



# Duet Module Help File

## **Newline Pro Series**

TABLE OF CONTENTS

Introduction .....3

Overview .....4

Implementation .....5

Port Mapping.....6

Command Events .....10

Command Feedback.....11

Important Notes.....12

Programming Notes .....13

Revision History

Date	Initials	Comments
6-19-2025	VR	v1.0.0 Initial Release

## **Introduction**

This is a reference manual to describe the interface provided between an AMX NetLinx system and a Newline Pro Series monitor. The AMX module connects with the device over Ethernet or RS232.

This module was written with DuetCafe version 1.8.85 build 85, and NetLinx Studio version 4 build 4.4.1950. The panel program was built using TPDesign5 version 1.5.0, build 111.

This module is compatible with NX series controllers only, minimum firmware version is 1.6.175.

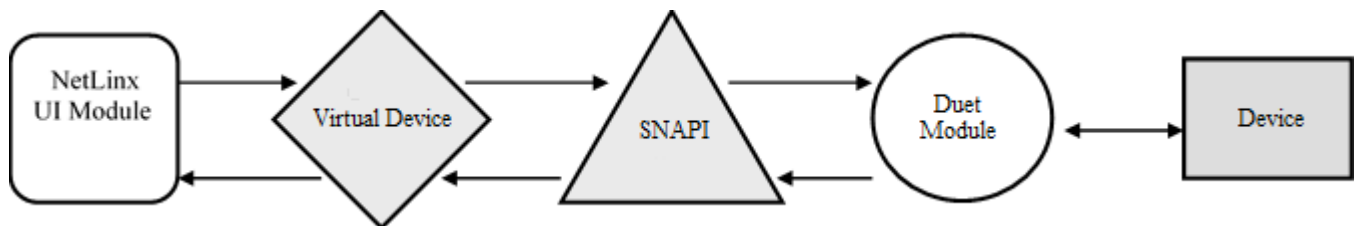
The module must be configured with the IP address and MAC address of the device it is to connect to if using IP control. This must be done manually via a telnet session or can be automated from the UI module. These properties must be set using the 'PROPERTY-' command. The 'REINIT' command is used to notify the module that new properties are now available and to start using them. When these commands are used in the appropriate order, the module will attempt to connect to the specified IP address periodically until a connection is made or a new IP address is submitted using the 'PROPERTY-' and 'REINIT' commands. Please see the [Programming Notes](#) section for additional information.

## **Overview**

The module translates between the standard interface described below and the device protocol. It parses the buffer for responses from the device, sends strings to control the device, 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 gives a graphical view of the interface between the interface code and the Duet module.



A sample UI module and a touch panel file are provided in the module package. These are not intended to cover every possible application scenario but can be expanded as needed by a dealer to meet the requirements of a particular installation.

## **Implementation**

To interface to the device module, the programmer must perform the following steps:

1. Define the device ID for the physical device that will be controlled.
2. Define the main virtual device ID that the device COMM module will use to communicate with the main program and User Interface. Duet virtual devices use device numbers 41000 - 42000.
3. If a touch panel interface is desired, a touch panel file (Newline Pro Series Demo v1\_0.TP5) and module (DEMO UI Module.aspx) have been created for testing.
4. The Newline\_ProSeries\_dr1\_0\_0.jar 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 device to be controlled, and the virtual device ID for communicating to the main program.

An example of how to do this is shown below.

```

DEFINE_DEVICE

// ----- PHYSICAL DEVICES

dvDevice = 0:3:0;
dvPanel = 10001:1:0;

// ----- VIRTUAL DEVICES

vdvDevice = 41001:1:0;

(*****
(*          CONSTANT DEFINITIONS GO BELOW          *)
*****)
DEFINE_CONSTANT

(*****
(*          STARTUP CODE GOES BELOW          *)
*****)
DEFINE_START

// ---- COMM MODULE

DEFINE_MODULE 'Newline_ProSeries_v1_0_0' modNewline(vdvDevice, dvDevice);

// ---- UI MODULE

DEFINE_MODULE 'DEMO UI Module' modNewlineUI(dvPanel, vdvDevice);

(*****
(*          THE EVENTS GO BELOW          *)
*****)
DEFINE_EVENT

DATA_EVENT[vdvDevice]
{
  ONLINE:
  {
    wait 50
    {
      SEND_COMMAND vdvDevice,'DEBUG-4';
      SEND_COMMAND vdvDevice,'"PROPERTY-IP_Address,192.168.1.1"'";
      SEND_COMMAND vdvDevice,'"PROPERTY-MAC_Address,00:00:00:00:00:00"'";

      SEND_COMMAND vdvDevice,'REINIT';
    }
  }
}

```

```

    }
  }
}

```

## **Port Mapping**

The module requires a single port for SNAPI channel, level, and command mapping. The module uses port 1. The port is defined as part of the device definition:

```

// ----- VIRTUAL DEVICES
vdevDevice      = 41001:1:0;

```

The virtual device identifier is 41001, the port is 1, and the system is 0 – indicating the module will use the system identifier defined within the processor configuration.

## **Channel Events**

The UI module controls the device via channel events for features indicated by on/off or enabled/disabled state. The channels supported by the 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. Not all channels will be available on all virtual ports.

Note: An '\*' indicates an extension to the standard SNAPI API

<b>Channel</b>	<b>Description</b>
9	PULSE: Device power toggle.
24	RAMP: Speaker volume increased while channel is active.
25	RAMP: Speaker volume decreased while channel is active.
26	PULSE: Toggle the speaker volume mute state.
27	PULSE: Power on the monitor.
28	PULSE: Power off the monitor.
44	PULSE: Open the menu of the device.
45	PULSE: Navigation up on the device.
46	PULSE: Navigation down on the device.
47	PULSE: Navigation left on the device.
48	PULSE: Navigation right on the device.
49	PULSE: Navigation ok on the device.
50	PULSE: Set the device to home. Also known as source input 9 (Smart System).
101	PULSE: Open the settings of the device.
104	PULSE: Navigation return on the device.
213	PULSE: Device video freeze toggle.
214	ON: Set and indicates the device video freeze is on. OFF: Set and indicates the device video freeze is off.
199	ON: Set and indicates the speaker volume mute on. OFF: Set and indicates the speaker volume mute off.
251	ON: Indicates the module is communicating with the device. OFF: Indicates the module is not communicating with the device.

252	ON: Indicates the module is initialized with all device state information. OFF: Indicates the module is not initialized.
-----	---

**Table 1 - Virtual Device Channel Events**



## **Level Events**

The UI module controls the device via level events sent to the module for features indicated by a range value. The commands supported by the module are listed below.

Note: An ‘\*’ indicates an extension to the standard API.

Level	Description
1	Level indicates current speaker volume level as a scaled percentage range from 0 to 100.

**Table 2 – Level Events**

## **Command Events**

The UI module controls the device via command events sent to the module for features other than on/off or range elements. The commands supported by the module are listed below.

Note: An ‘\*’ indicates an extension to the standard API.

Command	Description
DEBUG-<value>	<p>Set the state of debugging messages in the UI module and the Module. Use the first virtual port when sending this command.</p> <p>Note: See Programming Notes section.</p> <p>&lt;value&gt; : 1 = set only error messages on  2 = set error and warning messages on  3 = set error, warning &amp; debug messages on  4 = set all messages on</p> <p>Example:  DEBUG-1</p>
?DEBUG	Returns the current debug level, format is DEBUG-<level>.
?INPUTSELECT	<p>Set the input to switch the device to, format is INPUTSELECT-&lt;value&gt;.</p> <p>&lt;value&gt;:</p> <ul style="list-style-type: none"> <li>• 1 = HDMI 1</li> <li>• 2 = HDMI 2</li> <li>• 3 = HDMI 3</li> <li>• 4 = USB-C Front</li> <li>• 5 = USB-C Rear</li> <li>• 6 = Internal PC</li> <li>• 7 = Internal SDM</li> <li>• 8 = DP</li> <li>• 9 = Smart System (Home)</li> </ul> <p>EXAMPLE:  INPUTSELECT-1</p>

**Table 3 – Send Command Definitions**

## **Command Feedback**

The COMM module provides feedback to the User Interface module for all command events.

Command	Description
?DEBUG	<p>Request returns the state of debugging messages in the UI module and the Module. This is reported on the first virtual port.</p> <p>&lt;value&gt; : 1 = set only error messages on  2 = set error and warning messages on  3 = set error, warning and debug messages on  4 = set all messages on</p> <p>DEBUG-1</p>
?INPUTSELECT	<p>Returns the input selected reported by the device.</p> <p>&lt;value&gt;:</p> <ul style="list-style-type: none"> <li>• 1 = HDMI 1</li> <li>• 2 = HDMI 2</li> <li>• 3 = HDMI 3</li> <li>• 4 = USB-C Front</li> <li>• 5 = USB-C Rear</li> <li>• 6 = Internal PC</li> <li>• 7 = Internal SDM</li> <li>• 8 = DP</li> <li>• 9 = Smart System (Home)</li> </ul> <p>Example:  INPUTSELECT-1</p>

**Table 2 - Command Feedback Definitions**

## **Important Notes**

- Please contact manufacturer's customer service regarding support for the Newline Pro Series.
- The hardware used to test this module include an NX-1200 series controller with firmware version 1.1.48, and an MT-1002 touchpanel with firmware version 1.8.14.
- Properties can be modified by calling the SEND\_COMMAND with the property name and new value. See programming notes below.

Property Name	Default Value
IP Address	192.168.1.100
MAC Address	00:00:00:00:00:00

- The module will automatically initialize when the program starts up (or manually when the REINIT command is sent). Initialization consists of querying the device for all state information required for proper module operation.
- Response times can vary based on network connection and activity within the device. Some commands take longer to process than others. Note that the power commands take up to 15 seconds to process and return a response.

## **Programming Notes**

This module support RS232 or IP control of the Newline Pro Series monitor.

### **RS232 Notes:**

To establish a communication connection between the module and the device, the module has no required properties. Use the PROPERTY- command to set the DEBUG level and then use the REINIT command to force it to take effect. Here is an example of how these commands are used. Note that your virtual device (41001:1:0) may differ from this example. Substitute the appropriate values where necessary.

```
SEND_COMMAND vdvDevice,'DEBUG-4';
SEND_COMMAND vdvDevice,'REINIT';
```

### **IP Notes:**

To establish a communication connection between the module and the device, the module must be informed of the IP address of the device it is to connect to, and the MAC address to use for wake on lan. Use the PROPERTY- command to set this information and then use the REINIT command to force it to take effect. Here is an example of how these commands are used. Note that your virtual device (41001:1:0) and the IP address and MAC address will differ from this example. Substitute the appropriate values where necessary.

```
SEND_COMMAND vdvDevice,'DEBUG-4';
SEND_COMMAND vdvDevice,'"PROPERTY-IP_Address,',      IP_ADDRESS";
SEND_COMMAND vdvDevice,'"PROPERTY-MAC_Address,',      MAC_ADDRESS";
SEND_COMMAND vdvDevice,'REINIT';
```

These values have been defined as constant variables in the main module and should be changed to match your environment. The program takes care of passing the variables to the module.

```
(*****
(*      CONSTANT DEFINITIONS GO BELOW      *)
*****)
DEFINE_CONSTANT

////////////////////////////////////
// DEVICE CONNECTION PROPERTIES
////////////////////////////////////
CHAR IP_ADDRESS[]  = '192.168.1.100';
CHAR MAC_ADDRESS[] = '00:00:00:00:00:00'
```