NetLinx Module Interface Specification

for a

Denon
DN-V300 DVD Player

Professional Control Solutions LLC • 3804 Parchment Circle • Henrico, VA • 23233-7034
804-677-6794 • info@professionalcontrolsolutions.com
www.professionalcontrolsolutions.com

TABLE OF CONTENTS

LIST OF TABLES

Revision History

Date	Initials	Comments
07/22/05	DC	Initial Release v1.0

Professional Control Solutions LLC • 3804 Parchment Circle • Henrico, VA • 23233-7034
804-677-6794 • info@professionalcontrolsolutions.com
www.professionalcontrolsolutions.com

Overview

The AMX module communicates to the Denon DN-V300 at 9600, N, 8, 1; without hardware handshaking. The communication cable is a db-9 female with the following connections:

Denon	NetLinx
2 Rx	2 Rx
3 Tx	3 Tx
5 Gnd	5 Gnd (or 1 on NXI Phoenix)

The communication module is instantiated/called by adding the following line of code:

```
DEFINE_MODULE 'DenonComm' comm._code(virtual_name, real_name)
```

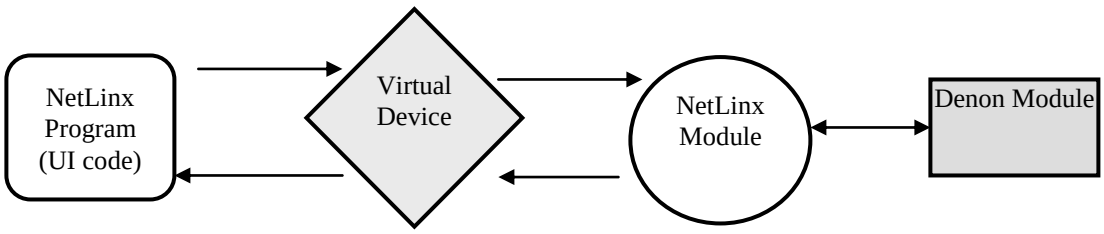
The Sample user interface module is instantiated/called by adding the following line of code:

```
DEFINE_MODULE 'DenonUI' ui_code(virtual_name, dvTPArray, nBUTTON_Array)
```

This document will define the common NetLinx module interface for a DENON DN-V300 player. Obviously there will always be features one system supports that another does not (or cannot). The model is not designed to be static. It is designed to be ever-growing while always supporting backwards compatibility. It is up to the programmer of each module to adhere to the model and to find the best way to fit the protocol of a piece of equipment to the model.

For features that are not part of the model the programmer may support additional commands that extend beyond the model to support those features. This is desirable because manufacturers want to expose the features of their system that make them unique and differentiate them from their competitors. Exposing control for those features should be done even if they are not part of the model. In this module, the 'PASSTHRU=' command provides this functionality.

The following diagram gives a graphical view of the interface between the interface code and the NetLinx module.



Command Interface

The interface code will control the DENON DN-V300 player via command events (NetLinx command *send_command)*. These commands will be sent to the module to affect control. Below are the commands supported.

Command	Description
ANGLE=<value>	Change the viewing angle of a DENON DN-V300. <value> = + ANGLE=+
CHAPTER=<value>:<title>	Select a specific chapter to play. If either variable is null, the command defaults to one for both. <value> : n = select chapter n, where n is the unique identifier for the chapter. <title> : title for chapter select CHAPTER=1:12 Note: If either the <value> or <title> parameters are not used or 0, then chapter 1 and title 1 are used by default.
CURSOR=<value>	Moves cursor in a particular direction. <value> : UP = moves cursor up : DOWN = moves cursor down : LEFT = moves cursor left : RIGHT = moves cursor right CURSOR=UP CURSOR=RIGHT
DEBUG=<state>	Set the state of the debugging flag. <state>: 0 = off 1 = on 'DEBUG=1'
ENTER=<value>	Issues the enter command <value> = + ENTER=+
MENU=<value>	Display the menu. <value> = + MENU=+

PASSTHRU=<string>	Allows user the capability of sending commands directly to whatever unit is attached without processing by the NetLinx module. User must be aware of the protocol implemented by the unit to use this command. This gives the user access to features which may not be directly supported by the module. The communication does not add automatically any characters to the passthru string. For more detail please read the Adding Functions to Modules section at the end of this document. <string> : string to send to unit PASSTHRU=THIS IS A COMMAND PASSTHRU=RESET
RETURN=<value>	Return to previous menu. <value> = + RETURN=+
SETUP=<value>	Displays the setup menu <value> = + SETUP=+
SUBTITLE=<value>	Turn off, on, or toggle subtitles. <value> : T = toggle SUBTITLE=T
TRACK?	Report current playing track 'TRACK?'
TRANSPORT=<func>	Change the current transport state. <func> : PLAY = play STOP = stop PAUSE = pause PREVIOUS = previous chapter NEXT = next chapter SEARCH+ = search forward SEARCH- = search backward TRANSPORT=PLAY
TRANSPORT?	Query for the current transport state. TRANSPORT?
TRAY=<value>	Open or close the disc tray. <value> : T toggle state of tray. TRAY=T
VERSION?	Query for the current version number of the NetLinx module. VERSION?

Table 1 – Send Command Definitions

String Feedback

The NetLinx module will provide feedback to the interface code for DENON DN-V300 player changes via string events. Below are the strings supported.

String	Description
DEBUG=<state>	State of the debug flag. Non-solicited feedback. <state>: 0 = off 1 = on 'DEBUG=1'
ERROR=<cmd>:<arg>	Error occurred. <cmd>: command that caused error <arg>: associated faulty argument to command (@#$jvdf8*) AUDIO: <arg> ANGLE: <arg> CURSOR: <arg> NUMPAD: <arg> SCAN: <arg> STEP: <arg> TRANSPORT: <arg> TRAY: <arg> ERROR=TRAY:BOLOGNA
TRACK=<id>	Report the track that is currently playing. <id> : Unique identifier of the track TRACK=12 TRACK=328631
TRANSPORT=<func>	Reports the current transport state for a zone. <func> : PLAY = play STOP = stop PAUSE = pause TRANSPORT=PLAY TRANSPORT=STOP
VERSION=<value>	Reports the current version number of the NetLinx module. <value> : current version number in xx.yy format VERSION=1.06

Table 2 - String Feedback Definitions

Adding Functions to Modules

Commands to the device

This module supplies a mechanism to allow additional device features to be added to software using the module. This is the PASSTHRU command, which allows protocol strings to be passed through the module. The device-specific protocol must be known in order to use this feature.

As an example, suppose that a module for a projector has not implemented the 'white balance adjustment' feature. The command that the projector protocol requires is 03H, 10H, 05H, 14H, followed by a checksum. The documentation for the PASSTHRU command specifies that the module will automatically generate the checksum. In this case, the following string should be sent from the UI code to implement 'white balance adjustment'.

```
send_string vdvDevice,'"PASSTHRU=',$03,$10,$05,$14"
```

The reason to use PASSTHRU instead of sending a protocol string directly to the device port is that the device may require command queuing, calculation of checksums, or other internal processing, which would not be done if the string was sent directly. Because of this, it is best to filter all communication TO the device through the module. (The module documentation will indicate any processing that will be automatically done to the PASSTHRU string like checksum calculation.)

Additional Feedback from the device

The module documentation indicates what feedback is provided. If additional feedback is required, a CREATE_BUFFER for the device must be implemented in the user code to process the strings from the device manually. Note that the module will still be processing the response strings independently and sending the interpreted feedback up to the user code.