Wireless Sensor Networks

Data Communications Protocol

MicroStrain, Inc.
459 Hurricane Lane, Suite 102
Williston, VT 05495
Phone 802-862-6629
Fax 802-863-4093
www.microstrain.com
support@microstrain.com

ISSUED: 20 January 2012

Information subject to change

Information in this document is subject to change without notice and does not represent a commitment on the part of MicroStrain, Inc. While MicroStrain, Inc. makes every effort as to the accurateness of this document, it assumes no responsibility for errors or omissions.

Trademarks

MicroStrain®, WSDA®, Node Commander®, V-Link®, SG-Link®, G-Link®, TC-Link®, mXRS™, SensorCloud™, HS-Link®, DVRT-Link™, Strain Wizard® and EH-Link® are trademarks of MicroStrain, Inc.

SDK

The Software Development Kit (SDK) is provided "as is" and any expressed or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall MicroStrain or its contributors be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this SDK, even if advised of the possibility of such damage. MicroStrain will make every effort to amplify the instructions contained in the Data Communications Protocol manual and sample code but will neither undertake to detail the functioning of the hardware or firmware in the Wireless Sensor Networks family nor debug the purchaser's code.

TABLE OF CONTENTS

OVERVIEW

This document describes the data communications protocol for the MicroStrain Wireless Sensor Networks mXRS™ wireless nodes and WSDA® base stations product line which includes the following devices:

Device	Firmware Version
V-Link® -mXRS™ Wireless Voltage Node	7.31 and higher
G-Link® -mXRS™ Wireless Accelerometer Node	7.29 and higher
SG-Link® -mXRS™ Wireless Strain Node	7.29 and higher
DVRT-Link™ -mXRS™ Wireless Displacement Node	7.29 and higher
SG-Link® -OEM-S Wireless Strain Node	7.29 and higher
TC-Link® -mXRS™ -6CH Wireless Thermocouple Node	7.29 and higher
TC-Link® -mXRS™ -1CH Wireless Thermocouple Node	7.29 and higher
TC-Link® -mXRS™ OEM Wireless Thermocouple Node	1.15 and higher
WSDA® -Base-101 Analog Output Base Station	2.37 and higher
WSDA® -Base-102 RS-232 Serial Output Base Station	2.37 and higher
WSDA® -Base-104 USB Base Station	2.37 and higher
WSDA® -1000 Gateway	2.37 and higher

If your equipment has firmware earlier than stated in the above table, there may be a requirement to upgrade in order to take full advantage of this protocol.

USB Interface

When communication between the WSDA®-Base and the host computer is via a USB 2.0 compliant connection using a Silicon Laboratories CP210x USB-to-UART Bridge chip on the base station, the connection is supported by a Silicon Laboratories Virtual Communications Port (VCP) driver installed on the host computer. Please see the section entitled *CP210x USB to UART Bridge Controller* for more details.

USB Virtual Communication Port Configuration	
Baud Rate	921, 600
Parity	None
Data Bits	8
Stop Bits	1

RS-232 Interface

When communication between the WSDA®-Base and the host computer is via an RS-232 connection, the physical layer provides the interface and there is no driver required by the base station.

RS-232 Port Configuration	
Baud Rate	115,200 (default) or 921,600
Parity	None
Data Bits	8
Stop Bits	1

COMMANDS OVERVIEW

The following two tables present a quick reference list of communication commands for controlling the base station and wireless sensor nodes. Please see the sections following for more detailed information on each command.

Base Station Commands

Command	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Ping Base Station	0x01						
Read Base Station EEPROM	0x73	EEprom Address		Checksum			
		MSB	LSB	MSB	LSB		
Write Base Station EEPROM	0x78	EEprom Address		Value		Checksum	
		MSB	LSB	MSB	LSB	MSB	LSB
Enable Beacon	0xBE	0xAC	Timestamp				
			MSB			LSB	
Disable Beacon	0xBE	0xAC	0xFF	0xFF	0xFF	0xFF	

Node Commands

Command	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15	Byte 16	Byte 17	Byte 18
Short Ping	0x02	Node Address																
		MSB	LSB															
Long Ping	0xAA	0x05	0x00	Node Address		0x02	0x00	0x02	Checksum									
				MSB	LSB				MSB	LSB								
Read Node EEPROM	0xAA	0x05	0x00	Node Address		0x04	0x00	0x03	EEprom Address		Checksum							
				MSB	LSB				MSB	LSB	MSB	LSB						
Write Node EEPROM	0xAA	0x05	0x00	Node Address		0x06	0x00	0x04	EEprom Address		Write Value		Checksum					
				MSB	LSB				MSB	LSB	MSB	LSB	MSB	LSB				
Initiate Sleep Mode	0x32	Node Address																
		MSB	LSB															
Stop Node	0xAA	0xFE	0x00	Node Address		0x02	0x00	0x90	Checksum									
				MSB	LSB				MSB	LSB								
Arm Node for Datalogging	0xAA	0x05	0x00	Node Address		0x02 + Text Length	0x00	0x0D	Custom User Entered Text (up to 50 Bytes)								Checksum	
				MSB	LSB												MSB	LSB
Trigger Armed Datalogging	0xAA	0x05	0x00	Node Address		0x0A	0x00	0x0E	Timestamp								Checksum	
				MSB	LSB				MSB							LSB	MSB	LSB
Download Page	0x05	Node Address		Page Index														
		MSB	LSB	MSB	LSB													
Erase Session Data	0x06	Node Address		0x08	0x10	0x0C	0xFF											
		MSB	LSB															
Initiate Streaming	0x38	Node Address																
		MSB	LSB															
Initiate Low Duty Cycle	0xAA	0x05	0x00	Node Address		0x02	0x00	0x38	Checksum									
				MSB	LSB				MSB	LSB								
Initiate Synchronized Sampling	0xAA	0x05	0x00	Node Address		0x02	0x00	0x3A	Checksum									
				MSB	LSB				MSB	LSB								
Read Single Sensor	0x03	Node Address		0x01	Channel #													
		MSB	LSB															
Auto-Balance Channel	0x62	Node Address		Channel #	Target Balance													
		MSB	LSB		MSB	LSB												

Data Format (MSB, LSB)

All communication is performed at the byte level. To represent values greater than a single byte the value is broken down into several bytes, transmitted, received as several bytes and reassembled into a single value. Throughout this document the notation of MSB (Most Significant Byte) and LSB (Least Significant Byte) will be used to describe 2 byte values, as well as the start and end of 4 byte values.

2-Byte Value Example

A value of 476 would yield an MSB of 1 and an LSB of 220.

	Decimal	Hex	Binary
2 Byte Value	476	01 DC	00000001 1101100
MSB	1	01	00000001
LSB	220	DC	1101100

Sample C++ Code

```
typedef unsigned char BYTE;
typedef unsigned short WORD;

void ValueToMSBandLSB(WORD value, BYTE& msb, BYTE& lsb)
{
    msb = value >> 8;     //Shift 8 bits right, drop the lower byte.
    lsb = value & 0x00ff; //Mask out the upper byte.
}

WORD ValueFromMSBandLSB(BYTE msb, BYTE lsb)
{
    return (msb << 8)| lsb; //Shift msb 8 bits left, OR the lsb in.
}
```

4-Byte Value Example

A value of 1,326,214,446 would yield the following:

	Decimal	Hex	Binary
4 Byte Value	1,326,214,446	4F 0C 6D 2E	01001111 00001100 01101101 00101110
Byte 1 (MSB)	79	4F	01001111
Byte 2	12	0C	00001100
Byte 3	109	6D	01101101
Byte 4 (LSB)	46	2E	00101110

Sample C++ Code

```
typedef unsigned short WORD;
typedef unsigned long DWORD;

void ValueTo4Bytes(DWORD value, BYTE& b1, BYTE& b2, BYTE& b3, BYTE& b4)
{
    b1 = value >> 24;          //Shift 24 bits right, drop lower bytes
    b2 = (value >> 16) & 0xff; //Shift 16 bits right, drop lower bytes, Mask out the upper bytes

    b3 = (value >> 8) & 0xff;  //Shift 8 bits right, drop lower byte, Mask out the upper bytes
    b4 = value & 0xff;         //Mask out the upper bytes
}

DWORD ValueFrom4Bytes(BYTE b1, BYTE b2, BYTE b3, BYTE b4)
{
    WORD hiWord = (b1 << 8)| b2;    //Shift msb 8 bits left, OR the lsb in.
    WORD loWord = (b3 << 8)| b4;    //Shift msb 8 bits left, OR the lsb in.

    return (hiWord << 16) | loWord; //Shift hiWord 16 bits left, OR the loWord in.
}
```

Checksum

Some commands and responses require or supply a checksum. The checksum is used to ensure that the data was transmitted without error. The documentation for each command or response that uses a checksum will detail how the checksum is calculated. The checksum is calculated by summing all the bytes protected by the checksum and taking the modulo N of the sum, where N is 256 for a 1 byte checksum and 65536 for a 2 byte checksum.

$$N\ Byte\ Checksum = \left(\sum_{i=x}^{I} Byte[i]\right)\ mod\ (256^N), \qquad N \in [1,2]$$

Single Byte Checksum Example

Protected Command Bytes

Byte	Value
Byte 1	10
Byte 2	121
Byte 3	37
Byte 4	235

```
Sum:        403 = 10 + 121 + 37 + 235
Checksum:   148 = 403 MOD 256
```

Two Byte Checksum Example

Protected Command Bytes

Byte	Value
Byte 1	10
Byte 2	121
Byte 3	37
Byte 4	235

```
Sum:        403 = 10 + 121 + 37 + 235
Checksum:   403 = 403 MOD 65536
```

RSSI

Received Signal Strength Indicator (RSSI) is a measurement of the power present in a received radio signal. Node commands such as Long Ping and Node Discovery return a true RSSI value, measured in dBm, in their response packet. This signed 1 byte value can range from -95 dBm to +5 dBm. The example below shows how to convert the value received from a node to a signed integer.

```
typedef unsigned char BYTE;

int GetRssiFromByte(BYTE valueFromEeprom)
{
    int result;

    //if the byte value is < 128
    if(valueFromEeprom < 128)
    {
        result = valueFromEeprom;
    }
    //if the byte value is >= 128
    else
    {
        //take the (value mod 128) - 128
        result = (valueFromEeprom % 128) - 128;
    }

    //return the result as an integer
    return result;
}
```

Some command responses return two RSSI values, **Node RSSI** and **Base Station RSSI**. The **Node RSSI** is the signal strength that the node received the command from the base station. The **Base Station RSSI** is the signal strength that the base station received the response back from the node.

LQI

The Link Quality Indicator (LQI) field is reserved for future use, and should be ignored.

BASE STATION COMMANDS

Ping Base Station

Use the **Ping Base Station** command to ensure that the host computer and the Base Station are properly communicating.

Command Packet:

Byte 1	0x01

Success Response:

Byte 1	0x01

Fail Response: No Response

Read Base Station EEPROM

Use the **Read Base Station EEPROM** command to read the value of a specific memory address from the Base Station's EEPROM. See *Base Station EEPROM Map* for specific memory address details.

Command Packet:

Byte 1	0x73	
Byte 2	EEPROM address (MSB)	
Byte 3	EEPROM address (LSB)	
Byte 4	Checksum (MSB)	Checksum of Bytes: 2 - 3
Byte 5	Checksum (LSB)	

Success Response:

Byte 1	0x73	
Byte 2	Value Read (MSB)	
Byte 3	Value Read (LSB)	
Byte 4	Checksum (MSB)	Checksum of Bytes: 2 - 3
Byte 5	Checksum (LSB)	

Fail Response:

Byte 1	0x21

See also: Data Format (MSB, LSB), Checksum

Write Base Station EEPROM

Use the **Write Base Station EEPROM** command to write a value to a specific memory address on the Base Station's EEPROM. The **Read Base Station EEPROM** command may be used to further verify that the value was written correctly. See *Base Station EEPROM Map* for specific memory address details.

Command Packet:

Byte 1	0x78	
Byte 2	EEPROM address (MSB)	
Byte 3	EEPROM address (LSB)	
Byte 4	Value to Write (MSB)	
Byte 5	Value to Write (LSB)	
Byte 6	Checksum (MSB)	Checksum of Bytes: 2 - 5
Byte 7	Checksum (LSB)	

Success Response:

Byte 1	0x78	
Byte 2	Value Written (MSB)	
Byte 3	Value Written (LSB)	
Byte 4	Checksum (MSB)	Checksum of Bytes: 2 - 3
Byte 5	Checksum (LSB)	

Fail Response:

Byte 1	0x21

See also: Data Format (MSB, LSB), Checksum

Enable Beacon

Use the **Enable Beacon** command to turn on the beacon on the base station. The beacon is used to synchronize and start a group of nodes when performing **Synchronized Sampling**.

Command Packet:

Byte 1	0xBE
Byte 2	0xAC
Byte 3	Timestamp Byte 1 (MSB)
Byte 4	Timestamp Byte 2
Byte 5	Timestamp Byte 3
Byte 6	Timestamp Byte 4 (LSB)

Success Response:

Byte 1	0xBE
Byte 2	0xAC

Command Packet Timestamp Bytes

Bytes 3-6 in the Command Packet are Timestamp bytes. These bytes represent the current UTC time in seconds (an unsigned 64-bit integer) with byte 3 being the MSB of the timestamp, and byte 6 being the LSB of the timestamp.

One should use whatever built-in method is available to the system or programming language to find the current PC time in UTC in seconds and split that into the 4 bytes shown above.

See also: Synchronized Sampling

Disable Beacon

Use the **Disable Beacon** command to turn off the beacon on the base station. Sending the **Stop Node** command from a base station that is beaconing will automatically turn off the beacon as well.

Command Packet:

Byte 1	0xBE
Byte 2	0xAC
Byte 3	0xFF
Byte 4	0xFF
Byte 5	0xFF
Byte 6	0xFF

Success Response:

Byte 1	0xBE
Byte 2	0xAC

See also: Synchronized Sampling , Stop Node

NODE COMMANDS

Short Ping

Use the **Short Ping** command to check the communication between the Base Station and the Node.

Command Packet:

Byte 1	0x02
Byte 2	Node Address (MSB)
Byte 3	Node Address (LSB)

Success Response:

Byte 1	0x02

Fail Response:

Byte 1	0x21

See also: Data Format (MSB, LSB)

Long Ping

Use the **Long Ping** command to check the communication between the Base Station and the Node.

Command Packet:

Byte 1	SOP(Start Of Packet)	0xAA	
Byte 2	Message Type	0x05	
Byte 3	Address Mode	0x00	
Byte 4	Node Address (MSB)	0xXX	
Byte 5	Node Address (MSB)	0xXX	
Byte 6	Payload Length	0x02	
Byte 7	Command ID (MSB)	0x00	
Byte 8	Command ID (LSB)	0x02	
Byte 9	Checksum (MSB)	0xXX	Checksum of Bytes: 2 - 8
Byte 10	Checksum (LSB)	0xXX	

Response:

The 1 byte response comes directly from the Base Station to verify that the Long Ping command was correctly received and sent to the Node.

Byte 1	0xAA

Success Response:

Byte 1	SOP	0xAA	
Byte 2	Message Type	0x07	
Byte 3	Address Mode	0x02	
Byte 4	Node Address (MSB)	0xXX	
Byte 5	Node Address (LSB)	0xXX	
Byte 6	Payload Length	0x02	
Byte 7	Payload (not used)	0x00	
Byte 8	Payload (not used)	0x00	
Byte 9	Node RSSI	0xXX	
Byte 10	Base Station RSSI	0xXX	
Byte 11	Checksum (MSB)	0xXX	Checksum of Bytes: 2 - 8
Byte 12	Checksum (LSB)	0xXX	

Fail Response: No Response

See also: Data Format (MSB, LSB), Checksum

Read Node EEPROM

Use the **Read Node EEPROM** command to read the value of a specific memory address from the Node's EEPROM. See *Node EEPROM Map* for specific memory address details.

Command Packet:

Byte 1	SOP (Start of Packet)	0xAA	
Byte 2	Message Type	0x05	
Byte 3	Address Mode	0x00	
Byte 4	Node Address (MSB)	0xXX	
Byte 5	Node Address (LSB)	0xXX	
Byte 6	Payload Length	0x04	
Byte 7	Command ID (MSB)	0x00	
Byte 8	Command ID (LSB)	0x03	
Byte 9	EEPROM Address (MSB)	0xXX	
Byte 10	EEPROM Address (LSB)	0xXX	
Byte 11	Checksum (MSB)	0xXX	Checksum of Bytes: 2 - 10
Byte 12	Checksum (LSB)	0xXX	

Response:

The 1 byte response comes directly from the Base Station to verify that the command was correctly received and sent to the Node.

Byte 1	0xAA

Success Response:

Byte 1	SOP	0xAA	
Byte 2	Message Type	0x00	
Byte 3	Address Mode	0x00	
Byte 4	Node Address (MSB)	0xXX	
Byte 5	Node Address (LSB)	0xXX	
Byte 6	Payload Length	0x02	
Byte 7	Value (MSB)	0xXX	
Byte 8	Value (LSB)	0xXX	
Byte 9	LQI	0xXX	
Byte 10	Base Station RSSI	0xXX	
Byte 11	Checksum (MSB)	0xXX	Checksum of Bytes: 2 - 8
Byte 12	Checksum (LSB)	0xXX	

Fail Response: No Response

See also: Data Format (MSB, LSB), Checksum

Write Node EEPROM

Use the **Write Node EEPROM** command to write a value to a specific memory address on the Node's EEPROM. See *Node EEPROM Map* for specific memory address details.

Command Packet:

Byte 1	SOP (Start of Packet)	0xAA	
Byte 2	Message Type	0x05	
Byte 3	Address Mode	0x00	
Byte 4	Node Address (MSB)	0xXX	
Byte 5	Node Address (LSB)	0xXX	
Byte 6	Payload Length	0x06	
Byte 7	Command ID (MSB)	0x00	
Byte 8	Command ID (LSB)	0x04	
Byte 9	EEPROM Address (MSB)	0xXX	
Byte 10	EEPROM Address (LSB)	0xXX	
Byte 11	Value (MSB)	0xXX	
Byte 12	Value (LSB)	0xXX	
Byte 13	Checksum (MSB)	0xXX	Checksum of Bytes: 2 - 12
Byte 14	Checksum (LSB)	0xXX	

Response:

The 1 byte response comes directly from the Base Station to verify that the command was correctly received and sent to the Node.

Byte 1	0xAA

Success Response:

Byte 1	SOP	0xAA	
Byte 2	Message Type	0x00	
Byte 3	Address Mode	0x00	
Byte 4	Node Address (MSB)	0xXX	
Byte 5	Node Address (LSB)	0xXX	
Byte 6	Payload Length	0x02	
Byte 7	Write Command MSB	0x00	
Byte 8	Write Command LSB	0x04	
Byte 9	LQI	0xXX	
Byte 10	Base Station RSSI	0xXX	
Byte 11	Checksum (MSB)	0xXX	Checksum of Bytes: 2 - 8
Byte 12	Checksum (LSB)	0xXX	

Fail Response: No Response

See also: Data Format (MSB, LSB), Checksum

Initiate Sleep Mode

Use the **Initiate Sleep Mode** command to put the Node in a low power state.

Command Packet:

Byte 1	0x32
Byte 1	Node Address (MSB)
Byte 2	Node Address (LSB)

Success Response: No Response
Fail Response: No response

Waking Node

A Node in sleep mode periodically awakes, listens for a command, and if none is received, returns to sleep. To wake a sleeping Node, send the **Stop Node** command.

Relevant Memory Locations

Node EEPROM 66: Sleep Interval

See also: Data Format (MSB, LSB), Short Ping, Stop Node

Stop Node

Use the **Stop Node** command to stop a Node that is either in **Streaming, LDC, Synchronized Sampling, Datalogging** or **Sleep** mode. It will also stop a Node in **High Speed Streaming** mode if streaming a single channel.

Command Packet:

Byte 1	SOP	0xAA	
Byte 2	Message Type	0xFE	
Byte 3	Address Mode	0x00	
Byte 4	Node Address (MSB)	0xXX	
Byte 5	Node Address (LSB)	0xXX	
Byte 6	Packet Length	0x02	
Byte 7	Command Byte (MSB)	0x00	
Byte 8	Command Byte (LSB)	0x90	
Byte 9	Checksum (MSB)	0xXX	Checksum of Bytes: 2 - 8
Byte 10	Checksum (LSB)	0xXX	

Response:

The 1 byte response comes directly from the Base Station to verify that the **Stop Node** command was correctly received and the Base Station is attempting to stop the Node.

Byte 1	0xAA

Success Response:

The Node was stopped and will now communicate.

Byte 1	0x90
Byte 2	0x01

Fail Response:

The **Stop Node** process was aborted by user sending any random byte to the Base Station.

Byte 1	0x21
Byte 2	0x01

The **Stop Node** command sends small packets as fast as possible in an attempt to communicate with the Node, and periodically checks, after 1000 packets, to see if the Node has responded to a ping request. The function will continue indefinitely until either the Node responds, or the user sends any random byte to the Base Station. The base station will send a 0x01 response when it has ceased sending the **Stop Node** packets, for both success and fail.

Broadcast Special Case

When the broadcast Node address 65535 is used, the Base Station does not check for a ping response. It will continue sending the stop Node command until interrupted by the user (any single byte sent to the Base Station).

See also: Data Format (MSB, LSB), Checksum

Arm Node (for Datalogging)

Use the **Arm Node** command to put the node in an armed state waiting for the **Trigger Armed Datalogging** command. A Node will stay in this armed state for a default of 10 seconds (EEPROM adjustable). To disarm an Armed Node, use the **Stop Node** command.

Command Packet:

Byte 1	SOP	0xAA	
Byte 2	Delivery Action Byte	0x05	
Byte 3	App Data Type	0x00	
Byte 4	Node Address (MSB)	0xXX	
Byte 5	Node Address (LSB)	0xXX	
Byte 6	Payload Length	0x02 + # of user bytes	
Byte 7	Arm Command (MSB)	0x00	
Byte 8	Arm Command (LSB)	0x0D	
Byte 9	User entered data (if applicable)	0xXX	
Bytes 10 through N-2	User entered data (if applicable)	0xXX	
Byte N-1	Checksum (MSB)	0xXX	Checksum of Bytes: 2 – (N-2)
Byte N	Checksum (LSB)	0xXX	

Response:

A 1-byte response comes directly from the Base Station to verify that the command was received.

Byte 1	0xAA

Success Response:

Byte 1	SOP	0xAA	
Byte 2	Delivery Action Byte	0x07	
Byte 3	App Data Type	0x00	
Byte 4	Node Address (MSB)	0xXX	
Byte 5	Node Address (LSB)	0xXX	
Byte 6	Payload Length	0x03	
Byte 7	Arm Command Received	0x00	
Byte 8	Arm Command Received	0x0D	
Byte 9	Fail Type	0x00	
Byte 10	LQI	0xXX	
Byte 11	Base Station RSSI	0xXX	
Byte 12	Checksum (MSB)	0xXX	Checksum of Bytes: 2 - 9
Byte 13	Checksum (LSB)	0xXX	

Fail Response:

No response is received or Byte 9 (Fail Type) is not 0x00.

User Entered Data

The user may send up to 50 bytes with the Arm Command which can then be downloaded from the node using the **Download Page** command.

See also: Data Format (MSB, LSB), Checksum, Trigger Armed Datalogging, Download Page

Trigger Armed Datalogging

Use the **Trigger Armed Datalogging** command to initiate a data capture session on-board the Node. The data will be stored in the Node's 2MB memory and may be downloaded at a later time. The **Trigger Armed Datalogging** command can be sent to the broadcast node address of 65535. This will be sent to all nodes on the operating frequency, but only nodes that are in the "Armed" state will start datalogging. This can be useful to start datalogging on multiple nodes at the same time.

Command Packet:

Byte 1	SOP	0xAA	
Byte 2	Delivery Action Byte	0x05	
Byte 3	App Data Type	0x00	
Byte 4	Node Address (MSB)	0xXX	
Byte 5	Node Address (LSB)	0xXX	
Byte 6	Payload Length	0x0A	
Byte 7	Trigger Command (MSB)	0x00	
Byte 8	Trigger Command (LSB)	0x0E	
Byte 9	UTC Timestamp, seconds (MSB)	0xXX	
Byte 10	UTC Timestamp, seconds	0xXX	
Byte 11	UTC Timestamp, seconds	0xXX	
Byte 12	UTC Timestamp, seconds	0xXX	
Byte 13	UTC Timestamp, nanoseconds	0xXX	
Byte 14	UTC Timestamp, nanoseconds	0xXX	
Byte 15	UTC Timestamp, nanoseconds	0xXX	
Byte 16	UTC Timestamp, nanoseconds (LSB)	0xXX	
Byte 17	Checksum (MSB)	0xXX	Checksum of Bytes: 2 - 16
Byte 18	Checksum (LSB)	0xXX	

Success Response: No Response
Fail Response: No Response

Timestamp

Bytes 9-16 make up the current UTC timestamp. Bytes 9-12 are the "seconds part" of the timestamp, and Bytes 13-16 are the "nanoseconds part" of the timestamp. When downloading this data from the node, this timestamp will be transmitted in the header information and can be used to find the exact timestamp of each data point.

One should use whatever built-in method is available to the system or programming language to find the current PC time in UTC, both in seconds and nanoseconds resolution, and split that into the 4 bytes shown above.

Notes

- If a Sensor Event Driven Trigger (SEDT) is being used, the equivalent of a trigger command is automatically issued within the Node's firmware as a result of an output ceiling, or output floor being reached. Note that if SEDT is enabled, the Node will *not* automatically enter Sleep mode after a user inactivity timeout.
- During a datalogging session the Node will not respond to any commands, except the **Stop Node** command.
- If continuous datalogging is enabled, the Node will continue datalogging until it uses all available memory, it receives a **Stop Node** command, or it is powered off.

Relevant Memory Locations

Node EEPROM 12: Active Channel Mask
Node EEPROM 14: Datalogging Sample Rate
Node EEPROM 16: Samples per Data Set
Node EEPROM 102: Continuous Datalogging Flag

See also: Data Format (MSB, LSB) , Checksum, Download Page, Erase Session Data, Arm Node

Download Page

When the Node is in datalogging mode, it stores the data to memory. Use the **Download Page** command to retrieve a logged data session from the Node.

Command Packet:

Byte 1	Command Byte	0x05
Byte 2	Node Address (MSB)	0xXX
Byte 3	Node Address (LSB)	0xXX
Byte 4	Page Index (MSB)	0xXX
Byte 5	Page Index (LSB)	0xXX

Success Response:

Byte 1	Command Byte	0x05	
Byte 2	Data Point 1 (MSB)	0xXX	
Byte 3	Data Point 1 (LSB)	0xXX	
Bytes 4-263	Data Points 2-131 (MSB:LSB)	0xXX	
Byte 264	Data Point 132 (MSB)	0xXX	
Byte 265	Data Point 132 (LSB)	0xXX	
Byte 266	Checksum (MSB)	0xXX	Checksum of Bytes: 2 - 265
Byte 267	Checksum (LSB)	0xXX	

Fail Response:

Byte 1	0x05
Byte 2	0x21

Page Index

Each Node contains 2MB of memory. These 2MB are mapped to 8192 pages of data, each page containing 132 data points. A data point is a 2 byte value.

Pages are numbered sequentially from 0 to 8191.

- Page 0 always contains data points that represent the current value in each of the Node's EEPROM locations (0-254).
- Page 1 is a 'blank' page and contains no usable information or data.
- Page 2 is the first page that will contain datalogging data.

Data Sessions

The Node can contain multiple consecutive datalogging sessions. Each of these sessions has a leading multi-byte header which is used to identify the start of a new datalogging session and the end of the previous session. The header contains the datalogging environment during the particular session.

Session Header

The datalogging session header is a multiple-byte leading string indicating the start of the next session. The header can be found anywhere on a downloaded page and it can wrap between pages. The header can be in one of many formats:

Header Format Version 1.0

Header Format Version 1.0 has the following format:

Data Point	Byte	Byte Value	Data Point Value	Description
1	1	255	65535	Fixed Header
	2	255		
2	3	253	64768-64772	Identifier
	4	0-4		Trigger ID
3	5	1	16	Header Version (Major)
	6	0		Header Version (Minor)
4	7	xxx	10 + # user bytes	Total # of bytes until channel data
	8	xxx		
5	9	xxx	100-65500	Samples per Data Set
	10	xxx		
6	11	xxx	1-65535	Session Index
	12	xxx		
7	13	0	1-255	Active Channel Mask
	14	1-255		
8	15	0	1-7	Datalogging Sampling Rate
	16	1-7		
9	17	0	0-50	# of user entered bytes
	18	xxx		
10 through M	19	xxx	xxxxx	Up to 50 user entered bytes + 1 buffer byte if odd number (if any)
	2M	xxx		
M + 1	2M + 1	0	10	Bytes Per Channel
	2M + 2	10		
M + 2	2M + 3	xxx	xxxxx	Channel 1 Action ID
	2M + 4	xxx		
M + 3	2M + 5	xxx	xxxxx	Channel 1 Action Slope (byte 1)
	2M + 6	xxx		Channel 1 Action Slope (byte 2)
M + 4	2M + 7	xxx	xxxxx	Channel 1 Action Slope (byte 3)
	2M + 8	xxx		Channel 1 Action Slope (byte 4)
M + 5	2M + 9	xxx	xxxxx	Channel 1 Action Offset (byte 1)
	2M + 10	xxx		Channel 1 Action Offset (byte 2)
M + 6	2M + 11	xxx	xxxxx	Channel 1 Action Offset (byte 3)
	2M + 12	xxx		Channel 1 Action Offset (byte 4)
M + 7 through N	2M + 13	xxx	xxxxx	Repeat Channel Action information for channels that are active (see channel mask)
	N			
N + 1	N + 1	xxx	xxxxx	# of bytes until end of header
	N + 2	xxx		
N + 2	N + 3	xxx	xxxxx	UTC Time, seconds (byte 1)
	N + 4	xxx		UTC Time, seconds (byte 2)
N + 3	N + 5	xxx	xxxxx	UTC Time, seconds (byte 3)
	N + 6	xxx		UTC Time, seconds (byte 4)
N + 4	N + 7	xxx	xxxxx	UTC Time, nanoseconds (byte 1)
	N + 8	xxx		UTC Time, nanoseconds (byte 2)
N + 5	N + 9	xxx	xxxxx	UTC Time, nanoseconds (byte 3)
	N + 10	xxx		UTC Time, nanoseconds (byte 4)

Header Format Version 2.0

Header Format Version 2.0 has the following format:

Data Point	Byte	Byte Value	Data Point Value	Description
1	1	255	65535	Fixed Header
	2	255		
2	3	253	64768-64772	Identifier
	4	0-4		Trigger ID
3	5	2	32	Header Version (Major)
	6	0		Header Version (Minor)
4	7	xxx	12 + # user bytes	Total # of bytes until channel data
	8	xxx		
5	9	xxx	100-65500	Samples per Data Set
	10	xxx		
6	11	xxx	1-65535	Session Index
	12	xxx		
7	13	0	1-255	Active Channel Mask
	14	1-255		
8	15	0	1-7	Datalogging Sampling Rate
	16	1-7		
9	17	1-3	xxxxx	Data Type
	18	xxx		Unused Byte
10	19	0	0-50	# of user entered bytes
	20	xxx		
11 through M	21	xxx	xxxxx	Up to 50 user entered bytes + 1 buffer byte if odd number (if any)
	2M	xxx		
M + 1	2M + 1	0	10	Bytes Per Channel
	2M + 2	10		
M + 2	2M + 3	xxx	xxxxx	Channel 1 Action ID
	2M + 4	xxx		
M + 3	2M + 5	xxx	xxxxx	Channel 1 Action Slope (byte 1)
	2M + 6	xxx		Channel 1 Action Slope (byte 2)
M + 4	2M + 7	xxx	xxxxx	Channel 1 Action Slope (byte 3)
	2M + 8	xxx		Channel 1 Action Slope (byte 4)
M + 5	2M + 9	xxx	xxxxx	Channel 1 Action Offset (byte 1)
	2M + 10	xxx		Channel 1 Action Offset (byte 2)
M + 6	2M + 11	xxx	xxxxx	Channel 1 Action Offset (byte 3)
	2M + 12	xxx		Channel 1 Action Offset (byte 4)
M + 7 through N	2M + 13	xxx	xxxxx	Repeat Channel Action information for channels that are active (see channel mask)
	N			
N + 1	N + 1	0	8	# of bytes until end of header
	N + 2	8		
N + 2	N + 3	xxx	xxxxx	UTC Time, seconds (byte 1)
	N + 4	xxx		UTC Time, seconds (byte 2)
N + 3	N + 5	xxx	xxxxx	UTC Time, seconds (byte 3)
	N + 6	xxx		UTC Time, seconds (byte 4)
N + 4	N + 7	xxx	xxxxx	UTC Time, nanoseconds (byte 1)
	N + 8	xxx		UTC Time, nanoseconds (byte 2)
N + 5	N + 9	xxx	xxxxx	UTC Time, nanoseconds (byte 3)
	N + 10	xxx		UTC Time, nanoseconds (byte 4)

Header Format Version 2.1

Header Format Version 2.1 has the following format:

Data Point	Byte	Byte Value	Data Point Value	Description
1	1	255	65535	Fixed Header
	2	255		
2	3	253	64768-64772	Identifier
	4	0-4		Trigger ID
3	5	2	32	Header Version (Major)
	6	0		Header Version (Minor)
4	7	xxx	12 + # user bytes	Total # of bytes until channel data
	8	xxx		
5	9	xxx	1-65535	Samples per Data Set ÷ 100
	10	xxx		
6	11	xxx	1-65535	Session Index
	12	xxx		
7	13	0	1-255	Active Channel Mask
	14	1-255		
8	15	0	1-7	Datalogging Sampling Rate
	16	1-7		
9	17	1-3	xxxxx	Data Type
	18	xxx		Unused Byte
10	19	0	0-50	# of user entered bytes
	20	xxx		
11 through M	21	xxx	xxxxx	Up to 50 user entered bytes + 1 buffer byte if odd number (if any)
	2M	xxx		
M + 1	2M + 1	0	10	Bytes Per Channel
	2M + 2	10		
M + 2	2M + 3	xxx	xxxxx	Channel 1 Action ID
	2M + 4	xxx		
M + 3	2M + 5	xxx	xxxxx	Channel 1 Action Slope (byte 1)
	2M + 6	xxx		Channel 1 Action Slope (byte 2)
M + 4	2M + 7	xxx	xxxxx	Channel 1 Action Slope (byte 3)
	2M + 8	xxx		Channel 1 Action Slope (byte 4)
M + 5	2M + 9	xxx	xxxxx	Channel 1 Action Offset (byte 1)
	2M + 10	xxx		Channel 1 Action Offset (byte 2)
M + 6	2M + 11	xxx	xxxxx	Channel 1 Action Offset (byte 3)
	2M + 12	xxx		Channel 1 Action Offset (byte 4)
M + 7 through N	2M + 13	xxx	xxxxx	Repeat Channel Action information for channels that are active (see channel mask)
	N			
N + 1	N + 1	0	8	# of bytes until end of header
	N + 2	8		
N + 2	N + 3	xxx	xxxxx	UTC Time, seconds (byte 1)
	N + 4	xxx		UTC Time, seconds (byte 2)
N + 3	N + 5	xxx	xxxxx	UTC Time, seconds (byte 3)
	N + 6	xxx		UTC Time, seconds (byte 4)
N + 4	N + 7	xxx	xxxxx	UTC Time, nanoseconds (byte 1)
	N + 8	xxx		UTC Time, nanoseconds (byte 2)
N + 5	N + 9	xxx	xxxxx	UTC Time, nanoseconds (byte 3)
	N + 10	xxx		UTC Time, nanoseconds (byte 4)

Fixed Header
Each new session is always marked with a 0xFFFF (decimal 65535) at its start.

Header Version
The header format that needs to be used can be determined by checking the header version bytes.

Trigger ID
Each datalogging session has an ID signifying how the session was started:

Trigger ID	Description
0	Started via a datalogging command
1	Ceiling SEDT
2	Floor SEDT
3	Ramp Up SEDT
4	Ramp Down SEDT

Samples per Data Set
The number of expected samples per data set. If the session was logged as a continuous datalogging session, this value is not applicable. If the session ended prematurely due to power failure or because the memory was completely filled, the value will not correspond to the actual number of samples. When parsing session data, the Samples per Data Set value should not be used to determine the end of the session. The end of the session should be determined by the start of the next session header. Header Versions 2.1 and above have the number of Samples per Data Set byte divided by 100 to allow for longer sessions. This value should be multiplied by 100 to obtain the true number of samples.

Session Index
Each data logging session is stored with an index. The first session's index is 1 and subsequent sessions would have consecutive index numbers. When the logged sessions are erased the index will be reset, and the first data logging session after the erase will have a Session Index of 1.

Active Channel Mask
The channel mask indicates the active channels that were logged in this session. Please see the section entitled *Channel Mask* for more details.

Sampling Rate
The Sample Rate that was used during the datalogging session. See the Sample Rates section for a table of the values and their corresponding sampling rates.

Data Type
Data type is the format which the node saves data to memory as. The following table shows the possible data values and their corresponding data types.

Value	Type
1, 3	bits
2	float

User Entered Bytes

Up to 50 optional user entered bytes may have been sent to the node when triggering an **Armed** Datalogging session. The "# of user entered bytes" byte specifies how many user bytes follow in the packet. If the number of user entered bytes is an odd number, there will be 1 extra "buffer" byte appended to the user data that should be discarded.

Channel Action

Channel Action shows which channel actions were enabled during data-logging for the active channels. The Action ID is a 2 byte value, while the Slope and Offset are 4 byte float values. For more information, please see the **Calibration Coefficients** section.

Timestamp

The UTC timestamp represents the starting time of the logged session (the time applied to the very first data point). The nanoseconds should be appended to the seconds of the given timestamp. Incrementing the initial UTC timestamp by the sampling rate, one can determine the exact time of each stored data point.

Session Data

During the actual datalogging session, the data from each active channel on the Node is written consecutively to the memory pages. For example, a Node with 3 active channels (CH1, CH3, CH4) would write data as CH1, CH3, CH4, CH1, CH3, CH4 and so forth. This data comes off the Node in the same pattern during download.

Response Packet Checksum

This response packet contains a checksum of bytes 2 -265.

Relevant Memory Locations

Node EEPROM 0: Current Log Page
Node EEPROM 2: Current Page Offset
Node EEPROM 4: Data Sets Stored

See also: Data Format (MSB, LSB) , Checksum, Arm Node, Trigger Armed Datalogging, Erase Session Data, Calibration Coefficients, Sample Rates

Erase Session Data

Use the **Erase Session Data** command to erase all datalogging sessions and data stored on the Node's memory.

Command Packet:

Byte 1	Command Byte	0x06
Byte 2	Node Address (MSB)	0xXX
Byte 3	Node Address (LSB)	0xXX
Byte 4	Command Byte	0x08
Byte 5	Command Byte	0x10
Byte 6	Command Byte	0x0C
Byte 7	Command Byte	0xFF

Success Response:

Byte 1	0x06

Fail Response:

Byte 1	0x21

Response

The response is returned immediately when the erasing processes begins, and no acknowledgment is sent when the process completes. The process will take approximately 5 seconds, and the Node will not respond to any commands until the erasing is complete. Completion of the erase can be detected by repeated short pinging of the Node; when the erase is complete, the Node will come back on-line and respond successfully to the ping.

See also: Data Format (MSB, LSB) , Trigger Armed Datalogging, Short Ping

Initiate Real-Time Streaming

Use the **Initiate Real-Time Streaming** command to start a real-time streaming session on a Node. The Node will respond by immediately sending a stream of data packets as the sensors are read.

Command Packet:

Byte 1	Command Byte	0x38
Byte 2	Node Address (MSB)	0xXX
Byte 3	Node Address (LSB)	0xXX

Response:

Parsing of the streaming packets should begin with the first 0xFF byte. The 0xFF byte is the header of the first valid data packet shown below as Byte 1. Each successive 0xFF indicates the start of a new data packet and the end of the previous packet.

Response Packet:

Byte 1	0xFF
Byte 2	Channel 1 Value (MSB)
Byte 3	Channel 1 Value (LSB)
…	
Byte (*N**2)-2	Channel *N* Value (MSB)
Byte (*N**2)-1	Channel *N* Value (LSB)
Byte (*N**2)	Checksum Byte

(N is the total number of channels)

Terminating Streaming

Continuous or Finite Streaming may be stopped at any time by issuing any byte to the Base Station. This will cause the Base Station to stop streaming. It will not however cause the Node to stop streaming. The Node will continue to stream until the set (finite) duration has elapsed or the power on the Node is cycled. The Node may be stopped from streaming by issuing the **Stop Node** command.

End of Stream

The normal end of a Finite stream is marked by 4-6 consecutive 0xAA (decimal 170) bytes. When parsing the stream, these bytes should be used as a signal that finite streaming has ended.

If, during streaming, the Base Station receives no data from the Node as a result of battery draw-down in the Node, radio interference between the Base Station and the Node, the Node's power is switched off, an out-of-radio-range condition between the Base Station and the Node, etc., random noise may "dribble" from the Base Station; this takes the form of a few odd bytes every second or so.

Response Packet Checksum

This response packet contains a checksum of data bytes 2 – (N-1) where **N** is the total number of bytes. Unlike most checksums which are 2 bytes, this checksum is a single byte.

Relevant Memory Locations

Node EEPROM 16: Samples per Data Set
Node EEPROM 100: Continuous Streaming/LDC Flag

Initiate Low Duty Cycle

Use the **Initiate Low Duty Cycle (LDC)** command to put the Node in LDC mode. The Node will send an LDC packet for every sample at the defined interval for the defined sample rate.

Command Packet:

Byte 1	SOP	0xAA	
Byte 2	Message Type	0x05	
Byte 3	Address Mode	0x00	
Byte 4	Node Address (MSB)	0xXX	
Byte 5	Node Address (LSB)	0xXX	
Byte 6	Payload Length	0x02	
Byte 7	Command Byte	0x00	
Byte 8	Command Byte	0x38	
Byte 9	Checksum (MSB)	0xXX	Checksum of Bytes: 2 - 8
Byte 10	Checksum (LSB)	0xXX	

Response:

The 1 byte response comes directly from the Base Station to verify that the Initiate Low Duty Cycle command was correctly received and sent to the Node.

Byte 1	0xAA

Data Packet:

These packets are received at various, defined intervals after an initiate low duty cycle command has been sent.

Byte 1	SOP	0xAA	
Byte 2	Message Type	0x07	
Byte 3	Address Mode	0x04	
Byte 4	Node Address(MSB)	0xXX	
Byte 5	Node Address(LSB)	0xXX	
Byte 6	Payload Length	0x0XX	
Byte 7	Application Identifier	0x02	
Byte 8	Channel Mask	0xXX	
Byte 9	Sample Rate	0xXX	
Byte 10	Data Type	0xXX	
Byte 11	Timer Tick (MSB)	0xXX	
Byte 12	Timer Tick (LSB)	0xXX	
Byte 13	Channel 1 Data (MSB)	0xXX	
Byte 14	Channel 1 Data (LSB)	0xXX	
Bytes 15 through N-4	Repeat MSB and LSB bytes for the channels that are active	0xXX	
Byte N-3	LQI	0xXX	
Byte N-2	Base Station RSSI	0xXX	
Byte N-1	Checksum (MSB)	0xXX	Checksum of Bytes: 2 – (N-4)
Byte N	Checksum (LSB)	0xXX	

Data Type

The response packet data type can contain the following values for the type of data transmitted:

- 0x01 = 2 byte integers
- 0x02 = 4 byte floats
- 0x03 = 2 byte integers (non-bit-shifted)

Note: A data type of 0x01 signifies a 2 byte integer that must be divided by 2 to get the correct value.

Sampling Rate

The Sample Rate that was used during the datalogging session. See the Sample Rates section for a table of the values and their corresponding sampling rates.

Relevant Memory Locations

Node EEPROM 12:	Active Channel Mask
Node EEPROM 16:	Samples Per Data Set
Node EEPROM 72:	LDC / Synchronized Sampling Rate
Node EEPROM 100:	Unlimited Sampling Flag

See also: Data Format (MSB, LSB) , Checksum , Stop Node

Initiate Synchronized Sampling

The **Synchronized Sampling** command is very similar to the Low Duty Cycle command. Use this command to put the node into Synchronized Sampling mode. Once in this mode, the node must receive a beacon from the Base Station to begin sampling and transmitting packets of data. These packets are unique in that they contain buffered, time-stamped data.

For Synchronized Sampling mode, the wireless sensor network needs to be configured so that each node is assigned an appropriate TDMA slot prior to issuing the Synchronized Sampling start command. To set up the wireless sensor network, use the Synchronized Sampling Wizard within Node Commander®. Node Commander® may also be used to start the network.

Note:
Failure to configure the network correctly will lead to wireless packets colliding, resulting in a large amount of lost data.

Command Packet:

Byte 1	SOP	0xAA	
Byte 2	Message Type	0x05	
Byte 3	Address Mode	0x00	
Byte 4	Node Address (MSB)	0xXX	
Byte 5	Node Address (LSB)	0xXX	
Byte 6	Payload Length	0x02	
Byte 7	Command ID (MSB)	0x00	
Byte 8	Command ID (LSB)	0x3B	
Byte 9	Checksum (MSB)	0xXX	Checksum of Bytes: 2 - 8
Byte 10	Checksum (LSB)	0xXX	

Response:

Byte 1	SOP	0xAA	
Byte 2	DSF	0x07	
Byte 3	Message Type	0x00	
Byte 4	Node Address (MSB)	0xXX	
Byte 5	Node Address (LSB)	0xXX	
Byte 6	Payload Length	0x03	
Byte 7	Command ID (MSB)	0x00	
Byte 8	Command ID (LSB)	0x3B	
Byte 9	Fail Type / Open Byte	0x00	
Byte 10	LQI	0xXX	
Byte 11	Base Station RSSI	0xXX	
Byte 12	Checksum (MSB)	0xXX	Checksum of Bytes: 2 - 9
Byte 13	Checksum (LSB)	0xXX	

Data Packet:

These packets are received at various, defined intervals after an initiate synchronized sampling command has been sent to an mXRS node.

Byte 1	SOP	0xAA	
Byte 2	Message Type	0x07	
Byte 3	Application Data Type	0x0A	
Byte 4	Node Address(MSB)	0xXX	
Byte 5	Node Address(LSB)	0xXX	
Byte 6	Payload Length	0xXX	
Byte 7	Application Identifier	0xXX	
Byte 8	Channel Mask	0xXX	
Byte 9	Sample Rate	0xXX	
Byte 10	Data Type	0xXX	
Byte 11	Sweep Tick (MSB)	0xXX	
Byte 12	Sweep Tick (LSB)	0xXX	
Byte 13	UTC Timestamp, seconds	0xXX	
Byte 14	UTC Timestamp, seconds	0xXX	
Byte 15	UTC Timestamp, seconds	0xXX	
Byte 16	UTC Timestamp, seconds	0xXX	
Byte 17	Timestamp, nanoseconds	0xXX	
Byte 18	Timestamp, nanoseconds	0xXX	
Byte 19	Timestamp, nanoseconds	0xXX	
Byte 20	Timestamp, nanoseconds	0xXX	
Byte 21	Channel 1 Sweep 1 Data (MSB)	0xXX	
Byte 22	Channel 1 Sweep 1 Data (LSB)	0xXX	
Byte 23	Channel 2 Sweep 1 Data (MSB)	0xXX	
Byte 24	Channel 2 Sweep 1 Data (MSB)	0xXX	
Byte 25	Channel 1 Sweep 2 Data (MSB)	0xXX	
Byte 26	Channel 1 Sweep 2 Data (LSB)	0xXX	
Byte 27	Channel 2 Sweep 2 Data (MSB)	0xXX	
Byte 28	Channel 2 Sweep 2 Data (LSB)	0xXX	
Bytes 29 through N-4	Repeat MSB and LSB bytes for the channels that are active, and for all sweeps	0xXX	
Byte N-3	LQI	0xXX	
Byte N-2	Base Station RSSI	0xXX	
Byte N-1	Checksum (MSB)	0xXX	Checksum of Bytes: 2 – (N-4)
Byte N	Checksum (LSB)	0xXX	

Application Identifier

The response packet application identifier can contain the following values for their current mode of operation:

- 0x01 = Burst Mode
- 0x02 = Continuous Mode

Data Type

The response packet data type can contain the following values for the type of data transmitted:

- 0x01 = 2 byte integers
- 0x02 = 4 byte floats
- 0x03 = 2 byte integers (non-bit-shifted)

Note: A data type of 0x01 signifies a 2 byte integer that must be divided by 2 to get the correct value.

Channel Data

Multiple sweeps of data per channel may be contained in one packet. The amount of sweeps in the packet can be determined via the following:

- **Bytes Before Data** = 12 (Bytes 7 – 20 are not channel data)
- **Length Multiplier** = 2 for 2 byte data, or 4 for float data (determined using Byte 10 – Data Type)
- **#Sweeps** = ((Payload Length – **Bytes Before Data**) / # of Active Channels) / **Length Multiplier**)

Timestamp/Tick

The Timestamp data (Bytes 13-20) can be built into the current UTC time in nanoseconds via the following:

- Convert Bytes 13-16 into a DWORD. This value is the current UTC time in seconds.
- Convert Bytes 17-20 into a DWORD. This value is the current nanoseconds.
- Convert the first value (UTC time - Bytes 13-16) from seconds to nanoseconds, and add to it the nanoseconds (Bytes 17-20). This result is the current UTC time in nanoseconds.

Although each packet may contain more than one sweep per channel, there is only one timestamp and tick acquired per packet. Therefore the tick and timestamp values must be incremented manually for each sweep within the packet. This can be done via the following:

- Using the sample rate of the node, determine your increment value:
  - If sample rate is 32 Hz, the increment is (1 / 32) = 0.03125
  - If sample rate is 1 per 2 seconds, the increment is (1 / 0.5) = 2
  - Multiply the increment by 1,000,000,000 to convert to nanoseconds.
- For each sweep in the packet, add the calculated value to the original timestamp.
- For each sweep, increment the tick by 1.

Sampling Rate

The Sample Rate that was used during the datalogging session. See the Sample Rates section for a table of the values and their corresponding sampling rates.

Relevant Memory Locations

Node EEPROM 12:	Active Channel Mask
Node EEPROM 16:	Samples Per Data Set
Node EEPROM 72:	LDC / Synchronized Sampling Rate
Node EEPROM 100:	Unlimited Sampling Flag
Node EEPROM 36:	TDMA Slot Address
Node EEPROM 262:	Sampling Mode

See also: Data Format (MSB, LSB) , Checksum , Enable Beacon, Disable Beacon , Stop Node

Read Single Sensor

Use the **Read Single Sensor** command to read the current value of a specified channel.

Command Data:

Byte 1	Command Byte	0x03
Byte 2	Node Address (MSB)	0xXX
Byte 3	Node Address (LSB)	0xXX
Byte 4	0x01	0x01
Byte 5	Channel Number (1-8)	0xXX

Success Response:

Byte 1	Command Byte	0x03	
Byte 2	Value (MSB)	0xXX	
Byte 3	Value (LSB)	0xXX	
Byte 4	Checksum (MSB)	0xXX	Checksum of Bytes: 2 - 3
Byte 5	Checksum (LSB)	0xXX	

Fail Response:

Byte 1	0x21

Channel Number

The Channel Number of the sensor to read. If a channel number is used that doesn't exist on the Node, erroneous response data will be returned.

See also: Data Format (MSB, LSB) , Checksum

Auto-Balance Channel

Use the **Auto-Balance Channel** command to auto-balance a particular channel on the Node. This command is only applicable to the differential channels on the V-Link®, SG-Link® and SG-Link® OEM.

Command Data:

Byte 1	Command Byte	0x62
Byte 2	Node Address (MSB)	0xXX
Byte 3	Node Address (LSB)	0xXX
Byte 4	Channel to Balance (hardware specific, 1-4)	0xXX
Byte 5	Target Balance Value (MSB)	0xXX
Byte 6	Target Balance Value (LSB)	0xXX

Success Response: No Response
Fail Response: No Response

Channel to Balance

Channels 1, 2, 3 and/or 4 on the V-Link®.
Channel 1 on the SG-Link® and SG-Link® OEM.

Target Balance Value

The target balance value represents the desired sensor output value in bits, with a range of 0-4096. All differential inputs have a programmable offset feature that allows the user to trim sensor offset (see hardware user manual for more information.) This programmable offset can be manually altered by writing to Node EEPROM locations 26-32, or auto-tuned such that the sensor output is balanced to a user-defined target. For example, a common use is to auto-balance to mid-scale (2048 bits) to obtain maximum bipolar dynamic range.

Relevant Memory Locations

Node EEPROM 26: PGA Offset 1
Node EEPROM 28: PGA Offset 2
Node EEPROM 30: PGA Offset 3
Node EEPROM 32: PGA Offset 4

See also: Data Format (MSB, LSB)

OTHER DETAILS

Sample Rates

Synchronized Sampling / Low Duty Cycle Sample Rates

The table below lists the data values and their corresponding Sample Rates that apply to the **Synchronized Sampling** and **Low Duty Cycle** sampling modes.

Value	Sample Rate
123	1 sample per 60 minutes
122	1 sample per 30 minute
121	1 sample per 10 minutes
120	1 sample per 5 minutes
119	1 sample per 2 minutes
118	1 sample per 60 seconds
117	1 sample per 30 seconds
116	1 sample per 10 seconds
115	1 sample per 5 seconds
114	1 sample per 2 seconds
113	1 Hz
112	2 Hz
111	4 Hz
110	8 Hz
109	16 Hz
108	32 Hz
107	64 Hz
106	128 Hz
105	256 Hz
104	512 Hz
103	1024 Hz
102	2048 Hz

Armed Datalogging Sample Rates

The table below lists the data values and their corresponding Sample Rates that apply to the **Armed Datalogging** sampling mode.

Value	Rate
1	2048 Hz
2	1024 Hz
3	512 Hz
4	256 Hz
5	128 Hz
6	64 Hz
7	32 Hz

Broadcast Address

A special node address referred to as the *Broadcast Address* exists. This Broadcast Address is 65535. Nodes shipped from the factory and any nodes in use by the customer should not be addressed as 65535; node addresses should always range between 1 and 65534.

When nodes are idling and listening for commands, they are waiting for commands directed at them specifically. For example, in the case of the Short Ping command, we send 3 bytes, being an initial command byte 0x02 followed by two bytes representing the address of a unique node. This method insures that only the target node responds to the command.

However, in many cases we want to command many nodes to do the same thing. To handle this situation, all nodes are always listening for commands to the Broadcast Address 65535 in addition to commands to their unique address. As an example, this special feature allows us to command many nodes to trigger a datalogging session all at the same time. This is useful to generate sensor data that has the same session start time.

This Broadcast Address can most obviously be used in the following commands:

- Initiate Low Duty Cycle
- Initiate Synchronized Sampling
- Trigger Armed Datalogging
- Erase Session Data
- Write Node EEPROM
- Initiate Sleep Mode
- Stop Node

In addition, commands for waking multiple nodes, cycling the power on multiple nodes, finding nodes with unknown addresses can all be fabricated with this special Broadcast Address.

Node Discovery

The Node has a specialized function whereby it sends out two identification packets every time it is turned on. The packets are actually sent out on all 16 radio channels, allowing any Base Station within range to receive the identification packets, regardless of the Base Station's current frequency assignment. The Base Station immediately passes these packets to the host serial port.
The Node will only send out a Node discovery packet when its power is cycled.

Node Discovery Packet:

Byte 1	SOP	0xAA	
Byte 2	Message Type	(Byte 2 & 0x08) == 0	
Byte 3	Address Mode	0x00	
Byte 4	Node Address (MSB)	0xXX	
Byte 5	Node Address (LSB)	0xXX	
Byte 6	Packet Length	0x03	
Byte 7	Radio Channel	0xXX	
Byte 8	Model Number (MSB)	0xXX	
Byte 9	Model Number (LSB)	0xXX	
Byte 10	LQI	0xXX	
Byte 11	Base Station RSSI	0xXX	
Byte 12	Checksum (MSB)	0xXX	Checksum of Bytes: 2 - 9
Byte 13	Checksum (LSB)	0xXX	

Message Type

The message type byte must be vetted to insure a valid packet. The byte must be ANDed with 0x08 and the result should be 0; if <>0, the packet is not valid.

Radio Channel

The radio channel value corresponds to the channel on which the Node is currently communicating. This is the same value stored in Node EEPROM 90 and sends only values that are valid within EEPROM 90. Please, reference the Node EEPROM map for position 90 to identify valid radio channels.

Model Number

The model number corresponds to the model number of the Node stored in Node EEPROM 112. This value identifies the type of Node that had its power cycled. Please reference the Node EEPROM map for position 112 to identify valid model numbers.

See also: Data Format (MSB, LSB) , Checksum

Calibration Coefficients

Calibration coefficients are linear scaling filters applied to convert a channel's raw bit value to physical or engineering units. Calibration coefficients can be generated through Node Commander® calibration wizards or via manual calibration by the end user. Please see the node hardware manual for more information on general calibration procedures. This documentation explains how to use existing calibration coefficients in custom software.

Calibration coefficient information is stored in non-volatile EEPROM memory on the node, at locations 150 through 228 inclusive. Each channel requires 10 bytes to hold the equation ID, unit ID, slope, and offset values.

Please note:
1) Calibration coefficient offset is different from a node's hardware offset settings.
2) Calibration coefficients represent a post-processing step because the conversion from bits to physical units takes place on the host computer and not on the Node.

EEPROM Location		Meaning		EEPROM Location		Meaning
150	MSB	ch1 Equation ID		190	MSB	ch5 Equation ID
	LSB	ch1 Unit ID			LSB	ch5 Unit ID
152		ch1 slope		192		ch5 slope
154				194		
156		ch1 offset		196		ch5 offset
158				198		
160	MSB	ch2 Equation ID		200	MSB	ch6 Equation ID
	LSB	ch2 Unit ID			LSB	ch6 Unit ID
162		ch2 slope		202		ch6 slope
164				204		
166		ch2 offset		206		ch6 offset
168				208		
170	MSB	ch3 Equation ID		210	MSB	ch7 Equation ID
	LSB	ch3 Unit ID			LSB	ch7 Unit ID
172		ch3 slope		212		ch7 slope
174				214		
176		ch3 offset		216		ch7 offset
178				218		
180	MSB	ch4 Equation ID		220	MSB	ch8 Equation ID
	LSB	ch4 Unit ID			LSB	ch8 Unit ID
182		ch4 slope		222		ch8 slope
184				224		
186		ch4 offset		226		ch8 offset
188				228		

Below are more in depth explanations of each calibration coefficient field.

Equation ID

The Equation ID is an integer value used to determine which calibration coefficient is applied to the current channel. The Equation ID and Unit ID occupy the same EEPROM location. The Equation ID is the MSB and the Unit ID is the LSB of this location.

The Standard Format equation is used by MicroStrain for all new devices. The Legacy Strain and Legacy Acceleration formulas were used by previous MicroStrain software and may be present on older nodes. The Legacy equations each imply a unit type, microstrain (με) for Legacy Strain and G's for Legacy Acceleration, while the standard equation is more flexible and doesn't enforce a specific unit. With the Standard Equation, the unit is stored separately. We recommend using the Standard Format for all custom calibration. The table below shows the three equations.

Equation ID	Name	Equation
0x01	Bits (None)	`value = bits`
0x01	Legacy Strain	`value = slope * (bits + offset)`
0x02	Legacy Acceleration	`value = (bits – offset) / slope`
0x04	Standard Format	`value = slope * bits + offset`

Any other value for the Equation ID will be treated as if no calibration coefficient is applied to the channel.

Unit Type ID

The Unit Type ID is an integer value used to determine which unit is applied to the current channel. The Unit Type ID value is the LSB of EEPROM 0. The table below shows the Unit Type ID values and their corresponding unit symbols and types.

Unit Type ID	Unit Symbol	Unit Type
0x00	bits	other
0x01	bits	bits
0x02	ε	Strain
0x03	με	microStrain
0x04	G	acceleration due to gravity
0x05	m/s^2	meters per second squared
0x06	V	volts
0x07	mV	millivolts
0x08	µV	microvolts
0x09	°C	degrees Celsius
0x0A	K	Kelvin
0x0B	°F	degrees Fahrenheit
0x0C	m	meters
0x0D	mm	millimeters
0x0E	µm	micrometers
0x0F	Lbf	pound force
0x10	N	Newtons
0x11	kN	kiloNewtons
0x12	kg	kilograms
0x13	bar	bar
0x14	psi	pounds per square inch
0x15	atm	atmospheric pressure
0x16	mmHg	millimeters of mercury
0x17	Pa	Pascal
0x18	MPa	megaPascal
0x19	kPa	kiloPascal
0x1A	degrees	degrees
0x1B	degrees/s	degrees per second
0x1C	rad/s	radians per second
0x1D	%	percent
0x1E	rpm	revolutions per minute
0x1F	Hz	hertz
0x20	%RH	relative humidity
0x21	mV/V	milliVolt/Volt

Slope and Offset

The slope and offset are 4-byte floating point values in IEE 754-single precision format. The example below demonstrates how to read the floating point values in C++.

```
typedef unsigned char BYTE;
typedef unsigned short WORD;

float MakeFloat(WORD eepromLoWord, WORD eepromHiWord)
{
	float f = 0;

	//map the float to a byte array
	BYTE* tmp = (BYTE*)&f;

	//set each byte of the float via the tmp byte pointer
	tmp[0] = eepromLoWord >> 8;
	tmp[1] = eepromLoWord & 0xFF;
	tmp[2] = eepromHiWord >> 8;
	tmp[3] = eepromHiWord & 0xFF;

	return f;
}
```

The following table displays the standard slope and offset values to convert single-ended input channels to volts, and the internal temperature sensor to degrees Celsius.

Unit	Slope	Offset	Applies To
Volts	.000732	0	V-Link – Channels 5, 6, 7 SG-Link – Channel 4 SG-Link OEM – Channel 4
Temp C	.117188	-67.84	V-Link – Channel 8 G-Link – Channel 4 SG-Link – Channel 3 SG-Link OEM – Channel 3

Example

Below is an example of converting EEPROM addresses to their corresponding calibration coefficient. The base address is already included in the EEPROM address.

EEPROM Address	Value
180	1033
182	17152
184	61501
186	5294
188	34754

The above EEPROM values correspond to channel 4. This can be told because the EEPROM address, EEPROM 180, corresponds to channel 4 in the EEPROM map.

First determine which calibration coefficient is to be applied to channel 4. The calibration coefficient Equation ID is the MSB of EEPROM 180. The MSB of 1033 is 4, 1033 >> 8 (i.e. 1033 / 256). The Equation ID therefore corresponds to the Standard Format. Thus, the formula to use is slope * bits + offset as described earlier.

Second, the Unit Type ID can be calculated. The calibration coefficient Unit Type ID is the LSB of EEPROM 180. The LSB of 1033 is 9, 1033 & 0x00FF (i.e. 1033 Mod 256). The Unit Type ID therefore corresponds to degrees Celsius.

Next, the slope and offset must be calculated. The slope has the values 17152 and 61501 for EEPROM addresses 182 and 184, respectively. Using the formula written earlier and knowing these 4 bytes are in big endian format, the floating point equivalent is 0.117188.

The offset has the values 5294 and 34754 for EEPROM addresses 186 and 188, respectively. Using the same formula as calculating the slope and knowing these 4 bytes are in big endian format, the floating point equivalent is -67.84.

As noted in the table of slopes and offsets for the Standard Format, this calibration coefficient converts incoming bits to degrees Celsius for channel 4.

Channel Mask

The Channel Mask is a value contained in EEPROM address 12 of a node which dictates which of the node's channels are active. This value sets the channels that will be sampled during datalogging, streaming, low duty cycle or high speed streaming. The mask is an 8-bit value. Each of the 8 bits of the mask correspond to one of the Nodes channels. Bit 1 corresponds to channel 1, bit 2 to channel 2, bit 8 to channel 8, etc. When the bit is set to 1, the session will sample data for that channel. If the bit is set to 0, the session will not sample data for that channel.

Mask Layout

Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1
CH8	CH7	CH6	CH5	CH4	CH3	CH2	CH1

Example showing channels 1, 3 and 4 active

Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1
0	0	0	0	1	1	0	1

In this example the mask has been set with channels 1, 3 and 4 active (they are set with 1's). This value equates as follows:

- Binary = 1101
- Decimal = 13
- Hex = 0x0D

C++ Example Functions:

```
typedef unsigned char BYTE;

BYTE ToMask(bool ch1,bool ch2,bool ch3,bool ch4,bool ch5,bool ch6,bool
ch7,bool ch8)
{
    BYTE channelMask = 0;
    if(ch1){ channelMask |= (1 << 0);} //channel 1 - 0000 0001
    if(ch2){ channelMask |= (1 << 1);} //channel 2 - 0000 0010
    if(ch3){ channelMask |= (1 << 2);} //channel 3 - 0000 0100
    if(ch4){ channelMask |= (1 << 3);} //channel 4 - 0000 1000
    if(ch5){ channelMask |= (1 << 4);} //channel 5 - 0001 0000
    if(ch6){ channelMask |= (1 << 5);} //channel 6 - 0010 0000
    if(ch7){ channelMask |= (1 << 6);} //channel 7 - 0100 0000
    if(ch8){ channelMask |= (1 << 7);} //channel 8 - 1000 0000
    return channelMask;
}

void FromMask(BYTE channelMask)
{
    bool ch1 = (channelMask & (1 << 0)) != 0; //channel 1 - 0000 0001
    bool ch2 = (channelMask & (1 << 1)) != 0; //channel 2 - 0000 0010
    bool ch3 = (channelMask & (1 << 2)) != 0; //channel 3 - 0000 0100
    bool ch4 = (channelMask & (1 << 3)) != 0; //channel 4 - 0000 1000
    bool ch5 = (channelMask & (1 << 4)) != 0; //channel 5 - 0001 0000
    bool ch6 = (channelMask & (1 << 5)) != 0; //channel 6 - 0010 0000
    bool ch7 = (channelMask & (1 << 6)) != 0; //channel 7 - 0100 0000
    bool ch8 = (channelMask & (1 << 7)) != 0; //channel 8 - 1000 0000
}
```

```
void SetChannelInMask(int channel,bool enable,BYTE& channelMask)
{
    //channel 1 is bit 0
    channel -= 1;
    if(enable)
    {
        //Adding a channel and 'or' in the channel's bit.
        channelMask |= (1<<channel);
    }
    else
    {
        //Removing a channel, mask out the channel's bit.
        channelMask &= 0xff ^(1<<channel);
    }
}

bool IsChannelSetInMask(int channel,BYTE& channelMask)
{
    //Channel 1 is bit 0
    channel-=1;
    //Check the channel's specific bit
    return (channelMask & (1<<channel)) != 0;
}
```

CP210x USB to UART Bridge Controller

The MicroStrain WSDA®-Base-104 USB Station and WSDA®-Base-101 Analog Output Base Station operate with a special driver installed in the operating system.

The USB interface, from a physical standpoint, resides in a communication cable connecting a standard USB connector to the host. The USB communication on the base station is provided via a Silicon Laboratories CP210x USB to UART Bridge chip. The CP210x is a single-chip USB to UART bridge that converts data traffic between USB and UART formats. The chip includes a complete USB 2.0 full-speed function controller, bridge control logic and a UART interface with transmit/receive buffers and modem handshake signals. Specifications for the chip may be found at:

https://www.silabs.com/products/interface/usbtouart/Pages/default.aspx

This physical architecture is supported by a Silicon Laboratories CP210x USB to UART Bridge Virtual COM Port (VCP) driver installed on the host. This driver is required for device operation as a Virtual COM Port to facilitate host communication. The driver is normally installed during installation of MicroStrain's Node Commander® software. The driver may also be downloaded from Silicon Labs at:

https://www.silabs.com/products/mcu/Pages/USBtoUARTBridgeVCPDrivers.aspx

With the installation of this driver, the developer will find that communication between host and base station will be, for all intents and purposes, typical serial communication. The various coding languages will interact as if a standard serial port existed.

To determine if the driver is properly installed, connect the base station to the host and follow these steps:

1. Click Start on the Windows desktop.
2. Click Control Panel.
3. Click System icon.
4. System Properties window appears.
5. Click Hardware tab.
6. Click Device Manager.
7. Device Manger window appears.
8. Locate Ports(Com&LPT) in the tree.
9. Double-click the + to further open the Ports tree.
10. Do you see a device named 'CP210x USB to UART Bridge Controller (ComX)' or 'Silicon Labs CP210x USB to UART Bridge (ComX)? (with X representing a comm port number).
11. If so, the driver is installed.
12. If not, the base station is malfunctioning or the driver is not installed.

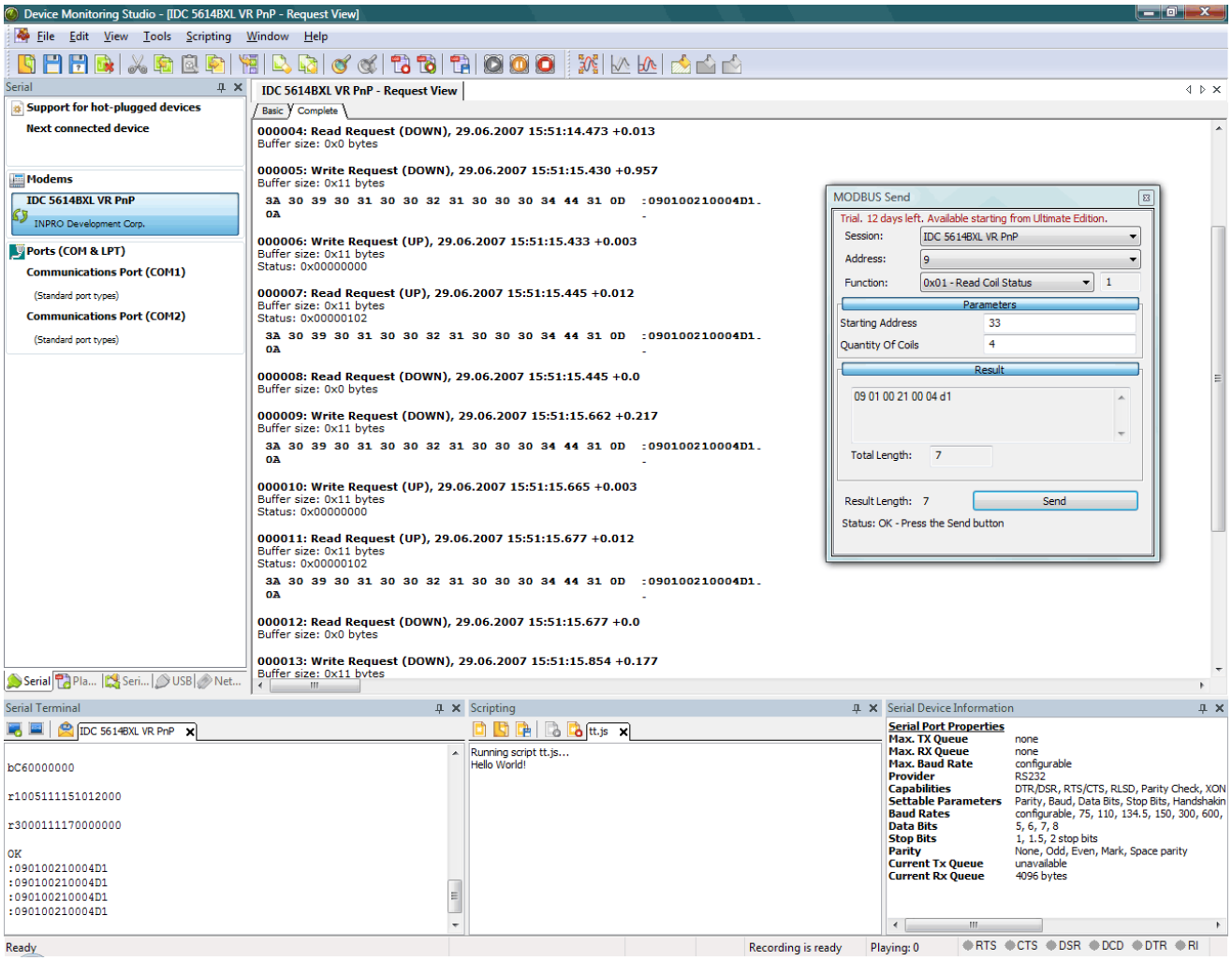
SUGGESTED DEBUGGING TOOLS

We have found that software port *sniffers* are incredibly valuable when building applications with serial communication. Here are some of our favorites; we highly recommend their use.

Serial Port Monitor

http://www.hhdsoftware.com/serial-monitor

Serial Port Monitor allows you to capture, display, analyze, record and replay all serial port data exchanged between the Windows application and the serial device. It can be successfully used in application development, device driver or serial hardware development and offers the powerful platform for effective coding, testing and optimization.



Comm Operator Pal

http://www.serialporttool.com/CommPalInfo.htm

Comm Operator Pal is a free tool to test and debug RS232 devices that communicated with serial port, TCP/IP or UDP. It supports data in Text, Decimal and Hex format. Data can be sent in a list automatically. Single data can be sent repeatedly. It has a built in check-sum calculator.