

# MODBUS COMMUNICATIONS MODEL 2500

---

**DANIEL FLOW PRODUCTS, INC.  
HOUSTON, TEXAS**

**Part Number: 3-9000-545  
REVISION D**

**November 1992**

**DANIEL** *Electronics*  
Flow Products, Inc



**THE DANIEL INDUSTRIES, INC.  
MODBUS COMMUNICATIONS  
MODEL 2500 HOST-SLAVE COMMUNICATIONS  
REFERENCE GUIDE**

**NOTICE**

DANIEL INDUSTRIES, INC. AND DANIEL FLOW PRODUCTS, INC. ("DANIEL") SHALL NOT BE LIABLE FOR TECHNICAL OR EDITORIAL ERRORS IN THIS MANUAL OR OMISSIONS FROM THIS MANUAL. **DANIEL MAKES NO WARRANTIES, EXPRESS OR IMPLIED, INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE WITH RESPECT TO THIS MANUAL AND, IN NO EVENT, SHALL DANIEL BE LIABLE FOR ANY SPECIAL OR CONSEQUENTIAL DAMAGES INCLUDING, BUT NOT LIMITED TO, LOSS OF PRODUCTION, LOSS OF PROFITS, ETC.**

PRODUCT NAMES USED HEREIN ARE FOR MANUFACTURER OR SUPPLIER IDENTIFICATION ONLY AND MAY BE TRADEMARKS/REGISTERED TRADEMARKS OF THESE COMPANIES.

COPYRIGHT © 1993  
BY DANIEL FLOW PRODUCTS, INC.  
HOUSTON, TEXAS, U.S.A.

*All rights reserved. No part of this work may be reproduced or copied in any form or by any means - graphic, electronic or mechanical - without first receiving the written permission of Daniel Flow Products, Inc., Houston, Texas, U.S.A.*

## **WARRANTY**

Daniel Flow Products, Inc. ("Daniel") warrants all equipment manufactured by it to be free from defects in workmanship and material, provided that such equipment was properly selected for the service intended, properly installed, and not misused. Equipment which is returned, transportation prepaid to Daniel within twelve (12) months of the date of shipment (eighteen (18) months from date of shipment for destinations outside of the United States), which is found after inspection by Daniel to be defective in workmanship or material, will be repaired or replaced at Daniel's sole option, free of charge, and return-shipped at lowest cost transportation. All transportation charges and export fees will be billed to the customer. Warranties on devices purchased from third party manufacturers not bearing a Daniel label shall have the warranty provided by the third party manufacturer.

*Extended warranty* - Models 2470, 2480 and 2500 are warranted for a maximum of twenty-four (24) months. The Danalyzer valves are warranted for the life of the instrument and the columns for five years.

*The warranties specified herein are in lieu of any and all other warranties, express or implied, including any warranty of merchantability or fitness for a particular purpose.*

Daniel shall be liable only for loss or damage directly caused by its sole negligence. Daniel's liability for any loss or damage arising out of, connected with, or resulting from any breach hereof shall in no case exceed the price allocable to the equipment or unit thereof which gives rise to the claim. Daniel's liability shall terminate one year after the delivery of the equipment except for overseas deliveries and extended warranty products as noted above.

In no event, whether as a result of breach of warranty or alleged negligence, shall Daniel be liable for special or consequential damages, including, but not limited to, loss of profits or revenue; loss of equipment or any associated equipment; cost of capital; cost of substitute equipment, facilities or services; downtime costs; or claims of customers of the purchaser for such damages.

**TABLE OF CONTENTS**

**SECTION 1    K.2 IMPLEMENTATIONS    . . . . . ES-10212-01**

**SECTION 2    L.8 IMPLEMENTATIONS    . . . . . ES-10212-02**

**SECTION 3    SET CLOCK COMMAND    . . . . . ES-10212-03**

**SECTION 4    MODEL 2251 CHROMATOGRAPH  
MODBUS INDICES    . . . . . ES-17128-05**

*This page intentionally left blank.*

**Engineering Specification**  
**2500 Host-Slave Communications**  
**Software Requirements**

**Revision D**

The information contained in this specification is proprietary to Daniel Industries, Inc. and may not be released without the written permission of the Electronics Division.

				TITLE: <b>2500 Host-Slave Communications Software Requirements</b>	DRN <b>Vonnie</b>	DATE <b>6/16/93</b>
					CHKD <b>E.M.</b>	DATE <b>12/90</b>
					APPD <b>E.D.</b>	DATE <b>12/90</b>
					REL <b>DLT</b>	DATE <b>12/90</b>
<b>D</b>	<b>7110</b>	<b>12/90</b>	<b>EM</b>	PART NO.	DRAWING NO. <b>ES-10212-01</b>	SHEET <b>1</b> of <b>41</b>
REV.	ECO No.	DATE	BY			

# MODBUS DATA ACQUISITION SPECIFICATION

## 1.0 PREFACE

Modbus is an industrial communications and distributed control system that integrates programmable controllers, computers, terminals and other monitoring, sensing and control devices.

Modbus and Modicon are trademarks of Gould Incorporated. The Modbus communications protocol is fully described in the reference guide entitled "Gould Modicon Modbus Protocol" publication PI-MBUS-300 and available from: Gould Incorporated, Programmable Control Division, P. O. Box 3083, Andover, Massachusetts 01810.

Modbus has been adapted for use by Daniel on the 2500 Series computers with permission of Gould Incorporated.

## 2.0 GENERAL

This specification defines the requirements for the Modbus Data Acquisition Software implemented and supported on the Model 2500 computer. The software is designed to operate on a one or two board system and supports a subset of the Gould Modicon Modbus Protocol for Host-Slave Communication. The software is written in PLM/86 and becomes part of Base25.

The hardware specifications of the 2500 computer are as specified on ES-10240-00. The software expects to have one serial port, either RS-232 or RS-485 dedicated for communications.

## 3.0 MODBUS SUPPORT

The Modbus protocol provides for one master and up to 247 slaves on a common line, with each slave assigned a fixed unique device address in the range of 1 to 247. Certain device restrictions may limit the number of slaves to less than 247.

Only the Master may initiate a transaction. The transactions are either a query/response type with only a simple slave addressed (address of 1 to 247) or a broadcast/no response with all slaves being addressed (address = 0). Only messages which do not need a response may be set as broadcast.

TITLE: <b>2500 Host-Slave Communications Software Requirements</b>	DRAWING NO. <b>ES-10212-01</b>	REVISION LEVEL <b>D</b>
	FILE NO.	SHEET <u>2</u> of <u>41</u>

### 3.1 COMMUNICATION TRANSMISSION MODES

There are two modes of transmission available for use in the Modbus system. Both modes provide the same capabilities of communication with slaves; but the mode is selected depending on the equipment used as a Master Host. Only one mode may be used at a time in a system. The modes are ASCII, and RTU (Remote Terminal Unit). The modes are defined in Table 1. The Phase I implementation will only include the ASCII protocol.

<u>CHARACTERISTIC</u>	<u>ASCII (7-Bit)</u>	<u>RTU (8-Bit)</u>
Coding System	Hexadecimal (Using only the ASCII characters 0 to 9 and A to F).	8 bit Binary
No. Bits/Char.		
Start bits	1	1
Data Bits (LSF)	7	8
Parity (Optional)	1 (1 even or odd) 0 (no parity)	1 (1 even or odd) 0 (no parity)
Stop bits	1 or 2	1 or 2
Error checking	LRC	CRC

**Table 1**

TITLE: <b>2500 Host-Slave          Communications Software          Requirements</b>	DRAWING NO. <b>ES-10212-01</b>	REVISION LEVEL <b>D</b>
	FILE NO.	SHEET <b>3</b> of <b>41</b>

### 3.2 2500 HOST-SLAVE PROTOCOL

The 2500 Host-Slave protocol is tailored to the type of communications that an industrial user needs for a network of controllers. In general, the interpretation of fields within a message are identical between the RTU and ASCII transmission modes (see Figures 1 and 2). The major differences are in the types of error checking performed on the message and that the ASCII message has approximately twice the number of characters used by the RTU.

Framing in ASCII transmission mode is accomplished by the use of the unique colon (:) character to indicate beginning of frame and carriage return (CR) line feed (LF) to delineate end of frame. The line feed character also serves as a synchronizing character which indicates that the transmitting host is ready to receive an immediate reply (see Figure 1).

BEG OF FRAME	ADDRESS	FUNCTION	DATA	ERROR CHECK	EOF	READY TO REC RESP
:	2-CHAR 16-BITS	2-CHAR 16-BITS	N X 4-CHAR N X 16-BITS	2-CHAR 16 BITS	CR	LF

CHAR = CHARACTER; 1 CHARACTER = 7 DATA BITS, 1 START BIT, 1 OR 2 STOP BITS AND OPTIONALLY - 1 PARITY BIT

**Figure 1 - ASCII Message Frame Format**

Frame synchronization can be maintained in RTU transmission mode only by simulating a synchronous message. The receiving device monitors the elapsed time between receipt of characters. If three and one-half character times elapse without a new character or completion of the frame, then the device flushes the frame and assumes that the next byte received will be an address (see Figure 2).

T1 T2 T3	ADDRESS	FUNCTION	DATA	CHECK	EOF	T1 T2 T3
	8-BITS	8-BITS	N X 8-BITS	16-BITS	16-BITS	

**Figure 2 - RTU Message Frame Format**

TITLE: <b>2500 Host-Slave Communications Software Requirements</b>	DRAWING NO. <b>ES-10212-01</b>	REVISION LEVEL <b>D</b>
	FILE NO.	SHEET <b>4</b> of <b>41</b>

The address field immediately follows the beginning of frame and consists of 8-bits (RTU) or 2 characters (ASCII). These bits indicate the user assigned address of the slave device that is to receive the message sent by the attached master.

Each slave must be assigned a unique address and only the addressed slave will respond to a query that contains its address. When the slave sends a response, the slave address informs the master which slave is communicating. In a broadcast message, an address of 0 is used. All slaves interpret this as an instruction to read and take action on the message, but not to issue a response message.

The function code field tells the addressed slaves what function to perform. The function codes are specifically designed for interacting with the Gould Modicon series of programmable controllers on a Modbus industrial communications system. Daniel has adopted a usage of the function codes to allow compatibility between the programmable controllers and the 2500 computer. Table 2 lists the function codes, their meaning, and the action they initiate. Also tabulated is the 2500 usage of the function codes.

The high order bit of the function code field is set by the slave device to indicate that other than a normal response is being transmitted to the Master device. (See Section 3.3 for a description of exception responses.) This bit remains 0 if the message is a query or a normal response message.

TITLE: <b>2500 Host-Slave Communications Software Requirements</b>	DRAWING NO. <b>ES-10212-01</b>	REVISION LEVEL <b>D</b>
	FILE NO.	SHEET <u>5</u> of <u>41</u>

**MODBUS FUNCTION CODES SUPPORTED BY 2500**

**TABLE 2**

<u>FUNCTION CODE</u>	<u>MODBUS DESCRIPTION</u>	<u>2500 USAGE</u>	<u>IMPLEMENTATION</u>
1	Read Coil Status	Read Boolean Status or read manual/live flag in numeric variable	Y
2	Read Input Status	-	N
3	Read Hold Registers	Read Multiple Variables (see note *)	Y
4	Read Input Registers	-	N
5	Force Coil	Set Boolean Variable or Set Manual/Live Flag in numeric variable	Y
6	Load Register	Set single 16 Bit Integer	Y
7	Read Exception Status	Read 2500 Status	Y
8	Loop Back Diagnostic	Loop Back Diagnostic	-
9	Program 484	-	N
10	Poll 484	-	N
11	Comm Event Counter	Comm Event Counter	-
12	Comm Event Log	Comm Event Log	-
13	Program - General	-	N
14	Poll - General	-	N
15	Force Multiple Coils	Set Multiple Boolean Variables	Y

TITLE:  
**2500 Host-Slave  
 Communications Software  
 Requirements**

DRAWING NO.  
**ES-10212-01**

REVISION LEVEL  
**D**

FILE NO.

SHEET 6 of 41

<u>FUNCTION CODE</u>	<u>MODBUS DESCRIPTION</u>	<u>2500 USAGE</u>	<u>IMPLEMENTATION</u>
16	Load Multiple Registers	Load Variables (See Note*)	Y
17	Report Slave ID	Report 2500 Status	-
18	Program	2500 Programming	Y
19	Reset Comm Link	-	N

---

**\* NOTE:** Depending on the definition of the variable type by the ConFig25 application as 16 bit integer, 32 bit integer or 32 bit floating point, will determine if multiple variables are set as 16 bit integers or Double word variables.

---

**Table 2 (Continued)**

<b>TITLE:</b> <b>2500 Host-Slave  Communications Software  Requirements</b>	<b>DRAWING NO.</b> <b>ES-10212-01</b>	<b>REVISION LEVEL</b> <b>D</b>
	<b>FILE NO.</b>	<b>SHEET <u>7</u> of <u>41</u></b>

### 3.3 MODBUS EXCEPTION RESPONSES

Operation errors are those involving illegal data in a message, or difficulty in communicating. These errors result in an exception response from either the master computer or the slave, depending on the type of error. The exception response codes are listed in Table 3. When a slave detects one of these errors, it sends a response message to the master consisting of slave address, function code, error code and error check fields. To indicate that the response is a notification of an error, the high order bit of the function code is set to 1. Figures 3 and 4 give an example of an incorrect query and the subsequent exception response codes.

TITLE: <b>2500 Host-Slave Communications Software Requirements</b>	DRAWING NO. <b>ES-10212-01</b>	REVISION LEVEL <b>D</b>
	FILE NO.	SHEET <b>8</b> of <b>41</b>

<u>CODE</u>	<u>NAME</u>	<u>MEANING</u>
01	ILLEGAL FUNCTION	The message function received is not an allowable action for addressed slave. If a poll command was issued, indicates no program function preceded it.
02	ILLEGAL DATA ADDRESS	The address referenced in the data field is not an allowable address for the addressed slave location.
03	ILLEGAL DATA VALUE	The value referenced in the data field is not allowable in the addressed slave location.
04	FAILURE IN ASSOCIATED DEVICE	The slave's sub-controller has failed to respond to a message or an abortive error occurred. (See Note 1.)
05	ACKNOWLEDGE	The slave has accepted and is processing the long duration program command. Issue a POLL PROGRAM COMPLETE message to find out when processing is finished. A poll message sent to the slave before it is finished will result in a rejected message response.
06	BUSY, REJECTED MESSAGE	The message was received without error, but the slave is engaged in processing a long duration program command. Retransmit later, when the slave may be free.

**NOTE 1:** NOTE than an EXCEPTION Code 4 may indicate that only part of the associated query was processed before an irrecoverable error occurred within the responder station. Receipt of EXCPT Code 4 requires immediate supervisory notification.

**Table 3**

TITLE: <b>2500 Host-Slave          Communications Software          Requirements</b>	DRAWING NO. <b>ES-10212-01</b>	REVISION LEVEL <b>D</b>
	FILE NO.	SHEET <b>9</b> of <b>41</b>

### QUERY MESSAGE

SLAVE ADDR	FUNC	H.O. START ADDR	L.O. START ADDR	H.O. NO. OF FLAG	L.O. NO. OF FLAG	ERROR CHECKFIELD	
0A	01	04	A1	00	01	4F	LRC

**Figure 3 - Read Boolean Status Query Message**

This query requests the status of Input 1185 (04A1 Hex) in Slave No. 10 (0A Hex). Since the defined index numbers are less than 1000, the following error response might be generated:

SLAVE ADDR	FUNC	EXCEPTION CODE	ERROR CHECK	
0A	81	02	73	LRC

**Figure 4 - Read Boolean Status Error Response**

The function code field is the original function code with the high order bit set (1000 0001 binary = 81 Hex). Exception Code 02 indicated an illegal data address.

TITLE: <b>2500 Host-Slave          Communications Software          Requirements</b>	DRAWING NO. <b>ES-10212-01</b>	REVISION LEVEL <b>D</b>
	FILE NO.	SHEET <u>10</u> of <u>41</u>

### 3.4 ERROR DETECTION

The Modbus protocol provides several levels of error checking in order to assure the quality of the data transmission. To detect multibit errors where the parity has not changed, the system uses redundancy checks: Cyclic Redundancy Check (CRC) and Longitudinal Redundancy Check (LRC). Which redundancy check is used is dependent upon the mode of transmission. RTU mode uses the cyclical check and the ASCII mode uses the longitudinal check. The generation of CRC and LRC is explained in Section 3.4.1 and 3.4.2.

#### 3.4.1 CRC-16 (Cyclic Redundancy Check) Error Check Sequence

The CRC-16 error check sequence is implemented as described in the following paragraphs.

The message (data bits only, disregarding start/stop and optional parity bits) is considered as one continuous binary number whose most significant bit (MSB) is transmitted first. The message is pre-multiplied by  $X^{16}$  (shifted left 16 bits), then divided by  $X^{16} + X^{15} + X^2 + 1$  expressed as a binary number (1100000000000101). The integer quotient digits are ignored and the 16-bit remainder (initialized to all ones at the start to avoid the case of all zeros being an accepted message) is appended to the message (MSB first) as the two CRC check bytes. The resulting message including CRC, when divided by the same polynomial ( $X^{16} + X^{15} + X^2 + 1$ ) at the receiver will give a zero remainder if no errors have occurred. (The receiving unit recalculates the CRC and compares it to the transmitted CRC). All arithmetic is performed modulo two (no carries). An example of the CRC-16 error check for message HEX 0207 (address 2, function 7 or a status request to slave number 2) is given by the following example.

The device used to serialize the data for transmission will send the conventional LSB or right-most bit of each character first. In generating the CRC, the first bit transmitted is defined as the MSB of the dividend. For convenience then, and since there are no carries used in arithmetic, let's assume while computing the CRC that the MSB is on the right. To be consistent, the bit order of the generating polynomial must be reversed. The MSB of the polynomial is dropped since it affects only the quotient and not the remainder. This yields 1010 0000 0000 0001 (Hex A001). Note that this reversal of the bit order will have no affect whatever on the interpretation or bit order of characters external to the CRC calculations.

TITLE: <b>2500 Host-Slave Communications Software Requirements</b>	DRAWING NO. <b>ES-10212-01</b>	REVISION LEVEL <b>D</b>
	FILE NO.	SHEET <b>11</b> of <b>41</b>

The step by step procedure to form the CRC-16 check bytes is as follows:

1. Load a 16-bit register with all 1's.
2. Exclusive OR the first 8-bit byte with the high order byte of the 16-bit register, putting the result in the 16-bit register.
3. Shift the 16-bit register one bit to the right.
- 4 a. If the bit shifted out to the right (flag) is one, exclusive OR the generating polynomial 1010 000 000 0001 with the 16-bit register.
- 4 b. If the bit shifted out to the right is a zero; return to step 3.
5. Repeat steps 3 and 4 until 8 shifts have been performed.
6. Exclusive OR the next 8-bit byte with the 16-bit register.
7. Repeat step 3 through 6 until all bytes of the message have been exclusive OR with the 16-bit register and shifted 8 times.
8. The contents of the 16-bit register are the 2 byte CRC error check and is added to the message most significant bits first.

		16- BIT REGISTER		MSB	FLAG
(Exclusive or)	1111	1111	1111	1111	
0			0000	0010	
	1111	1111	1111	1101	
Shift 1	0111	1111	1111	1110	1
Polynomial	1010	0000	0000	0001	
	1101	1111	1111	1111	
Shift 2	0110	1111	1111	1111	1
Polynomial	0101	0000	0000	0001	
	1100	1111	1111	1110	
Shift 3	0110	0111	1111	1111	0
Shift 4	0011	0011	1111	1111	1
Polynomial	1010	0000	0000	0001	

**Example - CRC-16 Generation - Read Exception Status of Slave 02**

TITLE: <b>2500 Host-Slave Communications Software Requirements</b>	DRAWING NO. <b>ES-10212-01</b>	REVISION LEVEL <b>D</b>
	FILE NO.	SHEET <u>12</u> of <u>41</u>

		16-BIT REGISTER		MSB	FLAG
Shift 5	1001	0011	1111	1110	0
	0100	1001	1111	1111	
Shift 6 Polynomial	0010	0100	1111	1111	1
	1010	0000	0000	0001	
Shift 7	1000	0100	1111	1110	0
	0100	0010	0111	1111	
Shift 8 Polynomial	0010	0001	0011	1111	1
	1010	0000	0000	0001	
07	1000	0001	0011	1110	
			0000	0111	
Shift 1	1000	0001	0011	1001	1
	0100	0000	1001	1100	
Polynomial	1010	0000	0000	0001	
Shift 2	1110	0000	1001	1101	1
	0111	0000	0100	1110	
Polynomial	1010	0000	0000	0001	
Shift 3	1101	0000	0010	1111	1
	0110	1000	0010	0111	
Polynomial	1010	0000	0000	0001	
Shift 4	1100	1000	0010	0110	0
	0110	0100	0001	0011	
Shift 5 Polynomial	0011	0010	0000	1001	1
	1010	0000	0000	0001	
Shift 6	1001	0010	0000	1000	0
	0100	1001	0000	0100	
Shift 7	0010	0100	1000	0010	0
Shift 8	0001	0010	0100	0001	0

**Example - CRC-16 Generation - Read Exception Status of Slave 02  
(Continued)**

TITLE: <b>2500 Host-Slave Communications Software Requirements</b>	DRAWING NO. <b>ES-10212-01</b>	REVISION LEVEL <b>D</b>
	FILE NO.	SHEET <b>13</b> of <b>41</b>

HEX 12

HEX 41

TRANSMITTED MESSAGE WITH CRC-16  
(MESSAGE SHIFTED TO RIGHT TO TRANSMIT)

0001      12      0010 0100      41      0001      0000      07      1111 0000      02      0010

Last Bit  
Transmitted

TRANSMISSION ORDER

First Bit  
Transmitted

Example - CRC-16 Generation - Read Exception Status of Slave 02  
(Continued)

TITLE:  
**2500 Host-Slave  
Communications Software  
Requirements**

DRAWING NO.  
**ES-10212-01**

REVISION LEVEL  
**D**

FILE NO.

SHEET 14 of 41

### 3.4.2 LRC (Longitudinal Redundancy Check) Error Check Sequence

The error check sequence for the ASCII mode is LRC. The error check is an 8-bit binary number represented and transmitted as two ASCII hexadecimal (hex) characters. The error check is produced by converting the hex characters to binary, adding the binary characters without wraparound carry, and two's complementing the result. At the received end the LRC is recalculated and compared to the sent LRC. The colon, CR, LF, and any imbedded non-ASCII hex characters are ignored in calculating the LRC.

Address	0	2	0000	0010
Function	0	1	0000	0001
Start Add H.O.	0	0	0000	0000
Start Add L.O.	0	0	0000	0000
Quantity of Pts	0	0	0000	0000
	0	8	+ 0000	1000
			0000	1011

1's Complement	1111	0100
	+1	<u>1</u>
2's Complement	1111	0101

Error Check	F	5	F	5
-------------	---	---	---	---

Receiving PC sums up all data bytes received including the LRC at the end. The 8-bit should all equal zero. (Note, the sum may exceed 8-bits - only the low order of bits should be saved.)	0000	0010
	0000	0001
	0000	0000
	0000	0000
	0000	0000
OK	0000	1000
<u>Error Check</u>	1111	0100
SUM	0000	0000

Example - LRC Generation - Read First 8 Flags from Slave 02

TITLE:  
**2500 Host-Slave  
 Communications Software  
 Requirements**

DRAWING NO.  
**ES-10212-01**

REVISION LEVEL  
**D**

FILE NO.

SHEET 15 of 41

#### 4.0 DETAILED EXPLANATION OF MODBUS FUNCTIONS SUPPORTED

The purpose of this section is to define the general format for the specific commands available to programmers of the Modbus System. The form of each query message, an example in ASCII transmission mode and an explanation of the function the query message performs is provided. Normal response messages are also included.

Messages with Function Codes 1-6, 15 & 16 indicate specifically which location(s) in the controller is (are) to be accessed. Function Codes 1, 5, & 15 refer to Boolean variables, and Codes 3, 6, and 16 refer to numeric (16 bit, 32 bit or 32 bit floating point) variables. All address references in the Modbus messages are numbered relative to the type of variable as defined by the ConFig25 application..

The examples given in this section will present the protocol independent of RTU or ASCII framing considerations. See Figures 1 and 2 for ASCII & RTU framing examples. The programmer implementing the software may use the following method to correctly frame the protocol for his or her specific application.

The protocol will be presented as much as possible throughout this section in the format shown in Figure 5. Numbers represented are in hexadecimal.

ADDR	FUNC	DATA START INDEX HO	DATA START INDEX LO	DATA # OF INDEXS HO	DATA # OF INDEXS LO	ERRORC HECKFI ELD	
06	03	00	6B	00	03	89	LRC

**Figure 5 - Presentation Example for Protocol**

TITLE: <b>2500 Host-Slave Communications Software Requirements</b>	DRAWING NO. <b>ES-10212-01</b>	REVISION LEVEL <b>D</b>
	FILE NO.	SHEET <b>16</b> of <b>41</b>

The example given is Read Output Registers 108-110 for the 2500 Slave Interface Unit addressed as 06. This message when specifically formatted for either RTU or ASCII will look as follows:

<u>QUERY MESSAGE</u>	<u>RTU</u>		<u>ASCII</u>	
Header	None			Colon
Address	0000	0110	0	6
Function	0000	0011	0	3
Starting H.O.	0000	0000	0	0
Variable I.O.	0110	1011	6	B
Quantity of H.O.	0000	0000	0	0
Registers L.O.	0000	0011	0	3
Error	0111	0101	8	9
Check	1010	0000		
Trailer	None		CR	LF
	8 Bytes Total		17 Bytes Total	

<u>RESPONSE MESSAGE</u>	<u>RTU</u>		<u>ASCII</u>		
Header	None			Colon	
Address	0000	0110	0	6	
Function	0000	0011	0	3	
Byte Count	0000	0110	0	6	
	H.O.	0000	0010	0	2
	L.O.	0010	1011	2	B
Data	H.O.	0000	0000	0	0
	H.O.	0000	0000	0	0
	L.O.	0110	0011	6	3
Error	CRC		6	1	
Check					
Trailer	None		CR	LF	
	11 Bytes Total		23 Bytes Total		

TITLE: <b>2500 Host-Slave          Communications Software          Requirements</b>	DRAWING NO. <b>ES-10212-01</b>	REVISION LEVEL <b>D</b>
	FILE NO.	SHEET <u>17</u> of <u>41</u>

It will always be the case that the ASCII message in its final form will be approximately twice the length in bytes of the comparable RTU message.

The Normal Response message format has an implied or explicit message length depending on the Modbus function code. The explicit form is used for variable length Responses. The byte count in the response refers to the number of 8-bit (1 byte) data fields included in the response. In the RTU communication mode, a byte of information is transmitted as a single 8-bit character. In the ASCII communication mode, each 8-bit ASCII character contains 4 data bits and 4 bits for formatting the specific ASCII character sent, consequently two ASCII characters are required to transmit 8 bits (1 byte) of information.

This general response format is used when responses are of a variable nature. For fixed length responses the byte count field is omitted and only the data fields are supplied.

**READ BOOLEAN STATUS (FUNCTION CODE 01)  
QUERY**

This function allows the user to obtain the ON/OFF status of logic Booleans used to control discrete outputs from the addressed slave 2500 or return the manual/live status of a defined numeric variable. Broadcast mode is not supported with this function code. In addition to the slave address and function fields, the message requires that the information field contain the initial Boolean address to be read (Starting Address) and the number of locations that will be interrogated to obtain status data.

The addressing allows up to 2000 Boolean variables to be obtained at each request; however, the specific slave device may have restrictions that lower the maximum quantity. The Booleans are numbered from 1001; (Boolean number 1 = 1001, Boolean number 2 = 1002, Boolean number 3 = 1003, etc.).

Figure 6 is a sample read output Status Request to read Booleans 1020 to 1056 from slave device number 17.

ADDR	FUNC	DATA START PT HO	DATA START PT LO	DATA # OF PTS HO	DATA # OF PTS LO	ERROR CHECK FIELD	
11	01	03	FC	00	25	CA	LRC

**Figure 6 - Read Boolean Status Query Message**

TITLE: <b>2500 Host-Slave Communications Software Requirements</b>	DRAWING NO. <b>ES-10212-01</b>	REVISION LEVEL <b>D</b>
	FILE NO.	SHEET <b>18</b> of <b>41</b>

## RESPONSE

An example response to Read Output Status is as shown in Figure 7. The data is packed one bit for each Boolean flag variable. The response includes the slave address, function code, quantity of data characters, the data characters, and error checking. Data will be packed with one bit for each Boolean flag (1 = ON, 0 = OFF). The low order bit of the first character contains the addressed flag, and the remainder follow. For Boolean quantities that are not even multiples of eight, the last characters will be filled in with zeros at high order end. The quantity of data characters is always specified as quantity of RTU characters, i.e. the number is the same whether RTU or ASCII is used.

Since the slave 2500 interface device is serviced at the end of the calculation scan, data will reflect Boolean status at the end of the scan. Some slaves will limit the quantity of Booleans provided each scan; thus, for large Boolean quantities, multiple PC transactions must be made using Boolean status from sequential scans.

ADDR	FUNC	BYTE COUNT	DATA FLAG STATUS 120-127	DATA FLAG STATUS 128-135	DATA FLAG STATUS 136-143	DATA FLAG STATUS 144-151	DATA FLAG STATUS 152-156	ERROR CHECK FIELD	
11	01	05	CD	6B	B2	OE	1B	D6	LRC

**Figure 7 - Read Status Response Message**

The status of Booleans 120-127 is shown as CD(HEX) = 1100 1101 (Binary). Reading left to right, this shows that Booleans 127, 126, 123, 122, and 120 are all on. The other Boolean data bytes are decoded similarly. Due to the quantity of Boolean statuses requested, the last data field, which is shown as 1B (HEX) = 0001 1011 (Binary), contains the status of only 5 coils (152-156) instead of 8 flags. The 3 left-most bits are provided as zeros to fill the 8-bit format

## READ VARIABLES (FUNCTION CODE 03)

### QUERY

Read Numeric Variables (03) allows the user to obtain the binary contents of numeric variables in the addressed slave.

These variables can store the numerical values of associated timers and counters which can be driven to external devices.

TITLE:  
**2500 Host-Slave  
 Communications Software  
 Requirements**

DRAWING NO.  
**ES-10212-01**

REVISION LEVEL  
**D**

FILE NO.

SHEET **19** of **41**

The addressing allows up to 125 16 bit variables to be obtained at each request; however, the specific slave device may have restrictions that lower this maximum quantity. The Indexes are numbered from 3001 (3001 = zero, 3002 = one, etc.).

Broadcast mode is not allowed.

The below example reads variables 3008 through 3010 from slave number 17.

ADDR	FUNC	DATA START INDEX HO	DATA START INDEX LO	DATA # OF INDEX HO	DATA # OF INDEX LO	ERRORC HECKFI ELD	
11	03	0B	C0	00	03	1E	LRC

**Figure 8 - Read Numeric Variable Query Message**

## RESPONSE

The addressed slave responds with its address and the function code, followed by the information field. The information field contains 2 bytes describing the quantity of data bytes to be returned. The contents of the registers requested (DATA) are 4 bytes each, with the binary content right justified within each pair of characters. The first byte includes the high order bits and the second, the low order bits.

Since the slave interface device is normally serviced at the end of the calculation scan, the data will reflect the variable content at the end of the scan. Some slaves will limit the quantity of variable content provided each scan; thus for large variable quantities, multiple transmissions will be made using variable content from sequential scans.

ADDR	FUNC	BYTE Count	DATA output index H.O. HOW	DATA output index L.O. LOW	DATA output index H.O. HOW	DATA output index L.O. LOW	DATA output index H.O. HOW	DATA output index L.O. LOW	ERROR CHECK FIELD	
11	03	06	02	2B	00	00	00	00	B9	LRC

**Figure 9 - Read Numeric Variable Response Message**

TITLE: <b>2500 Host-Slave Communications Software Requirements</b>	DRAWING NO. <b>ES-10212-01</b>	REVISION LEVEL <b>D</b>
	FILE NO.	SHEET <b>20</b> of <b>41</b>

## SET BOOLEAN VARIABLE (FUNCTION CODE 05)

### CAUTION

**Command (05) will override 2500 memory fixed parameter, and live flag.**

### QUERY

This message forces a single Boolean variable either ON or OFF or if used on a defined numerical variable switch from manual to live. Any Boolean variable that exists within the controller can be forced to either state (ON or OFF). Boolean variables are numbered from 1001 (Boolean 0001 = 1001, Boolean 0002 = 1002, etc.). The data value 65,280 (FF00 HEX) will set the Boolean ON and the value zero will turn it OFF; all other values are illegal and will not effect that flag.

The use of slave address 00 (Broadcast Mode) will force all attached slaves to modify the desired coil.

---

**NOTE:** Functions 5, 6, 15 and 16 are the only messages that will be recognized as valid for broadcast.

The below example is a request to slave number 17 (11 Hex) to turn ON Boolean 1073 (431 Hex).

---

ADDR	FUNC	DATA FLAG # HO	DATA FLAG # LO	DATA ON OFF IND	DATA	ERROR CHECK FIELD	
11	05	04	31	FF	00	B6	LRC

**Figure 10 - Force Single Boolean Query Message**

TITLE:  
**2500 Host-Slave  
Communications Software  
Requirements**

DRAWING NO.  
**ES-10212-01**

REVISION LEVEL  
**D**

FILE NO.

SHEET 21 of 41

The normal response to the Command Request is to retransmit the message as received after the Boolean state has been altered.

ADDR	FUNC	DATA FLAG # HO	DATA FLAG # LO	DATA ON OFF	DATA	ERROR CHECK FIELD	
11	05	04	31	FF	00	B6	LRC

**Figure 11 - Force Single Boolean Response Message**

### LOAD REGISTER (FUNCTION CODE 6)

#### QUERY

Any 16 bit short integer that has been defined by Config25 can have its contents changed by this message. The 16 bit short integers are numbered from 3001; (short integer number 1=3001, short integer number 2=3002, etc.) The example loads 33 into short integer 3001.

---

**NOTE:** Function codes 5, 6, 15, and 16 are the only messages that will be recognized as valid for broadcast.

---

ADDR	FUNC	DATA INDEX HO	DATA INDEX LO	DATA HO	DATA LO	ERROR CHECK FIELD	
01	06	0B	B9	00	21	14	LRC

**Figure 12 - Load Register Query Message**

TITLE:  
**2500 Host-Slave  
 Communications Software  
 Requirements**

DRAWING NO.  
**ES-10212-01**

REVISION LEVEL  
**D**

FILE NO.

SHEET 22 of 41

## RESPONSE

The addressed slave responds by retransmitting the query message.

ADDR	FUNC	DATA INDEX HO	DATA INDEX LO	DATA HO	DATA LO	ERROR CHECK FIELD	
01	06	0B	B9	00	21	14	LRC

**Figure 13 - Load Register Response Message**

## READ 2500 STATUS (FUNCTION CODE 07)

### QUERY

In many cases a short message requesting the status of certain events in a controller is desired. The read exception status code is designed to provide this functionality.

Function Code 7 allows the user to interrogate the status of eight Boolean variables within the 2500 Controller. These flags can be programmed to hold information of the 2500s control situation (e.g. machine ON/OFF, alarms status, etc.). Slave address 00 (Broadcast Mode) is not supported with the exception Status Request Message.

The eight Boolean statuses returned by this command depend on the slave PC type. The status of Flags 1-8 are returned.

STATUS	Boolean ASSIGNMENTS
<u>FLAG NO.</u>	<u>ASSIGNMENT</u>
1	Shutdown
2	Unknown Fail
3	Power Fail
4	Unacknowledged Alarms
5	Starting
6	Running
7	Warm Start
8	Cold Start

TITLE:  
**2500 Host-Slave  
Communications Software  
Requirements**

DRAWING NO.  
**ES-10212-01**

REVISION LEVEL  
**D**

FILE NO.

SHEET **23** of **41**

The below example displays a request to 2500 slave number 17 to respond with exception status.

ADDR	FUNC	ERROR CHECK FIELD	
11	7*	E8	LRC

**Figure 14 - Read 2500 Status Query Message**

\* No data fields required for this function.

### RESPONSE

The normal response contains the status of the eight flags packed into one data byte, one bit per flag.

SLAVE ADDR	FUNC	FLAG DATA	ERROR CHECK FIELD	
11	07	6D	7B	LRC

**Figure 15 - Read 2500 Status Response Message**

## SET MULTIPLE BOOLEAN VARIABLES (FUNCTION CODE 15)

### CAUTION

**Command (15) will override 2500 memory fixed Boolean value, and manual/live flag.**

### QUERY

This message forces each Boolean variable in a consecutive block of Boolean variables to a desired ON or OFF state. Any Boolean that exists within the controller can be forced to either state (ON or OFF). However, since the controller is actively scanning, unless the Booleans are disabled, the 2500 can also alter the state of the Boolean. Booleans are numbered from 1001 (Boolean 00001 = 1001, Boolean 00002 = 1002, etc.). The desired status of each Boolean is packed in the data field, one bit for each Boolean flag (1 = ON, 0 = OFF).

The use of slave address 00 (Broadcast Mode) will force all attached 2500 slaves to modify the desired flags.

---

**NOTE:** Functions 5, 6, 15 and 15 are the only messages that will be recognized as valid for broadcast. The following example forces 10 Booleans starting at address 1120 (3FC HEX). The two data fields, CD = 1100 1101 and 00 = 0000 0000, indicate that Booleans 1127, 1126, 1123, 1122 and 1120 are to be forced on.

---

ADDR	FUNC	H.O. ADDR	L.O. ADDR	QUANTITY	BYTE CNT	DATA FLAG STUS 120-127	DATA FLAG STUS 128-129	ERROR CHECK FIELD	
11	0F	04	60	00 0A	02	CD	00	A3	LRC

**Figure 16 - Force Multiple Booleans Query Message**

TITLE:  
**2500 Host-Slave  
 Communications Software  
 Requirements**

DRAWING NO.  
**ES-10212-01**

REVISION LEVEL  
**D**

FILE NO.

SHEET 25 of 41

## RESPONSE

The normal response will be an echo of the slave address, function code, starting address, and quantity of Booleans forced.

ADDR	FUNC	H.O. ADDR	L.O. ADDR	QUANTITY	ERROR CHECK FIELD	
11	0F	04	60	00 0A	72	LRC

**Figure 17 - Force Multiple Booleans Response Message**

The writing of Booleans via Modbus function 15 will be accomplished regardless of whether the addressed Booleans are disabled or not.

## LOAD NUMERIC VARIABLES (FUNCTION CODE 16)

### QUERY

### CAUTION

Function (16) will override 2500 controller memory.

Any numeric variable that has been defined on the numeric definition screens in ConFig25 can have its contents changed by this message. However, since the 2500 is actively scanning, it also can alter the content of any working variable location at any time. The values are provided in binary up to the maximum capacity of the 2500 controller (32 bit for the 2500) unused high order bits must be set to zero. When used with slave address zero (Broadcast Mode) all slave controllers will load the specified indexes with the contents specified.

---

**NOTE:** Function Codes 5, 6, 15 and 16 are the only messages that will be recognized as valid for broadcast.

---

TITLE:  
**2500 Host-Slave  
Communications Software  
Requirements**

DRAWING NO.  
**ES-10212-01**

REVISION LEVEL  
**D**

FILE NO.

SHEET **26** of **41**

ADDR	FUNC	H.O.	L.O.	QUANTITY	BYTE CNT	H.O. DATA	L.O. DATA	H.O. DATA	L.O. DATA	ERROR CHECK FIELD	
11	10	00	87	00 02	04	00	0A	01	01	45	LRC

**Figure 18 - Preset Multiple Variables Query Message**

## RESPONSE

The normal response to a function 16 (10 Hex) query is to echo the address, function code, starting address and number of registers to be loaded.

ADDR	FUNC	H.O. ADDR	L.O. ADDR	QUANTITY	ERROR CHECK FIELD	
11	10	00	87	00 02	56	LRC

**Figure 19 - Preset Multiple Registers Response Message**

---

**NOTE:** Depending on the ConFig25 numeric definition of 16 bit integer, 32 bit integer or 32 bit floating point will determine if multiple 16 bit variables are set or a single 32 bit value is assigned.

---

If the numeric variable that is referenced is a 32-bit integer or a single precision floating point number, the byte count must be a multiple of four (4). An example of a 32-bit integer assignment is as follows:

ADDR	FUNC	H.O. ADDR	L.O. ADDR	QUAN.	BYTE CNT	H.O. DATA HOW	L.O. DATA HOW	H.O. DATA LOW	L.O. DATA LOW	ERROR CHECK FIELD	
11	10	00	87	00 01	04	00	0A	01	02	46	LRC

**Figure 20 - Preset 32-Bit Integer Variable Query Message**

TITLE:  
**2500 Host-Slave  
Communications Software  
Requirements**

DRAWING NO.  
**ES-10212-01**

REVISION LEVEL  
**D**

FILE NO.

SHEET 27 of 41

32-Bit Single Precision Floating Point numbers are preset or read as groups of four (4) bytes with the following specific bit order.

SIGN (1 Bit)	EXPONENT (8 Bits)	MANTISSA (23 Bits)	
SEEEEEEE Byte 3	EMMMMMMM Byte 2	MMMMMMMM Byte 1	MMMMMMMM Byte 0

The "E" field is the two's exponent. It is a two's complimented value biased by 127 (decimal).

The "M" field is the 23-bit normalized mantissa. The most significant bit is always assumed to be 1, and so is not explicitly stored. This yields an effective precision of 24 bits.

The value of the floating point number described above is obtained by multiplying two raised to the power of the unbiased exponent, by the binary mantissa. The assumed bit of the binary mantissa (the most significant bit) has a value of 1.0, with the remaining bits providing a fractional value (i.e., the value of the mantissa is greater than or equal to 1.0 and less than 2.0). Note that the four data bytes of the floating point number are stored in lexicographic order. In the case of the LO/HI machines, this means the sign/exponent byte is stored at a higher memory address than the mantissa bytes.

TITLE: <b>2500 Host-Slave          Communications Software          Requirements</b>	DRAWING NO. <b>ES-10212-01</b>	REVISION LEVEL <b>D</b>
	FILE NO.	SHEET <u>28</u> of <u>41</u>

**2500 PROGRAMMING (Function Code 18)  
QUERY**

This message is used to read or write specific configuration information to or from a 2500 instrument and is specific to the 2500. The Function Code 18 is subdivided into six command functions which are summarized by the following:

<u>Command Codes</u>	<u>Description</u>
01	Read 2500 Configuration Table
02	Read 2500 RAM
03	Write 2500 Configuration Table
04	Write 2500 RAM
05	Halt and prepare for download
06	Download Complete, Start Running

**READ 2500 CONFIGURATION TABLE**

The following example issues a read Configuration Table Command to a 2500 at Address 17 (11 Hex).

ADDR	FUNC	CMD CODE	ERROR CHECK FIELD	
11	12	01	XX	LCR

**Figure 21 - Read 2500 Configuration Table Query Message**

The normal response will be an echo of the slave address, function code, command code, and the fixed length data which comprises the current configuration table resident in the 2500.

ADDR	FUNC	CMD CODE	DATA FROM CONFIG TABLE	ERROR CHECK FIELD	
11	12	01	XX...XX	XX	LRC

**Figure 22 - Read 2500 Configuration Table Response Message**

TITLE: <b>2500 Host-Slave Communications Software Requirements</b>	DRAWING NO. <b>ES-10212-01</b>	REVISION LEVEL <b>D</b>
	FILE NO.	SHEET <b>29</b> of <b>41</b>

## READ 2500 RAM AREA

This example issues a read 2500 RAM area beginning at 8040 Hex to a 2500 at address 17 (11 Hex).

ADDR	FUNCTION	CMD CODE	H.O. ADDRESS	L.O. ADDRESS	BYTE COUNT	ERROR CHECK FIELD	
11	12	02	80	40	FF	XX	LRC

**Figure 23 - Read 2500 RAM Area Query Message**

The normal response will be an echo of the slave address, function code, and command code followed by byte Count +1 of data requested.

ADDR	FUNC	CMD CODE	DATA	ERROR CHECK FIELD	
11	12	02	XX..XX	XX	LRC

**Figure 24 - Read 2500 RAM Area Response Message**

## WRITE 2500 CONFIGURATION TABLE

This example issues a write 2500 Configuration Table message to the 2500 instrument addressed at 17 (11 Hex).

ADDR	FUNC	CMD CODE	DATA	ERROR CHECK FIELD	
11	12	03	XX...XX	XX	LRC

**Figure 25 - Write 2500 Configuration Table Query Message**

TITLE:  
**2500 Host-Slave  
Communications Software  
Requirements**

DRAWING NO.  
**ES-10212-01**

REVISION LEVEL  
**D**

FILE NO.

SHEET **30** of **41**

The normal response will be an echo of the 2500 Address, Function Code and Command Code.

ADDR	FUNC	CMD CODE	ERROR CHECK FIELD	
11	12	03	XX	LRC

**Figure 26 - Write 2500 Configuration Table Response Message**

### WRITE 2500 RAM AREA

This example issues a write 2500 RAM Area beginning at 8040 Hex with length of 128 bytes.

ADDR	FUNC	CMD CODE	HOW HB Start	HOW LB Start	LOW HB Start	LOW LB Start	BYTE COUNT	DATA	ERROR CHECK	
11	12	04	00	00	80	40	7F	XX..XX	XX	LRC

**Figure 27 - Write 2500 RAM Area Query Message**

The normal response to this request will contain the Address, Function Code and Command Code

ADDR	FUNC	CMD CODE	ERROR CHECK	
11	12	04	XX	LRC

**Figure 28 - Write 2500 RAM Area Response Message**

## HALT AND PREPARE FOR DOWNLOAD

This command is used to prepare the 2500 for download of a new instrument configuration by forcing a cold start boot of the system. The example issues a Halt and Prepare for download of the following structure.

ADDR	FUNC	CMD CODE	ERROR CHECK FIELD	
11	12	05	XX	LRC

**Figure 29 - Halt 2500 and Prepare for Download Query**

The normal response will consist of the Address, Function Code and Command Code with the more significant byte = 1.

ADDR	FUNC	CMD CODE	ERROR CHECK FIELD	
11	12	05	XX	LRC

**Figure 30 - Halt Response Message**

## DOWNLOAD COMPLETE, RUN

This message is used to initiate running of the 2500 with the current 2500 configuration resident in memory.

ADDR	FUNC	CMD CODE	ERROR CHECK FIELD	
11	12	06	XX	LRC

**Figure 31 - Download Complete, Run Query Message**

TITLE:  
**2500 Host-Slave  
Communications Software  
Requirements**

DRAWING NO.  
**ES-10212-01**

REVISION LEVEL  
**D**

FILE NO.

SHEET **32** of **41**

The normal response will consist of an echo of the 2500 Address, Function Code and the Command Code which will change.

ADDR	FUNC	CMD CODE	ERROR CHECK FIELD	
11	12	01	XX	LRC

**Figure 32 - Download Complete, Run Response Message**

TITLE:  
**2500 Host-Slave  
Communications Software  
Requirements**

DRAWING NO.  
**ES-10212-01**

REVISION LEVEL  
**D**

FILE NO.

SHEET **33** of **41**

## 5.0 CONFIG25 MODBUS SUPPORT

Direct addressing of variables by the host that are defined in the 2500 computer is not possible using the Modbus protocol format. In order to support the standardized Modbus Communication Protocol on the 2500 Computer, a cross reference between an Index Number (Register) the variable name and its option must be provided.

Four additional screens would be added to the current ConFig25 program in order to allow user definition of the variables allowed to be accessed through the Modbus protocol. These ConFig25 screens are defined in Tables 4 through 8.

TITLE: <b>2500 Host-Slave Communications Software Requirements</b>	DRAWING NO. <b>ES-10212-01</b>	REVISION LEVEL <b>D</b>
	FILE NO.	SHEET <b>34</b> of <b>41</b>

Page

Created: 04/08/1985 09:37:23

Modifier: 04/18/1985 11:05:01

Report Communications Definition

**NOTE:** The protocol adds 500 to these index numbers

Index # Name

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18
- 19
- 20

**Table 4**

TITLE:  
**2500 Host-Slave  
 Communications Software  
 Requirements**

DRAWING NO.  
**ES-10212-01**

REVISION LEVEL  
**D**

FILE NO.

SHEET 35 of 41



Page

Created: 04/08/1985 09:37:23

Modifier: 04/18/1985 11:05:02

Short Integer Variable Communications Definition

**NOTE:** The protocol adds 3000 to these index numbers

Index #	Name	SubField
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		
19		
20		

**Table 6**

TITLE:  
**2500 Host-Slave  
Communications Software  
Requirements**

DRAWING NO.  
**ES-10212-01**

REVISION LEVEL  
**D**

FILE NO.

SHEET 37 of 41

Page

Created: 04/08/1985 09:37:23

Modifier: 04/18/1985 11:05:02

Long Integer Variable Communications Definition

**NOTE:** The protocol adds 5000 to these index numbers

Index #	Name	SubField
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		
19		
20		

**Table 7**

TITLE: <b>2500 Host-Slave                  Communications Software                  Requirements</b>	DRAWING NO. <b>ES-10212-01</b>	REVISION LEVEL <b>D</b>
	FILE NO.	SHEET <u>38</u> of <u>41</u>

Page

Created: 04/08/1985 09:37:23

Modifier: 04/18/1985 11:05:02

Floating Point Variable Communications Definition

**NOTE:** The protocol adds 7000 to these index numbers

Index #	Name	SubField
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		
19		
20		

**Table 8**  
**VALID POSSIBLE OPTION TABLE**

TITLE: <b>2500 Host-Slave                  Communications Software                  Requirements</b>	DRAWING NO. <b>ES-10212-01</b>	REVISION LEVEL <b>D</b>
	FILE NO.	SHEET <u>39</u> of <u>41</u>

<u>Boolean</u>	<u>16-Bit Integer</u>	<u>32-Bit Integer</u>	<u>Floating Point</u>
Manual/Live Flag Fixed Bit	Manual/Live Flag RAM Status (RO) Working/Selection No. R/(W)* Live Value Fixed Value Zero Scale Full Scale Scale Factor User Limit Lo Lo Limit Lo Limit Hi Limit Hi Hi Limit Rate Limit Time	Manual/Live Flag RAM Status (RO) Working/Selection No. R/(W)* Live Value Fixed Value Zero Scale Full Scale Scale Factor User Limit Lo Lo Limit Lo Limit Hi Limit Hi Hi Limit Rate Limit Time	Manual/Live Flag RAM Status (RO) Working/Selection No. R/(W)* Live Value Fixed Value Zero Scale Full Scale Scale Factor User Limit Lo Lo Limit Lo Limit Hi Limit Hi Hi Limit Rate Limit Time

---

**NOTE:** Actual valid options are dependent on variable definition in ConFig25.

---

All Options are read or write except those noted.

\*Write, only if variable is defined as selection or entry variable.

**Table 9**

TITLE: <b>2500 Host-Slave Communications Software Requirements</b>	DRAWING NO. <b>ES-10212-01</b>	REVISION LEVEL <b>D</b>
	FILE NO.	SHEET <b>40</b> of <b>41</b>

**ASSIGNED INDEX NUMBERS**

<u>Index No.</u>	<u>Description</u>	<u>Valid Function Code(s)</u>
		R/W
00	Full System Status Word	1/-
01	Time/Date	3/16
02	Unit Comm Address	3/16
03	Unit Comm Password	3/16
04	Unit User Password	3/16
05	Unit ID	3/16
06	Clear Unacknowledge List	-/1
07	Unassigned	
08	Unacknowledge Alarm List*	3/6,16
09	Current Alarms List*	3/-
10	Unassigned	
11		
:	Unassigned	
99		
100		
:	Application Dependent Controls	
500		

---

\* **NOTE:** This command returns the Index number that has previously been defined on the ConFig25 Tables. If a variable is in alarm which has not been defined by the modbus definition page, an invalid index number is returned with the appropriate error status for that variable.

---

**Table 10**

TITLE: <b>2500 Host-Slave          Communications Software          Requirements</b>	DRAWING NO. <b>ES-10212-01</b>	REVISION LEVEL <b>D</b>
	FILE NO.	SHEET <b>41</b> of <b>41</b>



**Model 2500**  
**Modbus Enhancements**  
**Engineering**  
**Software Specifications**  
  
**Revision A**

**The information contained in this specification  
is proprietary to Daniel Industries, Inc.  
and may not be released without the written  
permission of the Electronics Division.**

				TITLE: <b>Model 2500          Modbus Enhancements          Engineering          Software Specifications</b>	DRN <b>Hek</b>	DATE <b>10/93</b>
					CHKD <b>EM</b>	DATE <b>11/89</b>
					APPD <b>DJ</b>	DATE <b>11/89</b>
					REL <b>CR</b>	DATE <b>11/89</b>
<b>A</b>	<b>8938</b>	<b>11/89</b>	<b>EM</b>	PART NO.	DRAWING NO. <b>ES-10212-02</b>	SHEET <u>1</u> of <u>11</u>
REV.	ECO No.	DATE	BY			

# ENGINEERING SOFTWARE SPECIFICATION MODEL 2500 MODBUS ENHANCEMENTS

## 1.0 INTRODUCTION

This specification details enhancements to the Daniel version of Modbus communications. These enhancements impact Config25 and Base25 for the Model 2500, the PC Central data acquisition system, and SolarFlow Plus Modbus communications. Unless a specific exception is described, all items detailed in this specification apply equally to the Model 2500 and SolarFlow Plus (collectively referred to as RTUs). Note that there is no provision to support this processing using a Model 2500 as a Modbus master. Only slave mode of operation is supported.

The enhancements detailed are:

- Support for time-multiplexed data collection and alarm reporting
- Critical alarm archive definition
- Creation, retrieval, and clearing of an operator entry and alarm set/reset event log

## 1.2 RELEASE VERSIONS

For the Model 2500 these software enhancements are released in conjunction with the Model 2500 Baseline software L.8 and above. Only certain custom SolarFlow Plus applications contain these enhancements.

## 2.0 CRITICAL ALARM PROCESSING

The critical alarm processing enhancements consist of an extension to the existing Daniel implementation of the Modbus communications protocol. These enhancements are the detection and processing of the presently unimplemented function codes 65, 66, and 67. These function codes represent the messages Alarm Broadcast Request, Alarm Message, and Alarm Acknowledge, respectively.

<b>TITLE:</b> <b>Model 2500 Modbus Enhancements Engineering Software Specification</b>	<b>DRAWING NO.</b> <b>ES-10212-02</b>	<b>REVISION LEVEL</b> <b>A</b>
	<b>FILE NO.</b>	<b>SHEET <u>2</u> of <u>11</u></b>

These alarm messages are used to report critical alarms detected by an application running in a Model 2500 or SolarFlow Plus. As a Model 2500 application detects these critical alarms they are inserted into an archive area. SolarFlow Plus applications contain this processing as firmware.

Details of these new Modbus messages and the Model 2500 archive are described in the remainder of this section.

2.1 CRITICAL ALARM ARCHIVE

The Critical Alarm Archive is defined in the application developed with Config25. This archive is named CRITALRM so it can be located by Base25. The number of records in the archive is determined by the user. Each record contains the following integer entries:

Alarm Type  
TimeStamp  
DateStamp

where TimeStamp is in the form HHMMSS and DateStamp is in the form MMDDYY (YY = year - 1980).

The Alarm Type is a user defined code which represents a specific alarm condition, e.g. 77 means flow rate low limit exceeded. The other entries are derived from the standard Config25 time and date functions. The Alarm Type value of zero (0) is reserved and used to indicate no alarm data is present.

At startup the application must fill the CRITALRM archive with alarm type zeros. As alarms are detected they are inserted, using -1 as the record number, in the current record position. As the alarms are reported by Modbus to the central computer the alarm type is set to zero. The time stamp information is not changed.

TITLE: <b>Model 2500          Modbus Enhancements          Engineering          Software Specification</b>	DRAWING NO. <b>ES-10212-02</b>	REVISION LEVEL <b>A</b>
	FILE NO.	SHEET <b><u>3</u></b> of <b><u>11</u></b>

## 2.2 MODBUS ALARM FUNCTIONS

### 2.2.1 Overview

Three new Modbus functions are defined to implement critical alarm reporting. These functions and their function codes are:

65	Alarm Broadcast Request
66	Alarm Message
67	Alarm Acknowledge

The details of these messages are presented in the next sections. An explanation of the alarm processing is given following the message details.

### 2.2.2 Alarm Broadcast Request

The Alarm Broadcast Request is issued by the central computer to all RTUs on a Modbus network. Since it is a broadcast, the address field of the message always contains zero. The format of this message is:

<u>Byte</u>	<u>Field</u>
1	Address (0)
2	Function code (65)
3	Time periods allocated
4	Period in divisions of ten msecs
5	Null (0)
6	Null (0)

This message is padded with nulls to conform with the minimum message length of regular Modbus messages. This message requests any RTU having critical alarms to begin the alarm processing described below.

TITLE:

**Model 2500  
Modbus Enhancements  
Engineering  
Software Specification**

DRAWING NO.

**ES-10212-02**

REVISION LEVEL

**A**

FILE NO.

SHEET **4** of **11**

### 2.2.3 Alarm Message

The Alarm Message is the response generated by an RTU to an Alarm Broadcast Request. It contains one or more items of critical alarm information extracted from the CRITALRM archive area. The message format is:

<u>Byte</u>	<u>Field</u>
1	Address (RTU comm id)
2	Function code (66)
3	Number of alarms (HO byte: 16 bit int)
4	Number of alarms (LO byte: 16 bit int)
5	Alarm type (HO byte: 16 bit int)
6	Alarm type (LO byte: 16 bit int)
7-10	TimeStamp (HHMMSS: 32 bit FP)
11-14	DateStamp (MMDDYY, YY = year-1900: 32 bit FP)
:	:
:	:
n	Alarm type (HO byte: 16 bit int)
n+1	Alarm type (LO byte: 16 bit int)
n+7-10	TimeStamp (HHMMSS: 32 bit FP)
n+11-14	DateStamp (MMDDYY, YY = year-1900: 32 bit FP)

### 2.2.4 Alarm Acknowledge

The Alarm Acknowledge is sent by the central computer to a specific RTU to acknowledge alarms reported by the Alarm Message. The message format is:

<u>Byte</u>	<u>Field</u>
1	Address (RTU comm id)
2	Function code (67)
3	Number of alarms (HO byte: 16 bit int)
4	Number of alarms (LO byte: 16 bit int)
5	Alarm type (HO byte: 16 bit int)
6	Alarm type (LO byte: 16 bit int)
:	:
:	:
n	Alarm type (HO byte: 16 bit int)
n+1	Alarm type (LO byte: 16 bit int)

TITLE:  
**Model 2500  
 Modbus Enhancements  
 Engineering  
 Software Specification**

DRAWING NO.  
**ES-10212-02**

REVISION LEVEL  
**A**

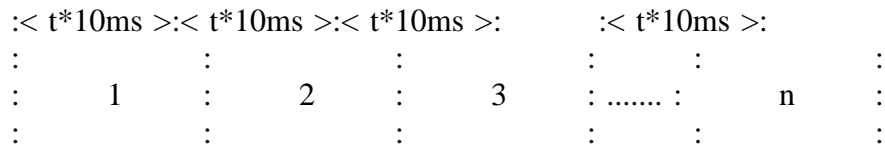
FILE NO.

SHEET **5** of **11**

The alarm type is the alarm reported in the Alarm Message. Upon receipt of this message the RTU removes the alarms from the CRITALRM archive area. This is done by setting the alarm type in the archive record to zero.

### 2.2.5 Alarm Processing

Periodically an RTU will receive an Alarm Broadcast Request message via Modbus. This message indicates the beginning of a time multiplexed interval. This interval allocates a time interval which is subdivided into a series of periods. The interval may have a single period or up to 255 periods. The Alarm Broadcast Request contains the number of periods allowed and the length in tens of milliseconds of the periods. Graphically, this appears as



where t is the time period and n is the number of time periods allocated.

When an Alarm Broadcast Request is received and the Model 2500 presently has alarms in the CRITALRM archive or a SolarFlow Plus has alarms in its buffer, it generates a random number, r, between 1 and n. At time period r, the Alarm Message is transmitted. In cases where n is 1 or 0 the Alarm Message is transmitted during period 1. After transmitting the Alarm Message the RTU returns to the normal mode of waiting for a Modbus message.

Upon receipt of an Alarm Acknowledge message, the alarms listed in the message are cleared from the buffers by setting the Alarm type to zero.

At times, multiple RTUs will select the same time period for sending their Alarm Message which will cause a collision. Since the central computer will not be able to process this jumbled message it will not respond with an Alarm Acknowledge. Therefore, the RTU must allow the central to request the alarms multiple times without the RTU receiving an acknowledge.

TITLE: <b>Model 2500          Modbus Enhancements          Engineering          Software Specification</b>	DRAWING NO. <b>ES-10212-02</b>	REVISION LEVEL <b>A</b>
	FILE NO.	SHEET <u>6</u> of <u>11</u>

### 3.0 OPERATOR AND ALARM EVENT LOG

The operator and alarm event log consists of an extension of the Daniel Modbus communications protocol to include retrieval of the event log and deletion of the retrieved events from the log. The event log consists of two records of data - an operator event record and an alarm event record. The formats of these records and the Modbus functions used are detailed in this section of the specification.

### 3.1 RECORD FORMATS

The two event log record formats are both the same size and have similar contents. The first word in a record is a bit map in which bit 9 indicates if the event record is an operator or an alarm event. The meanings of the other bits are specific to the operator or alarm event log records.

#### 3.1.1 Operator Event Record

The operator event record consists of the following:

<u>Bytes</u>	<u>Contents</u>
1-2	Operator change bit map (16 bit int)
3-4	Modbus register number of variable (16 bit int)
5-8	TimeStamp (HHMMSS: 32 bit FP)
9-12	DateStamp (MMDDYY: 32 bit FP)
13-16	Previous value of variable (32 bit FP)

TITLE:  
**Model 2500  
Modbus Enhancements  
Engineering  
Software Specification**

DRAWING NO.  
**ES-10212-02**

REVISION LEVEL  
**A**

FILE NO.

SHEET 7 of 11

The operator change bit map is:

<u>Bit</u>	<u>Value Changed</u>
0	Fixed value
1	Zero value
2	Full scale
3	Operator entry work value
4	Boolean fixed bit
5	Fixed/variable flag
6	Table entry change
7	System command change
8	
9	Operator change event identifier bit
10	Lolo limit
11	Lo limit
12	Hi limit
13	Hihi limit
14	Rate of change limit
15	

### 3.1.2 Alarm Event Record

The alarm event record consists of the following:

<u>Bytes</u>	<u>Contents</u>
1-2	Operator change bit map (16 bit int)
3-4	Modbus register number of variable (16 bit int)
5-8	TimeStamp (HHMMSS: 32 bit FP)
9-12	DateStamp (MMDDYY: 32 bit FP)
13-16	Current value of variable (32 bit FP)

The alarm bit map is:

<u>Bit</u>	<u>Value changed</u>
0-8	Unassigned
9	Operator change event identifier bit
10	Lolo alarm
11	Lo alarm
12	Hi alarm
13	Hihi alarm
14	Rate of change alarm
15	Set/reset alarm

TITLE:  
**Model 2500**  
**Modbus Enhancements**  
**Engineering**  
**Software Specification**

DRAWING NO.  
**ES-10212-02**

REVISION LEVEL  
**A**

FILE NO.

SHEET **8** of **11**

### 3.1.3 Event Log Request and Response

The Modbus request to read the event log uses the standard read function code 03 and the register number 32 (20H). In this request the number of registers is included to maintain format compatibility but is ignored by the receiving RTU. The request event log message appears as:

<u>Byte</u>	<u>Meaning</u>
1	Address (RTU comm id)
2	Function code (3)
3	Register (HO byte - 0: 16 bit int)
4	Register (LO byte - 32: 16 bit int)
5	Ignored
6	Ignored

In response to this request the RTU returns the current contents of the event log - up to the maximum size of a Modbus message (255 bytes). This message appears as:

<u>Byte</u>	<u>Meaning</u>
1	Address (RTU comm id)
2	Function code (3)
3	Number of bytes of data
4-19	Event record #1
20-35	Event record #2
:	:
:	:
:	:

TITLE:  
**Model 2500  
Modbus Enhancements  
Engineering  
Software Specification**

DRAWING NO.  
**ES-10212-02**

REVISION LEVEL  
**A**

FILE NO.

SHEET **9** of **11**

### 3.1.4 Clear Event Log Request and Response

The Modbus request to clear the event log uses the standard boolean write function code 05 and the register number 32 (20H). In this request the number of registers is always one (01). This request may be sent as a broadcast message, in which case no response is issued by the RTU. The clear event log request appears as:

<u>Byte</u>	<u>Meaning</u>
1	Address (RTU comm id)
2	Function code (5)
3	Register (HO byte - 0: 16 bit int)
4	Register (LO byte - 32: 16 bit int)
5	OFFH
6	0

In response to this request the RTU returns the message it received. This response message appears as:

<u>Byte</u>	<u>Meaning</u>
1	Address (RTU comm id)
2	Function code (5)
3	Register (HO byte - 0: 16 bit int)
4	Register (LO byte - 32: 16 bit int)
5	OFFH
6	0

In clearing the event log the RTU only clears the events transmitted to the Modbus master. Events which have occurred since the last event log request are not cleared.

TITLE:  
**Model 2500  
Modbus Enhancements  
Engineering  
Software Specification**

DRAWING NO.  
**ES-10212-02**

REVISION LEVEL  
**A**

FILE NO.

SHEET **10** of **11**

### 3.1.5 Example

Assume an event log containing five events. When the master requests the event log it receives these five events. A sixth event occurs. After this event, the master sends a clear request. The RTU clears the five events reported previously, leaving the sixth event in the log.

Again the master requests the event log and receives only the sixth event. A seventh and eighth event occur. Subsequently, the master requests the event log and receives the sixth, seventh and eighth events. Clearing the event log removes the sixth, seventh and eighth events.

TITLE: <b>Model 2500 Modbus Enhancements Engineering Software Specification</b>	DRAWING NO. <b>ES-10212-02</b>	REVISION LEVEL <b>A</b>
	FILE NO.	SHEET <b>11</b> of <b>11</b>



**Model 2500 Modbus Enhancement****Set Clock Command****Engineering Specification****Revision B**

**The information contained in this specification  
is proprietary to Daniel Industries, Inc.  
and may not be released without the  
written permission of the Electronics Division.**

				TITLE: <b>Model 2500 Modbus Enhancement Set Clock Command Engineering Specification</b>	DRN <b>Hek</b>	DATE <b>1/18/90</b>
					CHKD <b>MLT</b>	DATE <b>3/30/90</b>
					APPD <b>DJ</b>	DATE <b>1/18/90</b>
					REL <b>CRW</b>	DATE <b>3/30/90</b>
<b>B</b>	<b>6451</b>	<b>03/90</b>	<b>CW</b>	PART NO.	DRAWING NO. <b>ES-10212-03</b>	SHEET <b>1</b> of <b>3</b>
REV.	ECO No.	DATE	BY			

# MODBUS DATA ACQUISITION SPECIFICATION

## 1.0 PREFACE

Modbus is an industrial communications and distributed control system that integrates programmable controllers, computers, terminals and other monitoring, sensing and control devices.

Modbus and Modicon are trademarks of Gould Incorporated. The Modbus communications protocol is fully described in the reference guide entitled "Gould Modicon Modbus Protocol" publication PI-MBUS-300 and available from:

Gould Incorporated  
Programmable Control Division  
P.O. Box 3083  
Andover, Massachusetts 01810.

Modbus has been adapted for use by Daniel on the Model 2500 Series computers with permission of Gould Incorporated.

## 2.0 GENERAL

This specification defines the requirements for the Daniel Modbus Set Clock Command to be implemented and supported on the Model 2500 computer. This function is supported in the K.2 Base25 revision and higher releases.

TITLE: <b>Model 2500 Modbus Enhancement Set Clock Command Engineering Specification</b>	DRAWING NO. <b>ES-10212-03</b>	REVISION LEVEL <b>B</b>
	FILE NO.	SHEET <u>2</u> of <u>3</u>

Daniel Modbus set clock command:  
 (Uses format similar to Write Multiple Registers)

Modbus Function: 16  
 Register Nummer: 1  
 Number of Registers: 1  
 Byte Count: 20

ASCII Date/Time in form: 'DD-MMM-YYYY hh mm ss'  
 Where 'MMM' = 'JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL',  
 'AUG', 'SEP', 'OCT', 'NOV', OR 'DEC'

The Model 2500 will set the seconds to whatever is in the ' ss' field.

Example: Set clock to 12/25/89 11:30 AM

Message from Modbus Master to RTU:

Start Addr Func Reg Number Number of Reg's Byte Count  
 : xx 10 00 01 00 01 14

ASCII Date/Time

32 35 2D 44 45 43 2D 31 39 38 39 20 31 31 3A 33 30 3A 30 30  
 [ 2 5 - D E C - 1 9 8 9 1 1 : 3 0 : 0 0 ]

LRC End

xx CR LF

Response from RTU to Modbus Master:

Start Addr Func Reg Number Number of Reg's LRC End  
 : xx 10 00 01 00 01 xx CR LF

TITLE: <b>Model 2500 Modbus Enhancement          Set Clock Command          Engineering Specification</b>	DRAWING NO. <b>ES-10212-03</b>	REVISION LEVEL <b>B</b>
	FILE NO.	SHEET <u>3</u> of <u>3</u>



**Model 2251 Enhanced Specification  
Chromatograph Controller  
Modbus Communication Indices  
Revision B**

				<b>TITLE:</b> <b>Model 2251 Enhanced Specification Chromatograph Controller Modbus Communication Indices</b>	DRN <b>hk</b>	DATE <b>4-93</b>
					CHKD	DATE
					APPD	DATE
<b>B</b>	<b>7976</b>	<b>4-93</b>			REL	DATE
<b>A</b>	<b>9218</b>	<b>6-92</b>		Part No.	Drawing No. <b>ES-17128-005</b>	SHEET <u>1</u> of <u>11</u>
Rev.	ERO No.	DATE	BY			

**CHROMATOGRAPH CONTROLLER**  
**MODBUS COMMUNICATION INDICES**

The following tables contain descriptions of the various data available from the G.C. Controller via the MODBUS compatible communication interface as well as the assigned index numbers for each datum. Support of the standard MODBUS function codes in the Controller include Function Codes 3 and 16 as described in Daniel Engineering Specification ES-10212-01. In most instances it is anticipated that only Function Code 3 will be used to allow reading by a host computer of analysis results. Due to RAM limitations in the controller a single MODBUS query or response message is limited to sixty-four 16-bit integers or thirty-two 32-bit integer or floating point parameters.

	TITLE: <b>Model 2251 Enhanced Specification Chromatograph Controller Modbus Communication Indices</b>		
	Revision Level <b>B</b>	Drawing No. <b>ES-17128-005</b>	SHEET <u>2</u> of <u>11</u>

MODBUS COMMUNICATIONS INDICIES  
16-BIT INTEGERS

<u>INDEX</u>	<u>DESCRIPTION</u>	
3001	Component Table #1 - Component Index	# 1
3002	" " " - " "	# 2
3003	" " " - " "	# 3
3004	" " " - " "	# 4
3005	" " " - " "	# 5
3006	" " " - " "	# 6
3007	" " " - " "	# 7
3008	" " " - " "	# 8
3009	" " " - " "	# 9
3010	" " " - " "	#10
3011	" " " - " "	#11
3012	" " " - " "	#12
3013	" " " - " "	#13
3014	" " " - " "	#14
3015	" " " - " "	#15
3016	Component Table #1 - Component Index	#16
3017	Component Table #2 - Component Index	# 1
3018	" " " - " "	# 2
3019	" " " - " "	# 3
3020	" " " - " "	# 4
3021	" " " - " "	# 5
3022	" " " - " "	# 6
3023	" " " - " "	# 7
3024	" " " - " "	# 8
3025	" " " - " "	# 9
3026	" " " - " "	#10
3027	" " " - " "	#11
3028	" " " - " "	#12
3029	" " " - " "	#13
3030	" " " - " "	#14
3031	" " " - " "	#15
3032	" " " - " "	#16

TITLE:  
**Model 2251  
Enhanced Specification  
Chromatograph Controller  
Modbus Communication Indices**

Revision Level  
**B**

Drawing No.  
**ES-17128-005**

SHEET **3** of **11**

INDEX	DESCRIPTION
3033	Analysis Time (in 1/30ths of 1 second)
3034	Current Stream
3035	Mask of streams associated with Component Table #1 (Bit 2 <sup>n</sup> = 1 implies stream n included)
3036	Current Month (1-12)
3037	Current Day (1-31)
3038	Current Year (0-99)
3039	Current Hour (0-24)
3040	Current Minutes (0-59)
3041	Cycle Start Time - Month
3042	" " " - Day
3043	" " " - Year
3044	" " " - Hour
3045	" " " - Minutes
3046	See following page
3047	" " "
3048	" " "
3049	" " "
3050	" " "
3051	" " "
3052	" " "
3053	" " "
3054	" " "
3055	" " "
3056	" " "
3057	" " "
3058	New Data Flag (Set upon completion of calculations)
3059	Cal/Analysis Flag (Set = 1 If analysis data, 0 if calculation data)

32-BIT INTEGERS

5001	Cycle Time (in 1/30ths of 1 second)
5002	Calibration Cycle Time (in 1/30ths of 1 second)

TITLE:  
**Model 2251  
Enhanced Specification  
Chromatograph Controller  
Modbus Communication Indices**

Revision Level  
**B**

Drawing No.  
**ES-17128-005**

SHEET **4** of **11**

INDEX	BIT NUMBER																	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
3046	CHECKSUM FAILURE	ANALYZER FAILURE	D/A 3 HIGH	D/A 3 LOW	D/A 2 HIGH	D/A 2 LOW	D/A 1 HIGH	D/A 1 LOW	A/D CAL HIGH	A/D CAL LOW	A/D 2 HIGH	A/D 2 LOW	A/D 1 HIGH	A/D 1 LOW	A/D 0 HIGH	A/D 0 LOW		
3047	SPARE	SPARE	SPARE	SPARE	SPARE	SPARE	SPARE	SPARE	SPARE	SPARE	SPARE	SPARE	ADJUST PREAMP	PREAMP FAILURE	RF % DEV	POWER FAIL		
3048		#15 LOW	#14 LOW	#13 LOW	#12 LOW	#11 LOW	#10 LOW	#9 LOW	#8 LOW	#7 LOW	#6 LOW	#5 LOW	#4 LOW	#3 LOW	#2 LOW	#1 LOW	O.D.A. #1 LOW	} STREAM #1
3049		#15 HIGH	#14 HIGH	#13 HIGH	#12 HIGH	#11 HIGH	#10 HIGH	#9 HIGH	#8 HIGH	#7 HIGH	#6 HIGH	#5 HIGH	#4 HIGH	#3 HIGH	#2 HIGH	#1 HIGH	O.D.A. #1 HIGH	
3050		#15 LOW	#14 LOW	#13 LOW	#12 LOW	#11 LOW	#10 LOW	#9 LOW	#8 LOW	#7 LOW	#6 LOW	#5 LOW	#4 LOW	#3 LOW	#2 LOW	#1 LOW	O.D.A. #1 LOW	} STREAM #2
3051		#15 HIGH	#14 HIGH	#13 HIGH	#12 HIGH	#11 HIGH	#10 HIGH	#9 HIGH	#8 HIGH	#7 HIGH	#6 HIGH	#5 HIGH	#4 HIGH	#3 HIGH	#2 HIGH	#1 HIGH	O.D.A. #1 HIGH	
3052		#15 LOW	#14 LOW	#13 LOW	#12 LOW	#11 LOW	#10 LOW	#9 LOW	#8 LOW	#7 LOW	#6 LOW	#5 LOW	#4 LOW	#3 LOW	#2 LOW	#1 LOW	O.D.A. #1 LOW	} STREAM #3
3053		#15 HIGH	#14 HIGH	#13 HIGH	#12 HIGH	#11 HIGH	#10 HIGH	#9 HIGH	#8 HIGH	#7 HIGH	#6 HIGH	#5 HIGH	#4 HIGH	#3 HIGH	#2 HIGH	#1 HIGH	O.D.A. #1 HIGH	
3054		#15 LOW	#14 LOW	#13 LOW	#12 LOW	#11 LOW	#10 LOW	#9 LOW	#8 LOW	#7 LOW	#6 LOW	#5 LOW	#4 LOW	#3 LOW	#2 LOW	#1 LOW	O.D.A. #1 LOW	} STREAM #4
3055		#15 HIGH	#14 HIGH	#13 HIGH	#12 HIGH	#11 HIGH	#10 HIGH	#9 HIGH	#8 HIGH	#7 HIGH	#6 HIGH	#5 HIGH	#4 HIGH	#3 HIGH	#2 HIGH	#1 HIGH	O.D.A. #1 HIGH	
3056		#15 LOW	#14 LOW	#13 LOW	#12 LOW	#11 LOW	#10 LOW	#9 LOW	#8 LOW	#7 LOW	#6 LOW	#5 LOW	#4 LOW	#3 LOW	#2 LOW	#1 LOW	O.D.A. #1 LOW	} STREAM #5
3057		#15 HIGH	#14 HIGH	#13 HIGH	#12 HIGH	#11 HIGH	#10 HIGH	#9 HIGH	#8 HIGH	#7 HIGH	#6 HIGH	#5 HIGH	#4 HIGH	#3 HIGH	#2 HIGH	#1 HIGH	O.D.A. #1 HIGH	

TITLE:  
**Model 2251  
Enhanced Specification  
Chromatograph Controller  
Modbus Communication Indices**

Revision Level  
**B**

Drawing No.  
**ES-17128-005**

SHEET 5 of 11

FLOATING POINT (PROPOSED - IEEE KCS STANDARD FORMAT)

<u>INDEX</u>	<u>DESCRIPTION</u>
7001	Mole % - Component #1
7002	Mole % - Component #2
7003	" % - " #3
7004	" % - " #4
7005	" % - " #5
7006	" % - " #6
7007	" % - " #7
7008	" % - " #8
7009	" % - " #9
7010	" % - " #10
7011	" % - " #11
7012	" % - " #12
7013	" % - " #13
7014	" % - " #14
7015	" % - " #15
7016	Mole % - Component #16

(1)

(1) GPM is available on U.S. versions of the Models 2251 and 2255 only. ISO versions of the controller will contain weight % only.

TITLE: <b>Model 2251                  Enhanced Specification                  Chromatograph Controller                  Modbus Communication Indices</b>		
Revision Level <b>B</b>	Drawing No. <b>ES-17128-005</b>	SHEET <u>6</u> of <u>11</u>

<u>INDEX</u>	<u>DESCRIPTION</u>	
7017	GPM or Weight % - Component	#1
7018	" " " % - "	#2
7070	" " " % - "	#3
7020	" " " % - "	#4
7021	" " " % - "	#5
7022	" " " % - "	#6
7023	" " " % - "	#7
7024	" " " % - "	#8
7025	" " " % - "	#9
7026	" " " % - "	#10
7027	" " " % - "	#11
7028	" " " % - "	#12
7029	" " " % - "	#13
7030	" " " % - "	#14
7031	" " " % - "	#15
7032	GPM or Weight % - Component	#16

NOTE: If the CAL analysis flag is zero, the data in registers 7001 through 7039 should not be used. See Modbus Index 3059, herein.

TITLE:  
**Model 2251  
Enhanced Specification  
Chromatograph Controller  
Modbus Communication Indices**

Revision Level  
**B**

Drawing No.  
**ES-17128-005**

SHEET 7 of 11

U.S. Versions

ISO Version

<u>INDEX</u>	<u>DESCRIPTION</u>	<u>INDEX</u>	<u>DESCRIPTION</u>
7033	BTU - Dry	7033	HSUP - Dry
7034	BTU - Saturated	7034	HSUP - Saturated
7035	Specific Gravity	7035	Relative Density
7036	Compressibility	7036	Compressibility
7037	WOBBE Index	7037	WOBBE Index
7038	Total Unnormalized Mole %	7038	Total Unnormalized Mole %
7039	Total GPM	7039	N/A See note (1) on previous page
7040	Ratio #1		
7041	Ratio #2		
7042	Ratio #3		
7043	Ratio #4		
7044	Ratio #5		
7045	Rolling Average #1		
7046	" " #2		
7047	" " #3		
7048	" " #4		
7049	" " #5		
7050	" " #6		
7051	" " #7		
7052	" " #8		
7053	" " #9		
7054	Actual BTU		

TITLE:  
**Model 2251  
 Enhanced Specification  
 Chromatograph Controller  
 Modbus Communication Indices**

Revision Level  
**B**

Drawing No.  
**ES-17128-005**

SHEET **8** of **11**

U.S. Versions

ISO Version

<u>INDEX</u>	<u>DESCRIPTION</u>	<u>INDEX</u>	<u>DESCRIPTION</u>
7055	Not Used		
7056	" "		
7057	" "		
7058	" "		
7059	" "		
7060	" "		
7061	" "		
7062	" "		
7063	" "		
7064	" "		
7065	" "		
7066	" "		
7067	" "		
7068	" "		
7069	" "		

TITLE:  
**Model 2251  
 Enhanced Specification  
 Chromatograph Controller  
 Modbus Communication Indices**

Revision Level  
**B**

Drawing No.  
**ES-17128-005**

SHEET 9 of 11

<u>INDEX</u>	<u>DESCRIPTION</u>
7070	Not Used
7071	" "
7073	" "
7074	" "
7075	" "
7076	" "
7077	" "
7078	" "
7079	" "
7080	" "
7081	" "
7082	" "
7083	" "
7084	" "
7085	Analog Input #1 - Current Value in Engineering Units
7086	Analog Input #2 - Current Value in Engineering Units
7087	Actual BTU (last calibration)
7088	Dry BTU (last calibration)
7089	Sat BTU (last calibration)
7090	Wobbe Index (last calibration)
7091	Relative Density (last calibration)
7092	Compressibility (last calibration)
7093	Total GPM (last calibration)
7094	Total Unnormalized (last calibration)

TITLE:  
**Model 2251  
Enhanced Specification  
Chromatograph Controller  
Modbus Communication Indices**

Revision Level  
**B**

Drawing No.  
**ES-17128-005**

SHEET **10** of **11**

INDEX

DESCRIPTION

7095-7111	Response Factors (#1-16) Component Table #1
7111-7126	Response Factors (#1-16) Component Table #2
7127-7162	Extended Average (#1 - 36) Avg. Value - Current
7163-7198	Extended Average (#1 - 36) High Value - Current
7199-7234	Extended Average (#1 - 36) Low Value - Current
7235-7270	Extended Average (#1 - 36) Avg. Value - Previous
7271-7306	Extended Average (#1 - 36) High Value - Previous
7307-7342	Extended Average (#1 - 36) Low Value - Previous
7343-7378	Extended Average (#1 - 36) Avg. Value - Previous-1
7379-7414	Extended Average (#1 - 36) High Value - Previous-1
7415-7450	Extended Average (#1 - 36) Low Value - Previous-1
7451-7486	Extended Average (#1 - 36) Avg. Value - Previous-2
7487-7522	Extended Average (#1 - 36) High Value - Previous-2
7523-7558	Extended Average (#1 - 36) Low Value - Previous-2

TITLE:

**Model 2251  
Enhanced Specification  
Chromatograph Controller  
Modbus Communication Indices**

Revision Level  
**B**

Drawing No.  
**ES-17128-005**

SHEET **11** of **11**



## **WARRANTY CLAIM REQUIREMENTS**

To make a warranty claim, you, the Purchaser, must:

1. Provide Daniel with proof of the Date of Purchase and proof of the Date of Shipment of the product in question.
2. Return the product to Daniel within twelve (12) months of the date of original shipment of the product, or within eighteen (18) months of the date of original shipment of the product to destinations outside of the United States. The Purchaser must prepay any shipping charges. In addition, the Purchaser is responsible for insuring any product shipped for return, and assumes the risk of loss of the product during shipment.
3. To obtain Warranty service or to locate the nearest Daniel office, sales, or service center call (713) 467-6000, Fax (281) 897-2901, or contact:

Daniel Flow Products, Inc.  
Electronics  
P. O. Box 55435  
Houston, Texas 77255

When contacting Daniel for product service, the purchaser is asked to provide information as indicated on the following "Customer Problem Report".

Daniel Flow Products, Inc., Electronics offers both on call and contract maintenance service designed to afford single source responsibility for all its products.

Daniel Industries, Inc. reserves the right to make changes at any time to any product to improve its design and to insure the best available product.



**DANIEL INDUSTRIES, INC.  
CUSTOMER PROBLEM REPORT**

FOR FASTEST SERVICE, COMPLETE THIS FORM, AND RETURN IT ALONG WITH THE AFFECTED EQUIPMENT TO CUSTOMER SERVICE AT THE ADDRESS INDICATED BELOW.

COMPANY NAME: \_\_\_\_\_

TECHNICAL CONTACT: \_\_\_\_\_ PHONE: \_\_\_\_\_

REPAIR P. O. #: \_\_\_\_\_ IF WARRANTY, UNIT S/N: \_\_\_\_\_

INVOICE ADDRESS: \_\_\_\_\_

\_\_\_\_\_  
\_\_\_\_\_

SHIPPING ADDRESS: \_\_\_\_\_

\_\_\_\_\_  
\_\_\_\_\_

RETURN SHIPPING METHOD: \_\_\_\_\_

EQUIPMENT MODEL #: \_\_\_\_\_ S/N: \_\_\_\_\_ FAILURE DATE: \_\_\_\_\_

DESCRIPTION OF PROBLEM: \_\_\_\_\_

\_\_\_\_\_  
\_\_\_\_\_

WHAT WAS HAPPENING AT TIME OF FAILURE? \_\_\_\_\_

\_\_\_\_\_  
\_\_\_\_\_

ADDITIONAL COMMENTS: \_\_\_\_\_

\_\_\_\_\_  
\_\_\_\_\_

REPORT PREPARED BY: \_\_\_\_\_ TITLE: \_\_\_\_\_

IF YOU REQUIRE TECHNICAL ASSISTANCE, PLEASE FAX OR WRITE THE MAIN CUSTOMER SERVICE DEPARTMENT AT:

DANIEL FLOW PRODUCTS, INC.  
ATTN: CUSTOMER SERVICE  
19203 HEMPSTEAD HIGHWAY  
HOUSTON, TEXAS 77065

PHONE: (281) 897-2900  
FAX: (281) 897-2901





The sales and service offices of Daniel Industries, Inc. are located throughout the United States and in major countries overseas. Please contact the Daniel Industries, Inc., Electronics Division at P. O. Box 55435, Houston, Texas 77255, or phone (713) 467-6000 for the location of the sales or service office nearest you.

Electronics offers both on-call and contract maintenance service designed to provide single-source responsibility for all Electronics Products.

Daniel Industries, Inc. reserves the right to make changes to any of its products or services at any time without prior notification in order to improve that product or service and to supply the best product or service possible.

**DANIEL** *Electronics*  
Flow Products, Inc