Quantum using EcoStruxure™ Control Expert

140 ESI 062 10 ASCII Interface Module

User Manual

(Original Document)

12/2018

35012595.06

www.schneider-electric.com

Schneider Electric

The information provided in this documentation contains general descriptions and/or technical characteristics of the performance of the products contained herein. This documentation is not intended as a substitute for and is not to be used for determining suitability or reliability of these products for specific user applications. It is the duty of any such user or integrator to perform the appropriate and complete risk analysis, evaluation and testing of the products with respect to the relevant specific application or use thereof. Neither Schneider Electric nor any of its affiliates or subsidiaries shall be responsible or liable for misuse of the information contained herein. If you have any suggestions for improvements or amendments or have found errors in this publication, please notify us.

You agree not to reproduce, other than for your own personal, noncommercial use, all or part of this document on any medium whatsoever without permission of Schneider Electric, given in writing. You also agree not to establish any hypertext links to this document or its content. Schneider Electric does not grant any right or license for the personal and noncommercial use of the document or its content, except for a non-exclusive license to consult it on an "as is" basis, at your own risk. All other rights are reserved.

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.

Table of Contents

Safety Information 5
About the Book 9
Chapter 1 140 ESI 062 10 Hardware Description 11
Presentation 12
LED Indicators 13
External Connectors and Switches 15
Specifications 17
Chapter 2 Quantum Addressing Modes 21
Flat Addressing—800 Series I/O Modules 22
Topological Addressing—800 Series I/O Modules with Control Expert 23
Addressing Example 24
Discrete I/O Bit Numbering 25
Addressing the 140 ESI 062 10 Module 26
Chapter 3 Configuration Overview 27
140 ESI 062 10 Configuration 28
ASCII Message Formats 30
Data Flow 36
Parameter Configuration 40
Chapter 4 ESI Command Line Editors 43
Configuration Editor 44
ASCII Message Editor 47
Chapter 5 ESI Commands 49
Overview on ESI Commands 50
ESI Command Word 51
Command Processing 52
Command 0 - NO OPERATION 54
Command 1- READ ASCII MESSAGE 55
Command 2 - WRITE ASCII MESSAGE 57
Command 3 - GET DATA (Module to Controller) 60
Command 4 - PUT DATA (Controller to Module) 62
Command 5 - GET TOD (Time of Day) 64
Command 6 - SET TOD (Time of Day) 66
Command 7 - SET MEMORY REGISTERS 69
Command 8 - FLUSH BUFFER 71
Command 9 - ABORT 72

Command A - GET BUFFER STATUS 73
Response Structure for Illegal Commands 74
Module Status Word (Word 11) 75
Reading beyond Valid Register Range 77
Appendices 79
Appendix A Character Set 81
ASCII Character Set 81
Appendix B Introduction to ESI 062 10 85
Introduction to ESI Module 86
Application Criteria 87
Module Description 88
ESI Module Block Diagram 90
Index 91

Safety Information

Important Information

NOTICE

Read these instructions carefully, and look at the equipment to become familiar with the device before trying to install, operate, service, 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 a hazardous situation which, if not avoided, **will result in** death or serious injury.

⚠ WARNING

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

⚠ CAUTION

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

NOTICE

NOTICE is used to address practices not related to physical injury.

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 its 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.

⚠ WARNING

UNGUARDED EQUIPMENT

- Do not use this software and related automation equipment on equipment which does not have point-of-operation protection.
- Do not reach into machinery during operation.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

This automation equipment and related software is used to control a variety of industrial processes. The type or model of automation equipment suitable for each application will vary depending on factors such as the control function required, degree of protection required, production methods, unusual conditions, government regulations, etc. In some applications, more than one processor may be required, as when backup redundancy is needed.

Only you, the user, machine builder or system integrator can be aware of all the conditions and factors present during setup, operation, and maintenance of the machine and, therefore, can determine the automation equipment and the related safeties and interlocks which can be properly used. When selecting automation and control equipment and related software for a particular application, you should refer to the applicable local and national standards and regulations. The National Safety Council's Accident Prevention Manual (nationally recognized in the United States of America) also provides much useful information.

In some applications, such as packaging machinery, additional operator protection such as point-of-operation guarding must be provided. This is necessary if the operator's hands and other parts of the body are free to enter the pinch points or other hazardous areas and serious injury can occur. Software products alone cannot protect an operator from injury. For this reason the software cannot be substituted for or take the place of point-of-operation protection.

Ensure that appropriate safeties and mechanical/electrical interlocks related to point-of-operation protection have been installed and are operational before placing the equipment into service. All interlocks and safeties related to point-of-operation protection must be coordinated with the related automation equipment and software programming.

NOTE: Coordination of safeties and mechanical/electrical interlocks for point-of-operation protection is outside the scope of the Function Block Library, System User Guide, or other implementation referenced in this documentation.

START-UP AND TEST

Before using electrical control and automation equipment for regular operation after installation, the system should be given a start-up test by qualified personnel to verify correct operation of the equipment. It is important that arrangements for such a check be made and that enough time is allowed to perform complete and satisfactory testing.

⚠ WARNING

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 death, serious 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 temporary grounds that are not 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 all temporary grounds 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.

About the Book

At a Glance

Document Scope

This documentation explains the installation and usage of the ASCII interface module.

Validity Note

This documentation is valid for EcoStruxure™ Control Expert 14.0or later.

The technical characteristics of the devices described in the present document also appear online. To access the information online:

Step	Action
1	Go to the Schneider Electric home page *www.schneider-electric.com*.
2	In the **Search** box type the reference of a product or the name of a product range. ● Do not include blank spaces in the reference or product range. ● To get information on grouping similar modules, use asterisks (*).
3	If you entered a reference, go to the **Product Datasheets** search results and click on the reference that interests you. If you entered the name of a product range, go to the **Product Ranges** search results and click on the product range that interests you.
4	If more than one reference appears in the **Products** search results, click on the reference that interests you.
5	Depending on the size of your screen, you may need to scroll down to see the data sheet.
6	To save or print a data sheet as a .pdf file, click **Download XXX product datasheet**.

The characteristics that are presented in the present document should be the same as those characteristics that appear online. In line with our policy of constant improvement, we may revise content over time to improve clarity and accuracy. If you see a difference between the document and online information, use the online information as your reference.

Related Documents

Title of documentation	Reference number
EcoStruxure™ Control Expert, Program Languages and Structure, Reference Manual	35006144 (English), 35006145 (French), 35006146 (German), 35013361 (Italian), 35006147 (Spanish), 35013362 (Chinese)
Quantum using EcoStruxure™ Control Expert, Hardware Reference Manual	35010529 (English), 35010530 (French), 35010531 (German), 35013975 (Italian), 35010532 (Spanish), 35012184 (Chinese)
Quantum using EcoStruxure™ Control Expert, Discrete and Analog I/O, Reference Manual	35010516 (English), 35010517 (French), 35010518 (German), 35013970 (Italian), 35010519 (Spanish), 35012185 (Chinese)
Quantum using EcoStruxure™ Control Expert, Experts and Communication, Reference Manual	35010574 (English), 35010575 (French), 35010576 (German), 35014012 (Italian), 35010577 (Spanish), 35012187 (Chinese)
Grounding and Electromagnetic Compatibility of PLC Systems, Basic Principles and Measures, User Manual	33002439 (English), 33002440 (French), 33002441 (German), 33003702 (Italian), 33002442 (Spanish), 33003703 (Chinese)
Communication Services and Architectures, Reference Manual	35010500 (English), 35010501 (French), 35006176 (German), 35013966 (Italian), 35006177 (Spanish), 35012196 (Chinese)

You can download these technical publications and other technical information from our website at *www.schneider-electric.com/en/download*.

Chapter 1
140 ESI 062 10 Hardware Description

Introduction

This chapter describes the hardware features of the 140 ESI 062 10 ASCII interface module. Product specifications are included at the end of the chapter.

What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
Presentation	12
LED Indicators	13
External Connectors and Switches	15
Specifications	17

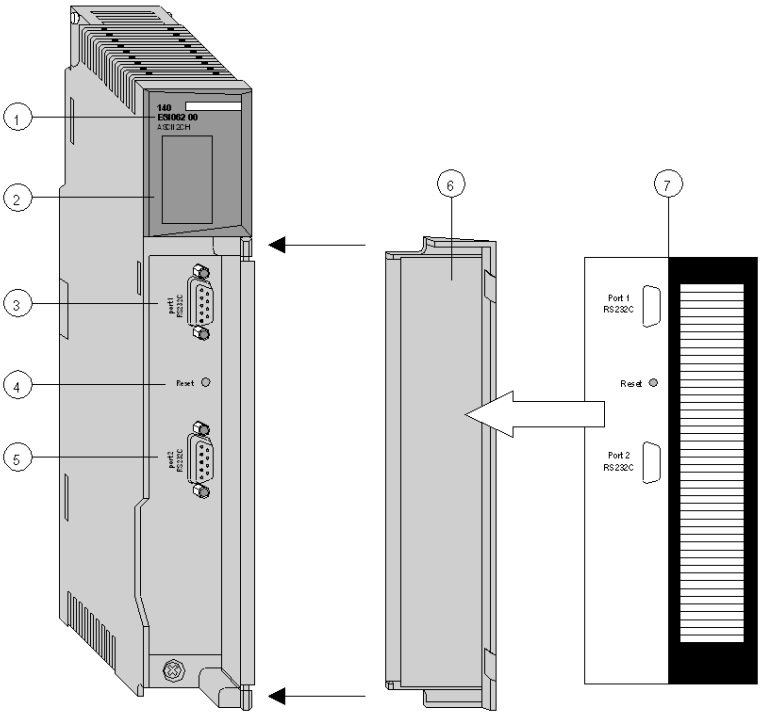
Presentation

Function

The 140 ESI 062 10 module is a Quantum communications interface module used to input messages and/or data from an ASCII device to the CPU, output messages and/or data from the CPU to an ASCII device, or bi directionally exchange messages and/or data between an ASCII device and the CPU.

Illustration

The following figure shows the140 ESI 062 10 module and its components.

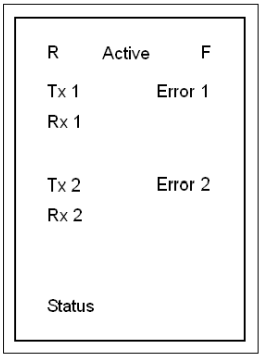


1 Model Number, Module Description, Color Code
2 LED Display
3 Port 1 Connector
4 Reset Button
5 Port 2 Connector
6 Removable door
7 Customer Identification Label (Fold label and place it inside door)

LED Indicators

LED Display Location

The LED display contains 10 indicators located on the top front of the 140 ESI 062 10 module.



Indications

The following table describes the indications when the LEDs are ON.

LEDs	Color	Indication
R	Green	The module has passed power-up diagnostics.
Active	Green	Bus communication is present.
F	Red	The module has detected a fault.
RX1	Green	Received data on RS-232 Port 1
TX1	Green	Transmitted data on RS-232 Port 1
RX2	Green	Received data on RS-232 Port 2
TX2	Green	Transmitted data on RS-232 Port 2
Status	Yellow	Status
Error 1	Red	There is an error condition on Port 1
Error 2	Red	There is an error condition on Port 2

Blink Sequences

The **F**, **Status**, **Error 1**, and **Error 2** LEDs can blink in sequence to indicate the following conditions:

F	Status	Error 1	Error 2	Condition
Blinking	Blinking	Blinking	Blinking	The ASCII module is initializing
				First power-up
OFF	ON	OFF	OFF	Programming mode
OFF	OFF	ON	N/A	Serial Port 1 has incurred a buffer overrun
OFF	OFF	N/A	ON	Serial Port 2 has incurred a buffer overrun
N/A	Blinking (see crash codes)	OFF	OFF	The module is in kernel mode and may have an error

Crash Code Indications

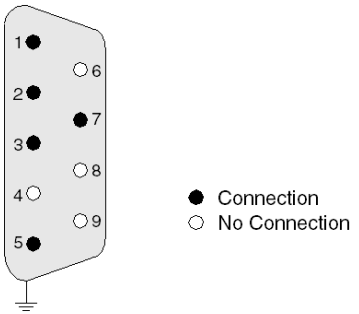
The **Status** LED blinks in various patterns to indicate the module's crash codes.

Number of Blinks	Code (in hex)	Error
Steady ON	0000	Requested kernel mode
4	6631	Bad microcontroller interrupt
5	6503	RAM address test error
8	6402	RAM data test error
7	6300	PROM checksum error (EXEC not loaded)
	6301	PROM checksum error
	630A	Flash message checksum error
	630B	Executive watchdog timeout error
8	8000	Kernel other error
	8001	Kernel PROM checksum error
	8002	Flash program error
	8003	Unexpected executive return

External Connectors and Switches

RS-232 Serial Ports

The ASCII module has two RS-232 serial ports which it uses to communicate with serial devices.



The following are the pinout connections for the two serial ports:

Pin	Signal Name	Description
1	DCD	Carrier Detect
2	RXD	Receive Data
3	TXD	Transmit Data
4	N/A	Not Connected
5	GND	Signal Ground
6	N/A	Not Connected
7	RTS	Request to Send
8	N/A	Not Connected
9	N/A	Not Connected
Shield	N/A	Chassis Ground

Programming Port

Port 1 can also be used as the programming port (port 0). Programming mode is entered by holding down the **Reset** button for more then 4 sec. In programming mode, the serial port is set to a standard terminal communications configuration.

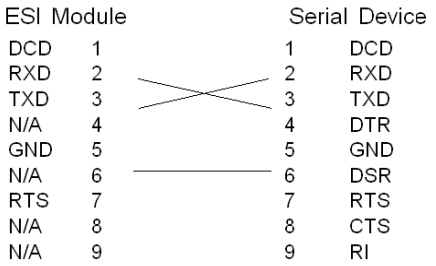
The port uses the following parameters in programming mode:

Parameter	Value
Baud rate	9600
Data bits	8
Stop bits	1
Parity bit	None (disabled)
Keyboard Mode	ON (character echo)
XON/XOFF	ON

The port configuration has been set this way so that it is a known configuration; it may or may not be the same configuration that is used when the module is running.

Minimum Cable Layout

The minimum required cable layout to connect the ESI module either to an external device or a programming terminal (PC) is shown in the following illustration:



Reset Push Button

A recessed push button is located on the front of the module. This **Reset** button has two functions:

- Reset the module by a short press
- Enter programming mode by holding the button down for more than 4 sec

Specifications

Data Interface

Data Interface

RS-232	2 serial ports (9-pin D-shell), non-isolated
Cabling (Maximum cable length 20 m shielded)	990 NAA 263 20, Modbus Programming Cable, RS 232, 12 ft (2.7 m)
	990 NAA 263 50, Modbus Programming Cable, RS 232, 50 ft (15.5 m)

Firmware

Firmware Specifications

Port Performance	Burst Speed: Continuous Speed:	19.2 k baud each port Application dependent
Depth of Nested Messages	8	
Buffer Size	255 Input 255 Output	
Number of Messages	255	
Maximum Message Length	127 characters plus 1 checksum	

Memory

Memory Specifications

RAM	256 kb for data and program + 2 kb dual port ram
Flash-ROM	128 kb for program and firmware

Power

Power Specifications

Power Dissipation	2 W max
Bus Current Required	300 mA

Fuses

Required Fuses

Internal	None
External	User discretion

I/O) Mapping

Required Addresses

In	12 Words
Out	12 Words

Compatibility

Compatibility

Programming Software	Concept 2.5 or higher, ProWorx NxT, ProWorx 32, Modsoft, Control Expert
Data Formats Supported	Text, Decimal, Fixed Point, Nested Write Message, Set Register Pointer, Print Time/Date, Repeat, Space, Newline, Control Code, Flush Buffer
Quantum Controllers	All, Executive V2.0 at a minimum
Battery Backup Module	140 XCP 900 00

Mechanical

Mechanical

IWeight	1 kg max
Dimensions (H x D x W)	250 mm x 103.85 mm x 40.34 mm
Material	(Enclosures and Bezels) Lexan
Space Requirements	1 backplane slot

Electrical

Electrical

RFI Immunity (IEC 1000-4-3)	27 ... 500 MHz, 10 V/m
Electrostatic Discharge (IEC 1000-4-2)	8 kV air / 4 kV contact
Fast Transients (IEC 1000-4-4)	0.5 kV common mode
Damped Oscillatory Transients	1 kV common mode 0.5 kV differential mode
Surge Withstand Capability (Transients) (IEC 1000-4-5)	1 kV common mode 0.5 kV differential mode

Environmental Conditions

Environmental Conditions for Operation

Temperature	0 ... 60°C (32 ... 140°F)
Humidity	0 ... 95% RH noncondensing @ 60°C
Chemical Interactions	Enclosures and bezels are made of Lexan, a polycarbonate that can be damaged by strong alkaline solutions.
Altitude	2,000 meters
Vibration	10 ... 57 Hz @ 0.075 mm d.a. 57 ... 150 Hz @ 1 g
Shock	+/-15 g peak, 11 ms, half-sine wave

Storage Conditions

Storage Conditions

Temperature	˜40 ... 85°C (-40 ... 185°F)
Humidity	0 ... 95% RH noncondensing @ 60°C
Free Fall	1 m

Agency Approvals

Agency Approvals

UL 508 CSA 22.2-142 Factory Mutual Class I, Div 2 European Directive on EMC 89/336/EEC

Chapter 2
Quantum Addressing Modes

Overview

In the functional description of this expert module, the %IW/%MW (3x/4x) register addressing mode established in the Quantum world is widely used. This chapter describes the different modes used in Control Expert to address the data from a Quantum module.

NOTE: Topological addresses overlapping (%IWr.m.c) is not supported by Quantum application, use flat addressing (%IWx) when memory overlapping control is needed.

What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
Flat Addressing—800 Series I/O Modules	22
Topological Addressing—800 Series I/O Modules with Control Expert	23
Addressing Example	24
Discrete I/O Bit Numbering	25
Addressing the 140 ESI 062 10 Module	26

Flat Addressing—800 Series I/O Modules

Introduction

800 series I/O modules follow a system of flat address mapping in Control Expert. To work properly. each module requires a determinate number of bits and/or words. The IEC addressing system is equivalent to the 984LL register addressing. Use the following assignments:

- 0x is now `%Mx`
- 1x is now `%Ix`
- 3x is now `%IWx`
- 4x is now `%MWx`

The following table shows the relationship between 984LL notation and IEC notation.

Outputs and Inputs	984LL Notation Register Addresses	IEC Notation		
		System Bits and Words	Memory Addresses	I/O Addresses
output	0x	System Bit	%Mx	%Qx
input	1x	System Bit	%Ix	%Ix
input	3x	System Word	%IWx	%IWx
output	4x	System Word	%MWx	%QWx

To access the I/O data of a module,

Step	Action
1	Enter the address range in the configuration screen.

Examples

The following examples show the relationship between 984LL register addressing and IEC addressing:

000001 is now `%M1`

100101 is now `%I101`

301024 is now `%IW1024`

400010 is now `%MW10`

Topological Addressing—800 Series I/O Modules with Control Expert

Accessing I/O Data Values

Use topological addressing to access I/O data items. Identify the topological location of the module within an 800 series I/O module with Control Expert using the following notation:

```
%<Exchangetype><Objecttype>[\b.e\]r.m.c[.rank]
```

where:

- **b** = bus
- **e** = equipment (drop)
- **r** = rack
- **m** = module slot
- **c** = channel

NOTE: When addressing,

1. The [\b.e\] defaults to \1.1\ in a local rack and does not need to be specified.
2. The rank is an index used to identify different properties of an object with the same data type (value, warning level, error level).
3. The rank numbering is zero-based, and if the rank is zero, omit the entry.

For detailed information on I/O variables, please refer to the *EcoStruxure™ Control Expert, Program Languages and Structure, Reference Manual*.

Reading Values: An Example

To read	Action
input value (rank = 0) from channel 7 of an analog module located in slot 6 of a local rack:	Enter `%IW1.6.7[.0]`
input value (rank = 0) from channel 7 of an analog module located in slot 6 of drop 3 of RIO bus 2:	Enter `%IW\2.3\1.6.7[.0]`
'out of range' value (rank = 1) from channel 7 of an analog module located in slot 6 of a local rack:	Enter `%I1.6.7.1[.0]`

Addressing Example

Comparing the 3 Addressing Modes

The following example compares the 3 possible addressing modes. An 8-channel thermocouple 140 ATI 030 00 module with the following configuration data is used:

- mounted in slot 5 of the CPU rack (local rack)
- starting input address is 201 (input word `%IW201`)
- end input address is 210 (input word `%IW210`)

To access the I/O data from the module you can use the following syntax:

Module data	Flat Addressing	Topological Addressing	IODDT Addressing	Concept Addressing
Channel 3 temperature	%IW203	%IW1.5.3	My_Temp.VALUE	300203
Channel 3 out of range	%IW209.5	%I1.5.3.1	My_Temp.ERROR	300209 Bit 5 to be extracted by user logic
Channel 3 range warning	%IW209.13	%I1.5.3.2	My_Temp.WARNING	300209 Bit 13 to be extracted by user logic
Module internal temperature	%IW210	%IW1.5.10	not accessible through IODDT	300210

NOTE: For the IODDT the data type `T_ANA_IN_VWE` is used and the variable `My_Temp` with the address `%CH1.5.10` was defined.

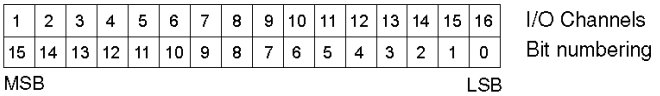
For comparison, the register addressing as used with Concept is added in the last column. As Concept does not support direct addressing of a bit in a word, the bit extraction has to be performed in the user program.

Discrete I/O Bit Numbering

Introduction

The numbering of channels of an I/O module usually starts with 1 and counts up to the maximum number of supported channels. The software however starts numbering with a 0 for the least significant bit in a word (LSB). The Quantum I/O modules have their lowest channel mapped to the most significant bit (MSB).

The following figure shows the mapping of I/O channels related to the bits in a word:.



Word Addressing Versus Bit Addressing

Mainly discrete I/O modules can be configured to deliver their I/O data either in word format or in bit format. This can be selected during configuration by selecting either `%IW` (`%MW`) or `%I` (`%M`). If you need to access a single bit from an I/O module configured to use an I/O word, you can use the syntax `%word.bit`. The following table gives you the connection between I/O point number and the associated I/O address in bit and word addressing.

The table shows a 32-point input module in the main rack, slot 4 configured with starting address `%I1` or `%IW1`:

I/O channel	Bit address (flat addressing)	Bit address (topological addressing)	Bit address extracted from word (flat addressing)	Bit address extracted from word (topological addressing)
1	%I1	%I1.4.1[.0]	%IW1.15	%IW1.4.1.1.15
2	%I2	%I1.4.2[.0]	%IW1.14	%IW1.4.1.1.14
3	%I3	%I1.4.3[.0]	%IW1.13	%IW1.4.1.1.13
• • •				
15	%I15	%I1.4.15[.0]	%IW1.1	%IW1.4.1.1.1
16	%I16	%I1.4.16[.0]	%IW1.0	%IW1.4.1.1.0
17	%I17	%I1.4.17[.0]	%IW2.15	%IW1.4.1.2.15
18	%I18	%I1.4.18[.0]	%IW2.14	%IW1.4.1.2.14
• • •				
31	%I31	%I1.4.31[.0]	%IW2.1	%IW1.4.1.2.1
32	%I32	%I1.4.32[.0]	%IW2.0	%IW1.4.1.2.0

Addressing the 140 ESI 062 10 Module

Flat Addressing

The 140 ESI 062 10 ASCII interface module requires 12 contiguous 16-bit input words (%IW), and 12 contiguous 16-bit output words (%QW).

Topological Addressing

The topological addresses for the 140 ESI 062 10 module are as follows:

Point	I/O Object	Comment
Input 1	%IW[\b.e\]r.m.1.1	Response word
	• • •	
Input 12	%IW[\b.e\]r.m.1.12	Data
Output 1	%QW[\b.e\]r.m.1.1	Command word
	• • •	
Output 12	%QW[\b.e\]r.m.1.12	Data

where: **b** = bus, **e** = equipment (drop), **r** = rack, **m** = module slot

NOTE: I/O words 2 ... 12 are used for data exchange between the module and the CPU, depending on the active command.

Chapter 3
Configuration Overview

Overview

This chapter describes the basics of the configuration mode of the ESI module. A description of the data flow between external devices and the PLC is included at the end of the chapter.

What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
140 ESI 062 10 Configuration	28
ASCII Message Formats	30
Data Flow	36
Parameter Configuration	40

140 ESI 062 10 Configuration

Overview

The 140 ESI 062 10 module has a built-in command line editor used to configure the port communication settings, the internal clock, and the ASCII messages.

Programming Port

The 140 ESI 062 10 module supports two RS 232 hardware ports that have their individual parameter settings at runtime. The first port also is used as a programming port. In this mode it has its own set of parameters.

Entering Configuration Mode

To enter the configuration mode, perform the following steps:

Step	Action
1	Connect a dumb terminal or a PC terminal emulator such as Hyperterminal to port 1. For information about the appropriate cable see *RS-232 Serial Ports, page 15*
2	Set the communication parameters of the terminal to 9600 baud, 8 data bit, no parity, 1 stop bit, and XON/XOFF flow control.
3	Press the **Reset** button on the front of the module for more then 4 sec.

The Command Line Editor

After you have entered configuration mode, the yellow **Status** LED on the front panel turns on, and the following message appears on your terminal screen:

```
Welcome
MODICON QUANTUM ASCII Module
Entering Program Mode ...
Current date is: Wed 01-01-2002
Current time is: 09:15:10a
CLI> _
```

Available Commands

The following command structure is provided in the command line editor:

Command		Description	Example
CLI		Sets programming mode to the Command Line Interpreter.	N/A
HELP		Displays available commands and a brief description on the command, or displays help on the command requested (e.g., CLI> HELP ASCII displays help on the ASCII command.)	N/A
RUN		Resets Module and goes into normal running mode.	N/A
CONFIG		Sets programming mode to Configuration Interpreter.	N/A
	DATE	Displays or sets the current date in the module.	See chapter Configuration Editor for examples
	TIME	Displays or sets the current time in the module.	
	PORT	Displays or sets the port parameter settings.	
ASCII		Sets programming mode to ASCII Message Interpreter.	N/A
	NEW	Enters the message editor and holds the new message in the work buffer.	`ASCII>new`
	EDIT	Displays a specified message, enters the message editor, and saves the specified message when done.	`ASCII>edit (message #)`
	VIEW	Displays an existing message for viewing.	`ASCII>view (message #)`
	SAVE	Saves changes made to a specified message in its work buffer.	`ASCII>save (message #)`
	CLR	Clears a specified message.	`ASCII>clr (message #)`
	COPY	Copies a specified message to another message.	`ASCII>copy (message #) (message #)`
	SIM	Simulates a specified message. Shows how many registers are used (for aid in mapping when creating user logic) and the maximum depth of nested messages (for additional debugging tool). Notification is sent if the maximum depth is greater than 8 and also shows the nested message path.	`ASCII>sim (message #)`
	DIR	Display a directory of all available messages. Use of CNTL S and CNTL Q can be used to stop and continue the data being displayed to the terminal.	N/A.
	DLOAD	Download messages from a PC to the module. See ASCII Message Transfer for more details.	N/A.
	ULOAD	Uploads all programmed messages (1 ... 255).	`ASCII>uload`
		Uploads a specified programmed message(s) from the module to a PC. See ASCII Message Transfer for more details.	`ASCII>uload (message # - message #)`

ASCII Message Formats

ASCII messages are used to send information from the 140 ESI 062 10 module to ASCII devices, e. g., terminal programs. The ASCII message formats define how data contained in the CPU get converted to a stream of serial characters and vice versa.

The following table lists the available message formats:

Format	Direction	Description
Text	Output	Static text
ASCII	Output/Input	ASCII characters
Hexadecimal	Output/Input	Hexadecimal numbers
Octal	Output/Input	Octal numbers
Binary	Output/Input	Binary numbers
Integer	Output/Input	Integer numbers
Fixed Point Decimal	Output/Input	Fixed Point Decimal numbers
Time/Date	Output	Time/Date information
Control Characters	Output	Space and Newline characters
Control Sequences	Output	3 digit octal control characters
Nesting	Output/Input	Nesting of messages

Text Format

An arbitrary ASCII string enclosed in single quotes (e.g. 'message string') is an output only format. Any message that contains this format sends the text, whether or not the message is started from a read or write message command.

'. . . (text) . . .'

ASCII Format

Here is a variable field of the ASCII format with number of registers and field length:

n**A**m

where:

- n is the number of registers 1..99 (format repeat)
- m is the field length 1..2 (number of characters)

For example, 2A2 as an input stands for 2 registers, each containing 2 ASCII characters.

Hexadecimal Format

Here is a variable field of the hexadecimal format with number of registers and field length:

n**H**m

where:

- n is the number of registers 1..99 (format repeat)
- m is the field length 1..4 (number of numbers)

For example, 2H3 as an input stands for 2 registers, each containing 3 hexadecimal numbers.

Octal Format

Here is a variable field of the octal format with number of registers and field length:

n**O**m

where:

- n is the number of registers 1..99 (format repeat)
- m is the field length 1..6 (number of numbers)

For example, 3O4 as an input stands for 3 registers, each containing 4 octal numbers.

Binary Format

Here is a variable field of the binary format with number of registers and field length:

n**B**m

where:

- n is the number of registers 1..99 (format repeat)
- m is the field length 1..16 (number of numbers)

For example, 1B8 as an input stands for 1 register containing 8 binary numbers.

Integer Format, Leading Spaces

Here is a variable field of the integer/decimal format using the leading spaces for the output with number of registers and field length. On input, this format accepts leading zeros and spaces as a zero.

n**I**m

where:

- n is the number of registers 1..99 (format repeat)
- m is the field length 1..5 (number of numbers)

For example, 2I5 as an input stands for 2 registers, each containing 5 integer/decimal numbers. The maximum value is 65,535.

Integer Format, Leading Zeroes

Here is a variable field of the integer/decimal format using the leading zeroes for the output with number of registers and field length. On input this format accepts leading zeroes and spaces as a zeros.

n**L**m

where:

- n is the number of registers 1..99 (format repeat)
- m is the field length 1..5 (number of numbers)

For example, 3L5 as an input stands for 3 registers, each containing 5 integer/decimal numbers. The maximum value is 65,535.

Fixed-point Decimal Format

Here is a variable field of the fixed-point decimal format using leading spaces for the output with number of registers and field length. On input, this format accepts leading zeros and spaces as a zeros.

n**P**m.q

where:

- n is the number of registers 1..99 (format repeat)
- m is the number of numbers + '.' 3..8
- q is the number of fraction numbers 1..5

For example 1P7.2 as an input stands for 1 register containing 4 decimal numbers followed by a decimal point and 2 more decimal numbers (the fraction part).

NOTE: Do not confuse this format with floating point format. The placement of the decimal point is for input/output formatting and has no influence on the value in the PLC register (e.g., all 3 values 23.456, 234.56 and 23456 refer to a register value of 23456).

Nested Message Format

The nesting message format allows one message to call another message. This format can be used within the repeat format. Repeat formats can be used in nested messages, allowing indirect nested repeats. The maximum allowable nested message level is 8. Recursive nesting is not allowed.

Mn

where n is the message number 1..255

For example, M6 runs message number 6.

Time Formats

Two different time formats can be used to display time, 12-hour format and 24-hour format. This is an output-only format.

T12 > hh:mm:ss AM/PM (12 hour time)

T24 > hh:mm:ss (24 hour time)

Date Formats

Five different date formats can be used to display the date, each having 2 types of formats for displaying the year. This is an output-only format.

Dnm

where:

- n is the day and month type 1..5
- m is the year type 2 or 4

D12 > dd/mm/yy

D14 > dd/mm/yyyy

D22 > mm/dd/yy

D24 > mm/dd/yyyy

D32 > dd mmm yy

D34 > dd mmm yyyy

D42 > mmm dd, yy

D44 > mmm dd, yyyy

D52 > dd.mm.yy

D54 > dd.mm.yyyy

dd = day (1..31)

mm = month (1..12)

mmm = month (JAN, FEB, .. , DEC)

yy = year (0..99) (90 - 99 in 1900's, 0 - 89 in 2000's)

yyyy = year (1990..2089)

Repetition of Several Formats

Nesting of repeat brackets is not valid.

n**(**...**)**

where n is the number of times to repeat what is in ( )1. .99

For example: 6('Item',1I2,4X,1I5,/) produces 6 lines, each containing the fields 'Item',1I2,4X,1I5, and a <CR, LF>.

Space

The ASCII message symbol for space is X. This is an output only format.

n**X**

where n is the number of spaces 1..99

Newline

The ASCII message symbol for a carriage return is **/**. This is an output-only format.

Control Codes

Control codes appear as 3-digit octal characters (in the range 000 377) enclosed in double quote delimiters. This is an output-only format.

"###"

where ### is the octal form of a character

For example: "033".

Flush

Flush the input buffer of the currently running serial port in one of four ways—the entire buffer, a number of characters, up to a character pair, or up to a character pair repeatedly

<0> flush entire buffer

<1;bbb**>** flush until number of characters removed

<2;hhhh**>** flush until character pair match

<3;rrr;hhhh**>** flush until character pair match repeatedly

where:

- bbb = number or characters (1..255)
- hhhh = character pair, in hexadecimal (0000..FFFF)
- rrr = number of repeats (1..255)

NOTE: The port buffer size is 255 characters.

ASCII Message Syntax Rules

Messages created with the module's ASCII Message Editor or downloaded using the ASCII Message Transfer are checked after being entered for general and format syntax violations. If any violations are found, the message either is not saved (ASCII Message Transfer) or the user is notified and the violation is pointed out (ASCII Message Editor).

- A format delimiter (,) must separate each format.
- All text formats must be closed.
- Formats A,H,O,B,I,L,P,X, and (can have a repeat/number of registers value from 1 to 99.
- Formats A,H,O,B,I, and L can have a total field size from 1 to 8.
- Format P can have a total field size from 3 to 8 and a fractional field size from 1 to 5 but the total field size must be at least 2 greater than the fractional field size.
- Format M (Nested Message) can have any message number 1 to 255 (decimal) as long as it is not recursive.
- Format T can have 1 of 2 formats: T12 or T24.
- Format D can have 1 of 10 formats: D12, D14, D22, D24, D32, D34, D42, D44, D52, and D54.
- Control Code format "###" accepts only 3 digit octal values from 000 to 377.
- Flush format can have 1 of 4 formats: <0>, <1;bbb>, <2;hhhh>, or <3;rrr;hhhh> where bbb = 1 to 255, hhhh = 0000 to FFFF, and rrr = 1 to 255.

Standard ASCII Message Preprocessing Rules

Messages created with the Module's ASCII Message Editor or downloaded using the ASCII Message Transfer are preprocessed after being entered to save space and to standardize the messages for interpretation during simulation or running mode.

- Text is not massaged at all.
  Example: >'This is text...' > >'This is text...'
- Spaces preceding the first format are removed.
  Example: > 1A4,2X > >1A4,2X
- Spaces trailing the last format are removed.
  Example: >1A4,2X (end) > >1A4,2X(end)
- Spaces around formats and delimiters are removed.
  Example: >1A4 , 2X > >1A4,2X
- Commas trailing the last format are removed.
  Example: >1A4,2X,,, > >1A4,2X
- Commas trailing the last format in a repeat format are removed.
  Example: >1A4,2X,3(1I2,1X,,),/ > >1A4,2X,3(1I2,1X),/
- Non text characters are capitalized.
  Example: >'text ',1a4,2x,/ > >'text ',1A4,2X,/
- All preceding 0's are removed from a number except 0's in flush format's repeat/number value and character pair value.
  Example: >01A004,0002X > >1A4,2X

Data Flow

Overview

Exchanging data between the Quantum processor and the serial ports of the ESI module involves the following steps:

Transmit direction:

- Transfer of the data from the PLC registers to the ESI register area through the 12 output registers assigned to the ESI module in the I/O configuration.
- Interpreting the data in the ESI registers based on the ASCII messages and transfer to the port transmit buffer.

Receive direction:

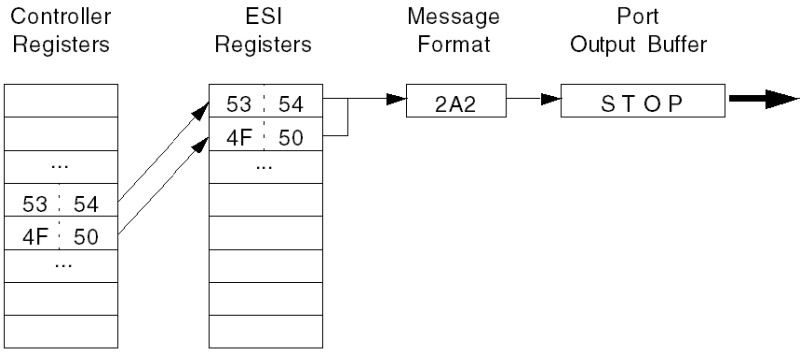
- Interpreting the data in the port receive buffer based on the ASCII messages and transfer to the ESI register area.
- Transfer of the data from the ESI register area to the PLC registers through the 12 input registers assigned to the ESI module in the I/O configuration.

ASCII Messages

The ASCII messages represent the central mechanism of how the data in the ESI registers are formated for the transmission through the RS-232 ports in either direction. A single 16-bit register for example could represent 2 ASCII characters and thus be transmitted as two characters it could also represent a single number which may be transmitted as an integer with leading spaces resulting in a string of five characters. For a detailed description of the available formats see *ASCII Message Formats, page 30*.

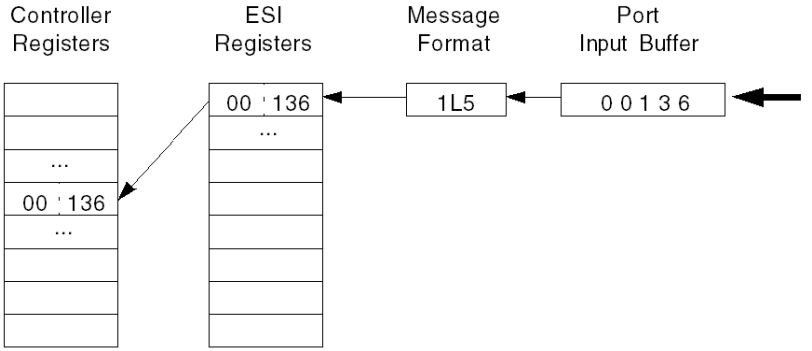
Transmitting Example

The following diagram is an example of transmitting 4 characters from the Quantum controller using the "2A2" message format (2 registers with 2 characters each). Port Buffer content is in ASCII format, register content in hex:



Receiving Example

The following diagram is an example of receiving 1 numerical value from the RS-232 port using the "1L5" message format (1 register, 5 digits with leading zeros). Port Buffer content is in ASCII format, register content in hex:

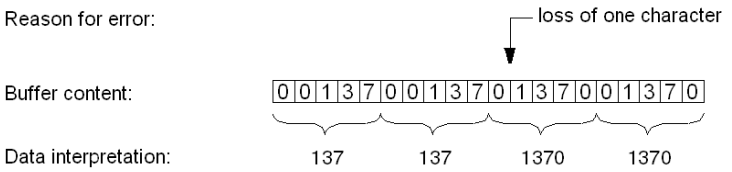


NOTE: Ensure the number of incoming characters match the number defined in the ASCII message. If in the above example the device sends "0013", the ESI module would not be able to finish the receive command and would wait until reception of a 5th character.

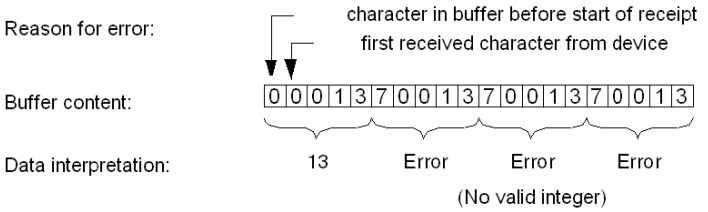
Possible Synchronisation Problems

As the ESI module only supports fixed length message formats without start or termination characters, any lost character (or additional unexpected character) can lead to a wrong interpretation of received data. The following examples show the result of 3 different error types. The assumed message format is "1L5 maximum 65,535":

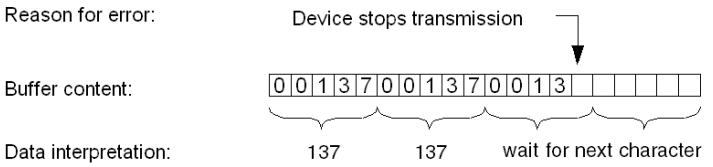
Effect of lost character:



Effect of buffer not empty at start of reception:



Effect of terminated reception:



FLUSH, ABORT, GET STATUS

To prevent mis-interpretation of data or locking the module the buffer related commands FLUSH BUFFER, ABORT, GET BUFFER STATUS should be used to control the data exchange.

For details of those commands see *List of ESI Commands, page 50*.

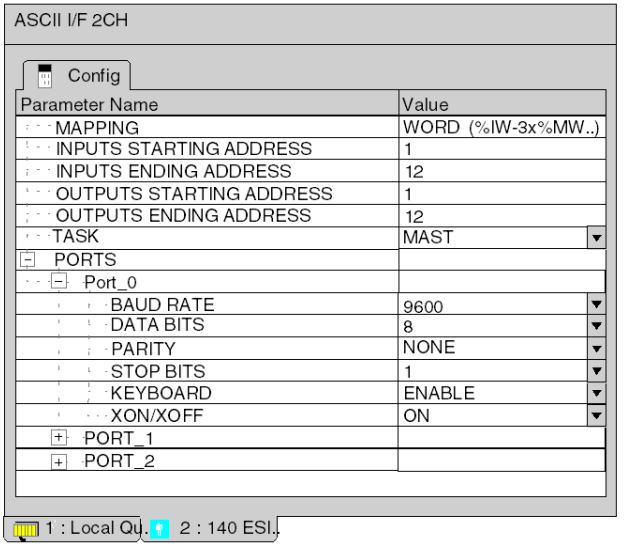
Parameter Configuration

Overview

The parameter editor is part of the Control Expert configuration of the ESI 062 10 module. The user is able to set several information about the input / output registers and the port parameter. The following figure display the different settings of the module.

Parameter and Default values

Parameter Configuration Window



Name	Default Value	Options	Description
Mapping	WORD (%IW-3X%MW-4X)	-	
Inputs Starting Address	1	-	
Inputs Ending Address	12	-	
Outputs Starting Address	1	-	
Outputs Ending Address	12	-	

Name	Default Value	Options	Description
Task (Grayed if module in other than local)	MAST	FAST AUX0 AUX1 AUX2 AUX3	fixed to MAST if module in other than local
PORTS			
PORT_0, PORT_1, PORT_2			
BAUD RATE	9600	300-19200	
DATA BITS	8	7	
PARITY	NONE (PORT_0) EVEN (PORT_1,PORT_2)	ODD	
STOP BITS	1	2	
KEYBOARD	ON (PORT_0) OFF (PORT_1,PORT_2)	ON / OFF	
XON/XOFF	ENABLE	DISABLE	

NOTE: the two following configurations must not be applied to the port 1:

- configuration 1:
  - data bits parameter set to 8
  - parity parameter set to either enabled or even or odd
  - stop bits parameter set to 2
- configuration 2:
  - data bits parameter set to 7
  - parity parameter set to none
  - stop bits parameter set to 1

If one of the two configurations is applied to the port 1 data transfer errors occur.

Chapter 4
ESI Command Line Editors

Overview

The ESI firmware contains an editing environment that can be accessed by a dumb terminal connected through port 1. This chapter describes how to use this editor to configure the module and to edit the ASCII message formats.

What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
Configuration Editor	44
ASCII Message Editor	47

Configuration Editor

Overview

The Configuration Editor Interface is part of the programming mode. It is used to configure the serial ports and the time of day clock of the module.

NOTE: Configuration of the serial ports can also be accomplished through the I/O map. The I/O map overrides any serial port configuration entered in the configuration editor.

NOTE: Configuration of the time of day clock can also be accomplished with the SET TOD command.

To enter the configuration editor type`CONFIG` at the `CLI>` prompt. The configuration editor displays the prompt `CONFIG>` .

Port Command

The Port Command displays or sets the port parameter settings. Acceptable command format variations include:

```
PORT [n[: [b] [,p] [,d] [,s] [,k] [,x]]]
```

```
PORT [n[: [BAUD=b] [,PARITY=p] [,DATA=d] [,STOP=s] [,KEYBOARD=k]
[,XON/XOFF=x]]
```

Description and range of the elements used in the PORT command:

Index	Description	Range
n	Port number	0, 1, 2
b	Baud rate	50, 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, 9600, 19200
p	Parity setting	N, O, E
d	Number of data bits	5, 6, 7, 8
s	Number of stop bits	1, 2
k	Keyboard mode (Character echo mode)	on, off
x	XON/XOFF mode (Software flow control)	on, off

Examples:

```
PORT 0:1200,n,8,1,on,on
```

```
PORT 0:baud=1200, parity=n, data=8, stop=1, keyboard=on, XON/XOFF=on
```

```
PORT 0
```

```
Current port parameters are: PORT 0: BAUD=1200, PARITY=NONE ...
```

```
Enter new parameters: 4800,n,8,1,off,on
```

After the Port settings in the module have been changed, the following message will appear:

```
Note: The port settings are temporary during this programming session.
```

NOTE: Ports 0 and 1 do not support all baud rate and data bit options. Refer to the Module Configuration screen for available options.

Date Command

Displays or sets the current date in the module. Acceptable command format variations include:

```
DATE     [mm dd [  yy]]
DATE     [mm/dd [/ yy]]
DATE     [mm.dd [.yy]]
DATE     [mm dd [ YYYY]]
DATE     [mm/dd [/YYYY]]
DATE     [mm.dd [.YYYY]]
```

Description and range of the elements used in the DATE command:

Index	Description	Range
mm	Month	1 ... 12
dd	Day	1 ... 31
yy	Year	00 ... 99
yyyy	Year	1990 ... 2089

Examples:

```
DATE 3 30 95
DATE 3/3 0/1995
DATE
Current date is Wed 3 29 1995
Enter new date: 3.30
```

NOTE: If the year does not need to be changed, then only the month and day need to be entered. The day of week is automatically figured out by the firmware. The yy years are mapped 00..89 = 2000..2089 and 90..99 = 1990..1999.

Time Command

Displays or sets the current time in the module. Acceptable command format variations include:

```
TIME [hh:mm[:ss][x]]
TIME [hh.mm[.ss][x]]
```

Description and range of the elements used in the TIME command:

Index	Description	Range
hh	Hour	1 ... 23
mm	Minute	1 ... 59
ss	Second	1 ... 59
x	Meridian	a, p

Examples:

```
TIME 3:26p
TIME 3.26.30p
TIME 15.26
TIME
Current time is 3:15:26p
Enter new time: 3.26.30p
```

NOTE: The time can be entered in either 12 or 24 hour time format. Not entering the meridian assumes AM unless the hour is 0 or 13 to 23.

ASCII Message Editor

Overview

The ASCII Message Editor Interface is used to program the ASCII message formats in the module. This interface consists of a simple command line interpreter (also similar to the CLI that is in the Modicon B885 002 module), which consists of commands that allow you to display, create, edit, transfer, save, clear, and test ASCII messages. Also included in the command set is a help command, which gives an online list of the available commands and the meaning of each command.

To enter the ASCII message editor type `ASCII` at the `CLI>` prompt. The ASCII message editor uses the prompt `ASCII>`

Chapter 5
ESI Commands

Introduction

The information in this chapter describes the commands which are sent by the CPU to control the communication functions of the ESI module and the response from the ESI module containing data and status information.

What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
Overview on ESI Commands	50
ESI Command Word	51
Command Processing	52
Command 0 - NO OPERATION	54
Command 1- READ ASCII MESSAGE	55
Command 2 - WRITE ASCII MESSAGE	57
Command 3 - GET DATA (Module to Controller)	60
Command 4 - PUT DATA (Controller to Module)	62
Command 5 - GET TOD (Time of Day)	64
Command 6 - SET TOD (Time of Day)	66
Command 7 - SET MEMORY REGISTERS	69
Command 8 - FLUSH BUFFER	71
Command 9 - ABORT	72
Command A - GET BUFFER STATUS	73
Response Structure for Illegal Commands	74
Module Status Word (Word 11)	75
Reading beyond Valid Register Range	77

Overview on ESI Commands

List of ESI Commands

There are 11 ASCII module commands which instruct the ESI module serial communications and other housekeeping utilities. These commands are sent to the ESI module by the Quantum controller. Data exchange between the ASCII device and the Quantum controller is integrated into the READ/WRITE command structure described in this section. The output data (the first 4x registers) contains the command; the first input register (3x) contains the response and also the echo of the command.

The following table is a summary of the ESI module commands:

Command	Name	Description
0	No operation	do nothing
1	READ ASCII message	start a read ASCII message
2	WRITE ASCII message	start a write ASCII message
3	GET DATA	transfer data from module to PLC
4	PUT DATA	transfer data from PLC to module
5	GET TOD	get time of day from module
6	SET TOD	set time of day in module
7	SET MEMORY REGISTERS	set registers to value
8	FLUSH BUFFER	flush serial port buffers
9	ABORT	abort ASCII message currently running
A	GET BUFFER STATUS	get port input buffer

ESI Command Word

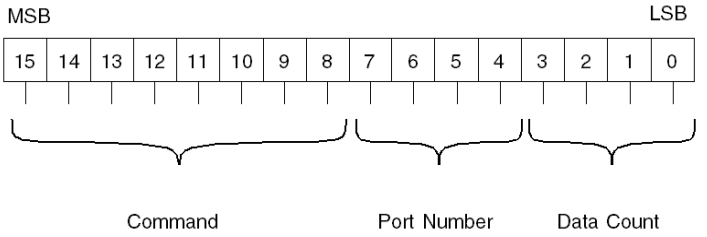
Command Word Format

The command word is the first output register mapped to the module.

The command word format for the ESI module is as follows:

- Bits 0 ... 3 - contain the data count (in words), range is 0 ... 9
- Bits 4 ... 7 - contain the port number, range is 1 ... 2
- Bits 8 ... 15 - contain the command, range is 0 ... A

Structure of the command word:



NOTE: The bit order is based on the IEC standard, where bit 15 is the most significant bit.

Command Processing

Register

The registers 3:x (PLC input register) and 4:x (PLC output register) are used to process commands with the ESI module. The x refers to the starting address of the ESI module in the PLC hardware configuration.

The command data processed by the ESI module is placed in the output registers (4:x) and the possible response information is placed in the input registers (3:x).

The following example shows the register occupancy through Command 5, Upload the ESI System Time and Command 6, Set the ESI System Time.

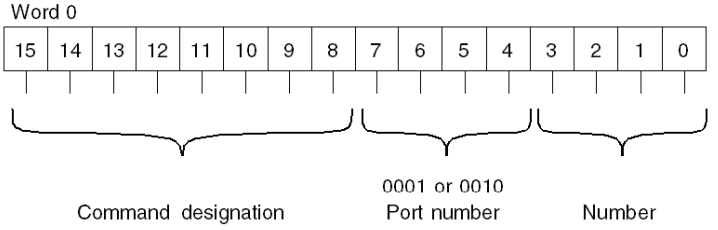
Example 5 GET TOD

Command 5 is used to upload the system time. For the command to be correctly executed, the command parameter must be written in the Word 0 of the ESI module output register. Word 0 is the first output register in the modules hardware configuration (PLC configuration).

NOTE: When addressing the hardware with the Start address 4:1 to the End address 4:12 in the PLC configuration, command word 0 corresponds to the address 4:1.

Command structure

Command word 0 is divided into the following areas:



For example: Word 0

0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0

Description of the command word:

Area (Bit)	Description	Example value
0 - 3	Number of the register to be uploaded or output. The number of the output register (3:x) is defined with command 5. This sets the value 0.	0
4 - 7	Port number. The ports are not used when commands 5 or 6 are executed. The data is only processed internally in the module using the register.	0
8 - 15	Command designation in Bit format. When the command value is set the command is directly processed.	5

NOTE: Command 0 can be set with the help of Move-Blocks or by external switches. Other variations are also possible.

Result

As a result of the action the ESI system time data is placed in registers 1 to 7 *(see page 65)*.

Data return is carried out via the PLC 3:x register. It corresponds to the input registers in the modules hardware configuration (PLC configuration)

NOTE: Register 0 (status register) shows the status of the command processing. The register corresponds to command word 0 when the command has been executed correctly. If faulty data occurs, the status of the MSB (Most Significant Bit) changes from 0 to 1.

Example 6 SET TOD

Command 6 is used to set the system time. As with command 5, the command parameters required must be written to Word 0 of the ESI module output register (4:x). The time and date parameters are additionally transferred when setting the system time. The parameters are placed in the registers following command word 0 *(see page 67)*.

NOTE: Before setting command word 0 the time and date information must be placed in the corresponding 4:x registers.
The successful execution of the command can be monitored during processing with the help of the status register.

Command 0 - NO OPERATION

Overview

The NO OPERATION command does nothing in or to the ESI module. It is present to allow multiple scan command builds (setting up of Command Words 1 to 11, then setting Command Word 0 to start the command execution) and toggling for repeating command that do not run continuously.

This command is executed continuously until Command Word 0 changes to a command other than NO OPERATION.

Command Structure

Word 0 0000 (hex)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

NOTE: Words 1 through 11 for Command 0 are not used.

Response Structure

Word 0 0000 (hex) Echo Command Word 0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Note: Bit 15 is the Status Word Valid bit.

•
•
•

Word 11 XXXX hex Module Status

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

NOTE: Words 1 through 10 for Command 0 return a 0.

Command 1- READ ASCII MESSAGE

Overview

The READ ASCII MESSAGE command is used to start running a read message on the module, that is, taking ASCII characters from the input/receive buffer of a serial port to fulfil the variable formats of the message. All output only formats still send ASCII characters to the serial port.

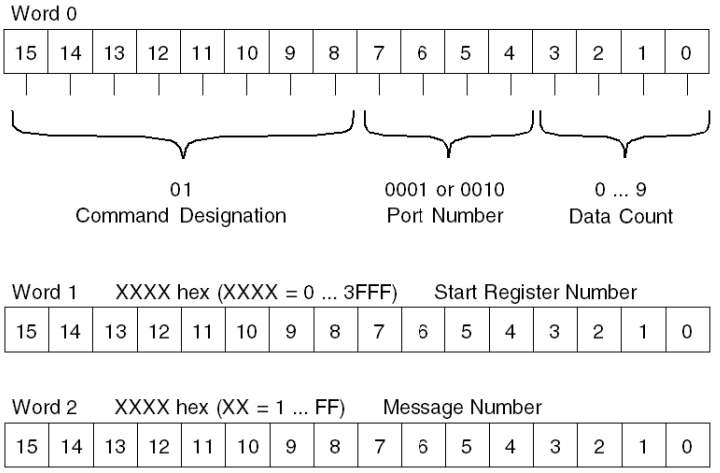
To start a message, the module needs to know the following:

- The port number to be used
- The starting module register number for the data that is processed
- The message number to run

In addition to starting a message, this command is capable of transferring up to nine registers of data from the module to the controller after the message has completed (this is the data count). The data returned is gotten from the starting register number provided in Command Word 1.

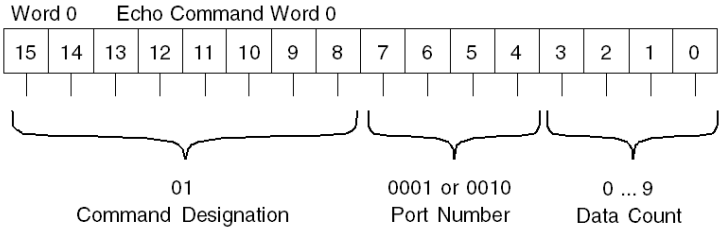
This command is executed only the first time it is received. To execute the command again, Command Words 0, 1, or 2 need to be changed. This is done so that the same message does not get continuously run until Command Word 0 changes to a command other than READ ASCII MESSAGE.

Command Structure

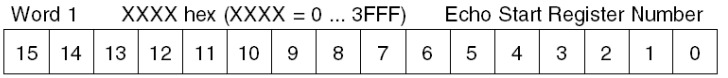


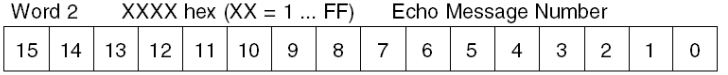
NOTE: Words 3 through 11 for Command 1 are not used.

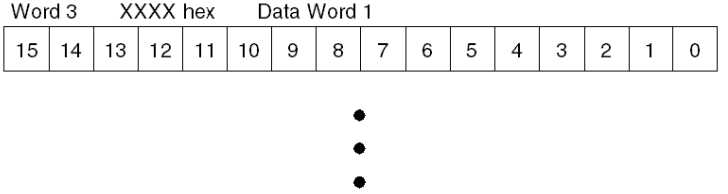
Response Structure



Note: Bit 15 is the Status Word Valid bit.







Command 2 - WRITE ASCII MESSAGE

Overview

The WRITE ASCII MESSAGE command is used to start running a write message on the module, that is, putting ASCII characters to the output/transmit buffer of a serial port.

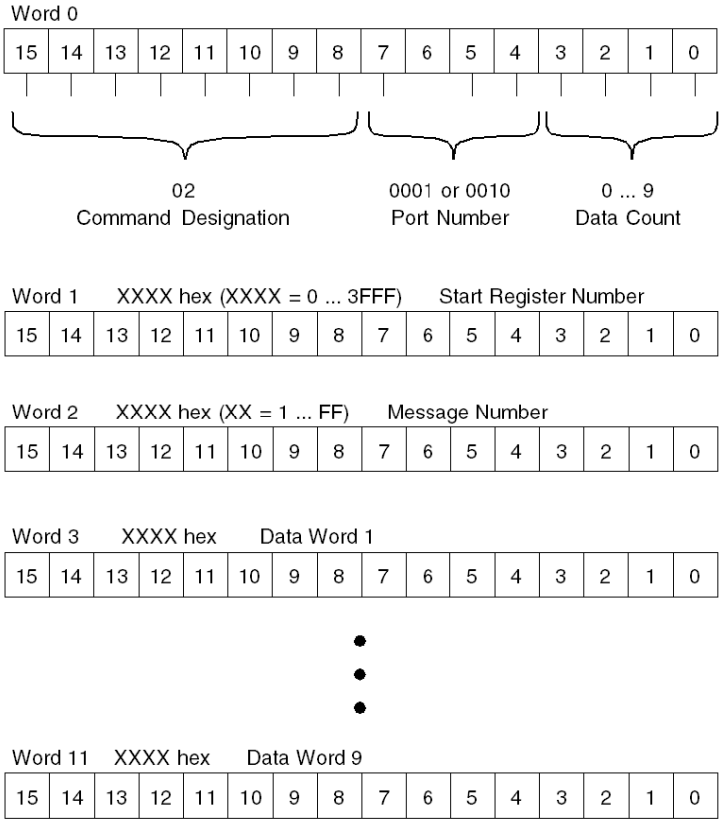
To start a message, the module needs to know the following:

- The port number to be used
- The starting module register number for the data that is processed
- The message number to run

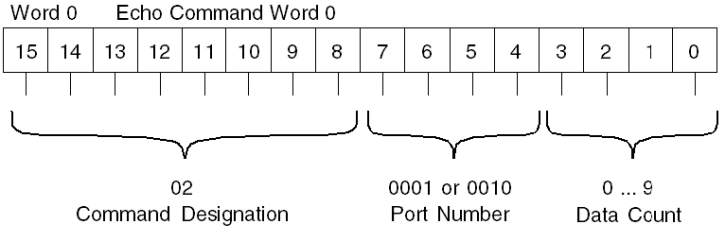
In addition to starting a message, this command is capable of transferring up to nine registers of data from the controller to the module before the message has started (this is the data count). The data sent is stored starting at the start register number provided in Command Word 1.

This command is executed only the first time it is received. To execute the command again, Command Words 0, 1, or 2 (plus any data word that is sent - keyed off the data count) need to be changed. This is done so that the same message does not get continuously run until Command Word 0 changes to a command other than WRITE ASCII MESSAGE.

Command Structure



Response Structure



Note: Bit 15 is the Status Word Valid bit.

NOTE: Words 3 through 10 for Command 2 return a 0.

Command 3 - GET DATA (Module to Controller)

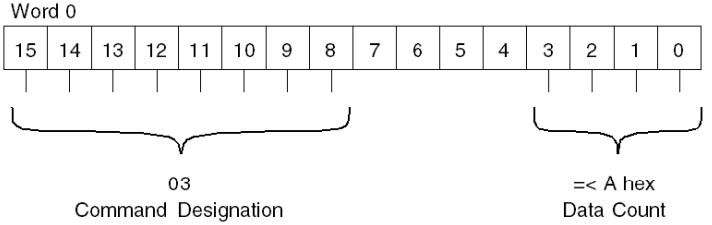
Overview

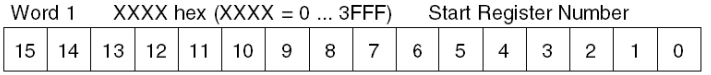
The GET DATA command reads up to 10 words/registers of data from the module starting at the start register number provided in Command Word 1. The data count provided in Command Word 0 determines the number of words to read. The data is returned in Response Words 2 through 11.

NOTE: If there is an error status to be reported (and is not a command syntax error) and the command requests 10 registers of data, the module will return only 9 words of data and use Response Word 11 for the module status. The Status Word Data bit will be set if Response Word 11 is the module status.

This command is executed continuously until Command Word 0 changes to a command other than GET DATA.

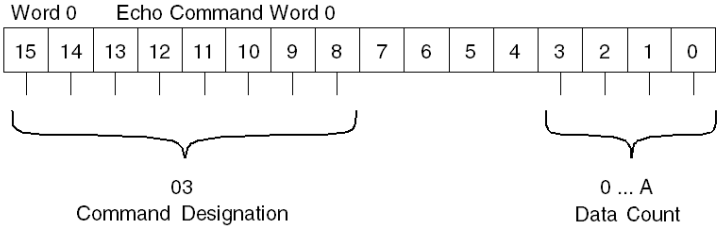
Command Structure



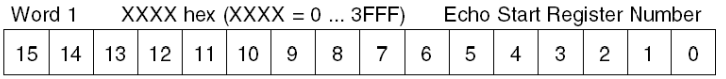


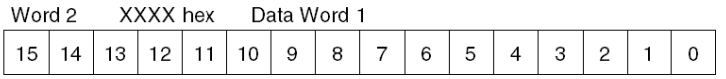
NOTE: Words 2 through 11 for Command 3 are not used.

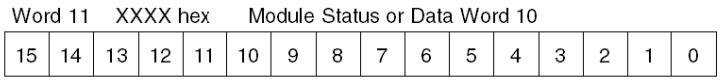
Response Structure



Note: Bit 15 is the Status Word Valid bit.







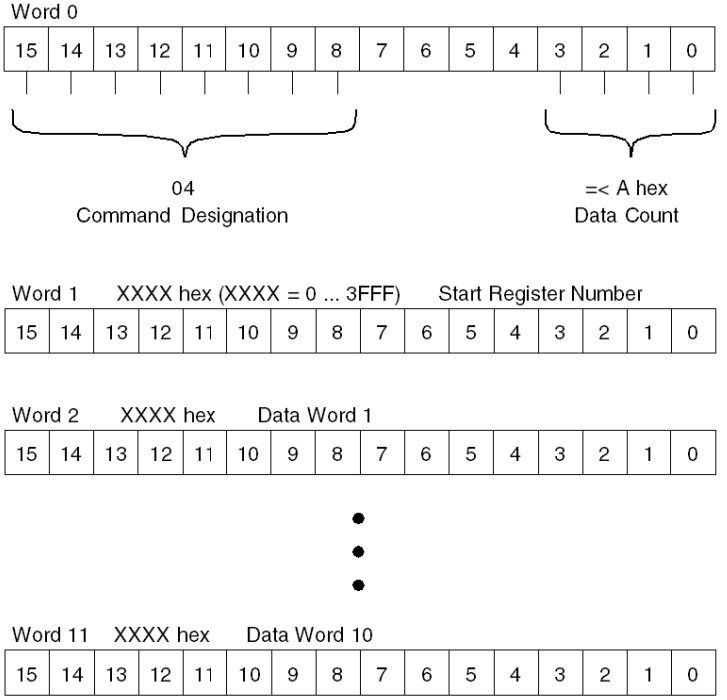
Command 4 - PUT DATA (Controller to Module)

Overview

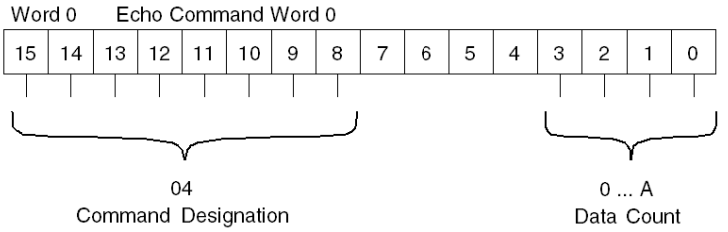
The PUT DATA command writes up to 10 words/registers of data to the module starting at the start register number provided in Command Word 1. The data is sent in Command Words 2 through 11.

This command is executed continuously until Command Word 0 changes to a command other than GET DATA.

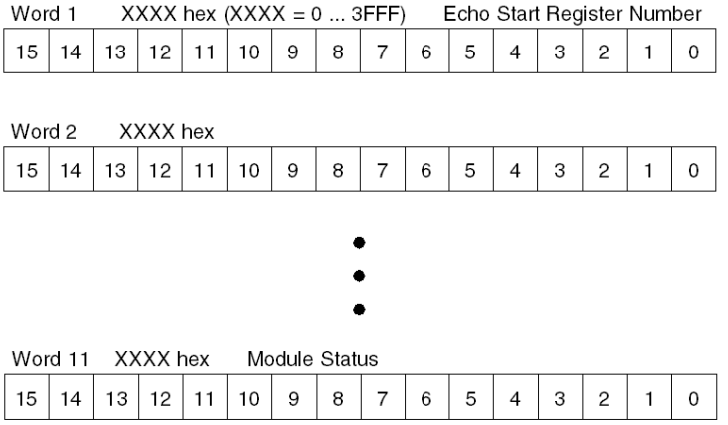
Command Structure



Response Structure



Note: Bit 15 is the Status Word Valid bit.



NOTE: Words 2 through 10 for Command 4 return a 0.

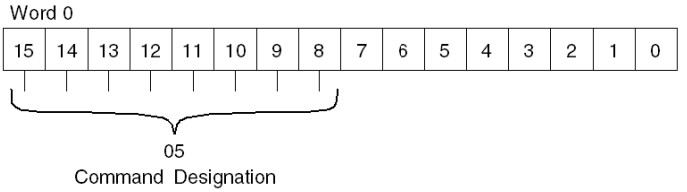
Command 5 - GET TOD (Time of Day)

Overview

The GET TOD command reads the module's TOD clock and returns the time of day and the date in the Response Words 1 to 7. The format for the time of day and date is identical to that used by the PLC time/date registers.

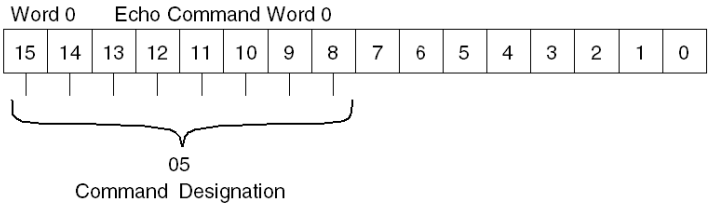
This command is executed continuously without the need for changing any of the command words.

Command Structure

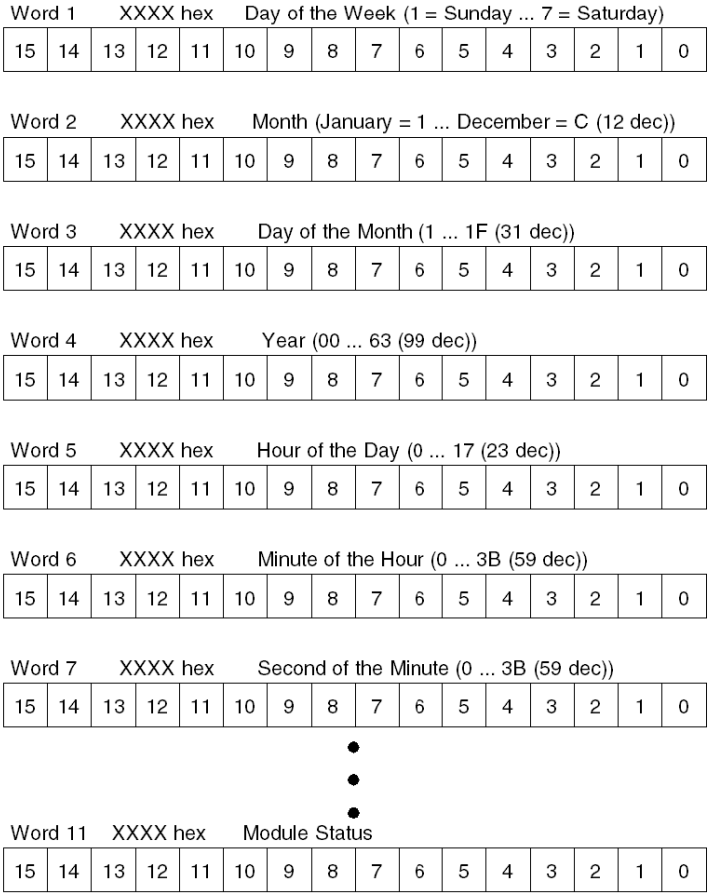


NOTE: Word 1 through Word 11 for Command 5 are not used.

Response Structure



Note: Bit 15 is the Status Word Valid bit.



NOTE: Words 8 through 10 for Command 5 return a 0.

Command 6 - SET TOD (Time of Day)

Overview

The SET TOD command loads the modules TOD clock with the time of day and the date as provided in the Command Words 1 to 7. The format for the time of day and date is identical to that used by the PLC time/date registers.

NOTE: To synchronize the module's and PLC's TOD clocks, do a block move of the PLC's seven time/date registers to Command Words 1 to 7 and set Command Word 0 to 0600 hex.

This command is executed only the first time it is received. To execute the command again, one of the Command Words, 0 to 7, needs to be changed. This is done so that the same time does not get continuously loaded until Command Word 0 changes to a command other than SET TOD.

Command Structure

Word 0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

06

Command Designation

Note: Bit 15 is the Status Word Valid bit.

Word 1 XXXX hex Day of the Week (1 = Sunday ... 7 = Saturday)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Word 2 XXXX hex Month (January = 1 ... December = C (12 dec))

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Word 3 XXXX hex Day of the Month (1 ... 1F (31 dec))

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Word 4 XXXX hex Year (00 ... 63 (99 dec))

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Word 5 XXXX hex Hour of the Day (0 ... 17 (23 dec))

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Word 6 XXXX hex Minute of the Hour (0 ... 3B (59 dec))

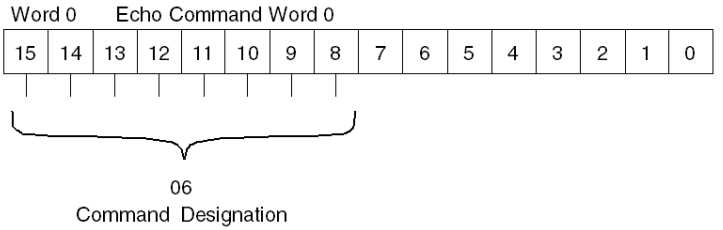
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Word 7 XXXX hex Second of the Minute (0 ... 3B (59 dec))

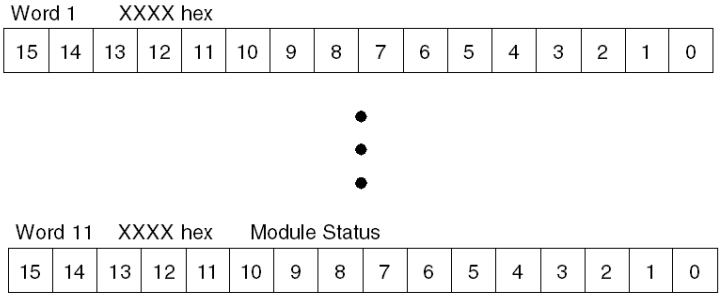
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

NOTE: Words 8 through 11 for Command 6 are not used.

Response Structure



Note: Bit 15 is the Status Word Valid bit.



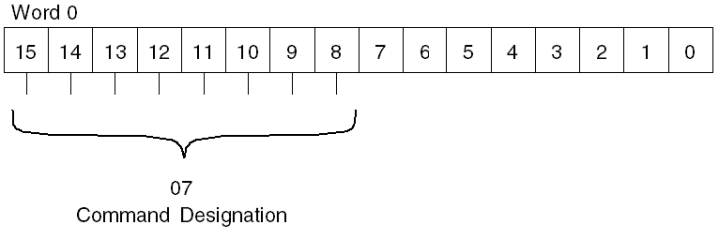
NOTE: Words 1 through 10 for Command 6 return a 0.

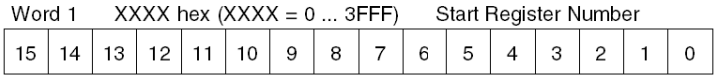
Command 7 - SET MEMORY REGISTERS

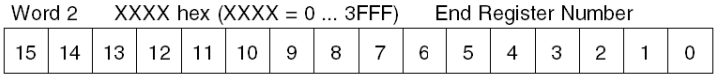
Overview

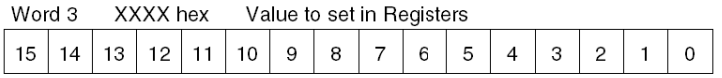
The `SET MEMORY REGISTERS` command sets module registers to the value provided in Command Word 3. The registers set are designated by the start register number and the end register number. All registers from the start register up to and including the end register number are set to the value provided.

Command Structure



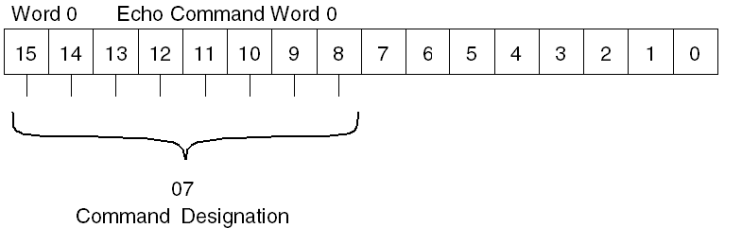




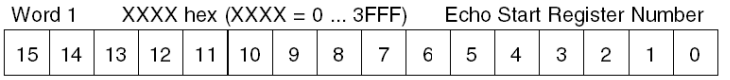


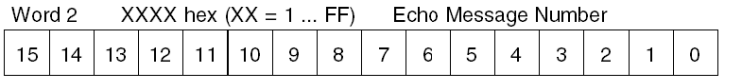
NOTE: Words 4 through 11 for Command 7 are not used.

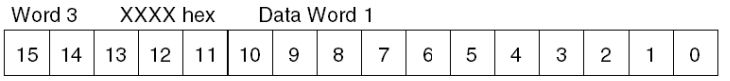
Response Structure

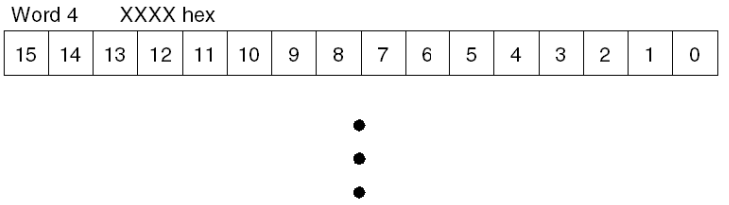


Note: Bit 15 is the Status Word Valid bit.









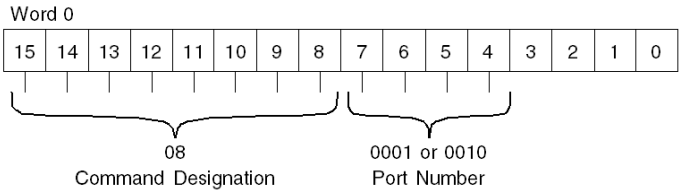
NOTE: Words 1 through Word 10 for Command 7 return a 0.

Command 8 - FLUSH BUFFER

Overview

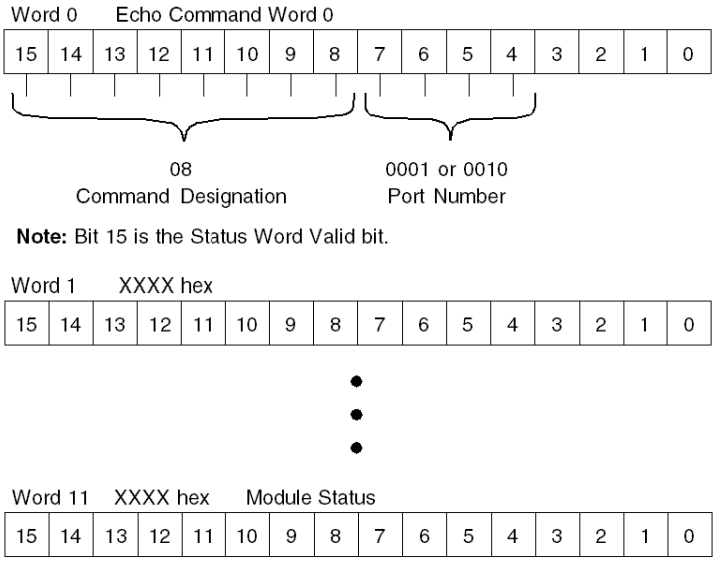
The FLUSH BUFFER command flushes the input buffer for the serial port number provided in the command word. The output buffer is not affected by this command.

Command Structure



NOTE: Words 1 through 11 for Command 8 are not used.

Response Structure



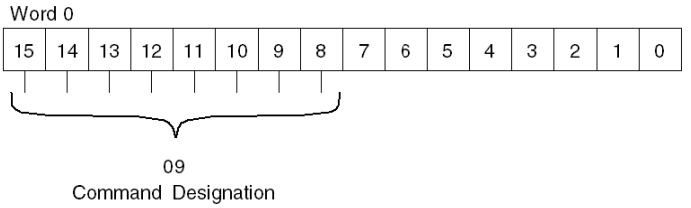
NOTE: Words 3 through 10 for Command 8 return a 0.

Command 9 - ABORT

Overview

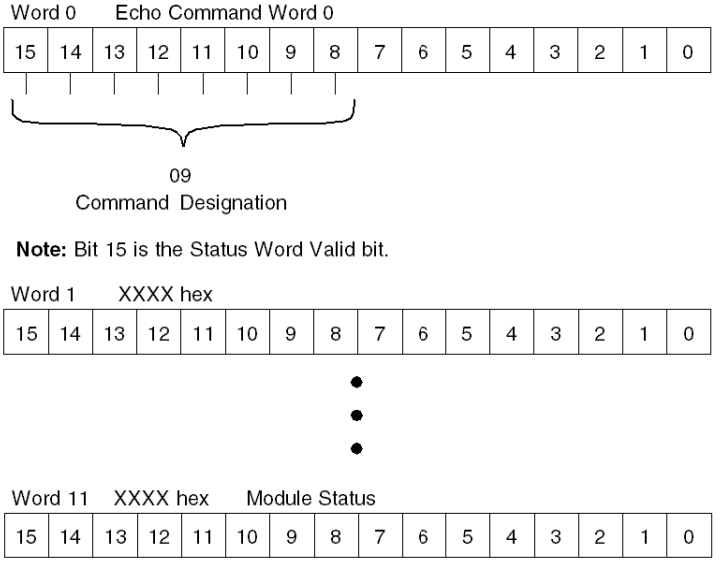
The ABORT command aborts a running READ or WRITE ASCII MESSAGE and the module is no longer in a busy status. The serial port buffers for the module are not affected by this command, only the message is currently running.

Command Structure



NOTE: Words 1 through 11 for Command 9 are not used.

Response Structure



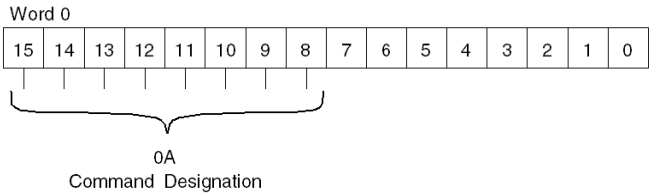
NOTE: Words 3 through 10 for Command 9 return a 0.

Command A - GET BUFFER STATUS

Overview

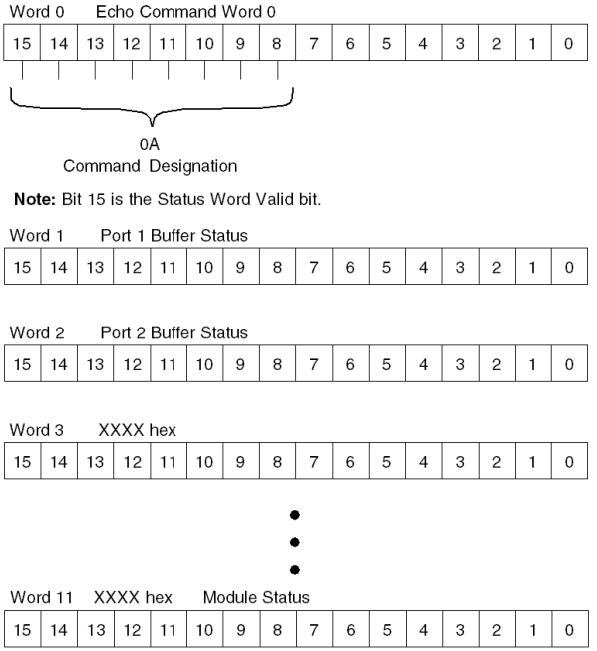
The GET BUFFER STATUS command reads the number of characters in the input buffer for each port. The range of characters is 1 ... 255.

Command Structure



NOTE: Words 1 through 11 for Command A are not used.

Response Structure



NOTE: Words 3 through 10 for Command A return a 0.

Response Structure for Illegal Commands

Response Structure

Word 0 Echo Command Word 0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Note: Bit 15 is the Status Word Valid bit.

•
•
•

Word 11 XXXX hex Module Status

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

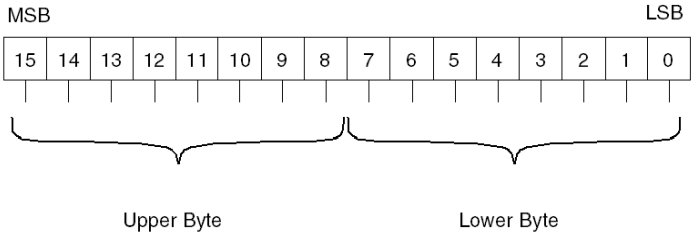
NOTE: Words 1 through 10 returns a 0.

Module Status Word (Word 11)

Overview

The Module Status Word (Word 11 in the response structure) contains valid module status information when bit 15 of Word 0 (in the response structure) is set. The state of this bit can be used to distinguish whether Word 11 in the response structure is being used for data or status.

Organization of the Status Word



NOTE: During normal operation, module status information is especially important when Word 11 is used for Module Status or Data Returned in the READ ASCII MESSAGE or GET DATA commands.

Content of the Status Word

Low Byte

Bit from Low Byte								Low Byte (Hex)	Description
7	6	5	4	3	2	1	0		
0	0	0	0	0	0	0	1	0001	Busy; command running on module
0	0	0	0	0	0	1	0	0002	Invalid message data during command run
0	0	0	1	0	0	0	0	0100	Register end during command run
0	0	1	0	0	0	0	0	0200	Serial buffer overrun error
0	1	0	0	0	0	0	0	0400	Checksum error on message in storage area see upper byte for message number
1	0	0	0	0	0	0	0	8000	Error; see upper byte for message number

High Byte

Bit from High Byte								High Byte (Hex)	Description
15	14	13	12	11	10	9	8		
0	0	0	0	0	0	0	1	0001	Invalid user logic parameter
0	0	0	0	0	0	1	0	0002	Invalid user logic command
0	0	0	1	0	0	0	0	0100	Count out of range
0	0	0	1	0	0	0	1	0101	Starting register out of range
0	0	0	1	0	0	1	0	0102	Ending register out of range
0	0	0	1	0	0	1	1	0103	Invalid register number order (end before start)
0	0	0	1	0	1	0	0	0104	Invalid serial port number requested
0	0	0	1	0	1	0	1	0105	Invalid message number requested
0	0	0	1	0	1	1	0	0106	Requested message number not programmed
0	0	0	1	0	1	1	1	0107	Requested message number in bad storage area
0	0	0	1	1	0	0	0	0108	Configuration parameter error
0	0	1	0	0	0	0	0	0200	Day of the week is incorrect

Reading beyond Valid Register Range

Overview

If the start register number and the data count are valid, but some of the registers to access are beyond the valid register range, only the data from the registers in the valid register range are read/written. The data count returned is the number of valid register data returned, and the error code 1280 Hex (end register number in out of range) is returned in the Module Status Word.

Example

The following example tries to read 10 registers, using the GET command, from the ESI module starting at register 3FFA Hex:

User Logic command = 030A Hex

Start register = 3FFA Hex

Therefore, the data count is 10 and the 6 valid registers (3FFA, 3FFB,3FFC, 3FFD, 3FFE, and 3FFF Hex) data are returned. The data count returned in the Command Word is 6 (8306 Hex).

The following data are assumed to be in the ESI Registers:

ESI Register	Content (Hex)
3FFA	1111
3FFB	2222
3FFC	3333
3FFD	4444
3FFE	5555
3FFF	6666

The following table shows the command sent to the ESI module and the response:

User Logic Command		User Logic Response	
Register	Content	Register	Content
4x+0	030A Hex	3x+0	8306 Hex
4x+1	3FFA Hex	3x+1	3FFA Hex
4x+2	0000 Hex	3x+2	1111 Hex
4x+3	0000 Hex	3x+3	2222 Hex
4x+4	0000 Hex	3x+4	3333 Hex
4x+5	0000 Hex	3x+5	4444 Hex
4x+6	0000 Hex	3x+6	5555 Hex
4x+7	0000 Hex	3x+7	6666 Hex
4x+8	0000 Hex	3x+8	0000 Hex

User Logic Command		User Logic Response	
Register	**Content**	**Register**	**Content**
4x+9	0000 Hex	3x+9	0000 Hex
4x+10	0000 Hex	3x+10	0000 Hex
4x+11	0000 Hex	3x+11	1280 Hex

Appendices

Overview

The Appendices provide additional information of general nature.

What Is in This Appendix?

The appendix contains the following chapters:

Chapter	Chapter Name	Page
A	Character Set	81
B	Introduction to ESI 062 10	85

Appendix A
Character Set

ASCII Character Set

Nonprintable ASCII Characters

The following table defines the ASCII character set in decimal, hexadecimal, character, and control character values.

Decimal	Octal	Hexadecimal	Character	Character Control
0	00	00	NUL	NULL
1	01	01	SOH	START OF HEADING
2	02	02	STX	START OF TEXT
3	03	03	ETX	END OF TEXT
4	04	04	EOT	END OF TRANSMISSION
5	05	05	ENQ	ENQUIRY
6	06	06	ACK	ACKNOWLEDGE
7	07	07	BEL	BEEP
8	10	08	BS	BACKSPACE
9	11	09	HT	HORIZONTAL TAB
10	12	0A	LF	LINE FEED
11	13	0B	VT	VERTICAL TAB (home)
12	14	0C	FF	FORM FEED
13	15	0D	CR	CARRIAGE RETURN
14	16	0E	SO	SHIFT OUT
15	17	0F	SI	SHIFT IN
16	20	10	DLE	DATALINK ESCAPE
17	21	11	DC1	DEVICE CONTROL ONE
18	22	12	DC2	DEVICE CONTROL TWO
19	23	13	DC3	DEVICE CONTROL THREE
20	24	14	DC4	DEVICE CONTROL FOUR
21	25	15	NAK	NEGATIVE ACKNOWLEDGE
22	26	16	SYN	SYNCHRONOUS IDLE
23	27	17	ETB	END OF TRANSMISSION BLOCK

Decimal	Octal	Hexadecimal	Character	Character Control
24	30	18	CAN	CANCEL
25	31	19	EM	END OF MEDIUM
26	32	1A	SUB	SUBSTITUTE
27	33	1B	ESC	ESCAPE
28	34	1C	FS	FILE SEPARATOR (cursor right)
29	35	1D	GS	GROUP SEPARATOR (cursor left)
30	36	1E	RS	RECORD SEPARATOR (cursor up)
31	37	1F	US	UNIT SEPARATOR (cursor down)

Printable ASCII Characters

The following table defines the ASCII set in decimal, hexadecimal and character.

Decimal	Octal	Hexa-decimal	Character		Decimal	Octal	Hexa-decimal	Character
32	40	20	SPACE		58	72	3A	:
33	41	21	!		59	73	3B	;
34	42	22	"		60	74	3C	<
35	43	23	#		61	75	3D	=
36	44	24	$		62	76	3E	>
37	45	25	%		63	77	3F	?
38	46	26	&		64	100	40	@
39	47	27	'		65	101	41	A
40	50	28	(		66	102	42	B
41	51	29	)		67	103	43	C
42	52	2A	*		68	104	44	D
43	53	2B	+		69	105	45	E
44	54	2C	,		70	106	46	F
45	55	2D	-		71	107	47	G
46	56	2E	.		72	110	48	H
47	57	2F	/		73	111	49	I
48	60	30	0		74	112	4A	J
49	61	31	1		75	113	4B	K
50	62	32	2		76	114	4C	L
51	63	33	3		77	115	4D	M
52	64	34	4		78	116	4E	N
53	65	35	5		79	117	4F	O
54	66	36	6		80	120	50	P
55	67	37	7		81	121	51	Q
56	70	38	8		82	122	52	R
57	71	39	9		83	123	53	S

Printable ASCII Character Set continued:

Decimal	Octal	Hexadecimal	Character		Decimal	Octal	Hexadecimal	Character
84	124	54	T		106	152	6A	j
85	125	55	U		107	153	6B	k
86	126	56	V		108	154	6C	l
87	127	57	W		109	155	6D	m
88	130	58	X		110	156	6E	n
89	131	59	Y		111	157	6F	o
90	132	5A	Z		112	160	70	p
91	133	5B	[		113	161	71	q
92	134	5C	\		114	162	72	r
93	135	5D	]		115	163	73	s
94	136	5E	^		116	164	74	t
95	137	5F	_		117	165	75	u
96	140	60	`		118	166	76	v
97	141	61	a		119	167	77	w
98	142	62	b		120	170	78	x
99	143	63	c		121	171	79	y
100	144	64	d		122	172	7A	z
101	145	65	e		123	173	7B	{
102	146	66	f		124	174	7C	|
103	147	67	g		125	175	7D	}
104	150	68	h		126	176	7E	~
105	151	69	i		127	177	7F	

Appendix B
Introduction to ESI 062 10

Introduction

This chapter provides an overview of the 140 ESI 062 10 ASCII communication module functionality and offers help to distinguish whether the module is appropriate for a given application.

What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
Introduction to ESI Module	86
Application Criteria	87
Module Description	88
ESI Module Block Diagram	90

Introduction to ESI Module

Overview

The Quantum ASCII interface module is a general purpose ASCII interface module providing the ability to communicate and exchange data with third party devices. These devices, typically, are found in industrial environments that do not utilize a standard communication method familiar to industrial automation. Such standard communication methods are using the industry standard Modbus communications, which defines the data query and response strings necessary, along with the physical interface required to communicate between programmable devices.

There are many communications standards and field busses available in industrial automation today. Few of these standards are based on RS 232C physical media for serial data streams. Much of the serial data information is not based on one of the available standards; therefore, the need for ASCII interfacing is required. ASCII communications are based on a custom serial protocol using RS232 or RS422/485 physical media.

Physical Media

Features of different physical media:

Standard	Maximum Distance	Physical Attributes	Data Rate Range
RS232	50 feet	Point to Point Multi drop using modems	180 bps to 19200 bps
RS422	400 feet	Point to Point Multi drop using modems	180 bps to 19200 bps
RS485	Wide Range	Multi drop (internal modems) 2 Wire or 4 Wire standards	180 bps to 19200 bps

Serial Device Applications

The majority of these ASCII applications talk directly to printers, bar code readers and scanners, serial devices such as weigh scales, meters and other measurement devices, as well as to other control systems used within the industrial automation application.

These third party devices require communications in a language they can understand in order to enable data transmission to occur between the third party device and the ASCII module.

For example, a scale measuring the total weight of a package, may respond to receiving a ’control A’ ASCII character <^A> by returning the package weight. This data is placed into the memory of the ASCII module, which in turn is read by the Quantum controller. The controller may need to make a logical decision of where the package should go if the weight is above a certain pre defined amount. The ASCII module therefore allows integration of data typically found within automation applications by simply knowing the protocol or language the foreign device needs in order to communicate.

Application Criteria

Introduction

The Quantum PLC family offers various solutions for communication with external devices. Depending on the needs of the application the user may select software solutions (XMIT function block using a CPU Modbus port) or hardware solutions (ESI module or ASCII Basic module). The following information helps to find the appropriate solution for a given application.

Application Criteria

The chart below identifies typical applications and the recommended product for that solution. As always when looking at solving application problems, this information is provided as a guide only and not the only answer to application problem.

Application	Description	Recommended Solution
Printer Interface	Generate local reports with imbedded data from the controller or the ASCII module.	ESI Module, J892, or ASCIIBasic Module
Communicate to simple Device	Send control characters and receive data from measurement devices.	ESI Module, J892, or XMIT
Bar Code Interface	Send and receive data from bar code reader/scanner.	ESI Module or ASCII Basicmodule
Communicate to Device	Send control characters and receive data from measurement devices, leading zero's or leading spaces may be sent by the device.	ESI Module or J892
Controller to Controller Interfacing	Emulate manufacturers protocol which supports several sub functions. Protocol. Generation for sophisticated device protocol.	ASCII Basic Module
External Data Storage	Store data outside of the controller.	ESI Module or ASCII Basic module
Modbus Master and/or Modem Support	Generate full spectrum of Modbus master commands and/or support dial up modems with control characters.	XMIT Function block and controllers local Modbus port
Multiple RS-232 ports	Multiple ports to communicate with external devices are required	ESI Module or ASCII Basic module
RS-232 ports in Distributed I/O	External devices have to be connected to Distributed I/O	ESI Module or ASCII Basic module

Module Description

Overview

The ESI module consists of 5 major functional elements:

- Serial ports for device communication
- Interface to the Quantum controller through the backplane
- Port buffer
- Register memory
- ASCII message storage memory
- Firmware

Serial Ports

The ESI module has implemented 3 logical communication ports. Port 1 and Port 2 are used to communicate to external serial devices while Port 0 is used for programming the module. Port 0 and Port 1 share one physical port. All 3 ports can be set up independently. For a detailed description of the port setup see *Port Command, page 44*.

Interface to Quantum Controller

The ESI module exchanges data with the Quantum controller through the use of 12 output words for commands and data from the Quantum controller and 12 input words for data to the Quantum controller and command echo and status information. For detailed information about the structure of the command and response structures see *ESI Command Word, page 51*.

Port Buffer

The 2 physical ports of the ESI module have an input and an output buffer of 255 characters each. The device side of those buffers is maintained automatically by the optional XON/XOFF handshake. For data transfer from and to the Quantum controller, for buffer control and status testing several commands are available which are described in detail in *Data Flow, page 36*.

Register Memory

The ESI module has a 32 kbyte memory which is organized as 16k 16-bit registers. These registers hold all data coming from and going to the serial ports. They can be accessed by the PUT and the GET command.

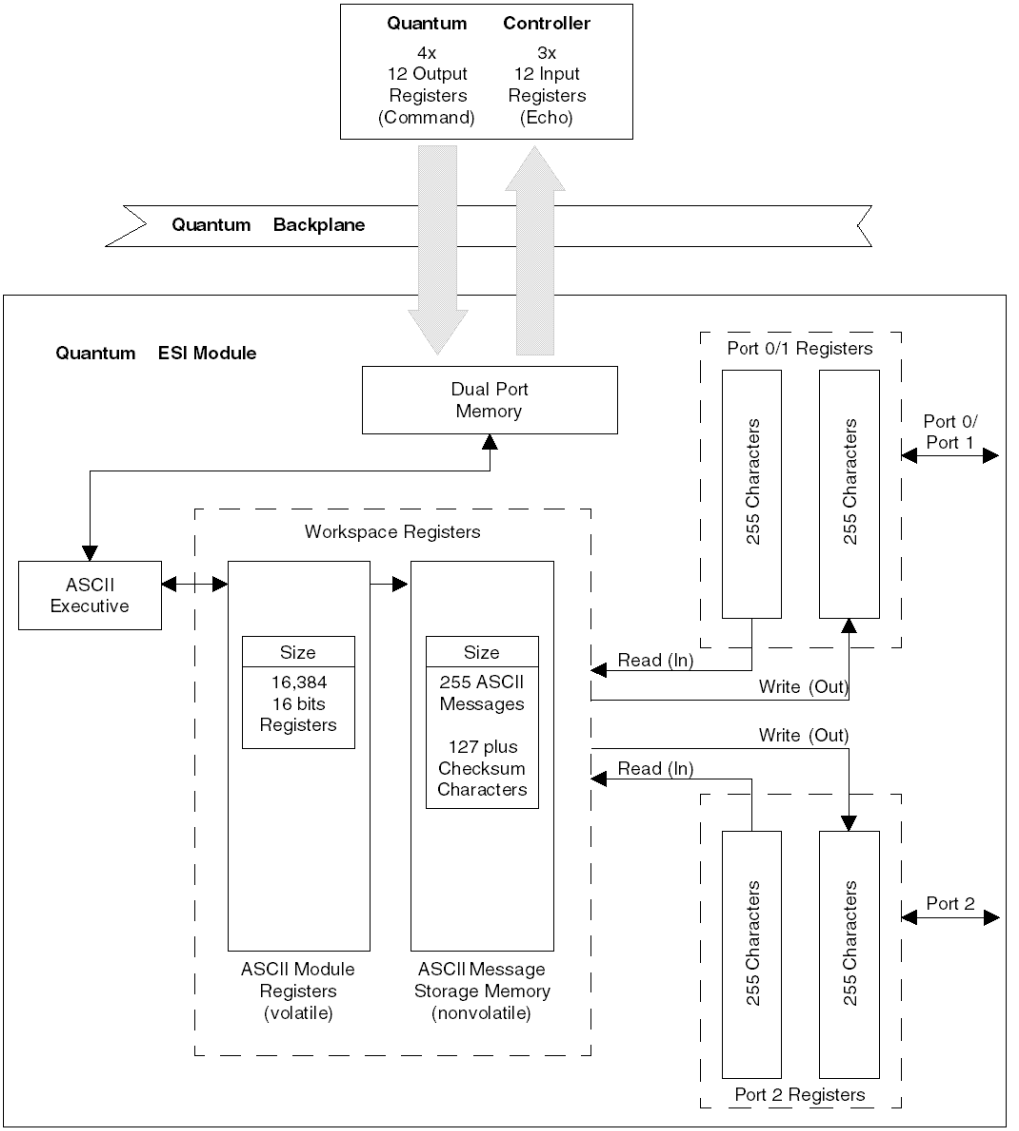
ASCII Message Storage

The ESI module can hold up to 255 ASCII messages with 127 characters plus checksum character each. These ASCII messages can be either static texts to be sent to an external device or a definition of how data contained in the register area is to be translated into or from a stream of serial ASCII characters, or a combination of both.

Firmware

The firmware of the ESI module can be loaded over the local I/O backplane. Upgrades and changes in functionality are supported by updating the flash executive firmware within the ESI module. Users should be aware that the update procedure can only occur over the local I/O backplane, even though the module can be placed in local, remote, or distributed locations. If you are using the ESI module in remote or distributed backplanes, plan on having an empty slot available in the local backplane, or a spare controller system to accommodate future executive upgrades.

ESI Module Block Diagram



Index

0-9
140ESI06210, *11*, *85*

A
aborting read/write messages, *72*
addressing
  flat, *21*, *22*
  topological, *21*
ASCII character set, *81*

B
bit order for discrete I/O, *21*

C
command line editors, *43*
commands, *49*
configuring ASCII interface modules, *27*
crash codes, *14*

E
ESI commands
  ABORT, *72*
  FLUSH BUFFER, *71*
  GET BUFFER STATUS, *73*
  GET DATA, *60*
  GET TOD, *64*
  NO OPERATION, *54*
  PUT DATA, *62*
  READ ASCII MESSAGE, *55*
  SET MEMORY REGISTERS, *69*
  SET TOD, *66*
  WRITE ASCII MESSAGE, *57*

F
flushing the input buffer, *71*

G
getting data from the module, *60*

M
message formats, *30*

N
NO OPERATION, *54*

R
reading ASCII messages, *55*
reading characters in the input buffer, *73*
reading module's time of day clock, *64*

S
setting memory registers, *69*
setting module's time of day clock, *66*

T
time of day clock, *64*, *66*

W
writing ASCII messages, *57*
writing data to the module, *62*