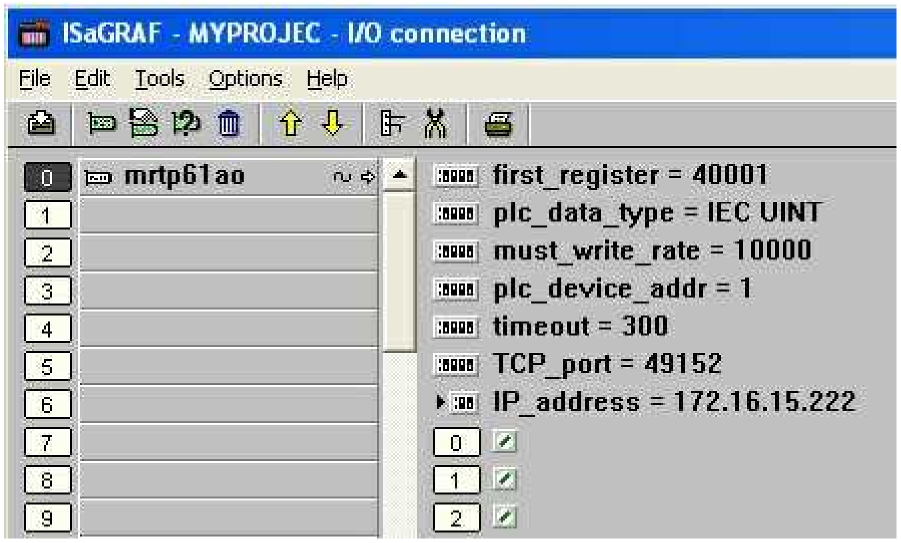
For more information see the *SCADAPack E ISaGRAF Technical Reference* manual.

PLC device communications resulting from the use of the OPERATE functin can be controlled by an ISaGRAF function block: **mtcpctrl**. The En_WR parameter on the block affects control operations being sent to the PLC device. For more information see *SCADAPack E Function Block Reference* manual for more information.

5.4.2 Modbus RTU in TCP Output Boards

Modbus RTU in TCP PLC Output Board data is written to the PLC when an ISaGRAF application changes the value of a variable attached to the Output board. In addition, output board data is written to the PLC under the following conditions:

- When the ISaGRAF application starts, output board data is written
- If the PLC does not respond to a control, it is re-sent until it is responded
- Output board data is rewritten at a background update rate


ISaGRAF - MYPROJEC - I/O connection
File Edit Tools Options Help
0
1
2
3
4
5
6
7
8
9

mrtp61ao

first_register = 40001
plc_data_type = IEC UINT
must_write_rate = 10000
plc_device_addr = 1
timeout = 300
TCP_port = 49152
IP_address = 172.16.15.222

Figure 6.3: ISaGRAF Modbus RTU in TCP output board

Output Board Parameters

first_register: specifies the Modbus RTU in TCP PLC data registers to access when reading from PLC data into ISaGRAF variables. The PLC data type accessed is the same as Modbus PLC Device I/O boards detailed in Section ***Modbus Registers*** 18 . .

plc_data_type: specifies the Modbus RTU in TCP PLC data register type. Various data types are supported. See Section ***Data Conversion*** 37 for more information.

plc_device_address: This parameter specifies the PLC device (unit) address. Modbus PLC devices accessed at the same IP address (e.g. via a Modbus bridge) need to have a unique unit address in order to be identified. ISaGRAF may access data from different units on the same IP address or at different IP addresses. In these cases a separate I/O board will be required for each device.

must_write_rate: The unit for this parameter is milliseconds (ms) and specifies the rate at which the data for the output board is written to the PLC. Between "*must_write_rate*" periods, data is only written to the PLC when the ISaGRAF output variable values change. Individual I/O boards may have different must write rates allowing prioritization of data sent to PLC Device(s). See Section ***PLC Output Board Default Background Update Rate*** 64 .

timeout: The Modbus RTU in TCP device driver provides a parameter for specifying the communications timeout on an individual I/O board (i.e. the timeout applies to communications associated with that board). Where this value is "0", the PLC device driver will use the default timeout (1200ms). The unit for this field is the millisecond (ms).

TCP_port: This parameter specifies the port number of the Modbus RTU in TCP server.

IP_address: This parameter specifies the IP network address that the SCADAPack E RTU connects to for communication with the PLC for this I/O board. Enter the IP address of the Modbus RTU in TCP PLC, or Modbus gateway or bridge, as applicable.

PLC device communications resulting from the use of the OPERATE functin can be controlled by an ISaGRAF function block: **mtcpctrl**. The En_WR parameter on the block affects control operations being sent to the PLC device. For more information see *SCADAPack E Function Block Reference* manual for more information.

5.4.3 Modbus RTU in TCP Registers

The following table describes the relationship between ISaGRAF Modbus RTU in TCP I/O board types, the Modbus address ranges and functions used.

Table 6.1: ISaGRAF Modbus/TCP PLC Device I/O Board Register Access

ISaGRAF Modbus Slave I/O Board Type	ISaGRAF Modbus Board Address	Type of PLC Register	Uses "Open Modbus/TCP" Function Code
mrtpnnDI	1 - 9999	Reads discrete COILS	1
OPERATE on **mrtpnnDI** variable	1- 9999	Presets discrete COILS	5
mrtpnnDI	10001-19999	Reads discrete INPUTS	2
mrtpnnDI	40001-65535	Reads HOLDING Registers as bits	3
OPERATE on **mrtpnnDI** variable	40001-65535	Presets HOLDING Register	16
mrtpnnDO	1- 9999	Writes discrete COILS	5
mrtpnnDO	40001- 65535	Writes bits to HOLDING Registers	16
mrtpnnAI	30001-39999	Reads INPUT Registers	4
mrtpnnAI	40001-65535	Reads HOLDING Registers	3
OPERATE on **mrtpnnAI** variable	40001-65535	Presets HOLDING Register	16
mrtpnnAO	40001-65535	Writes HOLDING Registers	16

5.4.4 Modbus RTU in TCP Board Types

The following Modbus RTU in TCP I/O board types are available:

Board Name	ref+	ISaGRAF Data Type	PLC Data Types Supported
mrtp16di	000E	16 Boolean Inputs	IEC DISCRETE, 984 DISCRETE from Input Status, Coil Registers, Holding Registers
mrtp32di	000E	32 Boolean Inputs	
mrtp16do	000F	16 Boolean Outputs	IEC DISCRETE, 984 DISCRETE to Coil Registers, Holding Registers
mrtp32do	000F	32 Boolean Outputs	
mrtp1ai	0010	1 Analog Input*[1]	IEC UINT, IEC INT, IEC DINT, IEC REAL, SWAP REAL from Input Registers, Holding Registers
mrtp4ai	0010	4 Analog Inputs*[1]	
mrtp8ai	0010	8 Analog Inputs*[1]	
mrtp16ai	0010	16 Analog Inputs*[1]	
mrtp32ai	0010	32 Analog Inputs*[1]	
mrtp64ai	0010	64 Analog Inputs*[1]	
mrtp1ao	0011	1 Analog Output*[1]	IEC UINT, IEC INT, IEC DINT, IEC REAL, SWAP REAL to Holding Registers
mrtp4ao	0011	4 Analog Outputs*[1]	
mrtp8ao	0011	8 Analog Outputs *[1]	
mrtp16ao	0011	16 Analog Outputs * 1	
mrtp32ao	0011	32 Analog Outputs * 1	
mrtp64ao	0011	64 Analog Outputs * 1	

+ ref value is in Hexadecimal and is an internal ISaGRAF I/O board field.

*1 Analog input and output board conversion may be used. ISaGRAF “Operate” functions may also be used.

For more information see the *SCADAPack E ISaGRAF Technical Reference*

Information for Advanced ISaGRAF users:

Other I/O Boards, I/O configurations or Complex Equipment types based on the reference numbers shown in the above table are possible. The following guidelines need to be followed when configuring new I/O interface types:

- There is an upper limit of 32 I/O channels per digital board for the various Modbus RTU in TCP board types.
- There is an upper limit of 64 I/O channels per analog board for the various Modbus RTU in TCP board types.
- A plc_data_type user parameter is defined for PLC Device I/O boards. The value of this parameter field is a string field describes the data type of data being accessed (See Section ***Data Conversion*** 37 for more information.) – e.g. “IEC UINT”.

5.5 Data Conversion

A single ISaGRAF PLC I/O board will support one Modbus PLC data type for channels on that I/O board. Where real ISaGRAF analog variables are attached to an integer ISaGRAF PLC I/O board, or where integer ISaGRAF analog variables are attached to a real ISaGRAF PLC I/O board, conversion rules apply as detailed in Section ***Modbus Data Conversion*** 40 and in the *SCADAPack E ISaGRAF Technical Reference* manual.

The exception to this is where the *PLC data type* for an I/O board is “IEC DINT”. Integer and real ISaGRAF analog variables will in this case use “IEC DINT” and “IEC REAL” PLC data types respectively on the same I/O board. Data conversion handling depends upon the actual PLC Data type. See the following sections:

- ***Modbus PLC Data Types*** 38
- ***Modbus Data Conversion*** 40

5.5.1 Modbus PLC Data Types

The following data types are supported by the SCADAPack E Modbus serial and Open Modbus/TCP PLC interfaces.

- IEC DISCRETE

Binary (discrete) data packed into an 8-bit value where the least significant bit of the value represents the low discrete bit number. For a protocol message that contains 16 discrete coils at addresses 11-26 for example, coil 11 is represented by the least significant bit of the first byte in the protocol, and coil 26 is represented by the most significant bit of the second byte in the protocol. This data type can be used to access PLC inputs, coils or holding register bits.

- 984 DISCRETE

Binary (discrete) data usually packed into a 16-bit value where the least significant bits of the 16-bit value represent the high discrete bit numbers. For a protocol message that contains 16 discrete coils at addresses 30-45 for instance, coil 30 is represented by the most significant bit of the 16-bit value, and coil 45 is represented by the least significant bit of the 16-bit value. This data type can be used to access PLC inputs, coils or holding register bits.

- IEC UINT

Unsigned 16-bit integer value. Valid values are 0 ~ 65535. This is the default data type used by the RTU for Modbus PLC register data. This data type can be used to access PLC input registers or holding registers.

- IEC INT

Signed 16-bit integer value. Valid values are –32768 ~ 32767. This data type can be used to access PLC input registers or holding registers.

- IEC DINT

Signed 32-bit double integer value, organized as two words in the protocol in Little Endian format (least significant word first). Valid values are –231 ~ 231-1. I/O boards utilizing this data type will automatically select between IEC DINT data format for ISaGRAF integer analog variables, and IEC REAL data format for ISaGRAF real analog variables on the I/O board. This data type generally accesses a consecutive pair of 16-bit holding registers.

- IEC REAL

IEEE-754 format 32-bit floating point real value, organized as two words in the protocol in Little Endian register format (least significant word in first register). This data type generally accesses a consecutive pair of 16-bit holding registers.

- SWAP REAL

IEEE-754 format 32-bit floating point real value, organized as two words in the protocol in swapped (Big Endian) register format (most significant word in first register). This data type generally accesses a consecutive pair of 16-bit holding registers.

- IEC UDINT

Unsigned 32-bit double integer value, organized as two words in the protocol in Little Endian format (least significant word first). Valid values are 0 ~ 232-1. This data type is **not** supported in ISaGRAF I/O boards. This data type is supported by Modbus Slave and Modbus/TCP Server interfaces. See Modbus Slave / Server ***Analog Addresses*** 75. This data type generally accesses a consecutive pair of 16-bit holding registers.

5.5.2 Modbus Data Conversion

mbusxxdi, mbusxxdo, mtcpxxdi, mtcpxxdo I/O boards require no data conversion other than the order of discrete bit numbers. “984 DISCRETE” data type reverses the bit order within 16-bit groups (in comparison to "IEC DISCRETE").

mbusxxai, mbusxxao, mtcpxxai, mtcpxxao I/O boards require no data conversion other than register pair ordering for 32-bit values. “SWAP REAL” data type reverses the holding register order for 32-bit floating point register pair (in comparison to "IEC REAL").

Table 7.140 below provides examples of Modbus data conversion for SCADAPack E RTU analog input and analog output Modbus PLC I/O boards. Conversion examples show integer / real conversion results, including numeric range truncation (for ***mbusxxao*** & ***mtcpxxao***)

ISaGRAF “OPERATE” function for ***mbusxxai*** and ***mtcpxxai*** boards uses the same numeric conversion and range truncation when initializing input board variable values as does the corresponding ***mbusxxao*** or ***mtcpxxao*** board when setting output board registers.

The Open Modbus/TCP Server data conversion for SCADAPack E database point values is detailed in Section ***Supported Data Types***81.

Table 7.1: ISaGRAF / Modbus Data Conversion

SCADAPack E I/O Board Type	**Data Type / Range**	**ISaGRAF Variable Type**	**Conversion Examples**
mbusxxai mtcpxxai mrtpxxai	IEC UINT 0 ~ 65535	Integer	Reg value = 40000, Variable = 40000
		Real	Reg value = 45678, Variable = 45678.0
mbusxxai mtcpxxai mrtpxxai	IEC INT -32768 ~ 32767	Integer	Reg value = 20000, Variable = 20000
		Real	Reg value = -15000, Variable = -15000.0
mbusxxai mtcpxxai mrtpxxai	IEC REAL SWAP REAL	Integer	Reg pair value = 12345.678 Variable = 12345
		Real	Reg pair value = -159.876 Variable = -159.876
mbusxxai mtcpxxai mrtpxxai	IEC DINT -231 ~ 231-1	Integer (IEC DINT)	Reg pair value = 12345678 Variable = 12345678
		Real (IEC REAL)	Reg pair value = 9988.77 Variable = 9988.77
			Variable value = -987, Reg = 0

mbusxxao **modxxao** **mtcpxxao** **mrtpxxao**	IEC UINT 0 ~ 65535	Integer	Variable value = 50000, Reg = 50000 Variable value = 70000, Reg = 65535
		Real	Variable value = -99.33, Reg = 0 Variable value = 45678.3, Reg = 45678 Variable value = 123456.7, Reg = 65535
mbusxxao **mtcpxxao** **mrtpxxao**	IEC INT -32768 ~ 32767	integer	Variable value = -39000, Reg = -32768 Variable value = 10000, Reg = 10000 Variable value = 45000, Reg = 32767
		real	Variable value =-45678.0, Reg = -32768 Variable value = -99.33, Reg = -99 Variable value = 34568.7, Reg = 32767

mbusxxao
mtcpxxao
mrtpxxao
	IEC REAL
	SWAP REAL
		integer	Variable = -99000, Reg pair = -99000.0
			Variable = 100000, Reg pair = 100000.0

		real	Variable = -77.06, Reg pair = --77.06
			Variable =123456.7, Reg pair = 123456.7

mbusxxao **mtcpxxao** **mrtpxxao**	IEC DINT -231 ~ 231-1	integer (IEC DINT)	Variable=12345678, Reg pair =12345678
		real (IEC REAL)	Variable = 9988.77, Reg pair = 9988.77

5.6 Modbus TCP Complex Equipment Types

ISaGRAF complex equipment types can be used with the SCADAPack E RTU to provide a simple device template to an ISaGRAF application.

Complex equipment types are usually specific to particular equipment models.

ISaGRAF “Complex Equipment” types are comprised of configurations similar to I/O boards.

When a Complex Equipment type includes PLC Device I/O board configurations, each such I/O board configuration within the Complex Equipment type counts towards the limit of 50 Slave I/O boards and has a corresponding pair of system points as described in Section **System Points** 85.

Up to 40 Complex Equipment types, each with 2 or 3 I/O boards (summing up to about 100 I/O boards) can inter-operate with a single SCADAPack E RTU at the same time.

5.6.1 Complex Equipment Types Summary

The following ISaGRAF Complex Equipment types are provided for use with the SCADAPack E RTU.

Table 8.1: ISaGRAF Complex Equipment Types

Complex Equipment Type	Equipment Model	Description
adi34000	170 ADI 340 00	Schneider TSX Momentum 16 Pt. Digital Input module with 170 ENT 110 00 Ethernet communication adapter
adi35000	170 ADI 350 00	Schneider TSX Momentum 32 Pt. Digital Input module with 170 ENT 110 00 Ethernet communication adapter
adm35010	170 ADM 350 10	Schneider TSX Momentum mixed 16 Pt. Digital Input / 16. Pt. Digital Output module with 170 ENT 110 00 Ethernet communication adapter
aai03000	170 AAI 030 00	Schneider TSX Momentum Analog 8 Channel Differential Input module with 170 ENT 110 00 Ethernet communication adapter
aai14000	170 AAI 140 00	Schneider TSX Momentum Analog 16 Channel Single-Ended Input module with 170 ENT 110 00 Ethernet communication adapter
ado35000	170 ADO 350 00	Schneider TSX Momentum 32 Pt. Digital Output module with 170 ENT 110 00 Ethernet communication adapter

For more information see Schneider Automation documents: 870 USE 002 00 and 870 USE 112 00.

5.6.2 "adi34000" TSX Momentum 170 ADI 340 00

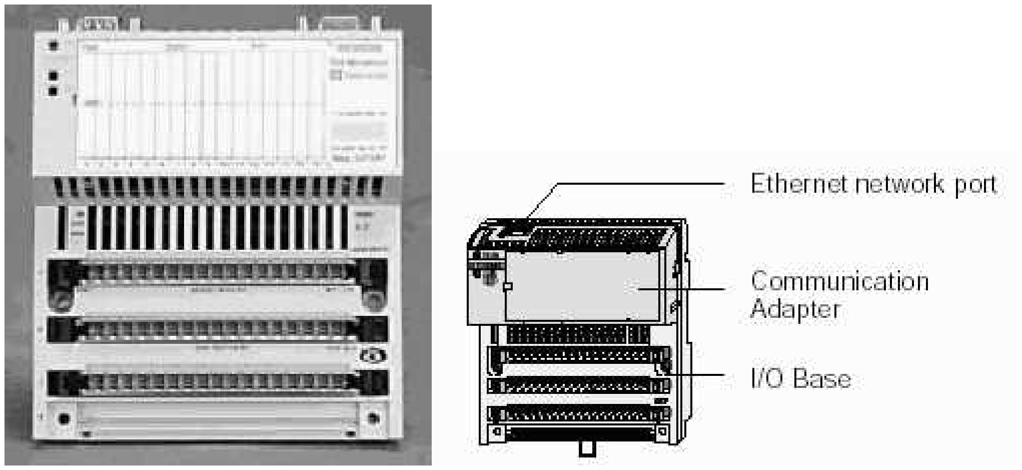
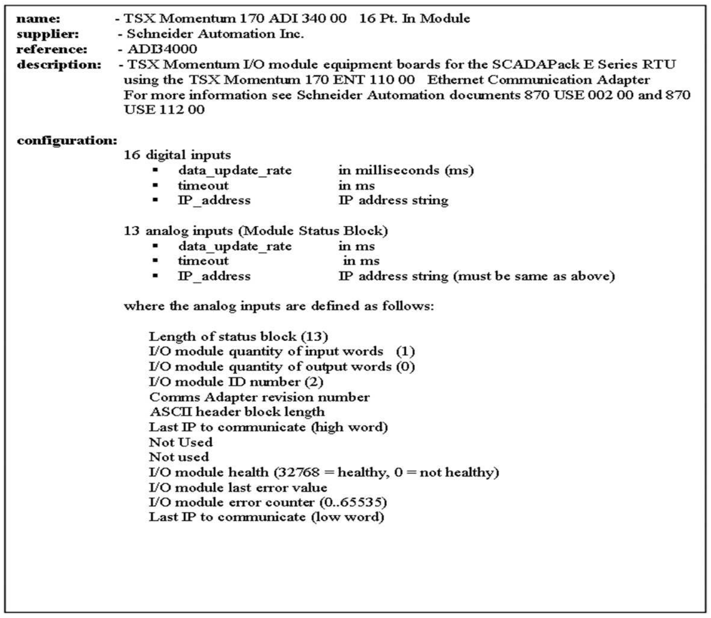


Figure 8.1: Schneider TSX Momentum I/O Units

This complex equipment type permits a TSX Momentum 16 Point Input module to be used as distributed Ethernet I/O on a SCADAPack E RTU.

Using this module requires a BOOTP entry be added to the SCADAPack E RTU BOOTP server configuration table. This may be added via the RTU command prompt, or via SCADAPack E Configurator. For more information see Section ***How Do I Change a Modbus/TCP I/O Device*** 65 or the *SCADAPack E Operational Reference* manual.

An IP address needs to be entered for each I/O board's "IP_address" parameter.
The value needs to be the same for boards within the same complex equipment type.
Figure 8.2 44 details the ISaGRAF Complex Equipment Technical Note.



```
name:          - TSX Momentum 170 ADI 340 00   16 Pt. In Module
supplier:      - Schneider Automation Inc.
reference:     - ADI34000
description:   - TSX Momentum I/O module equipment boards for the SCADAPack E Series RTU
                 using the TSX Momentum 170 ENT 110 00   Ethernet Communication Adapter
                 For more information see Schneider Automation documents 870 USE 002 00 and 870
                 USE 112 00

configuration:
                 16 digital inputs
                     ▪  data_update_rate        in milliseconds (ms)
                     ▪  timeout                 in ms
                     ▪  IP_address              IP address string

                 13 analog inputs (Module Status Block)
                     ▪  data_update_rate        in ms
                     ▪  timeout                  in ms
                     ▪  IP_address              IP address string (must be same as above)

                 where the analog inputs are defined as follows:

                     Length of status block (13)
                     I/O module quantity of input words   (1)
                     I/O module quantity of output words (0)
                     I/O module ID number (2)
                     Comms Adapter revision number
                     ASCII header block length
                     Last IP to communicate (high word)
                     Not Used
                     Not used
                     I/O module health (32768 = healthy, 0 = not healthy)
                     I/O module last error value
                     I/O module error counter (0..65535)
                     Last IP to communicate (low word)
```

Figure 8.2: ISaGRAF "adi34000" Complex Equipment Technical Note

5.6.3 "adi35000" TSX Momentum 170 ADI 350 00

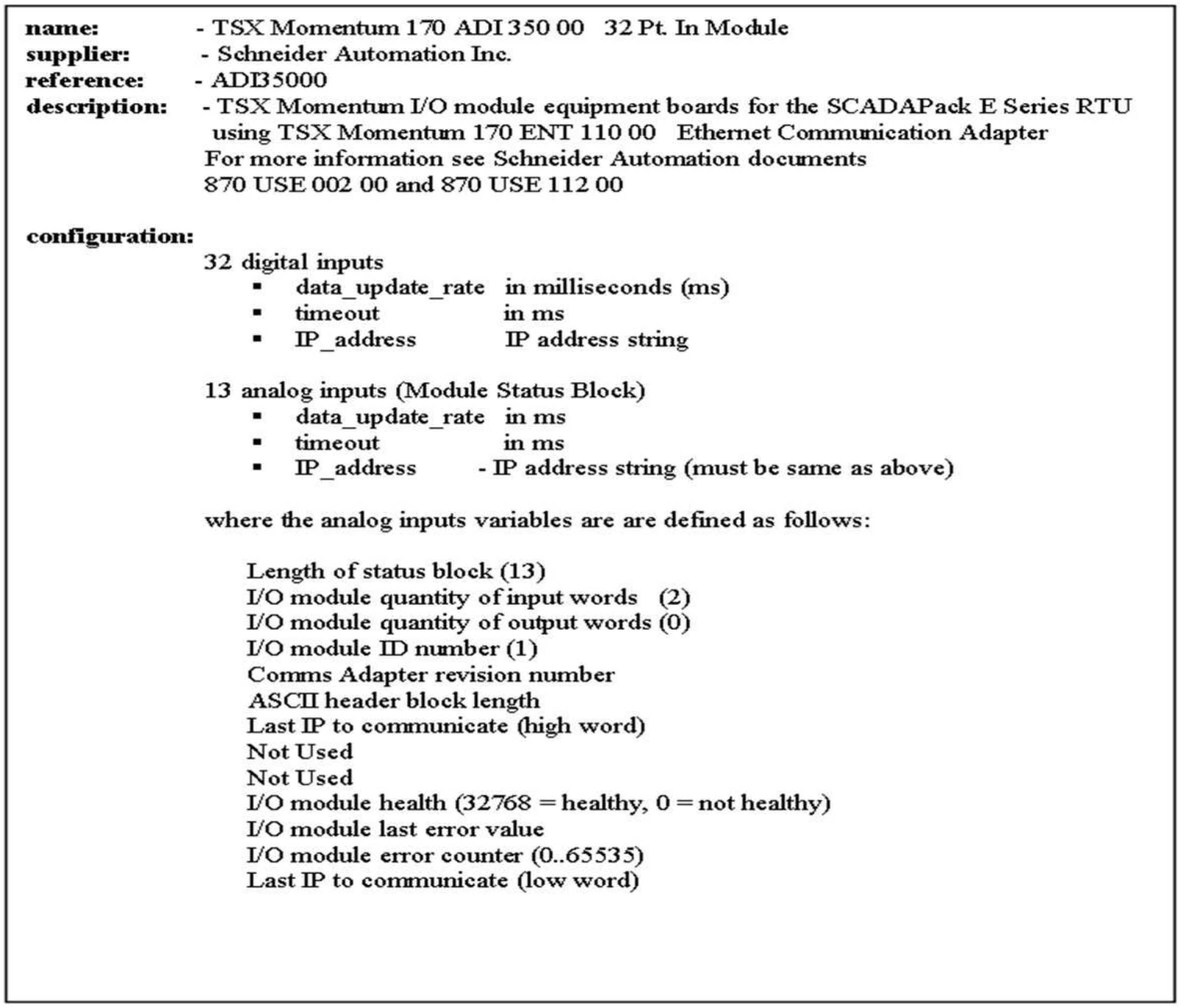
This complex equipment type permits a TSX Momentum 32 Point Input module to be used as distributed Ethernet I/O on a SCADAPack E RTU.

Using this module requires a BOOTP entry be added to the SCADAPack E RTU BOOTP server configuration table. This may be added via the RTU command prompt, or via SCADAPack E Configurator. For more information see Section ***How Do I Change a Modbus/TCP I/O Device*** 65 or the *SCADAPack E Operational Reference* manual.

An IP address needs to be entered for each I/O board's "IP_address" parameter.

The value needs to be the same for boards within the same complex equipment type.

Figure 8.3 46 details the ISaGRAF Complex Equipment Technical Note.



```
name:           - TSX Momentum 170 ADI 350 00   32 Pt. In Module
supplier:       - Schneider Automation Inc.
reference:      - ADI35000
description:    - TSX Momentum I/O module equipment boards for the SCADAPack E Series RTU
                  using TSX Momentum 170 ENT 110 00   Ethernet Communication Adapter
                 For more information see Schneider Automation documents
                 870 USE 002 00 and 870 USE 112 00

configuration:
                 32 digital inputs
                     ▪   data_update_rate   in milliseconds (ms)
                     ▪   timeout            in ms
                     ▪   IP_address         IP address string

                 13 analog inputs (Module Status Block)
                     ▪   data_update_rate   in ms
                     ▪   timeout            in ms
                     ▪   IP_address       - IP address string (must be same as above)

                 where the analog inputs variables are are defined as follows:

                     Length of status block (13)
                     I/O module quantity of input words   (2)
                     I/O module quantity of output words (0)
                     I/O module ID number (1)
                     Comms Adapter revision number
                     ASCII header block length
                     Last IP to communicate (high word)
                     Not Used
                     Not Used
                     I/O module health (32768 = healthy, 0 = not healthy)
                     I/O module last error value
                     I/O module error counter (0..65535)
                     Last IP to communicate (low word)
```

Figure 8.3: ISaGRAF "adi35000" Complex Equipment Technical Note

5.6.4 "adm35010" TSX Momentum 170 ADM 350 10

“adm35010” TSX Momentum 170 ADM 350 10

This complex equipment type permits a TSX Momentum mixed 16 Point Input / 16 Point Output module to be used as distributed Ethernet I/O on a SCADAPack E RTU.

Using this module requires a BOOTP entry be added to the SCADAPack E RTU BOOTP server configuration table. This may be added via the RTU command prompt, or via SCADAPack E Configurator. For more information see Section ***How Do I Change a Modbus/TCP I/O Device*** 65 or the *SCADAPack E Operational Reference* manual.

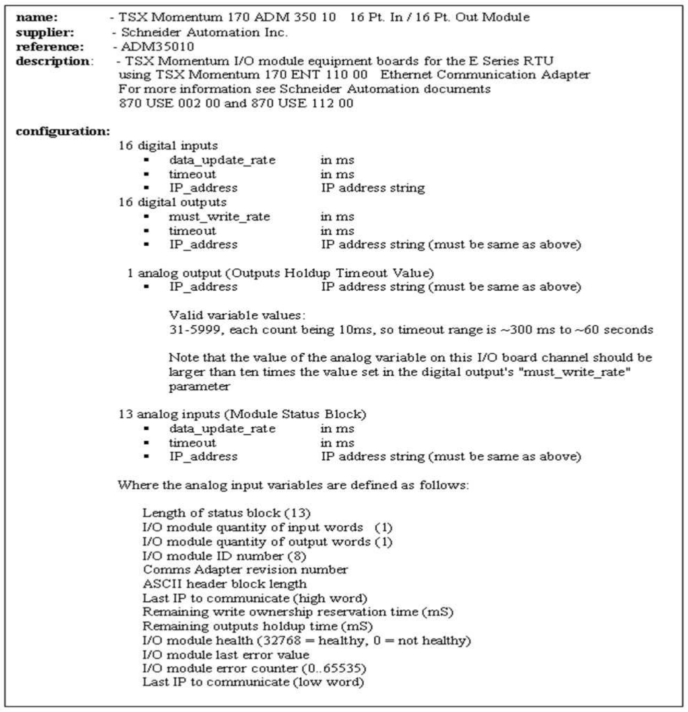
Special timing parameter considerations need to be made for this Complex I/O equipment type when using the digital output channels. The module will drop its outputs after a period of no communication with the RTU. The period is specified by the value of this variable. The value of the variable must be much longer than the “must_write_rate” parameter field for the digital outputs (e.g. 3 times longer). 1 count of the variable value is equivalent to 10ms. The “must_write_rate” field is entered in ms. The PLC cache interface treats communications with remote PLC devices synchronously that is a control may have to wait for an outstanding poll response from the PLC prior to being able to deliver a queued control request. Particularly where multiple complex equipment types are used on the same SCADAPack E RTU, consideration needs to be given to potential delays, which may cause outputs to be temporarily cleared by the unit due to a perceived no communications with the RTU..

The digital output board on this complex equipment type uses hidden string “mtr” for the plc_dev_type. “m” is for advanced Modbus, “t” for TCP socket interface, and “r” causes the RTU to clear the outputs to the module when the ISaGRAF application is stopped.

An IP address must be entered for each I/O board’s “IP_address” parameter.

The value needs to be the same for I/O boards within the same complex equipment type.

Figure 8.4 47 details the ISaGRAF Complex Equipment Technical Note.



```
name:          - TSX Momentum 170 ADM 350 10    16 Pt. In / 16 Pt. Out Module
supplier:      - Schneider Automation Inc.
reference:     - ADM35010
description:   - TSX Momentum I/O module equipment boards for the E Series RTU
                 using TSX Momentum 170 ENT 110 00   Ethernet Communication Adapter
                 For more information see Schneider Automation documents
                 870 USE 002 00 and 870 USE 112 00

configuration:
                 16 digital inputs
                   ▪  data_update_rate      in ms
                   ▪  timeout               in ms
                   ▪  IP_address            IP address string
                 16 digital outputs
                   ▪  must_write_rate       in ms
                   ▪  timeout               in ms
                   ▪  IP_address            IP address string (must be same as above)

                  1 analog output (Outputs Holdup Timeout Value)
                   ▪  IP_address            IP address string (must be same as above)

                      Valid variable values:
                      31-5999, each count being 10ms, so timeout range is ~300 ms to ~60 seconds

                      Note that the value of the analog variable on this I/O board channel should be
                      larger than ten times the value set in the digital output's "must_write_rate"
                      parameter

                 13 analog inputs (Module Status Block)
                   ▪  data_update_rate      in ms
                   ▪  timeout               in ms
                   ▪  IP_address            IP address string (must be same as above)

                 Where the analog input variables are defined as follows:

                    Length of status block (13)
                    I/O module quantity of input words   (1)
                    I/O module quantity of output words (1)
                    I/O module ID number (8)
                    Comms Adapter revision number
                    ASCII header block length
                    Last IP to communicate (high word)
                    Remaining write ownership reservation time (mS)
                    Remaining outputs holdup time (mS)
                    I/O module health (32768 = healthy, 0 = not healthy)
                    I/O module last error value
                    I/O module error counter (0..65535)
                    Last IP to communicate (low word)
```

Figure 8.4: ISaGRAF “adm35010” Complex Equipment Technical Note

5.6.5 "aai03000" TSX Momentum 170 AAI 030 00

“aai03000” TSX Momentum 170 AAI 030 00

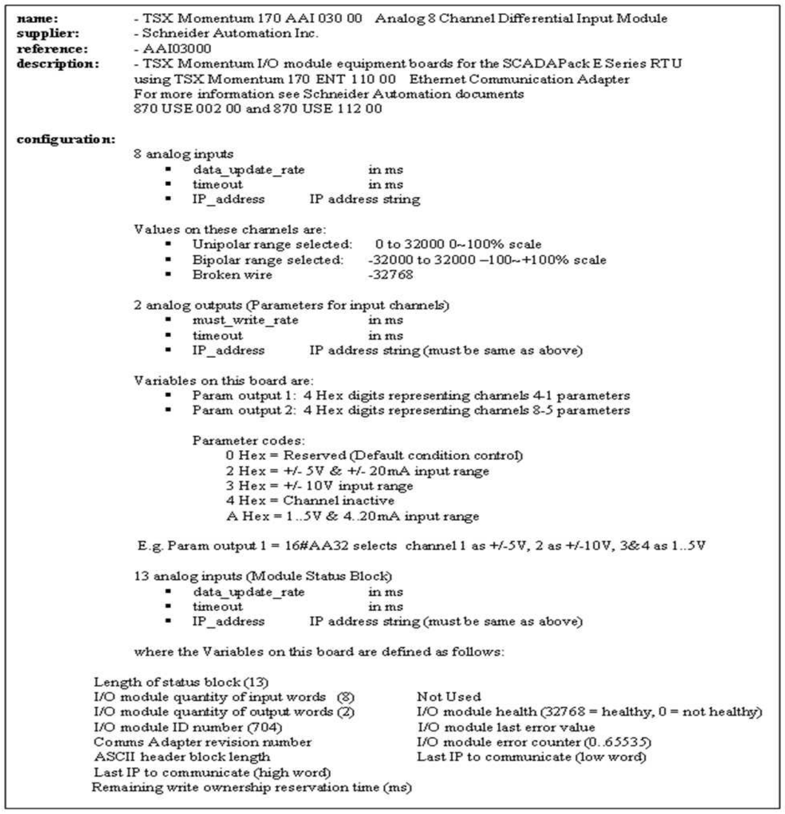
This complex equipment type permits a TSX Momentum Analog 8 Channel Differential Input module to be used as distributed Ethernet I/O on a SCADAPack E RTU.

Using this module requires a BOOTP entry be added to the SCADAPack E RTU BOOTP server configuration table. This may be added via the RTU command prompt, or via SCADAPack E Configurator. For more information see Section ***How Do I Change a Modbus/TCP I/O Device*** 65 or the *SCADAPack E Operational Reference* manual.

An IP address needs to be entered for each I/O board’s “IP_address” parameter. The value needs to be the same for boards within the same complex equipment type.

Special configuration parameters need to be set up for correct operation of this module. It is recommended that ISaGRAF variables be attached to the “Param” I/O board channels with initial values to configure the input range for the analog input channels on this device. It is suggested for clarity that the attached ISaGRAF output variables be set with a display format “+1A2B” to indicate hexadecimal when viewed. Each Hex digit then represents the configuration mode for 1 Analog Input channel. The configuration digit is specific to this type of module. For clarity it is recommended that a comment field be applied to the ISaGRAF output variable explicitly describing the AI channel parameters.

Figure 8.5 49 details the ISaGRAF Complex Equipment Technical Note.


```
name:           - TSX Momentum 170 AAI 030 00   Analog 8 Channel Differential Input Module
supplier:       - Schneider Automation Inc.
reference:      - AAI03000
description:    - TSX Momentum I/O module equipment boards for the SCADAPack E Series RTU
                using TSX Momentum 170 ENT 110 00   Ethernet Communication Adapter
                For more information see Schneider Automation documents
                870 USE 002 00 and 870 USE 112 00

configuration:
                8 analog inputs
                    ▪   data_update_rate            in ms
                    ▪   timeout                     in ms
                    ▪   IP_address        IP address string

                Values on these channels are:
                    ▪   Unipolar range selected:     0 to 32000 0~100% scale
                    ▪   Bipolar range selected:     -32000 to 32000 –100~+100% scale
                    ▪   Broken wire                 -32768

                2 analog outputs (Parameters for input channels)
                    ▪   must_write_rate             in ms
                    ▪   timeout                     in ms
                    ▪   IP_address        IP address string (must be same as above)

                Variables on this board are:
                    ▪   Param output 1:  4 Hex digits representing channels 4-1 parameters
                    ▪   Param output 2:  4 Hex digits representing channels 8-5 parameters

                        Parameter codes:
                            0 Hex = Reserved (Default condition control)
                            2 Hex = +/- 5V & +/- 20mA input range
                            3 Hex = +/- 10V input range
                            4 Hex = Channel inactive
                            A Hex = 1..5V & 4..20mA input range

                 E.g. Param output 1 = 16#AA32 selects  channel 1 as +/-5V, 2 as +/-10V, 3&4 as 1..5V

                13 analog inputs (Module Status Block)
                    ▪   data_update_rate            in ms
                    ▪   timeout                     in ms
                    ▪   IP_address        IP address string (must be same as above)

                where the Variables on this board are defined as follows:

          Length of status block (13)
          I/O module quantity of input words   (8)       Not Used
          I/O module quantity of output words (2)        I/O module health (32768 = healthy, 0 = not healthy)
          I/O module ID number (704)                     I/O module last error value
          Comms Adapter revision number                  I/O module error counter (0..65535)
          ASCII header block length                      Last IP to communicate (low word)
          Last IP to communicate (high word)
         Remaining write ownership reservation time (ms)
```

Figure 8.5: ISaGRAF “aai03000” Complex Equipment Technical Note

5.6.6 "aai14000" TSX Momentum 170 AAI 140 00

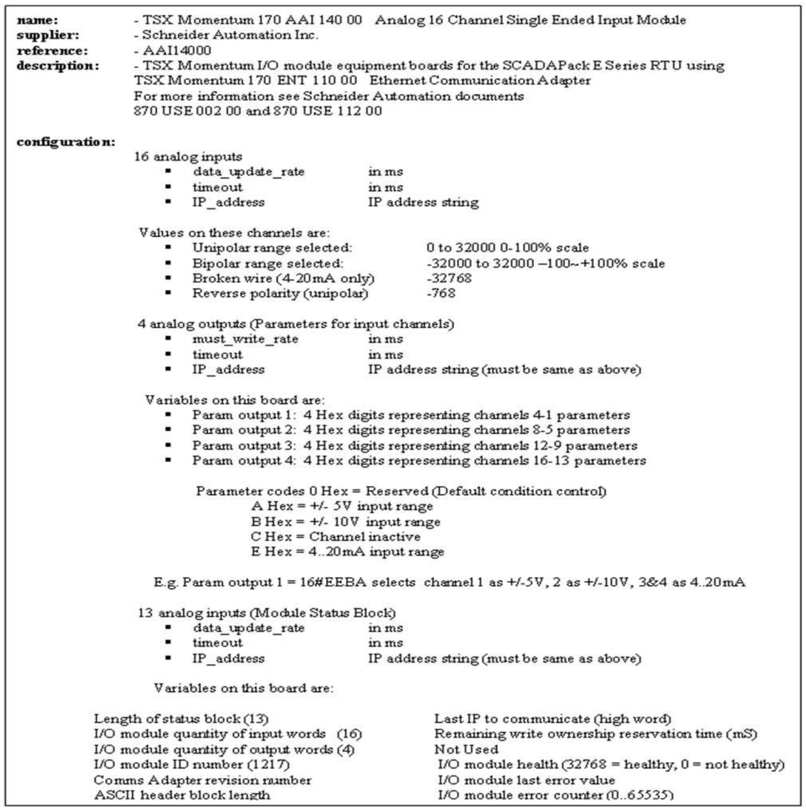
This complex equipment type permits a TSX Momentum Analog 16 Channel Single-Ended Input module to be used as distributed Ethernet I/O on the SCADAPack E RTU.

Using this module requires a BOOTP entry be added to the RTU BOOTP server configuration table. This may be added via the RTU command prompt, or via SCADAPack E Configurator Tool. For more information see Section ***How Do I Change a Modbus/TCP I/O Device*** 65 or the *SCADAPack E Operational Reference* manual.

An IP address needs to be entered for each I/O board's "IP_address" parameter. The value needs to be the same for boards within the same complex equipment type.

Special configuration parameters need to be set up for correct operation of this module. It is recommended that ISaGRAF variables be attached to the "Param" I/O board channels with initial values to configure the input range for the analog input channels on this device. It is suggested for clarity that the attached ISaGRAF output variables be set with a display format "+1A2B" to indicate hexadecimal when viewed. Each Hex digit then represents the configuration mode for 1 Analog Input channel. The configuration digit is specific to this type of module. For clarity it is recommended that a comment field be applied to the ISaGRAF output variable explicitly describing the AI channel parameters.

Figure 8.6 50 details the ISaGRAF Complex Equipment technical note.



```
name:           - TSX Momentum 170 AAI 140 00   Analog 16 Channel Single Ended Input Module
supplier:       - Schneider Automation Inc.
reference:      - AAI14000
description:    - TSX Momentum I/O module equipment boards for the SCADAPack E Series RTU using
                TSX Momentum 170 ENT 110 00   Ethernet Communication Adapter
                For more information see Schneider Automation documents
                870 USE 002 00 and 870 USE 112 00

configuration:
                16 analog inputs
                    ▪  data_update_rate          in ms
                    ▪  timeout                   in ms
                    ▪  IP_address                IP address string

                 Values on these channels are:
                    ▪  Unipolar range selected:          0 to 32000 0-100% scale
                    ▪  Bipolar range selected:           -32000 to 32000 –100-–+100% scale
                    ▪  Broken wire (4-20mA only)         -32768
                    ▪  Reverse polarity (unipolar)       -768

                4 analog outputs (Parameters for input channels)
                    ▪  must_write_rate           in ms
                    ▪  timeout                   in ms
                    ▪  IP_address                IP address string (must be same as above)

                 Variables on this board are:
                    ▪  Param output 1:  4 Hex digits representing channels 4-1 parameters
                    ▪  Param output 2:  4 Hex digits representing channels 8-5 parameters
                    ▪  Param output 3:  4 Hex digits representing channels 12-9 parameters
                    ▪  Param output 4:  4 Hex digits representing channels 16-13 parameters

                      Parameter codes 0 Hex = Reserved (Default condition control)
                                  A Hex = +/- 5V input range
                                  B Hex = +/- 10V input range
                                  C Hex = Channel inactive
                                  E Hex = 4..20mA input range

                  E.g. Param output 1 = 16#EEBA selects  channel 1 as +/-5V, 2 as +/-10V, 3&4 as 4..20mA

                13 analog inputs (Module Status Block)
                    ▪  data_update_rate          in ms
                    ▪  timeout                   in ms
                    ▪  IP_address                IP address string (must be same as above)

                  Variables on this board are:

          Length of status block (13)                  Last IP to communicate (high word)
          I/O module quantity of input words  (16)     Remaining write ownership reservation time (mS)
          I/O module quantity of output words (4)      Not Used
          I/O module ID number (1217)                   I/O module health (32768 = healthy, 0 = not healthy)
          Comms Adapter revision number                 I/O module last error value
          ASCII header block length                     I/O module error counter (0..65535)
```

Figure 8.6: ISaGRAF "aai14000" Complex Equipment Technical Note

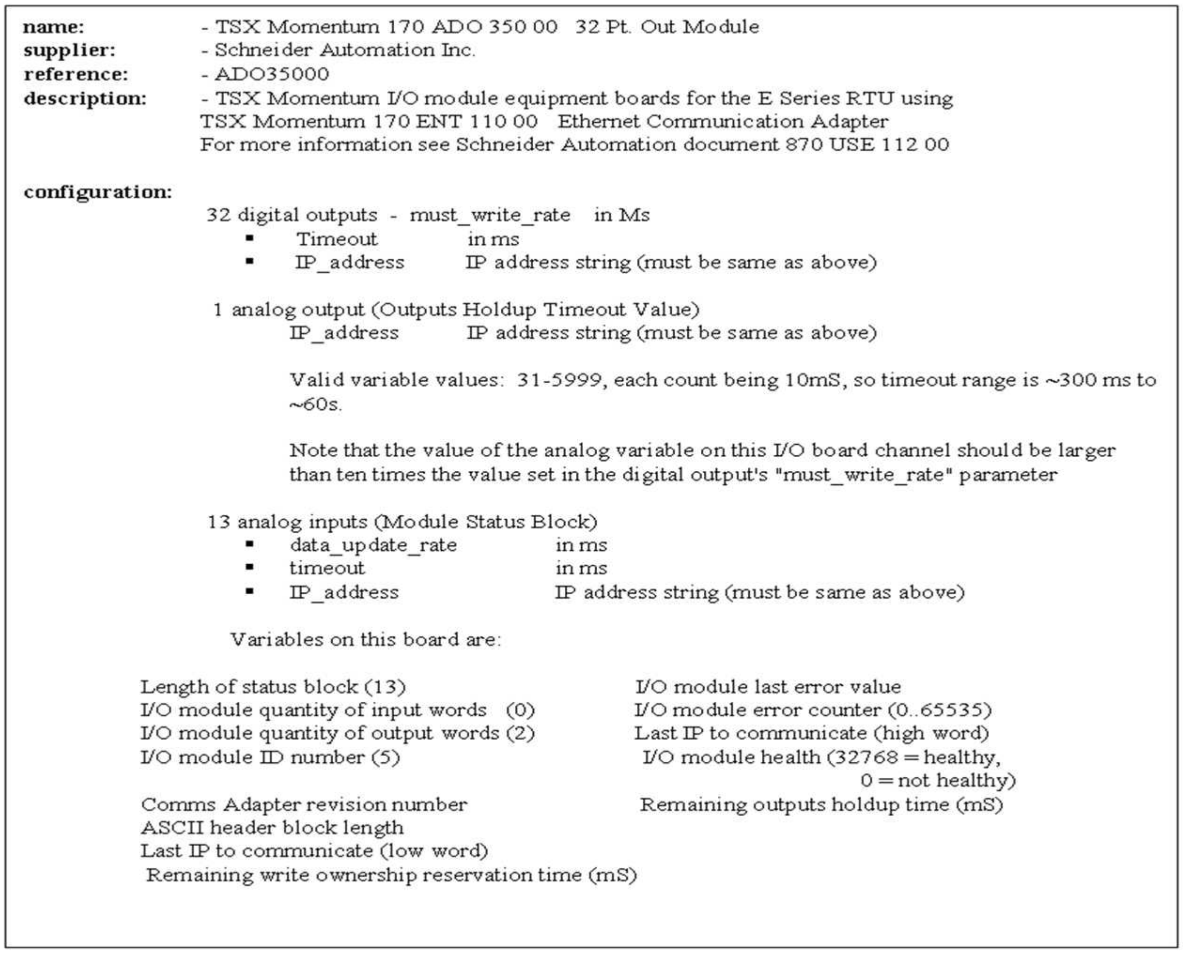
5.6.7 "ado35000" TSX Momentum 170 ADO 350 00 (continue)

This complex equipment type permits a TSX Momentum 32 Point Output module to be used as distributed Ethernet I/O on a SCADAPack E RTU.

Using this module requires a BOOTP entry be added to the RTU's BOOTP server configuration table. This may be added via the RTU command prompt, or via SCADAPack E Configurator. For more information see Section ***How Do I Change a Modbus/TCP I/O Device*** 65 or the *SCADAPack E Operational Reference* manual.

Figure 8.7 51 details the ISaGRAF Complex Equipment Technical Note.

An IP address needs to be entered for each I/O board's "IP_address" parameter. The value needs to be the same for boards within the same complex equipment type.



name: - TSX Momentum 170 ADO 350 00 32 Pt. Out Module
supplier: - Schneider Automation Inc.
reference: - ADO35000
description: - TSX Momentum I/O module equipment boards for the E Series RTU using TSX Momentum 170 ENT 110 00 Ethernet Communication Adapter
For more information see Schneider Automation document 870 USE 112 00

configuration:

32 digital outputs - must_write_rate in Ms
- Timeout in ms
- IP_address IP address string (must be same as above)

1 analog output (Outputs Holdup Timeout Value)
IP_address IP address string (must be same as above)

Valid variable values: 31-5999, each count being 10mS, so timeout range is ~300 ms to ~60s.

Note that the value of the analog variable on this I/O board channel should be larger than ten times the value set in the digital output's "must_write_rate" parameter

13 analog inputs (Module Status Block)
- data_update_rate in ms
- timeout in ms
- IP_address IP address string (must be same as above)

Variables on this board are:

Length of status block (13)
I/O module quantity of input words (0)
I/O module quantity of output words (2)
I/O module ID number (5)
Comms Adapter revision number
ASCII header block length
Last IP to communicate (low word)
Remaining write ownership reservation time (mS)
I/O module last error value
I/O module error counter (0..65535)
Last IP to communicate (high word)
I/O module health (32768 = healthy, 0 = not healthy)
Remaining outputs holdup time (mS)

Figure 8.7: ISaGRAF "ado35000" Complex Equipment Technical Note

5.7 Communications Interface

- *Serial Modbus Communications* 53
- *Modbus/TCP Client Communications* 54
- *Modbus/TCP Server Communications* 55
- *BOOTP Server Configuration* 57

5.7.1 Serial Modbus Communications

When using serial Modbus master communications, the SCADAPack E RTU communicates with the PLC or peripheral devices using RTU serial port(s) configured as '***PLC Device***'.

Each port must be configured to communicate at the same rate and in the same format as the peripheral device(s). For example 9600 bps, 8 data bits, 1 stop bit, and no parity.

The SCADAPack E RTUs will not assert any hardware handshaking lines when communicating using RS232, RS422 or 4-wire RS485 with its Modbus PLC device driver. If the Modbus PLC requires hardware handshaking (e.g. CTS asserted), it must be provided in the cabling to the PLC (as shown above).

When 2-wire RS485 communications is used, the SCADAPack E RTU provides RS485 transmitter/ receiver control internally.

A sample cable configuration for connecting a PLC to a SCADAPack ES RTU RS232 port is shown in ***Figure 9.1***[53].

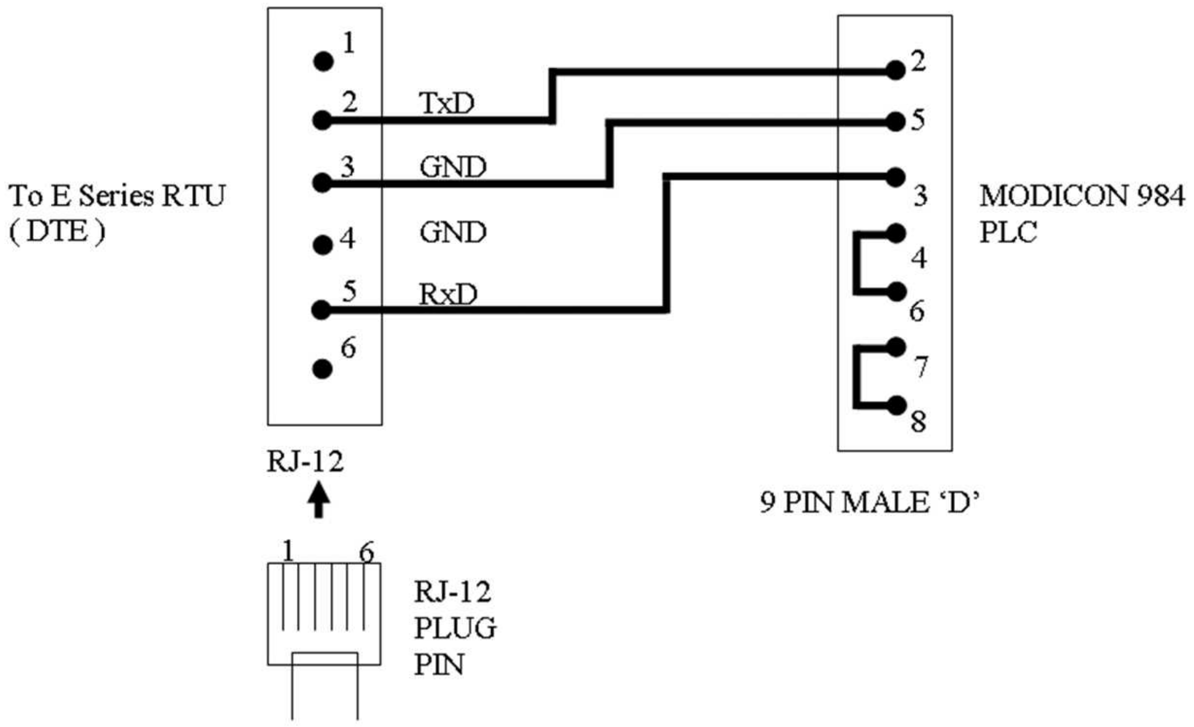


Figure 9.1: SCADAPack ES RTU to PLC Cable

5.7.2 Modbus/TCP Client Communications

Modbus/TCP Client communications is support by the SCADAPack E RTU.

When using Modbus/TCP communications, the RTU communicates with the PLC or peripheral devices using a TCP/IP interface. This may be the RTU's Ethernet interface configured as "***TCP/IP Enable***" or "***TCP/IP + RemIO***" (depending on the RTU model), or a serial port configured as '***PPP-TCP/IP***'.

In addition, the RTU's "IP Services" configuration needs to have "Modbus/TCP Client" service enabled for operation of Modbus/TCP protocol. This configuration can be made on the SCADAPack E Configurator "TCP/IP" page. For more information see the *SCADAPack E TCP/IP Technical Reference* manual.

Enabling Modbus/IP Client service requires the RTU to be restarted in order to start the RTU's PLC Cache task.

The SCADAPack E RTU may connect to any Open Modbus/TCP protocol device including PLCs, I/O modules and Modbus Bridges.

Each Modbus/TCP I/O board specifies an "*IP_address*" parameter that needs to be configured with the address of the Modbus/TCP device that the RTU is communicating with (for that I/O board). It is assumed that the SCADAPack E RTU and the Modbus/TCP device(s) have fixed IP address which are unique on the IP network to which they are connected.

The RTU's Modbus/TCP client attaches to TCP port number "502" in each target Modbus/TCP server device. (Can be changed by advanced ISaGRAF users if required).

Some Modbus/TCP devices expect to obtain their IP addresses from another device on their network rather than being configured with their address, locally. For this purpose, the SCADAPack E RTU supports BOOTP Server capability. BOOTP Server needs to also be enabled in the RTU's "IP Services" configuration to support this capability. See Section ***BOOTP Server Configuration*** 57 for more information.

For further information on connecting to the SCADAPack E RTUs to TCP/IP networks, refer to the *SCADAPack E TCP/IP Reference* manual.

5.7.3 Modbus/TCP Server Communications

Modbus/TCP Server communications is also supported by the SCADAPack E RTU.

When using Modbus/TCP Server communications, the RTU communicates with Modbus/TCP clients using one of its TCP/IP interfaces. This may be the RTU's Ethernet interface configured as "***TCP/IP Enabled***" or "***TCP/IP + RemIO***" (depending on the RTU model), or a serial port configured as '***PPP-TCP/IP***'.

In addition, the RTU's "IP Services" configuration needs to have "**Modbus/TCP Server**" service enabled for operation of Modbus/TCP protocol. This configuration can be made on SCADAPack E Configurator's "TCP/IP" or "Slave / Modbus" page. For more information see the *SCADAPack E TCP/IP Technical Reference* manual.

Enabling Modbus/TCP Server service requires the RTU to be restarted in order to start the RTU's Modbus/TCP Server listening task.

The RTU's Modbus/TCP Server 'listens' on TCP port number "**502**" for any Modbus/TCP client devices attempting to connect.

The SCADAPack E Modbus/TCP server supports a maximum of concurrent clients depending on the controller type.

- For the SCADAPack ER and SCADAPack ES controllers the Open Modbus/TCP server supports a maximum of 20 concurrent client connections.
- For the SCADAPack 300E controllers the Open Modbus/TCP server supports a maximum of 5 concurrent client connections.
- For the 386eNet controllers the Open Modbus/TCP server supports a maximum of 5 concurrent client connections.

An open socket will be closed if there is no activity detected for 120 seconds (see Section ***TCP / Operating System Issues*** 88 for more information regarding the inactivity disconnect timeout).

For further information on connecting SCADAPack E RTUs to TCP/IP networks, refer to the *SCADAPack E TCP/IP Reference* manual.

5.7.4 Modbus RTU in TCP Client Communications

Modbus RTU in TCP Client communications is support by the SCADAPack E RTU.

When using Modbus RTU in TCP communications, the RTU communicates with the PLC or peripheral devices using a TCP/IP interface. This may be the RTU's Ethernet interface configured as "***TCP/IP Enable***" or "***TCP/IP + RemIO***" (depending on the RTU model).

In addition, the RTU's "IP Services" configuration needs to have "Modbus/IP (Client)" service enabled for operation of Modbus RTU in TCP protocol. This configuration can be made on the SCADAPack E Configurator "TCP/IP" page. For more information see the *SCADAPack E TCP/IP Technical Reference* manual.

Enabling Modbus/IP (Client) service requires the RTU to be restarted in order to start the RTU's PLC Cache task.

The SCADAPack E RTU may connect to any Modbus RTU in TCP protocol device including PLCs, I/O modules and Modbus Bridges.

Each Modbus RTU in TCP I/O board specifies an "*IP_address*" parameter that needs to be configured with the address of the Modbus RTU in TCP device that the RTU is communicating with (for that I/O board). It is assumed that the SCADAPack E RTU and the Modbus RTU in TCP device(s) have fixed IP address which are unique on the IP network to which they are connected.

The RTU's Modbus RTU in TCP client attaches to TCP port number "49152" in each target Modbus RTU in TCP server device. (Can be changed by advanced ISaGRAF users if required).

Some Modbus RTU in TCP devices expect to obtain their IP addresses from another device on their network rather than being configured with their address, locally. For this purpose, the SCADAPack E RTU supports BOOTP Server capability. BOOTP Server needs to also be enabled in the RTU's "IP Services" configuration to support this capability. See Section ***BOOTP Server Configuration*** 57 for more information.

For further information on connecting to the SCADAPack E RTUs to TCP/IP networks, refer to the *SCADAPack E TCP/IP Reference* manual.

5.7.5 BOOTP Server Configuration

BOOTP is a TCP/IP application protocol that utilizes UDP socket communications.

The SCADAPack E RTU may be configured to start a BOOTP server by selecting it from the SCADAPack E Configurator's "TCP/IP Services" configuration.

The BOOTP server (SCADAPack E RTU) listens for requests from a BOOTP client (typically an IP device on a LAN).

Typically, the BOOTP client uses its Ethernet MAC address to identify itself, and via *broadcast* IP messages, requests a BOOTP server to configure its parameters.

The SCADAPack E RTU is capable of configuring a BOOTP client's "your-ip" address. However, the RTU will not configure any of the other BOOTP standard or extended fields. (see RFC 951 and later).

SCADAPack E Configurator provides a configuration interface for the SCADAPack E RTU's BOOTP server. The user enters an Ethernet-MAC / IP address pair for each node requiring BOOTP configuration of its IP address.

The SCADAPack E RTU will not answer a BOOTP request from a client node unless there is a corresponding entry in the Ethernet-MAC / IP address table.

Devices refer to their Ethernet-MAC address in different ways, such as "IEEE Global Address". In each case, the Ethernet-MAC address will contain a 12 digit hexadecimal number.

Ethernet-MAC address entry in the SCADAPack E RTU, for BOOTP, may be in any of the following formats (12 hexadecimal digits, case insensitive):

00:1A:2B:3C:4D:5E

00-1A-2B-3C-4D-5E

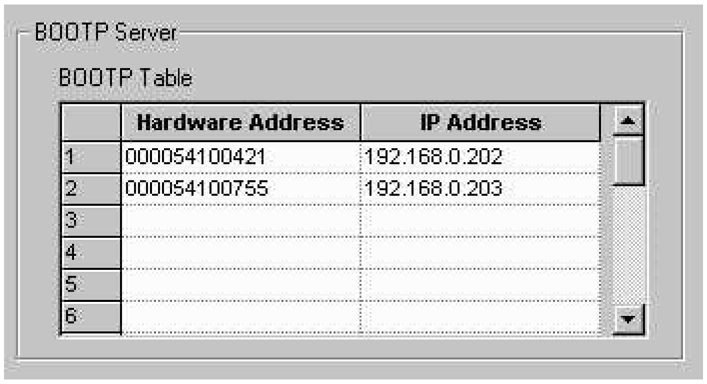
001A2B3C4D5E

IP address entry needs to be in the following format (leading zeroes not required): 192.168.1.97

- ***Configuring BOOTP with SCADAPack E Configurator*** 58
- ***Configuring BOOTP from the Command Line*** 59

5.7.5.1 Configuring BOOTP with SCADAPack E Configurator

The following figure shows the SCADAPack E Configuration of the BOOTP table. This is found on the “Advance TCP/IP” page of SCADAPack E Configurator.



	Hardware Address	IP Address
1	000054100421	192.168.0.202
2	000054100755	192.168.0.203
3		
4		
5		
6		

Figure 9.2: SCADAPack E Configuration of the BootP Table

Enter the BOOTP client details in the table, in the same format as described above.

The “Hardware Address” field is the Ethernet-MAC address in one of the three formats described.

The configured “IP Address” is dowloaded to the peripheral device that has the matching hardware address.

Changes to the BOOTP Table in SCADAPack E Configurator should be followed by “Write RTU Configuration”.

BOOTP Table changes become active in the SCADAPack E RTU immediately (i.e. no need to restart the RTU).

5.7.5.2 Configuring BOOTP from the Command Line

In addition to configuring the BOOTP table via SCADAPack E Configurator, the SCADAPack E RTU command line provides a management command to manipulate the BOOTP server configuration table.

```
C:\> bootp /?
BOOTP protocol loads IP address to remote Ethernet devices
BOOTP command manipulates configuration table
(changes ARE permanent)
Usage:
BOOTP PRINT
BOOTP ADD remote-MAC remote-IP
BOOTP DELETE   [remote-MAC]
               [remote-IP]
```

For example, the following command prints the BOOTP configuration table:

```
C:\> bootp print

BOOTP entries:
Ethernet MAC Addr          IP Addr Loaded  Load Count
00-01-02-03-04-05          192.168.0.242   1
01-02-03-04-05-06          158.234.186.168 2
```

This indicates the configured BOOTP server entries (what IP Address will be loaded to which Ethernet MAC address), and indicates how many times the BOOTP server has sent BOOTP response commands to the appropriate BOOTP client.

The following command ADDS or REPLACES an entry in the BOOTP configuration table:

```
C:\> bootp add 01-02-03-04-AA-BB 158.234.186.168
```

An existing entry with a matching Ethernet MAC address OR matching IP address will be replaced by the new entry. This is typically used if an existing Modbus/TCP or Modbus RTU in TCP device is replaced by another device with a different Ethernet MAC Addr, for example.

The following command REMOVES an entry in the BOOTP configuration table:

```
C:\> bootp del 01-02-03-04-AA-BB
```

Either the Ethernet-MAC address or IP address may be used to specify which BOOTP entry to remove, but the string used in the “del” command needs to exactly match the string in the BOOTP entry in order for the entry to be removed successfully.

Changes made to the BOOTP configuration table via SCADAPack E Configurator or command line are retained in NON-VOLATILE MEMORY not requiring the SCADAPack E RTU to be restarted in order to take effect. However, remember to make a record of the RTU’s configurations after they are modified.

When BOOTP diagnostics are enabled (via TCPDIAG command), the SCADAPack E RTU diagnostic stream indicates when a remote device is configured via BOOTP by the RTU. E.g.

```
BOOTP>>loaded IP: 158.234.186.168  to MAC: 01-02-03-04-AA-BB
```

5.8 System Points

RTU system points are provided to indicate the status of some ISaGRAF I/O boards that are used for Slave I/O communications with peripheral devices such as PLCs.

Where multiple ISaGRAF Slave I/O boards are present in an ISaGRAF application, consecutive, sequential system point pairs are used for the next Slave I/O board, regardless of what PLC port the boards are connected to. Each ISaGRAF kernel is allocated a separate set of system points for Slave I/O boards.

The status for the ISaGRAF Slave I/O boards reported (according to the above rules) has two system points associated with it. The *communications status*, and the *data cache time*.

ISaGRAF Complex Equipment types internally contain ISaGRAF I/O board information. Each I/O board within a complex equipment type is the equivalent to a separate I/O board, and so may also have a pair of system points associated with each I/O board within the complex equipment type.

The *communication status* indicates the status of the communication with the PLC for the data on the I/O board. For more information see Section ***Modbus Status Values*** 62.

The age of the cached data for a slave Input Boards is stored in the *cache time* system point for that board. For more information see Section ***Data Cache Time*** 85.

A separate RTU system point is provided to set the *background update rate* of PLC Output Boards. For more information see Section ***PLC Output Board Default Background Update Rate*** 64.

The RTU Slave I/O board status system points for a user application loaded for ISaGRAF Kernel 1 are as follows:

System Point Description	Point Number	Point Type
ISaGRAF Kernel 1 Slave I/O board 1 communication status	53300	16-bit unsigned integer (read-only)
ISaGRAF Kernel 1 Slave I/O board 1 data cache time	53301	16-bit unsigned integer (read-only)
ISaGRAF Kernel 1 Slave I/O board 2 communication status	53302	16-bit unsigned integer (read-only)
ISaGRAF Kernel 1 Slave I/O board 2 data cache time	53303	16-bit unsigned integer (read-only)
…		
ISaGRAF Kernel 1 Slave I/O board 60 communication status	53418	16-bit unsigned integer (read-only)
ISaGRAF Kernel 1 Slave I/O board 60 data cache time	53419	16-bit unsigned integer (read-only)

The RTU Slave I/O board status system points for a user application loaded for ISaGRAF Kernel 2 are as follows:

System Point Description	Point Number	Point Type
ISaGRAF Kernel 2 Slave I/O board 1 communication status	53422	16-bit unsigned integer (read-only)
ISaGRAF Kernel 2 Slave I/O board 1 data cache time	53423	16-bit unsigned integer (read-only)
ISaGRAF Kernel 2 Slave I/O board 2 communication status	53424	16-bit unsigned integer (read-only)
ISaGRAF Kernel 2 Slave I/O board 2 data cache time	53425	16-bit unsigned integer (read-only)
…		
ISaGRAF Kernel 2 Slave I/O board 14 communication status	53448	16-bit unsigned integer (read-only)
ISaGRAF Kernel 2 Slave I/O board 14 data cache time	53449	16-bit unsigned integer (read-only)

5.8.1 Modbus Status Values

The ***mod.. mbus.. mtcp..*** and ***mrtp..*** I/O board communication status system point values are updated by the SCADAPack E RTU Modbus PLC driver as follows. See ***System Points*** 85 for the system points updated with this status.

Modbus Exception Code	**Status**	**Comment**	**RTU Status**
-	Success	Normal operation	0
0x01	Illegal Function	Slave device does not support requested Modbus function code	103
0x02	Illegal Data Address	Reading or Writing an invalid register address was attempted. This may be returned by RTU's device driver, or by Modbus device	103
0x03	Data Value out of Range	Reported by the Modbus device if register value was outside supported value range and could not be written	108
0x04	Illegal Response Length	A request was rejected by the Modbus device as it would have resulted in a response exceed the maximum allowed size (256 bytes)	102
0x0B	Gateway Target Failed	Gateway reported Modbus device did not respond	104
-	Timeout	The Modbus device did not respond	104
-	Socket connect Failed	Could not connect the TCP socket to a server at the configured IP address	104
-	Invalid Message	The message from the Modbus device was not understood by the RTU	106
other	Unknown Error	An undefined error has occurred	101

5.8.2 Data Cache Time

Data Cache Time

The age of the data in the RTU cache for Modbus PLC and Modbus/TCP Input board data is presented in data 'Cache Time' system points. The cache time is initialized to zero when the ISaGRAF application starts and increases until a successful read occurs, after which time the value is reset to zero.

The system point corresponding to a PLC Device input board may be used by the ISaGRAF application to determine the suitability of using the input data from the input board. (I.e. if the value is too high, then the data is stale and the ISaGRAF application may choose not to use it).

Each Input board has its own data cache time system point. The data cache time system points for Output boards will indicate zero.

5.8.3 PLC Output Board Default Background Update Rate

The following SCADAPack E RTU system point controls the default *background update rate* of all PLC Device Output Boards on the RTU. Where an I/O board's "*must write rate*" parameter is zero, or if the older style PLC ISaGRAF I/O boards (that don't have a *must write rate*) are in use, the SCADAPack E RTU writes ISaGRAF PLC output board variables to the appropriate PLC at this rate. This occurs regardless of whether changes are occurring on the ISaGRAF output variable, or not. The purpose of the "must write" is so RTU output variable values are updated in the PLC.

For example, if the PLC is initialized or replaced, then the output values are re-written by the RTU. Similarly, a Modbus/TCP device may clear its outputs upon no communications unless a periodic write is made to its outputs.

The default value of the *background update rate* is 60 seconds. It may be adjusted by the user or specified in an RTU configuration, and is a non-volatile RTU system point that is retained by the RTU.

Changes in the *background update rate* take effect when an ISaGRAF application is loaded and started, or re-started.

System Point Description	Point Number	Point Type
ISaGRAF PLC Output Board Background Update Rate (Secs)	53420	32-bit unsigned integer

The background updates are disabled by setting the system point value to 0 (zero). This may be used to optimize the PLC Device communications bandwidth where background writes are not appropriate or necessary.

5.9 How Do I Change a Modbus/TCP or Modbus RTU in TCP Device (that uses BOOTP)

Modbus/TCP devices using BOOTP may require SCADAPack E RTU re-configuration if they are replaced.

This will be necessary if the Modbus/TCP device has a different Ethernet hardware address.

See Sections:

- ***Change a Modbus/TCP Device Using SCADAPack E Configurator*** 66
- ***Change a Modbus/TCP Device Using COMMAND LINE*** 67.

If the device does not use BOOTP to configure its own IP address, it needs to be reconfigured with the correct IP address by following the procedure detailed in the device's user manual.

⚠WARNING

Having two or more devices with the same IP address can cause unpredictable operation of your network. Before removing any adapter from service, or adding any adapter, check that there is no possibility of a duplicate address appearing on your network. Failure to observe this precaution can result in injury or equipment damage.

REMEMBER: **After changing the configuration of an RTU, make a permanent record of the RTU's new configuration**

5.9.1 Change a Modbus/TCP or Modbus RTU in TCP Device Using SCADAPack E Configurator

Establish communication with the RTU using SCADAPack E Configurator either locally, or remotely.

To replace an existing device:

- Find the BOOTP table in SCADAPack E Configurator's *Advanced TCP/IP* page.
- Identify the relevant BOOTP entry in the SCADAPack E Configurator's BOOTP Configuration Table.
- Change the entry's Ethernet-MAC address to that of the new device. Use one of the following formats:

  000054A12104 or 00-00-54-A1-21-04 or 00:00:54:A1:21:04

- Write the configuration to the RTU. The BOOTP entry is now active.
- Connect the new device to the network & power it up.

To add a new device:

- Choose the first free BOOTP entry in the SCADAPack E Configurator's BOOTP Configuration Table.
- Add the new device's Ethernet-MAC address to the table. Use one of the following formats:

  000054A12104 or 00-00-54-A1-21-04 or 00:00:54:A1:21:04

- Add the desired IP address for the entry.
- Write the configuration to the RTU. The BOOTP entry is now active.
- Connect the new device to the network & power it up.

5.9.2 Change a Modbus/TCP or Modbus RTU in TCP Device Using COMMAND LINE

Change a Modbus/TCP Device Using COMMAND LINE

The SCADAPack E RTU's command line is available:

Using Telnet, when enabled on the RTU
Using a terminal program plugged into an RTU (e.g. the DIAG port)
Using a terminal program plugged into an RTU's ISaGRAF port, and by pressing <Enter><Enter><Enter>

The SCADAPack E RTU command line can be use to replace an existing Modbus/TCP device BOOTP entry, or add a new Modbus/TCP device BOOTP entry. This is only applicable for devices that use BOOTP as a means of configuring their IP addresses.

If you are replacing an existing device you will need to know the IP address being used by the old device. You will also need to know the new device's Ethernet-MAC address (12-digit hexadecimal number).

If you are adding a new BOOTP entry, you will need to know the desired IP address for the new device as well as its Ethernet-MAC address.

You can check configured BOOTP entries by using the command:

```
bootp print
```

From the SCADAPack E RTU command line, enter the following command to add or change a BOOTP entry:

`bootp add` *Ethernet-MAC-addr IP-address*

For example: `bootp add 01020304AABB 158.234.186.168`

You can re-check the changed BOOTP entries by using:

```
bootp print
```

Check BOOTP Diagnostics

You can check BOOTP operation with a new device by performing the following procedures:

- Enable BOOTP diagnostics & enter diagnostic mode with the commands:

```
tcpdiag enable BOOTP  <enter>
diag
```

Connect the new device to the network & power it up. When the new device sends a BOOTP request for an IP address, the SCADAPack E RTU should display something similar to:

```
BOOTP>>loaded IP: 158.234.186.168  to MAC: 01020304AABB
```

Communication may also be verified by issuing a PING command. For example:

```
ping 158.234.186.168
```

6 RTU as a Modbus Slave / Server

The following sections detail SCADAPack E RTU communication where the RTU is a Modbus Slave and Modbus/TCP Server.

- ***Setting up Modbus Slave / Server*** 68
- ***Modbus Slave and Modbus /TCP Server Implementation Conditions*** 69
- ***System Points*** 85
- ***Diagnostics*** 87

6.1 Setting up Modbus Slave / Server

The following sections document the conditions specific to the native Modbus Slave / Server driver.

Conditions common to the Modbus/TCP Server and the Modbus serial Slave such as conformance classes, mapping of Modbus addresses to the RTU point address space, and the circumstances under which specific response exception codes are generated, are detailed in Section ***Modbus/TCP Server and Modbus Slave Implementation Conditions*** 69 .

The SCADAPack E RTU supports a native Modbus Slave driver which responds to Modbus requests by accessing RTU point data directly.

The native SCADAPack E Modbus Slave driver is only operational if at least one serial port function is set to ***Modbus Slave*** or if ***Modbus/TCP (Server***) is enabled as part of the RTU *TCP/IP Services* selection. Note that the RTU must be restarted to activate a port mode or TCP Services change.

Modbus Slave communications can be independently enabled on multiple serial ports, though the single **Slave Address** is applied to each instance of the Modbus Slave driver (see Section ***Modbus Slave Address*** 86 for details regarding Modbus Slave configuration system points).

Multiple Modbus/TCP server communications sessions are supported (up to 5), though the single Modbus **Unit Identifier** is applied to each Modbus/TCP connection. Also see section ***Modbus/TCP Server Unit Identifier*** 85.

6.2 Modbus Slave and Modbus /TCP Server Implementation Conditions

This following sections detail implementation conditions that are common to both the Modbus/TCP Server and the native Modbus Slave:

- ***Conformance Classes and Function Codes 7 & 8*** 70
- ***Modbus Address Mapping to RTU Point Address Space*** 71
- ***Exception Codes*** 82

Open Modbus/TCP Server

The SCADAPack E RTU Modbus/TCP server is only operational if the RTU's “IP Services” configuration has the “Modbus/TCP Server” service enabled.

This configuration can be made from the SCADAPack E Configurator's “TCP/IP” page or "Slave / Modbus" page.

6.2.1 Conformance Classes & Supported Function Codes

Conformance Classes

The following Modbus conformance classes (and function codes) are supported by the Modbus/TCP Server and the native Modbus Slave.

- Class 0 (function codes 3 and 16)
- Class 1 (function codes 1, 2, 4, 5, 6, and 7)
- Class 2 (function code 15 only).

Function Code 7

This function code allows a client to request the SCADAPack E Modbus server/slave to return an exception status that is stored in a pre-determined range of 8 coils. RTU binary system (scratchpad) points 50000 to 50007 are allocated for this purpose.

- RTU binary point 50000 will map to the least significant bit of the response byte.
- RTU binary point 50007 will map to the most significant bit of the response byte.

Function Code 8

This function code allows a client to request the SCADAPack E Modbus slave to return a response to a "Return Query Data" request. The following functionality is provided:

- Function Code 8 is supported on Modbus Slave serial connections only
- A Function Code 8 request on a Modbus/TCP connection returns Exception response with error code 1 (Illegal function)
- Sub function 00 00 is the only sub function supported for a Modbus function code 8 request
- A response will only be generated to a FC8 Sub-function 00 00 request when the data field size is 2
- The Response to a FC8 Sub-function 00 00 (data field size 2) request is an echo of the request
- A request for Sub functions other than 00 00 returns an Exception response with error code 3 (Illegal Data Value)
- There will be no response to a FC8 Sub-function 00 00 request if the data field size is not 2

6.2.2 Modbus Address Mapping to RTU Point Address Space

This section identifies how the binary and analog Modbus addresses are mapped to the RTU point address space.

The “Modbus address” referenced in this section refers to the Modicon PLC equivalent register address i.e. "protocol address" + 1. The "protocol address" is also referred to as the "reference address".

- SCADAPack E Modbus Slave / Server register addresses are mapped directly to the SCADAPack E RTU database point number in a one to one relationship
- Modbus input register types map to RTU Physical Input point types and RTU Derived point types
- Modbus output register types map to RTU Physical Output point types and RTU Derived point types
- The points used by the SCADAPack E DATA CONCENTRATOR are converted to Physical point types automatically, according to their protocol data type (e.g. DNP3 Static Object Type). Modbus register types map to Data Concentrator points using the same rules as RTU Physical point types. Take care to use the appropriate Modbus request to access the appropriate physical point type.
- For more information see *Binary Addresses* 73 and *Analog Addresses* 75 sections
- The data format presented to Modbus is dependent on the *DNP3 Static Object Type* attribute of the RTU point configuration

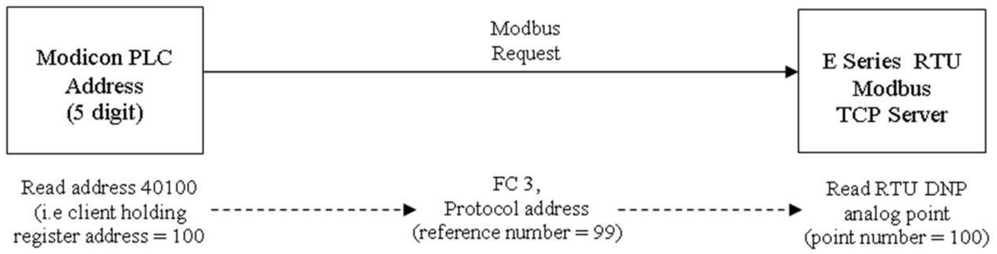
For analog multiple read/writes, this mapping is relevant for the start address only. The mapping of subsequent registers to RTU points is dependent on the RTU configuration point DNP static object types (e.g 16-bit, 32-bit and floating point object types).

As a result of these mapping rules, it is possible to reference a different 32-bit analog point in the RTU with the same Modbus register, based on different reference numbers and word counts of separate Modbus requests.

Section *Modbus Register / 32-bit Analog Point Mapping Configuration* 79 discusses the required configurations for consistent and deterministic mapping of Modbus registers to 32-bit analog points.

See Sections *Reading Analog Registers* 76 and *Writing Analog Registers* 78 for more information on function codes that reference analog points.

This standard mapping is illustrated in the following example using 5 digit addressing


Modicon PLC Address (5 digit)
Modbus Request
E Series RTU Modbus TCP Server
Read address 40100 (i.e client holding register address = 100
FC 3, Protocol address (reference number = 99)
Read RTU DNP analog point (point number = 100)

For clients using 5 digit addressing, the addressable range of both the client register address and the RTU point numbers are from 1 to 9999, and this corresponds to a protocol address (reference number) range from 0 – 9998.

For clients using 6 digit addressing, the addressable range of both the client register address and the RTU point numbers are from 1 to 65535, and this corresponds to a protocol address (reference number) range from 0 – 65534.

The following sections describe how each function code is affected by point mapping.

See Section ***Exception Codes*** 82 for information on how multiple read / write requests are handled when some of the requested addresses are invalid.

- ***Binary Addresses*** 73
- ***Analog Addresses*** 75

6.2.2.1 Binary Addresses

RTU database binary point data is provided to the Modbus serial Slave and Modbus/TCP server in IEC_DISCRETE format. For more information refer to ***Modbus PLC Data Types*** 38.

The following information is applicable to the next sections for Function Code 1 and 2 reads and Function Code 5 and 15 writes:

- when "x" is referenced it means Modbus Address = *x*
- NONE of the requested points exist if for **every** modbus address *x* referenced in the request, no RTU configuration point *x* exists.

Function Code 1: Read Discrete Coils

FC 1 will invoke point reads for every Modbus address *x* referenced in the request, described as follows …

- if NONE of the requested points exist then return ILLEGAL FUNCTION (01)
- if *x* exists in the SCADAPack E RTU as a *Physical Out binary point* or a *Derived binary point* then the current value of point *x* will be read
- The SCADAPack E Data Concentrator converts Derived binary points that present as DNP3 binary output objects into *Physical Out binary points*
- if at least one of the points in the Modbus request does exist, and point *x* does NOT exist, then a zero value will be returned in the response for Modbus register *x*

Function Code 2: Read Discrete Inputs

FC 2 will invoke point reads for every modbus address *x* referenced in the request, described as follows …

- if NONE of the requested points exist then this returns ILLEGAL FUNCTION (01)
- if *x* exists in the SCADAPack E RTU as a *Physical In binary point* or a *Derived binary point* then the current value of point *x* will be read
- The SCADAPack E Data Concentrator converts Derived binary points that present as DNP3 binary input objects into *Physical In binary points*
- if at least one of the points in the Modbus request does exist, and point *x* does NOT exist, then a zero value will be returned in the response for Modbus register *x*

Function Code 5: Preset Discrete Coil

FC 5 will invoke point writes, described as follows …

- if *x* exists in the SCADAPack E RTU as a *Physical Out binary point* or a *Derived binary point* then point *x* will be controlled
- The SCADAPack E Data Concentrator converts Derived binary points that present as DNP3 binary output objects into *Physical Out binary points*
- if point *x* does NOT exist then this returns ILLEGAL FUNCTION (01)

Function Code 15: Write Multiple Coils

FC 15 will invoke point writes for every modbus address *x* referenced in the request, described as follows …

- if *x* exists in the SCADAPack E RTU as a *Physical Out binary point* or a *Derived binary point* then point *x* will be controlled
- The SCADAPack E Data Concentrator converts Derived binary points that present as DNP3 binary output objects into *Physical Out binary points*
- if *x* does NOT exist as a *binary output point* then stop processing request and return ILLEGAL FUNCTION (01)

6.2.2.2 Analog Addresses

This section details the specific mapping for function codes that reference analog addresses.

As noted earlier, it is possible for mixed 16-bit and 32-bit Modbus data value's register mapping to be inconsistent where Modbus registers map to RTU points configured in 32-bit value format. Section ***Modbus Register / 32-bit Analog Point Mapping Configuration*** 79 demonstrates the configurations required for consistent mapping of Modbus registers to 32-bit analog points.

The data types returned in Modbus responses are influenced by the configuration of the **DNP Static Object Type** in each database analog point configuration.

Analog Point Configuration: **DNP3 Static Object Type**	**Modbus Response:** **Number of Registers**	**Modbus data type**
g30 v1 32-bit Analog In g30 v3 32-bit Analog In No Flags	**2**	**IEC DINT** (signed 32-bit integer)
g20 v1 32-bit Counter g20 v5 32-bit Counter No Flags	**2**	**IEC UDINT** (unsigned 32-bit integer)
g30 v5 Short Floating Point (Engineering value)	**2**	**IEC REAL** (32-bit floating point)
g30 v2 16-bit Analog In g30 v4 16-bit Analog In No Flags	**1**	**IEC INT** (signed 16-bit integer)
g20 v2 16-bit Counter g20 v6 16-bit Counter No Flags	**1**	**IEC UINT** (unsigned 16-bit integer)

For more information refer to ***Modbus PLC Data Types*** 38.

The following information is applicable to the next sections for Function Code 3 and 4 reads and Function Code 6 and 16 writes:

- when "x" is referenced it means modbus address = *x*
- corresponding analog database point = *y (*depends on DNP static object type of point. If points referenced are “16 bit analogs” then y = *x)*
- "NONE of the requested points exist" means that for **every** modbus address *x* referenced in the request, no RTU configuration point y exists.

See the following sections for descriptions of individual Modbus function codes and how Modbus register requests map to SCADAPack E point types:

- ***Reading Analog Registers*** 76
- ***Writing Analog Registers*** 78

6.2.2.2.1 Reading Analog Registers

Function Code 3: Read Holding Registers

FC 3 will invoke point reads for every modbus address *x* referenced in the request, described as follows …

- if NONE of the requested points exist then return ILLEGAL FUNCTION (01)
- if y exists in the SCADAPack E RTU as a *Physical Out analog point* or a *Derived analog point* then the current value of point y will be read
- The SCADAPack E Data Concentrator converts Derived analogs points that present as DNP3 analog output objects into *Physical Out analog points*
- if at least one of the points in the Modbus request does exist, and point y does NOT exist, then a zero value will be returned in the response for Modbus register *x*

Function Code 4: Read Input Registers

FC 4 will invoke point reads for every modbus address *x* referenced in the request, described as follows …

- if NONE of the requested points exist then return ILLEGAL FUNCTION (01)
- if y exists in the SCADAPack E RTU as a *Physical In analog point* or a *Derived analog point* then the current value of point y will be read
- The SCADAPack E Data Concentrator converts Derived analogs points that present as DNP3 analog input objects into *Physical In analog points*
- if at least one of the points in the Modbus request does exist, and point y does NOT exist, then a zero value will be returned in the response for Modbus register *x*

Reading Multiple Registers

The register address to RTU database point mapping described in this section relates primarily to the start register address specified in the Modbus request.

The word count included in the request, in conjunction with the DNP static object type of the mapped points, will affect the number of RTU database points included in the response.

The following example illustrates this mapping for function code 3:

Consider the Modbus request as follows (starting from the function code)

… 03 03 e8 00 03

which translates to …"read 3 holding registers at protocol reference number 1000" (41001 in Modicon style register addressing).

The protocol reference number 1000 would therefore map to RTU point number 1001.

Consider the following RTU points configurations:

RTU Analog Point 1001 : DNP static object type → 16 bit analog
RTU Analog Point 1002 : DNP static object type → 32 bit analog

RTU Analog Point 1003 : DNP static object type ➔ 16 bit analog …

This would map RTU analog points to Modbus client holding register addresses as follows

RTU Analog point 1001 maps to client holding register address 1001
RTU Analog point 1002 maps to client holding register addresses 1002 and 1003.

RTU Analog point 1003 would not be included in the response as it exceeds the register length requested.

The same functionality applies for function code 3 and 4.

See *Modbus Register / 32-bit Analog Point Mapping Configuration* 79 for more information on handling more complex combinations of 16 and 32 bit registers.

6.2.2.2.2 Writing Analog Registers

Function Code 6: Write Single Register

FC 6 and FC 16 will invoke point writes for every modbus address *x* referenced in the request, described as follows …

- if y exists in the SCADAPack E RTU as a *Physical Out analog point* or a *Derived analog point* then point y will be controlled
- if point *y* does NOT exist as a *Physical Out analog point* or a *Derived analog point* then processing is stopped ILLEGAL FUNCTION (01) is returned

Function Code 16: Write Multiple Registers

- if y exists in the SCADAPack E RTU as a *Physical Out analog point* or a *Derived analog point* then point y will be controlled
- if point *y* does NOT exist as a *Physical Out analog point* or a *Derived analog point* then processing is stopped ILLEGAL FUNCTION (01) is returned

Writing Multiple Registers

The register address point mapping described above for function code 16, relates primarily to the start register address specified in the modbus request. The word count included in the request, in conjunction with the DNP static object type of the mapped points, will affect the number of RTU points controlled by the request. The following example illustrates this mapping.

Consider the modbus request as follows (starting from the function code)

```
…        10 03 e8 00 03 06 00 08 00 04 00 00
```

which translates to …"write 3 holding registers at protocol reference number 1000" (41001 in Modicon style register addressing).

The protocol reference number 1000 would therefore map to RTU point number 1001. Consider the following RTU points configurations:

RTU Analog point 1001 : DNP static object type ➔ 16 bit analog
RTU Analog point 1002 : DNP static object type ➔ 32 bit analog
RTU Analog point 1003 : DNP static object type ➔ 16 bit analog …

The Modbus request would result in the following controls :

RTU Analog point 1001 is assigned the integer value 8
RTU Analog point 1002 is assigned the integer value 4.

RTU Analog point 1003 is not controlled as the point is outside the register range of the Modbus write request.

See *Modbus Register / 32-bit Analog Point Mapping Configuration* 79 for more information on handling more complex combinations of 16 and 32 bit registers.

6.2.2.3 Modbus Register / 32-bit Analog Point Mapping Configuration

As noted earlier, it is possible to reference a different 32-bit analog point in the RTU with the same modbus register, based on different reference numbers and word counts of separate modbus requests.

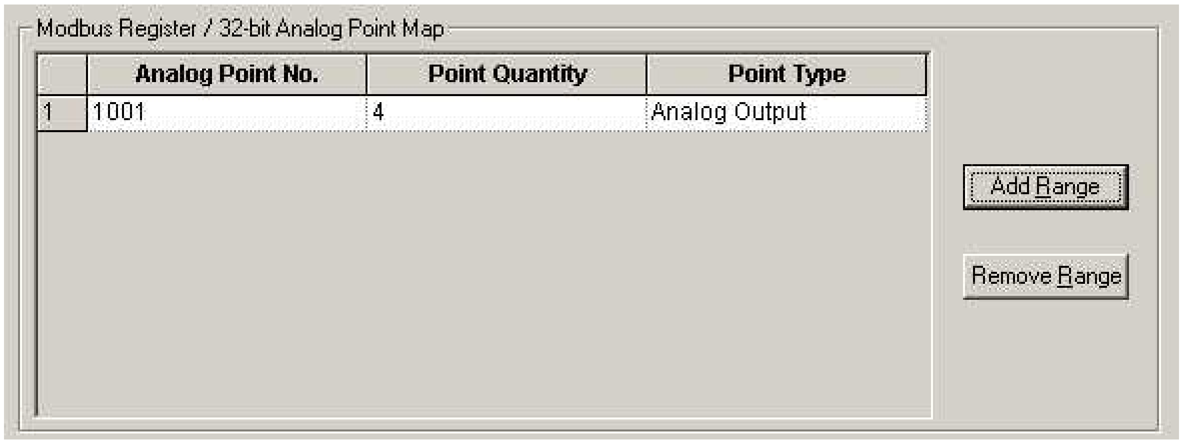
For a consistent and deterministic mapping for 32-bit values point values to Modbus registers for the Modbus/TCP Server and the native Modbus Slave, use the **Modbus Register / 32-bit Analog Point Map** table on the SCADAPack E Configurator **Slave | Modbus** page.

TIP: Add RTU points to this table when you want to access them as 32-bit Modbus registers. (e.g. Long Integers, Short Floating Point values, etc.)

It is strongly recommended that points are allocated consecutively so that the register mapping is grouped together in a single entry in the table. The Point Quantity field indicates how many consecutive points will be read or written as 32-bit Modbus registers.

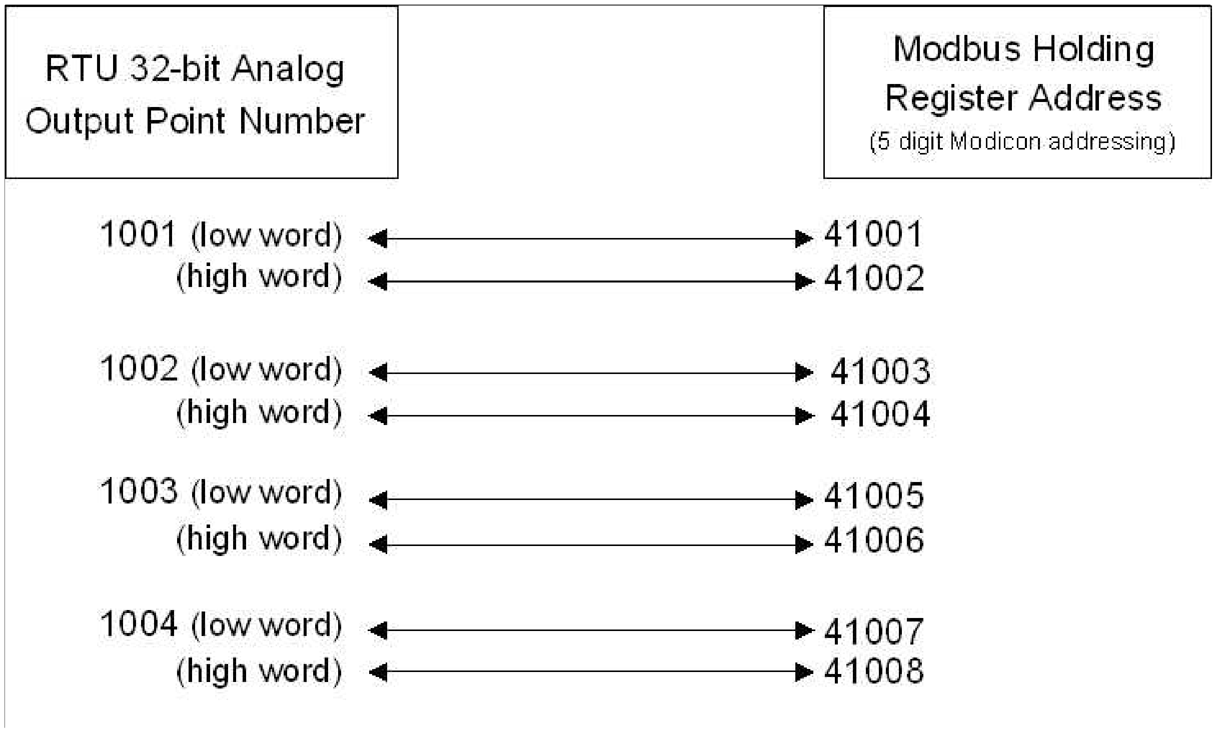
Points accessed as 16-bit Modbus registers should NOT be added to this table.

The following image displays the SCADAPack E Configurator interface, which show an example for RTU analog points 1001-1004 as writeable 32-bit Modbus holding registers.



	Analog Point No.	Point Quantity	Point Type
1	1001	4	Analog Output

This configuration would result in the consistent register / point mapping as shown in the following diagram, irrespective of the protocol reference number (register address) and the word count specified in a Modbus request.



Example: a Modbus request to write to the holding register pair 41005/41006 would result in controlling RTU analog point 1003. Without the additional mapping configuration shown above, this control would have otherwise mapped to analog point 1005.

An attempt to control single holding register 41006 would result in an exception response, as this register maps to the high word of analog point 1003. In general, multiple register requests whose start reference number is the high word of a designated 32-bit analog register pair will be considered invalid and return an exception response.

This configuration can also be directly manipulated in a configuration file using the **MR** table format. Consult the *SCADAPack E Configuration File Format* document for more information regarding the **MR** table format.

6.2.2.4 Supported Data Types

The SCADAPack E RTU Modbus/TCP Server and the native Modbus Slave shall support the IEC61131 data types as described in the Open Modbus/TCP specification.

The following table lists the IEC61131 data type interpreted for function codes 3, 4 and 16.

The interpreted data type is dependant on the DNP static object type of the RTU configuration point (configurable on a per point basis).

Refer to the *SCADAPack E RTU Configuration Technical Reference* manual for more information.

Table 12.1: Mapped Point Data Types

Mapped Point's DNP Static Object Type	**IEC61131 Data Type (returned in Modbus response)**
DNP object Group 1 Var 1 (binary input no status) DNP object Group 1 Var 2 (binary input with status) DNP object Group 10 Var 2 (binary output status)	*DISCRETE* Binary (discrete) data packed into 8-bit values where least significant bit represents low discrete bit numbers.
DNP object Group 30 Var 1 (32-bit analog with status) DNP object Group 30 Var 3 (32-bit analog no status) DNP object Group 40 Var 1 (32-bit analog output status)	*DINT* (signed 32-bit integer value) Bits 15 - 0 of 1st register = bits 15 - 0 of DINT Bits 15 – 0 of 2nd register = bits 31 - 16 of DINT
DNP object Group 30 Var 2 (16-bit analog with status) DNP object Group 30 Var 4 (16-bit analog no status) DNP object Group 40 Var 2 (16-bit analog output status)	*INT* (signed 16-bit integer value) Bits 15 – 0 of register = bits 15 - 0 of INT
DNP object Group 30 Var 5 (short float point with status) DNP object Group 40 Var 3 (short float point output status)	*REAL* (32-bit Intel single precision real) Bits 15 – 0 of first register = bits 15 – 0 of REAL (bits 15 - 0 of significance) Bits 15 – 0 of second register = bits 31 - 16 of REAL (exponent + bits 23-16 of significance)
DNP object Group 20 Var 1 (32-bit counter with status) DNP object Group 20 Var 5 (32-bit counter no status)	*UDINT* (unsigned 32-bit integer value) Bits 15 – 0 of first register = bits 15 - 0 of UDINT Bits 15 – 0 of second register = bits 31-16 of UDINT
DNP object Group 20 Var 2 (16-bit counter with status) DNP object Group 20 Var 6 (16-bit counter no status)	*UINT* (unsigned 16-bit integer value) Bits 15 - 0 of register = bits 15 - 0 of INT

6.2.3 Exception Codes

This section lists some specific circumstances under which response exception codes may be generated. Refer to the Open Modbus/TCP Specification for the full list of exception codes and their descriptions:

- ***Read & Write Multiple Coils/Register Exceptions*** 83
- ***Writes to RTU Point under ISaGRAF Control & Invalid Addresses*** 84

Useful *Exception Codes* are listed in the following table:

Exception Code	Code Description	Comment
0x01	Illegal Function	slave doesn't support function in request
0x02	Illegal Data Address	slave doesn't have register specified in request
0x03	Illegal Data Value	value in request out of range for register in slave
0x04	Illegal Response Length	request would cause response to exceed 256 bytes
0x0A	Gateway Target Device Failed to Respond	returned by Gateway when no response from remote device

6.2.3.1 Read & Write Multiple Coils / Register Exceptions

Read Multiple Coils/Register Exceptions

Requests to read multiple coils/registers will generate a successful response, if at least one of the requested addresses is valid, and the invalid data addresses shall be returned with a zero value. If requested addresses are invalid, the response exception code shall be set to ILLEGAL FUNCTION (01).

Write Multiple Coils/Register Exceptions

Requests to write to multiple coils/registers may generate a successful response only if the requested addresses are valid and controls succeed. If at least one of the requested addresses is invalid or a control does not succeed then the remaining controls in the request are ignored and an exception response is returned with exception code ILLEGAL FUNCTION (01).

6.2.3.2 Exceptions Writing to RTU Points

Writes to RTU Points under ISaGRAF Control

Any requests to write to coils / registers that may be under the exclusive control of the RTU sequencer (ISaGRAF), will generate the response exception code ILLEGAL FUNCTION (01) even if previous controls in the same multiple write request have been successful. Read requests to points under ISaGRAF control will be successful.

Invalid Addresses

Any requests that reference RTU point numbers outside of the range 0-65535 will generate the response exception code ILLEGAL DATA ADDRESS (02). See Section ***Modbus Address Mapping to RTU Point Address Space*** 71 for details on how Modbus addresses map to RTU point numbers.

Bad Point Quality

Any requests that write to an RTU point that has bad quality (e.g. Point Failed quality) will update the point in the RTU database, but generate an exception response with exception code ILLEGAL FUNCTION (01).

6.3 System Points

RTU system points are provided for configuration of Modbus communication parameters.

- ***Modbus/TCP Server Unit Identifier*** 85
- ***Modbus Slave Address*** 86

6.3.1 Modbus/TCP Server Unit Identifier

Data Cache Time

The age of the data in the RTU cache for Modbus PLC and Modbus/TCP Input board data is presented in data ‘Cache Time’ system points. The cache time is initialized to zero when the ISaGRAF application starts and increases until a successful read occurs, after which time the value is reset to zero.

The system point corresponding to a PLC Device input board may be used by the ISaGRAF application to determine the suitability of using the input data from the input board. (I.e. if the value is too high, then the data is stale and the ISaGRAF application may choose not to use it).

Each Input board has its own data cache time system point. The data cache time system points for Output boards indicate zero.

Modbus/TCP Server Unit Identifier

The following SCADAPack E RTU system point determines the configuration value of the Modbus/TCP Server *Unit Identifier*. Responses are sent to Open Modbus/TCP requests that include a unit identifier that matches this configuration value. If the unit identifier included in the request differs from the configuration value, the request is therefore determined to invalid for this RTU, and the socket is closed.

The default value of the *Unit Identifier* is 1. It may be adjusted by the user or specified in an RTU configuration, and is a non-volatile RTU system point that is retained by the RTU. Changes in the *Unit Identifier* take effect when the RTU is restarted.

System Point Description	Point Number	Point Type
Open Modbus/TCP Server Unit Identifier	54038	32-bit unsigned integer

6.3.2 Modbus Slave Address

The following SCADAPack E RTU system point determines the configuration value of the Modbus Slave Address. This applies only to the native Modbus Slave driver. The slave address of the ISaGRAF based Modbus Slave is detailed in the SCADAPack E ISaGRAF Technical Reference.

System Point Description	Point Number	Point Type
Modbus Slave Address	52014	16-bit unsigned integer

The rules that determine how the Modbus Slave driver responds to requests according to the specified slave address are detailed as follows:

- Responses are sent to serial Modbus requests that include a slave address that matches the *Modbus Slave Address* configured value in the RTU.
- If the slave address included in the request is zero, i.e. broadcast address, the Modbus Slave will respond irrespective of the configuration value of the *Modbus Slave Address.*
- If the specified slave address is non-zero AND differs from the configuration value, no response is sent for the request.

The default value of the *Slave Address* is 1. It may be adjusted by the user or specified in an RTU configuration, and is a non-volatile RTU system point that is retained by the RTU. Changes in the *Slave Address* take effect when the RTU is restarted. Valid *Slave Address* configuration values are in the range of 1 – 247.

6.4 Diagnostics

The SCADAPack E RTU indicates configuration or communication diagnostics via Diagnostic Display mode from a Command line session.

Configuration diagnostics are indicated via ISaGRAF I/O board messages if ISaGRAF PLC I/O boards are not opened. These are always displayed when in Diagnostic Display mode (use DIAG command at command prompt).

Communication diagnostics for the Modbus serial and Modbus/TCP drivers are controlled by the PLCDIAG command at the RTU command prompt. The syntax is as follows:

```
PLCDIAG DISABLE filter-name [filter-name…]
PLCDIAG ENABLE  filter-name [filter-name…]
```

Where filter name is one, or more of the following combinations:

`*`	all Modbus diagnostic messages
`TX`	transmit packet bytes display for Modbus Master / Client Indicating transmitted <--Modbus Master- data by Or <--Modbus/TCP Client -
`RX`	receive packet bytes display Indicating received data – Modbus Master--> by Or – Modbus/TCP Client-->
`COMMS_ERROR`	communication error diagnostics Including timeout and TCP socket connection information
`PLC_ERROR`	error diagnostics on error messages returned by the PLC
`ISAGRAF`	rx/tx packet diagnostics for RTU “ISaGRAF” serial port – not applicable to PLC Cache operation (see the *SCADAPack E ISaGRAF Technical Reference*)
`Modbus (ISaGRAF)`	rx/tx packet diagnostics for RTU “Modbus” serial port – not applicable to PLC Cache operation (see the *SCADAPack E ISaGRAF Technical Reference*)
`MODTCP_SRV`	rx/tx packet diagnostics for RTU Open Modbus/TCP Server
`MOD_SLAVE`	rx/tx packet diagnostics for RTU native Modbus Slave

Multiple filters may be specified at the same time with the PLCDIAG command. Use the command line DIAG command to enter the SCADAPack E RTU Diagnostic Display mode after the filters are set.

E.g.

```
PLCDIAG DISABLE TX RX
PLCDIAG ENABLE COMMS_ERROR PLC_ERROR
DIAG
```

7 Modbus/TCP Operation

Modbus / TCP Operating Constraints

The SCADAPack E Modbus/TCP server listens on port 502 for any incoming connections. On detecting an incoming connection a new task is created to handle the client connection. A new socket will be opened with an inactivity timeout of 120 seconds (2 minutes). The inactivity timer is re-triggered each time a Modbus/TCP request is received.

Conventionally, following a successful transaction, it will be the client that closes the connection, though if the inactivity timer expires with the socket still open, the SERVER in the SCADAPack E RTU will close the connection. On disconnection, the task created to handle this transaction will be destroyed.

If the Modbus/TCP Server receives only part of a message, a shorter inactivity timeout of 30 seconds will be applied.

The SCADAPack E Modbus/TCP server supports a maximum of concurrent clients depending on the controller type.

- For the SCADAPack ER and SCADAPack ES controllers the Open Modbus/TCP server supports a maximum of 20 concurrent client connections.
- For the SCADAPack 300E controllers the Open Modbus/TCP server supports a maximum of 5 concurrent client connections.
- For the 386eNet controllers the Open Modbus/TCP server supports a maximum of 5 concurrent client connections.

Open Modbus/TCP Socket Communication

Open Modbus/TCP communications are initiated by the client (e.g. SCADAPack E RTU, SCADA master station, etc.).

The client opens a TCP socket on a Modbus/TCP Server (e.g. PLC, I/O block, Gateway, Bridge).

The socket is associated with the configured IP address of the server, using assigned TCP port number "**502**".

Open Modbus/TCP protocol packets are exchanged between the client and the server across the open TCP socket.

Open Modbus/TCP Client Procedures

A communication disruption causes the socket to be disconnected by the client.

Before the client sends a new request, it attempts to open a new socket at the assigned port number on the server.

Open Modbus/TCP Server Procedures

The Open Modbus/TCP Server may disconnect a connected client (closing the socket) under the following conditions

- error detected - invalid header (i.e. does not conform to Open Modbus/TCP Specification).
- error detected - invalid message (e.g. length differs to that specified in header, etc.).
- requested Unit Identifier differs from RTU configuration value
- inactivity timeout (see Modbus/TCP Operating Constraints 88 for more information regarding

inactivity timeout).

- RTU orderly shutdown (e.g. remote RTU restart received) disconnects connected clients and disconnects from servers.

8 Modbus RTU in TCP Operation

Modbus RTU in TCP Socket Communication

Modbus RTU in TCP communications are initiated by the client (e.g. SCADAPack E RTU).

The client opens a TCP socket on a Modbus RTU in TCP Server (e.g. PLC, I/O block, Gateway, Bridge).

The socket is associated with the configured IP address of the server, using assigned TCP port number "**49152**".

Modbus RTU in TCP protocol packets are exchanged between the client and the server across the open TCP socket.

Modbus RTU in TCP Client Procedures

A communication disruption causes the socket to be disconnected by the client.

Before the client sends a new request, it attempts to open a new socket at the assigned port number on the server.

9 Modbus Protocol Technical Information

The following sections detail the framing of the various Modbus data communication protocols used by the RTU:

- ***Modbus Serial Communication Format*** 92
- ***CRC16 Calculation Method*** 93
- ***Open Modbus/TCP Communication Format*** 93

Modbus application layer protocol is used by both serial Modbus, and Open Modbus/TCP.

Details of the application layer protocol may be found in readily available Modbus or Open Modbus/TCP documentation.

9.1 Modbus Serial Communication Format

The basic frame structure for Modbus RTU serial protocol is as follows:

❖ Request

Slave ID	Function Code	Function dependent request data ….	CRC16 (msb)	CRC16 (lsb)

Maximum request frame size 256 bytes.

❖ Response

Slave ID	Function Code	Function dependent response data ….	CRC16 (msb)	CRC16 (lsb)

The *Slave ID* of the request is returned in the Response. The *Function Code* of the request is returned in the response if there were no errors. An error response has the most significant bit of the request function code set on (see Error Response). Maximum response frame size 256 bytes.

❖ Error Response

Slave ID	Function Code	Exception Code	CRC16 (msb)	CRC16 (lsb)

The *Slave ID* of the request is returned in the Response. The *Function Code* in an error response has the most significant bit of the request function code set on. I.e. Error Function Code = Request Function Code + 0x80. Maximum response frame size 256 bytes.

Useful *Exception Codes* are:

0x01 = Illegal Function	(slave doesn’t support function in request)
0x02 = Illegal Data Address	(slave doesn’t have register specified in request)
0x03 = Illegal Data Value	(value in request out of range for register in slave)
0x04 = Illegal Response Length	(request would cause response to exceed 256 bytes)

9.2 CRC16 Calculation Method

CRC checking is only performed for Modbus serial communications. Two CRC check codes are appended to the end of both Modbus request and reply messages.

The CRC method used is a standard CRC-16 with the following polynomial:

G(x) = x16 + x15 + x2 + x1

Starting Value = FFFFH

Feedback = A001H

The CRC is calculated using the body and header of the message (i.e. whole message excluding CRC bytes).

9.3 Open Modbus/TCP Communication Format

The basic frame structure for Open Modbus/TCP protocol is as follows. This is the stream data transported via the TCP socket connection and does not include TCP/IP protocol bytes.

❖ Request

Transaction ID (msb) 0*	Transaction ID (lsb) 0*	Protocol ID (msb) 0	Protocol ID (lsb) 0	Length (msb) 0	Length (lsb)	
Unit ID	Function Code	Function dependent request data ….				

Transaction ID is echoed by the Modbus/TCP server and may be used by a client. The SCADAPack E RTU sets these bytes to 0 in requests.

Protocol ID identifies the message protocol following in the data stream. When both these bytes are 0, it indicates Modbus/TCP protocol.

Length (lsb) indicates the number of bytes following in the rest of the frame. Minimum value is 3, maximum value is 255.

Unit ID uniquely identifies the Modbus device, and is equivalent to Slave ID of serial Modbus.

CRC checking is not used for Open Modbus/TCP communications. Instead it relies on TCP/IP stack layers to provide error free transmission of data.

Function Code and following data is equivalent to serial Modbus RTU protocol.

❖ Response

Transaction ID (msb)	Transaction ID (lsb)	Protocol ID (msb) 0	Protocol ID (lsb) 0	Length (msb) 0	Length (lsb)	
Unit ID	Function Code	Function dependent response data ….				

Transaction ID is echoed by the Modbus/TCP server in the response.

Length (lsb) indicates the number of bytes following in the rest of the frame. Minimum value is 3, maximum value is 255.

Unit ID is equivalent to Slave ID of serial Modbus.

❖ Exception Response

Transaction ID (msb)	Transaction ID (lsb)	Protocol ID (msb) 0	Protocol ID (lsb) 0	Length (msb) 0	Length (lsb) 3
Unit ID	Function Code	Exception Code			

Transaction ID is echoed by the Modbus/TCP server in the response

The *Unit ID* of the request is returned in the Response.

The *Function Code* in an exception response has the most significant bit of the request function code set on, i.e. Exception Function Code = Request Function Code + 0x80

See ***Exception Codes*** 82 for a list of useful *Exception Codes.*

9.4 Modbus RTU in TCP Communication Format

Modbus RTU in TCP message format is exactly same as that of the Modbus RTU protocol. The main difference is that Modbus RTU in TCP protocol communicates with a device through the Internet and Modbus RTU protocol through the serial port. The Modbus RTU in TCP protocol does not include a six-byte header prefix, as with the Modbus\TCP, but does include the Modbus CRC16 check fields. The Modbus RTU in TCP message format supports Modbus RTU message format.

The basic frame structure for Modbus RTU serial protocol is as follows:

❖ Request

Slave ID	Function Code	Function dependent request data ….	CRC16 (msb)	CRC16 (lsb)

Maximum request frame size 256 bytes.

❖ Response

Slave ID	Function Code	Function dependent response data ….	CRC16 (msb)	CRC16 (lsb)

The *Slave ID* of the request is returned in the Response. The *Function Code* of the request is returned in the response normally. An exception response has the most significant bit of the request function code set on (see Error Response). Maximum response frame size 256 bytes.

❖ Exception Response

Slave ID	Function Code	Exception Code	CRC16 (msb)	CRC16 (lsb)

The *Slave ID* of the request is returned in the Response. The *Function Code* in an exception response has the most significant bit of the request function code set on. I.e. Exception Function Code = Request Function Code + 0x80. Maximum response frame size 256 bytes.

Useful *Exception Codes* are:

0x01 = Illegal Function	(slave doesn't support function in request)
0x02 = Illegal Data Address	(slave doesn't have register specified in request)
0x03 = Illegal Data Value	(value in request out of range for register in slave)
0x04 = Illegal Response Length	(request would cause response to exceed 256 bytes)

Modbus RTU in TCP protocol has some differences from the Modbus/TCP protocol as follows.

1. Modbus RTU in TCP doesn't include a six-byte header.
2. Modbus RTU in TCP includes a two-byte CRC .
3. The message format supports Modbus RTU message.