NetLinx Module Interface Specification

for a

Clock Audio CDT100

4-Channel Dante Microphone Aggregator

TABLE OF CONTENTS

Introduction 3
Overview 3
Implementation 3
Port Mapping 4
Channels 5
Command Control 6
Command Feedback 8
Programming Notes 9
Adding Functions to Modules 9

Revision History

Date	Initials	Comments
01-23-15	DGG	Initial release (CCI)

Introduction

This is a reference manual to describe the interface provided between an AMX NetLinx system and a Clock Audio CDT100. The CDT100 supports IP communication via UDP. The interface was tested using version 99.5 of the firmware.

Overview

The COMM module translates between the standard interface described below and the CDT100 UDP protocol. It parses the buffer for responses from the CDT100, sends strings to control the CDT100, and receives commands from the UI module or telnet sessions.

A sample touch panel file and demo program are provided in the module package. The demo program uses the standard interface described below and parses the command responses for feedback. These are not intended to cover every possible application, but can be expanded as needed by a dealer to meet the requirements of a particular installation.

Some functionality in the device interface may not be implemented in the API interface. In cases where device functions are desired but not API-supported, the PASSTHRU command may be used to send any and all device-protocol commands to the device. See the PASSTHRU command and the Adding Functions to Modules section for more information.

Implementation

To interface to the AMX Clock Audio CDT100 module, the programmer must perform the following steps:

1. Define the device ID for the CDT100 that will be controlled.
2. Define the NetLinx virtual device ID that the CDT100 COMM module will use to communicate with the main program and User Interface.
3. If a touch panel interface is desired, a touch panel file (Clock Audio-CDT100_Demo_Panel.tp4) and demo program (ClockAudio_CDT100_Demo.axs) have been created for testing.
4. The CDT100 module must be included in the program with a DEFINE_MODULE command. This command starts execution of the module and passes in the following key information: the device ID of the CDT100, and the virtual device ID for communicating to the main program.
5. Issue the following SEND_COMMAND's to the virtual device in the virtual device ONLINE event:
    a. PROPERTY-IP_ADDRESS,<IP Address of the CDT100>
    b. PROPERTY_IP_PORT,<UDP Port used to communicate with the CDT100>
    c. REINIT

An example of how to do this is shown below.

```
DEFINE_DEVICE

dvCDT100            =      0:3:0
vdvCDT100           = 33001:1:0
dvTP_CH32_In        = 10003:1:0
dvTP_CH32_Out       = 10003:2:0
dvTP_Arm_C          = 10003:3:0
dvTP_Phantom        = 10003:4:0
dvTP_Preset         = 10003:5:0

DEFINE_MODULE 'ClockAudio_CDT100_Module' ClockAudioCDT100(vdvCDT100, dvCDT100)

DATA_EVENT[vdvCDT100]
{
   ONLINE:
   {
      SEND_COMMAND vdvCDT100,"PROPERTY-IP_ADDRESS,192.168.187.219"
      SEND_COMMAND vdvCDT100,"PROPERTY-IP_PORT,49494"
      SEND_COMMAND vdvCDT100,"REINIT"
   }
}
```

Upon re-initialization the AMX Comm module will communicate with the CDT100 and information will be exchanged.

Port Mapping

This module uses multiple virtual device ports in order distinguish events for one channel on the CDT100 from another (i.e. virtual device port 1 = Channel 1, virtual device port 2 = Channel 2, etc.).

Virtual Device	Button Events	Channels	Levels	Control	Feedback
33001:1:0 – 33001:4:0	1	1, 2, 3, 4, 5, 6, 251, 252	None	See tables	See tables

Table 1 - Port Mapping

Channels

The UI module controls the CDT100 via channel events (NetLinx commands *on, pulse and off*) sent to the COMM module for the appropriate virtual device (1 – 4). The channels supported by the COMM module are listed below. These channels are associated with the virtual device(s) and are independent of the channels associated with the touch panel device.

Channel	Description
1	ON: Set CH32 Red LED On - used for feedback also OFF: Set CH32 Red LED Off **Send to virtual device 1 - 4**
2	ON: Set CH32 Green LED On - used for feedback also OFF: Set CH32 Green LED Off **Send to virtual device 1 - 4**
3	ON: Set ARM-C Up - used for feedback also OFF: Set ARM-C Down **Send to virtual device 1 only**
4	ON: Set Phantom Power On - used for feedback also OFF: Set Phantom Power Off **Send to virtual device 1 - 4**
5	PULSE: Save Preset 0 **Send to virtual device 1 only**
6	PULSE: Load Preset 0 **Send to virtual device 1 only**
251	ON: Device is Online - used for feedback only OFF: Device is not Online **Feedback from virtual device 1 only**
252	ON: Data is Initialized - use for feedback only OFF: Data is not Initialized **Feedback from virtual device 1 only**

Table 2 - Virtual Device Channel Events

Command Control

The UI module controls the CDT100 via command events (NetLinx command *send_command*) sent to the COMM module on virtual device 1 only. The commands supported by the COMM module are listed below.

Command	Description
DEBUG-<value>	Set the state of debugging messages in the UI module and the Comm. module. **Send to virtual device 1 only** <value> : 1 = set only error messages on 2 = set error and warning messages on 3 = set error, warning & info messages on 4 = set all messages on DEBUG-1
PASSTHRU-<string>	Allows user the capability of sending commands directly to the CDT100 with minimal processing by the Duet module. User must be aware of the protocol implemented by the unit to use this command. This gives the user access to features that may not be directly supported by the module. For more information, see the "Adding Functions to Modules" section below. **Send to virtual device 1 only** <string> : string to send to unit PASSTHRU-<string>
PROPERTY-<key>,<value>	Set the value of property <key> to <value>. This must be followed by the REINIT command to take effect. These values are not initialized by default. **Send to virtual device 1 only** <key> : IP_ADDRESS <value> : string representing the IP address of the CDT100 you wish to connect to. valid values: 4-octet IP address <key> : IP_PORT <value> : string representing the UDP port number of the CDT100 you wish to connect on. valid values: 1 - 65535 PROPERTY-IP_ADDRESS,192.168.187.219 PROPERTY-IP_PORT,49494

REINIT	Re-initializes the communication link and data. **Note:** This command deletes any messages waiting to go out to the device. **Send to virtual device 1 only** REINIT

Table 3 – Send Command Definitions

Command Feedback

The COMM module provides feedback to the User Interface module via command events from virtual device 1 only. The responses available are listed below.

Response	Description
IP_ADDRESS,<value>	Returns the IP Address of the CDT100 that has been passed in to the module. **Feedback from virtual device 1 only** IP_ADDRESS,192.168.187.219
PRESET_SAVED,<value>	Message returned when a preset has been saved. **Note:** As of this writing, only preset 0 is available. **Feedback from virtual device 1 only** PRESET_SAVED,0
SWITCH_ADDRESS,<value>	Returns the Switch Address of the CDT100 as set on the front panel of the unit itself. **Feedback from virtual device 1 only** SWITCH_ADDRESS,A

Table 4 – Command Feedback Definitions

Programming Notes

- The following SEND_COMMAND's must be issued to the virtual device in the virtual device ONLINE event for communication to occur:
    a. PROPERTY-IP_ADDRESS,<IP Address of the CDT100>
    b. PROPERTY_IP_PORT,<UDP Port used to communicate with the CDT100>
    c. REINIT

- The module will automatically register for asynchronous feedback from the device upon re-initialization.

- The module utilizes a "Heartbeat" to maintain communication with the CDT100. This heartbeat will begin once preliminary initialization is complete. Should a heartbeat response not be received, the module will attempt to send the heartbeat two more times in succession. If neither of these responses is received, the module will consider communication with the CDT100 to be broken, will clear all existing data and output signals, and will attempt to re-establish communication with the CDT100 every 10 seconds. Once communication has been re-establish, the module will automatically re-initialize.

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.

Example:
SEND_COMMAND vdvDevice, "'PASSTHRU-QUERY',$0D"

The reason to use 'PASSTHRU-' instead of sending a protocol string directly to the device port is that the device may require command queuing, 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.