

iSeries

CE

Temperature & Humidity
Controllers

Communication Manual



NEWPORT Electronics, Inc.

<http://www.newportUS.com/i>



1. Overview

Generally the Communication Functions of **iTH** Controllers operates as the iSeries Temperature Controllers i8, i16, and i32 in the temperature measurement models. However Some Number of same or unsupported communication command indexes, which are operated differently and based on the functions and the feature of these models, must be clarified and described in this manual.

This manual describes how to use a digital communication format, function command/data configuration and NEWPORT or MODBUS communication protocols to operate iTH Temperature/Humidity (%RH) controllers. It has been assumed that the user has some experience of communication protocols and some familiarity with iTH Temperature/Humidity (%RH) controllers.

COMMUNICATION SETUP

1.1 Flow chart.

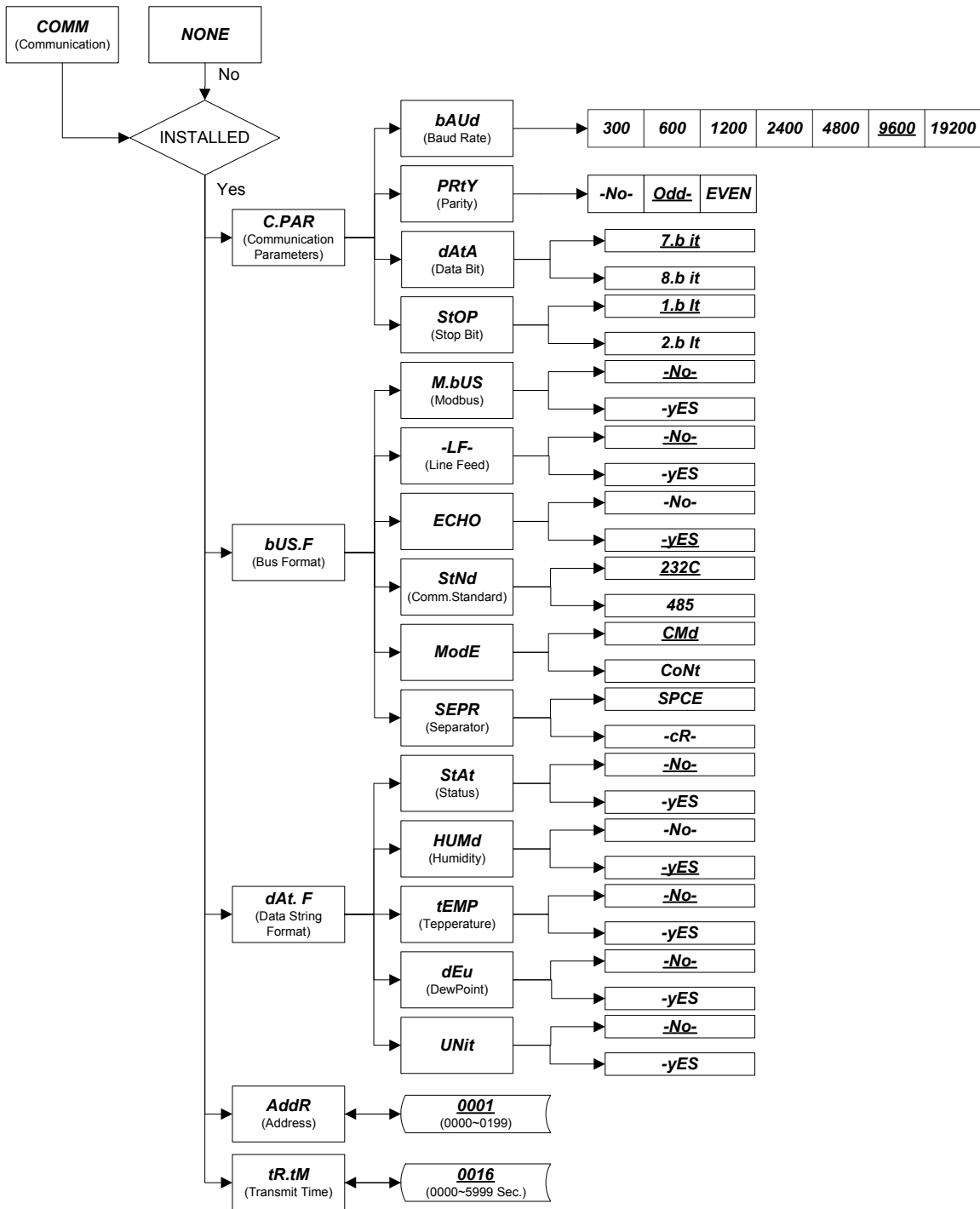


Figure 1.1 Flow Chart for Communication Option

Note: Underlined options and values of The Flow Chart are default ones by factory for more detail description of the chart items, refer to Table 1.1.

1.2 Setup the controller through the front panel

You can setup you controller by pressing the push buttons on the front panel

ENTER COMMUNICATION OPTION MENU:

- Press **⏩** **1)** Press **⏩** until "**CNFG**" prompt appears.
 Press **⏪** **2)** Display advances to "**INPt**" Input Menu.
 Press **⏩** **3)** Press **⏩**, until Display advances to "**COMM**" Communication options menu.
 Press **⏪** **4)** Display advances to "**C.PAR**" Communication Parameters submenu

⏩ - Use this button to advance/navigate through all communication menu items.

⏪ - Press this button to access the submenus from a top level of communication menu item.

Press this button to store a submenu selection

▲ - Press the up button to scroll through "flashing" selection. When a numerical value is displayed, press this key to change a value of this parameter.

▼ - Press this button to go back to a previous Communication menu item. Press this button twice to Reset the controller to Run mode.

1.3 Abbreviations, range, default setup

The Communication Menu Display items use some abbreviations and compact wording shown on table 1.1

Table 1.1: Communication Flow Chart / LED Display Menu Items Description.

Display (abbreviations)	Function	Range/Definition	Factory Default
C.PAR	Communication Parameter Menu:		
bAUd	Baud rate	300, 600,1200, 2400, 4800, 9600, 19200	9600
PRtY (odd , EVEN , No)	Parity	Odd, Even, None	odd
dAtA (7.bit , 8.bit)	Data bit	7 bit, 8 bit	7.bit
StOP (1.bit , 2.bit)	Stop bit	1 bit, 2 bit	1.bit
bus.F	Bus Format Menu:		
M.bUS	Modbus protocol	Yes – Modbus protocol enabled No – Newport protocol enabled	_No_
LF	Line feed	Yes – print on every other line No – print on every line	_No_
ECHO	Echo	Yes – echo the command parameter No – no echo	_YES
StNd (232C , 485)	Communication Standard	2 Serial Communication Standards: RS232 or RS485	232C
ModE (CMd , CoNt)	Data Flow Mode	* Command – operate in Command mode (respond to valid command). * Continuous – operate in Continuous mode (transmit different measurement values continuously on the bus).	CMd_
SEPR (SPCE , _cR_)	Data Separation Character	Space – space inserted after each piece of data. Carriage Return – carriage return inserted after each piece of data	SPCE
dAt.F	Data Format Menu:		

stAt	Alarm Status	Yes – enables the transmission of alarms value No – disable	_No_
HUMd	Humidity	Yes – enables the transmission of Humidity value No – disable	_Yes
tEMP	Temperature	Yes – enables the transmission of Temperature value No – disable	_No_
dEu	DewPoint	Yes – enables the transmission of DewPoint value No – disable	_No_
UNit	Units	Yes – enables the transmission of units of measurement No – disable	_No_
Ad:dR	Multipoint Address	0000 to 0199 – addressed meter	0001
tR.tM	Transmit Time Interval	0000 to 5999 sec – transmission time interval between consecutive transmissions in continuous mode.	0016
	Recognition Character	20 Hex to 7F Hex (32 to 127 Dec) – refer to ASSII Character Chart or Appendix A, except “^”, “A”, “E”	*

Note: 1. There is no **Continuous** mode, when device is configured to use the RS485 interface standard.

2. The **Multipoint Address** will be included in the transmission data if RS485 standard has been selected in menu items.

3. **Transmit time** is available only when device has configured for Continuous mode and RS232 Standard

4. If the meter in point-to-point **Continuous mode**, it ignores any transmitted commands except Ctrl S , which will stop transmission.

2. Communication Hardware & Wiring Interfaces: See **Note (!)**
3. Command Type/Prefix Letter: See **Note (!)**
4. Command Format: See **Note (!)**
5. Communication Protocols: Serial RS232 or RS485 are identical to iSeries Controllers. See **Note (!)**

Note (!): Refer to iSeries Monitor/Controller Communications Manual or it can be found via this weblink: <http://newportus.com/Pdf/M3397CN.pdf>

6. Communication Operations of iTH Controllers:

Since Communication Command Type, Format, Parameters and operation of these controllers are similar to iSeries Temperature Controllers, therefore there are particular Command Index / Register Locations operate in different manners for iTH’s new functional features as following Table 6.1 Guideline for Manual Reader/User below as referring to Table 7.1. Description of Command Index/Register Location of iTH Humidity (%RH) Controllers:

Command Index / Register Location # Function	In Table 6.2
Humidity (%RH) Measurement Operation	1
	Example: 1/ AL1CNF Humidity Alarm Configuration Function 2/ X1 : send request command for RH% Humidity reading
Temperature (°C or °F) Measurement Operation	2
	Example: 1/ AL2CNF Temperature Alarm Configuration Function 2/ X2 : send request command for Temperature reading
Exception:	Only X3 : send request command for DewPoint reading
	<ul style="list-style-type: none"> Or if identified as:
Unused / Not Applicable	N/A
iTH featured property	*

Table 6.1. Guideline for Manual Reader/User

7. For detailed Description of How to use the following command indexes in Table 7.1. , Refer to iSeries Monitor/Controller Communications Manual or it can be found via this weblink: <http://newportus.com/Pdf/M3397CN.pdf>

Command	Command Index/ Register Location #	Function	Command # of bytes	# Of Characters	Default Value (hex)
RW	01	SP1 (*)	3	6	200000
RW	02	SP2 (**)	3	6	200000
-	03	N/A	-	-	-
RW	04	Analog Output Offset	3	6	C00000
RW	05	ID (Access Security ID)	2	4	0000
-	06	N/A	-	-	-
-	07	N/A	-	-	-
GPRW	08	RDGCNF (Reading Config.) (***)	1	2	4B
RW	09	ALR1CNFG	1	2	00
RW	0A	ALR2CNFG	1	2	00
RW	0B	LOOP BREAK TIME	2	4	003B
RW	0C	OUT1CNF (MSB is for internal use)	1	2	81
RW	0D	OUT2CNF	1	2	60
RW	0E	RAMP TIME	2	4	0000
RW	0F	Analog Output Scale	3	6	7186A0
RW	10	COMM.PARAMETERS	1	2	0D
RW	11	COLOR	1	2	09
RW	12	ALR1LO (*)	3	6	200000
RW	13	ALR1HI (*)	3	6	200320
-	14	N/A	-	-	-
RW	15	ALR2LO (**)	3	6	200000
RW	16	ALR2HI (**)	3	6	200320
GPRW	17	PB1/DEAD BAND	2	4	00C8
GPRW	18	RESET 1	2	4	00B4
GPRW	19	RATE 1	2	4	0000
GPRW	1A	CYCLE 1	1	2	07
-	1B	N/A	-	-	-
GPRW	1C	PB2/DEAD BAND	2	4	00C8
GPRW	1D	CYCLE 2	1	2	07
RW	1E	SOAK TIME	2	4	0000
RW	1F	BUS FORMAT(2 MSB are for internal used)	1	2	94
GPRW	20	DATA FORMAT (****)	1	2	02
RW	21	ADDRESS	1	2	01
RW	22	Transit Time Interval	2	4	0010
-	23	N/A	-	-	-
RW	24	Miscellaneous	1	2	00
RW	25	Reading Adjust	3	6	200000
RW	26	Recognition Character	1	2	2A
RW	27	%LOW	1	2	00
RW	28	%HI	1	2	63
D	01	DISABLE ALARM 1	0	0	-
D	02	DISABLE ALARM 2	0	0	-
D	03	STANDBY	0	0	-
D	04	DISABLE SELF	0	0	-
E	01	ENABLE ALARM 1	0	0	-

E	02	ENABLE ALARM 1	0	0	-
E	03	DISABLE STANDBY	0	0	-
E	04	ENABLE SELF	0	0	-
X	01	SEND %RH READING	0	0	-
X	02	SEND TEMPERATURE READING	0	0	-
X	03	SEND DEWPOINT READING	0	0	-
U	01	SEND ALARM STATUS	0	0	-
U	03	SEND SW VERSION	0	0	-
V	01	SEND DATA STRING	0	0	-
Z	02	HARD RESET	0	0	-

Table 7.1 Description of Command Index/Register Location of iTH Humidity (%RH) Controllers.

(*):Command Index / Register Location: **01: (SetPoint /SP1); 12(ALR1Lo) and 13(ALR1Hi) Configuration:**

SetPoint Configuration Command Index/Register Location 01; 12 and 13 BIT position (3 bytes=8 Hex. Characters=24 Binary Bits)							Function
23	22	21	20	19	~	0	
0							Positive sign (+)
N/A							N/A for % RH Measurement
	0	1	0				Decimal Point Position: FFF.F Only supported
				0	~	0	SetPoint Value
				1	~	1	
				1	~	1	

Table 7.2 : Description of SetPoint 1(SP1) Register 01; Alarm1 Low(ALR1/ALR.L) Register 12 and Alarm1 High (ALR1/ALR.H) Register 13 Configuration.

(**):Command Index / Register Location **02: SetPoint 2 (SP2); 15 (ALR2 Lo) and 16 (ALR2 Hi) Configuration:**

SetPoint Configuration Command Index/Register Location 02, 15 and 16 BIT position (3 bytes=8 Hex. characters=24 Binary Bits)							Function
23	22	21	20	19	~	0	
0							Positive sign (+) or SP2 is Positive Temp.
1							Negative sign (-) or SP2 is Negative Temp.
	0	1	0				Decimal Point Position: FFF.F Only supported
				0	~	0	SetPoint Value
				1	~	1	
				1	~	1	

Table 7.3 : Description of SetPoint 2(SP2) Register 02, Alarm2 Low(ALR2/ALR.L) Register 15 and Alarm2 High 2(ALR2/ALR.H) Register 16 Configuration.

Examples:

1. To reset the controller, send ***Z02** (table 8.)
2. To read set point 1, send ***R01** (table 6.2)
3. To change Set Point 1 to 100.0, send ***W012003E8** (see explanation below)

Description: The value to write for changing Set Point is: **2003E8 = 3 hex bytes = (3 x 8 =24 binary bit positions)** 23~0 bit positions of binary number/value (2 hex. characters in each byte= 8 binary bits)

The 23rd Binary Bit Position (from right to left)
=

0 : positive sign

1 : negative sign

The 22nd, 21st & 20th Position (from right to left) = binary value is default as 010 :
Decimal Point Position 2 (FFF.F)
supported only.

The remaining binary positions:
19~0 = Set point data (Value)

Therefore to change SetPoint value to 100.0:

- Positive sign = 0
- Decimal Point 2 = 010 Binary or FFF.F
- Set point data 1000 = 3E8 Hex = 001111101000 Bin

Then the command data = 0010 0000 0000 0011 1110 1000 Bin = 2003E8 Hex.

2 0 0 3 E 8 Hex

Send ***W012003E8**

where:

- *W01: write to Set Point 1
- 2003E8: Set point data in hexadecimal format including sign and decimal point

4. To send the same as above for RS485 with transmit address 02, the command is

Send *02W012003E8.

(**): Command Index / Register Location **08: Reading Configuration**

Reading Configuration Command Index/Register Location 08 BIT position (Default value: 4B = 0100 1011)								Function
7	6	5	4	3	2	1	0	
							0	°F° Sensor
							1	% RH Sensor
				0				°C
				1				°F
0	0	0						Filter Constant 1
0	0	1						Filter Constant 2
0	1	0						Filter Constant 4
0	1	1						Filter Constant 8
1	0	0						Filter Constant 16
1	0	1						Filter Constant 32
1	1	0						Filter Constant 64
1	1	1						Filter Constant 128

Table 7.4 : Description of Reading Configuration (**RdG**) Register 08.

(****) Command Index / Register Location **20: Data Format**

DATA (String) FORMAT Command Index/ Register Location 20 BIT position (Default Value: 02 = 0000 0010)								Function
7	6	5	4	3	2	1	0	
							0	ALARM STATUS: NO (not Included)
							1	ALARM STATUS: YES (Included)
							0	%RH READING: NO (not Included)
							1	%RH READING: YES (Included)
					0			TEMPERATURE READING: NO (not Included)
					1			TEMPERATURE READING: YES (Included)
				0				DEWPOINT READING: NO (not Included)
				1				DEWPOINT READING: YES (Included)
	0							MEASURE UNIT: NO (not Included)
	1							MEASURE UNIT: YES (Included)

Table 7.5 : Description of DATA FORMAT(**dAt.F**) Register 20 Configuration.

8. MODBUS PROTOCOL

Note: To Enable the Modbus Protocol, set Modbus menu item to “**Yes**” in the Bus Format submenu of the Communication menu.

8.1 Introduction

Modbus protocol defines a message structure that controllers will recognize and use, regardless of the type of networks over which they communicate. It describes the process a controller uses to request access to another device, how it will respond to requests from the other devices, and how errors will be detected and reported. It establishes a common format for the layout and contents of message fields. The Modbus protocol provides the internal standard that the Newport iTH Temperature/Humidity (%RH) controllers use for parsing messages. During communications on a Modbus network, the protocol determines how each controller will know its device address, recognize a message addressed to it, determine the kind of action to be taken, and extract any data or other information contained in the message. If a reply is required, the controller will construct the reply message and send it using Modbus protocol.

Modbus defines a digital communication network to have only one MASTER and one or more SLAVE devices. Either a single (point-to-point) or multi-drop network (multipoint) is possible.

Newport iTH Temperature/Humidity (%RH) controllers communicate on standard Modbus networks using RTU (Remote terminal unit) transmission mode.

8.2 RTU mode

In RTU mode, each eight-bit byte in a message contains two four-bit hexadecimal characters. The main advantage of this mode is that its greater character density allows better data throughput than ASCII for the same baud rate. Each message must be transmitted in a continuous stream.

The following format used for each byte sent and received by Newport iTH Temperature/Humidity (%RH) controllers in RTU mode:

1. Eight-bit binary, Hexadecimal (0 ... 9, A ... F)
2. Two hexadecimal characters contained in each eight-bit field of the message
3. 1 start bit, 8 data bits, 1 Stop Bit (No Parity Bit)

The figure below shows the bit sequences when byte transmitted in RTU mode.

	LSB					MSB				
START	1	2	3	4	5	6	7	8	STOP	

- LSB – Least Significant Bit sent first

❖ The Modbus Message frame is shown below

DEVICE ADDRESS	FUNCTION CODE	DATA	CRC CHECK
8 BITS hh	8 BITS hh	k x 8 BITS hhh....	16 BITS hhhh

where:

- h (hex. Number) – character,
- k – integers depend on the contents of the data format.

8.3 Device Address

The address message frame contains eight bits. The slave device addresses are in the range of 1 ... 199 decimal. A master addresses 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 this address field of the response to let the master know which slave is responding.

Address 0 is used for the write command broadcast that commands all controllers on network, which all slave devices recognize

8.4 Function Code

The function code field of a message frame contains eight bits (RTU). Valid codes are in the range of 1 ... 255 decimal. Of these, some codes are applicable for i-series controllers.

When a message is sent from a master to a slave device the function code field tells the slave what kind of action to perform.

The following functions are supported by Newport iTH Temperature/Humidity (%RH) controllers:

Function Code	Function	Description
03	Read holding register	Reads the binary contents of holding registers in the slave
04	Read input register	Reads the binary contents of input register in the slave.
06	Preset (Write to) single register	Preset (Write) a value into single holding register
08	Diagnostic	Series of tests for checking communication between master and slave

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

8.5 Data Field

The data field is constructed using sets of two hexadecimal digits, in the range of 00 to FF hexadecimal. The data field of messages sent from a master to slave devices contains additional information, which the slave must use to take the action defined by the function code. This can include items like discrete and register addresses, the quantity of items to be handled, and the count of actual data bytes in the field.

8.6 CRC Checking

With RTU mode the error checking field contains a 16-bit value implemented as two eight-bit bytes (HI-order byte and Low-order byte). The error check value is the result of a Cyclical Redundancy Check (CRC) calculation performed on the message contents.

After building a message (address, function code, data) the transmitting device calculates a CRC code and puts it to the end of the message. A receiving device will calculate a CRC code from the message it has received and compare against transmitted CRC code. If these CRC codes are different, there has been a communication error. Newport iTH Temperature/Humidity (%RH) controllers will not reply if they detect a CRC error.

Sequences of CRC calculation:

1. Load a 16 bit CRC register with all 1's.
2. Apply first 8 bit byte of the message to the least significant bit (LSB) of the contents of the register.
3. Exclusive OR these 8 bit with the register contents.
4. Shift the result one bit to the right with zero entering into the most significant bit (MSB) position and evaluate the LSB.
5. If over flow bit in LSB is 1, exclusive OR the latest register contents with A001 Hex value.
6. If over flow bit in LSB is 0, no exclusive OR occurs (repeat step 4).
7. Repeat steps 4, 5 and 6 until 8 shifts have been performed.
8. Apply next 8 bit byte of the message to the LSB contents of the register.
9. Exclusive OR these 8 bit with the register contents.
10. Repeat steps 4 to 9 until all bytes of the message have been processed.
11. The final content of the register is the CRC value.

Example of CRC calculation sees in Appendix B

Note: When CRC is placed into the end of the message, the low order byte of the CRC will be transmitted first, followed by the high order byte.

8.7 Modbus RTU Registers

The table below shows the Modbus registers supported by Newport iTH Temperature/Humidity (%RH) controllers:

Table 8.1 Modbus Registers

FUNCTION CODE	REGISTER	FUNCTION	VALUE, RANGE (Decimal)
NO	00	N/A	
03/04, 06	01	SET POINT 1	0 to 100
03/04, 06	02	SET POINT 2	-40 to 254
NO	03	N/A	
NO	04	N/A	
03/04, 06	05	ID	0 to 9999
NO	06	N/A	
NO	07	N/A	0 to 255
03/04, 06	08	RDGCNF	0 to 255
03/04, 06	09	ALR1CNF	0 to 255
03/04, 06	0A	ALR2CNF	0 to 255
03/04, 06	0B	LOOP BREAK TIME	00:00 to 99:59
03/04, 06	0C	OUT1CNF	0 to 255
03/04, 06	0D	OUT2CNF	0 to 255
03/04, 06	0E	RAMP TIME	00:00 to 99:59
NO	0F	N/A	
03/04, 06	10	COMM. PARAMETERS	0 TO 255
NO	11	N/A	
03/04, 06	12	ALR1 LOW	0 to 100
03/04, 06	13	ALR1 HI	0 to 100
NO	14	N/A	0 to 255
03/04, 06	15	ALR2 LOW	-40 to 254
03/04, 06	16	ALR2 HI	-40 to 254
03/04, 06	17	PB1/DEAD BAND 1	0 to 9999
03/04, 06	18	RESET 1	0 to 3999
03/04, 06	19	RATE 1	0 to 399.9
03/04, 06	1A	CYCLE 1	1 to 199
NO	1B	N/A	
03/04, 06	1C	PB2/DEAD BEND 2	0 to 9999
03/04, 06	1D	CYCLE 2	1 to 199
03/04, 06	1E	SOAK TIME	00:00 to 99:59
03/04, 06	1F	BUS FORMAT	0 to 255
03/04, 06	20	DATA FORMAT	0 to 255
03/04, 06	21	ADDRESS	0 to 199
03/04, 06	22	TRANSMIT TIME	0 to 9999
NO	23	N/A	
NO	24	N/A	
NO	25	N/A	
03/04, 06	26	RECOGNITION CHAR.	32 to 126
03/04	27	RH(%) VALUE	0 to 100
03/04	28	TEMP. VALUE	-40 to 254
03/04	29	DEWPOINT VALUE	
03/04	2A	SOFTWARE VERSION	
06	2B	RESET	0000

8.8 Command Format

The following formats are used to SEND commands by computer and RETURNED by device.

8.8.1 Read Multiple Register (03 or 04)

Sent to device:

DEVICE ADDRESS	FUNCTION CODE 03 or 04	DATA				CRC	
		STARTING REGISTER		NUMBER OR REGISTERS			
1 BYTE hh	1 BYTE 03	HB 00	LB hh	HB 00	LB hh	LB hh	HB hh

Returned by device:

DEVICE ADDRESS	FUNCTION CODE 03 or 04	DATA						CRC	
		NUMBER OF BYTES	FIRST REGISTER		...	n REGISTER			
1 BYTE hh	1 BYTE 03	1 BYTE hh	HB hh	LB hh	...	HB hh	LB hh	LB hh	HB hh

Where: HB – High Order Byte
 LB – Lower Order Byte
 Unused bits are set to zero

Note: Newport iTH Temperature/Humidity (%RH) controllers support only Read Single Register, so the number of registers should always set to 1.

Example:

Send to the device: Address 1, Read (03) register 1 (Set point 1)

DEVICE ADDRESS	FUNCTION CODE	DATA				CRC	
		STARTING REGISTER		NUMBER OR REGISTERS			
01	03	00	01	00	64	29	E1

Note: To determine the appropriate registers see table 8.1

Returned by device: Set point 1 set to 100.0

DEVICE ADDRESS	FUNCTION CODE	DATA			CRC	
		NUMBER OF BYTES	VALUE OF REGISTER			
01	03	02	00	64	B9	AF

0064 Hex = 100.0 Dec

Since Newport iTH Temperature/Humidity (%RH) controllers support only one fixed Decimal Point Position 2 (FFF.F). Therefore user must refer to Table 7.2 for details of Decimal Point Position and Register 01/Setpoint1 Configuration Format.

8.8.2 Write to single Register (06)

The following command will write a parameter to the single register.

Sent to/Return from device :

DEVICE ADDRESS	FUNCTION CODE 06	DATA				CRC	
		REGISTER		DATA / VALUE			
1BYTE hh	1BYTE 06	HB 00	LB hh	HB hh	LB hh	LB hh	HB hh

Example: Set Alarm2 Low (register 15) to -20.0 Dec (FF38 Hex)

Send to device: Address 01, write (06) to register 15 value -20.0 (FF38 Hex) as displayed on the 1st Command String and read (03) the same register 15 to verify the value of write command executed:

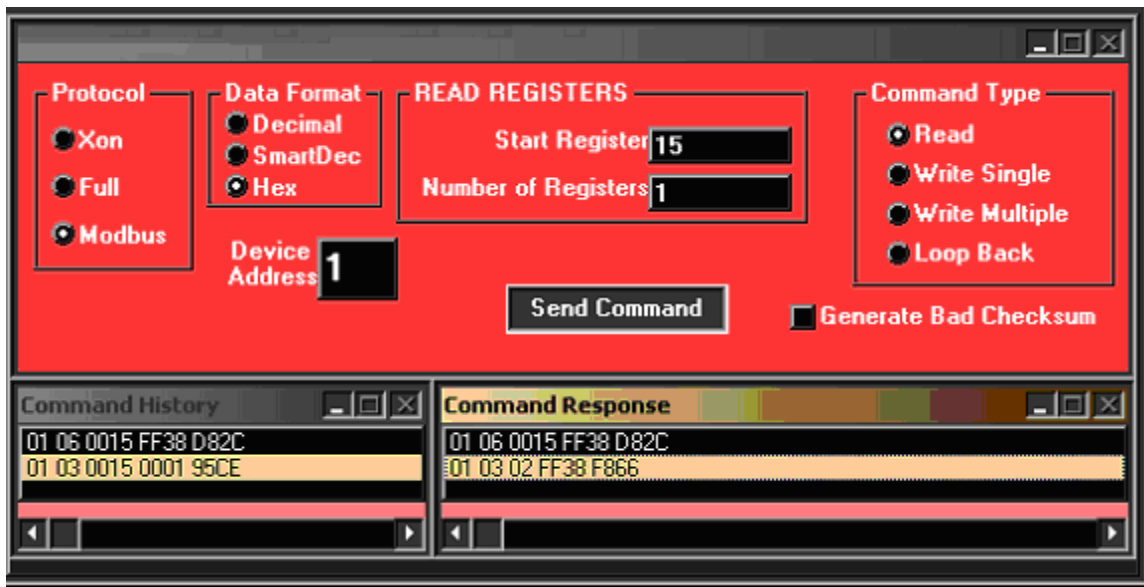


Figure 8.1: Modbus Command String Format Example.

Note: iTH Temperature/Humidity (%RH) controllers support only Write to Single Register command

DEVICE ADDRESS	FUNCTION CODE	DATA				CRC	
		REGISTER		DATA / VALUE			
01	06	00	15	FF	38	D8	82

Returned from device:

DEVICE ADDRESS	FUNCTION CODE	DATA				CRC	
		REGISTER		DATA / VALUE			
01	06	00	15	FF	38	D8	82

*Since Newport iTH Temperature/Humidity (%RH) controllers support only one fixed Decimal Point Position 2 (FFF.F), therefore we only need to convert –200 dec. to hex. value as following and disregard Decimal point:

N = +200 Dec = 0000 0000 1100 1000 Bin = 2 bytes or 16 bits

1's complement of N = 1111 1111 0011 0111 Bin = Not N

2's complement of N = 1111 1111 0011 1000 Bin = 1's complement of N + 1_{LSB}

EQUIVALENT: **F F 3 8** Hex

8.8.3 Diagnostic command.

This command echoes the sent message to indicate that the communication link is established correctly.

Send to/Return from device:

DEVICE ADDRESS	FUNCTION CODE	DIAGNOSTIC CODE		LOOPBACK DATA		CRC	
		HB	LB	HB	LB	LB	HB
1 BYTE hh	1 BYTE 08	00	00	hh	hh	hh	hh

Where: Diagnostic Code is two byte code to determine the type of test to be performed.

Newport iTH Temperature/Humidity (%RH) controllers supported only "00" code which requested slave to echo sent command back to the master

Example:

Send to device: Address 01, Diagnostic command (08), data value 8755

DEVICE ADDRESS	FUNCTION CODE	LOOPBACK DATA		DIAGNOSTIC CODE		CRC	
01	08	22	33	00	00	BE	B8

Dec (2233 Hex)

DEVICE ADDRESS	FUNCTION CODE	LOOPBACK DATA		DIAGNOSTIC CODE		CRC	
01	08	22	33	00	00	BE	B8

Returned by device:

8.8.4 Error Response

When a device can not properly respond to the command due to incorrect or corrupted command it will respond with an error message. The error message has the following format:

DEVICE ADDRESS	FUNCTION CODE	ERROR RESPONSE	CRC	
1 BYTE hh	1 BYTE hh	1 BYTE hh	LB hh	HB hh

Newport iTH Temperature/Humidity (%RH) controllers support the following error code messages:

02 – read from/write to the illegal register – read from/write to the register, which is inactive, or not supported by Newport iTH Temperature/Humidity (%RH) controllers.

03 – write an illegal value – write out of range value

Example:

Send to device: Address 05, read (03) register 04 - inactive (see table 8.1)

DEVICE ADDRESS	FUNCTION CODE	STARTING REGISTER		NUMBER OF REGISTERS		CRC	
05	03	00	04	00	01	C5	CB

Returned by device:

DEVICE ADDRESS	FUNCTION CODE	ERROR RESPONSE	CRC	
01	83	02	C0	F1

Example:

Send to device: Address 120 (Hex 78), write (06) to register 35 (Hex 23) - inactive (see table 8.1)

DEVICE ADDRESS	FUNCTION CODE	REGISTER		DATA/VALUE		CRC	
78	06	00	23	00	00	78	00
78	86	02		C3	A1		

Returned by device:

Example:

Send to device: Address 01, write (06) to register 12 (Hex C) value 300 (Hex 12C) –out of range (see table 8.1)

DEVICE ADDRESS	FUNCTION CODE	REGISTER		DATA/ VALUE		CRC	
01	06	00	0C	01	2C	01	2C

Returned by device:

DEVICE ADDRESS	FUNCTION CODE	ERROR RESPONSE	CRC	
01	86	03	02	61

Note: When device returns an error message, it add 80 Hex to the Function Code (03 + 80 = 83 or 06 + 80 = 86)

APPENDIX A :**ASCII Chart**

ASCII Char	Dec	Hex	Binary No Parity	ASCII Char	Dec	Hex	Binary No parity
NUL	00	00	00000000	@	64	40	01000000
SOH	01	01	00000001	A	65	41	01000000
STX	02	02	00000010	B	66	42	01000010
ETX	03	03	00000011	C	67	43	01000011
EOT	04	04	00000100	D	68	44	01000100
ENQ	05	05	00000101	E	69	45	01000101
ACK	06	06	00000110	F	70	46	01000110
BEL	07	07	00000111	G	71	47	01000111
BS	08	08	00001000	H	72	48	01001000
HT	09	09	00001001	I	73	49	01001001
LF	10	0A	00001010	J	74	4A	01001010
VT	11	0B	00001011	K	75	4B	01001011
FF	12	0C	00001100	L	76	4C	01001100
CR	13	0D	00001101	M	77	4D	01001101
SO	14	0E	00001110	N	78	4E	01001110
SI	15	0F	00001111	O	79	4F	01001111
DLE	16	10	00010000	P	80	50	01010000
DC1	17	11	00010001	Q	81	51	01010001
DC2	18	12	00010010	R	82	52	01010010
DC3	19	13	00010011	S	83	53	01010011
DC4	20	14	00010100	T	84	54	01010100
NAK	21	15	00010101	U	85	55	01010101
SYN	22	16	00010110	V	86	56	01010110
ETB	23	17	00010111	W	87	57	01010111
CAN	24	18	00011000	X	88	58	01011000
EM	25	19	00011001	Y	89	59	01011001
SUB	26	1A	00011010	Z	90	5A	01011010
ESC	27	1B	00011011	[91	5B	01011011
FS	28	1C	00011100	\	92	5C	01011100
GS	29	1D	00011101]	93	5D	01011101
RS	30	1E	00011110	^	94	5E	01011110

US	31	1F	00011111	_	95	5F	01011111
SP	32	20	00100000	`	96	60	01100000
!	33	21	00100001	a	97	61	01100001
“	34	22	00100010	b	98	62	01100010
#	35	23	00100011	c	99	63	01100011
\$	36	24	00100100	d	100	64	01100100
%	37	25	00100101	e	101	65	01100101
&	38	26	00100110	f	102	66	01100110
‘	39	27	00100111	g	103	67	01100111
(40	28	00101000	h	104	68	01101000
)	41	29	00101001	i	105	69	01101001
*	42	2A	00101010	j	106	6A	01101010
+	43	2B	00101011	k	107	6B	01101011
,	44	2C	00101100	l	108	6C	01101100
-	45	2D	00101101	m	109	6D	01101101
.	46	2E	00101110	n	110	6E	01101110
/	47	2F	00101111	o	111	6F	01101111
0	48	30	00110000	p	112	70	01110000
1	49	31	00110001	q	113	71	01110001
2	50	32	00110010	r	114	72	01110010
3	51	33	00110011	s	115	73	01110011
4	52	34	00110100	t	116	74	01110100
5	53	35	00110101	u	117	75	01110101
6	54	36	00110110	v	118	76	01110110
7	55	37	00110111	w	119	77	01110111
8	56	38	00111000	x	120	78	01111000
9	57	39	00111001	y	121	79	01111001
:	58	3A	00111010	z	122	7A	01111010
;	59	3B	00111011	{	123	7B	01111011
<	60	3C	00111100		124	7C	01111100
=	61	3D	00111101	}	125	7D	01111101
>	62	3E	00111110	~	126	7E	01111110
?	63	3F	00111111	DEL	127	7F	01111111

ASCII Control Codes

ASCII Char	Dec	Hex	Ctrl Key Equiv.	Definition	ASCII Char	Dec	Hex	Ctrl Key Equiv.	Definition
NUL	00	00	Ctrl @	Null Character	DC1	17	11	Ctrl Q	Data Control 1 - XON
SOH	01	01	Ctrl A	Start of Header	DC2	18	12	Ctrl R	Data Control 2
STX	02	02	Ctrl B	Start of Text	DC3	19	13	Ctrl S	Data Control 3 - XOFF
ETX	03	03	Ctrl C	End of Text	DC4	20	14	Ctrl T	Data Control 4
EOT	04	04	Ctrl D	End of Transmission	NAK	21	15	Ctrl U	Negative Acknowledge
ENQ	05	05	Ctrl E	Inquiry	SYN	22	16	Ctrl V	Synchronous Idle
ACK	06	06	Ctrl F	Acknowledge	ETB	23	17	Ctrl W	End of Trans Block
BEL	07	07	Ctrl G	Bell	CAN	24	18	Ctrl X	Cancel
BS	08	08	Ctrl H	Back Space	EM	25	19	Ctrl Y	End of Medium
HT	09	09	Ctrl I	Horizontal Tabulation	SUB	26	1A	Ctrl Z	Substitute
LF	10	0A	Ctrl J	Line Feed	ESC	27	1B	Ctrl [Escape
VT	11	0B	Ctrl K	Vertical Tabulation	FS	28	1C	Ctrl \	File Separator
FF	12	0C	Ctrl L	Form Feed	GS	29	1D	Ctrl]	Group Separator
CR	13	0D	Ctrl M	Carriage Return	RS	30	1E	Ctrl	Record Separator
SO	14	0E	Ctrl N	Shift Out	US	31	1F	Ctrl _	Unit Separator
SI	15	0F	Ctrl O	Shift In	SP	32	20		Space
DLE	16	10	Ctrl P	Data Link Escape					

APPENDIX B : Example of CRC calculation

Device address 06, read (03), starting register 0008, number of registers 0001

Table 6.1 CRC Calculation

Function code	Two byte (16 bit) Register				Overflow Bit
	MSB		LSB		
Load 16 bit register to all 1's	1111	1111	1111	1111	0
First byte is address 06			0000	0110	
Exclusive OR	1111	1111	1111	1001	
1 st shift	0111	1111	1111	1100	1
A001	1010	0000	0000	0001	
Exclusive OR	1101	1111	1111	1101	
2 nd shift	0110	1111	1111	1110	1
A001	1010	0000	0000	0001	
Exclusive OR	1100	1111	1111	1111	
3 rd shift	0110	0111	1111	1111	1
A001	1010	0000	0000	0001	
Exclusive OR	1100	0111	1111	1110	
4 th shift	0110	0011	1111	1111	0
5 th shift	0011	0001	1111	1111	1
A001	1010	0000	0000	0001	
Exclusive OR	1001	0001	1111	1110	
6 th shift	0100	1000	1111	1111	0
7 th shift	0010	0100	0111	1111	1
A001	1010	0000	0000	0001	
Exclusive OR	1000	0100	0111	1110	
8 th shift	0100	0010	0011	1111	0
Second byte Read 03			0000	0011	
Exclusive OR	0100	0010	0011	1100	
1 st shift	0010	0001	0001	1110	0
2 nd shift	0001	0000	1000	1111	0
3 rd shift	0000	1000	0100	0111	1
A001	1010	0000	0000	0001	
Exclusive OR	1010	1000	0100	0110	
4 th shift	0101	0100	0010	0011	0
5 th shift	0010	1010	0001	0001	1
A001	1010	0000	0000	0001	
Exclusive OR	1000	1010	0001	0000	
6 th shift	0100	0101	0000	1000	0
7 th shift	0010	0010	1000	0100	0
8 th shift	0001	0001	0100	0010	0
Third byte Starting reg. 00			0000	0000	
Exclusive OR	0001	0001	0100	0010	
1 st shift	0000	1000	1010	0001	0
2 nd shift	0000	0100	0101	0000	1
A001	1010	0000	0000	0001	
Exclusive OR	1010	0100	0101	0001	
3 rd shift	0101	0010	0010	1000	1

A001	1010	0000	0000	0001	
Exclusive OR	1111	0010	0010	1001	
4 th shift	0111	1001	0001	0100	1
A001	1010	0000	0000	0001	
Exclusive OR	1101	1001	0001	0101	
5 th shift	0110	1100	1000	1010	1
A001	1010	0000	0000	0001	
Exclusive OR	1100	1100	1000	1011	
6 th shift	0110	0110	0100	0101	1
A001	1010	0000	0000	0001	
Exclusive OR	1100	0110	0100	0100	
7 th shift	0110	0011	0010	0010	0
8 th shift	0011	0001	1001	0001	0
Fourth Byte 08			0000	1000	
Exclusive OR	0011	0001	1001	1001	
1 st shift	0001	1000	1100	1100	1
A001	1010	0000	0000	0001	
Exclusive OR	1011	1000	1100	1101	
2 nd shift	0101	1100	0110	0110	1
A001	1010	0000	0000	0001	
Exclusive OR	1111	1100	0110	0111	
3 rd shift	0111	1110	0011	0011	1
A001	1010	0000	0000	0001	
Exclusive OR	1101	1110	0011	0010	
4 th shift	0110	1111	0001	1001	0
5 th shift	0011	0111	1000	1100	1
A001	1010	0000	0000	0001	
Exclusive OR	1001	0111	1000	1101	
6 th shift	0100	1011	1100	0110	1
A001	1010	0000	0000	0001	
Exclusive OR	1110	1011	1100	0111	
7 th shift	0111	0101	1110	0011	1
A001	1010	0000	0000	0001	
Exclusive OR	1101	0101	1110	0010	
8 th shift	0110	1010	1111	0001	0
Fifth Byte 00			0000	0000	
Exclusive OR	0110	1010	1111	0001	
1 st shift	0011	0101	0111	1000	1
A001	1010	0000	0000	0001	
Exclusive OR	1001	0101	0111	1001	
2 nd shift	0100	1010	1011	1100	1
A001	1010	0000	0000	0001	
Exclusive OR	1110	1010	1011	1101	
3 rd shift	0111	0101	0101	1110	1
A001	1010	0000	0000	0001	
Exclusive OR	1101	0101	0101	1111	
4 th shift	0110	1010	1010	1111	1
A001	1010	0000	0000	0001	

Exclusive OR	1100	1010	1010	1110	
5 th shift	0110	0101	0101	0111	0
6 th shift	0011	0010	1010	1011	1
A001	1010	0000	0000	0001	
Exclusive OR	1001	0010	1010	1010	
7 th shift	0100	1001	0101	0101	0
8 th shift	0010	0100	1010	1010	1
A001	1010	0000	0000	0001	
Exclusive OR	1000	0100	1010	1011	
Sixth Byte 01			0000	0001	
Exclusive OR	1000	0100	1010	1010	
1 st shift	0100	0010	0101	0101	0
2 nd shift	0010	0001	0010	1010	1
A001	1010	0000	0000	0001	
Exclusive OR	1000	0001	0010	1011	
3 rd shift	0100	0000	1001	0101	1
A001	1010	0000	0000	0001	
Exclusive OR	1110	0000	1001	0100	
4 th shift	0111	0000	0100	1010	0
5 th shift	0011	1000	0010	0101	0
6 th shift	0001	1100	0001	0010	1
A001	1010	0000	0000	0001	
Exclusive OR	1011	1100	0001	0011	
7 th shift	0101	1110	0000	1001	1
A001	1010	0000	0000	0001	
Exclusive OR	1111	1110	0000	1000	
8 th shift	0111	1111	0000	0100	0
CRC error	7	F	0	4	

Transmitted Message:

DEVICE ADDRESS	FUNCTION CODE	STARTING REGISTER		NUMBER OF REGISTERS		CRC	
06	03	00	08	00	01	04	7F

Example of CRC calculation in “C” language

This subroutine used to do CRC calculation

```
#define POLY 0xA001;

unsigned int crc_calculation (unsigned char *start_string, unsigned char number_byte)
{
    unsigned int crc;
    unsigned char bit_counter;
    unsigned char *data_pointer;

    data_pointer= start_string;
    crc = 0xffff;           // Initialize crc

    while (number_byte>0)
    {
        crc ^= *data_pointer // crc XOR with data
        bit_counter=0;      // reset counter

        while (bit_counter < 8)
        {
            if (crc & 0x0001)
            {
                {
                    crc >>= 1; // shift to the right 1 position
                    crc ^= POLY; // crc XOR with POLY
                }
            }

            else
            {
                {
                    crc >>=1; // shift to the right 1 position
                }
            }

            bit_counter++; // increase counter
        }

        data_pointer++; // adjust byte counter
    }
    return (crc); // final result of crc
}
```