WESTLOCK
CONTROLS CORPORATION

INTELLIS

Modbus Direct Network Monitor

System Installation and Operation Manual

280 Midland Ave., Saddle Brook, NJ 07663 USA
Phone: (201) 794-7650 □ Fax: (201)794-0913
www.westlockcontrols.com

Revision History

Revision 1.0	30 April, 2002	Initial Version

280 Midland Ave., Saddle Brook, NJ 07663 USA
Phone: (201) 794-7650 □ Fax: (201) 794-0913
www.westlockcontrols.com

Table Of Contents

Chapter 1

Modbus Protocol

- Introduction
- Modbus Message Framing
- Error Checking Methods

Introduction

The Westlock Controls Corp. Modbus Direct Network Monitor utilizes the **Modbus** protocol to serially communicate with a DCS or PLC. The **Modbus** protocol establishes a common format for the layout and contents of all message fields. Operating commands, sent via the **Modbus** network update the I/O of the Network Monitors on the network.

Modbus Interface Transaction

A **Modbus** compatible controller (designated as Master) communicates with the Westlock Modbus Direct Network Monitor (designated as one or more Slaves) by a Master-Slave technique, in which only one device (the Master) can initiate transactions. The Network Monitor (Slave) responds by supplying the requested data to the Master, and by taking the action requested.

A Master is capable of sending a message (called a 'query') to one specific Slave or to the entire group of Slaves (called a 'broadcast'). A Slave will acknowledge the query that contains its individual slave address, by sending back a message (called a 'response'). Responses are not returned if the Master sends the message to all the Slaves (broadcast).

The **Modbus** protocol establishes a format for the Master's query message, which contains a slave (or broadcast) address, a function code defining the requested action, all data to be sent, and an error-checking field. The Slave response message, as defined by the **Modbus** protocol, contains the slave address, a function code confirming the action taken, all data to be returned, and an error-checking field.

If a Slave detects an error in communications (incorrect stop/start bits, wrong parity or mismatch in error-checking field) it will not generate a response. If the Slave receives the communication correctly but is unable to perform the requested action, then an error message is sent back as a response to the **Modbus** Master.

At the physical layer the **Modbus** connection between the Master and Slaves utilizes RS-485.

The Query-Response Cycles

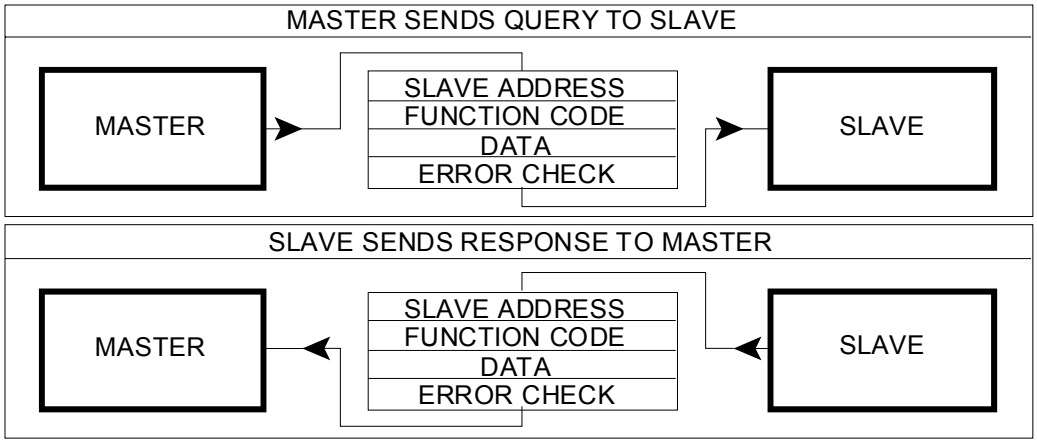

Figure 1 Master-Slave Query-Response Cycle

The Query: The function code in the query informs the addressed Slave of a required action. Data bytes contain all additional information needed in order to perform the function. For example, function code 05 will query the slave to force coil. The data field contains information informing the Slave of which coil to force. The error check field provides a method enabling the Slave to validate the integrity of the message contents.

The Response: When a Slave initiates a normal response, the function code in the response is an echo of the function code in the query. Data bytes contain all data collected by the Slave, such as register values or status. If an error should occur, the function code is modified to indicate that the response is an error response, and the data bytes will contain a code describing the error. An error check field allows the Master to confirm that the message contents are valid.

Serial Transmission Mode

Serial Transmission Mode
Modbus Controllers may be configured to communicate by either of two transmission modes: ASCII or RTU.

The Westlock's Modbus Direct Network Monitor communicates through a Modbus network utilizing the RTU mode only!

RTU (Remote Terminal Unit) mode
Each 8-bit byte in a message contains two 4-bit hexadecimal characters. Each message must be transmitted in a continuous stream.

RTU format for Modbus Direct Network Monitor is:

Coding System:
8-bit binary
Two hexadecimal digits (0-9, A-F) contained in each 8-bit character field.

Bits per Character:
1 start bit
8 data, least significant bit sent first
1 parity bit (even)
1 stop bit

Error Check Field:

Modbus Message Framing

Modbus Message Framing
A message is placed, by the transmitting device, in a message frame incorporating a known beginning and ending point. This allows the receiving devices to recognize the start of a message, read the address portion and determine which device is addressed (or all devices, in a broadcast message), and to acknowledge when the message has been completed.

RTU Framing
In the RTU mode, all messages begin with a silent interval of a least 3.5 character times. This is most easily implemented as a multiple of character times at the baud rate that is being used on the network (shown as T1-T2-T3-T4 in the figure below). The first field transmitted is the device address.

The RTU mode uses 8-bit binary values (two hexadecimal digits) for each character of the message. The Modbus Direct Network Monitor continually monitors the communications line even during 'silent' intervals. When the first field (the slave address field) is received, the Salve will decode it in order to ascertain whether it is the device being addressed or the message is for another Slave.

Following the last transmitted character, a similar interval of at least 3.5 character times marks the end of the message. A new message can begin only when the preceding interval has ended.

The entire message frame must be transmitted as a continuous stream. If a silent interval of more than 1.5 character times (1.7 mSec at 9600 Baud) occurs before completion of the frame, then the receiving device may flush the incomplete message and assumes that the next byte will be the address field of a new message.

Similarly if a new message begins earlier than 3.5 character time intervals (4 mSec at 9600 Baud) then the receiving device may consider it as a continuation of the previous message. This will result in an error, since the value in the final CRC field will not be valid for the combined messages.

A typical message frame is shown below.

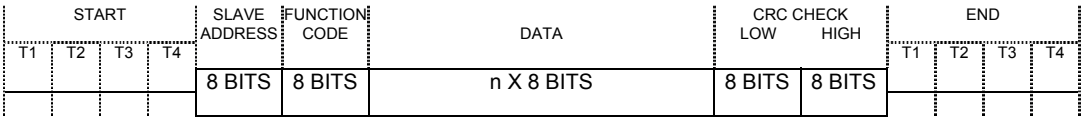

Figure 2 RTU Message Frame

ASCII Framing
Modbus ASCII framing is not implemented in the Modbus Direct Network Monitor.

Modbus Message Framing (cont.)

Handling the Address Field (RTU)
In the RTU mode the address field of a message frame contains eight bits. Modbus protocol defines valid Slave device addresses are in the range of 1 - 247 decimal. The individual Slave addresses are programmed into the Westlock Modbus Direct Network Monitor by a DIP switch settings on the Mpac module or via the Handheld Programmer (HHP).

A Master communicates with a Slave by placing the Slave address in the address field of the message. When the Slave sends its response, it places its own address, in the address field, of the response as identification to the Master as to which Slave is responding.

Address 0 is used for the broadcast address. All Slaves act upon messages received with this address, but none would generate a response message. When Modbus is utilized for higher level networks, broadcasts may not be allowed or may be replaced with other methods. For example, Modbus Plus uses a shared global database that can be updated with each token rotation.

Handling the Function Field (RTU)
In the RTU mode the function code field of a message frame contains eight bits. Valid codes are in the range of 1 - 255 decimal. Of these, some codes are applicable to all Modicon controllers while some codes apply only to certain models, and others are reserved for future use. Only certain function codes are implemented on the Westlock Modbus Direct Network Monitor.

When a message is sent from a Master to a Slave device, the function code informs the Slave of the kind of action to perform. Examples are: to read the On/Off state of one single or multiple coils or inputs; to read the diagnostic status of the Slave; to write to single/multiple coils or read input registers.

When the Slave responds to the Master, it uses the function code field to indicate that either a normal (error-free) response has occurred or that some type of error has occurred (called an exception response). With a normal response, the Slave simply echoes the original function code. With an exception response, the Slave returns a code that is equivalent to the original function code with its most-significant bit set to logic 1.

For example, a message from a Master to a Slave that requires the reading of the input registers, would have the following function code:
00000100 (Hexadecimal 04)
If the Slave device initiates the requested action without error, it will return the identical code in its response. If an exception occurs, it returns:
10000100 (Hexadecimal 84)

In addition to the modification of the function code for an exception response, the Slave places a unique code into the data field of the response message. This informs the Master of the type of error that just occurred, or the reason for the exception.

Modbus Message Framing (cont.)

The Master's device application program has the responsibility of handling exception responses. Typical processes are to post subsequent retries of the message, transmit diagnostic data to the slave, and notify operators through alarm functions. All possible Exception Response Codes are listed in Chapter 3.

Data Field (RTU)
In the RTU mode the data field of a message frame contains individual bytes of eight bits. Valid data byte values are in the range of 0 - 255 decimal. Two successive 8-bit bytes (High byte followed by a Low byte) may be used to represent numbers larger than 255.

The data field of messages sent from a Master to a Slave contains additional information, which the Slave utilizes to implement action defined by the function code. This includes items such as discrete and register addresses, the quantity of items to be handled and the count of actual data bytes in the field.

If an error does not occur, the data field of a response from a Slave to the Master contains the data requested. If an error should occur, the field contains an exception code that the Master can use to determine the next action to be taken.

Error Checking Field (RTU) In the RTU mode, the error checking field contains a 16-bit value sent as two 8-bit bytes. The error check value is the result of a Cyclical Redundancy Check calculation performed on the message's contents.

The CRC field is appended to the message as the last field in the message. When this is completed, the low-order byte of the field is appended first, followed by the high-order byte. The CRC high-order byte is the final byte to be sent in the message.

Additional information about error checking is contained later in this chapter. Detailed steps for generating the CRC field can also be found in this section.

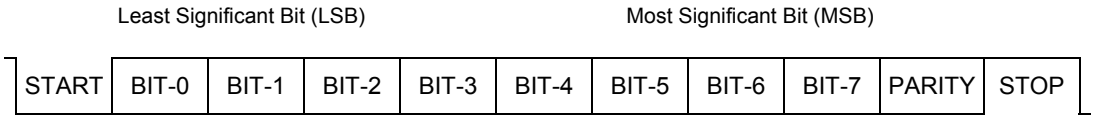

Figure 3 Bit Order - RTU with Parity Checking

Serial Transmission of Characters (RTU)
When messages are transmitted via the standard Modbus serial port, each character or 8-bit byte (shown as bit-0 to bit-7) is sent in the order shown above (beginning with the Start bit on the left and ending with the Stop bit on the right).

Error Checking Methods (RTU)

Two types of errors can occur in a communications system: transmission errors and programming or operation errors. The Modbus method of data transmission has specific methods for dealing with either type.

Transmission errors usually consist of an incorrectly received (changed) bit or bits in a message. It is uncommon to have a bit added or deleted from a message. The most prominent cause of a communication error is 'noise' in the communication channel. Sources of the 'noise' are electrical or magnetic interference from machinery, improper grounding, impulse noise (spikes), damage to the wires on the communication line, etc.

For these types of errors, Modbus (RTU) utilizes three types of error checking, frame check (on each character), parity check (on each character) and message content checking (CRC, applied to the entire message).

These character check and message check are applied to the message contents (by the transmitting device) before being sent. The receiving device then checks each character and the entire message to see if it was received correctly.

If a Slave detects a transmission error in a message it receives, then it ignores that message (will not act upon it and will not generate a response). The Master is programmed to expect a response within a certain amount of time, if the Master does not get a response within that time, it assumes the Slave will not respond (communication time-out). The Master's program determines how to handle the time-out, it may retransmit the message or assume the Slave is no longer active after a number of queries are not responded to.

Note: a message addressed to a non-existent Slave device will also cause a time-out.

Transmission Errors - Frame Checking
Each character sent must begin with a Start bit, contain a certain number of bits (8 data and one parity bits), and end with a Stop bit. Each bit must be a fixed length of time (104 μSec for 9600 baud). If any of these conditions are not met for any character sent, the entire message is ignored.

Transmission Errors - Parity Checking
Modbus protocol allows for Odd, Even, or No Parity checking. The Intellis Network Monitor uses even parity checking only!

In even parity checking, the number of logic 1 bits is counted in the data portion of each character (8 bits). The parity bit is then set to a logic 0 or logic 1 in order to achieve an even number of 1 bits. For example, in the number 213 (hex D5h or binary 11010101b) the quantity of 1 bits is five. For even parity check the parity bit will be set to 1, making the quantity of 1 bits an even number (six).

When the message is transmitted, the parity bit is calculated and sent with each character. The receiving device counts the quantity of 1 bits it receives and sets an error if they do not match the parity bit.

Note that parity checking can only detect an error if an odd number of bits are incorrectly received in each character transmitted. For example, if Even Parity checking is used and two 1 bits are dropped from a character containing four 1 bits, the result is still an even count of 1 bits.

Error Checking Methods (RTU) (cont.)

Transmission Errors - CRC Checking
In the Modbus RTU mode, messages include an error-checking field that is based on a Cyclical Redundancy check (CRC) method. The CRC field checks the content of the entire message. It is applied regardless of any parity checking method used for the individual characters of the message.

Action	16-Bit CRC	C
Load with 1's	1111 1111 1111 1111	X
Exclusive OR with 1st byte (04h)	0000 0100 1111 1111 1111 1011	x
1st Shift right	0111 1111 1111 1101	1
Exclusive OR Polynomial (A001h)	1010 0000 0000 0001 1101 1111 1111 1100	X
2nd Shift right	0110 1111 1111 1110	0
3rd Shift right	0011 0111 1111 1111	0
4th Shift right	0001 1011 1111 1111	1
Exclusive OR Polynomial (A001h)	1010 0000 0000 0001 1011 1011 1111 1110	X
5th Shift right	0101 1101 1111 1111	0
6th Shift right	0010 1110 1111 1111	1
Exclusive OR Polynomial (A001h)	1010 0000 0000 0001 1000 1110 1111 1110	X
7th Shift right	0100 0111 0111 1111	0
8th Shift right	0010 0011 1011 1111	1
Exclusive OR Polynomial (A001h)	1010 0000 0000 0001 1000 0011 1011 1110	X
Exclusive OR with 2nd byte (0Ch)	0000 1100 1000 0011 1011 0010	X
1st Shift right	0100 0001 1101 1001	0
2nd Shift right	0010 0000 1110 1100	1
Exclusive OR Polynomial (A001h)	1010 0000 0000 0001 1000 0000 1110 1101	X
3rd Shift right	0100 0000 0111 0110	1
Exclusive OR Polynomial (A001h)	1010 0000 0000 0001 1110 0000 0111 0111	X
4th Shift right	0111 0000 0011 1011	1
Exclusive OR Polynomial (A001h)	1010 0000 0000 0001 1101 0000 0011 1010	X
5th Shift right	0110 1000 0001 1101	0
6th Shift right	0011 0100 0000 1110	1
Exclusive OR Polynomial (A001h)	1010 0000 0000 0001 1001 0100 0000 1111	X
7th Shift right	0100 1010 0000 0111	1
Exclusive OR Polynomial (A001h)	1010 0000 0000 0001 1110 1010 0000 0110	X
8th Shift right	0111 0101 0000 0011	0
	7 5 h 0 3 h	

Message sent would be 04H - 0Ch - 03H - 75h

The CRC field is comprised of two bytes containing a 16-bit binary value. The CRC value is calculated by the transmitting device, which appends the CRC to the message. The receiving device calculates the CRC on the message it receives, and compares the calculated value to the value in the CRC field of the message. If the two values are not equal, a communication error results.

The CRC calculation begins by pre-loading the 16-bit CRC to all 1's then the following steps are taken.

1) Exclusive OR a data byte with the low byte of the 16-bit CRC.

2) Shift the 16-bit CRC one bit to the right, shifting the LSB out to the carry and shifting a 0 into the MSB.

3) If the carry is a 1 then exclusive OR the CRC polynomial (A001h) with the 16-bit CRC.

4) Repeat steps 2 and 3 until it has been shifted 8 times.

5) Go back to step 1 for the next data byte until all the bytes in the message are done.

6) Append the 16-bit CRC to the message sending the least significant byte followed by the most significant byte.

Figure 4 Example of CRC Calculation

Error Checking Methods (RTU) (cont.)

Operation Errors

Operation errors occur if the slave receives the query without communication errors but is unable to handle the request (for example, if the request is to read a non-existent coil or register). In this case, the Slave will return a response called an Exception Response, thereby informing the Master of the nature of the error. The Exception Response contains a field called any Exception Code that defines the type of error (see the lists of Exception Codes for each Function Code in the following chapter).

Chapter 2

Modbus Data and Control Functions

Implemented Function codes

- 01 Read Output Status (Read Output Status)
- 02 Read Input Status
- 03 Read Holding Registers
- 04 Read Input Registers
- 05 Force Single Output
- 08 Diagnostic Test
- 09 Program
- 15 (0Fh) Force Multiple Outputs

Non-Implemented Function codes

- 06 Preset Single Register
- 07 Read Exception Status
- 11 (0Bh) Fetch Communication Event Counter
- 12 (0Ch) Fetch Communication Event Log
- 16 (10h) Preset Multiple Registers
- 17 (11h) Report slave ID
- 20 (14h) Read General Reference
- 21 (15h) Write General Reference
- 22 (16h) Mask Write 4x Registers
- 23 (17h) Read/Write 4x Registers
- 24 (18h) Read FIFO Query

Function 01 - Read Output Status

Function Code 01
Reads the On/Off status of discrete outputs status in the slave. Broadcast is not supported for this function code.

On the Westlock Direct Intellis I/O module, there are two independent outputs. The output with address zero is connected to a solenoid that will operate an actuator to open a valve. This function reports the last state of the solenoid driver! If everything is operating properly, the state of the valve position is equal to the state of the solenoid driver output (valve open when solenoid driver is on, valve closed when solenoid driver is off). This function does not report the actual valve position (solenoid driver can be energized but the valve may be stuck in the closed position). To report the valve position, use Function 2: Read Input status.

Query
The query message specifies the starting output and quantity of outputs to be read. Since the Direct Intellis I/O module uses two distinct outputs, the Starting Output Address cannot more than one and the Number of Outputs should not more than two minus Starting Address.

Below is an example of a request to read Output Address zero (coil) status from Modbus address 10 (0Ah).

Query for Function 01		Example Message Bytes (Hex)	Example Message Bytes Decimal	
Field Name				**Explanation**
Slave Address		(0Ah)	10	Slave number 10
Modbus Function		(01h)	1	Read Output Status
Starting Output Address	(Hi byte)	(00h)	0	Starting at Output 0
	(Lo byte)	(00h)	0	
Number of Outputs	(Hi byte)	(00h)	0	Number of outputs to send is 1
	(Lo byte)	(01h)	1	
Error Check	(CRC Lo)	(FCh)	252	
	(CRC Hi)	(B1h)	177	

Function 01 - Read Output Status (cont.)

Response
The output status in the message is packed as one binary bit per output in the data field. Status is indicated as ; 1=On), 0=Off.

The Least two Significant Bit of the first data byte contains the output status and the higher order bits are all 0's (since the Direct Intellis I/O module uses two driver output).

Response for Function 01		**Example Message Bytes**			
Field Name		(Hex)	Decimal	Binary	**Explanation**
Slave Address		(0Ah)	10	0000 1010	Slave number 10
Modbus Function		(01h)	1	0000 0001	Read Output Status
Byte Count		(01h)	1	0000 0001	1 Byte of data to follow
Data with Output Status		(01h)	1	0000 0001	Solenoid is energized
Error Check	(CRC Lo)	(92h)	146	1001 0010	
	(CRC Hi)	(6Ch)	108	0110 1100	

Exception codes for Function 01:
02 - ILLEGAL DATA ADDRESS
The data address received in the query is not an allowable address for the slave. This error will be returned if Starting Address is more than one (since the Direct Intellis I/O module uses two outputs).

03 - ILLEGAL DATA VALUE
A value contained in the query data field is not an allowable value for the Slave. This error will be returned if the Number of Outputs is more than two minus Starting Address.

Function 02 - Read Input Status

Function Code 02
Reads the position (On/Off) status of discrete inputs in the slave. Broadcast is not supported for this function code.
These discrete inputs contain control valve position (limit switches), status, and other valve monitoring functions connected to the Slave inputs. Each Slave has 16 bits of status information. The table below shows the definitions of the 16 bits.

BIT	STATUS INFORMATION
0	"**Valve Closed**" Limit Switch (0=Contacts Open, 1=Contacts Closed)
1	"**Valve Open**" Limit Switch (0=Contacts Open, 1=Contacts Closed)

2	**User Fault Input # One -** (Example: Fugitive Emission Monitor)
	0 = User Fault Switch Contacts Open, 1 = User Fault Switch Contacts Closed

3	**User Fault Input # Two**
	0 = User Fault Switch Contacts Open, 1 = User Fault Switch Contacts Closed

4	**User Fault Input # Three**
	0 = User Fault Switch Contacts Open, 1 = User Fault Switch Contacts Closed

5	**User Fault Input # Four**
	0 = User Fault Switch Contacts Open, 1 = User Fault Switch Contacts Closed

6	**Valve Electronics Excessive Ambient Temperature Switch** 0 = Temperature Okay, 1 = Over /Under Temperature (Temperature is above 180F)
7	**Valve Slew Timeout Failure / Limit Switch Misalignment** 0 = Okay, 1 = Failure (valve did not reach commanded position within time limit)
8	Future
9	Future
10	Future
11	Future
12	Future
13	Future
14	Future
15	Future

Figure 5 Valve Status Bit Fields

Function 02 - Read Input Status (cont)

Query
The query message specifies the starting input bit and the quantity of input bits to be read. In the case of the Direct Intellis I/O module, there is only one valve (with status bits 0-15) so the range of the Starting Bit Address is 0 to 15, and the Starting Bit Address + the Number of Bits cannot exceed 16.

Below is an example of a request to read all 16 bits of input status from the valve at Modbus address 10 (0Ah).

Query for Function 02		**Example Message Bytes** (Hex)	Decimal	
Field Name				**Explanation**
Slave Address		(0Ah)	10	Slave number 10
Modbus Function		(02h)	2	Read Input Status
Starting Bit Address	(Hi byte)	(00h)	0	Starting at Bit 0
	(Lo byte)	(00h)	0	
Number of Bits	(Hi byte)	(00h)	0	Number of Bits to send is 16
	(Lo byte)	(10h)	16	
Error Check	(CRC Lo)	(78h)	120	
	(CRC Hi)	(BDh)	189	

Response
The input status in the response message is packed as bits in the data field. The Least Significant Bit of the first data byte contains the input bit addressed in the Query Starting Bit Address field. The other inputs follow toward the high order bits of the byte, and from 'low order to high order' in the subsequent data bytes.

If the returned input bit quantity is not a multiple of eight, the remaining bits in the final data byte will be padded with zeros (toward the high order end of the byte). The Byte Count field specifies the quantity of complete bytes of data.

Function 02 - Read Input Status (cont)

Response for Function 02		Example Message Bytes (Hex)	Decimal	Binary	
Field Name					**Explanation**
Slave Address		(0Ah)	10	0000 1010	Slave number 10
Modbus Function		(02h)	2	0000 0010	Read Input Status
Byte Count		(02h)	2	0000 0010	2 Bytes of data to follow
Data with Input Status	1st Data	(08h)	8	0000 1000	Valve closed
	2nd Data	(02h)	2	0000 0010	User fault input #3
Error Check	(CRC Lo)	(9Ah)	154	1001 1010	
	(CRC Hi)	(78h)	120	0111 1000	

The bit status is as shown below:

Bit 0=0 "Valve Closed" Limit Sw contacts open
Bit 1=0 "Valve Open" Limit Sw contacts open
Bit 2=0 User fault input #1 Sw contacts open
Bit 3=1 User fault input #2 Sw contacts closed
Bit 4=0 User fault input #3 Sw contacts open
Bit 5=1 User fault input #4 Sw contacts closed
Bit 6=0 Temperature is OK
Bit 7=0 No Valve Slew Timeout

Exception codes for Function 02:

02 - ILLEGAL DATA ADDRESS
The data address received in the query is not an allowable address for the slave. This error will be returned if the starting bit address is not between 0 and 15 (since the Direct Intellis I/O module has only one valve, with 16 bit status, per Slave address).

03 - ILLEGAL DATA VALUE
A value contained in the query data field is not an allowable value for the Slave. This error will be returned if the starting bit address plus the number of bits is greater than 16 (since the Direct Intellis I/O module uses 16 bits of status).

Function 03 - Read Holding Registers

Function Code 04
Reads the binary content of input registers in the Slave. Broadcast is not supported for this function code. Each valve has two holding registers.
The lower address is the Low Calibration Value; higher is the High Calibration Value

Each value is represented by an unsigned 16 bit count (sent as two 8-bit bytes). The maximum value 1023 and is limited by A/D converter.

Low Calibration Value is the value measured by A/D converter when valve was in position close and calibration button was pressed or command remote calibration was released (function 9 code 0). This value can also be programmed in explicit way (function 9 code 2).

High Calibration Value is the value measured by A/D converter when valve was in position "open" and calibration button was pressed or command remote calibration was released (function 9 code 0). This value can also be programmed in explicit way (function 9 code 3).

Query
The query message specifies the starting register and the quantity of registers to be read. In the Direct Intellis I/O module, there is only one valve (with two 16 bit registers) so the range of the Starting Register Address is 0 to 1, and the Starting Bit Address + the Number of Registers cannot exceed 2.

Below is an example of a request to read both 16 bit registers from the valve at Modbus address 10 (0Ah).

Query for Function 04

Field Name		**Example Message Bytes** (Hex)	Decimal	**Explanation**
Slave Address		(0Ah)	10	Slave number 10
Modbus Function		(04h)	4	Read Input Registers
Starting Register Address	(Hi byte)	(00h)	0	Starting at Register 0
	(Lo byte)	(00h)	0	
Number of Registers	(Hi byte)	(00h)	0	Number of Registers to send is 2
	(Lo byte)	(02h)	2	
Error Check	(CRC Lo)	(70h)	112	
	(CRC Hi)	(B0h)	176	

Function 03 - Read Holding Registers (Cont.)

Response
The register data in the response message is packed as 2 bytes per register in the data field. The first data byte contains the high order byte of the register addressed in the Query Starting Register Address field. The next byte contains the low order byte.

Response for Function 04		**Example Message Bytes**			
Field Name		(Hex)	Decimal	Binary	**Explanation**
Slave Address		(0Ah)	10	0000 1010	Slave number 10
Modbus Function		(04h)	4	0000 0100	Read Input Registers
Byte Count		(04h)	4	0000 0100	4 Bytes of data to follow
Data of Register value	1st Data	(00h)	0	0000 0000	Breakaway time
	2nd Data	(AFh)	175	1010 1111	(00AFh)=175=1.75 seconds
Data of Register value	3rd Data	(02h)	2	0000 0010	Cycle time (020Dh)=525=5.25
	4th Data	(0Dh)	13	0000 1101	seconds
Error Check	(CRC Lo)	(B1h)	177	1011 0001	
	(CRC Hi)	(C0h)	192	1100 0000	

Exception codes for Function 04:

02 - ILLEGAL DATA ADDRESS
The data address received in the query is not an allowable address for the slave. This error will be returned if the starting register address is not 0 or 1 (since the Direct Intellis I/O module has only one valve, with two 16 bit registers, per Slave address).

03 - ILLEGAL DATA VALUE
A value contained in the query data field is not an allowable value for the Slave. This error will be returned if the starting register address plus the number of registers to send is greater than 2 (since the Direct Intellis I/O module uses two 16 bits registers).

Function 04 - Read Input Registers

Function Code 04
Reads the binary content of input registers in the Slave. Broadcast is not supported for this function code.

This allows for the reading of control valve breakaway time, cycle time and valve position. Each register is represented by an unsigned 16 bit count (sent as two 8-bit bytes). The units of the first two registers are 0.01 seconds, e.g. a count of 352 decimal represents 3.52 seconds. Third register is a raw data from A/D converter.

Each valve has three registers. The lowest address is the Break Away Time Register, middle the Cycle Time Register and the higher is Valve position (Raw A/D value).

Break Away Time
The time between the slaves reception of a change in coil (Output Address zero) status command and the deactivation of the present position limit switch.

Cycle Time
The time between the slaves reception of a change in coil status command and the activation of the commanded position limit switch.

Valve Position
Determined by reading position of the potentiometer attached to the valve shaft. The value is measured using ratio measurement method. Moving potentiometer wiper from one extreme to another will result reading from 0 to 1023. (Maximum value of the 10 bit A/D)

Note: **Travel Time** is the time from the deactivation of the present position limit switch to the activation of the commanded position limit switch. Travel time is the cycle time minus the breakaway time.

Query
The query message specifies the starting register and the quantity of registers to be read. In the Direct Intellis I/O module, there is only one valve (with three 16 bit registers) so the range of the Starting Register Address is 0 to 2, and the Starting Address + the Number of Registers cannot exceed 3.

Below is an example of a request to read first two 16 bit registers from the valve at Modbus address 10 (0Ah).

Query for Function 04		**Example Message Bytes**		
Field Name		(Hex)	Decimal	**Explanation**
Slave Address		(0Ah)	10	Slave number 10
Modbus Function		(04h)	4	Read Input Registers
Starting Register Address	(Hi byte)	(00h)	0	Starting at Register 0
	(Lo byte)	(00h)	0	
Number of Registers	(Hi byte)	(00h)	0	Number of Registers to send is 2
	(Lo byte)	(02h)	2	
Error Check	(CRC Lo)	(70h)	112	
	(CRC Hi)	(B0h)	176	

Function 04 - Read Input Registers (cont)

Response
The register data in the response message is packed as 2 bytes per register in the data field. The first data byte contains the high order byte of the register addressed in the Query Starting Register Address field. The next byte contains the low order byte.

Response for Function 04		**Example Message Bytes**			
Field Name		(Hex)	Decimal	Binary	**Explanation**
Slave Address		(0Ah)	10	0000 1010	Slave number 10
Modbus Function		(04h)	4	0000 0100	Read Input Registers
Byte Count		(04h)	4	0000 0100	4 Bytes of data to follow
Data of Register value	1st Data	(00h)	0	0000 0000	Breakaway time
	2nd Data	(AFh)	175	1010 1111	(00AFh)=175=1.75 seconds
Data of Register value	3rd Data	(02h)	2	0000 0010	Cycle time (020Dh)=525=5.25 seconds
	4th Data	(0Dh)	13	0000 1101	
Error Check	(CRC Lo)	(B1h)	177	1011 0001	
	(CRC Hi)	(C0h)	192	1100 0000	

Exception codes for Function 04:
02 - ILLEGAL DATA ADDRESS
The data address received in the query is not an allowable address for the slave. This error will be returned if the starting register address is not 0, 1 or 2 (since the Direct Intellis I/O module has only one valve, with three 16 bit registers, per Slave address).

03 - ILLEGAL DATA VALUE
A value contained in the query data field is not an allowable value for the Slave. This error will be returned if the starting register address plus the number of registers to send is greater than 3 (since the Direct Intellis I/O module uses three 16 bits registers).

Function 05 - Force Single Output

Function Code 05
Forces a single output to either On or Off. When broadcast, the function forces the same output reference in all the attached Slaves.

On the Westlock Direct Intellis I/O module, the output address zero is connected to a solenoid that will operate an actuator to open a valve. This function allows a single valve solenoid to be individually energized or de-energized.

Query
The query message specifies the output reference (Output Address) to be forced. There are two independent outputs per Slave. The address zero is dedicated to solenoid. The Output Address can be zero or one. The requested On/Off (energized/de-energized) state is specified as a constant in the query data field. A value of FF00h requests the output to be forced On (energized). A value of 0000h requests the output to be forced Off (de-energized). All other data values are illegal and will not affect the output.

Below is an example of a request to force the output address zero (coil) on Modbus address 10 (0Ah) to the On state.

Function 05 - Force Single Output (cont)

Query for Function 05		Example Message Bytes (Hex)	Decimal	Explanation
Slave Address		(0Ah)	10	Slave number 10
Modbus Function		(05h)	5	Force Single Output
Output Address	(Hi byte)	(00h)	0	Output Address 0
	(Lo byte)	(00h)	0	
Force Data	(Hi byte)	(FFh)	255	Force Output On
	(Lo byte)	(00h)	0	
Error Check	(CRC Lo)	(8Dh)	141	
	(CRC Hi)	(41h)	65	

Response
The normal response is an echo of the query message returned after the output state has been forced.

Response for Function 05		Example Message Bytes (Hex)	Decimal	Binary	Explanation
Slave Address		(0Ah)	10	0000 1010	Slave number 10
Modbus Function		(05h)	5	0000 0101	Force Single Output
Outputl Address	(Hi byte)	(00h)	0	0000 0000	Output Address 0
	(Lo byte)	(00h)	0	0000 0000	
Force Data	(Hi byte)	(FFh)	255	1111 1111	Output Forced On
	(Lo byte)	(00h)	0	0000 0000	
Error Check	(CRC Lo)	(8Dh)	141	1000 1101	
	(CRC Hi)	(41h)	65	0100 0001	

Exception codes for Function 02:
02 ILLEGAL DATA ADDRESS
The data address received in the query is not an allowable address for the slave. This error will be returned if the Output Address is more than one since the Direct Intellis I/O module has only two outputs.

03 ILLEGAL DATA VALUE
A value contained in the query data field is not an allowable value for the Slave. This error will be returned if the data is not 0000h (close valve) or FF00h (open valve).

Function 08 - Diagnostic Test

Function Code 08
Provides for testing the communication system between the Master and Slave and for checking various error conditions within the Slave. Broadcast is not supported for this function code.

This function uses a two-byte diagnostic code field in the query to define the type of test to be performed. The slave echoes back both the function code and the diagnostic code in a normal response.
In general, issuing a diagnostic function to a Slave does not affect the running of the user program in the Slave. Westlock Direct Intellis I/O Module supports the following diagnostic code:

1. 00h (Return Query Data)
2. 02h (Return Cycle Count Register)
3. 0Ah (Reset Cycle Count Register)

Diagnostic code 00h (Return Query Data)
The data sent in the query data field is returned (looped back) in the response. The entire response message is identical to the query message.

Query
Below is an example of a request, to the Slave at Modbus address 10 (0Ah), to return Query data. This uses Diagnostic Code 0000h with the data to be returned as A537h.

Query for Function 08

Field Name		Example Message Bytes (Hex)	Example Message Bytes Decimal	Explanation
Slave Address		(0Ah)	10	Slave number 10
Modbus Function		(08h)	8	Diagnostic Test
Diagnostic Code	(Hi byte)	(00h)	0	Return Query Data
	(Lo byte)	(00h)	0	
Data Field	(Hi byte)	(A5h)	165	Data to Return
	(Lo byte)	(37h)	55	(can be any data)
Error Check	(CRC Lo)	(DBh)	219	
	(CRC Hi)	(F6h)	246	

Function 08 - Diagnostic Test (cont)

Response
The normal response to the Return Query Data diagnostic code is to loop back the same data. The function code and diagnostic code are also echoed back making the response message the same as the query message.

Response for Function 08		**Example Message Bytes**			
Field Name		(Hex)	Decimal	Binary	**Explanation**
Slave Address		(0Ah)	10	0000 1010	Slave number 10
Modbus Function		(08h)	8	0000 1000	Diagnostic Test
Diagnostic Code	(Hi byte)	(00h)	0	0000 0000	Return Query Data
	(Lo byte)	(00h)	0	0000 0000	
Data Field	(Hi byte)	(A5h)	165	1010 0101	Data from Query
	(Lo byte)	(37h)	55	0011 0111	
Error Check	(CRC Lo)	(DBh)	219	1101 1011	
	(CRC Hi)	(F6h)	246	1111 0110	

Diagnostic code 02h (Return Cycle Count Register)
The data sent in the query data field is replaced with Cycle Count Register in the response. The response message is identical to the query message except the data field.

Query
Below is an example of a request, to the Slave at Modbus address 10 (0Ah), to return Cycle Count Register. This uses Diagnostic Code 0002h with the query data as 0000h.

Query for Function 08		**Example Message Bytes**		
Field Name		(Hex)	Decimal	**Explanation**
Slave Address		(0Ah)	10	Slave number 10
Modbus Function		(08h)	8	Diagnostic Test
Diagnostic Code	(Hi byte)	(00h)	0	Return Cycle Count Register
	(Lo byte)	(02h)	2	
Data Field	(Hi byte)	(00h)	00	Data will be ignored
	(Lo byte)	(00h)	00	(can be any data)
Error Check	(CRC Lo)	(40h)	64	
	(CRC Hi)	(B0h)	176	

Function 08 - Diagnostic Test (cont)

Response
The normal response is to return Cycle Count Register in the data field. A cycle count is defined as the valve of the slave has made a complete full open and then close operation. The function code and diagnostic code are also echoed back.

Response for Function 08		**Example Message Bytes**			
Field Name		(Hex)	Decimal	Binary	**Explanation**
Slave Address		(0Ah)	10	0000 1010	Slave number 10
Modbus Function		(08h)	8	0000 1000	Diagnostic Test
Diagnostic Code	(Hi byte)	(00h)	0	0000 0000	Return Cycle Count Register
	(Lo byte)	(02h)	2	0000 0010	
Data Field	(Hi byte)	(11h)	17	0001 0001	Cycle Count Register=4372
	(Lo byte)	(14h)	20	0001 0100	
Error Check	(CRC Lo)	(4Ch)	76	0100 1100	
	(CRC Hi)	(EFh)	239	1110 1111	

Diagnostic code 0Ah
(Reset Cycle Count Register)
The data sent in the query data field will be ignored and the entire response message is identical to the query message. The Cycle Count Register will be reset to zero.

Query
Below is an example of a request, to the Slave at Modbus address 10 (0Ah), to reset Cycle Count Register. This uses Diagnostic Code 000Ah with the query data as 0000h.

Query for Function 08		**Example Message Bytes**		
Field Name		(Hex)	Decimal	**Explanation**
Slave Address		(0Ah)	10	Slave number 10
Modbus Function		(08h)	8	Diagnostic Test
Diagnostic Code	(Hi byte)	(00h)	0	Return Query Data
	(Lo byte)	(0Ah)	10	
Data Field	(Hi byte)	(00h)	0	Data to Return
	(Lo byte)	(00h)	0	(can be any data)
Error Check	(CRC Lo)	(C1h)	193	
	(CRC Hi)	(72h)	114	

Function 08 - Diagnostic Test (cont)

Response
The normal response to the Return Query Data diagnostic code is to loop back the same data. The function code and diagnostic code are also echoed back making the response message the same as the query message.

Response for Function 08		**Example Message Bytes**			
Field Name		(Hex)	Decimal	Binary	**Explanation**
Slave Address		(0Ah)	10	0000 1010	Slave number 10
Modbus Function		(08h)	8	0000 1000	Diagnostic Test
Diagnostic Code	(Hi byte)	(00h)	0	0000 0000	Return Query Data
	(Lo byte)	(0Ah)	10	0000 1010	
Data Field	(Hi byte)	(00h)	0	0000 0000	Data from Query
	(Lo byte)	(00h)	0	0000 0000	
Error Check	(CRC Lo)	(C1h)	193	1100 0001	
	(CRC Hi)	(72h)	114	0111 0010	

Exception codes for Function 08
03 ILLEGAL DATA VALUE
A value contained in the Diagnostic Code is not an allowable value for the Slave. This error will be returned if the Diagnostic Code is not implemented in Westlock Intellis I/O Module.

Function 09 - Program

Function Code 09
Provides for programming and calibration of the slave. Broadcast is not supported for this function code.

This function uses a two-byte sub-function code field in the query to define the type of task to be performed. The next two bytes are data bytes or are irrelevant In a normal response the slave echoes back the function code, diagnostic code and data bytes.

Intellis I/O Module supports the following diagnostic code:
Supports Codes:

1. 00h (Low/High Calibration -remotely).
2. 02h (Low Calibration – explicit value)
3. 03h (High Calibration – explicit value)
4. 04h (Program Slew Timeout)
5. 05h (Program Slave Address)
6. 06h (Communication Fail time/mode)
7. 07h (Burn info into EEPROM)

Program code 00h (Low/Hi Calibration)
The data sent in the query data field is returned (looped back) in the response. The entire response message is identical to the query message.

Query
Below is an example of a request, to the Slave at Modbus address 10 (0Ah), to perform Low or Hi calibration (depends on position of the valve limit switches).

Query for Function 09		**Example Message Bytes** (Hex)	Decimal	**Explanation**
Slave Address		(0Ah)	10	Slave number 10
Modbus Function		(09h)	8	Program
Program Code	(Hi byte)	(00h)	0	Low/high calibration
	(Lo byte)	(00h)	0	
Data Field	(Hi byte)	(00h)	0	Data to Return
	(Lo byte)	(00h)	0	(can be any data)
Error Check	(CRC Lo)	(xxh)	xx	
	(CRC Hi)	(xxh)	xx	

Function 09 - Program (cont)

Response
The normal response to the Return Query Data diagnostic code is to loop back the same data. The function code and diagnostic code are also echoed back making the response message the same as the query message.

Response for Function 09		Example Message Bytes (Hex)	Decimal	Binary	Explanation
Slave Address		(0Ah)	10	0000 1010	Slave number 10
Modbus Function -program		(09h)	9	0000 1001	Program
Sub Code	(Hi byte)	(00h)	0	0000 0000	Returned code
	(Lo byte)	(00h)	0	0000 0000	
Data Field	(Hi byte)	(00h)	0	0000 0000	Data from Query
	(Lo byte)	(00h)	0	0000 0000	
Error Check	(CRC Lo)	(xxh)	xx	xxxxxxxx	
	(CRC Hi)	(xxh)	xx	xxxxxxxx	

Program code 02h (Low Calibration)
The data sent in the query data field is returned (looped back) in the response. The entire response message is identical to the query message.

Query
Below is an example of a request, to the Slave at Modbus address 10 (0Ah), to perform Low Calibration using values placed in the data bytes.

Query for Function 09		Example Message Bytes (Hex)	Decimal	Explanation
Slave Address		(0Ah)	10	Slave number 10
Modbus Function		(09h)	8	Program
Program Code	(Hi byte)	(00h)	0	Low calibration
	(Lo byte)	(02h)	0	
Data Field	(Hi byte)	(00h)	0	Calibration data 0 to 1023
	(Lo byte)	(23h)	35	
Error Check	(CRC Lo)	(xxh)	xx	
	(CRC Hi)	(xxh)	xx	

Function 09 - Program (cont)

Response
The normal response to the Return Query Data diagnostic code is to loop back the same data. The function code and diagnostic code are also echoed back making the response message the same as the query message.

Response for Function 09		**Example Message Bytes**			
Field Name		(Hex)	Decimal	Binary	**Explanation**
Slave Address		(0Ah)	10	0000 1010	Slave number 10
Modbus Function -program		(09h)	9	0000 1001	Program
Sub Code low cal.	(Hi byte)	(00h)	0	0000 0000	Returned sub code
	(Lo byte)	(02h)	0	0000 0010	
Data Field	(Hi byte)	(00h)	0	0000 0000	Returned Calibration Value
	(Lo byte)	(23h)	35	0011 0101	
Error Check	(CRC Lo)	(xxh)	xx	xxxxxxxx	
	(CRC Hi)	(xxh)	xx	xxxxxxxx	

Function 09 - Program (cont)

Program code 03h (High Calibration)
The data sent in the query data field is returned (looped back) in the response. The entire response message is identical to the query message.

Query
Below is an example of a request, to the Slave at Modbus address 10 (0Ah), to perform High Calibration using values placed in the data bytes.

Query for Function 09		Example Message Bytes (Hex)	Decimal	Explanation
Slave Address		(0Ah)	10	Slave number 10
Modbus Function		(09h)	8	Program
Program Code	(Hi byte)	(00h)	0	High calibration
	(Lo byte)	(03h)	0	
Data Field	(Hi byte)	(03h)	3	Calibration data 0 to 1023
	(Lo byte)	(23h)	35	
Error Check	(CRC Lo)	(xxh)	Xx	
	(CRC Hi)	(xxh)	Xx	

Response
The normal response to the Return Query Data diagnostic code is to loop back the same data. The function code and diagnostic code are also echoed back making the response message the same as the query message.

Response for Function 09		Example Message Bytes (Hex)	Decimal	Binary	Explanation
Slave Address		(0Ah)	10	0000 1010	Slave number 10
Modbus Function -program		(09h)	9	0000 1001	Program
Sub Code high cal.	(Hi byte)	(00h)	0	0000 0000	Returned sub code
	(Lo byte)	(03h)	0	0000 0010	
Data Field	(Hi byte)	(03h)	3	0000 0011	Returned Calibration Value
	(Lo byte)	(23h)	35	0011 0101	
Error Check	(CRC Lo)	(xxh)	xx	xxxxxxxx	
	(CRC Hi)	(xxh)	xx	xxxxxxxx	

Function 09 - Program (cont)

Program code 04h (Program Slew Timeout)
The data sent in the query data field is returned (looped back) in the response. The entire response message is identical to the query message.

Query
Below is an example of a request, to the Slave at Modbus address 10 (0Ah), to perform program slew timeout using values placed in the data bytes. Resolution is in 0.1sec. Example 35 is equal to 3.5sec

Query for Function 09		Example Message Bytes (Hex)	Decimal	Explanation
Slave Address		(0Ah)	10	Slave number 10
Modbus Function		(09h)	8	Program
Program Code	(Hi byte)	(00h)	0	Slew timeout
	(Lo byte)	(04h)	0	
Data Field	(Hi byte)	(00h)	0	Calibration data 0 to 255
	(Lo byte)	(23h)	35	
Error Check	(CRC Lo)	(xxh)	Xx	
	(CRC Hi)	(xxh)	Xx	

Response
The normal response to the Return Query Data diagnostic code is to loop back the same data. The function code and diagnostic code are also echoed back making the response message the same as the query message.

Response for Function 09		Example Message Bytes (Hex)	Decimal	Binary	Explanation
Slave Address		(0Ah)	10	0000 1010	Slave number 10
Modbus Function -program		(09h)	9	0000 1001	Program
Sub Code .	(Hi byte)	(00h)	0	0000 0000	Returned sub code
	(Lo byte)	(04h)	0	0000 0010	
Data Field	(Hi byte)	(00h)	0	0000 0000	Returned timeout Value
	(Lo byte)	(23h)	35	0011 0101	
Error Check	(CRC Lo)	(xxh)	xx	xxxxxxxx	
	(CRC Hi)	(xxh)	xx	xxxxxxxx	

Function 09 - Program (cont)

Program code 05h (Program Slave Address)
The data sent in the query data field is returned (looped back) in the response. The entire response message is identical to the query message.

Query
Below is an example of a request, to the Slave at Modbus address 10 (0Ah), to change its address to the new one, using values placed in the data bytes field. Slave will use this value after receiving function 9 code 7 (burn data into EEPROM)

Query for Function 09		Example Message Bytes (Hex)	Decimal	Explanation
Slave Address		(0Ah)	10	Slave number 10
Modbus Function		(09h)	8	Program
Program Code	(Hi byte)	(00h)	0	Program New ID
	(Lo byte)	(05h)	0	
Data Field	(Hi byte)	(00h)	0	Calibration data 1 to 127
	(Lo byte)	(23h)	35	
Error Check	(CRC Lo)	(xxh)	Xx	
	(CRC Hi)	(xxh)	Xx	

Response
The normal response to the Return Query Data diagnostic code is to loop back the same data. The function code and diagnostic code are also echoed back making the response message the same as the query message.

Response for Function 09		Example Message Bytes (Hex)	Decimal	Binary	Explanation
Slave Address		(0Ah)	10	0000 1010	Slave number 10
Modbus Function -program		(09h)	9	0000 1001	Program
Sub Code	(Hi byte)	(00h)	0	0000 0000	Returned sub code
	(Lo byte)	(05h)	0	0000 0010	
Data Field	(Hi byte)	(00h)	0	0000 0000	Returned timeout Value
	(Lo byte)	(23h)	35	0011 0101	
Error Check	(CRC Lo)	(xxh)	xx	xxxxxxxx	
	(CRC Hi)	(xxh)	xx	xxxxxxxx	

Function 09 - Program (cont)

Program code 06h (Program Communication Fail Option and Fail Timeout)

The data sent in the query data field is returned (looped back) in the response. The entire response message is identical to the query message. After loosing communication slave will change its state to fail mode – de-energize solenoid. Data value 00 means fail last – no change.

Query

Below is an example of a request, to the Slave at Modbus address 10 (0Ah), to program Communication fail timeout, using values placed in the data bytes field. Slave will use this value after receiving function 9 code 7 (burn data into EEPROM)

Query for Function 09		**Example Message Bytes** (Hex)	Decimal	**Explanation**
Slave Address		(0Ah)	10	Slave number 10
Modbus Function		(09h)	8	Program
Program Code	(Hi byte)	(00h)	0	Program Communication Fail Timeout
	(Lo byte)	(06h)	0	
Data Field	(Hi byte)	(00h)	0	Calibration data 0 to 256
	(Lo byte)	(5h)	5	(timeout in seconds)
Error Check	(CRC Lo)	(xxh)	Xx	
	(CRC Hi)	(xxh)	Xx	

Response

The normal response to the Return Query Data diagnostic code is to loop back the same data. The function code and diagnostic code are also echoed back making the response message the same as the query message.

Response for Function 09		**Example Message Bytes** (Hex)	Decimal	Binary	**Explanation**
Slave Address		(0Ah)	10	0000 1010	Slave number 10
Modbus Function –program		(09h)	9	0000 1001	Program
Sub Code	(Hi byte)	(00h)	0	0000 0000	Returned sub code
	(Lo byte)	(06h)	0	0000 0010	
Data Field	(Hi byte)	(00h)	0	0000 0000	Returned timeout Value
	(Lo byte)	(05h)	5	0000 0101	
Error Check	(CRC Lo)	(xxh)	xx	xxxxxxxx	
	(CRC Hi)	(xxh)	xx	xxxxxxxx	

Function 09 - Program (cont)

Program code 07h (Program non volatile memory with recently updated values)
The data sent in the query data field is returned (looped back) in the response. The entire response message is identical to the query message.

Query
Below is an example of a request, to the Slave at Modbus address 10 (0Ah), to program non-volatile memory with Slave address and timeouts values (burn data into EEPROM)

Query for Function 09		Example Message Bytes (Hex)	Decimal	Explanation
Slave Address		(0Ah)	10	Slave number 10
Modbus Function		(09h)	8	Program
Program Code	(Hi byte)	(00h)	0	Burn data into EEPROM
	(Lo byte)	(07h)	0	(Slave Address, and timeouts)
Data Field	(Hi byte)	(00h)	0	
	(Lo byte)	(00h)	0	
Error Check	(CRC Lo)	(xxh)	Xx	
	(CRC Hi)	(xxh)	Xx	

Response
The normal response to the Return Query Data diagnostic code is to loop back the same data. The function code and diagnostic code are also echoed back making the response message the same as the query message.

Response for Function 09		Example Message Bytes (Hex)	Decimal	Binary	Explanation
Slave Address		(0Ah)	10	0000 1010	Slave number 10
Modbus Function –program		(09h)	9	0000 1001	Program
Sub Code	(Hi byte)	(00h)	0	0000 0000	Returned sub code
	(Lo byte)	(07h)	0	0000 0111	
Data Field	(Hi byte)	(00h)	0	0000 0000	
	(Lo byte)	(00h)	0	0000 0000	
Error Check	(CRC Lo)	(xxh)	xx	xxxxxxxx	
	(CRC Hi)	(xxh)	xx	xxxxxxxx	

Function 15 (0Fh) - Force Multiple Outputs

Function Code 15
Forces each output in a sequence of outputs to either On or Off. When broadcast, the function forces the same output references in all attached slaves to the states specified.

Warning

Note: The function will override the controller's memory protect state and a output's disable state. The forced states will remain valid until the next time the controller's logic solves each output. The outputs not programmed in the controller's logic will remain forced, and are not automatically cleared upon power up. Thus, if an output not programmed in the controllers logic, is set On by function code 15 that output will remain On indefinitely.

Query
The query message specifies the outputs to be forced (starting Output Address and Number of Output). There are two independent outputs per Slave. The address zero is dedicated to solenoid. The Output Address can be zero or one and the number of outputs allowed should not be more than 2 minus Output Address.

The requested On/Off states are specified by the contents of the query data field. A logical 1 in a bit position of the field requests the corresponding output to be forced On and a logical 0 requests it to be forced Off. The least significant bit in the first Output Data Field will correspond to the output in the Output Address Field.

Query for Function 15		**Example Message Bytes**		
Field Name		(Hex)	Decimal	**Explanation**
Slave Address		(0Ah)	10	Slave number 10
Modbus Function		(0Fh)	15	Force Multiple Outputs
Output Address Field	(Hi byte)	(00h)	0	Output Address 0
	(Lo byte)	(00h)	0	
Quantity of Outputs Field	(Hi byte)	(00h)	0	One Output
	(Lo byte)	(01h)	1	
Byte Count		(01h)	1	One byte to follow
Output Data Field	1st byte	(01h)	1	0000 0001 Forces Output On
Error Check	(CRC Lo)	(AEh)	174	
	(CRC Hi)	(E4h)	228	

Function 15 (0Fh) Force Multiple Outputs (cont)

Response
The normal response returns the slave address, function code, starting output address, and the quantity of outputs forced.

Response for Function 15

Field Name		Example Message Bytes (Hex)	Decimal	Binary	Explanation
Slave Address		(0Ah)	10	0000 1010	Slave number 10
Modbus Function		(0Fh)	15	0000 1111	Force Multiple Outputs
Outputl Address Field	(Hi byte)	(00h)	0	0000 0000	Output Address 0
	(Lo byte)	(00h)	0	0000 0000	
Quantity of Output Field	(Hi byte)	(00h)	0	0000 0000	One Output Forced
	(Lo byte)	(01h)	1	0000 0001	
Error Check	(CRC Lo)	(95h)	149	1001 0101	
	(CRC Hi)	(70h)	112	0111 0000	

Exception codes for Function 15
02 ILLEGAL DATA ADDRESS
The data address received in the query is not an allowable address for the slave. This error will be returned if the output address is more than one (since the Direct Intellis I/O module has only two outputs).

03 ILLEGAL DATA VALUE
A value contained in the query data field is not an allowable value for the Slave. This error will be returned if the quantity of output data is more than two minus Output Address

Non-Implemented Modbus Commands

Non-Implemented Function codes

06	Preset Single Register
07	Read Exception Status
11 (0Bh)	Fetch Communication Event Counter
12 (0Ch)	Fetch Communication Event Log
16 (10h)	Preset Multiple Registers
17 (11h)	Report slave ID
20 (14h)	Read General Reference
21 (15h)	Write General Reference
22 (16h)	Mask Write 4x Registers
23 (17h)	Read/Write 4x Registers
24 (18h)	Read FIFO Query

Non-Implemented Diagnostic Codes for Function 08 Diagnostic Test

01	Restart Communications Option
03	Change ASCII Input Delimiter
04	Force Listen Only Mode
11 (0Bh)	Return Bus Message Count
12 (0Ch)	Return Bus Communication Error Count
13 (0Dh)	Return Bus Exception Error Count
14 (0Eh)	Return Slave Message Count
15 (0Fh)	Return Slave No Response Count
16 (10h)	Return Slave NAK Count
17 (11h)	Return Slave Busy Count
18 (12h)	Return Bus Character Overrun Count
19 (13h)	Return IOP Overrun Count (884)
20 (14h)	Clear Overrun counter and Flag (884)
21 (15h)	Get/Clear Modbus Plus Statistics

Chapter 3

Exception Responses

- Exception Responses
- Exception Codes

Exception Responses

Except for broadcast messages, when the Master device sends a query to a Slave it expects a normal response. One of four possible events can occur after the Masters query.

If the Slave receives the query without a communication error and can handle the query normally, it returns a normal response.

If the Slave does not receive the query due to a communication error, no response is returned. The Master program will eventually process a timeout condition of the query.

If the Slave receives the query but detects a communication error (framing, parity or CRC), no response is returned. The Master program will eventually process a timeout condition of the query.

If the Slave receives the query without a communication error, but cannot handle it (for example, if the request is to read a non-existent output or register), the Slave will return an exception response informing the Master of the nature of the error.
The exception response message is comprised of two fields that differentiate it from a normal response:

Function Code Field:
In a normal response, the Slave echoes the function code of the original query in the function code field of the response. All function codes in binary form have a most-significant bit of 0 (their values are all below 80h). In an exception response, the slave sets the most-significant bit of the function code to a 1. This makes the function code value in an exception response exactly 80 hexadecimal higher than the value would be for a normal response.

When a response is returned with the function code's most-significant bit set, the Master's application program can recognize the exception response and can examine the data field for the exception code.

Data Field:
In a normal response, the Slave may return data statistics in the data field (any information that was requested in the query). In an exception response, the Slave returns an exception code in the data field. This defines the Slave condition that caused the exception.

Exception Responses (cont)

Exception Response Example
The following is an example of a Master query massage and the Slave response.

The Master address a query to Slave device 10 (0Ah), with function code 01 (01=Read Output Status), starting at output address 1185 (04A1h), and requesting status for one output.

Query		**Example Message Bytes** (Hex)	Decimal	**Explanation**
Field Name				
Slave Address		(0Ah)	10	Slave number 10
Modbus Function		(01h)	1	Read Output Status
Starting Output Address	(Hi byte)	(04h)	4	Starting at Output (04A1h)=1185
	(Lo byte)	(A1h)	161	
Number of Outputs	(Hi byte)	(00h)	0	Number of outputs to send is 1
	(Lo byte)	(01h)	1	
Error Check	(CRC Lo)	(ACh)	172	
	(CRC Hi)	(63h)	99	

Exception Response		**Example Message Bytes** (Hex)	Decimal	Binary	**Explanation**
Field Name					
Slave Address		(0Ah)	10	0000 1010	Slave number 10
Modbus Function		(81h)	129	1000 0001	Read Output Status Exception
Exception Code		(02h)	2	0000 0010	Illegal Data Address
Error Check	(CRC Lo)	(B0h)	176	1011 0000	
	(CRC Hi)	(53h)	83	0101 0011	

The output address does not exist in the Slave, so the Slave returns the exception response for the function code 01.

A listing of exception codes begins on the next page.

Exception Codes

Implemented Exception Codes

Code	Name	Meaning (general description)
01	**ILLEGAL FUNCTION**	The function code received in the query is not an allowable action for this Salve. If a Poll Program Complete command was issued, this code indicates that no program function preceded it.
02	**ILLEGAL DATA ADDRESS**	The data address received in the query is not an allowable address for this Slave.
03	**ILLEGAL DATA VALUE**	A value contained in the query data field is not an allowable value for this Slave.
04	**SLAVE DEVICE FAILURE**	An unrecoverable error occurred while the Slave was attempting to perform the requested action.
06	**SLAVE DEVICE BUSY**	The Slave is engaged in processing a long duration program command. The Master should retransmit the message later when the Slave is free.

Non-Implemented Exception Codes

Code	Name	Meaning (general description)
05	**ACKNOWLEDGE**	The Slave has accepted the request and is processing it, but a long duration of time will be required to do so. This response is returned to prevent a time-out error from occurring in the Master. The Master can next issue a Poll Program Complete message to determine if processing is completed.
07	**NEGATIVE ACKNOWLEDGE**	The Slave cannot perform the program function received in the query (unsuccessful request using function code 13 or 14 decimal).
08	**MEMORY PARITY ERROR**	The Slave attempts to read extended memory, but detected a parity error in the memory. The Master can retry the request, but service may be required on the Slave.

Appendix A

Summary

- **Implemented Modbus Commands**
- **Modbus port setup**

Implemented Modbus Command

Function 01. Read Output Status
Is used for bitwise reading at the control valve solenoids (reads last state of solenoid driver). If bit corresponding to valve is:
1 = Solenoid of that valve is in the energized state.
0 = Solenoid of that valve is in the de-energized state.

Function 02. Read Input Status
Is used for reading (bitwise) status of valves. The 16 bits of status information are bitmapped as shown below:
Bit 0 = Valve Closed Limit Switch
Bit 1 = Valve Open Limit Switch
Bit 2 = User Input 1
Bit 3 = User Input 2
Bit 4 = User Input 3
Bit 5 = User Input 4
Bit 6 = Temperature Fault
Bit 7 = Slew Timeout Failure
Bits 8- 15 = Future Use

Function 03. Read Holding Registers
Is used for reading control valve Low and Hi calibration values. Each valve has two registers with two bytes in each register.

Function 04. Read Input Registers
Is used for reading control valve breakaway time, cycle time and valve position. Each valve has three registers with two bytes in each register.

Function 05. Force Single Output
Is used to force the solenoid of one valve.
FF00 = Force solenoid On (energized)
0000 = Force solenoid Off (de-energized)

Function 08. Diagnostic Test
Supports Diagnostic Code:
1. 00h (Return Query Data).
2. 02h (Return Cycle Count Register)
3. 0Ah (Reset Cycle Count Register)

Function 09. Program
Supports Codes:
1. 00h (Low/High Calibration -remotely).
2. 02h (Low Calibration – explicit value)
3. 03h (High Calibration – explicit value)

4. 04h (Program Slew Timeout)
5. 05h (Program Slave Address)
6. 06h (Communication Fail time/mode)
7. 07h (Burn info into EEPROM)

Function 15. Force Multiple Outputs
Is used to force the solenoids on multiple control valves (bitwise). If bit corresponding to valve is:
1 = Force solenoid On (energized)
0 = Force solenoid Off (de-energized)

Modbus Port Setup

1. Mode of Transmission
Only RTU (Remote Terminal Unit) mode is used. The format for each byte is:
1 start bit
8 data bits (least significant bit sent first)
1 parity bit, (even parity used)
1 stop bit

2. Slave Address
From 1 to 127 - selectable by DIP switch or programmed remotely over MODBUS.

3. Error Check Field
Uses 2 byte Cyclical Redundancy Check (CRC) at end of each message.

4. Baud Rate
Two Baud Rates, 9600 or 19200 (selectable by DIP switch).

Implemented Modbus Command

5. Inter-Byte Timing
Response messages are normally transmitted with zero gap between the stop bit of one byte and the start bit of the next byte.
Slaves can tolerate gaps in received messages of up to 1.7 mSec.
Gaps 4 mSec or greater are considered a new message frame.

6. Hardware Configuration
Uses an RS-485 compatible 2-wire (twisted pair with grounded shield) multidrop circuit.

7. Communication Line Length
The length of the RS-485 communication line should be less than 4000 feet.