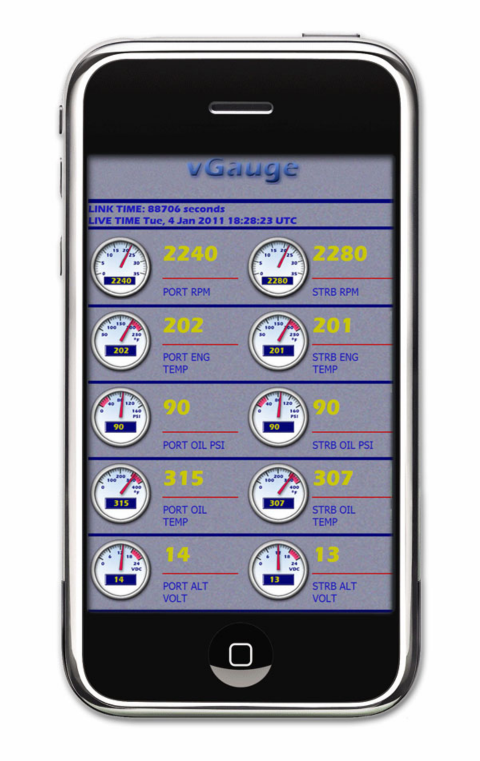
SeaSmart.Net™

Version 1.1.0 – Protocol Specification



Chetco Digital Instruments

Preliminary Specification 032511

WARNING!

USE THIS UNIT ONLY AS AN AID TO MONITORING ENGINE PERFORMANCE INFORMATION.

CAUTION

When showing sensor data, this unit will only show information based on the sender used and its installed position.

The operating and storage temperature for your unit is from -4 degrees to+167 degrees Fahrenheit (-20 to +75 degrees Celsius). Extended storage temperatures higher or lower than specified will cause the liquid crystal display to fail. Neither this type of failure nor its consequences are covered by the warranty. For more information, consult the factory customer service department.

All features and specifications subject to change without notice.

Chetco Digital Instruments may find it necessary to change or end our policies, regulations, and special offers at any time. We reserve the right to do so without notice.

All screens in this manual are simulated.

NOTICE!

Free software upgrades will be available on our website at http:// www.chetcodigital.com as they are released. Please check our website periodically for these and other information as they become available.

Thank you for choosing Chetco Digital Instruments

This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (1) this device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

Note:

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna.

- Increase the separation between the equipment and receiver.

- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.

- Consult the factory customer service department for help.

SPECIFICATIONS

NMEA 2000 Instrumentation Sentences Supported

126992 – System Time
127250 – Vessel Heading
127257 – Vessel Attitude
127251 – Rate of Turn
127488 – Engine Data - Rapid Update
127489 – Engine Data - Dynamic Update
127493 – Transmission Data - Dynamic Update
127505 – Fluid Data - Dynamic Update
127508 – Battery Status - Dynamic Update
127501 – Binary Switch Status - Dynamic Update
130306 – Wind Data
130311 – Environmental Data
130312 – Temperature Data
129025 – Position Data - Rapid
130323 – Weather Station Location Data
129026 – SOG and COG Rapid Update

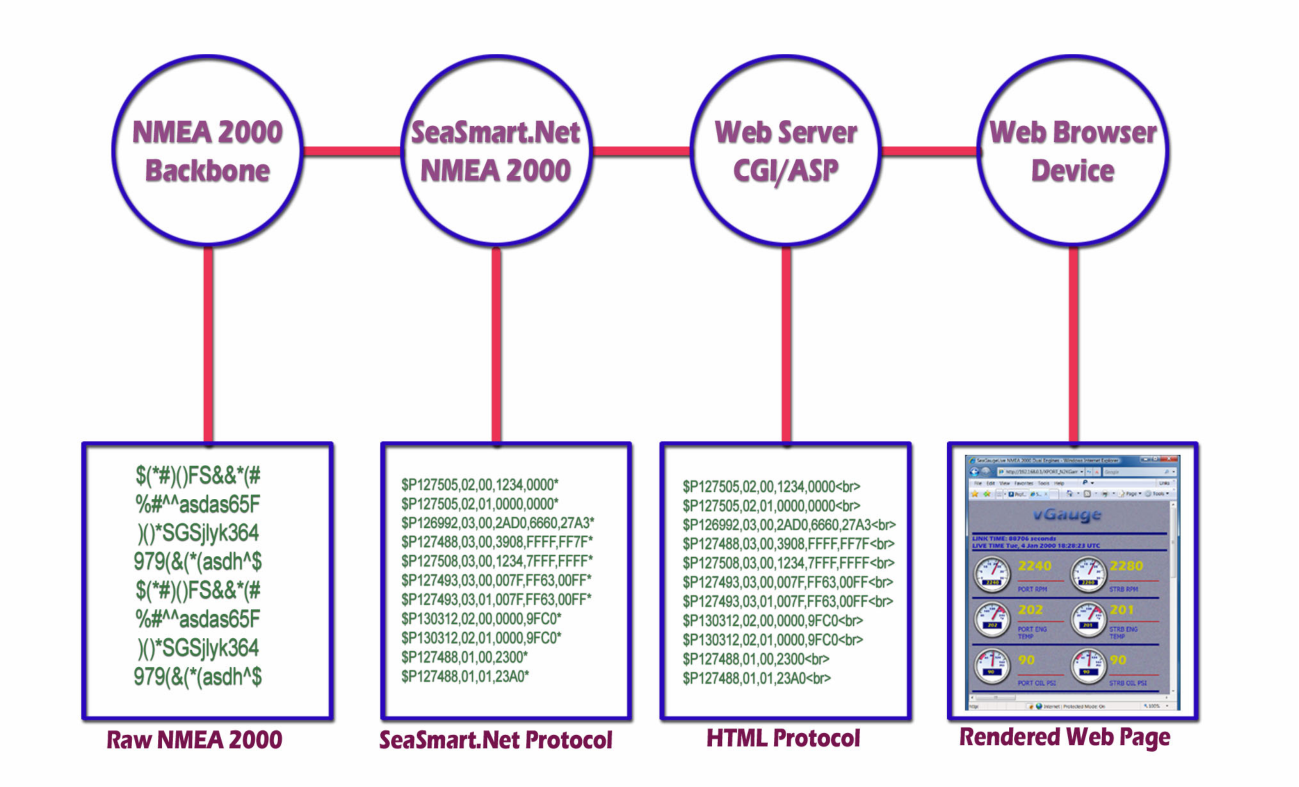
Table of Contents

Introduction

Welcome

Thank you for purchasing a Chetco Digital Instruments product.

SeaSmart.Net™ is a hardware and software system that converts raw NMEA 2000 PGNs into a protocol compatible with standard Web Browsers. SeaSmart.Net™ consists of a NMEA 2000 gateway, HTTP Protocol Translator, and embedded Web Server. The Web Server stores HTML documents which render NMEA 2000 PGNs into real-time graphical display by using the JavaScript programming language. The embedded CGI engine dynamically creates a simple HTML document which is processed by the client Browser.



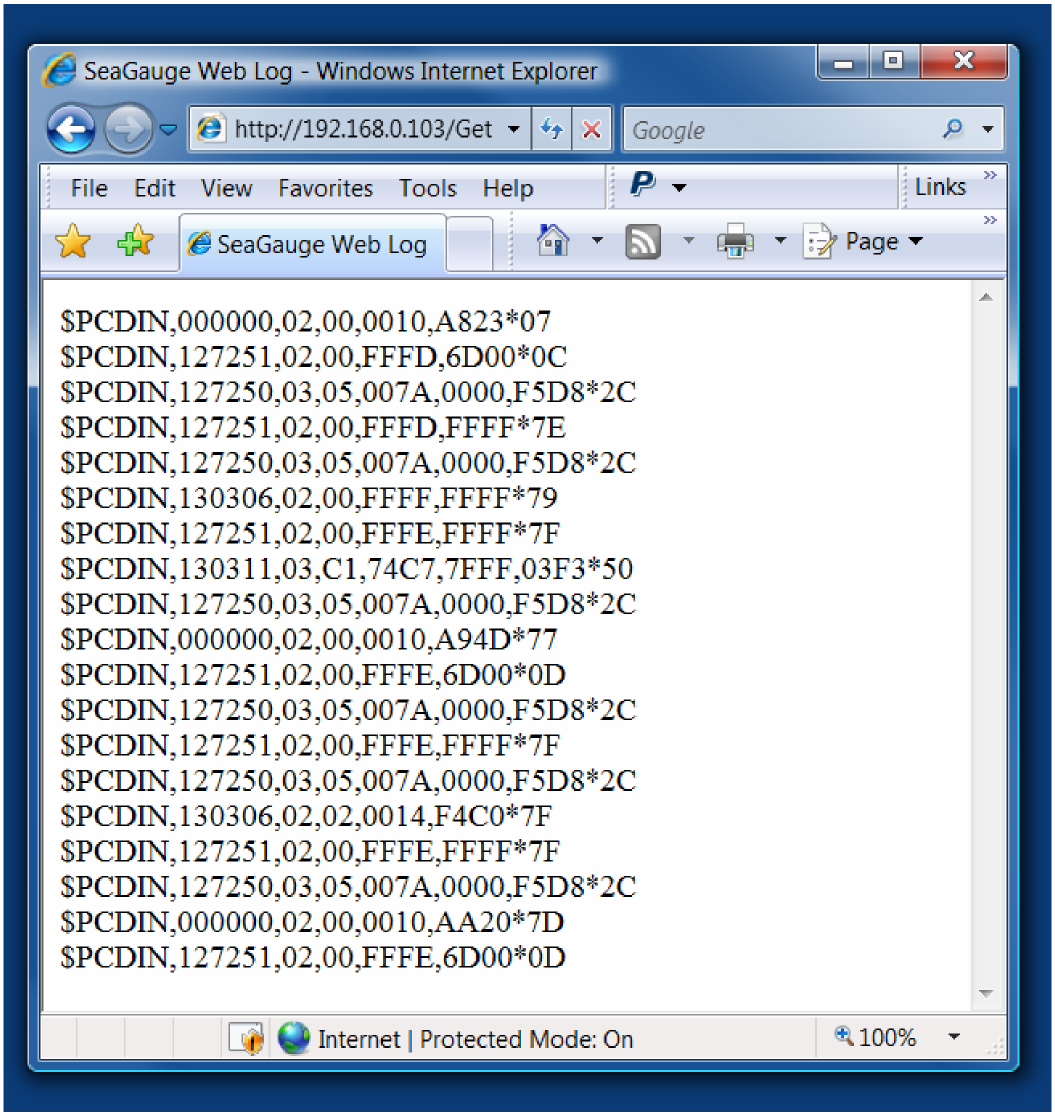
```
$(*#)()FS&&*(#
%#^^asdas65F
)()*SGSjlyk364
979(&(*(asdh^$
$(*#)()FS&&*(#
%#^^asdas65F
)()*SGSjlyk364
979(&(*(asdh^$
```

```
$P127505,02,00,1234,0000*
$P127505,02,01,0000,0000*
$P126992,03,00,2AD0,6660,27A3*
$P127488,03,00,3908,FFFF,FF7F*
$P127508,03,00,1234,7FFF,FFFF*
$P127493,03,00,007F,FF63,00FF*
$P127493,03,01,007F,FF63,00FF*
$P130312,02,00,0000,9FC0*
$P130312,02,01,0000,9FC0*
$P127488,01,00,2300*
$P127488,01,01,23A0*
```

```
$P127505,02,00,1234,0000<br>
$P127505,02,01,0000,0000<br>
$P126992,03,00,2AD0,6660,27A3<br>
$P127488,03,00,3908,FFFF,FF7F<br>
$P127508,03,00,1234,7FFF,FFFF<br>
$P127493,03,00,007F,FF63,00FF<br>
$P127493,03,01,007F,FF63,00FF<br>
$P130312,02,00,0000,9FC0<br>
$P130312,02,01,0000,9FC0<br>
$P127488,01,00,2300<br>
$P127488,01,01,23A0<br>
```

Raw NMEA 2000 data is read directly from the Backbone and translated to SeaSmart.Net protocol so it can be passed on and processed by Web Servers. The Web Servers then in turn format the data into HTML friendly documents which can be retrieved by common Browser based devices. The Browsers render the live data into graphical displays that are easily customizable via standard Web design tools.

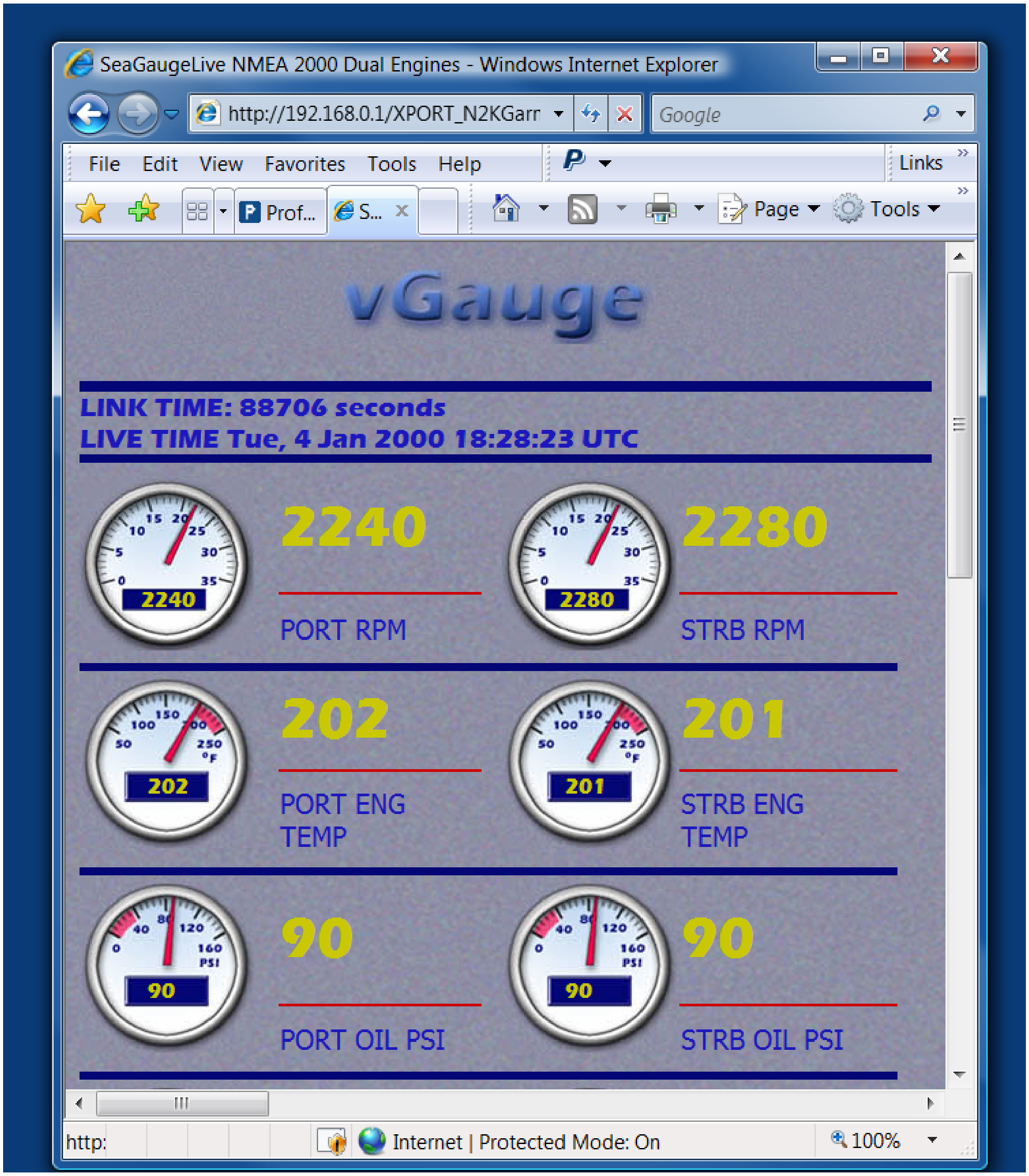
The native NMEA 2000 PGNs are processed in real time to create a new HTML document each second. The following is a example of typical update page.



```
$PCDIN,000000,02,00,0010,A823*07
$PCDIN,127251,02,00,FFFD,6D00*0C
$PCDIN,127250,03,05,007A,0000,F5D8*2C
$PCDIN,127251,02,00,FFFD,FFFF*7E
$PCDIN,127250,03,05,007A,0000,F5D8*2C
$PCDIN,130306,02,00,FFFF,FFFF*79
$PCDIN,127251,02,00,FFFE,FFFF*7F
$PCDIN,130311,03,C1,74C7,7FFF,03F3*50
$PCDIN,127250,03,05,007A,0000,F5D8*2C
$PCDIN,000000,02,00,0010,A94D*77
$PCDIN,127251,02,00,FFFE,6D00*0D
$PCDIN,127250,03,05,007A,0000,F5D8*2C
$PCDIN,127251,02,00,FFFE,FFFF*7F
$PCDIN,127250,03,05,007A,0000,F5D8*2C
$PCDIN,130306,02,02,0014,F4C0*7F
$PCDIN,127251,02,00,FFFE,FFFF*7F
$PCDIN,127250,03,05,007A,0000,F5D8*2C
$PCDIN,000000,02,00,0010,AA20*7D
$PCDIN,127251,02,00,FFFE,6D00*0D
```

This HTML document is then available to client Browsers to process and render in any fashion.

A typical JavaScript enabled Browser may use a combination of XMLHttpRequest and images to render as in the following example



SeaGaugeLive NMEA 2000 Dual Engines - Windows Internet Explorer

http://192.168.0.1/XPORT_N2KGarr

LINK TIME: 88706 seconds

LIVE TIME Tue, 4 Jan 2000 18:28:23 UTC

The purpose of this document is to describe the HTTP compatible translated NMEA 2000 Protocol so that customized Web Pages can be created.

SeaSmart.Net Protocol

Protocol Format

The SeaSmart.Net Protocol is an 7-bit ASCII based format to retain compatibility with all types of Web Browsers. Each received PGN instance is converted in to a sequence of comma separated fields and terminated with the standard NMEA 0183 "*" character and twp character (1 byte) checksum.

Each field is a fixed length with a variable number of fields depending on the type of data.

The Protocol header starts with a "$" symbol (0x24 HEX) followed by the letters "PCDIN" then the specified six digit NMEA 2000 PGN number. This PGN number corresponds to the type of data to follow as well as the number of parameters.

The following is an example for the PGN 127505 (Fluid Level)

Start	PGN	Num of Fields	Instance Number	Field 1	Field 2	Termination	Check Sum
$PCDIN	127505	02	00	1234	0000	*	58

Each field in the protocol is a fixed length separated by a comma. The only variable on the overall length is the number of data fields included in each PGN.

The following table summarizes the size for each field.

Field	Size	Range	Type
Start	2 Bytes	-	ASCII Character
PGN Number	6 Bytes	-	ASCII Characters
Number of fields	2 Bytes	0 - 256	ASCII Hexadecimal
Instance Number	2 Bytes	0 - 256	ASCII Hexadecimal
Data Field	4 Bytes	0 - 65535	ASCII Hexadecimal
End	1 Bytes	-	ASCII Characters
Check Sum	2 Bytes	-	ASCII Characters

A single update interval may contain one or more PGN sentences depending on the number of newly received NMEA 2000 PGNs. Each unique PGN Instance will have an individual sentence. Only PGNs recognized by the NMEA 2000 Gateway will be processed with all others being ignored to reduce network load.

$PCDIN,000000 – Link Up Time

Field	Name	Length	Resolution	Value	Comment
1	Start	6 Bytes	ASCII CHAR	$PCDIN	
2	PGN ID	6 Bytes	ASCII CHAR	000000	
3	Num of Fields	2 Bytes	ASCII HEX	2	
4	Instance	2 Bytes	ASCII HEX	0	
5	Seconds LB	4 Bytes	1	Seconds	
6	Seconds HB	4 Bytes	1	Seconds	Seconds * 65536
8	END	1 Bytes	ASCII CHAR	*	
9	Check Sum	1 Bytes	ASCII CHAR	HEX	XOR from $ to *
Sample	$PCDIN,000000,02,00,06F5,7568*07				
Notes	Total seconds = Seconds HB * 65536 + Seconds LB				

$PCDIN,126992 – System Time

Field	Name	Length	Resolution	Value	Comment
1	Start	6 Bytes	ASCII CHAR	$PCDIN	
2	PGN ID	6 Bytes	ASCII CHAR	126992	
3	Num of Fields	2 Bytes	ASCII HEX	3	
4	Instance	2 Bytes	ASCII HEX	0	
				1	
				2	
				3	
				4	
				5	Crystal Clock
5	Days	4 Bytes	1	Day	Days since 1970
6	Seconds LB	4 Bytes	1	Seconds	
7	Seconds HB	4 Bytes	1	Seconds	Seconds * 65536
8	END	1 Bytes	ASCII CHAR	*	
9	Check Sum	1 Bytes	ASCII CHAR	HEX	XOR from $ to *
Sample	$PCDIN,126992,03,05,3AB7,9E10,16F4*5E				
Notes	Total seconds = Seconds HB * 65536 + Seconds LB				

$PCDIN,127250 – Vessel Heading

Field	Name	Length	Resolution	Value	Comment
1	Start	6 Bytes	ASCII CHAR	$PCDIN	
2	PGN ID	6 Bytes	ASCII CHAR	127250	
3	Num of Fields	2 Bytes	ASCII HEX	3	
4	Reference	2 Bytes	ASCII HEX	0	True
				1	-
				2	Apparent
				3	-
				4	Ref to Water
5	Heading	4 Bytes	.0001	radians	Degrees = X * 57.29 * .0001
6	Deviation	4 Bytes	.0001	radians	Degrees = X * 57.29 * .0001
7	Variation	4 Bytes	.0001	radians	Degrees = X * 57.29 * .0001
8	END	1 Bytes	ASCII CHAR	*	
9	Check Sum	1 Bytes	ASCII CHAR	HEX	XOR from $ to *
Sample	$PCDIN,127250,03,05,F1C0,0000,F5D8*5E				
Notes					

$PCDIN,127257 – Vessel Attitude

Field	Name	Length	Resolution	Value	Comment
1	Start	6 Bytes	ASCII CHAR	$PCDIN	
2	PGN ID	6 Bytes	ASCII CHAR	127257	
3	Num of Fields	2 Bytes	ASCII HEX	2	
4	Instance	2 Bytes	ASCII HEX	0 - 255	
5	Pitch	4 Bytes	.0001	radians	Degrees = X * 57.29 * .0001
6	Roll	4 Bytes	.0001	radians	Degrees = X * 57.29 * .0001
8	END	1 Bytes	ASCII CHAR	*	
9	Check Sum	1 Bytes	ASCII CHAR	HEX	XOR from $ to *
Sample	$PCDIN,127257,03,00,7FFF,F79E,1211*28				
Notes					

$PCDIN,127251 – Rate of Turn

Field	Name	Length	Resolution	Value	Comment
1	Start	6 Bytes	ASCII CHAR	$PCDIN	
2	PGN ID	6 Bytes	ASCII CHAR	127251	
3	Num of Fields	2 Bytes	ASCII HEX	2	
4	Instance	2 Bytes	ASCII HEX	0 - 255	
5	Rate LB	4 Bytes	.0001	radians	Degrees = X * 57.29 * .0001
6	Rate HB	4 Bytes	.0001 * 65536	radians	Degrees = X * 57.29 * .0001
8	END	1 Bytes	ASCII CHAR	*	
9	Check Sum	1 Bytes	ASCII CHAR	HEX	XOR from $ to *
Sample	$PCDIN,127251,02,00,FFFF,F4C0*7D				
Notes	Total ROT = ROT HB * 65536 + ROT LB				

$PCDIN,127488 – Engine Data - Rapid Update

Field	Name	Length	Resolution	Value	Comment
1	Start	6 Bytes	ASCII CHAR	$PCDIN	
2	PGN ID	6 Bytes	ASCII CHAR	127488	
3	Num of Fields	2 Bytes	ASCII HEX	3	
4	Instance	2 Bytes	ASCII HEX	0 - 255	
5	RPM	4 Bytes	.25	Rev/sec	
6	BOOST	4 Bytes	.01	Pascal’s	
7	TRIM	4 Bytes	.01	Percent	
8	END	1 Bytes	ASCII CHAR	*	
9	Check Sum	1 Bytes	ASCII CHAR	HEX	XOR from $ to *
Sample	$PCDIN,127488,03,00,0CD0,FFFF,FF7F*25				
Notes					

$PCDIN,127489 – Engine Data - Dynamic Update

Field	Name	Length	Resolution	Value	Comment
1	Start	6 Bytes	ASCII CHAR	$PCDIN	
2	PGN ID	6 Bytes	ASCII CHAR	127489	
3	Num of Fields	2 Bytes	ASCII HEX	3	
4	Instance	2 Bytes	ASCII HEX	0 - 255	
5	OIL Pressure	4 Bytes	.01	Pascal’s	
6	OIL Temp	4 Bytes	.10	Kelvin	
7	Engine Temp	4 Bytes	.01	Kelvin	
8	Alternator Volts	4 Bytes	.01	Volts	
9	Fuel Rate	4 Bytes	.10	L/Hr	
10	Engine Hours LB	4 Bytes	.01	Seconds	
11	Engine Hours HB	4 Bytes	.01	Seconds	
12	Coolant Pressure	4 Bytes	.10	Pascal’s	
13	Fuel Pressure	4 Bytes	.01	Pascal’s	
14	END	1 Bytes	ASCII CHAR	*	
15	Check Sum	1 Bytes	ASCII CHAR	HEX	XOR from $ to *
Sample	$PCDIN,127489,09,01,2840,FFFF,6000,0000,00FF,03C0,0000,FFFF,FFFF*21				
Notes	Total Engine Hours = Engine Hours HB * 65536 + Engine Hours LB				

$PCDIN,127493 – Transmission Data - Dynamic Update

Field	Name	Length	Resolution	Value	Comment
1	Start	6 Bytes	ASCII CHAR	$PCDIN	
2	PGN ID	6 Bytes	ASCII CHAR	127493	
3	Num of Fields	2 Bytes	ASCII HEX	3	
4	Instance	2 Bytes	ASCII HEX	0 - 255	
5	Tran Pressure	4 Bytes	.01	Pascal’s	
6	Tran Temp	4 Bytes	.01	Kelvin	
7	Tran Status	4 Bytes			
8	END	1 Bytes	ASCII CHAR	*	
9	Check Sum	1 Bytes	ASCII CHAR	HEX	XOR from $ to *
Sample	$PCDIN,127493,03,01,0000,FFFF,0000*58				
Notes					

$PCDIN,127505 – Fluid Data - Dynamic Update

Field	Name	Length	Resolution	Value	Comment
1	Start	6 Bytes	ASCII CHAR	$PCDIN	
2	PGN ID	6 Bytes	ASCII CHAR	127505	
3	Num of Fields	2 Bytes	ASCII HEX	4	
4	Instance	2 Bytes	ASCII HEX	0 - 255	
5	Fluid Type	4 Bytes	-	0 - 15	Type of Fluid Tank
6	Level	4 Bytes	.01	Percent	
7	Capacity LB	4 Bytes	.10	Liters	
8	Capacity HB	4 Bytes	.10 * 65536	Liters	
9	END	1 Bytes	ASCII CHAR	*	
10	Check Sum	1 Bytes	ASCII CHAR	HEX	XOR from $ to *
Sample	$PCDIN,127505,04,00,0001,9500,0000,0000*7E				
Notes	Total Capacity = Capacity HB * 65536 + Capacity LB				

$PCDIN,127508 – Battery Status - Dynamic Update

Field	Name	Length	Resolution	Value	Comment
1	Start	6 Bytes	ASCII CHAR	$PCDIN	
2	PGN ID	6 Bytes	ASCII CHAR	127508	
3	Num of Fields	2 Bytes	ASCII HEX	3	
4	Instance	2 Bytes	ASCII HEX	0 - 255	
5	Battery Volts	4 Bytes	.01	Volts	
6	Battery Current	4 Bytes	.01	AMPS	
7	Battery Temp	4 Bytes	.01	Kelvin	
8	END	1 Bytes	ASCII CHAR	*	
9	Check Sum	1 Bytes	ASCII CHAR	HEX	XOR from $ to *
Sample	$PCDIN,127508,03,00,1245,3652,6300*7E				
Notes					

$PCDIN,127501 – Binary Switch Status - Dynamic Update

Field	Name	Length	Resolution	Value	Comment
1	Start	6 Bytes	ASCII CHAR	$PCDIN	
2	PGN ID	6 Bytes	ASCII CHAR	127508	
3	Num of Fields	2 Bytes	ASCII HEX	3	
4	Instance	2 Bytes	ASCII HEX	0 - 255	
5	Switch Status LB	4 Bytes	ON, OFF	Switch 0-7	00 = OFF, 01 = ON
6	Switch Status LB	4 Bytes	ON, OFF	Switch 8-15	10 = UNDEF, 11 = ERR
8	END	1 Bytes	ASCII CHAR	*	
9	Check Sum	1 Bytes	ASCII CHAR	HEX	XOR from $ to *
Sample	$PCDIN,127501,04,00,0000,FF00,FFFF,FFFF*78				
Notes					

$PCDIN,130306 – Wind Data

Field	Name	Length	Resolution	Value	Comment
1	Start	6 Bytes	ASCII CHAR	$PCDIN	
2	PGN ID	6 Bytes	ASCII CHAR	130306	
3	Num of Fields	2 Bytes	ASCII HEX	3	
4	Reference	2 Bytes	ASCII HEX	0	True
				1	-
				2	Apparent
				3	-
				4	Ref to Water
5	Speed	4 Bytes	.01	m/sec	Knots = X * 1.9438 * .01
6	Direction	4 Bytes	.0001	radians	Degrees = X * 57.29 * .0001
7	END	1 Bytes	ASCII CHAR	*	
8	Check Sum	1 Bytes	ASCII CHAR	HEX	XOR from $ to *
Sample	$PCDIN,130306,02,02,0005,F4C0*7F				
Notes					

$PCDIN,130311 – Environmental Data

Field	Name	Length	Resolution	Value	Comment
1	Start	6 Bytes	ASCII CHAR	$PCDIN	
2	PGN ID	6 Bytes	ASCII CHAR	130311	
3	Num of Fields	2 Bytes	ASCII HEX	3	
4	Instance	2 Bytes	ASCII HEX	0 -255	
5	Air Temp	4 Bytes	.01	Kelvin	
6	Humidity	4 Bytes	.01	%	
7	Barometric	4 Bytes	.01	inHg	
8	END	1 Bytes	ASCII CHAR	*	
9	Check Sum	1 Bytes	ASCII CHAR	HEX	XOR from $ to *
Sample	$PCDIN,130311,03,C1,7549,7FFF,03EF*5E				
Notes					

$PCDIN,130312 – Temperature Data

Field	Name	Length	Resolution	Value	Comment
1	Start	6 Bytes	ASCII CHAR	$PCDIN	
2	PGN ID	6 Bytes	ASCII CHAR	130312	
3	Num of Fields	2 Bytes	ASCII HEX	3	
4	Instance	2 Bytes	ASCII HEX	0 -255	
5	Temp Type	4 Bytes		0-255	
6	Temp	4 Bytes	.01	Kelvin	
6	Set Temp	4 Bytes	.01	Kelvin	
8	END	1 Bytes	ASCII CHAR	*	
9	Check Sum	1 Bytes	ASCII CHAR	HEX	XOR from $ to *
Sample	$PCDIN,130312,03,00,0086,6D00,03EF*2D				
Notes					

$PCDIN,129025 –Position Data - Rapid

Field	Name	Length	Resolution	Value	Comment
1	Start	6 Bytes	ASCII CHAR	$PCDIN	
2	PGN ID	6 Bytes	ASCII CHAR	129025	
3	Num of Fields	2 Bytes	ASCII HEX	4	
4	Type	2 Bytes	ASCII HEX	0	
6	Latitude LB	4 Bytes	.0000001	degrees	
7	Latitude HB	4 Bytes	.0000001	degrees	
8	Longitude LB	4 Bytes	.0000001	degrees	
9	Longitude HB	4 Bytes	.0000001	degrees	
10	END	1 Bytes	ASCII CHAR	*	
11	Check Sum	1 Bytes	ASCII CHAR	HEX	XOR from $ to *
Sample	$PCDIN,129025,04,00,A2C9,190A,2C81,B603*7A				
Notes	Degrees = (X HB * 65536 + X LB) * .0000001				

$PCDIN,130323 – Weather Station Location Data

Field	Name	Length	Resolution	Value	Comment
1	Start	6 Bytes	ASCII CHAR	$PCDIN	
2	PGN ID	6 Bytes	ASCII CHAR	130323	
3	Num of Fields	2 Bytes	ASCII HEX	4	
4	Type	2 Bytes	ASCII HEX	0	
5	Latitude LB	4 Bytes	.0000001	degrees	
6	Latitude HB	4 Bytes	.0000001	degrees	
7	Longitude LB	4 Bytes	.0000001	degrees	
8	Longitude HB	4 Bytes	.0000001	degrees	
9	END	1 Bytes	ASCII CHAR	*	
10	Check Sum	1 Bytes	ASCII CHAR	HEX	XOR from $ to *
Sample	$PCDIN,130323,07,00,FFFF,FFFF,FFFF,FFFF,7FFF,FFFF,7FFF*57				
Notes	Degrees = (X HB * 65536 + X LB) * .0000001				

$PCDIN,129026 – SOG and COG Rapid Update

Field	Name	Length	Resolution	Value	Comment
1	Start	6 Bytes	ASCII CHAR	$PCDIN	
2	PGN ID	6 Bytes	ASCII CHAR	129026	
3	Num of Fields	2 Bytes	ASCII HEX	3	
4	Reference	2 Bytes	ASCII HEX	0	True
				1	-
				2	Apparent
				3	-
				4	Ref to Water
5	Speed	4 Bytes	.01	m/sec	Knots = X * 1.9438 * .01
6	Direction	4 Bytes	.0001	radians	Degrees = X * 57.29 * .0001
7	END	1 Bytes	ASCII CHAR	*	
8	Check Sum	1 Bytes	ASCII CHAR	HEX	XOR from $ to *
Sample	$PCDIN,129026,02,07,FFFF,FFFF*77				
Notes					

Protocol Polling

The SeaSmart.Net Protocol is embedded in a simple HTML document created dynamically from a Browser GET request. One simple method is to use the XMLHttpRequest Object to issue a GET of the target HTML file from the server.

```
var datafile = window.location.href.substring(0,
     window.location.href.lastIndexOf("/") + 1) +
     "NMEAN2KData.htm?";

objXml = new XMLHttpRequest();


objXml.open("GET",datafile , true);
```

This will create a new Xml Object and copy the contents of the NMEAN2KData.htm file located in the same directory as the calling Web Page.

Use the objXml.onreadystatechange event to determine when the new data is ready.

```
objXml .onreadystatechange = function()
{
      if(objXml.readyState == 4)
      {
            if(objXml.status == 200)
            {
                  mydata= objXml .responseText;
            }
      }
}
```

The **mydata** object will now contain a copy of the NMEAN2KData.htm file

Protocol Parsing

The SeaSmart.Net Protocol is easily parsed in JavaScript by using the Java .split function

```
            mydata= objXml .responseText;

            mySubStrings = mydata.split("$PCDIN");
```

This will create a variable number of array elements based on the starting “$PCDIN” character that correspond to each of the received PGN sentences.

From there, each PGN can be decoded by fixed reference to character position in the string.

Start by extracting the PGN number to determine the number and types of data fields.

```
for( myIndex = 0; myIndex < myArrayLength; myIndex++)
{

// First extract the six character PGN number
      myHexStr = mySubStrings[myIndex];
      myHexStr = myHexStr.substr(0,6) ;
      myPGN = parseInt(myHexStr);

      // Then Parse based on PGN Number
      if(myPGN == 130311) // Environ Data
      {
            myDataLabels[myIndex]="BARO";
            myHexStr = mySubStrings[myIndex];
            myHexStr = myHexStr.substr(18,4) ;
            myHexStr = "0x" + myHexStr;
            myPGNValue = parseInt(myHexStr);
            myHexStr =((myPGNValue * 0.0295229) );

            myDataValues[myIndex]=myHexStr.toFixed(2) ;
            myDialIndexes[myIndex]=Math.floor(((myPGNValue*0.0295229) -28)*64 );

            myDataLabels[myIndex]="AIR TEMP";
            myHexStr = mySubStrings[myIndex];
            myHexStr = myHexStr.substr(13,4) ;
            myHexStr = "0x" + myHexStr;
            myPGNValue = parseInt(myHexStr);
            myDataValues[myIndex]=Math.floor(((myPGNValue * 0.018) - 459));

            myDialIndexes[myIndex] =Math.floor(((myPGNValue * 0.018) - 459)*2);

      }

}
```

Refer to the SeaSmart.Net Protocol descriptions for definitions on each of the received PGN data structures

The resulting data variables can be easily written to the target Browser Window using a variety of methods. The document Object is one option

```
document.getElementById("dataLabel7").innerHTML = myDataLabels[myIndex];
document.getElementById("dialValue7").innerHTML = myDataValues[myIndex];
```

HTTP POST Protocol

SeaSmart.Net can support forwarding data to external Web Servers on the Local or Global network by using the HTTP POST Protocol. When this option is enabled, a TCP connection is made to the target IP address (or Host Name if DNS is available) and incoming NMEA 2000 data transferred in blocks using the HTTP POST once a second. The TCP connection is maintained as long as new data is available within the interval.

The SeaSmart.Net module is configured with the HOST IP Address or HOST NAME and the target file to handle the POST which is usually an Active Server Script (ASP) or CGI Script depending on the Server Platform type.

A typical POST message may look like:

```
POST /XPORTN2KWrite.asp HTTP/1.1
Host: 192.168.0.1:80
Content-Length: 361
Content-type: application/x-www-form-urlencoded

Name=
$PCDIN,000000,02,00,06BC,BA6D*08
$PCDIN,127251,02,00,FFFC,7FFF*08
$PCDIN,127250,03,05,F1C0,0000,F5D8*5E
$PCDIN,127489,09,01,2840,FFFF,6000,0000,00FF,03C0,0000,FFFF,FFFF*21
$PCDIN,127505,02,00,0000,6200*7E
$PCDIN,127505,02,01,0000,6200*7F
$PCDIN,127493,03,00,0000,FFFF,0000*59
$PCDIN,127493,03,
1,0000,FFFF,0000*58
$PCDIN,130306,02,02,0019,104B*04
```

Where XPORTN2KWrite.asp is the handler and 192.168.0.1:80 is the host IP address/Port.

The number of PGNs transferred in each POST is dependent on the number received within the one second interval. The hosting server is then responsible for processing the data and passing it along. In most cases, the .ASP script will just write it to a local .htm file so that Browser Apps can access it.

The resulting HTML file would be.

```
<html>
<body>
$PCDIN,127489,09,01,2840,FFFF,6000,0000,00FF,03C0,0000,FFFF,FFFF*21<BR>
$PCDIN,127505,02,00,0000,6200*7E<BR>
$PCDIN,127505,02,01,0000,6200*7F<BR>
$PCDIN,127493,03,00,0000,FFFF,0000*59<BR>
$PCDIN,130312,03,01,0087,6D00,FFFF*2D<BR>
$PCDIN,000000,02,00,06F5,7568*07<BR>
$PCDIN,127488,03,01,0000,FFFF,FF7F*23<BR>
$PCDIN,127488,03,02,0000,FFFF,FF7F*20<BR>
$PCDIN,126992,03,05,3AB7,9E10,16F4*5E<BR> </body>
</html>
```

Sample .ASP script

The following sample .ASP file simply takes the incoming post data and writes out to existing file while replacing the “*” with “<BR>” to be compatible with HTML syntax.

```
<html>

<head>
<meta name="GENERATOR" content="Microsoft FrontPage 5.0">
<title>Main</title>

<meta name="Microsoft Border" content="none">
</head>

<body>

<%
Dim name, oldN2KData
' Declare our vaiables
Dim objFSO, objCountFile  ' object vars for FSO and File
Dim strCountFileName      ' filename of count text file
Dim iCount                ' count variable

Dim objOldFSO, objOldDataFile  ' object vars for FSO and File
Dim strOldDataFileName      ' filename of count text file
Dim iOldDataCount           ' count variable

Dim myHTTPHeader
Dim myHTTPFooter

myHTTPHeader = "<HTML><head><Title> SeaGauge Web Log File </Title></head><Body>"
myHTTPFooter = "</Body></HTML>"


name = Request.Form("Name")


' Old N2K data File Name
strOldDataFileName = Server.MapPath("OldNMEAN2KData.txt")

' Create FileSystemObject to deal with file access
Set objOldFSO = Server.CreateObject("Scripting.FileSystemObject")

' Open Old file and get a text stream to new one
Set objOldDataFile = objOldFSO.OpenTextFile(strOldDataFileName, 1 )

  ' Read from the file.
   If objOldDataFile.AtEndOfStream Then
       oldN2KData = ""
   Else
       oldN2KData = objOldDataFile.ReadAll
   End If

objOldDataFile.Close
```

```
'  file's filename
strCountFileName = Server.MapPath("NMEAN2KData.htm")

' Create FileSystemObject to deal with file access
Set objFSO = Server.CreateObject("Scripting.FileSystemObject")

' Overwrite existing file and get a text stream to new one
Set objCountFile = objFSO.CreateTextFile(strCountFileName, True)

' Write updated count
objCountFile.Write myHTTPHeader

oldN2KData = Replace(oldN2KData,"*","<BR>")

' Write Old Data
objCountFile.WriteLine oldN2KData

oldN2KData = Replace(name,"*","<BR>")
' Write new data
objCountFile.WriteLine oldN2KData

' Write updated count
objCountFile.Write myHTTPFooter

' Close the file and destroy the object
objCountFile.Close
Set objCountFile = Nothing

' Open Old file and get a text stream to new one
Set objOldDataFile   = objOldFSO.OpenTextFile(strOldDataFileName, 2, True)

' Write new data
objOldDataFile.WriteLine name

objOldDataFile.Close
Set objOldDataFile = Nothing

' Destroy the FSO object
Set objFSO = Nothing
Set objOldFSO = Nothing

%>

</body>
</html>
```

Embedded Server vs External Server

There is a small difference between Web Pages designed for the Embedded Web Server and those that Run on an External Web Server.

The Embedded Server uses a special form of CGI Script to capture the translated data from the NMEA 2000 backbone and renders it for further on processing by Browser Apps that may call for it. Since the Embedded Server has no mass storage device, it keeps the incoming data temporally in memory until the receive buffers are filled, after which it will dump the oldest data as newer data arrives. With a buffer size of over 32 Kbytes, that is over 10 minutes of traffic on a heavily loaded bus.

Therefore, the Embedded Server CGI Script will service Browser Requests from Memory and not file Storage.

The External Web Server on the other hand usually has plenty of mass storage so that data can be written to files and files overwritten as new data arrives.

With this in mind, the Get Request for the Embedded Server is slightly different then the External Server.

Embedded Server GET Request file Name

```
var datafile = window.location.href.substring(0,
    window.location.href.lastIndexOf("/") + 1) +
    "GetNMEANData?";
```

Calls CGI script to grab data directly from NMEA 2000 Bus

External Server GET Request file Name

```
var datafile = window.location.href.substring(0,
    window.location.href.lastIndexOf("/") + 1) +
    "GetNMEANData.htm?";
```

Gets contents of stored HTML file created by a HTTP POST

Other then how the GET REQUEST is called and processed, all other elements of the .HTML document remain the same for both environments with support for Dynamic HTML and JavaScript.

One Year Warranty

"We", "our", or "us" refers to Chetco Digital Instruments, the manufacturer of this product. "You" or "your" refers to the first person who purchases this product as a consumer item for personal, family, or household use.

We warrant this product against defects or malfunctions in materials and workmanship, and against failure to conform to this product's written specifications, all for one year (1) from the date of original purchase by you. WE MAKE NO OTHER EXPRESS WARRANTYOR REPRESENTATION OF ANY KIND WHATSOEVER CONCERNING THIS PRODUCT. Your remedies under this warranty will be available so long as you can show in a reasonable manner that any defect or malfunction in materials or workmanship, or any nonconformity with the product's written specifications, occurred within one year from the date of your original purchase, which must be substantiated by a dated sales receipt or sales slip. Any such defect, malfunction, or non-conformity which occurs within one year from your original purchase date will either be repaired without charge or be replaced with a new product identical or reasonably equivalent to this product, at our option, within a reasonable time after our receipt of the product. If such defect, malfunction, or non-conformity remains after a reasonable number of attempts to repair by us, you may elect to obtain without charge a replacement of the product or a refund for the product. THIS REPAIR, REPLACEMENT, OR REFUND (AS JUST DESCRIBED) IS THE EXCLUSIVE REMEDY AVAILABLE TO YOU AGAINST US FOR ANY DEFECT, MALFUNCTION, OR NON-CONFORMITY CONCERNING THE PRODUCT OR FOR ANY LOSS OR DAMAGE RESULTING FROM ANY OTHER CAUSE WHATSOEVER. WE WILL NOT UNDER ANY CIRCUMSTANCES BE LIABLE TO ANYONE FOR ANY SPECIAL, CONSEQUENTIAL, INCIDENTAL, OR OTHER INDIRECT DAMAGE OF ANY KIND.

Some states do not allow the exclusion or limitation of incidental or consequential damages, so the above limitations or exclusions may not apply to you.

This warranty does NOT apply in the following circumstances: (1) when the product has been serviced or repaired by anyone other than us, (2) when the product has been connected, installed, combined, altered, adjusted, or handled in a manner other than according to the instructions furnished with the product, (3) when any serial number has been effaced, altered, or removed, or (4) when any defect, problem, loss, or damage has resulted from any accident, misuse, negligence, or carelessness, or from any failure to provide reasonable and necessary maintenance in accordance with the instructions of the owner's manual for the product.

We reserve the right to make changes or improvements in our products from time to time without incurring the obligation to install such improvements or changes on equipment or items previously manufactured.

This warranty gives you specific legal rights and you may also have other rights which may vary from state to state.

REMINDER: You must retain the sales slip or sales receipt proving the date of your original purchase in case warranty service is ever required.

Chetco Digital Instruments, INC.
14377 Highway 101 South Unit C
Harbor, OREGON 97415
541-661-2051

VDASH SOFTWARE LICENSE AGREEMENT

THIS IS A LEGAL AGREEMENT BETWEEN THE END-USER WHOFIRST PURCHASES THIS PRODUCT AS A CONSUMER ITEM FORPERSONAL, FAMILY, OR HOUSEHOLD USE ("YOU") AND CHETCO DIGITAL INSTRUMENTS, INC., THE MANUFACTURER OF THIS PRODUCT. ("WE", "OUR", OR "US"). USING THE PRODUCT ACCOMPANIED BY THIS LICENSE AGREEMENT CONSTITUTES ACCEPTANCE OF THESE TERMS AND CONDITIONS.

1. This License Agreement applies to the microcode and one or more lookup tables that your product may contain. We refer to these singly as a "SOFTWARE".

2. The SOFTWARE that your product may contain are licensed, not sold. We grant to you the nonexclusive, non-assignable right to use these SOFTWARE for monitoring sensor/sender data, but only as long as you comply with the terms and conditions of this License Agreement. We reserve the right to terminate this license if you violate any aspect of this License Agreement.

3. The SOFTWARE housed in your product are protected by the copyright notices appearing on the product or its screen(s). You may NOT modify, adapt, translate, reverse engineer, decompile, disassemble, rent, lease, or resell any SOFTWARE, and you may NOT create derivative works based upon any SOFTWARE or its contents.. Any unauthorized reproduction, use, or transfer of a SOFTWARE may be a crime and may subject you to damages and attorney fees.

4. This License Agreement will terminate immediately without prior notice from us if you fail to comply with or violate any of the provisions of this Agreement. Upon termination, you will promptly return all products containing one or more SOFTWARE to us.

5. Prices and programs are subject to change without notice.

6. This License Agreement shall be governed by the laws of the State of Oregon and comprises the complete and exclusive understanding between you and us concerning the above subject matter.

How to Obtain Service

We back your investment in quality products with quick, expert service and genuine replacement parts. If you're in the United States and you have questions, please contact the Factory Customer Service Department using our number listed below. You must send the unit to the factory for warranty service or repair. Please call the factory before sending the unit. You will be asked for your unit's serial number (shown above). Use the following number:

541-469-4783

U.S.A.only. Monday through Friday, except holidays.

Your unit is covered by a full one-year warranty. (See inside for complete warranty details.) If your unit fails and the failure is not covered by the original warranty, Chetco Digital Instruments has a flat-rate repair policy that covers your unit and accessories packed with the unit at the factory. There is a 180-day warranty on all non-warranty repairs from the factory, which is similar to the original warranty, but is for 180 days rather than one year. For further details, please call us at the above number.

Remember, non-warranty repairs are subject to Chetco Digital Instruments published flat rate charges and 180-day warranty.

CHETCO DIGITAL INSTRUMENTS, INC

BOX 5359

Brookings, OR 97415

541-469-4783

http://www.chetcodigital.com