

Duet Module Interface Specification

for

**d&b audiotechnik**

**D20 / D80 / 10D / 30D Amplifiers**



Picture: D80

TABLE OF CONTENTS

Revision History 2

Introduction 3

Overview 3

Implementation 3

Port Mapping 6

Channels 7

Levels 9

Command Control 10

Command Feedback 12

Device Notes 13

Programming Notes 14

Adding Functions to Modules 15

Commands to the device 15

Responses from the device 15

Log messages 16

## Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Initials** | **Version** | **Comments** |
| 12/10/09 | martinr d&b | v1.0.0 | Initial release |
| 03/29/10 | martinr d&b | v1.0.1 | Note that commands are polled through the channel interface |
| 04/05/10 | RCO | v1.0.2 | Updated the module to include SNAPI 1.15 support. |
| 09/16/16 | tobiasw d&b | v2.0.0 | Update from CAN module to OCA / Ethernet module |

## Introduction

This is a reference manual to describe the interface provided between an AMX NetLinx system and d&b audiotechnik D20 / D80 / 10D / 30D amplifiers. The d&b audiotechnik D20 / D80 / 10D / 30D amplifiers support a CAN protocol as well as the OCA protocol which is based on Ethernet. Where in the last version of this AMX module the CAN protocol (in combination with an Ethernet to CAN interface) was used for communication with the amplifiers this updated version the network communication is based on Ethernet / OCA only.

Software versions used for this module:

Café Duet firmware version 3.00.316

NetLinx Studio version 4.3.1519

Café Duet Version: 1.8.85 Build Number: 85

Platform Version: 1.15.230 Build Number: 230

Runtime Version: 2.0.5 Build Number: 200

## Overview

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

A User Interface (UI) module is also provided. This module uses the standard interface described below and parses the command responses for feedback.

The following diagram provides a graphical view of the interface between the interface code and the Duet module.

d&b

Amplifiers

SNAPI

Duet

COMM

Module

NetLinx

UI Module

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](#_Adding_Functions_to) section for more information.

A sample UI module and a touch panel file are provided in the module package. 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.

## Implementation

To interface to the AMX d&b audiotechnik D20 / D80 / 10D / 30D Amplifier module, the programmer must perform the following steps:

1. Define the device IDs for each d&b amplifier.
2. Define the virtual device IDs that the d&b audiotechnik OCA COMM module will use to communicate with the main program and User Interface. Duet virtual devices use device numbers 41000 - 42000. Note that each channel (!) on each amplifier needs its own virtual device ID. Use port numbers to identify the channels. E.g. 41000:1:0 for amplifier 1, channel A; 41000:2:0 for amplifier 1, channel B; …; 41001:1:0 for amplifier 2, channel A
3. If a touch panel interface is desired, a touch panel file D\_B\_UI\_Test.TP4 and module (D&B\_Test\_UI\_Module.axs and D\_B\_UI\_mod.axs) have been created for testing.
4. The Duet d&b audiotechnik D20 / D80 / 10D / 30D Amplifier module must be included in the program with a DEFINE\_MODULE command. Every amplifier needs its own module instance. So for several amplifiers the module has to be included several times by means of the DEFINE\_MODULE command.

An example of how to do this is shown below.

DEFINE\_DEVICE

// physical devices (device number : port number : system number):

dvAmp1 = 0:5:0 // amplifier 1

dvAmp2 = 0:6:0 // amplifier 2

// virtual devices (and channels):

// amplifier 1:

vdvDBaudioAmp1ChannelA = 41001:1:0

vdvDBaudioAmp1ChannelB = 41001:2:0

vdvDBaudioAmp1ChannelC = 41001:3:0

vdvDBaudioAmp1ChannelD = 41001:4:0

// amplifier 2:

vdvDBaudioAmp2ChannelA = 41002:1:0

vdvDBaudioAmp2ChannelB = 41002:2:0

vdvDBaudioAmp2ChannelC = 41002:3:0

vdvDBaudioAmp2ChannelD = 41002:4:0

…

// touch display:

dvDBaudioTP = 10001:1:0

DEFINE\_VARIABLE //Define arrays of button channels used on your own touch panel

VOLATILE INTEGER nBtn\_Amp\_1\_Channel\_A[] = { 1,2,3,4,5,6,7,8,9,10,

...

VOLATILE INTEGER nLvl\_amp\_1\_channel\_A[] = { 1 }

...

DEFINE\_START // Place define\_module calls to the very end of the define\_start section.

DEFINE\_MODULE 'D\_B\_UI\_mod' Amp1ChannelA(vdvDBaudioAmp1ChannelA, dvDBaudioTP, nBtn\_Amp\_1\_Channel\_A, nLvl\_amp\_1\_channel\_A)

DEFINE\_MODULE 'D\_B\_UI\_mod' Amp1ChannelB(vdvDBaudioAmp1ChannelB, dvDBaudioTP, nBtn\_Amp\_1\_Channel\_B, nLvl\_amp\_1\_channel\_B)

...

DEFINE\_MODULE 'D\_B\_UI\_mod' Amp2ChannelA(vdvDBaudioAmp2ChannelA, dvDBaudioTP, nBtn\_Amp\_2\_Channel\_A, nLvl\_amp\_2\_channel\_A)

…

// The final driver

// java module - parameters: first (virtual) amplifier (= duet device), d&b R70 (= physical device)

DEFINE\_MODULE 'DbAudiotechnik\_OCA\_Comm\_dr1\_0\_0' mAmpDev1(vdvDBaudioAmp1ChannelA, dvAmp1)

DEFINE\_MODULE 'DbAudiotechnik\_OCA\_Comm\_dr1\_0\_0' mAmpDev2(vdvDBaudioAmp2ChannelA, dvAmp2)

DEFINE\_MODULE 'ModuleComponent' mMdlCmp1(vdvAmpArray, dvTP)

The IP addresses of the amplifiers have to be passed into to the modules by defining the respective property. In many cases the virtual device channel A is representing functionalities that concern the whole amplifier. So only the channel A of each amplifier has to be delivered into the respective Java Comm module as paramter. Also the IP address has to be sent only to the channel A of each amplifier.

**Note:** The IP Port is set to the fixed value of 30013 by default and should work fine with up-to-date firmware. For legacy support it’s possible to pass in custom IP Ports.

DATA\_EVENT[vdvDBaudioAmp1ChannelA]

{

ONLINE:

{

SEND\_COMMAND vdvDBaudioAmp1ChannelA, "'PROPERTY-IP\_Address,', '10.0.0.87'"

// setting port might only be necessary for devices with old firmware

SEND\_COMMAND vdvDBaudioAmp2ChannelA, "'PROPERTY-Port,', '51499'"

WAIT 10

SEND\_COMMAND vdvDBaudioAmp1ChannelA, "'REINIT'" }

}

Upon initialization the AMX Comm module will communicate with the amplifier and information will be exchanged.

## Port Mapping

This module uses multiple virtual devices in order distinguish events for one zone from another.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Virtual Device** | **Channels** | **Levels** | **Control** | **Feedback** |
| 41001:1:0 Amplifier1 Channel A, Power1, & Volume1 | All Channels | All Levels | All Control Commands | All Feedback Commands |
| 41001:2:0 Amplifier1 Channel B, Power2, & Volume2 | All Channels | All Levels | All Control Commands | All Feedback Commands |
| 41001:3:0 Amplifier1 Channel C, Power3, & Volume3 | All Channels | All Levels | All Control Commands | All Feedback Commands |
| 41001:4:0 Amplifier1 Channel D, Power4, & Volume4 | All Channels | All Levels | All Control Commands | All Feedback Commands |
| 41002:1:0 Amplifier2 Channel A, Power5, & Volume5 | All Channels | All Levels | All Control Commands | All Feedback Commands |
| 41002:2:0 Amplifier2 Channel B, Power6, & Volume6 | All Channels | All Levels | All Control Commands | All Feedback Commands |
| 41002:3:0 Amplifier2 Channel C, Power7, & Volume7 | All Channels | All Levels | All Control Commands | All Feedback Commands |
| 41002:4:0 Amplifier2 Channel D, Power8, & Volume8 | All Channels | All Levels | All Control Commands | All Feedback Commands |

## Channels

The UI module controls the Amplifier via channel events (NetLinx commands *pulse, on, and off*) sent to the COMM module. 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** |
| 9 | PULSE: Cycle power |
| 24 | ON: Ramp Volume Up – used for feedback also  OFF: Stop Ramping |
| 25 | ON: Ramp Volume Down – use for feedback also  OFF: Stop Ramping |
| 26 | PULSE: Cycle mute |
| 27 | PULSE: Set Power On |
| 28 | PULSE: Set Power Off |
| 199 | ON: Set Volume Mute On – used for feedback also  OFF: Set Volume Mute Off |
| 251 | ON: Device is online – used for feedback only  OFF: Device is not online |
| 252 | ON: Data is Initialized – use for feedback only  OFF: Data is not Initialized |
| 255 | ON: Set power on – used for feedback also  OFF: Set power off |
| \*302 | PULSE: Toggle Load Monitoring on/off for current amp channel selected |
| \*321 | PULSE: Switch to AmpPreset No 1 |
| \*322 | PULSE: Switch to AmpPreset No 2 |
| \*323 | PULSE: Switch to AmpPreset No 3 |
| \*324 | PULSE: Switch to AmpPreset No 4 |
| \*325 | PULSE: Switch to AmpPreset No 5 |
| \*326 | PULSE: Switch to AmpPreset No 6 |
| \*327 | PULSE: Switch to AmpPreset No 7 |
| \*328 | PULSE: Switch to AmpPreset No 8 |
| \*329 | PULSE: Switch to AmpPreset No 9 |
| \*330 | PULSE: Switch to backup current settings to AmpPreset 13 and  switch to AmpPreset No 10 |
| \*331 | PULSE: Switch to backup current settings to AmpPreset 14 and  switch to AmpPreset No 11 |
| \*332 | PULSE: Switch to backup current settings to AmpPreset 15 and  switch to AmpPreset No 12 |
| \*333 | PULSE: Switch to AmpPreset Recall AmpPreset 13 |
| \*334 | PULSE: Switch to AmpPreset Recall AmpPreset 14 |
| \*335 | PULSE: Switch to AmpPreset Recall AmpPreset 15 |
| \*336 | PULSE: Toggle Input Monitoring for analog input channel A1 |
| \*337 | PULSE: Toggle Input Monitoring for analog input channel A2 |
| \*338 | PULSE: Toggle Input Monitoring for analog input channel A3 |
| \*339 | PULSE: Toggle Input Monitoring for analog input channel A4 |
| \*340 | PULSE: Toggle Input Monitoring for digital input channel D1 |
| \*341 | PULSE: Toggle Input Monitoring for digital input channel D2 |
| \*342 | PULSE: Toggle Input Monitoring for digital input channel D3 |
| \*343 | PULSE: Toggle Input Monitoring for digital input channel D4 |
| \*402 | ON: Load monitoring is on (feedback only)  OFF: Load monitoring is off (feedback only) |
| \*409 | ON: Power Status is Good (feedback only)  OFF: Power Status is Bad (feedback only) |
| \*410 | ON: General Error exists (feedback only)  OFF: No General Error (feedback only) |
| \*411 | ON: Amplifier Channel Error exists (feedback only)  OFF: No Amplifier Channel Error (feedback only) |
| \*415 | ON: Selected AmpPreset is modified (feedback only)  OFF: Selected AmpPreset is still active (feedback only) |
| \*436 | ON: Input Monitoring for analog input channel A1 is on (feedback only) OFF: Input Monitoring for analog input channel A1 is off (feedback only) |
| \*437 | ON: Input Monitoring for analog input channel A2 is on (feedback only) OFF: Input Monitoring for analog input channel A2 is off (feedback only) |
| \*438 | ON: Input Monitoring for analog input channel A3 is on (feedback only) OFF: Input Monitoring for analog input channel A3 is off (feedback only) |
| \*439 | ON: Input Monitoring for analog input channel A4 is on (feedback only) OFF: Input Monitoring for analog input channel A4 is off (feedback only) |
| \*440 | ON: Input Monitoring for digital input channel D1 is on (feedback only) OFF: Input Monitoring for digital input channel D1 is off (feedback only) |
| \*441 | ON: Input Monitoring for digital input channel D2 is on (feedback only) OFF: Input Monitoring for digital input channel D2 is off (feedback only) |
| \*442 | ON: Input Monitoring for digital input channel D3 is on (feedback only) OFF: Input Monitoring for digital input channel D3 is off (feedback only) |
| \*443 | ON: Input Monitoring for digital input channel D4 is on (feedback only) OFF: Input Monitoring for digital input channel D4 is off (feedback only) |

Table 1 - Virtual Device Channel Events

## Levels

The UI module controls the Amplifier via level events (NetLinx command *send\_level*) sent to the COMM module. The levels supported by the COMM module are listed below. These levels are associated with the virtual device(s) and are independent of the levels associated with the touch panel device.

|  |  |
| --- | --- |
| **Level** | **Description** |
| 1 | Volume Level (range 0…255) (device range –57.5dB to 6.0dB) |

Table 2 - Virtual Device Level Events

## Command Control

The UI module controls the Amplifier via command events (NetLinx command *send\_command*) sent to the COMM module. The commands supported by the COMM module are listed below.

|  |  |
| --- | --- |
| **Command** | **Description** |
| ?DEBUG | Request the state of the debug feature.  ?DEBUG |
| DEBUG-<value> | Set the state of debugging messages in the UI module and the COMM module.  **Note:** See Programming Notes section.  <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 |
| PASSBACK-<state> | Enable or disable response reporting from the device. When enabled device responses will be sent as strings to the virtual device.  **Note:** By default, this is set to off at startup.  <state> : 0 = Off (default)  1 = On  PASSBACK-0 |
| PASSTHRU-<string> | Allows user the capability of sending commands directly to whatever unit is attached 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](#_Adding_Functions_to)” section below.  <string> : string to send to unit  **Note:** The module doesn’t add any bytes to the message by itself.  "'passthru-',$3b,$00,$01,$00,$00,$00,$1b,$01,$00,$01,$00,$00,$00,$12,$00,$00,$7f,$ff,$10,$00,$82,$05,$00,$04,$00,$02,$01,$01"  (= mute channel A) |
| ?PROPERTY-<key> | Get the value of a property <key>. If the value is not initialized, an empty string is returned.  **Note:** The properties must be addressed to the first virtual device.  <key> : IP\_Address  Port |
| 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.  **Note:** The properties must be addressed to the first virtual device.  <key> : IP\_Address  <value> : string representing the amplifier IP address  <key> : Port  <value> : string representing the amplifier IP port |
| REINIT | Re-initialize the communication link and data.  **Note:** This command deletes any messages waiting to go out to the device.  REINIT |
| ?VERSION | Query for the current version number of the Duet module.  ?VERSION |

Table 3 – Send Command Definitions

|  |
| --- |
| iption |

## Command Feedback

The COMM module provides feedback to the User Interface module for Amplifier changes via command events. The commands supported are listed below.

**PLEASE NOTE:** Feedback is only provided when there is a state change. If no state change resulted from the command sent in, then no feedback will be returned.

To poll the feedbacks use the channel interface.

|  |  |
| --- | --- |
| **Command** | **Description** |
| AMPTEMERATURE-<value> | Return the temperature of the selected amp.  <value>: string representing the current amp temperature in degrees Celsius.  AMPTEMERATURE-26 |
| AMP\_CHANNEL\_ERROR\_NO-<value> | Return any error number that the selected amplifier channel might have  <value>: string representing any error number of the assigned amplifier channel.  AMP\_CHANNEL\_ERROR\_NO-08 |
| AMP\_CHANNEL\_ERROR\_TXT-<value> | Return any error text for the selected amplifier channel.  <value>: string representing any error text for the assigned amplifier channel.  AMP\_CHANNEL\_ERROR\_TXT-Input Monitoring Fault |
| AMP\_NAME-<value> | Return the name assigned to the selected amplifier.  <value>: string representing the current name assigned to the amplifier.  AMP\_NAME-D12 V2.18 |
| DEBUG-<value> | Return the state of debugging messages in the UI module and the COMM module.  <value> : 1 = set only error messages on  2 = set error and warning messages on  3 = set error, warning and info messages on  4 = set all messages on  DEBUG-1 |
| DEV\_ERROR\_NO-<value> | Return any error number that the selected amplifier might have.  <value>: string representing any error number of the assigned amplifier.  DEV\_ERROR\_NO-01 |
| DEV\_ERROR\_TXT-<value> | Return any error text that the selected amplifier might have.  <value>: string representing any error text of the assigned amplifier. |
| POTI\_LEVEL\_DB-<value> | Return the Poti Level in dB of the selected amplifier.  <value>: string representing the Poti Level in dB of the the assigned amplifier.  POTI\_LEVEL\_DB--57,5 |
| PROPERTY-<key>,<value> | Feedback on the value of property <key>.  **Note**: An empty string is returned if the property has no value.  The properties must be addressed to the first virtual device.  <key> : IP\_Address  <value> : string representing the amplifier IP address  <key> : Port  <value> : string representing the amplifier IP port |
| VERSION-<version> | Report the version number of the module.  <version> : xx.yy.zz = module version number  VERSION-1.0.0 |

## Device Notes

* Device firmware v3.50.430 must be used for the NI master controller.
* If the device is turned off (not standby), it will stop responding to all heartbeat messages. This will cause the module to be taken offline.
* Ramping volume up/down is faked.
* Volume Preset and amp Preset channels perform the same function.
* Module may take a while to initialize depending upon configuration.
* Amp Presets 10..12 save the current preset to Presets 13..15 before recalling its programmed functions.

## 

## Programming Notes

The module will initially query for power, mute, poti level, channel and device errors, temperature and load / input monitoring states. For updates of those properties the OCA subscription mechanism is used. In contrast to the previously used polling mechanism the Comm module now subscribes on the amplifier for the mentioned amplifier properties and gets informed by the amplifier on changes. This mechanism reduces network traffic and leads to faster UI updates.

## 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. The command has to be sent to virtual device channel A of the desired device.

// MUTE ON, channel A

SEND\_COMMAND vdvDBaudioAmp1ChannelA, "'passthru-',$3b,$00,$01,$00,$00,$00,$1b,$01,$00,$01,$00,$00,$00,$12,$00,$00,$7f,$ff,$10,$00,$82,$05,$00,$04,$00,$02,$01,$01"

The OCA protocol uses so called handles to assign responses of the amplifier to previous requests. A handle is a number stored in four bytes of the message that is shared between amplifier and controller (in that case the Comm module). This number is used to map an incoming message from the amplifier to a previous request. So the controller knows which request the response belongs to.

Usually this handle mechanism is handled by the Comm module. When using the passthru command the complete OCA message has to be constructed manually. As the handle is part of the OCA message it also has to be created manually. In order not to disturb the handle mechanism implemented by the Comm module there’s a range for handle numbers that can be used for manually created OCA messages. Handle numbers with a value >= 7FFF (hexadecimal) can be used for the passthru functionality. **Handle numbers < 7FFF hex are reserved for the automatic handle system.**

## Log messages

The AMX d&b COMM module writes log messages into the NetLinx log. Depending on the active log level (ERROR / WARNING / INFO / DEBUG) these messages are more or less verbose. Here’s an excerpt of those log messages for some common cases. Please note that due to the big number of possible log messages only the most important or frequent messages are listed here.

To set the desired log level use the terminal command “msg on debug” (info / warning / error) **in combination with** the DEBUG-4 (3 / 2 / 1) command.

|  |  |  |
| --- | --- | --- |
| **Category** | **Text** | **Description** |
| ERROR | No IP address found. Cancelling (re)initialization. | Indicates that no IP address has been set for virtual device in NetLinx script.  Solution:  Set IP address property via SEND\_COMMAND and force REINIT afterwards |
| WARNING | [IP address] Loading AmpPreset failed. | Amplifier preset could not be loaded.  Possible reasons:   * Desired amplifier preset slot doesn’t contain any content * A technical problem occurred   Solution:  Store an amplifier setting in the desired slot. |
| WARNING | Lost connection to OCA device. Trying to reconnect. | Indicates that the connection to an amplifier got lost.  Possible reasons:   * Unplugging of the device * Switching the amplifier off * Forcing of “Reinitialize” * Three missed heartbeat messages from an amplifier in a row   Solution:   * Make sure that all devices are connected and switched on * The COMM module will automatically try to reconnect |
| INFO | [IP address] Amplifier offline! |
| INFO | Starting another attempt to reconnect. | The COMM module makes an attempt to connect with an amplifier via TCP/IP |
| INFO | [IP address] Amplifier online! | States that the TCP/IP connection to an amplifier succeeded. |
| INFO | [IP address] Amplifier data initialized. Net  Linx is in sync with amplifier. | Indicates that COMM module initially was able to read the power property state from an amplifier. The states of the user interface controls should be in sync with the amplifier. |
| DEBUG | Receiving heart beat with interval 5. | Message gets displayed when heartbeat message of amplifier arrived. States that TCP/IP connection is active and working. |
| DEBUG | Diag: [] - SocketConnection: run(): IOException - java.net.SocketException: Socket closed | Occurs if the Ethernet connection with an amplifier gets lost.  Possible reasons:   * Unplugging of the device * Switching the amplifier off * Forcing of “Reinitialize”   Solution:   * Make sure that all devices are connected and switched on * The COMM module will automatically try to reconnect |
| DEBUG | Diag: [] - SocketConnection: run: failed data read  (0001133229) java.net.SocketException: Socket closed (0001133231)  at java.net.SocketInputStream.socketRead0(Native Method) … |
| DEBUG | OcaAmplifier$1: [] - SocketConnection: createSocket: IO exception | Indicates that TCP/IP connection can’t be established.  Possible reasons:   * Amplifier offline * Device unplugged   Solution:   * Make sure that all devices are connected and switched on   The COMM module will automatically try to reconnect |
| DEBUG | OcaAmplifier$1: [] - SocketConnection: createSocket: socketcreate failed |