

HALOGEN



Halogen Software

AMX Control Systems Guide Version 3
AMX® and the AMX logo are registered trademarks of AMX.

Table of Contents

CHAPTER 1: Overview	1
About This Document	1
Using the HAL System Documentation	1
Getting Started	3
CHAPTER 2: Introduction to Using External Control Systems with HAL	4
Configuring External Controls in Halogen	4
Connecting to a Halogen/HAL Control Server	4
Communicating with a Halogen/HAL Control Server	5
Using a Telnet Client to Test and Monitor the System	5
CHAPTER 3: Example HAL1x Configuration	6
Creating the Example Configuration	6
Configuration Summary	15
CHAPTER 4: Integrating AMX Control Systems with HAL	16
Overview	16
Checklist	17
Getting Started	17
Configuring IP Addresses	19
TCP/IP Client	22
Toggles	23
Toggle Presets	26
Wall Toggles	27
Commands	29
Command Presets	30
Levels	31
Sliders	33
Selectors	36
Source Selector	39
Preset Selector	40

Troubleshooting	43
APPENDIX A: HAL External Control Message Protocol	45
Details of external control messages	45
All Controls	46
Level Controls	46
Toggle Controls	48
Selector Controls	49
Command Controls	52
Communications Monitoring	54
APPENDIX B: Using PuTTY to Test External Control Systems	55
Trademarks	61

CHAPTER 1: Overview

About This Document

This guide explains how to use a HAL DSP Processor with AMX control systems. It is divided into three major sections:

- "Introduction to Using External Control Systems with HAL" on page 4 (Chapter 2)
- "Example HAL1x Configuration" on page 6 (Chapter 3)
- "Integrating AMX Control Systems with HAL" on page 16 (Chapter 4)

Chapter 2 explains the general approach for configuring a HAL for use with any external control system and how to connect and communicate with the HAL Control Server. The second section, Chapter 3, walks you through the construction of an example Halogen configuration that includes one or two common uses for each type of external control. Chapter 4 shows you how to set up and program an AMX control system to work with a HAL that is running the example configuration.

The appendices provide helpful reference information about the HAL external control message protocol and how to use a telnet client to monitor and troubleshoot the operation of a control system at the message protocol level.

Using the HAL System Documentation

A variety of documentation is available to help you get started with and use your HAL System:

Halogen Help System

A comprehensive help system is installed with the Halogen software. It contains all the information you need to work with the system. There are several ways to access the Help System:

- Click the **Help** icon on the application toolbar:  Clicking the icon itself opens the Help System. Clicking the down arrow displays a list of options including such things as access to the Rane website, checking for updates to the Halogen software, and sending an email to Rane.
- Click the Help icon  that appears in the upper right corner of Halogen dialog boxes. Clicking this icon opens the Help topic related to the specific dialog box. From there you can access the entire Help System, if needed.
- Right-click on different elements in the user interface. A **Help** option appears in the context menu. Click this option to open the relevant Help topic. Pressing F1 when an area of the user interface has focus also displays its relevant Help topic.
- To search for information within the Help System, you can use the tabs on the Help Viewer's left pane to search the index (click the **Index** tab) or perform a full-text search (click the **Search** tab). You can also use the **Quick search** box in the Help System toolbar to search for text within the currently displayed topic: 

HAL System Design Guide

This guide is offered as a PDF file and contains a product overview, details about the HAL System's key features, and best practices for designing a HAL audio system. Note that the information in this guide is also available in the Halogen Help System. You can find this guide on the Rane website (<http://rane.com/hal>) as

well as on the product DVD.

HAL System Installation Guide

This guide is offered as a PDF file and includes step-by-step instructions on installing the HAL hardware, loading a configuration, and testing the system. This information is also included in the Halogen Help System. You can find this guide on the Rane website (<http://rane.com/hal>) as well as on the product DVD.

AMX Control Systems Guide

This guide, includes an introduction to using external control systems with HAL. It also discusses an example HAL1 configuration and how to set up an AMX controller and touch panel to communicate with a Halogen/HAL Control Server. In addition, an appendix is included with reference information on the HAL external control message protocol and how to use a telnet client to monitor and troubleshoot the operation of a control system at the message protocol level.

The guide is designed to be used in conjunction with the files found in the AMX Support Package. The contents of this support package include:

- AMXControlSystems_Guide.pdf - The AMX Guide pdf file
- ControlSystemSample.hal - Halogen configuration file intended for loading in Halogen or your HAL1x
- Rane_HAL.apw - NetLinx Studio program project file
- Main.axs - NetLinx program source file
- Rane_HAL_TP.TP4 - TPDesign4 touch panel project file

The AMX Support Package is installed with the Halogen software and can be accessed from the Windows Start Menu under Rane Corporation -> Halogen -> Guides -> AMX. If you want to access the files directly they are available on the product DVD or from the Halogen install directory:

- Windows 8, 7 or Vista - C:\Program Files (x86)\Rane Corporation\Halogen\Guides\Support Packages\AMX directory
- Windows XP - C:\Program Files\Rane Corporation\Halogen\Guides\Support Packages\AMX directory.

You can also download the most up-to-date version of this support package from the Rane website (<http://rane.com/hal>).

Crestron Control Systems Guide

This guide, includes an introduction to using external control systems with HAL. It also discusses an example HAL1x configuration and how to set up a Crestron controller and virtual touch panel to communicate with a Halogen/HAL Control Server. In addition, an appendix is included with reference information on the HAL external control message protocol and how to use a telnet client to monitor and troubleshoot the operation of a control system at the message protocol level.

The guide is designed to be used in conjunction with the files found in the Crestron Support Package. The contents of this support package include:

- CrestronControlSystems_Guide.pdf - The Crestron Guide pdf file
- ControlSystemSample.hal - Halogen configuration file intended for loading in Halogen or your HAL1x
- Rane_HAL.smw - SIMPL Windows program project file
- Rane_HAL_TP.vtp - VisionTools Pro-e touch panel project file

- Rane HAL Level Processor.usp - SIMPL+ user module source file
- Rane HAL Level Processor.ush - Compiled user module

The Crestron Support Package is installed with the Halogen software and can be accessed from the Windows Start Menu under Rane Corporation -> Halogen -> Guides -> Crestron. If you want to access the files directly they are available on the product DVD or from the Halogen install directory:

- Windows 8, 7 or Vista - C:\Program Files (x86)\Rane Corporation\Halogen\Guides\Support Packages\Crestron directory
- Windows XP - C:\Program Files\Rane Corporation\Halogen\Guides\Support Packages\Crestron directory.

You can also download the most up-to-date version of this support package from the Rane website (<http://rane.com/hal>).

Stardraw Control Systems Guide

This guide, includes an introduction to using external control systems with HAL. It also discusses an example HAL1 configuration and an example Stardraw Control application and driver that communicates with a Halogen/HAL Control Server. In addition, an appendix is included with reference information on the HAL external control message protocol and how to use a telnet client to monitor and troubleshoot the operation of a control system at the message protocol level.

The guide is designed to be used in conjunction with the files found in the Stardraw Control Support Package. The contents of this support package include:

- StardrawControlSystems_Guide.pdf - The Stardraw Control Guide pdf file
- ControlSystemSample.hal - Halogen configuration file intended for loading in Halogen or your HAL1x
- HAL1 Stardraw Example Project.s03 - Stardraw Control Project
- HAL1 Example Driver.cs - Stardraw HAL1 driver source code. This is part of the Stardraw Control Project but is included separately here for reference.

The Stardraw Control Support Package is installed with the Halogen software and can be accessed from the Windows Start Menu under Rane Corporation -> Halogen -> Guides -> Stardraw Control. If you want to access the files directly they are available on the product DVD or from the Halogen install directory:

- Windows 8, 7 or Vista - C:\Program Files (x86)\Rane Corporation\Halogen\Guides\Support Packages\Stardraw Control directory
- Windows XP - C:\Program Files\Rane Corporation\Halogen\Guides\Support Packages\Stardraw Control directory.

You can also download the most up-to-date version of this support package from the Rane website (<http://rane.com/hal>).

Getting Started

Read this document in sequence if you are completely new to learning about and using external control systems with HAL and the Halogen software. Chapters 2 and 3 appear in other control systems guides so you can skip these if you've already read them. The AMX example in chapter 4 refers to the example HAL1 configuration often, so you may want to familiarize yourself by reading chapter 3 before starting on chapter 4.

CHAPTER 2: Introduction to Using External Control Systems with HAL

This section is a brief overview of the support that the HAL system provides for external controls and how your system can connect to and use these controls. For more information on this topic, please see the Halogen Online Help.

Configuring External Controls in Halogen

Halogen provides the ability to configure controls for use with external control systems. These controls can be of any of the four types that Halogen supports: Level, Toggle, Selector, and Command. Once you create an external control in Halogen using the Control Systems dialog, you can then link it to any other linkable control of the same type in your Halogen configuration. This lets your external control system set and monitor any linkable parameter in your HAL configuration. Since the external controls can link to any other linkable control in a configuration, you have tremendous flexibility in the access that you provide to end users and seamless integration of your control system with internal controls, such as those in DSP block parameters and Rane Digital Remotes.

Each control that you configure for external access includes a unique number that allows external control systems to identify each control. For example, you might define a level control to be number 1 while a toggle control is number 2.

Connecting to a Halogen/HAL Control Server

The HAL system contains a Control Server to provide a way for your control system to connect and use the configured external controls. To integrate your external control system with a HAL system, your control system simply connects to the Control Server over TCP/IP. In the spirit of allowing you to configure and test your HAL system as much as possible without needing actual HAL hardware present, there are two ways that you can connect to a HAL Control Server to develop, test and use your external control system.

First, when developing and testing your system, the Halogen software includes the Control Server, which runs whenever Halogen is not connected to a HAL. Even though the configured HAL system hardware is not present, using the Control Server built into Halogen allows you develop your configuration, including all of the external controls, and test them with your external control system. Of course, because the actual hardware is not present, you cannot process audio or use physical devices such as Digital Remotes.



NOTE: To use the Control Server built into Halogen, connect to port 4996 on the TCP/IP address of the PC that is running Halogen.

The other way to use the Control Server is to connect to a HAL DSP Processor itself. This server is always available on a HAL at port 4996 on any of its TCP/IP addresses. Once you have created and applied a HAL configuration that contains external controls, these controls are available to your external control system via the Control Server. We recommend that you configure a static IP address on your HAL when using the HAL Control Server, so that the server is always available to your control system at a stable address.



NOTE: To use the Control Server built into HAL, connect to port 4996 on the TCP/IP address of the HAL.

Communicating with a Halogen/HAL Control Server

Once connected to either the Control Server built into Halogen or to the server in an actual HAL, your external control system communicates with the Control Server using the HAL system External Control Message Protocol. This is an ASCII text based protocol that allows one or more control systems to access the external controls in your HAL configuration. Appendix A "HAL External Control Message Protocol" on page 45 fully defines these messages, but an example would be: <L&4&510>. This message is a 'set level' message that tells the HAL to set external level control number 4 to a new value of 51.0%. The protocol provides similar messages for getting and setting values for each of the four control types.

In addition to responding to messages it receives, the Halogen/HAL Control Server also sends messages to connected control systems whenever one of the external controls configured in HAL changes. This helps all of the connected external control systems remain up to date as values or links change.

Using a Telnet Client to Test and Monitor the System

Because the message protocol is ASCII text, you can use a standard telnet client to connect to a Halogen or HAL Control Server and send/receive messages. This can be a great way to initially test your control system and to troubleshoot any problems that occur while developing and deploying your system. Appendix B "Using PuTTY to Test External Control Systems" on page 55 provides complete information for you to use a simple public domain Windows telnet client, PuTTY, with the Halogen/HAL Control Server.

CHAPTER 3: Example HAL1x Configuration

The external control system example uses a HAL1x configuration that includes a set of external controls linked to processing block and preset controls that demonstrate some common uses of the end user external controls. There is at least one control of each type: Command, Level, Selector, and Toggle. We've also constructed this configuration using only inputs and outputs available on the HAL1x so that you can load and apply this configuration to any HAL1x without having to attach and configure other devices such as RADs or DRs.

"Creating the Example Configuration" below shows how to build the complete configuration from scratch. Of course, we also provide the configuration file for you so you don't have to go through all of the work, but it's your choice. You can either follow the directions below to create the configuration or open the file we provide and use the instructions below to learn about what's in the configuration.

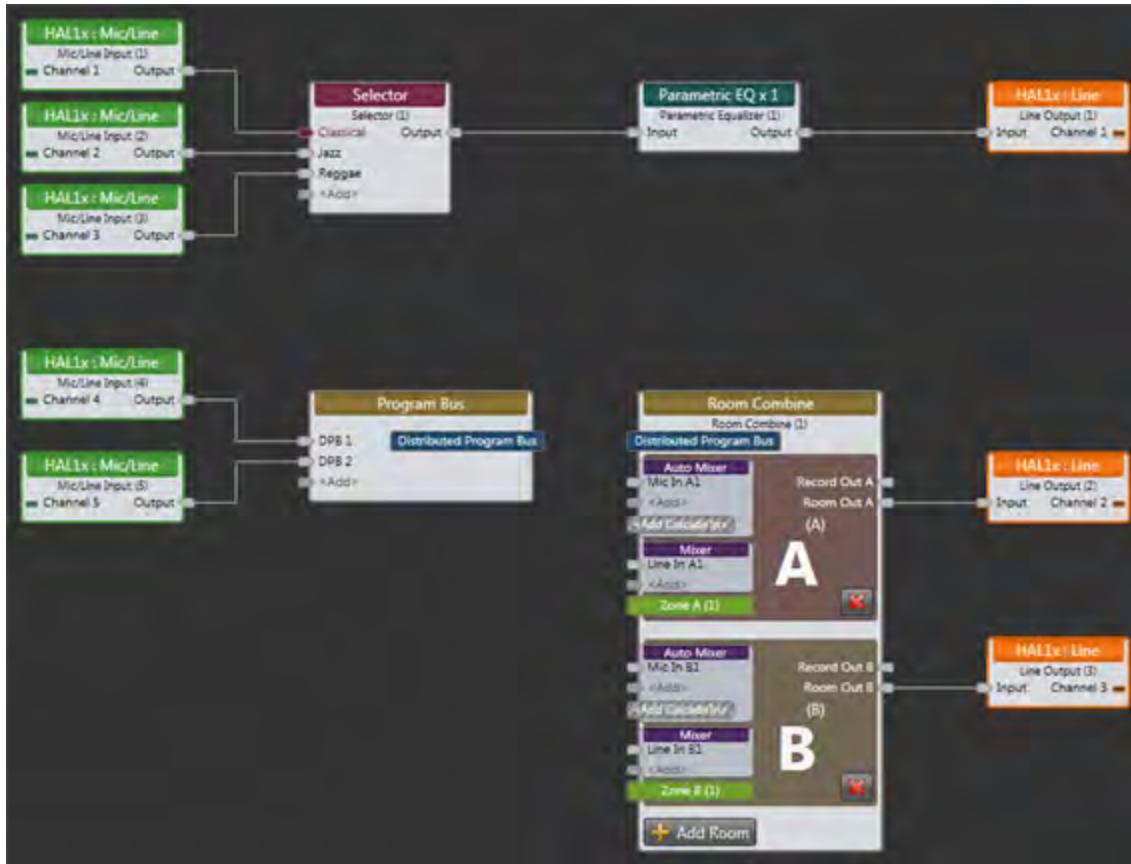
Alternatively, you can just read the "Configuration Summary" on page 15 to learn what external controls this configuration provides.

Creating the Example Configuration

To get started, all of the work is done in the processing workspace, so click on the **Processing** tab in the upper right section of the Halogen application.

Click on the tab for the **I/O** palette and drag the first five HAL1x Mic/Line inputs from the palette and drop them on the left side of the processing map. In a similar way, drag the first three HAL1x Line Outputs from the palette and drop them on the right side of the processing map.

Next, fill in the middle processing area by clicking on the **DSP** palette tab and dragging four processing blocks to the processing map: Parametric EQ, Distributed Program Bus, Room Combine, and Selector. For the selector block, create one additional input and set the input names to Classical, Jazz, and Reggae. Also, delete room C from the Room Combine Processor by clicking the red 'X' in room C. Wire these together with your I/O blocks so that your processing map looks like this:



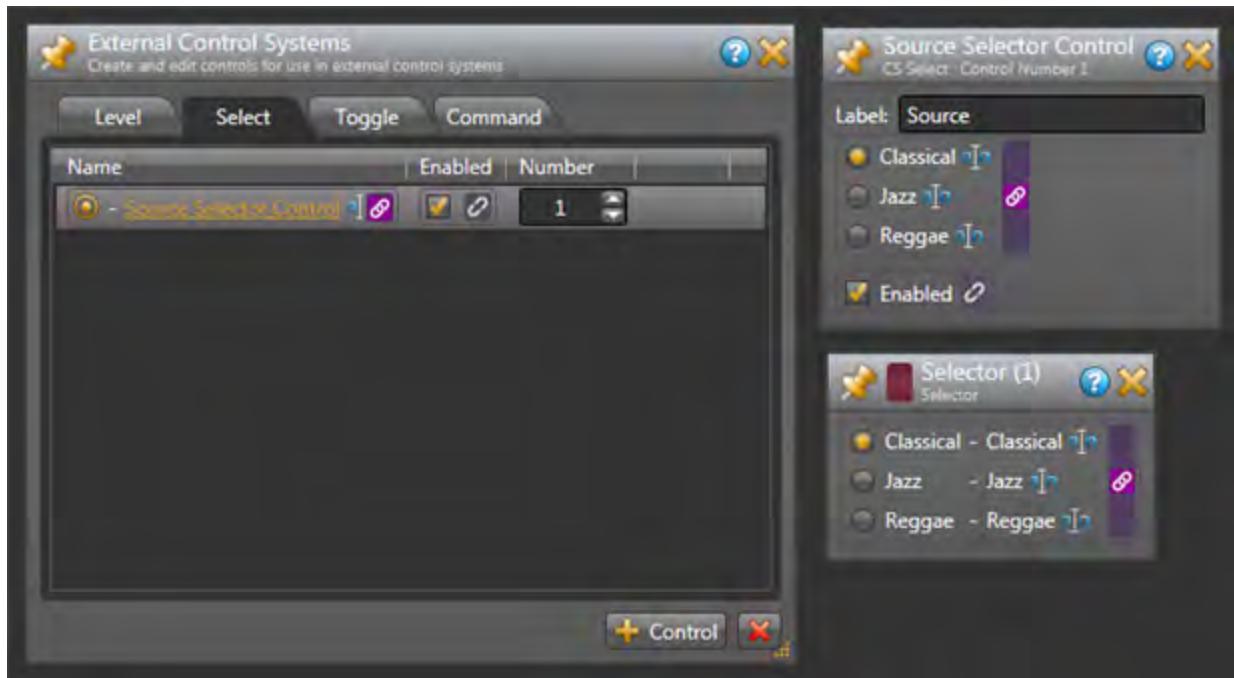
The top row of blocks provides audio for a lounge that has three program source inputs and some PEQ processing and the bottom row is a simple room combine configuration.

Next, we'll add some external controls using the External Control Systems dialog and link them to some of the controls available in the processing blocks we are using.

First, we'll create a control for the lounge input program selector:

1. Just above the upper left of the Processing Workspace, click on the **Control Systems** button then select **External Controls** to open the External Control Systems property dialog.
2. In the External Control Systems dialog, click on the **Select** tab.
3. Create a new Selector control by clicking on the **+Control** button.
4. Change the name of this new control by clicking the edit icon and entering *Source Selector Control* followed by the Enter key.
5. Set its number to *1*.
6. Double click the Selector block in the Processing Map to open its property dialog
7. Link the new Selector control in the External Control Systems dialog to the selector block's selector control. Do this by dragging the link icon in the External Control Systems dialog to the right of the control name and dropping it on the link icon for the Selector Block's selector control.
8. If you click on the *Source Selector Control* hyperlink, the property dialog for the *Source Selector Control* link opens. Change the Label in the *Source Selector Control* to "Source". Together, the selector property dialogs are as follows:

CHAPTER 3: Example HAL1x Configuration



Your external control system can now get and set the lounge input source using the External Control Message Protocol.

Next, we'll add a control for the lounge volume level:

1. In the External Control Systems dialog, click on the **Level** tab.
2. Create a new Level control by clicking on the **+Control** button.
3. Change the name of this new control to *Lounge Level Control* and set its number to *6*.
4. Open the property dialog for the lounge output block, HAL1x Line Output (1).
5. Link the new Level external control to the level control in the Line Output block property dialog. Do this by dragging the link icon in the External Control Systems dialog to the right of the control name and dropping it on the link icon for the Line Output Block's level control.
6. If you click on the Lounge Level Control hyperlink, you will see its property dialog. In the Lounge Level Control dialog, change the Label to "Lounge Volume". All three dialogs appear below as follows:



In a Room Combine application, end users typically need to let the audio system know when they open or close a movable wall. To create a Toggle control for this:

1. Open the Room Combine Processor property dialog.
2. Drag both Room A and Room B onto the Layout & Control area and arrange them so that they are next to each other.
3. Add a movable wall between them by dragging the movable wall icon between the rooms until the wall highlights, then dropping the wall to create a wall toggle control.
4. In the External Control Systems dialog, click on the **Toggle** tab.
5. Add a new Toggle control by clicking on the **+Control** button.
6. Change the name of the new control to *Wall Toggle Control* and set its number to 5.
7. Link the new Toggle external control to the wall toggle control in the Room Layout & Control area. Do this by dragging the link icon  in the External Control Systems dialog to the right of the control name and dropping it on the link icon  for the wall toggle control between the two rooms.
8. If you click on the *Wall Toggle Control* hyperlink, you will see its property dialog. Change the Label in the Wall Toggle Control dialog to "Wall Toggle". The dialogs appear below as follows:

CHAPTER 3: Example HAL1x Configuration



At this point, your external control system has access to three controls – a selector to get and set the lounge input program source, a level to control the lounge volume and a toggle to open and close the room combine wall.

Now we'll add some controls and link them to presets of various kinds to show how you can provide access to presets to your external control system.

First, we'll add a toggle control to mute all of the outputs in our configuration:

1. On the Processing Workspace Toolbar, click on the **Presets** button to open the All Presets property dialog.
2. In the All Presets dialog, click on the **Toggle** tab.
3. Add a new toggle preset by clicking on the **+Preset** button.
4. Rename this preset to *Mute All Preset*.
5. In the External Control Systems dialog, click on the **Toggle** tab if it is not already displaying the toggle controls.
6. Add a new Toggle external control by clicking on the **+Control** button and set its number to *4*.
7. Rename this new control to *Mute All Toggle Control*.
8. Link the new Mute All Toggle Control to the toggle control for the Mute All Preset. Do this by dragging the link icon in the External Control Systems dialog to the right of the control name and dropping it on the link icon for the Mute All Preset's toggle control.

9. If you click on the Mute Toggle Control hyperlink, you will see its property dialog. Change the Label in the Mute All Toggle Control dialog to "Mute All". The dialogs appear below as follows:



10. Open the three HAL1x output blocks, set each of the mute checkboxes to checked (muted) and set their level controls all the way down (Off).
11. Add the three HAL1x output blocks to the Mute All Preset. To do this, hover your mouse over the HAL1x output block for the lounge. Drag the blue preset icon  that appears and drop it on the blue rectangle for the Mute All Preset in the All Presets dialog. Repeat for the other two HAL1x output blocks.
12. Test this by activating and deactivating the Mute All Preset from the All Presets dialog. All three HAL1x output blocks mute when the preset is active and un-mute when the preset is not active.

CHAPTER 3: Example HAL1x Configuration



Now, we'll add a Command control to reset all outputs to a default operating state:

1. In the All Presets dialog, click on the **Command** tab.
2. Create a new command preset by clicking on the **+Preset** button.
3. Rename this new preset to *Default Level Preset*.
4. In the External Control Systems dialog, change to the **Toggle** tab if it is not already selected.
5. Open the Mute All Toggle Control's property dialog by clicking on its hyperlink and make sure the **Mute All Toggle Control** toggle is un-checked.
6. Open the three HAL1x output blocks and set the level for each to *-20.0 dB* and un-check the mute toggle control.
7. Add the three HAL1x output blocks to the new Default Level Preset by dragging the blue preset icon next to each HAL1x output block and dropping them on the blue rectangle for the Default Level Preset in the All Presets dialog.
8. In the External Control Systems dialog, drag the blue preset icon for the Mute All Toggle Control and drop it on the blue rectangle for the Default Level Preset in the All Presets dialog. This ensures that the Mute All toggle is reset (unmuted) when the Default Level Preset is asserted.
9. In the External Control Systems dialog click on the **Command** tab.
10. Add a new Command external control by clicking on the **+Control** button.
11. Rename this new control to *Default Command Control* and set its number to *3*.

12. Link the Default command control to the Default Level Preset's Assert button. Do this by dragging the link icon  in the External Control Systems dialog to the right of the control name and dropping it on the link icon  for the Default Level Preset's Assert button.
13. If you click on the Default Command Control hyperlink, you will see its property dialog. Change the Default Command Control Label to "Default Level". The dialogs appear below as follows:



Finally, we'll create a selector preset to provide three options for the PEQ settings in the lounge

1. In the All Presets dialog, click on the **Selector** tab.
2. Create a new Selector preset by clicking on the **+Selector** button.
3. Rename this selector preset to *PEQ Preset Selector*.
4. Add three presets to this selector by clicking the **+Preset** button three times.
5. Change the names of these presets to *Softens Vocals*, *Smooth*, and *Boost Bass*.
6. Activate the *Softens Vocals* preset by clicking on the radio button to the left of its name in the All Presets dialog. Then uncheck the *Include 'No Selector'* checkbox at the bottom of the PEQ Selector Preset.
7. Open the property dialog for the PEQ block and set the Gain to *-15.0* and the Frequency to *3276*.
8. Add the PEQ block to the *Softens Vocals* preset – that is the first preset in the selector preset. To do this, drag the blue Preset icon next to the PEQ block in the processing map and drop it on the blue '*<Add>*' rectangle for the Softens Vocals preset.
9. Set the PEQ frequency to *1589* and add the PEQ block to the Smooth preset.

CHAPTER 3: Example HAL1x Configuration

10. Set the PEQ frequency to 393, the Gain to +12.0 and add the PEQ block to the Boost Bass preset.
11. Test the preset selector by clicking on the Active radio buttons and observing the PEQ change.
12. In the External Control Systems dialog click on the **Select** tab.
13. Create a new Selector external control by clicking on the **+Control** button.
14. Rename this new control to *PEQ Selector Control* and set its number to 2.
15. Link the PEQ Selector Control to the PEQ Preset Selector control in the All Presets dialog. Do this by dragging the link icon  in the External Control Systems dialog to the right of the control name and dropping it on the link icon  for the PEQ Preset Selector control.
16. If you click on the PEQ Selector Control hyperlink, you will see its property dialog. Change the PEQ Selector Control Label to "Tone". The dialogs appear below as follows:



Wow, that's the whole configuration. Be sure to save your configuration before quitting Halogen.

Configuration Summary

The example configuration provides the following external controls:

1. A Selector for lounge program source – control number 1

There are three selections: Classical, Jazz, and Reggae

2. A Selector for lounge tone (PEQ) – control number 2

There are three selections: Soften Vocals, Smooth, and Boost Bass.

3. A Level for lounge volume – control number 6

This changes the output block's level for the lounge.

4. A Toggle to mute all audio outputs – control number 4

This mutes all output blocks and sets their levels to Off. While setting the levels is not necessary, we've included it here to cause the lounge output level to change so the Halogen/HAL Control Server will send a 'set level' message.

5. A Toggle to open and close the wall of a room combine block – control number 5

6. A Command to reset audio outputs to default levels – control number 3

This sets the three outputs to -20.0dB and un-mutes them. In addition, it sets the Mute All toggle control to unchecked so the state is consistent with the outputs being un-muted.

CHAPTER 4: Integrating AMX Control Systems with HAL

Overview

This guide describes how to set up an AMX controller and touch panel to communicate with a Halogen/HAL Control Server. It is designed to be used in conjunction with the files found in the AMX Support Package. The contents of this support package include:

- AMXControlSystems_Guide.pdf - The guide you are reading now
- ControlSystemSample.hal - Halogen configuration file intended for loading in Halogen or your HAL1x
- Rane_HAL.apw - NetLinx Studio program project file
- Main.aks - NetLinx program source file
- Rane_HAL_TP.TP4 - TPDesign4 touch panel project file

The AMX Support Package is installed with the Halogen software and can be accessed from the Windows Start Menu under Rane Corporation > Halogen > Guides > AMX. If you want to access the files directly they are available on the product DVD or from the Halogen install directory:

- Windows 7 or Vista - C:\Program Files (x86)\Rane Corporation\Halogen\Guides\Support Packages\AMX directory
- Windows XP - C:\Program Files\Rane Corporation\Halogen\Guides\Support Packages\AMX directory.

You can also download the most up-to-date version of this support package from the Rane website (<http://rane.com/hal>).

 **NOTE:** You do not need to connect your controller to an actual HAL in order to test your control system. A PC running Halogen can simulate all the feedback your controller would get from a HAL running the same configuration.

The HAL system has four types of controls: toggles, commands, levels and selectors. All four are accessible to external TCP/IP clients like an AMX controller. This guide walks you through the code for a TCP/IP client before looking closely at each of the HAL control types one-by-one. We dive into code for these HAL control types by implementing a series of touch panel applications. These applications are:

- Toggle button to activate and deactivate a preset
- Toggle button to open and close a wall in a room combine block
- Command button to assert a preset
- Slider to adjust the level on an output
- Radio buttons to select between input sources
- Radio buttons to select between presets

The guide concludes with a troubleshooting section that covers the most common problems encountered when configuring a target system for the examples.

Checklist

To use the files from the AMX Support Package and follow along with the examples you will need:

- A Windows desktop or laptop with an Ethernet port and the latest Rane Halogen (1.1.0 or higher) and AMX NetLinx Studio and TPDesign software installed.
- An AMX NetLinx controller. The NI-700 is sufficient.
- An AMX Modero NXT-CV7 touch panel. Alternatively, you could create a new TPDesign project targeting a different touch panel model and copy the example over.
- An Ethernet switch and cables for the PC, controller and touch panel.

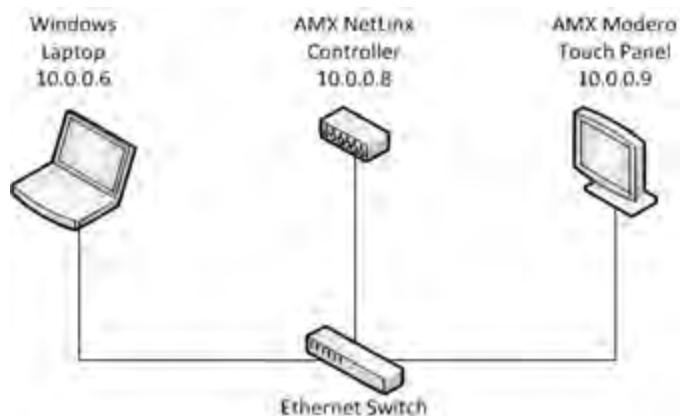
Although you ultimately connect a HAL to your control system, all of the integration and testing can be accomplished without a HAL.

This guide assumes that the reader has had some exposure to AMX programming and is able to navigate the AMX software. A typical workflow for developing an AMX application is:

- Define constants for devices, buttons and levels in NetLinx Studio.
- Code event handlers and feedback statements in NetLinx Studio.
- Compile the program and transfer it to your controller.
- Draw a custom touch panel user interface in TPDesign.
- Assign device channel numbers from the program to buttons and sliders in the user interface.
- Compile the user interface and transfer it to your touch panel[s].

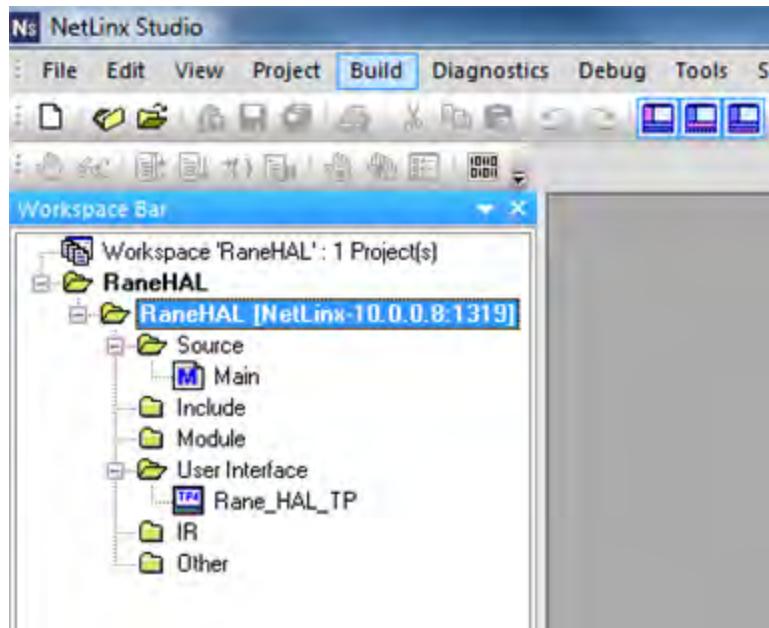
Once your application is running, you can use NetLinx Studio's debugger to set breakpoints, watch variables and step through your program.

Getting Started



This section shows how to set up your system so that you can get the most out of the examples. The three major components of your system are the PC, controller and touch panel. All three need to be on the same network so they can communicate. Configure their IP settings and plug them into the Ethernet switch using the Ethernet cables.

Launch Halogen on the PC so that it can pretend to be a HAL. A virtual HAL starts running on your PC when you launch Halogen. This process goes away when you exit the application. Make sure to have the right configuration file open when testing your system. Use the ControlSystemSample.hal file that came with the AMX Support Package when working with the examples.



Run the example program on the controller so that it can talk to the HAL and the touch panel. Open the Rane_HAL.apw workspace file in NetLinx Studio and select the RaneHAL project as the **Active System**. Next, open the project **Communication Settings** and change the 10.0.0.8 master address to match the IP of your controller. Then build the system and transfer the compiled Main.tkn file to your controller.

Keep the Main.src source file open so you can make changes to it. Find `dvTP` in the `DEFINE_DEVICE` section of the program.

```
dvTP = 10001:1:1
```

This is the touch panel's device address. If your touch panel has a different device address enter it here or change the device address settings on your touch panel.

Apply the example design to the touch panel so that it can talk to the controller. Buttons and indicators in the design are assigned specific channel numbers in the program so the two depend on each other. Open the Rane_HAL_TP.tp4 design file in TPDesign and send the design to the touch panel. If your touch panel is not a Modero NXT-CV7 you will need to create a new design for your panel and paste the contents of the example into it before you can transfer it over.

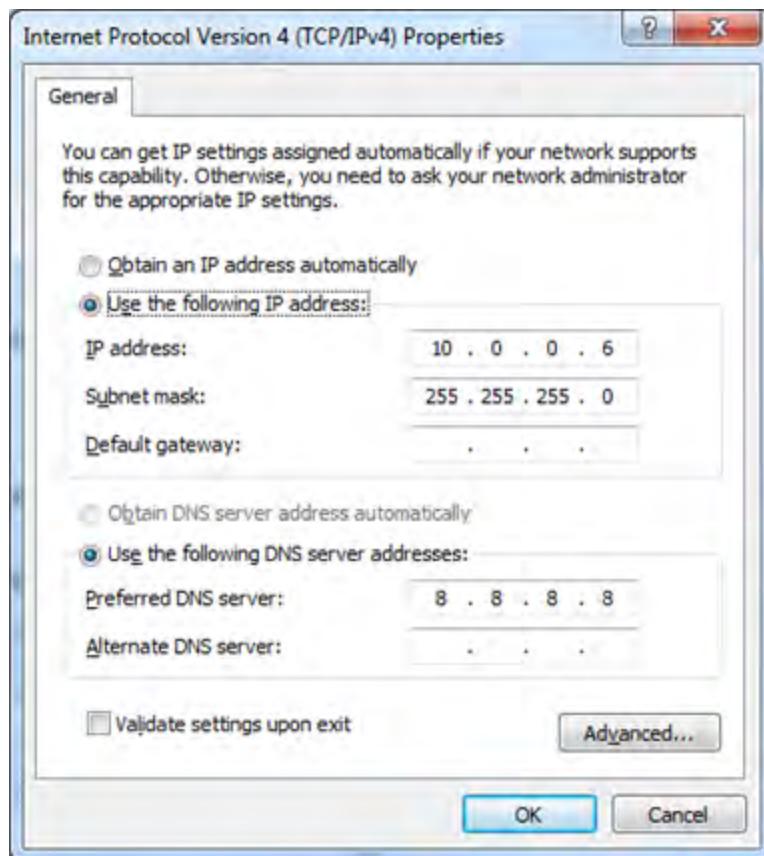
You know all three components of your system are communicating when the connected status indicator on the upper right corner of your touch panel says "Connected". Refer to the Troubleshooting section if you find yourself struggling to establish communications.

Configuring IP Addresses

An AMX controller connects to a HAL or PC running Halogen via TCP/IP. The easiest way to accomplish this connectivity is by assigning the HAL and PC a static IP address.

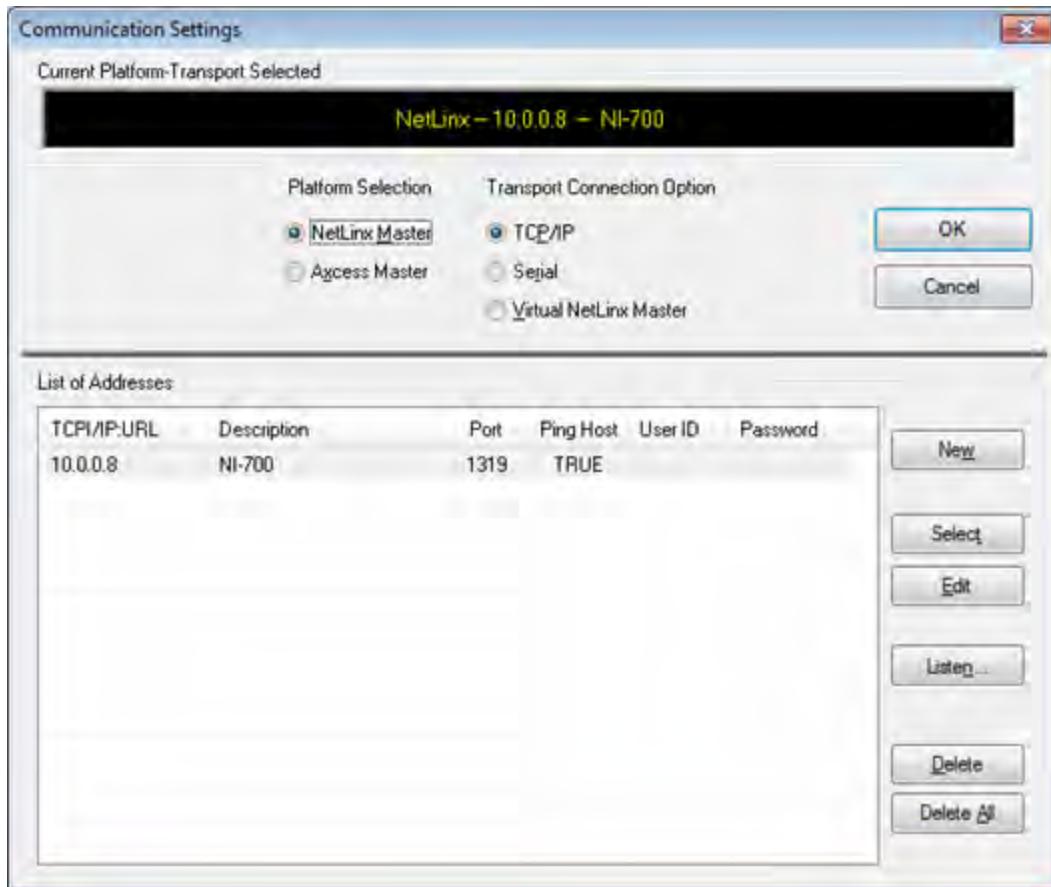


You configure a HAL's static IP addresses in the **Host Device Settings** dialog in the Hardware workspace. To access this dialog, connect to your HAL then hover your mouse over the **HAL bar** at the top of the **Hardware** Map, then click on the gear icon that appears. This dialog is only available when Halogen is connected to a HAL. Host Device Settings like static IP assignments reside solely on that HAL and do not travel with the Halogen configuration file.



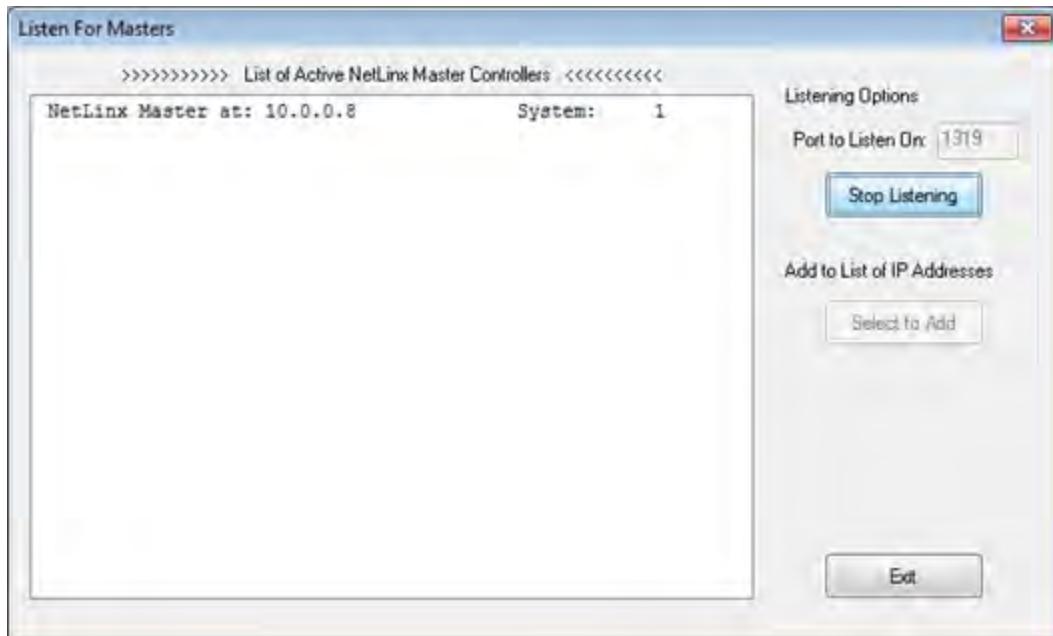
PCs are frequently setup to use dynamic IPs but a static IP is preferable if you want to use your PC in place of a HAL during development. The controller tries to connect to a TCP server (your PC in this case) at a specific IP address. If your PC has a dynamic IP address assigned to it, then that address may change if it is disconnected from the network or the DHCP lease expires. You also probably do not want to configure your PC with the same static IP as your HAL in the event that both are connected to the same network.

Control system networks are typically private networks with no direct route to the Internet except through a gateway or proxy server. Certain IP address ranges are reserved for private networks. 10.0.0.0 to 10.255.255.255 is one of those address ranges. Start with a small self-contained network. Plug your PC, controller, touch panel and HAL into the same Ethernet switch and assign each one a static IP in the 10.0.0.0 network.

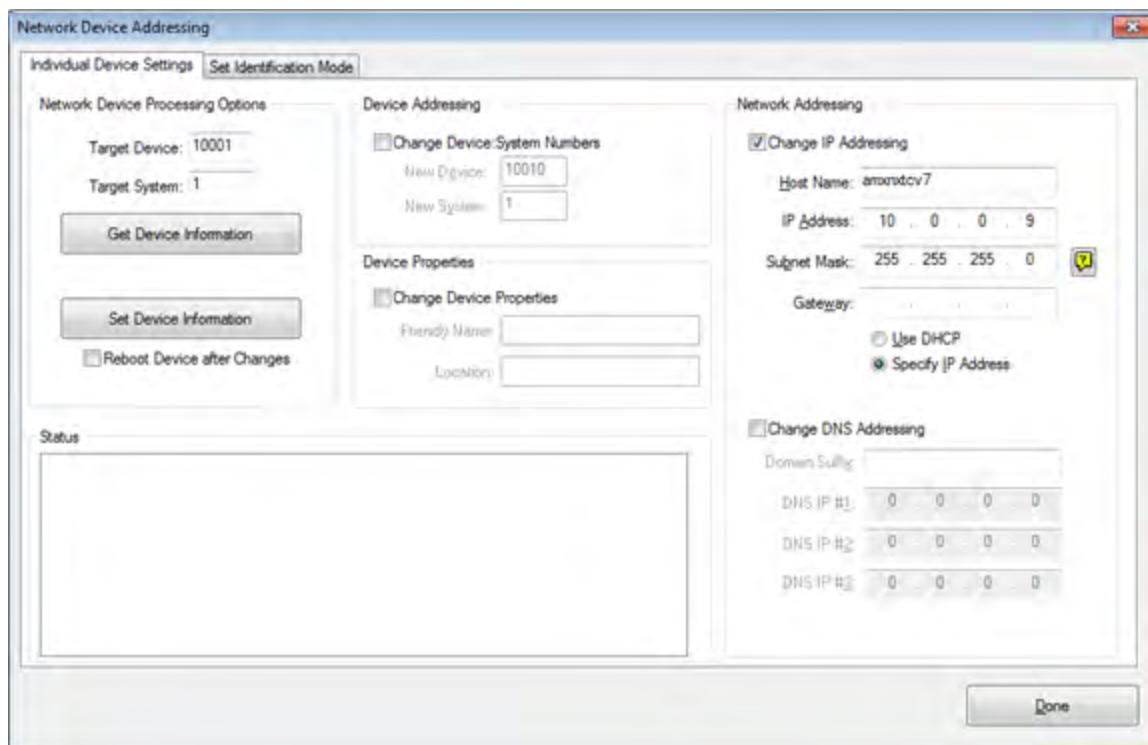


To change the IP address of your controller NetLinx Studio must first connect to it. This can be done via Ethernet rather than serial RS-232 if the controller already has an IP address. NetLinx Studio provides a handy tool for finding the IP address of your controller. Click the Listen button in the **Communication Settings** dialog to bring up this **Listen For Masters** tool. Once you know your controller's current IP address you can then add an entry with that IP to the **Address List** in the Communication Settings dialog.

CHAPTER 4: Integrating AMX Control Systems with HAL



The AMX touch panel also has an IP address. To see it right-click on NX-CV7 or whichever model touch panel you in the **Online Tree** and select **Network Device Addressing**. Set the **Target Device** address of the touch panel to *10001* and Target System to *1* so that it matches the example program. You can also change the IP address settings of the touch panel from this dialog by checking the **Change IP Addressing** box. Press the **Set Device Information** button to push the new settings out to the touch panel.



TCP/IP Client

An AMX controller needs a TCP/IP client to communicate with the HAL. This client exchanges string messages with the HAL using the "HAL External Control Message Protocol" on page 45. The example program included in the AMX Support Package provides a reference implementation of such a client. Let's take a look at it.

Open the Main.axs source file in NetLinx Studio. Find the dvIPClient and dvHAL device addresses in the `DEFINE_DEVICE` section of the program. There you will see:

```
dvIPClient = 0:2:1  
dvHAL = 0:FIRST_LOCAL_PORT:0
```

Device addresses in AMX are comprised of a device number, a port number and a system number. The dvIP-Client definition is the device address of the TCP/IP client. The dvHAL definition is the device address of the HAL itself. Their device number is 0 indicating that their ports reside on the NetLinx master controller.

Find the `DEFINE_CONSTANT` section of the program. There you will see four TCP/IP client-related constants:

```
sRemoteIP = '10.0.0.6'  
nRemotePort = 4996  
nLocalPort = FIRST_LOCAL_PORT  
nTCP = 1
```

The `sRemoteIP` constant is the IP address of the Control Systems server. This server could be running on a HAL or it could be running on your PC when Halogen is open and not connected to a HAL. We'll assume you will test your control system using Halogen, so change this IP address to match the IP address of your PC for now. The transport protocol is always TCP and the remote port is always 4996 to match the TCP port that the Halogen/HAL Control Server listens on.

The `nLocalPort` constant is the local port on the controller that identifies a client connection. A NetLinx controller's available local ports start at `FIRST_LOCAL_PORT`. Since this program supports no more than one connection at a time the same local port can be used every time the controller tries to open a connection. If you want more simultaneous connections you will need to define more local ports like: `FIRST_LOCAL_PORT + 1`, `FIRST_LOCAL_PORT + 2`, etc.

Find the `DEFINE_START` section of the program. It contains the controller's first attempt to open a client connection upon startup.

```
ip_client_open(nLocalPort, sRemoteIP, nRemotePort, nTCP)
```

Connection retries happen in the Mainline when the client is disconnected. The Mainline is located in the `DEFINE_PROGRAM` section of the program. The code in the Mainline executes whenever the controller is idle (not processing an event handler).

Find the `DATA_EVENT` handler in the program. This is the event handler for any data the client receives from the HAL.

```

DATA_EVENT [dvHAL]
{
    ONLINE:
    {
        ...
        bConnected = 1
        ...
    }
    OFFLINE:
    {
        bConnected = 0
        ip_client_close(nLocalPort)
    }
    STRING:
    {
        ...
    }
    ...
}

```

The ONLINE event is triggered when an `ip_client_open` call succeeds in connecting to the HAL. The OFF-LINE event is raised when the connection to the HAL is dropped. A variable named `bConnected` keeps track of the client's connected state. This variable sends feedback to a connection status indicator on the touch panel.

The ONLINE event is handled by requesting the live values of various controls from the HAL. This initial query ensures that the values displayed on the touch panel reflect the current state of the HAL.

Any actual data that the client receives from the HAL gets processed in the STRING event handler. This handler contains code to parse HAL control values that are sent back in response to a query from the IP client, or that arrive unsolicited because some other client changed or requested them. Refer to the button and level event handlers later on in this guide for code to build and send HAL control values.

Toggles

This section explains how to integrate HAL toggle controls with buttons on an AMX touch panel by examining a couple of applications: a toggle preset button and a wall toggle button on a touch panel. We start by dissecting the code for the HAL toggle controls and touch panel toggle buttons. Then we demonstrate how to bind HAL toggle controls to buttons in the touch panel user interface.

Toggles in a HAL system have only an on or off state. They are represented as checkboxes in the Halogen software and DR remotes. Because toggles only have two states they are modeled as integers with values of 0 and 1 in NetLinx. The example program provides code for handling toggle button events from the touch panel and parsing toggle message strings from a HAL. Let's take a look at it.

Open the Main. axs source file in NetLinx Studio. Find the `DEFINE_CONSTANT` section of the program. It contains some toggle-related constants:

```

BTN_MUTE_ALL = 4
BTN_WALL_OPEN = 5

```

...

```
INTEGER nToggleChannels[] = { BTN_MUTE_ALL, BTN_WALL_OPEN }
```

This code says that there are two HAL toggle controls in this program and that their touch panel device channel numbers are *4* and *5*. The *4* and the *5* are also HAL control numbers used to uniquely identify controls on the HAL. These control numbers appear in the **External Control Systems** dialog of the Halogen software. Matching device channel numbers with HAL control numbers makes programming multiple HAL toggle controls easy.

The example toggle code is reusable across applications. If you want more toggle buttons on the touch panel just add more device channel numbers to the *nToggleChannels* array. Coding a new toggle application is simply a matter of matching HAL toggle control numbers in Halogen with newly-added toggle button device channel numbers in AMX.

There is one button event handler for all of the toggle buttons on the touch panel. Feel free to copy and paste event handler into your own programs.

```
BUTTON_EVENT [dvTP, nToggleChannels]
{
    RELEASE:
    {
        nToggleIndex = GET_LAST(nToggleChannels)

        if (nToggleIndex > 0 &&
            nToggleIndex <= length_array(nToggleChannels))
        {
            bToggleValues[nToggleIndex] =
                !bToggleValues[nToggleIndex]

            Send_String dvHAL, "'<T&',
                itoa(nToggleChannels[nToggleIndex]), '&',
                itoa(bToggleValues[nToggleIndex]), '>''"
        }
    }
}
```

A button event handler does not have to take a single device channel as its channel reference parameter. It can accept an array of device channel numbers. In the case of our touch panel's toggle button event handler, all of the toggle buttons' device channel numbers are passed in through the *nToggleChannels* argument. The handler knows which toggle button on the touch panel fired the event because the *GET_LAST* call returns the index of that button in the *nToggleChannels* array.

Whenever a toggle button on a touch panel is released the toggle button event handler sends a message string with a new toggle value to the HAL. These 'set toggle' messages contain a toggle control type, HAL control number and value. Valid 'set toggle' messages for the two HAL toggle controls in this example would be *<T&4&0>*, *<T&4&1>*, *<T&5&0>* and *<T&5&1>*.



NOTE: Refer to Appendix A "HAL External Control Message Protocol" on page 45 for more information on toggle messages.

CHAPTER 4: Integrating AMX Control Systems with HAL

The HAL sends these same ‘set toggle’ messages to the controller when another client changes a HAL toggle control value. Here is the STRING event handler code to parse incoming HAL toggle control values.

```
active(find_string(cData, '<T&', 1)):  
{  
    remove_string(cData, '<T&', 1)  
  
    nToggleIndex = FindToggleIndex(atoi(cData))  
  
    remove_string(cData, '&', 1)  
  
    if (nToggleIndex > 0 &&  
        nToggleIndex <= length_array(nToggleChannels))  
    {  
        bToggleValues[nToggleIndex] = atoi(cData)  
    }  
    remove_string(cData, '>', 1)  
}
```

The Mainline contains feedback statements for all the defined toggles. These feedback statements update the visible state of toggle buttons on the touch panel in response to HAL toggle control value changes triggered by touch panel button presses or messages from the HAL.

```
for (nToggleIndex = 1;  
     nToggleIndex <= length_array(nToggleChannels);  
     nToggleIndex++)  
{  
    [dvTP, nToggleChannels[nToggleIndex]] =  
        bToggleValues[nToggleIndex]  
}
```

The controller requests HAL toggle control values from the HAL upon connecting. This ensures that the toggle buttons on the touch panel are in sync with the live toggle values on the HAL. The controller does this by sending ‘get toggle’ messages to the HAL when the ONLINE event is fired. These ‘get toggle’ messages consist of a toggle control type and HAL control number. See the toggles for loop in the client’s ONLINE event handler for how to encode and send them.

```
for (i = 1;  
     i <= length_array(nToggleChannels);  
     i++)  
{  
    Send_String dvHAL, "'<T&', itoa(nToggleChannels[i]), '>'"  
}
```

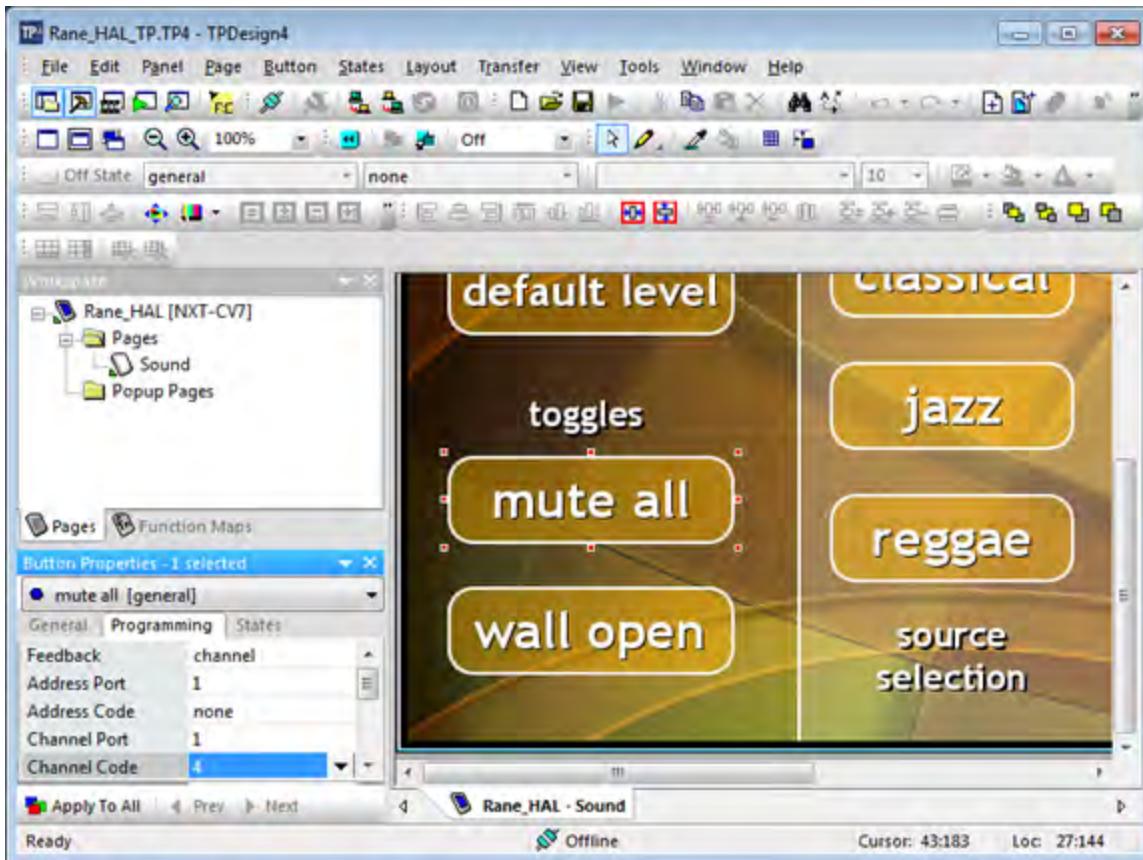
The ‘get toggle’ messages for this example are <T&4> and <T&5>. Upon receipt of these ‘get toggle’ messages the HAL responds with ‘set toggle’ messages for those two HAL toggle controls.

Toggle Presets

This example demonstrates how to get a toggle button on a touch panel to activate and deactivate a preset on a HAL. The example Halogen configuration file contains a toggle preset for muting and unmuting all the outputs on the HAL. This Mute All Preset is linked to a HAL Control Systems toggle control named Mute All Toggle Control. Toggling the HAL toggle control activates or deactivates the preset.



Open the `ControlSystemSample.hal` example configuration file in Halogen. Switch to the **Processing** workspace by clicking on the **Processing** tab. Next, open the **External Control Systems** dialog and select the **Toggle** tab. Notice that the *Mute Toggle Control* is assigned a HAL control number of 4. For this application to work, the *Mute Toggle Control* number needs to match the device **Channel Code** of the **Mute All** button on the touch panel. This is what is meant by matching HAL control numbers with device channel numbers.



Open the Rane_HAL_TP.tp4 design file in TPDesign. Next open the **Sound** page window to view the example touch panel design. Select the **Mute All** button in the Sound page window then select the **Programming** tab in **Button Properties**. Notice that the **Channel Code** for the Mute All button on the touch panel is set to **4** to match the HAL *Mute Toggle Control* number and the **Feedback** is set to **channel** for latching. The Mute All button appears lighter on the touch panel when the channel is off and darker when the channel is on.

Verify that the **Connected** status indicator in the upper right corner of the touch panel says **Connected**. Refer to the Troubleshooting section if it says **Disconnected**. Press the **Mute All** button on the touch panel. The button will remain in its new on or off state after you lift your finger. This is what is meant by latching. Watch the **Active** state of the *Mute All Preset* in Halogen as you press the button on the touch panel. Then toggle the *Mute All Preset* from Halogen to see that the Mute All button and **Lounge** bar graph on the touch panel respond.

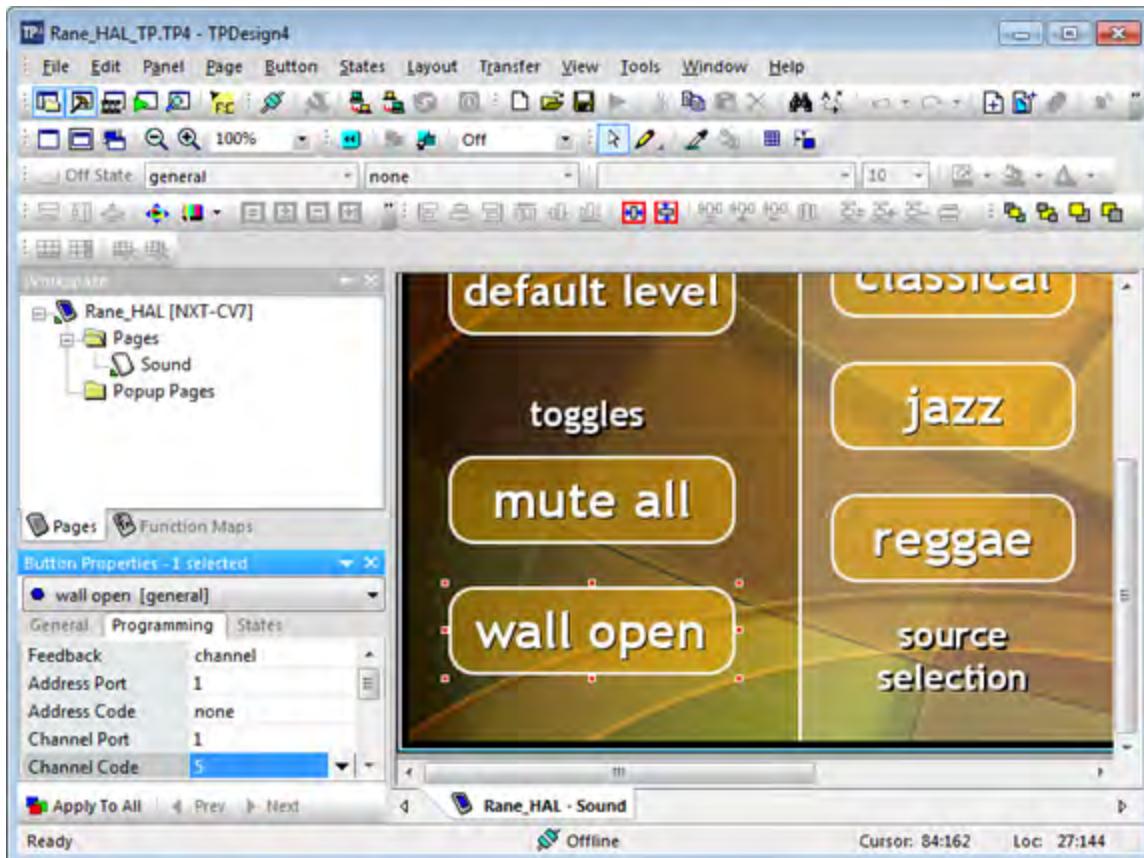


NOTE: Refer to Appendix B "Using PuTTY to Test External Control Systems" on page 55 for help with debugging if the system does not behave as described here.

Wall Toggles

This example demonstrates how to get a toggle button on a touch panel to open and close a movable wall in a HAL room combine block. The example Halogen configuration file contains a room combine block with two rooms A and B. The movable wall in between these two rooms is linked to a HAL Control Systems toggle control named *Wall Toggle Control*. Toggling the HAL toggle control opens and closes the wall.

Open the ControlSystemSample.hal example configuration file in Halogen. Switch to the **Processing** workspace by clicking on the **Processing** tab. Next, open the **External Control Systems** dialog and select the **Toggle** tab. Notice that the *Wall Toggle Control* is assigned a HAL control number of 5. For this application to work, the HAL *Wall Toggle Control* number needs to match the device **Channel Code** of the Wall Open button on the touch panel.



Open the Rane_HAL_TP.tp4 design file in TPDesign. Next, open the **Sound** page window to view the example touch panel design. Select the **Wall Open** button in the Sound page window then select the **Programming** tab in **Button Properties**. Notice that the **Channel Code** for the Wall Open button is set to 5 to match the HAL *Wall Toggle* number and the **Feedback** is set to channel for latching.

Touch panel buttons can display different text for their on and off states. That often makes a lot of sense in the context of toggle buttons. Double-click the **Off** state image for the **Wall Open** button in the **State Manager**. This will pull up the **Off** state **Button Properties** for that button. From there you can change that button's **Off** state text to say "wall closed" if you prefer.

Verify that the **Connected** status indicator in the upper right corner of the touch panel says **Connected**. Refer to the Troubleshooting section if it the status indicator says **Disconnected**. In Halogen, double-click the *Room Combine (1)* block in the Halogen **Processing** map then select the **Layout and Control** tab in the *Room Combine (1)* dialog. Watch the wall between rooms A and B open and close in Halogen as you press the **Wall Open** button on the touch panel open and closed. Now toggle the wall open and closed from the *Room Combine (1)* dialog in Halogen and watch that same button turn on and off on the touch panel.



NOTE: Refer to Appendix B "Using PuTTY to Test External Control Systems" on page 55 for help with debugging if the system does not behave as described here.

Commands

This section explains how to integrate a HAL command control with a button on an AMX touch panel by examining a simple application: a touch panel button that asserts a HAL preset. We start by dissecting the code for the HAL command control and touch panel command button. Then we demonstrate how to bind the HAL command control to a button in the touch panel user interface.

Command controls are stateless in a HAL system. Unlike toggles they have no on or off values. Command controls are merely triggers for predefined actions like asserting presets. Think of them as "Fire and forget." Because they are stateless the Feedback property for a command button on a touch panel must be set to momentary or blink. The example program provides code for handling command button events from a touch panel. Let's take a look at it.

Open the Main.axs source file in NetLinx Studio. Find the `DEFINE_CONSTANT` section of the program. It contains some command-related constants:

```
BTN_DEFAULT_LEVEL = 3

...
INTEGER nCommandChannels[] = { BTN_DEFAULT_LEVEL }
```

This code says that there is one HAL command control in this program and that its touch panel device **Channel Code** is 3. The 3 is also a unique control number on the HAL. These control numbers appear in the **External Control Systems** dialog of the Halogen software. Matching device Channel Codes with HAL control numbers makes programming multiple HAL command controls easier.

The example command code is reusable across applications. If you want more command buttons on your touch panel just add more device channel numbers to the `nCommandChannels` array. Coding a new command application is simply a matter of matching HAL command control numbers in Halogen with newly-added button device channel numbers in AMX.

There is one button event handler for all of the command buttons on the touch panel. Feel free to copy and paste this event handler into your own programs:

```
BUTTON_EVENT [dvTP, nCommandChannels]
{
    PUSH:
    {
        nCommandIndex = GET_LAST(nCommandChannels)

        if (nCommandIndex > 0 &&
            nCommandIndex <= length_array(nCommandChannels))
        {
            Send_String dvHAL, "'<C&',
                            itoa(nCommandChannels[nCommandIndex]), '>'"
        }
    }
}
```

{}

Whenever a command button is pressed on the touch panel this handler sends a message string to the HAL. These 'command' messages contain a command control type and a HAL control number. The only valid 'command' message for the command in this example would be <C&3>.

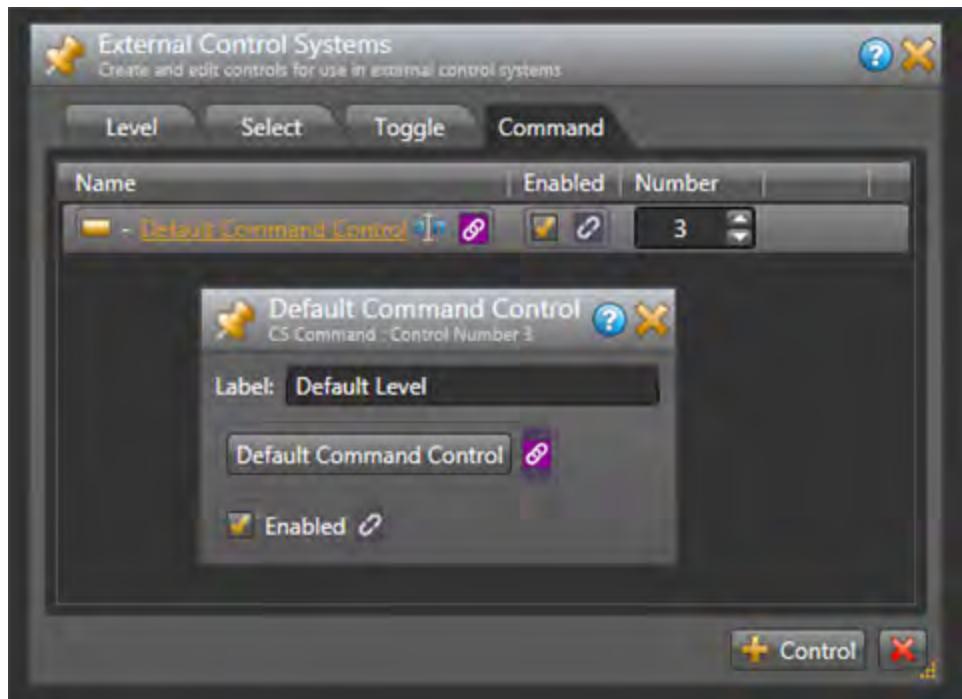


NOTE: Refer to Appendix A "HAL External Control Message Protocol" on page 45 for more information on command messages.

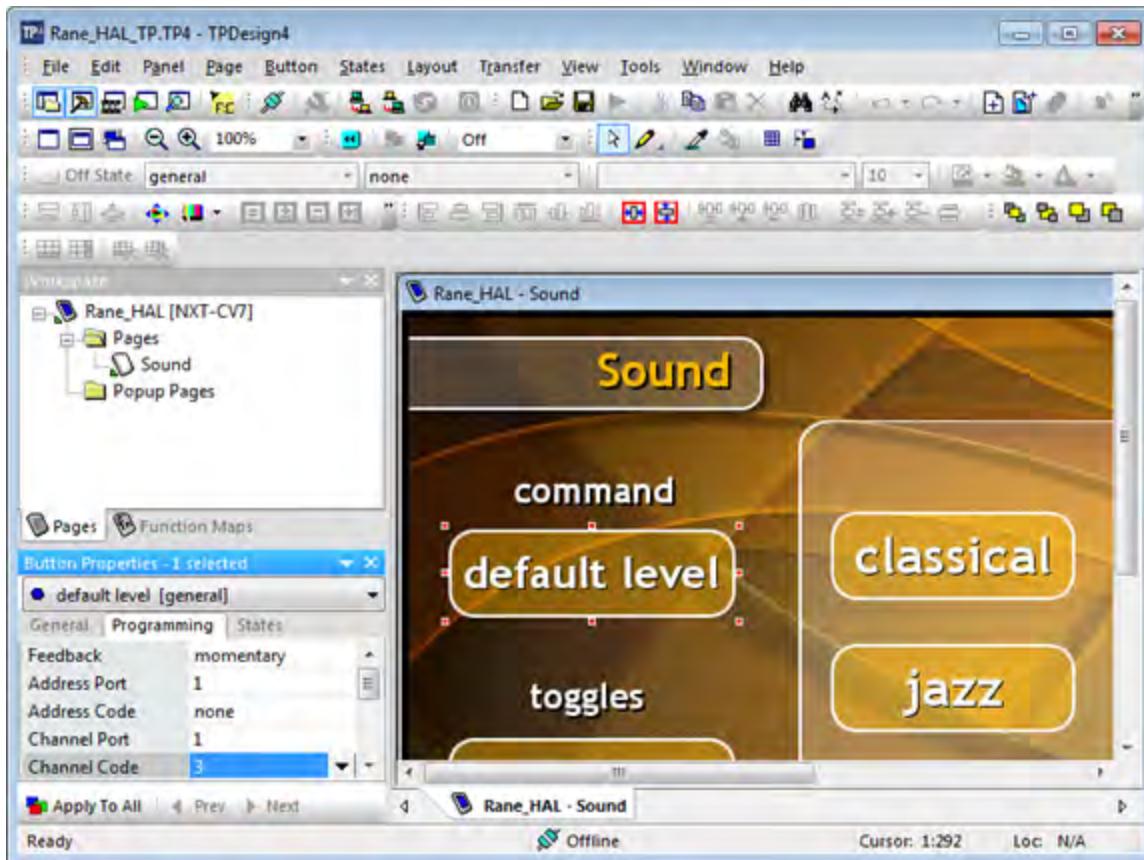
When a client sends a 'command' message the HAL echoes it back to all its other clients. The controller currently ignores unsolicited command messages but you can add code to parse them if you want your touch panel to exhibit feedback when they arrive. See the STRING section of the TCP/IP client's DATA_EVENT handler for parsing techniques.

Command Presets

This example demonstrates how to get a button on a touch panel to trigger a command preset on a HAL. The example Halogen configuration contains a command preset that sets the levels of all the HAL line outputs to a default level of -40.0 dB and un-mutes them. A preset like this is handy when a user wants to revert undesired audio settings back to their original defaults. The preset is linked to a HAL Control Systems command control named *Default Command Control*. Invoking this HAL command control asserts the preset.



Open the ControlSystemSample.hal example configuration file in Halogen. Next, open the **External Control Systems** dialog and select the **Command** tab. Notice that the *Default Command Control* is assigned a control number of 3. For this application to work, the *Default Command Control* number needs to match the device **Channel Code** of the **Default Level** button on the touch panel.



Open the Rane_HAL_TP.tp4 design file in TPDesign. Next open the **Sound** page window to view the example touch panel design. Select the **Default Level** button in the Sound page window then select the **Programming** tab in **Button Properties**. Notice that the **Channel Code** is set to 3 just like the control number in Halogen and that **Feedback** is set to momentary so that the touch panel button behaves like a trigger.

Verify that the **Connected** status indicator in the upper right corner of the touch panel says **Connected**. Refer to the Troubleshooting section if it the status indicator says **Disconnected**. Drag the **Lounge** slider on the touch panel up to about 90%. Now press the **Default Level** button on the touch panel. The Lounge slider jumps to 50%. Adjust the Lounge slider down to about 10% then assert the *Default Level Preset* from Halogen. The Lounge slider on the touch panel jumps back to the center again.



NOTE: Refer to Appendix B "Using PuTTY to Test External Control Systems" on page 55 for help with debugging if the system does not behave as described here.

Levels

This section explains how to integrate HAL level controls with sliders on an AMX touch panel by examining a volume slider on a touch panel. We start by dissecting the code for the HAL level control and touch panel slider. Then we demonstrate how to bind the HAL level control to the slider on the touch panel user interface.

The HAL level controls that are accessible to a controller have a value range of 0 to 1000. Compare that with the 0 to 255 range of a level control in AMX. This discrepancy requires that an AMX controller convert the level values that it gets from and sends to Halogen/HAL. HAL levels map naturally to "active" bar graphs on AMX

touch panels. These touch-sensitive bar graphs act as sliders so this guide refers to them as such. The example program provides code for handling level events and parsing level strings from Halogen/HAL. Let's take a look at it.

Open the Main. axs source file in NetLinx Studio. Find the `DEFINE_CONSTANT` section of the program. It contains some level-related constants:

```
LVL_LOUNGE_VOLUME = 6

...
INTEGER nLevelChannels[] = { LVL_LOUNGE_VOLUME }
```

This code says that there is one HAL level control in this program and that its touch panel device **Level Code** is 6. The 6 is also a unique control number on the HAL. These control numbers appear in the **External Control Systems** dialog of the Halogen software. Matching device channel numbers with control numbers makes programming multiple HAL level controls easier.

The example level code is reusable across applications. If you want more sliders on your touch panel just add more device channel numbers to the `nLevelChannels` array. Coding a new level application is simply a matter of matching HAL level control numbers in Halogen with newly-added slider device **Level Codes** in AMX.

There is one level event handler for all of the sliders on the touch panel. Feel free to copy and paste this event handler into your own programs:

```
LEVEL_EVENT[dvTP, nLevelChannels]
{
    nLevelIndex = GET_LAST(nLevelChannels)

    if (nLevelIndex > 0 &&
        nLevelIndex <= length_array(nLevelChannels) )
    {
        if (LEVEL.VALUE <> nLevelValues[nLevelIndex])
        {
            nLevelValues[nLevelIndex] = LEVEL.VALUE

            Send_String dvHAL, "'<L&',
                                itoa(nLevelChannels[nLevelIndex]), '&',
                                itoa(nLevelValues[nLevelIndex] * 1000/255),
                                '>'"
        }
    }
}
```

When a slider on a touch panel is dragged the level event handler sends a message string with a new level value to the HAL. These 'set level' messages contain a level control type, HAL control number and level value. Notice the 1000/255 multiplier for the conversion from AMX to HAL level values. Valid 'set level' messages for the HAL level control in this example program would be `<L&6&0>`, `<L&6&1>`, `<L&6&2>` all the way to `<L&6&1000>`.

The HAL sends these same set level strings to the controller whenever another client changes a HAL level control value. Here is the `STRING` event handler code to parse the incoming level values:

```

active(find_string(cData, '<L&', 1)):
{
    remove_string(cData, '<L&', 1)

    nLevelIndex = FindLevelIndex(atoi(cData))

    remove_string(cData, '&', 1)

    if (nLevelIndex > 0 &&
        nLevelIndex <= length_array(nLevelChannels))
    {
        nLevelValues[nLevelIndex] = atoi(cData) * 255/1000

        send_level dvTP, nLevelChannels[nLevelIndex],
                    nLevelValues[nLevelIndex]
    }

    remove_string(cData, '>', 1)
}

```

Notice the 255/1000 multiplier for the conversion from HAL to AMX level values. A `send_level` call updates the slider on the touch panel to the new level that came from the HAL. Unlike toggles and selectors, there are no feedback statements for levels in the Mainline.

The controller requests HAL level control values from the HAL upon connecting. This ensures that the sliders on the touch panel are in sync with the live level value on the HAL. It does this by sending ‘get level’ messages to the HAL when the `ONLINE` event is fired. These ‘get level’ messages consist of a level control type and HAL control number. See the levels for loop in the client’s `ONLINE` event handler for how to encode and send them:

```

for (i = 1;
     i <= length_array(nLevelChannels);
     i++)
{
    Send_String dvHAL, "'<L&',
                      itoa(nLevelChannels[i]), '>'"
}

```

The ‘get level’ message for the level in this example is `<L&6>`. Upon receipt of this ‘get level’ message the HAL responds with a ‘set level’ message for that HAL level control.



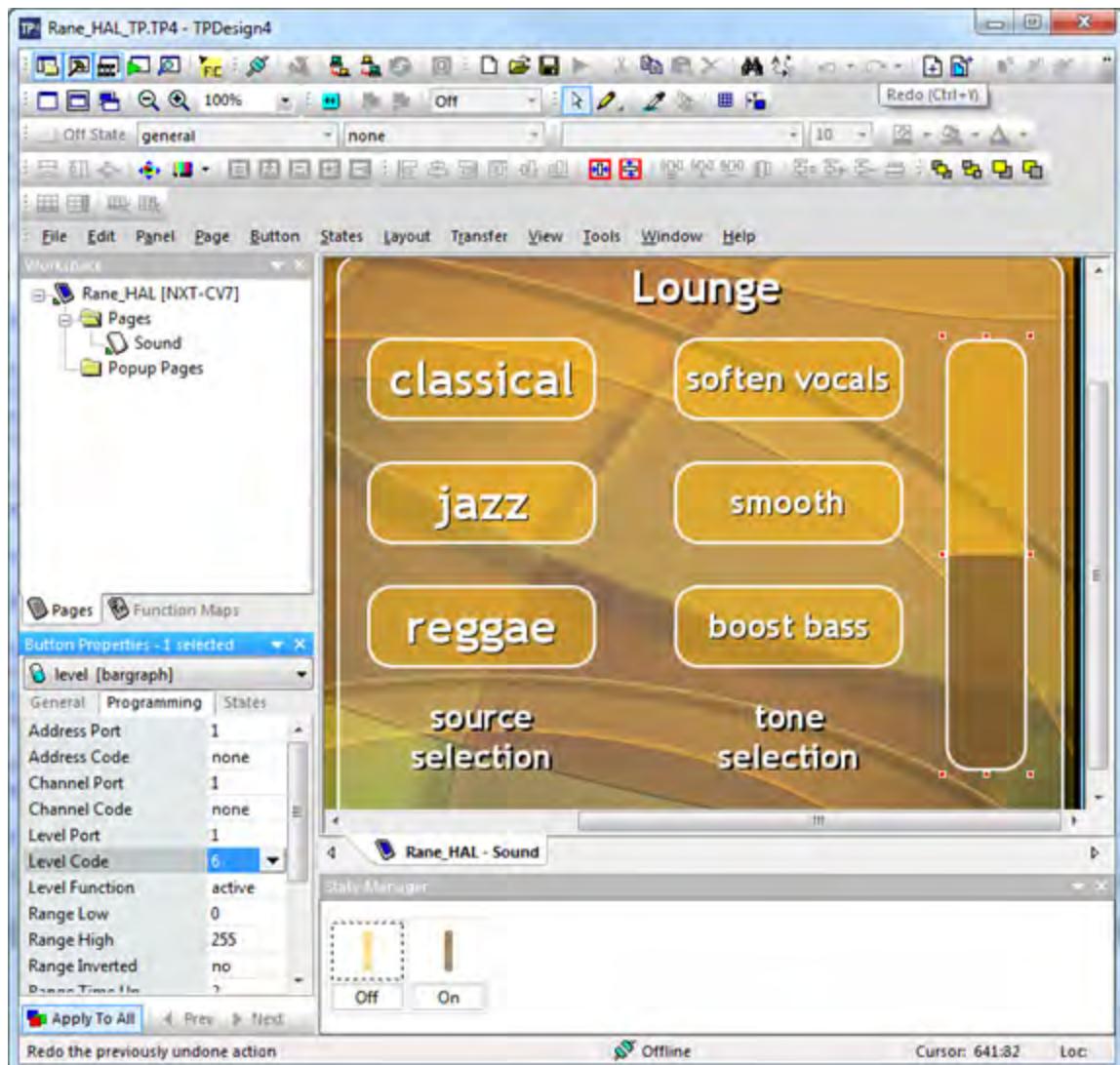
NOTE: Refer to Appendix A "HAL External Control Message Protocol" on page 45 for more information on level messages.

Sliders

This example demonstrates how to get a slider on a touch panel to manipulate the level on a HAL output. The example Halogen configuration contains a HAL line output wired to a lounge. The remaining HAL line outputs go to rooms A and B. The level on the lounge output is linked to a HAL Control Systems level control named *Lounge Level Control*. Adjusting the HAL level control alters the level.



Open the `ControlSystemSample.hal` example configuration file in Halogen. Next, open the **External Control Systems** dialog and select the **Level** tab. Notice that the *Level Command Control* is assigned a HAL control number of 6. For this application to work, the *Lounge Level Control* number needs to match device **Level Code** of the **Lounge** slider on the touch panel.



Open the Rane_HAL_TP.tp4 design file in TPDesign. Next open the **Sound** page window to view the example touch panel design. Select the Lounge bar graph near the right edge of the Sound page window, then select the **Programming** tab in **Level Properties**. Notice that the **Level Code** is set to 6 just like the HAL control number and the **Level Function** is set to active so that the bar graph acts as a slider.

Verify that the **Connected** status indicator in the upper right corner of the touch panel says **Connected**. Refer to the Troubleshooting section if it the status indicator says **Disconnected**. Double-click the *HAL Line Output (1)* block in the Halogen **Processing** map to bring up its properties dialog. Press on the **Lounge** bar graph on the touch panel and slide your finger up and down to adjust the level continuously. The gain for the *HAL Line Output (1)* in Halogen follows your finger movement. Now drag the slider for the level on *HAL Line Output (1)* back and forth in Halogen. The bar graph on the touch panel rises and falls in tandem with the level.

 **NOTE:** Refer to Appendix B "Using PuTTY to Test External Control Systems" on page 55 for help with debugging if the system does not behave as described here.

Selectors

This section explains how to integrate HAL selector controls with buttons on an AMX touch panel by examining a couple of touch panel applications: input source selector radio buttons and preset selector radio buttons. We start by dissecting the code for the HAL selector controls and radio buttons. Then we demonstrate how to bind the HAL selector controls to radio buttons in the touch panel user interface.

At its simplest, a HAL selector control is just a selection integer value. The range of this selection value is constrained by the number of positions on the selector. For instance a HAL three-way selector can only have a value of 0, 1 or 2. Lets look at one way to code a fixed selector.

Open the Main.axs source file in NetLinx Studio. Find the `DEFINE_CONSTANT` section of the program. It contains some constants for a three-way source selector implemented as three radio buttons:

```
BTN_CLASSICAL = 10
BTN_JAZZ = 11
BTN_REGGAE = 12
...
nSourceSelNumber = 1
INTEGER nSourceSelChannels[] = { BTN_CLASSICAL, BTN_JAZZ, BTN_REGGAE }
```

10, 11 and 12 are the device **Channel Codes** of the source selector's three touch panel radio buttons. *1* is the unique control number of this source selector on the HAL. Unlike the preceding toggle, command and level code, the selector's device Channel Codes do not double as HAL control numbers. This is because there is not a one-to-one relationship between radio buttons and a selector controls.

The three-way tone selector has similar constants. `BTN_SOFTEN_VOCALS`, `BTN_SMOOTH` and `BTN_BOOST_BASS` are the device channel numbers of the tone selector's three radio buttons. *2* is the tone selector's unique control number on the HAL.

Find the `DEFINE_VARIABLE` section of the program. It contains some source-selector-related variables:

```
nSourceSelValue
bSourceSelRadios[max_length_array(nSourceSelChannels)]
```

The selection value of the source selector is kept in `nSourceSelValue`. The on/off states of the source selector's three radio buttons are maintained in `bSourceSelRadios` array. We'll see shortly that the source selector in this example is modeled as a collection of mutually exclusive toggles.

The source selector has a dedicated event handler for its touch panel radio buttons. You will need to customize it for your own fixed selectors:

```
BUTTON_EVENT [dvTP, nSourceSelChannels]
{
```

CHAPTER 4: Integrating AMX Control Systems with HAL

```
PUSH:
{
    nChannelIndex = GET_LAST(nSourceSelChannels)

    if (nChannelIndex > 0 &&
        nChannelIndex <= length_array(nSourceSelChannels))
    {
        nSourceSelValue = nChannelIndex - 1

        UpdateSourceSelectorRadios(
            nSourceSelChannels[nChannelIndex])

        Send_String dvHAL, "'<S&',
                            itoa(nSourceSelNumber), '&',
                            itoa(nSourceSelValue), '>'"
    }
}
}
```

The device **Channel Codes** for the selector's three radio buttons are passed in through the `nSourceSelChannels` argument. The handler knows which radio button on the touch panel fired the event because the `GET_LAST` call returns the index of that button in the `nSourceSelChannels` array. It then subtracts *1* from that index to get the new selection value. A call to `UpdateSourceSelectorRadios` ensures the other radio buttons on the touch panel are toggled off.

Likewise, the tone selector has its own dedicated event handler and `UpdateToneSelectorRadios` function. The selection value of the tone selector is kept in `nToneSelValue`. The on/off states of the tone selector's three radio buttons are maintained in the `bToneSelRadios` array.

Whenever a touch panel radio button is pressed, the corresponding selector's event handler sends a message string with a new selection value to the HAL. These 'set selector' messages contain a selector control type, HAL control number and value. The valid 'set selector' messages for the source selector in this program are `<S&1&0>`, `<S&1&1>` and `<S&1&2>`. The valid 'set selector' messages for the tone selector in this program are `<S&2&0>`, `<S&2&1>` and `<S&2&2>`

The HAL sends these same 'set selector' messages to the controller whenever another client changes the selection value. Here is the STRING event handler code in the client to parse incoming selection values for both selectors:

```
active(find_string(cData, '<S&', 1)):
{
    remove_string(cData, '<S&', 1)

    nSelectorNumber = atoi(cData)

    remove_string(cData, '&', 1)

    if (nSelectorNumber == nSourceSelNumber)
    {
        nSourceSelValue = atoi(cData)
```

```

        UpdateSourceSelectorRadios(
            nSourceSelChannels[nSourceSelValue + 1])
    }
    else if (nSelectorNumber == nToneSelNumber)
    {
        nToneSelValue = atoi(cData)

        UpdateToneSelectorRadios(
            nToneSelChannels[nToneSelValue + 1])
    }

    remove_string(cData, '>', 1)
}

```

The Mainline contains feedback statements for the selectors' radio buttons. These feedback statements update the visible state of the selectors' radio buttons on the touch panel in response to selection value changes triggered by button presses from the touch panel or messages from the HAL:

```

for (nChannelIndex = 1;
     nChannelIndex <= length_array(nSourceSelChannels);
     nChannelIndex++)
{
    [dvTP, nSourceSelChannels[nChannelIndex]] =
        bSourceSelRadios[nChannelIndex]
}

for (nChannelIndex = 1;
     nChannelIndex <= length_array(nToneSelChannels);
     nChannelIndex++)
{
    [dvTP, nToneSelChannels[nChannelIndex]] =
        bToneSelRadios[nChannelIndex]
}

```

The controller requests HAL selector control values from the HAL upon connecting. This ensures that the radio buttons on the touch panel are in sync with the live selector values on the HAL. It does this by sending 'get selector' messages to Halogen/HAL when the ONLINE event is fired. These 'get selector' messages consist of a selector control type and HAL control number. See the source selector's 'get selector' message in the client's ONLINE event handler.

```

Send_String dvHAL, "'<S&', itoa(nSourceSelNumber), '>'"
Send_String dvHAL, "'<S&', itoa(nToneSelNumber), '>'"

```

The valid 'get selector' message string for the two selectors in this example are <S&1> and <S&2>. The HAL responds to these 'get selector' messages with corresponding 'set selector' messages.



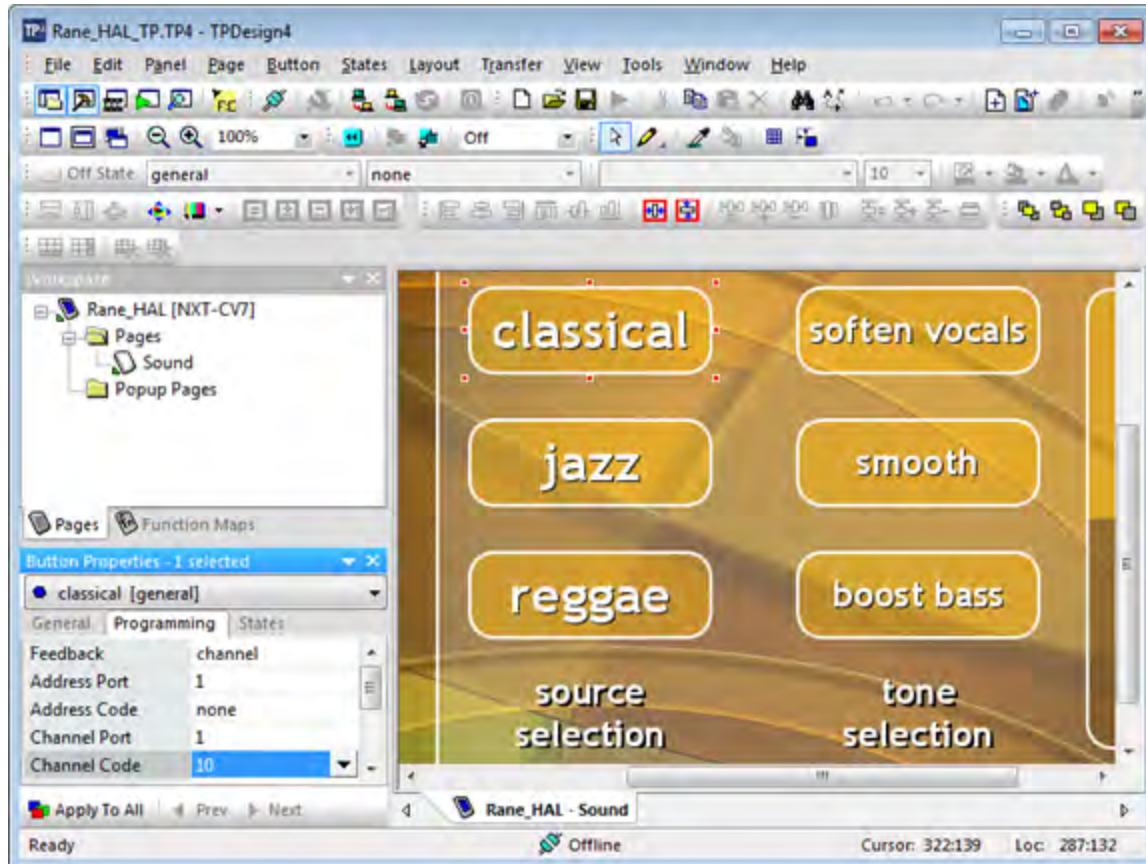
NOTE: Refer to Appendix A "HAL External Control Message Protocol" on page 45 for more information on selector messages.

Source Selector

This example demonstrates how to get radio buttons on a touch panel to switch between input source selections on a HAL. The example Halogen configuration file contains a selector block wired to three HAL Mic/Line Inputs. The input nodes on the selector block are named *Classical*, *Jazz* and *Reggae* for the genre of music being fed into each of them. This selector block is linked to a HAL Control Systems selector control named *Source Selector Control*. Switching selections on the *Source Selector Control* switches the selected input on the selector block.



Open the `ControlSystemSample.hal` example configuration file in Halogen. Switch to the **Processing** workspace by clicking on the **Processing** tab. Next, open the **External Control Systems** dialog and select the **Selector** tab. Notice that the *Source Selector Control* is assigned a HAL control number of *1*. There is no direct one-to-one relationship between this selector's unique HAL control number and any device **Channel Codes** on the touch panel. That's because the *Source Selector Control* has three touch panel buttons associated with it. The device channels of these radio buttons are *10*, *11* and *12*.



Open the Rane_HAL_TP.tp4 design file in TPDesign. Next, open the **Sound** page window to view the example touch panel design. Select the **Classical** button in the Sound page window then select the **Programming** tab in **Button Properties**. Notice that the **Channel Code** of this button is set to *10* and the **Feedback** is set to channel for latching so that it behaves like a toggle. Verify that the Channel Codes of the **Jazz** and **Reggae** buttons are *11* and *12* respectively.

Verify that the **Connected** status indicator in the upper right corner of the touch panel says **Connected**. Refer to the Troubleshooting section if it the status indicator says **Disconnected**. Double-click the *Selector (1)* block in the **Processing** map to bring up the Selector Block (1) properties dialog. Click the **Jazz** radio button in the Selector Block (1) dialog. Then press the **Classical** button on the touch panel. The source selection in Halogen switches from *Jazz* to *Classical*. Now click the *Reggae* radio button in Halogen. The **Classical** radio button is selected and the **Reggae** radio button is deselected on the touch panel.



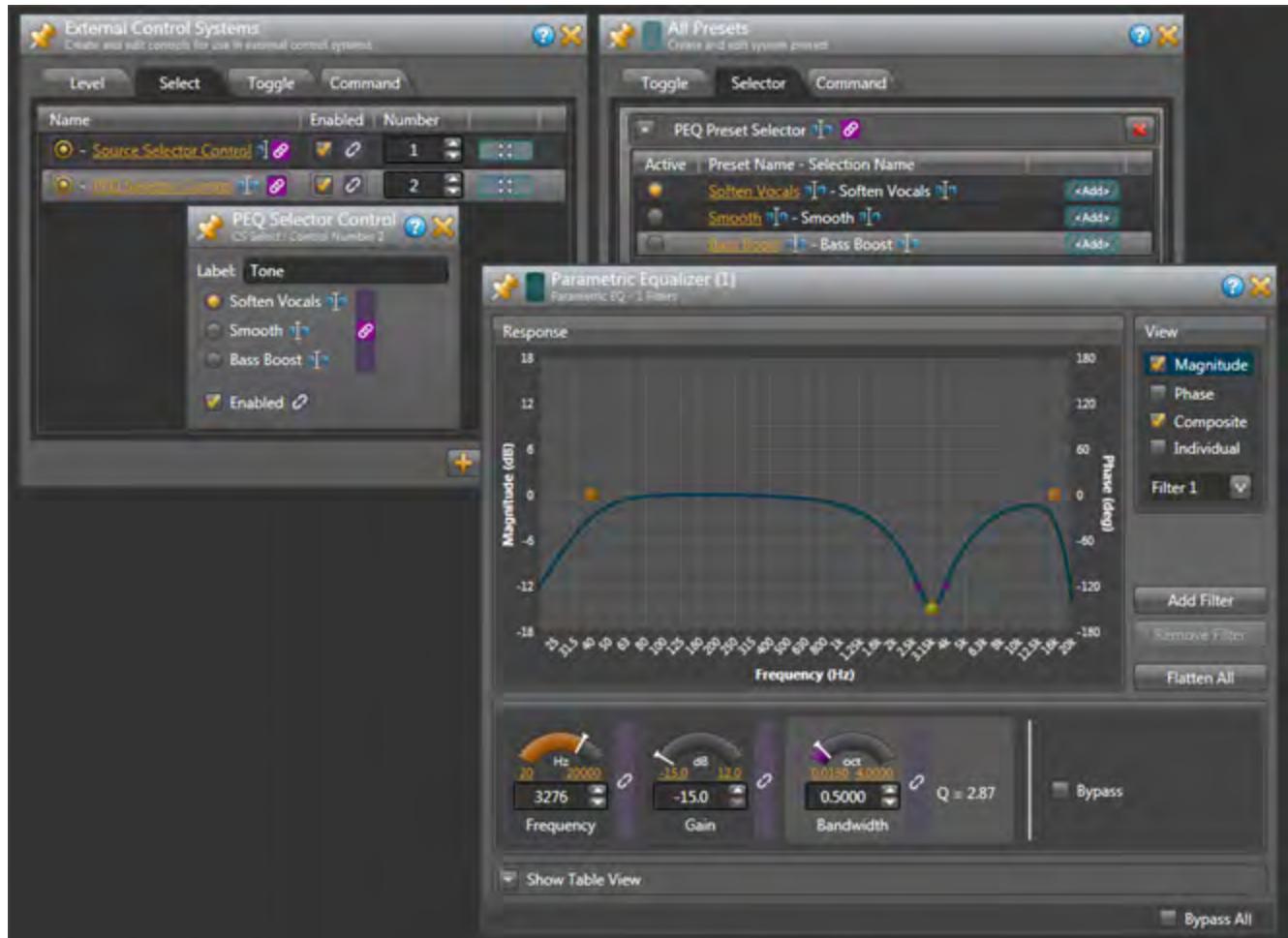
NOTE: Refer to Appendix B "Using PuTTY to Test External Control Systems" on page 55 for help with debugging if the system does not behave as described here.

Preset Selector

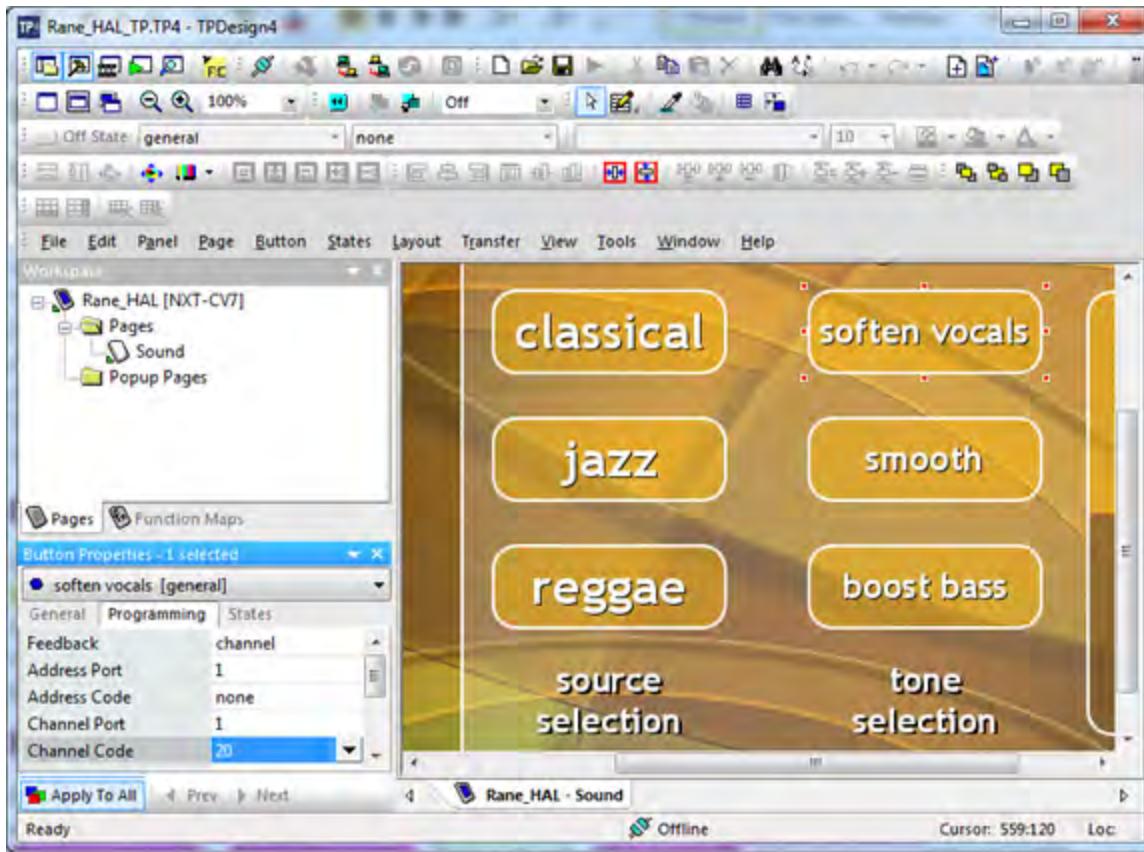
This example demonstrates how to get radio buttons on a touch panel to switch between presets on a HAL. The example Halogen configuration file contains a Parametric EQ block. Different EQ filter settings for this block are saved to different presets. One preset softens vocals, another smooths out the mid-range and a third boosts the bass. These three presets belong to the same *PEQ Preset Selector*. This preset selector is linked to a HAL Control Sys-

CHAPTER 4: Integrating AMX Control Systems with HAL

tems selector control named *PEQ Selector Control*. Switching selections on the *PEQ Selector Control* switches the active preset in the *PEQ Preset Selector*.



Open the `ControlSystemSample.hal` example configuration file in Halogen. Next, open the **External Control Systems** dialog and select the **Selector** tab. Notice that the *PEQ Selector Control* is assigned a HAL control number of 2. There is no direct one-to-one relationship between this selector's unique HAL control number and any device channels on the touch panel. That's because the *PEQ Selector Control* has three touch panel buttons associated with it. The device **Channel Codes** of these radio buttons are 20, 21 and 22.



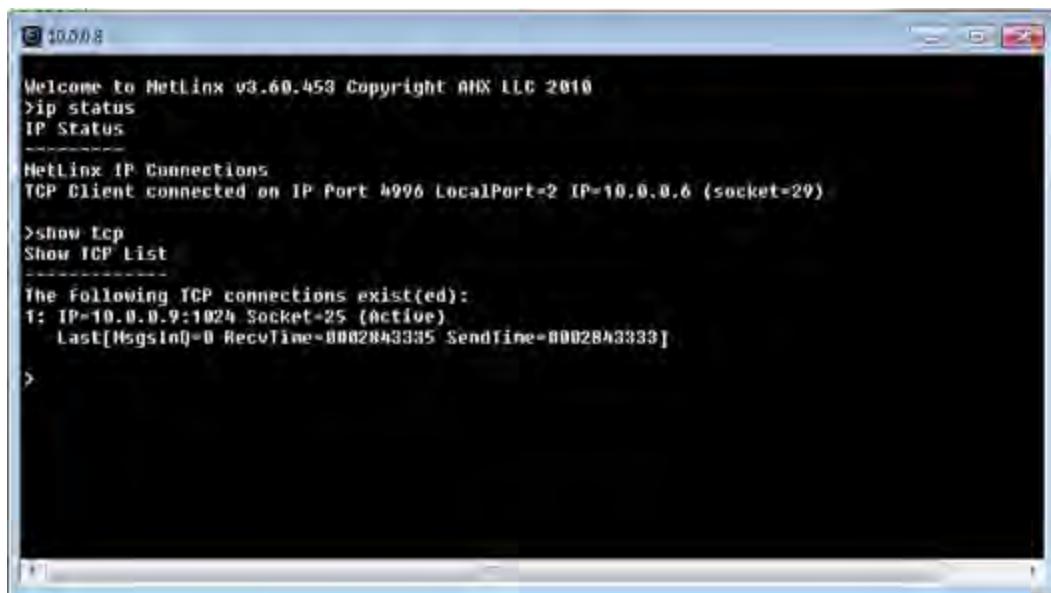
Open the Rane_HAL_TP.tp4 design file in TPDesign. Next open the **Sound page** window to view the example touch panel design. Select the **Softens Vocals** button in the Sound page window then select the **Programming** tab in **Button Properties**. Notice that the **Channel Code** of this button is set to 20 and the **Feedback** is set to channel for latching so that it behaves like a toggle. Verify that the Channel Codes of the **Smooth** and **Boost Bass** buttons are 21 and 22 respectively.

Verify that the **Connected** status indicator in the upper right corner of the touch panel says **Connected**. Refer to the Troubleshooting section if the status indicator says **Disconnected**. Click the **Presets** icon in the **Processing** toolbar to bring up the **All Presets** dialog. Choose the **Selector** tab in that dialog and expand the *PEQ Preset Selector* so that its contents are visible. Click the *Smooth* radio button in the **All Presets** dialog. Then click on the **Softens Vocals** radio button on the touch panel. The selected EQ preset in Halogen switches from *Smooth* to *Softens Vocals*. Now click on the **Boost Bass** radio button in Halogen. The **Boost Bass** radio button is selected and the **Softens Vocals** radio button is deselected on the touch panel.

 **NOTE:** Refer to Appendix B "Using PuTTY to Test External Control Systems" on page 55 for help with debugging if the system does not behave as described here.

Troubleshooting

1. If you are having problems applying the example design to your touch panel:
 - Remember that design files are transferred to an AMX touch panel by way of the controller.
 - Open the Rane_HAL_TP.tp4 design file in TPDesign.
 - Select **Connect** from the **Transfer** menu to bring up the **Connect** dialog.
 - If there is no entry for your controller in the Connect dialog's connection list add one by clicking the **New** button.
 - Once your controller has a connection in the list, select **Send To Panel** from the Transfer menu to transfer the design to the touch panel.
 - NetLinx Studio provides tools to help with troubleshooting connectivity problems.
 - Select **Telnet Session** from the Tools menu.
 - Add an entry with your controller's IP to the **List of Addresses** in Telnet Settings if one does not exist.
 - Select your controller's address entry and click the **OK** button to start the telnet session.
 - Type *help* from the telnet prompt to see the various command line tools at your disposal.
 - Verify that your controller has established communications with the control server running on your PC by entering *ip status* from the telnet prompt.
 - Verify that your touch panel has established communications with your controller by entering *show tcp* from the telnet prompt.



```
Welcome to NetLinx v3.60.453 Copyright AMX LLC 2010
>ip status
IP Status

NetLinx IP Connections
TCP Client connected on IP Port 4996 LocalPort=2 IP=10.0.0.6 (socket=29)

>show tcp
Show TCP List
-----
The Following TCP connections exist(ed):
T: IP=10.0.0.9:1024 Socket=25 [Active]
  Last[MsgsIn]=0 RecvTime=8002843335 SendTime=8002843333
>
```

2. If the AMX controller cannot connect to the Halogen Control Server:
 - Make sure the PC and controller are on the same network and have compatible IP addresses.
 - Make sure the IP address configured for the HAL1 in the controller is the IP address of the PC running Halogen and that the port is 4996.
 - Ensure that the firewall on the PC allows TCP/IP communication on port 4996.

3. If the AMX controller connects to the Halogen Control Server but the controls do not operate as expected:

- Make sure you have opened the example configuration in Halogen (ControlSystemSample.hal)
- Check to see if you are using *0* as a device channel code in your AMX system



NOTE: *0* is a valid control number in Halogen (and other external control systems) but a device channel code of *0* has special significance in AMX. An AMX event handler with a device channel code of *0* fires when any event of that type is received from the device. For instance, pressing any button on a touch panel will trigger an event from device channel code *0* resulting in a message erroneously being sent to HAL control *0*.

To avoid using a control number of *0* in Halogen

- Open the Control Systems dialog by clicking on the Control Systems icon in the Processing toolbar
- Select each tab (Level, Select, Toggle and Command) in the Control Systems dialog and verify that none of the controls have a HAL control number of *0* assigned to them
- If a HAL control has a number of *0* assigned to it change it to a non-zero HAL control number that matches the device channel code of the corresponding user interface widget in the AMX touch panel.

APPENDIX A: HAL External Control Message Protocol

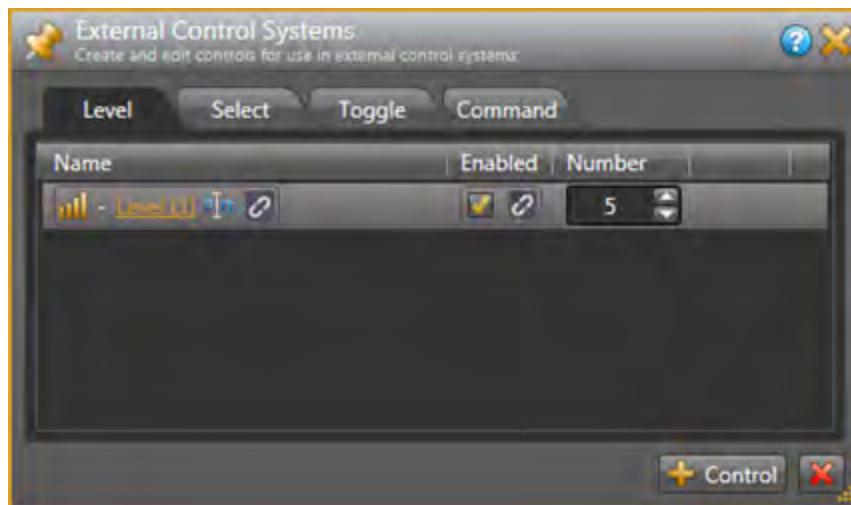
Once you have connected to either the Halogen or HAL external control server, you can communicate with it using simple, text-based messages. This section defines this message protocol.

All messages consist of ASCII text and a single message always starts with a ‘<’ character and ends with a ‘>’ character. Inside the start and end characters is the message body, which always includes at least a message type followed by a control number. Each part of the message body is separated from another part by an ‘&’ character. Messages can also have additional parts depending on the type of message. The table below shows the format for each specific message type.

When sending messages to the Halogen/HAL Control Server, it:

- Allows white space anywhere in the message, which includes space, tab, carriage return, and line feed characters
- Ignores all characters outside of a message (that is, before the start character and after the end character)
- Ignores the case of alpha characters – you can use upper or lower case in any combination

Halogen’s External Control Systems dialog defines the unique control number for each control, which is always ‘n’ in the message table below. For example, suppose you have defined a Level control in the Halogen External Control Systems dialog as follows:



In this instance, the control number is 5, and that is what you use in the ‘set level’ and ‘get level’ messages when developing external control systems. In this case the ‘get level’ message for this control is <L&5>.

Details of external control messages

In the following tables, normal characters are literal while bold characters are variable and defined in the description.

All Controls

Message Format	Description	How Used	Example
<?>	Get all values message.	Send this to HAL to request the current values of all controls.	<?> - request the current values of all controls in the configuration.

Level Controls

Message Format	Description	How Used	Example
<L&n>	Get level message, where: n is the control number.	Send this to HAL to request the current value of a Level external control.	<L&1> - request the current value of Level external control number 1.
<L&n&v>	Set level message, where: n is the control number. v is the control value. The range for v is 0 to 1000, corresponding to 0 to 100.0%.	Send this to HAL to set a Level external control to a new value. HAL sends this: <ol style="list-style-type: none"> 1. in response to a get level message 2. whenever a Level external control changes value 	<L&1&456> - set the value of Level external control number 1 to 45.6%.
<LA&n>	Get label message, where: n is the control number.	Send this to HAL to request the current label value of a Level external control.	<LA&1> - request the current label value of Level external control number 1.

APPENDIX A: HAL External Control Message Protocol

Message Format	Description	How Used	Example
<code><LA&n&string></code>	<p>Set label message, where:</p> <p>n is the control number.</p> <p>string is the label for the control.</p>	<p>Send this to HAL to set the label for a Level external control to a new value.</p> <p>HAL sends this:</p> <ol style="list-style-type: none"> 1. in response to a get label message 2. whenever a label for a Level external control changes value 	<p><code><LA&1&Volume></code> - set the label value for Level control number 1 to "Volume".</p>
<code><LE&n></code>	<p>Get enable message, where:</p> <p>n is the control number.</p>	<p>Send this to HAL to request the current enable value of a Level external control.</p>	<p><code><LE&4></code> - request the current enable value of Level external control number 4.</p>
<code><LE&n&v></code>	<p>Set enable message, where:</p> <p>n is the control number.</p> <p>v is the enable value.</p> <p>The values for v are 0 (disabled) or 1 (enabled).</p>	<p>Send this to HAL to set the enable for a Level external control to a new value.</p> <p>HAL sends this:</p> <ol style="list-style-type: none"> 1. in response to a get enable message 2. whenever an enable for a Level external control changes value 	<p><code><LA&4&0></code> - set the enable value for Level control number 4 to 0 (disabled).</p>

Toggle Controls

Message Format	Description	How Used	Example
<T&n>	Get toggle message, where: n is the control number.	Send this to HAL to request the current value of a Toggle external control.	<T&23> - request the current value of Toggle external control number 23.
<T&n&v>	Set toggle message, where: n is the control number. v is the control value. The values for v are 0 (un-checked) or 1 (checked).	Send this to HAL to set a Toggle external control to a new value. HAL sends this: <ol style="list-style-type: none"> 1. in response to a get toggle message 2. whenever a Toggle external control changes value 	<T&23&1> - set the value of Toggle external control number 23 to 1 (checked).
<TA&n>	Get label message, where: n is the control number.	Send this to HAL to request the current label value of a Toggle external control.	<TA&1> - request the current label value of Toggle external control number 1.
<TA&n&string>	Set label message, where: n is the control number. string is the label for the control.	Send this to HAL to set the label for a Toggle external control to a new value. HAL sends this: <ol style="list-style-type: none"> 1. in response to a get label message 2. whenever a label for a Toggle external control changes value 	<TA&1&Mute> - set the label value for Toggle control number 1 to "Mute".

APPENDIX A: HAL External Control Message Protocol

Message Format	Description	How Used	Example
<TE&n>	Get enable message, where: n is the control number.	Send this to HAL to request the current enable value of a Toggle external control.	<TE&4> - request the current enable value of Toggle external control number 4.
<TE&n&v>	Set enable message, where: n is the control number. v is the enable value. The values for v are 0 (disabled) or 1 (enabled).	Send this to HAL to set the enable for a Toggle external control to a new value. HAL sends this: <ol style="list-style-type: none">1. in response to a get enable message2. whenever an enable for a Toggle external control changes value	<TA&4&0> - set the enable value for Toggle control number 4 to 0 (disabled).

Selector Controls

Message Format	Description	How Used	Example
<S&n>	Get selection message, where: n is the control number.	Send this to HAL to request the current value of a Select external control.	<S&6> - request the current value of Select external control number 6.
<S&n&v>	Set selection message, where: n is the control number. v is the control value. The range for v is 0 to one less than the number of selections.	Send this to HAL to set a Select external control to a new value. HAL sends this: <ol style="list-style-type: none">1. in response to a get selection message2. whenever a Select external control changes value	<S&6&3> - set the value of Select external control number 6 to 3 (the fourth item in the set of selections).

Message Format	Description	How Used	Example
<SL&n>	Get selection link message, where: n is the control number.	Send this to HAL to request the current number of selections for Select external control.	<SL&7> - get the number of selections for Select external control number 7.
<SL&n&x>	Set selection link message, where: n is the control number. x is the number of selections. 0 indicates that no selections are available – i.e. the selector link is inactive.	<p>It is not meaningful to send this message to HAL because an external control system cannot change a selection link.</p> <p>HAL sends this:</p> <ol style="list-style-type: none"> 1. in response to a get selection link message 2. whenever the link activation changes for the Select external control 	<SL&7&4> - set the number of selections for Select external control number 7 to 4.
<SN&n&i>	Get selection name message, where: <ol style="list-style-type: none"> 1. n is the control number. 2. i is the selection item. The range for i is 0 to one less than the number of selections. 	Send this to HAL to request the current value of a selection name for a particular selection item of a Select external control.	<SN&12&2> - get the current selection name for selection item 2 of Select control number 12. Selection item 2 is the third item in the set of selections.

APPENDIX A: HAL External Control Message Protocol

Message For- mat	Description	How Used	Example
<SN& n & v & string >	<p>Set selection name message, where:</p> <p>n is the control number.</p> <p>v is the control value. The range for v is 0 to one less than the number of selections.</p> <p>string is the current display name for selection v.</p>	<p>It is not meaningful to send this message to HAL because an external control system cannot change a selection name.</p> <p>HAL sends this:</p> <ol style="list-style-type: none"> 1. in response to a get selection name message 2. whenever the link activation changes for the Select external control 3. whenever one of the selection names change 	<p><SN&12&2&Jazz> - set the current selection name for selection item 2 of Select control number 12 to "Jazz".</p> <p>Selection item 2 is the third item in the set of selections.</p>
<SA& n >	<p>Get label message, where:</p> <p>n is the control number.</p>	<p>Send this to HAL to request the current label value of a Selector external control.</p>	<p><SA&1> - request the current label value of Selector external control number 1.</p>
<SA& n & string >	<p>Set label message, where:</p> <p>n is the control number.</p> <p>string is the label for the control.</p>	<p>Send this to HAL to set the label for a Selector external control to a new value.</p> <p>HAL sends this:</p> <ol style="list-style-type: none"> 1. in response to a get label message 2. whenever a label for a Selector external control changes value 	<p><SA&1&Program> - set the label value for Selector control number 1 to "Program".</p>

Message Format	Description	How Used	Example
<SE&n>	Get enable message, where: n is the control number.	Send this to HAL to request the current enable value of a Level external control.	<SE&4> - request the current enable value of Selector external control number 4.
<SE&n&v>	Set enable message, where: n is the control number. v is the enable value. The values for v are 0 (disabled) or 1 (enabled).	Send this to HAL to set the enable for a Selector external control to a new value. HAL sends this: <ol style="list-style-type: none"> 1. in response to a get enable message 2. whenever an enable for a Selector external control changes value 	<SA&4&0> - set the enable value for Selector control number 4 to 0 (disabled).

Command Controls

Message Format	Description	How Used	Example
<C&n>	Command message, where: n is the control number.	Send this to HAL to fire a Command external control. HAL sends this whenever a Command external control fires.	<C&4> - Fire Command control number 4.
<CA&n>	Get label message, where: n is the control number.	Send this to HAL to request the current label value of a Command external control.	<CA&1> - request the current label value of Command external control number 1.

APPENDIX A: HAL External Control Message Protocol

Message For- mat	Description	How Used	Example
<CA& n & string >	Set label message, where: n is the control number. string is the label for the control.	Send this to HAL to set the label for a Command external control to a new value. HAL sends this: <ol style="list-style-type: none"> 1. in response to a get label mes- sage 2. whenever a label for a Command external con- trol changes value 	<CA&1&Flatten> - set the label value for Command control number 1 to "Flatten".
<CE& n >	Get enable message, where: n is the control number.	Send this to HAL to request the current ena- ble value of a Command external control.	<CE&4> - request the current enable value of Command external control number 4.
<CE& n & v >	Set enable message, where: n is the control number. v is the enable value. The values for v are 0 (disabled) or 1 (en- abled).	Send this to HAL to set the enable for a Com- mand external control to a new value. HAL sends this: <ol style="list-style-type: none"> 1. in response to a get enable message 2. whenever an enable for a Command external con- trol changes value 	<CA&4&0> - set the enable value for Command control number 4 to 0 (disabled).

Communications Monitoring

Message Format	Description	How Used	Example
<PING>	Ping message	Send this to HAL to request a pong message.	<PING>
<PONG>	Pong message	It is not meaningful to send this message to HAL. HAL sends this whenever it receives a ping message from an external control system.	<PONG>

APPENDIX B: Using PuTTY to Test External Control Systems

The Halogen/HAL external control message protocol is ASCII text based, which means that you can use a standard telnet client to connect to the control server and exchange messages. This is often useful when developing and testing a HAL configuration that contains external controls, either to monitor the operation of your external control system or to serve as a simple form of external control system before the production system is ready or available.

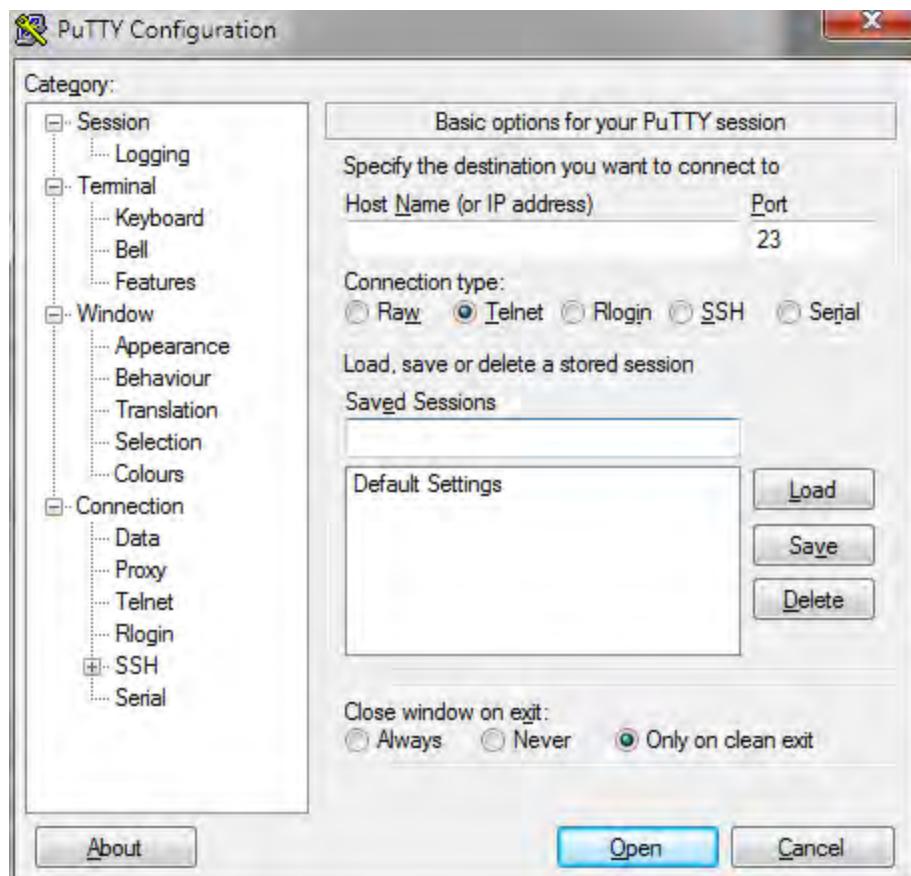
Any telnet client will work, but this section shows how to do this with PuTTY, an open source telnet client application that is free, easy to install and use, and runs on all versions of Windows.

How to obtain and install PuTTY

The top level web page for PuTTY is <http://www.putty.org>. That page contains a link to a page that has links for downloading the PuTTY application. All you need is the latest release version of putty.exe for Windows on Intel x86. Download this file and put it in an accessible folder on your PC. Since the complete installation is a single file, putting it on the desktop works great.

Starting PuTTY and configuring it for communications with the Halogen control server

Start PuTTY by double clicking on the putty.exe file. This brings up the PuTTY Configuration window:

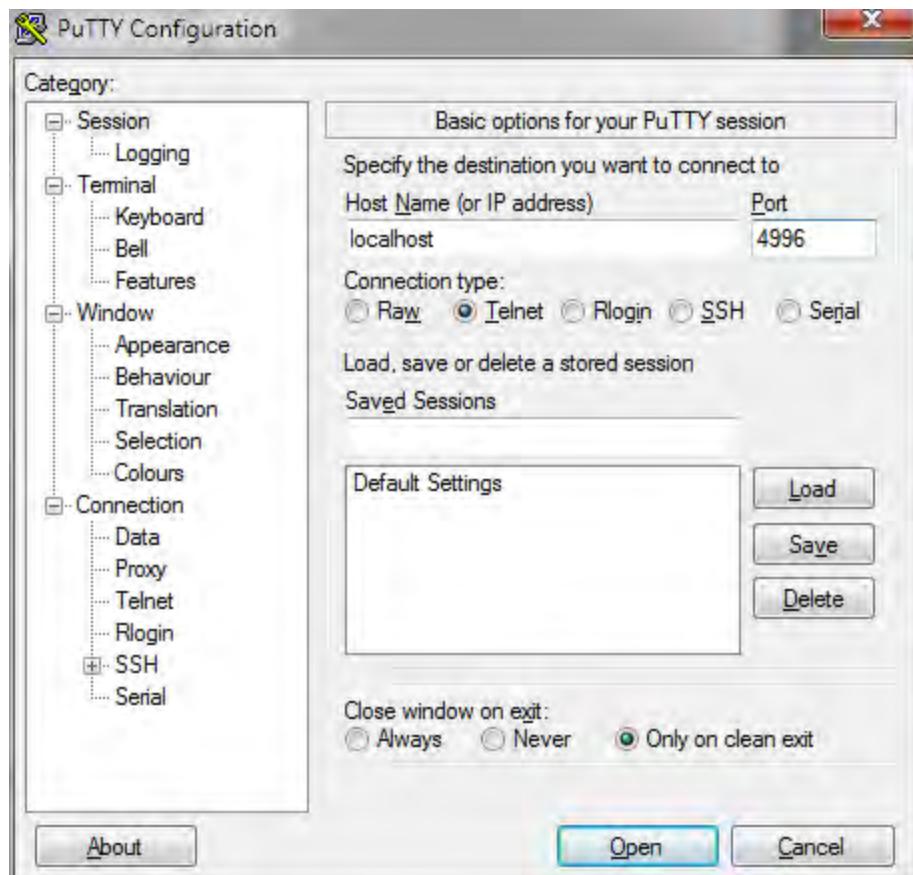


To connect to the Halogen external control server, do the following:

1. Start Halogen and load your configuration that contains the external controls you'd like to test

APPENDIX B: Using PuTTY to Test External Control Systems

2. In the PuTTY configuration window:
 - a. Enter localhost in the Host Name text box
 - b. Change the Connection type to Telnet
 - c. Change the port number to 4996
 - d. Click the Open button



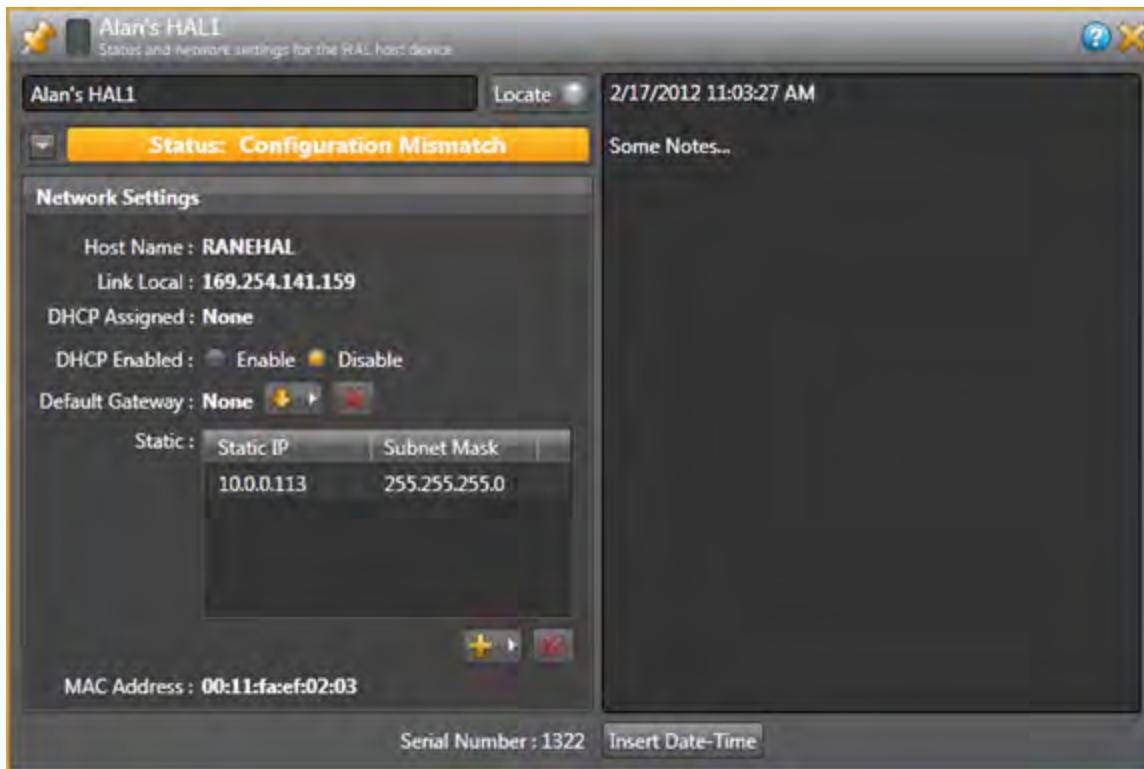
To connect to the HAL external control server, do the following:

1. Start Halogen and Apply your configuration that contains the external controls you'd like to test to the HAL.
2. Open the HAL property dialog by double clicking on the HAL title bar in the Halogen Hardware Work-space.



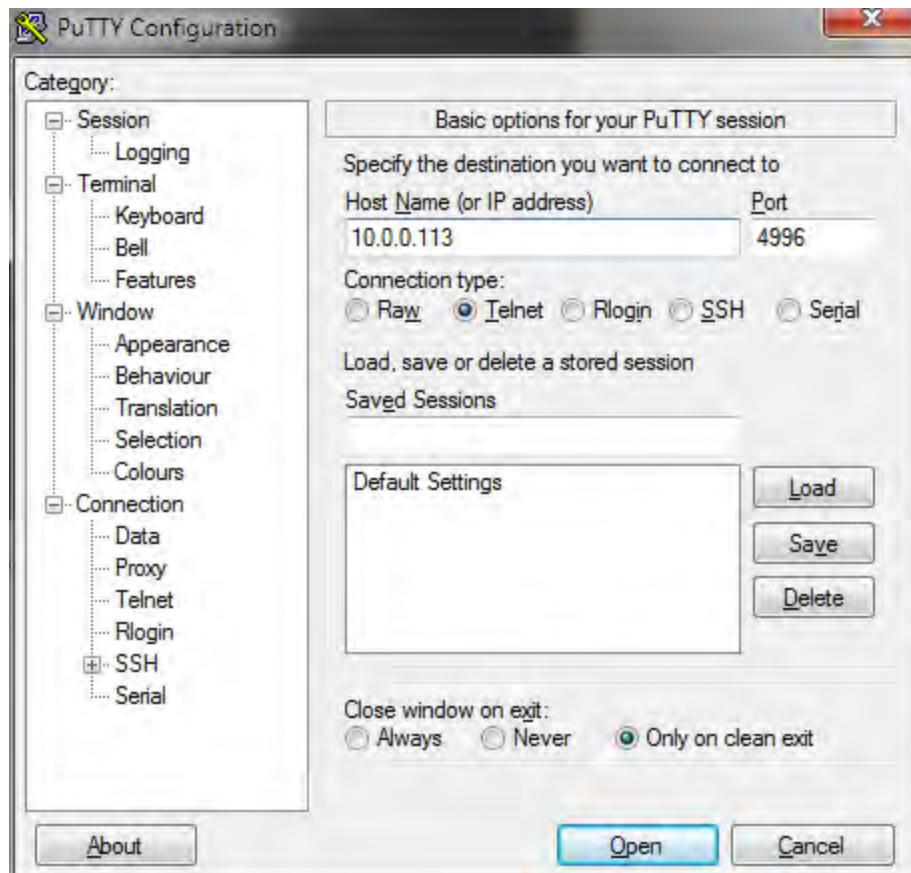
3. Observe the HAL's configured IP addresses and pick one that you want to use with PuTTY. The address you choose needs to be on the same network as your PC that is running PuTTY.
4. In the PuTTY configuration window:
 - a. Enter HAL's IP address in the Host Name text box
 - b. Change the Connection type to Telnet
 - c. Change the port number to 4996
 - d. Click the Open button

For example, the HAL1 host properties dialog will look something like this:



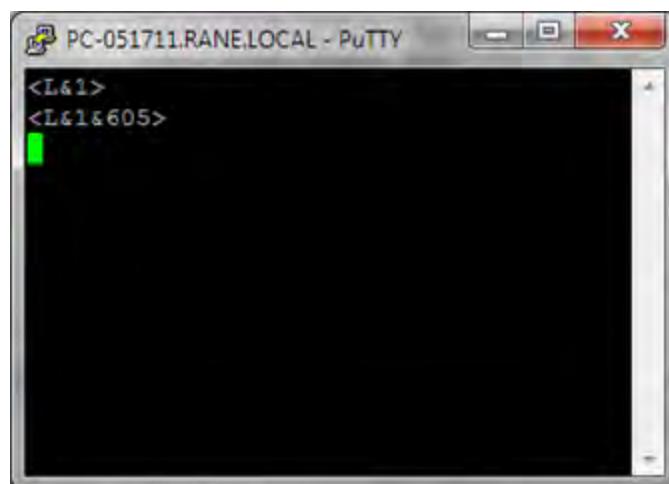
Here, We've added a static IP of 10.0.0.113. We'll use this in the PuTTY Configuration window:

APPENDIX B: Using PuTTY to Test External Control Systems



Communicating with the Halogen/HAL control server

Connecting PuTTY to either Halogen or HAL establishes communication between the telnet client and the control server. Now you can send and receive messages using the PuTTY window that appears. For example, suppose you have a Level control configured in Halogen that is control number 1 and currently has a value of 60.5%. To request the current value of this control, type `<L&1>` into the PuTTY window and then press Enter. This will send the message to Halogen, which will respond with `<L&1&605>`. If all is right with your test system, you'll see something like this:



Your Halogen configuration will be something like this: (the Level external control is linked to the Level block's level control)



How to use PuTTY when testing external control systems

Here are a few ideas for what you can do:

1. Test your HAL external controls by sending and receiving external control messages

To do this, connect PuTTY to your HAL or Halogen external control server and type external control messages into the PuTTY window and observe what HAL sends in response.

2. See how the HAL control server responds when you change external controls either from Halogen or from actual remotes in the HAL system.

Connect PuTTY to your HAL or Halogen external control server and change external controls from within the HAL system. Observe what the control server sends to PuTTY.

3. See how the HAL control server responds when configured presets activate or deactivate that affect external control values or link activations.

Connect PuTTY to your HAL or Halogen external control server and activate or deactivate presets that contain blocks or links related to external controls. Observe what the control server sends to PuTTY.

4. Use PuTTY to monitor the operation of your external control system

You can connect any number of external control clients to a Halogen or HAL Control Server. This means that when you are developing and testing your control system with a commercial product, such as an AMX system, you can also connect to the control server using PuTTY.

APPENDIX B: Using PuTTY to Test External Control Systems

 **NOTE:** This lets you monitor most of the interaction between your external control system and your Halogen/HAL Control Server. This works because when external controls change, the Halogen/HAL Control Server sends these changes to all connected external control clients. So, when your external controller changes a control on the HAL, the HAL control server sends the change to your PuTTY client.

To use PuTTY in this way, first connect PuTTY to the Halogen/HAL Control Server that you want to monitor. Next connect your control system to the same Halogen/HAL Control Server. As your external control system interacts with the HAL system, external control messages will appear in the PuTTY window, giving you information about the operation of your external control system.

AMX® and the AMX logo are registered trademarks of AMX.

Stardraw Control is a trademark of Stardraw.com Ltd.

Crestron® is a registered trademark of Crestron Electronics, Inc.

Lexan® is a registered trademark of General Electric.

Decora® is a registered trademark of Leviton.

Windows® is a registered trademark of Microsoft Corporation in the United States and other countries.

Apple, Mac, Macintosh, iTunes, Safari, QuickTime, GarageBand, and OS X are registered trademarks of Apple Inc., registered in the U.S. and other countries.

Halogen is a trademark of Rane Corporation.