NetLinx Module Interface Specification
for the

Powersoft “8 canali” Series Amplifier

IP Control Interface

Programmed for Powersoft by
www.romani-controls.com

romani
ships&home automation

TABLE OF CONTENTS

LIST OF TABLES

Revision History

Date	Initials	Revision	Comments
10-01-13	EV	v1.0.0	Initial release of Documentation for V1.0

Introduction

This document describes the software module for NetLinx in use with the Powersoft “8 canali” series Amplifiers. This module was written using NetLinx Studio version 3.3.1 and **Master Firmware version 4.1.373**.

It will not work on Netlinx Master with firmware below 4.xx

Overview

The Powersoft Amplifier may be controlled using this NetLinx module. This module requires a single UDP port from a NetLinx Controller to the Amplifier being controlled. For installations containing more than one amplifier, multiple instantiations of this module may be used. The module implements the actual Powersoft protocol for communicating to the unit but exposes a more simplified, NetLinx-friendly protocol to the programmer.

Implementation

To interface to the Powersoft module, the programmer must do each of the following:

1. Define the NetLinx UDP connection that will be used to communicate to the amplifier.
2. Define the virtual device id the module will use to communicate with the main program. NetLinx virtual devices start with device number 33001.
3. This NetLinx 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: NetLinx UDP port id to which the unit is connected and the virtual device id to be used for communicating with the main program. There should only be one instantiation of the module for a given amplifier device.

An example of how to do this is shown below.

```
DEFINE_DEVICE

dvPower8      =     0: 6: 0          // Powersoft 8 Series Amplifier          Tcp-Ip


// Touch Panels
dvModero1a    =  11001: 1: 0
dvModero2a    =  11002: 1: 0

// Virtual Devices
vdvPower8     =  33002: 1: 0       // Powersoft   8 Series Amplifier          Virtual


DEFINE_VARIABLE

// General Debug


constant TpCount = 2
volatile dev dvModeroA[TpCount] = { dvModero1a ,dvModero2a}
```

```
//  Dynamic TEXT messages

volatile char DspTxtForMsgBox[28][50] =                 // TXT MsgBoxs
{

   'Please Wait.....',                                  // Progress Bar Recall Presets
   'Load Preset.',
   '',
   '',
   'Attention !',                                       // MsgBox Module #xx Not Ready
   'Module is NOT Ready.',
   '',
   '',
   'Please Wait until',                                 // MsgBox Connection
   'reloading DSP information...',
   '',
   '',
   'Please Wait until',                                 // MsgBox Connection
   'the DPS is rebooting...',
   '',
   '',
   'Please Wait!',                                      // MsgBox Wait load preset
   'Preset is loading.',
   'Try later....',
   '',
   'Please Wait!',                                      // MsgBox Wait general
   'DSP is not ready.',
   'Try later....',
   '',
   'Attention!',                                        // MsgBox Device Status
   'DSP do not answer.',
   'Check Power or TCP/IP connection.',
   ''
}


volatile integer DsptxtMsgBox[] =                       // MsgBox TP address channels
{
   3996,
   3997,
   3998,
   3999
}
```

```
volatile char DspMsgBoxPopPages[4][50]=          // MsgBox PopUp Page Name
{
   'PPON-PopUpMSG box',
   'PPON-PopUpAnswerbox',
   'PPON-PopUpProgressBar',
   'PPOF-PopUpProgressBar'
}

volatile integer DspLvlProgressBar = 28          // Remember to change level value on the TP Graphic

volatile char DSP2IpAddress[32] = '192.168.2.30'  // DSP IP address
volatile long DSP2IpPort        =  8002           // DSP Connection Port
volatile long DSP2DiscoveryPort = 30718           // DSP Discovery Port

volatile integer Dsp2ID = 65                      // DSP ID

volatile integer DSP2DisableButton = 4001         // No Function Button
volatile integer Dsp2ConnectBtn = 1000            // 1 = connect , 0 = disconnect
volatile integer Dsp2ConnectToAmp = 1             // 1 = enabled , 0 = disable

volatile integer Dsp2moduleNumber = 4             // Number of DSP Module

volatile integer Dsp2EnterPageBtn = 12
volatile integer Dsp2PollingBtn = 1120

volatile integer Dsp2PollEnable = 1               // 1 or 0

// LEGEND  =  [C] : Channel value  - [A] : Address value – [L] : Level value

volatile integer Dsp2nchControl[]=                //  Main DSP Controls  [C]
{
   1001, // Power On       - Will Power ON if 12V PSU Is Present
   1002, // Power Off      - Will Power OFF if 12V PSU Is Present else the command recall reboot function.
   1003, // Power Toggle   - Will Power TOGGLE if 12V PSU Is Present

   // First Module
   1004, // Led Blinking
   1005, // Mute On
   1006, // Mute Off
   1007, // Mute Toggle
   1008, // SwMute Off Ch1
   1009, // SwMute On Ch1
   1010, // SwMute Off Ch2
   1011, // SwMute On Ch2
   1012, // SwMute Toggle Ch1
   1013, // SwMute Toggle Ch2
```

4001, // Spare
4001, // Spare
4001, // Spare
4001, // Spare

// Second Module - If Present
1018, // Led Blinking
1019, // Mute On
1020, // Mute Off
1021, // Mute Toggle
1022, // Mute Off CH3
1023, // Mute On CH3
1024, // Mute Off CH4
1025, // Mute On CH4
1026, // SwMute Toggle Ch3
1027, // SwMute Toggle Ch4
4001, // Spare
4001, // Spare
4001, // Spare
4001, // Spare

// Third Module - If Present
1032, // Led Blinking
1033, // Mute On
1034, // Mute Off
1035, // Mute Toggle
1036, // Mute Off CH5
1037, // Mute On CH5
1038, // Mute Off CH6
1039, // Mute On CH6
1040, // SwMute Toggle Ch5
1041, // SwMute Toggle Ch6
4001, // Spare
4001, // Spare
4001, // Spare
4001, // Spare

// Fourth Module - If Present
1046, // Led Blinking
1047, // Mute On
1048, // Mute Off
1049, // Mute Toggle
1050, // Mute Off CH7
1051, // Mute On CH7
1052, // Mute Off CH6
1053, // Mute On CH6

```
    1054, // SwMute Toggle Ch7
    1055, // SwMute Toggle Ch8
    4001, // Spare
    4001, // Spare
    4001, // Spare
    4001, // Spare

//  General
    1060, // Device Type
    1061  // Firmware Info
}


// Group MUTE

volatile integer Dsp2Mute[]=                    // [C]
{
    1121, // Mute ch1
    1122, // Mute ch2
    1123, // Mute ch3 - If Present
    1124, // Mute ch4 - If Present
    1125, // Mute ch5 - If Present
    1126, // Mute ch6 - If Present
    1127, // Mute ch7 - If Present
    1128, // Mute ch8 - If Present

    1129, // UnMute ch1
    1130, // UnMute ch2
    1131, // UnMute ch3 - If Present
    1132, // UnMute ch4 - If Present
    1133, // UnMute ch5 - If Present
    1134, // UnMute ch6 - If Present
    1135, // UnMute ch7 - If Present
    1136, // UnMute ch8 - If Present

    1137, // HW Mute ch1&2
    1138, // HW Mute ch3&4 - If Present
    1139, // HW Mute ch5&6 - If Present
    1140, // HW Mute ch7&8 - If Present

    1141, // HW UnMute ch1&2
    1142, // HW UnMute ch3&4 - If Present
    1143, // HW UnMute ch5&6 - If Present
    1144  // HW UnMute ch7&8 - If Present

}
```

```
// Group Volumes

volatile integer nchDsp2VolUp[]=                    // [C]
{
   1201, // Module 1 Ch.1
   1202, // Module 1 Ch.2
   1203, // Module 2 Ch.1 - If Present
   1204, // Module 2 Ch.2 - If Present
   1205, // Module 3 Ch.1 - If Present
   1206, // Module 3 Ch.2 - If Present
   1207, // Module 4 Ch.1 - If Present
   1208  // Module 4 Ch.2 - If Present
}

volatile integer nchDsp2VolDown[]=                  // [C]
{
   1211, // Module 1 Ch.1
   1212, // Module 1 Ch.2
   1213, // Module 2 Ch.1 - If Present
   1214, // Module 2 Ch.2 - If Present
   1215, // Module 3 Ch.1 - If Present
   1216, // Module 3 Ch.2 - If Present
   1217, // Module 4 Ch.1 - If Present
   1218  // Module 4 Ch.2 - If Present
}

// Gesture Ipad controls
volatile integer nchDsp2ReloadVolume[]=             // [C]
{
   1301, // Module 1 Ch.1/2
   1302, // Module 2 Ch.3/4 - If Present
   1303, // Module 3 Ch.5/6 - If Present
   1304  // Module 4 Ch.7/8 - If Present
}

volatile integer lvlDsp2Volumes[] =                 // [L]
{
   11,12,       // TP volume levels
   13,14,15,16,17,18 // TP volume levels - If Present
}

volatile integer nchDsp2AddressValue[] =            // [C]
{
   1201, // Tp Channel Address for Module #1
   1202, // Tp Channel Address for Module #1
   1203, // Tp Channel Address for Module #2 - If Present
```

```
    1204, // Tp Channel Address for Module #2 - If Present
    1205, // Tp Channel Address for Module #3 - If Present
    1206, // Tp Channel Address for Module #3 - If Present
    1207, // Tp Channel Address for Module #4 - If Present
    1208  // Tp Channel Address for Module #4 - If Present
}


volatile float Dsp2VoldBStep = 1.0                  // dB - Start from 0.1 dB


volatile integer txtDsp2Info[] =                    // [A]Address and/or [C]channel Feedback
{
    1001, // Temperature Module #1                  [A]
    1002, // RMS Current Module #1 CH1              [A]
    1003, // RMS Current Module #1 CH2              [A]
    1004, // Impedence Module #1 CH1                [A]
    1005, // Impedence Module #1 CH2                [A]
    1006, // BitmapProtection Module #1             [A][C]
    1007, // Aux Positive Voltage                   [A]
    1008, // Aux Negative Voltage                   [A]
    4001, // Hardware Mute                          [A][C]
    1010, // Module #1 Ready                        [A][C]
    1011, // External Power                         [A][C]
    1012, // Preset Module #1                       [A]
    1013, // Temperature Module #2                  [A]
    1014, // RMS Current Module #2 CH3              [A]
    1015, // RMS Current Module #2 CH4              [A]
    1016, // Impedence Module #2 CH3                [A]
    1017, // Impedence Module #2 CH4                [A]
    1018, // BitmapProtection Module #2             [A][C]
    1019, // Aux Pos Voltage  Module #2             [A]
    1020, // Aux Neg Voltage  Module #2             [A]
    4001, // Hardware Mute                          [A][C]
    1022, // Module #2 Ready                        [A][C]
    4001, // Spare
    1024, // Preset Module #2                       [A][C]
    1025, // Temperature Module #3                  [A]
    1026, // RMS Current Module #3 CH5              [A]
    1027, // RMS Current Module #3 CH6              [A]
    1028, // Impedence Module #3 CH5                [A]
    1029, // Impedence Module #3 CH6                [A]
    1030, // BitmapProtection Module #3             [A][C]
    1031, // Aux Pos Voltage  Module #3             [A]
    1032, // Aux Neg Voltage  Module #3             [A]
    4001, // Hardware Mute                          [A][C]
```

```
    1034, // Module #3 Ready                    [A][C]
    4001, // Spare
    1036, // Preset Module #3                   [A]
    1037, // Temperature Module #4              [A]
    1038, // RMS Current Module #4 CH7          [A]
    1039, // RMS Current Module #4 CH8          [A]
    1040, // Impedence Module #4 CH7            [A]
    1041, // Impedence Module #4 CH8            [A]
    1042, // BitmapProtection Module #4         [A][C]
    1043, // Aux Pos Voltage  Module #4         [A]
    1044, // Aux Neg Voltage  Module #4         [A]
    4001, // Hardware Mute                      [A][C]
    1046, // Module #4 Ready                    [A][C]
    4001, // Spare
    1048, // Preset Module #4                   [A]
    4001, // Spare
    4001, // Spare
    1061, // Alarm Mod #1                       [C]
    1062, // Alarm Mod #2                       [C]
    1063, // Alarm Mod #3                       [C]
    1064, // Alarm Mod #4                       [C]
    1065, // Bat Former Mod #1                  [C]
    1066, // Bat Former Mod #2                  [C]
    1067, // Bat Former Mod #3                  [C]
    1068, // Bat Former Mod #4                  [C]
    1069, // Device Type                        [A]
    1070, // Firmware Info                      [A]
    4001  // Device Name                        [A]

}
// Hold down channel to edit names
volatile integer nchDsp2Presets[] =            // [A][C]
{
    1314, // Preset #1 - 1
    1315, // Preset #2
    1316, // Preset #3
    1317, // Preset #4
    4001, // Preset Aux

    1328, // Preset #1 - 2 (if Present)
    1329, // Preset #2
    1330, // Preset #3
    1331, // Preset #4
    4001, // Preset Aux

    1342, // Preset #1 - 3 (if Present)
```

```
    1343, // Preset #2
    1344, // Preset #3
    1345, // Preset #4
    4001, // Preset Aux

    1356, // Preset #1 - 4 (if Present)
    1357, // Preset #2
    1358, // Preset #3
    1359, // Preset #4
    4001  // Preset Aux

}

volatile integer nchDsp2PresetControls[] =                    // [C]
{
    1401, // Edit Name
    4001, // Spare
    1403  // Abort

}

persistent char Dsp2PresetsName[20][50] =
{
    'Preset #1','Preset #2','Preset #3','Preset #4','Preset Aux', // Module #1
    'Preset #1','Preset #2','Preset #3','Preset #4','Preset Aux', // Module #2 if Present
    'Preset #1','Preset #2','Preset #3','Preset #4','Preset Aux', // Module #3 if Present
    'Preset #1','Preset #2','Preset #3','Preset #4','Preset Aux'  // Module #4 if Present
}

volatile char Dsp2EditPopUpPageName[1][20] =
{
    'PresetSetup_PopUp_8'

}

volatile integer dsp2TpMask[2] =                          // [A]
{
    1511,1512
}
```

```
// Only 1 DEFINE_MODULE defined per Amplifier

define_module 'Powersoft 8ch Series rev1mod' AmpDSP2Mld2(dvPower8,
                                                         dvModeroA,
                                                         vdvPower8,

                                                         DSP2IpAddress,
                                                         DSP2IpPort,
                                                         DSP2DiscoveryPort,

                                                         Dsp2ID,
                                                         Dsp2nchControl,
                                                         Dsp2moduleNumber,
                                                         Dsp2EnterPageBtn,
                                                         Dsp2PollingBtn,
                                                         Dsp2PollEnable,
                                                         Dsp2Mute,
                                                         nchDsp2VolUp,
                                                         nchDsp2VolDown,
                                                         lvlDsp2Volumes,
                                                         nchDsp2ReloadVolume,
                                                         nchDsp2AddressValue,
                                                         Dsp2VoldBStep,
                                                         nchDsp2Presets,
                                                         nchDsp2PresetControls,
                                                         Dsp2PresetsName,
                                                         Dsp2EditPopUpPageName,

                                                         txtDsp2Info,

                                                         DsptxtMsgBox,
                                                         DspTxtForMsgBox,
                                                         DspMsgBoxPopPages,
                                                         DspLvlProgressBar,
                                                         Dsp2TpMask,
                                                         Dsp2ConnectBtn,
                                                         Dsp2ConnectToAmp,
                                                         Dsp2DisableButton)
```

See the Powersoft.axs source file for details on the module parameters.

Upon initialization the module will communicate with the Amplifier Device and information will be exchanged.
The Touch panel file supplied with the demo is designed for an iPad equipped with TPControl App.

Command Interface

The interface code will control the amplifier via command events (NetLinx command send_command). These commands will be sent to the module to affect control. Below are the commands supported. Each command processed provides, where possible, asynchronous feedback upon completion of the command.

SEND_COMMAND TO VIRTUAL DEVICE (CMD Format: <[COMMAND]>[VALUE].)

Command	Description
<DEBUG>x.	Command to Enable o Disable module debugger (X Parameters) 1 - for DEBUG ON 0 - for DEBUG OFF
<POWER>x.	Command to switch ON or OFF the amplifier. (X Parameters) 1 - for switch ON 2 - for switch OFF These commands take effect only if the 12v external power is present.
<CONNECT>x.	Command to Connect or Disconnect Device from Master (X Parameters) 1 - for CONNECT 0 - for DISCONNECT
<PRESET>z-y,x.	Command to recall presets from Device (X – Y – Z Parameters) **Note:** By default, this is set to off at startup. z = 1 - max TP count (Used for feedback) y = 1 - 4 (Module ID) x = 1 - 4 (Preset number)

Table 1 – Send Command Definitions

Device Notes

Please note that the Amplifier must be equipped with an external 12V DC
power supply in order to enable Power On/Off functions
The Amplifier accept only one connection at the time, so to be able to use
the Powersoft “Armonia” software you must disconnect the Amplifier from AMX with the above specified button. (DSP1ConnectToDSP)