

# Ethernet/RS232 Protocol for NAD Products v2.03

April 27, 2012

## Introduction

The purpose of this document is to define the protocol used to communicate to NAD products that have either an RS-232 port, Ethernet, or both. This document is defining the second generation NAD protocol V2.X. The new protocol, V2.X, is all ASCII based, with the goal of making it easier to use. This protocol allows NAD products to be controlled by a PC or any other device which has either an RS-232 port or Ethernet connectivity.

## RS-232 Specifications

The NAD product should use a standard DB-9 female connector with the following pinout:

| Pin | Function      |
|-----|---------------|
| 2   | Transmit Data |
| 3   | Receive Data  |
| 5   | Signal Ground |

This pinout allows the NAD product to be connected to a PC with a standard straight throw serial cable.

All communication should be done at a rate of 115200 bps with 8 data bits, 1 stop bit and no parity bits. No flow control should be performed.

## Ethernet Specifications

- You can connect via a raw TCP/IP socket on port 23. Port 23 is the standard port for telnet so you can use most telnet clients to communicate with an Ethernet connected device. The commands used to control the device are 100% identical to the RS232 commands.
- By default the IP address of the device gets assigned via DHCP. On our AVR's you can see the current address in the System Info screen (press Source Down + Source Up).
- The IP address is also discoverable over the network using DNS-SD over mDNS (also called Bonjour by Apple).

## Data Format

All commands sent to the device and responses sent back have the following basic format:

<Prefix> . <Variable> <Operator> <Value> <CR> (and/or) <LF>

**Note 1:** All data is both sent and received as ASCII. Therefore, any command can be sent and received using any terminal program capable of communicating with a computer's serial ports.

**Note 2:** It is a good idea to also send a <CR>(0x0D) and/or a <LF>(0x0A) character at the beginning of any message to get rid of any unwanted data (ie. noise) that may have been sent to the receiving device.

<prefix>                      Used to help group the different variables of a unit.

<variable>                    Name of the variable to be queried or set.

<operator>                    For non-read only <variables>, any 1 of 3 <operators> may be used: + / - / =.

The “+” and “-“ <operator> act as a toggle and will cycle through all the valid <values> for the specified <variable>.

The “=” operator can be used to set a <variable> to a specific <value>, if the <value> is not valid for the <variable> being set, it will be ignored.

To query any <variable>, the “?” operator is used. The response will always be returned using the “=” operator.

<value>            The ASCII representation of the <value> to be assigned to the <variable>

Any data received which does not follow this format should be ignored. Also, invalid <values> will be ignored.

### **Command Set**

The different <prefixes>, <variables>, and <values> are documented in a table within a different document specific to the NAD product.

Additions to the <prefix>, and <variable> lists will be done on an ongoing basis as new functions are needed for new products.

### **Example 1**

This example will describe exactly what is sent when the Identification of the unit is queried.

The above is accomplished by sending the following ASCII character sequence:

Command: Main.Model?<CR>

If the identification string of the unit was “T775” then the response to the above command would be:

Response: Main.Model=T775<CR>

### **Example 2**

This example will describe how to set the volume to a level of -3 dB.

Command: Main.Volume=-3<CR>

If the volume is successfully set to -3 dB, then the response sent back would be:

Response: Main.Volume=-3<CR>

### **Example 3**

This example will describe how to increment the volume. This will take the existing <value> and increase it by 1.

Command: Main.Volume+<CR>

If the volume is successfully increment to -2 dB, then the response sent back would be:

Response: Main.Volume=-2<CR>