MediaMatrix®

A Division of Peavey Electronics Corporation

NWare

NWare User Guide

Version 1.4.2.0

May 13, 2008

Prepared by Peavey Digital Research, 6 Elm Place, Eynsham, Oxford, OX29 4BD, UK.

Email:mmservice@peavey-eu.com.

Contents

Chapter 1

About this guide

In This Chapter

Scope

This guide describes the features of NWare and how to use them.

Documentation conventions

The following are used in the documentation to highlight particular sections of information.

Tip: Suggests alternative ways of completing a task and shortcuts that might not otherwise be obvious.

Note: Indicates important information that should not be ignored.

Caution: Indicates that unless you are careful, your actions could result in equipment damage or loss of data.

Warning: Indicates that unless you are careful, your actions could result in injuries to personnel.

Manual set

This guide is part of the MediaMatrix documentation set.

The table below shows which user guides to refer to when you want to find out how to accomplish various tasks.

Tasks	**Relevant Guides**
Creating and managing projects using NWare	NWare User Guide
Using devices available from the NWare device tree.	NWare Device Reference
Finding out about new features added to releases of NWare and NION software	NWare Release Notes
Using different protocols, such as PASHA and SNMP, to remotely control and monitor devices in an NWare project.	External Control User Guide
Physical installation and initial configuration of a NION digital audio processor.	NION Hardware Documentation

Tasks	**Relevant Guides**
Understanding the features and physical characteristics of the NION digital audio processor.	NION Hardware Documentation
Physical installation and initial configuration of a Cab4n CobraNet audio bridge.	Cab4n Manual
Physical installation of a ControlManager server. Installation and configuration of associated software.	Control Manager Primer
Understanding how Pandad works and using it on your network.	Pandad Primer

Chapter 2

Introduction

In This Chapter

Introduction to NION

NION hardware and software allows you to design and implement network-distributed systems composed of audio inputs and outputs, audio processing and control. The systems implemented with NION can range in size from a few channels to many hundreds of channels, with no change in the hardware or software components used. NION control and audio streams are distributed using standard Ethernet wiring and switches, and can co-exist with a facility's other network data.

What is NWare?

NWare, a graphical software application running on a standard Windows PC (XP or Vista), is used to design, deploy and control a system of NioNodes, ControlNodes and CAB 4n units. Once a system is deployed and adjusted, the NWare PC is not required for the system to operate, even through power cycles.

Minimum operating requirements

Windows XP™ Home or Professional (with Service Pack 1 or higher)

or

Windows Vista

Other versions of Windows may work but they are not supported or recommended.

NWare Kiosk

A control-only application, called NWare Kiosk, can be used to present customized user-interfaces to the end-user. For more information, see *Running NWare in Kiosk mode* (on page 17).

NioNodes and ControlNodes

The NioNodes run embedded Linux, a powerful operating system capable of supporting sophisticated standalone features. NioNodes also run Python, a scripting language, which is used for control programming.

The ControlNodes run under Windows. Most of the same control devices and scripting that is available on the NioNodes is available on ControlNodes. In addition, ControlNodes support SNMP management and other control features that are not available on the NioNodes.

The system design phase starts with the designer creating a Project and entering into it a graphical representation of inputs, outputs and signal processing. The designer can also create custom control panels and custom control interaction scripts.

Each target NioNode and ControlNode specified in the Project is given a unique Role name, which is matched up with the physical target Nodes later during deployment.

Deploying a project

When a Project is Deployed, a design check is performed, DSP code is generated, the results are downloaded to the target NioNodes, and the host PC is connected to the targets, ready for control. The design check generates a report which display results indicating hardware usage and any problems encountered in the design.

This phase requires that the NWare PC be networked with the target set of Nodes onto which a Project is to be installed. During deployment, attached Nodes are discovered and displayed to the designer. The Designer then assigns the Roles that exist within the Project to the proper Nodes. Role assignment results in each Node downloading, and storing locally, the configuration data that it needs to fulfill its Role during runtime.

If desired, instead of deploying, a limited emulation of the hardware can be run on the PC that allows, all without the target hardware present: control gesturing, preset saves and loads, control script exercise and even external control.

Note: PC-based NioNode system emulation does not emulate NioNode audio signal processing.

Once a Project has been deployed, the set of Nodes targeted by the Project perform the signal and control processing contained in the design. One or more PCs running NWare_Design or NWare_Kiosk can connect to the target Nodes to control and monitor them. Third-party control systems can be attached as well.

Chapter 3

Getting started

In This Chapter

Creating a project

Launching the NWare application presents a blank Project, with a default name, ready for editing.

Project Designs are created, in general, by dragging devices from the Device Library onto the current page in the Design frame, and then wiring the Device nodes together.

Some of the Devices represent hardware elements of the target system, others represent DSP-based signal processing algorithms and others represent control functions.

Devices are positioned while in Edit mode, and are wired together while in Wire mode.

The NioNode device

The NioNode device represents the hardware components of a single NioNode. Through the Device Properties of the NioNode, the Role name, the NioNode model, whether the CobraNet CM-1 is used, GPIO pin configuration, the types of the cards populating the Expansion Slots, and other such things are declared.

Inside the NioNode device window are tabbed Control Pages containing the controls for the components of the NioNode that were configured using its Device Properties. For instance, if Expansion Slot 1 has been configured with an analog audio input card, the controls and meters for that card will appear in a Control Page titled Slot 1 Input.

The flyoffs (audio signal wiring nodes) that correspond to the audio inputs and outputs of the Expansion Slots and CobraNet bundles are found in the Flyoffs page of the Utility frame. Flyoffs are dragged into a Design page and then wired to signal processing Devices.

Deploying a project to hardware

When the File | Deploy command is invoked (shortcut key F9), a design check is performed, DSP code is generated, Roles are assigned to the discovered NioNodes, files are downloaded to the NioNodes, and the host PC is connected, ready for control.

Before downloading to the target hardware, you are prompted with the Role Deployment window where you assign the Roles in your Project to the NioNodes discovered on the network.

A report is generated as a result of the design check, and is displayed in the Output frame.

Emulating a project without hardware

When you are without target hardware (NioNodes, CAB 4ns, etc.), or don't want to disturb a currently running system, the target hardware can be emulated on the NWare PC. This makes it possible to gesture controls, save and load presets, exercise the control programming and verify external control. Such actions are runtime features of the target hardware, and therefore don't normally run on the PC.

Note: Audio processing is not done in the Emulator.

To emulate, invoke the File | Emulate command (shortcut key Ctrl-F9).

Chapter 4

Project Deployment and Hardware Installation

In This Chapter

Deploying, Connecting, Emulating and Uploading

Deploy

The File | Deploy command starts a chain of events:

- A Check and Compile Project is performed
- DSP code and control configuration files are generated
- The **Role Deployment** dialog box displays the NioNodes discovered on the network
- You assign the Roles in your Project to the discovered NioNodes
- Configuration files are download to the NioNodes
- NWare is connected to the NioNodes, at which point control and monitoring is possible

The **Role Deployment** dialog box dynamically displays the NioNodes found on the network and allows them to be assigned to the Roles in the current Project. For each discovered NioNode, a network Name, Project, Role and Status is displayed. Role assignment is done by dragging a NioNode onto one of the Project Roles, though nothing will actually happen until OK is clicked. Once OK has been clicked, the assignments that have been made will take effect and a Connect will be performed. This can take a few moments as it involves the NioNode(s) downloading Project files to local storage, and then beginning audio processing.

A File | Disconnect performs a disconnect from the running hardware, disabling control and monitoring.

Tip: Once a Project has been deployed to one or more NioNodes, those nodes will boot and run the Project after a power cycle.

Connect

The File | Connect command is used to connect to already running hardware. Based on the currently open Project, NWare will look for the hardware running that Project and connect to it, at which point control and monitoring is possible.

A File | Disconnect performs a disconnect from the already running hardware, disabling control and monitoring.

Emulate

When you are without target hardware (NioNodes, CAB 4ns, etc.), or don't want to disturb a currently running system, the target hardware can be emulated on the NWare PC. This makes it possible to gesture controls, create and load presets and exercise the control programming. Such actions are runtime features of the target hardware, and therefore do not normally run on the PC.

Note: Audio processing is not done in the Emulator.

To emulate, invoke the **File | Emulate** command (shortcut key Ctrl-F9).

Selecting **File | Disemulate** disconnects from the Emulator and halts it.

Compile Report

An HTML-based report is generated as a result of a Check design. The report displays notes, warnings, errors, and a resource report table showing the target resources used by the design.

Many of the notes, warnings and errors can be clicked on to highlight, in the Design Frame, those elements associated with the note. This is also the case with many of the resource report table entries. The highlight can be cleared by clicking elsewhere in the report.

Upload and Connect

The project can be stored on and retrieved from the NioNodes. When the **Role Deployment** dialog box is displayed during Deploy, you can check the **Save Project to Target** option, which will download and store the complete project on the NION.

Note: This option is only available if you have saved the project at least once. For example, an unsaved project called *untitled2* cannot be saved to the target hardware.

The **File | Upload and Connect** command presents a list of projects running on NIONs on the network. Selecting a project will upload it to the PC, present a **Save As** dialog box, and after saving locally, perform a Connect to the running project.

NioNode Hardware Monitoring

Remote Log

The Remote Log page in the Output Frame contains a list of the currently discovered NioNodes. You can enable polling of a NioNode log with a right-click | Enable Polling. You display the log of those NioNodes that are being polled by clicking on it. A right-click menu in the display area of the log allows you to configure the filtering of what is being displayed.

NioNode Configuration

Network Configuration

Deployment requires that all target hardware (NioNodes) and the NWare PC be on the same network, and there is a front panel interface page for entering the network settings.

Host Name

NioNodes can be given host names to make identifying them easier, and all NWare-based interfaces display the host name. However, DNS is not currently supported, so you cannot communicate with a NioNode using the host name when using ping or when using the TCP/IP-based external control protocol, but must instead use the IP address.

Note: A host name is not the same as a Role name you can choose to make them the same if you want, but there are situations where you might not want them to be the same.

MODE

Select DHCP or Static IP

IP Address

NioNodes can either be assigned a static IP address, or can use DHCP to get an IP address from a DHCP server.

IP Subnet

Set the subnet mask, which defaults to 255.255.255.0.

Firmware Upgrade Procedure

The firmware in the NioNodes can be checked and upgraded using the Update Firmware utility. This utility is launched with the Tools | Update NioNode Firmware menu command. The dialog box displays a list of discovered NioNodes, and their status. Right-click the NioNode you want to upgrade, and select the upgrade package. It will then be downloaded and installed on the unit.

Time of Day

To support some features, NioNodes need to know what time it is. There is a front panel interface page to set the time zone and time. When the time is set for one NioNode through this interface, the time of all other NioNodes on the network will sync to it. Setting the time zone, however, does not propagate to the other NioNodes.

Monitoring and troubleshooting

Front panel LED indicators

There are several LEDs on the front panel that help indicate unit status.

Fault	
RED	Audio not working
OFF	Audio working

The Fault LED tracks the state of the fault relay. (The fault relay signals appear on pins of the GPIO connector.)

Note: The RED state does not necessarily indicate a problem, but that audio is not actively being processed and passed. This state occurs normally when the unit is rebooting or has not yet been configured to run a Project. It also occurs when the unit has encountered a problem and can no longer process and pass audio.

X-DAB	
GREEN	Primary and backup rings are operational
GREEN BLINKING	Running in backup with single cable missing - all audio present
RED BLINKING	Running in backup with multiple cables/nodes missing - some audio gone
OFF	XDAB is not in use (in the current Project/Role for this NioNode)

Note: The XDAB LED is updated only when XDAB is used by the NioNode in the current Project and the NioNode is up and passing audio.

LAN	
GREEN	Ethernet link
OFF	No Ethernet Link

CobraNET	
GREEN	Ethernet link
GREEN BLINKING	Ethernet link and NioNode is CobraNet conductor
RED	No link or CM-1 missing or malfunctioning, and NioNode is configured to use CM-1
OFF	No link or CM-1 missing or malfunctioning, and NioNode is not configured to use CM-1

Note: Even if the NioNode has not been configured to use its CM-1 (in the Project design), link and conductor status are still reported and the unit will be visible as a CobraNet node on the network it is attached to. You will only see RED, however, if the NioNode has been configured to use its CM-1 and there is a problem with link or with the CM-1 module.

IDE	
GREEN	Mass storage (CompactFlash) read or write activity
OFF	No activity

Tip: The IDE LED indicator is remarkably similar to the disk status LED you may have on the front of your PC or on the bezel of your laptop.

Attn	
RED BLINKING	Indicates there is an error or warning condition of interest

OFF	No status change of interest

Tip: This indicator is paired with a button. When the LED blinks, the button can be pressed to cause the LCD to navigate to a page displaying the source of the condition.

Attn	
OFF	Unit is off
ORANGE	Unit is booting up or down
GREEN	Unit has booted up

Chapter 5

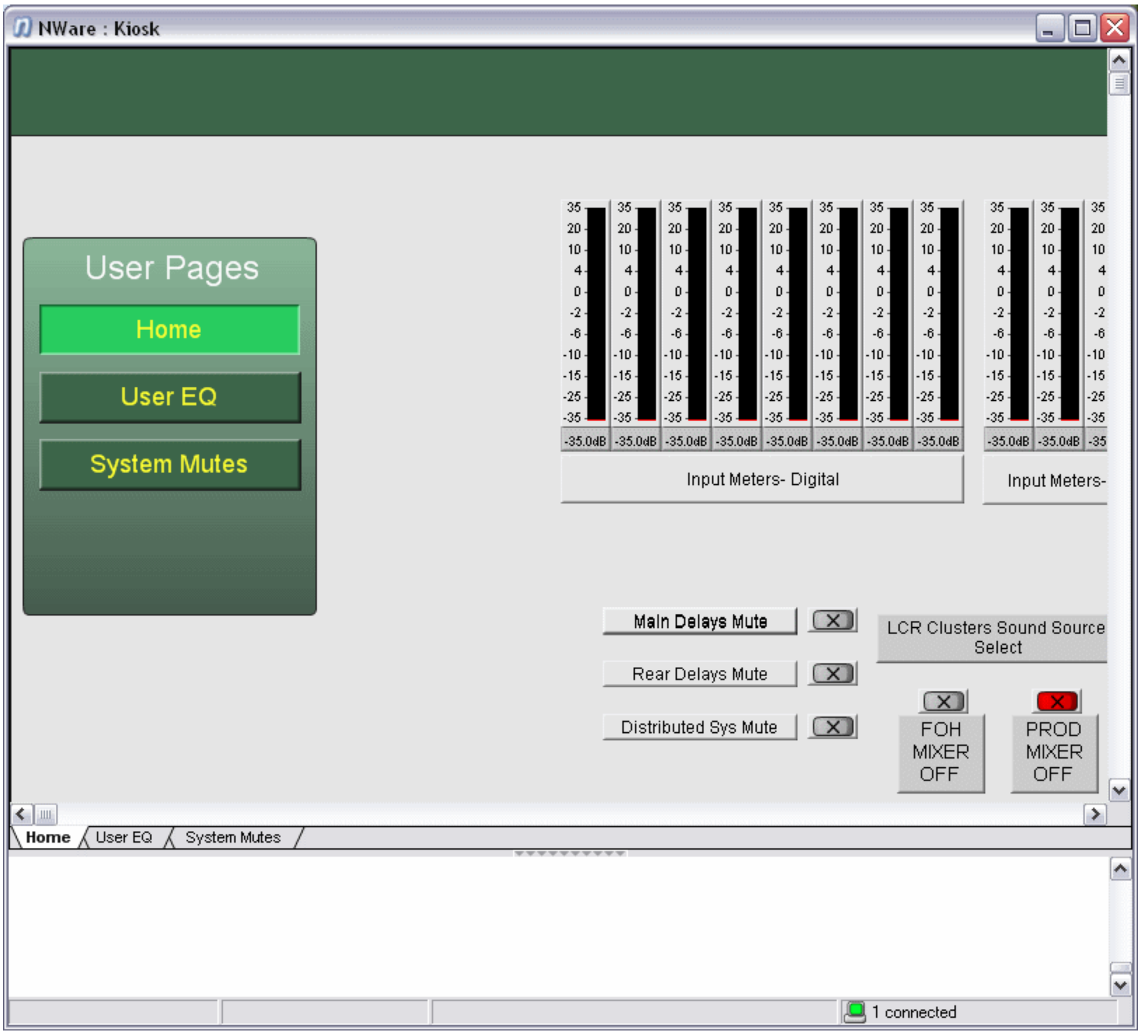
Running NWare in Kiosk mode

In This Chapter

Introduction

NWare:Kiosk or just *Kiosk* is a special NWare operating mode that allows a user to control the features of a project, but not change settings made by the designer.

When you run Kiosk, you can use the controls shown on pages like the one below, but some user interface (UI) features normally available in NWare, such as the tree view, will not be available.



Kiosk offers a number of benefits, including:

- Multiple user/password and security options.
- Remote control over Ethernet or wi-fi.
- Multiple project monitoring and control.
- Multiple user connections to the same NWare project.

Tip: Switching between NWare:Kiosk mode and regular NWare mode can be done using a simple command line option. This means it is possible for users to remove the command line option and run the full version of NWare on their PCs. If you would like to prevent users from running the full version of NWare, you can install a special version of NWare that will only run in Kiosk mode. You can download this software from the *Media Matrix website* (http://mm.peavey.com).

Task overview

Below is a summary of what you will need to do in order to make pages from an NWare project available in Kiosk. We recommend that you read through the steps carefully, even if you are an experienced NWare user.

1. Open NWare.
2. Load the project containing the pages you want to make available in Kiosk.
3. *Export pages from the project* (on page 20).
4. *Check the user preferences and make changes if necessary* (on page 22).
5. *Deploy the project to a NION* (on page 10).
6. *Run Kiosk and select the project* (on page 24).

If you want Kiosk to start automatically and load a project, see *Logging on and loading a project automatically* (on page 26).

If you want to show or hide specific UI features in Kiosk, or improve password security when logging on automatically, see *Controlling Kiosk features using a personality file* (on page 27).

Exporting pages from an NWare project

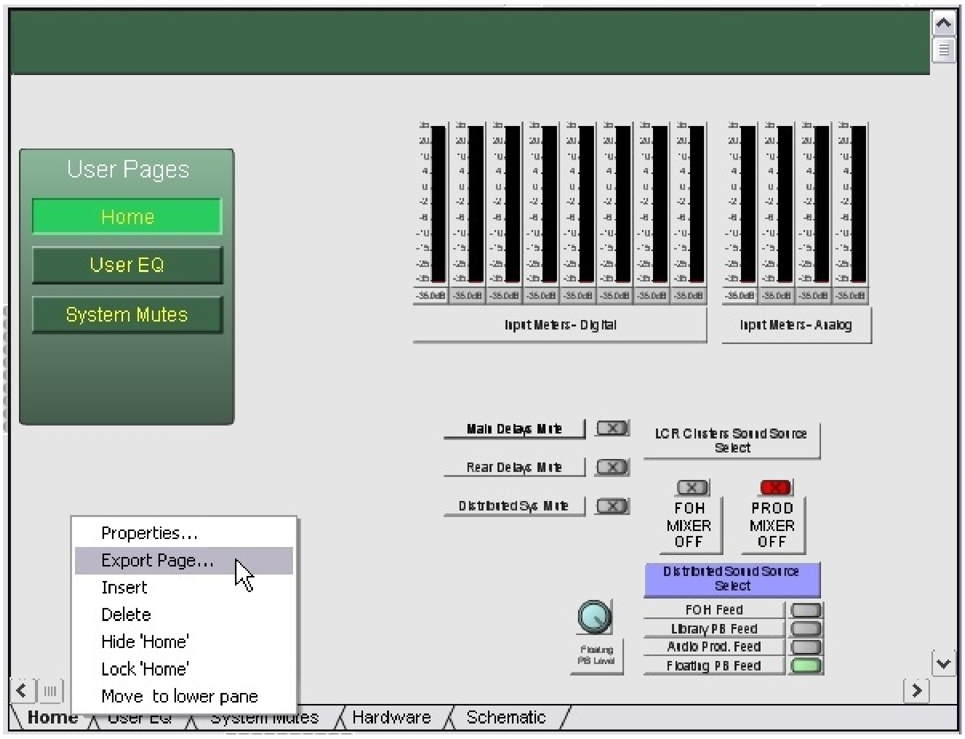
In order to make pages from your NWare project available to Kiosk users, they must be exported. The pages are still stored within the NWare *.npa* file, but they will be made available when you load the project into Kiosk.

The order in which you export pages does not affect the order in which they are displayed as tabs at the bottom of the Kiosk window - the order will be the same as it is in NWare. If the project contains only one page group, the last page that was exported will be the page that is displayed when you open Kiosk. However, if the project contains several page groups, the page displayed will be the page that is last in alphabetical order.

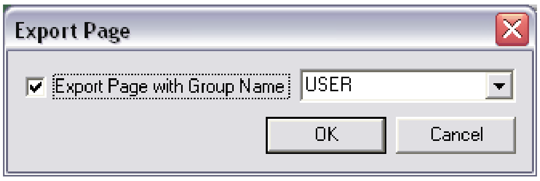
When you run Kiosk, you can drag the page tabs to change the page order. However, when you restart Kiosk and connect to the project, the pages will be returned to the original order.

⏩ *To export pages from an NWare project*

1. Right-click the page tab, and then click **Export Page**.



The **Export Page** dialog box is displayed.



☑ Export Page with Group Name: USER

2. Select the **Export page with Group Name** check box.
3. In the list, click a group name for the page.

Note: In order for a Kiosk user to be able to access the page you are exporting, they must be a member of the group you specify.

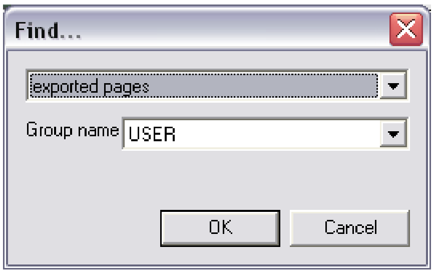
Tip: You can create a new group by typing a new name in the list.

4. Click **OK**.

Finding exported pages

1. In NWare, on the **Edit** menu, click **Find**.

   The **Find** dialog box is displayed.

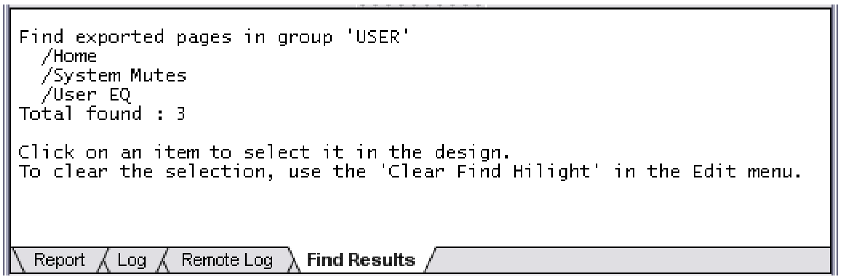


exported pages

Group name USER

2. In the first list, click **exported pages**.
3. If the pages are in a group, click the group name in the **Group name** list, otherwise click ***** in the **Group name** list.
4. Click **OK**.

   The results of the search are shown in the output frame at the bottom of the window.



```
Find exported pages in group 'USER'
  /Home
  /System Mutes
  /User EQ
Total found : 3

Click on an item to select it in the design.
To clear the selection, use the 'Clear Find Hilight' in the Edit menu.
```

Tip: You can jump directly to the pages in the search results by clicking them.

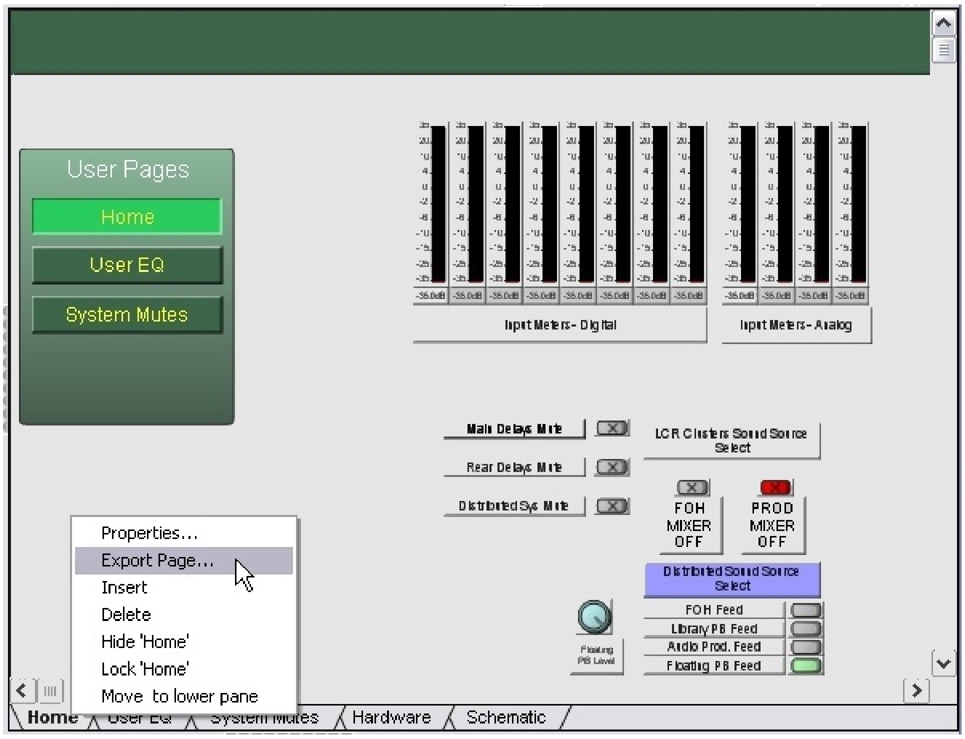
Deleting a page group

Page groups are deleted automatically when they no longer contain any exported pages. In order to delete a page group, you must change the export settings for each page in the group and remove it from the group.

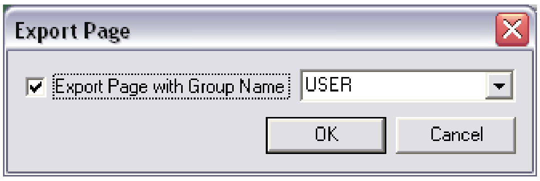
Tip: You can locate all the pages in a particular page group using the Find function. For more information, see *Finding exported pages* (on page 21).

⏩ *To delete a page group*

1. Right-click the page tab of a page in the group you want to delete, and then click **Export Page**.



The **Export Page** dialog box is displayed.



☑ Export Page with Group Name: USER

2. Clear the **Export page with Group Name** check box.
3. Click **OK**.
4. Repeat the above steps for all the pages in the group.

Once you have completed the process, the group will be deleted and the group membership settings of the users that used to belong to the group will be updated.

Setting the preferences in user accounts

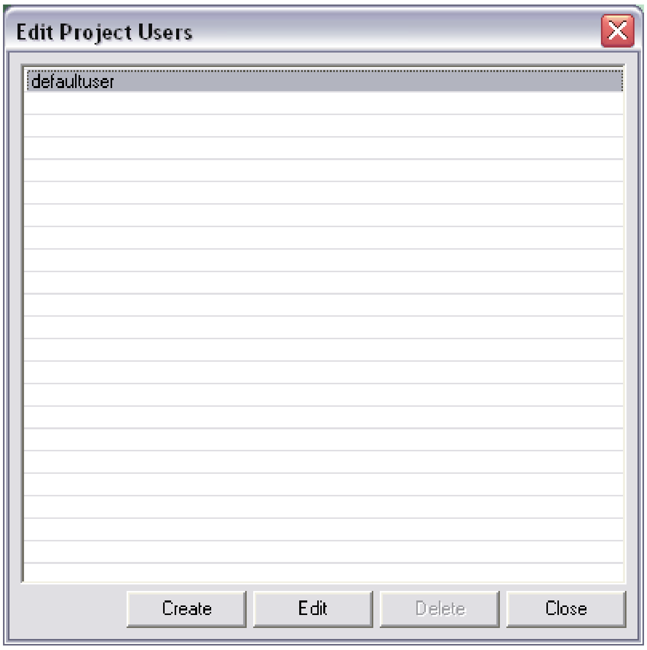
When you run Kiosk, you will be required to log on before you can view exported pages. You can specify any username from the NWare project, as long as the user has been granted access to the exported pages.

Tip: The procedure below shows how to specify the settings for a typical Kiosk user. For particular installations, however, you may want to use different settings, to restrict user actions further or allow a greater number of actions to be performed.

⏩ *To set the preferences in user accounts*

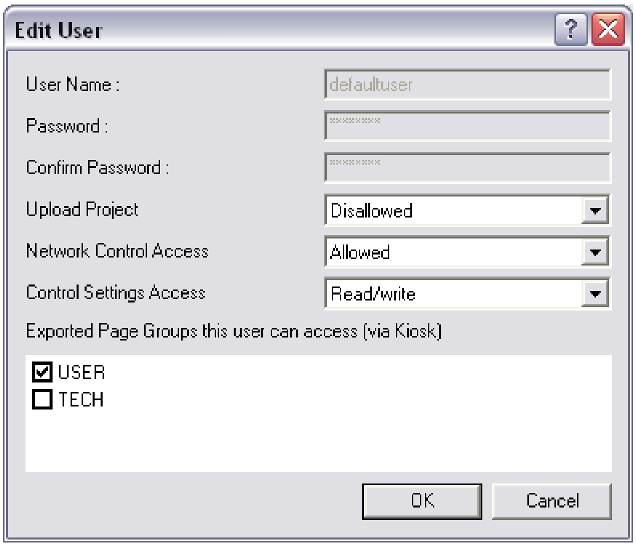
1. On the **File** menu, click **User Accounts**.

The **Edit Project Users** dialog box is displayed.



2. Click the user you want to configure, then click **Edit**.

   The **Edit User** dialog box is displayed.



Edit User

User Name : defaultuser

Password : ********

Confirm Password : ********

Upload Project: Disallowed

Network Control Access: Allowed

Control Settings Access: Read/write

Exported Page Groups this user can access (via Kiosk)

☑ USER

☐ TECH

3. In the **Upload Project** list, click **Disallowed**.
4. In the **Network Control Access** list, click **Allowed**.
5. In the **Control Settings Access** list, click **Read/write**.
6. In the **Exported Page Groups this user can access (via Kiosk)** list, select the check box next to each page group the user will access.

Note: Pages are organized into specific groups when they are exported, so you must specify the groups the user may access, otherwise they will not be able to use the pages in Kiosk.

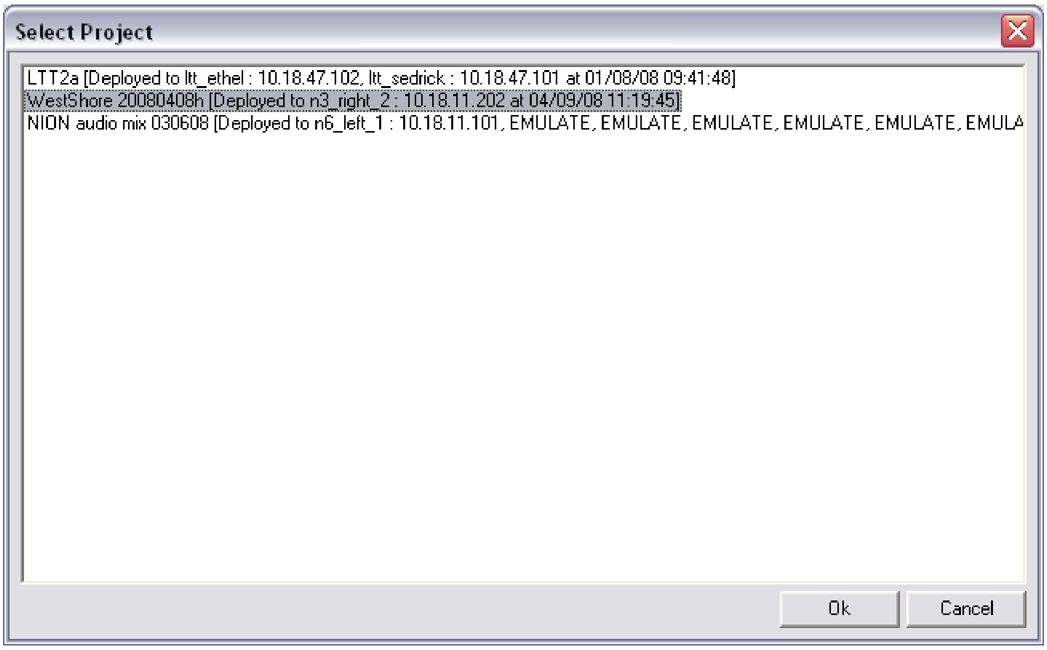
---

7. Click **OK**.
8. Click **Close**.

Starting Kiosk and selecting a project

1. On the **Start** menu, point to **All Programs**. Point to **NWare**, and then click **NWare Kiosk <version>**.

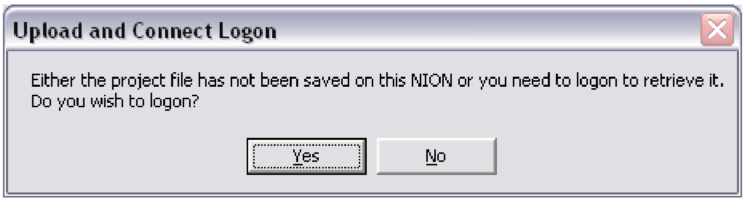
   The **Select Project** dialog box is displayed.



LTT2a [Deployed to ltt_ethel : 10.18.47.102, ltt_sedrick : 10.18.47.101 at 01/08/08 09:41:48]
WestShore 20080408h [Deployed to n3_right_2 : 10.18.11.202 at 04/09/08 11:19:45]
NION audio mix 030608 [Deployed to n6_left_1 : 10.18.11.101, EMULATE, EMULATE, EMULATE, EMULATE, EMULATE, EMULA

2. Click the project you want to run in Kiosk.
3. Click **Ok**.

   If a logon is required, the **Upload and Connect Logon** dialog box will be displayed.

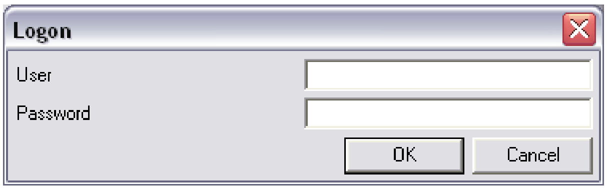


Either the project file has not been saved on this NION or you need to logon to retrieve it.
Do you wish to logon?

4. Click **Yes**.

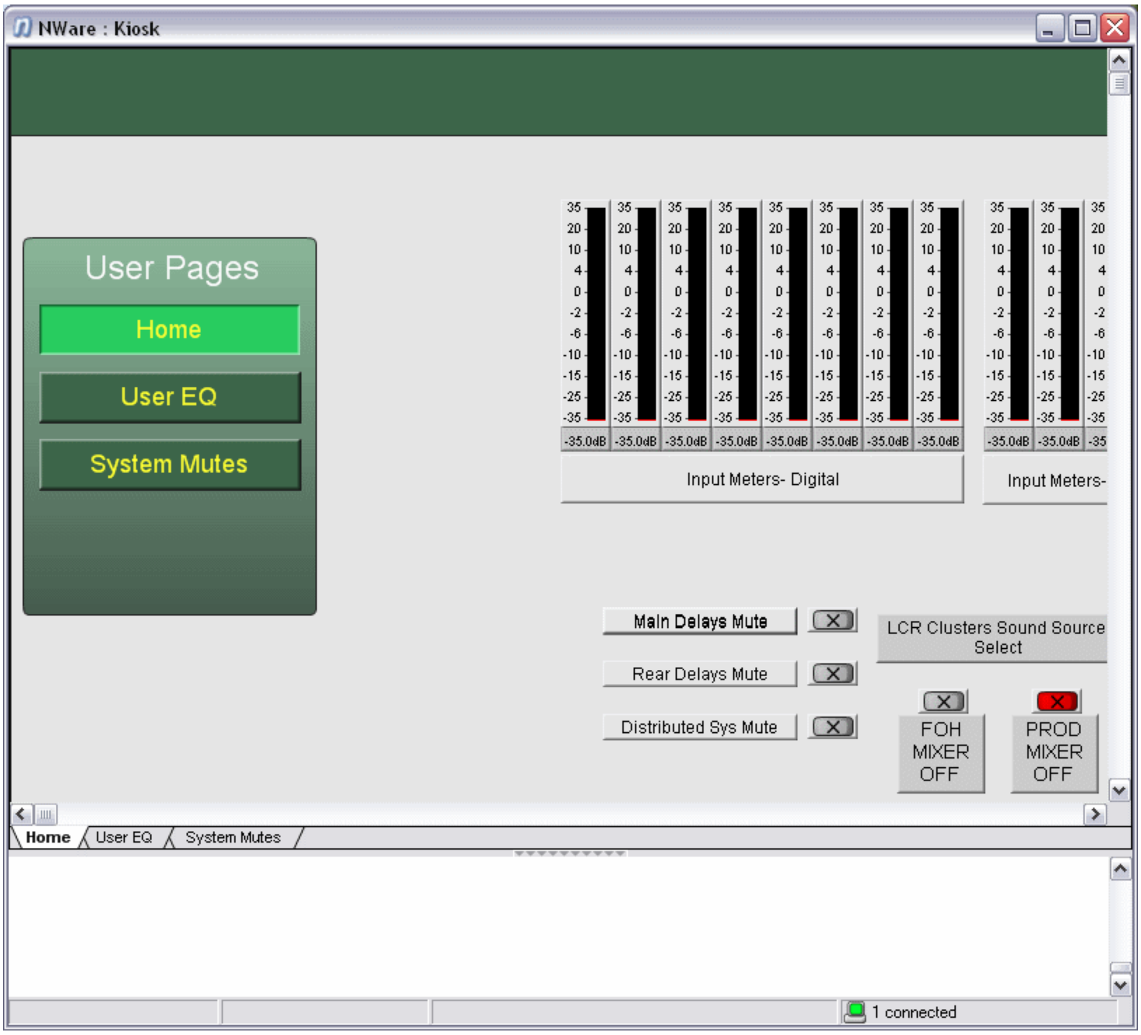
Tip: The part of the message about the project not being saved can be ignored. Only the part about logging on is relevant here.

The **Logon** dialog box will be displayed.



5. Type the username and password, and then click **OK**.

The Kiosk main window will be displayed.



The window will show individual tabs for the pages you have exported.

Tip: Some UI features normally available in NWare will be disabled. You can also disable further UI features using an XML configuration file. For more information, see *Controlling the display of Kiosk UI features using a personality file* (on page 27).

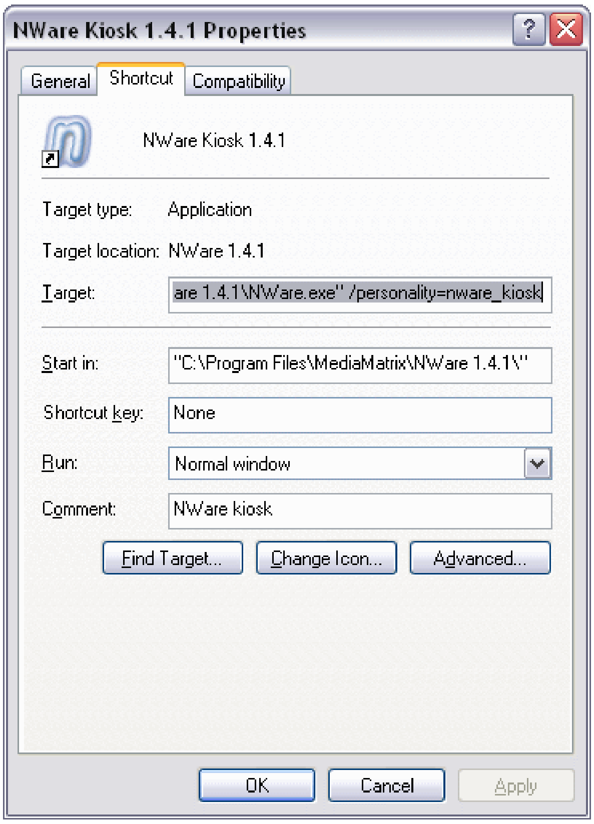
Logging on and loading a project automatically

You can avoid having to specify a username, password and a project name each time you start Kiosk, by editing the Kiosk Start menu shortcut and specifying command line options.

⏩ *To log on and load a project automatically*

1. On the **Start** menu, point to **All Programs**. Point to **NWare**, and then right-click **NWare Kiosk <version>**.
2. Click **Properties**.

   The **NWare Kiosk <version> Properties** dialog box is displayed.



NWare Kiosk 1.4.1 Properties

NWare Kiosk 1.4.1

Target type: Application

Target location: NWare 1.4.1

Target: are 1.4.1\NWare.exe" /personality=nware_kiosk

Start in: "C:\Program Files\MediaMatrix\NWare 1.4.1\"

Shortcut key: None

Run: Normal window

Comment: NWare kiosk

3. Add the following switches to the command in the **Target** box:

```
/user = <username>
/pass = <password>
/open = <project name>
```

   Where `<project name>` is the name of the NWare *.npa* file without the `.npa` included.

   Here is an example:

```
"C:\Program Files\MediaMatrix\NWare 1.4.1\NWare.exe"
/personality=nware_kiosk /user=user1 /pass=passwd /open="nion project"
```

4. Click **OK**.

   When you start Kiosk, the program will attempt to log on to the NION using the username and password you specified, then load the project file.

Tip: When you are running the full version of NWare, the `/personality=nware_kiosk` switch is used to specify the NWare Kiosk operating mode, as opposed to the NWare designer operating mode.

Controlling Kiosk features using a personality file

Introduction

You can use an XML file, called a *personality* file, to control the availability of certain UI features in Kiosk, and also for automatically logging on to Kiosk when it is started. The personality file is specified as part of the shortcut for Kiosk.

If you are running the full version of NWare, the shortcut looks like this:

```
"C:\Program Files\MediaMatrix\NWare 1.4.1\NWare.exe"
/personality=nware_kiosk my_personality_file
```

If you have installed the special version of NWare that only operates in Kiosk mode, the shortcut looks like this:

```
"C:\Program Files\MediaMatrix\NWare 1.4.1\NWare.exe"
my_personality_file
```

Tip: If your personality file is located in a different folder to *NWare.exe*, you will need to add the path to the file name. For example: `"C:\Program Files\MediaMatrix\NWare 1.4.1\NWare.exe" /personality=nware_kiosk "C:\Program Files\MediaMatrix\NWare 1.4.1\personality\my_personality_file"`.

As well as being an alternative to the `/user`, `/pass` and `/open` command line options (see *Logging on and loading a project automatically* (on page 26)), the automatic log on feature also provides a way to hide the password using hashing, so it cannot be read by someone who edits the XML file.

Example XML file

```
<kiosk>
    <name value="kiosk_test" />
    <username value="user1" />
    <hash value="" /> <!-- can use prehashed password if you don't
want to store the password in plain text in this file -->
    <password value="mypass" />
    <hide_root_tab value="true" /> <!-- defaults to false -->
    <hide_cursors value="true" /> <!-- defaults to false -->
</kiosk>
```

Specifying the basic project settings

1. Add a line to the Kiosk XML file to specify the name of the project *.npa* file (without the *.npa* extension). Here is an example:

   `<name value="kiosk_test" />`

2. If you want to log on automatically using a username and password, specify these settings:

   `<username value="user1" />`

```
<password value="passwd1" />
```

3. Save the personality file.

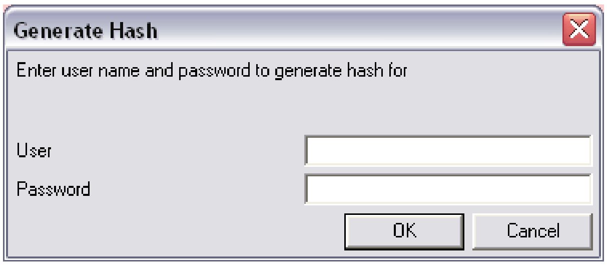
Hiding the password using hashing

The XML personality file is a text file that can be opened in any text editor, such as Windows Notepad. This means it is possible for a user to locate the file and read the password. To protect against this happening, you can encrypt the password using hashing, and specify the encrypted version in the personality file, instead of the original version.

▸▸ *To hide the password using hashing*

1. In NWare, press CTRL+SHIFT+H.

   The **Generate Hash** dialog box is displayed.

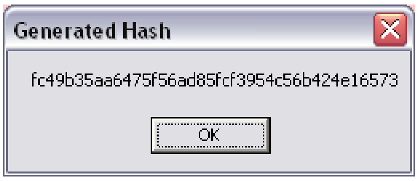


Generate Hash

Enter user name and password to generate hash for

2. In the **User** box, type the username.
3. In the **Password** box, type the password.
4. Click **OK**.

   The **Generated Hash** dialog box is displayed.



Generated Hash

fc49b35aa6475f56ad85fcf3954c56b424e16573

5. Make a note of the hash value.
6. Edit the personality file and remove the `<password value= ... />` line.
7. Add in a new line to specify the hash value. For example:

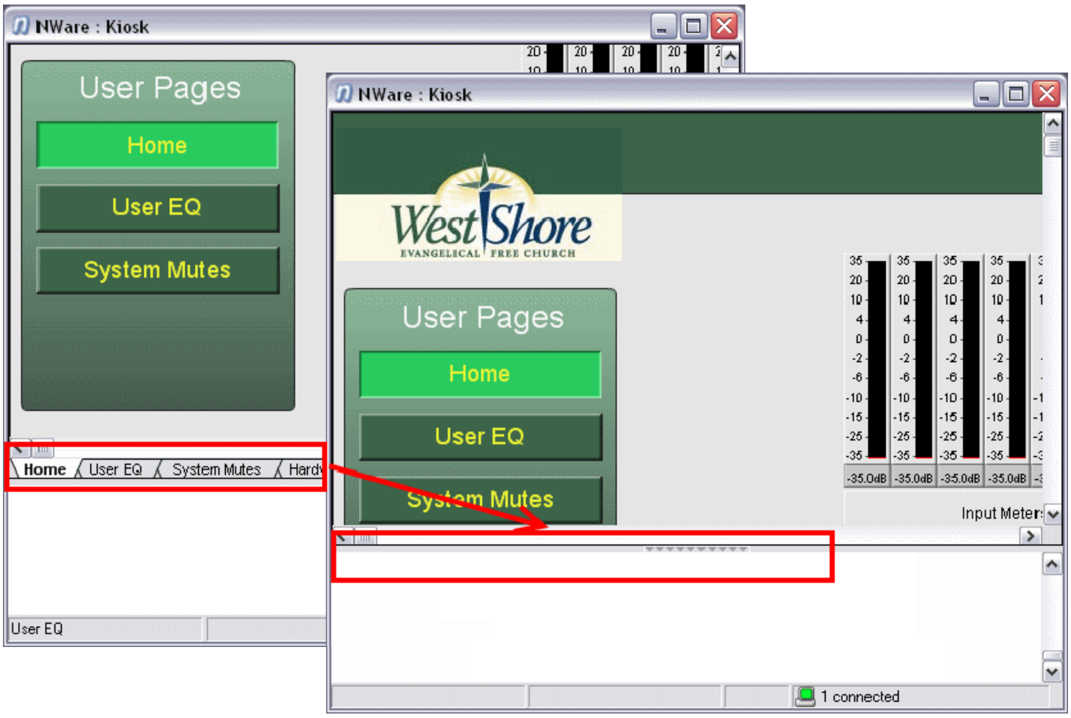
```
<hash value="fc49b35aa6475f56ad85fcf3954c56b424e16573" />
```

8. Save the personality file.

Hiding Kiosk UI features

Hiding page tabs

You can hide the page tabs that normally appear at the bottom of the Kiosk window.



⏩ *To hide the page tabs*

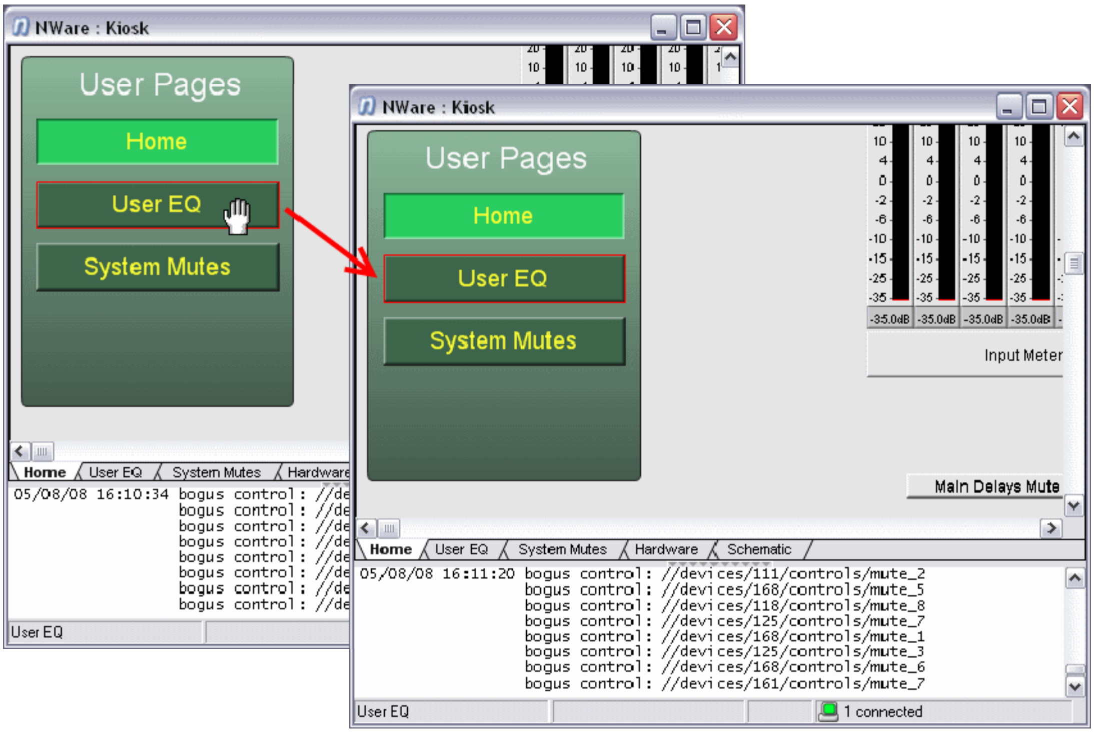
1. Edit the personality file and add the following line:

```
<hide_root_tab value = "true" />
```

2. Save the personality file.

Hiding the cursor

You can hide the cursor that normally appears when you move the mouse pointer over the Kiosk window.



```
05/08/08 16:10:34 bogus control: //de
                  bogus control: //de
                  bogus control: //de
                  bogus control: //de
                  bogus control: //de
                  bogus control: //de
                  bogus control: //de
                  bogus control: //de
```

```
05/08/08 16:11:20 bogus control: //devices/111/controls/mute_2
                  bogus control: //devices/168/controls/mute_5
                  bogus control: //devices/118/controls/mute_8
                  bogus control: //devices/125/controls/mute_7
                  bogus control: //devices/168/controls/mute_1
                  bogus control: //devices/125/controls/mute_3
                  bogus control: //devices/168/controls/mute_6
                  bogus control: //devices/161/controls/mute_7
```

⏩ *To hide the cursor*

1. Edit the personality file and add the following line:

   `<hide_cursors value = "true" />`

2. Save the personality file.

Troubleshooting

This section contains descriptions of some problems you could run into while using Kiosk, and some suggested solutions.

No projects are listed on the Select Project dialog box when Kiosk starts

Pandad has crashed.

When Pandad crashes, it can be recovered without the user shutting down NWare.

⏩ *To recover Pandad after a crash*

1. On the control computer, press CTRL+ALT+DEL to bring up the Windows Task Manager.
2. Click the **Processes** tab.
3. In the **Image Name** column, click **pandad.exe**.

Tip: You can click on the column header to sort all entries alphabetically.

4. Click **End Process** to close pandad.exe.
5. Close or minimize Task Manager.

6. Open Windows Explorer, then navigate to the root of the version of NWare you are using (the root is where you find *NWare.exe*).
7. Click to start pandad.exe in *Interactive Mode*.

   A command line box will open.

Pandad.exe should once again be present in the Windows Task Manager. In NWare all the NioNodes should once again present. It may take a few seconds for the list to be completed and updated. It will now be possible to continue with an interrupted deploy or finish an upload and connect. The NioNodes will also appear in your remote log.

The project you want to use is not listed on the Select Project dialog box when Kiosk starts, but other projects are

Check that you can connect to the NION across the network.

Check that the project has been deployed to the NION you are connected to.

I cannot see a particular page I expected to see in Kiosk

Check that the page has been exported from the project.

Check that the page has been exported to a page group that you are authorized to access.

My user account does not allow me access to Kiosk

Check that the designer has added the page group (to which the exported pages belong) to your user account settings.

Check that the designer has selected **Allowed** next to **Network Control Access** in your user account settings.

Chapter 6

Control devices

In This Chapter

Control Programming

Control Programming refers to the set of features that allow the designer to create and customize interactions between Device controls. Keep in mind that these interactions do not occur in the graphical user interface, but are part of the configuration of the NioNodes this is good because that means they work without the NWare PC and can be stimulated by an external 3rd party control system.

Control Settings

A Controls setting is represented in three ways: by position, string and value. Position is the fractional number corresponding to the graphic knob or slider position in the user interface, and ranges between 0 and 1. String is the text display string for the Control in the user interface. Value is the 32-bit internal number representation of the Control, and while not normally portrayed in the user interface, can be of use in Control Programming. As an example, a gain settings position might be 0.55, while its string is -45dB and its value is -45.0.

Control Wiring and Groups

Controls can be graphically wired to one another to form Control Groups. A Controls wiring properties are accessible in its Control Properties dialog in the Wiring tab. In the Control Wiring dialog box, you can independently enable the display of the three available Control Wiring nodes for that Control: slave, peer and master. These nodes display, left to right, underneath the Control.

Here are the types of wired group relationships:

- Master-to-slave - a Control master node is wired to one or more Control slave nodes. The slave Control(s) will track the master, and become read-only, meaning that it (they) cannot be directly gestured.
- Peer-to-peer - a Control peer node is wired to one or more other Control peer nodes. In this case, any of the Controls can be gestured, and they will track each other.
- Slave-to-slave - wiring Control slave nodes to one another can be done, but does not accomplish anything unless a master is also wired into the set.

Controls wired directly to one another are linked by position, that is, the graphical position, between all the way down and all the way up, of the linked Controls will track.

There is a shortcut to forming a peer-to-peer group of two or more Controls. Select the desired Controls, and then either run the Tools | Control Group menu command, or perform a right-click | Control Group.

Control Processing

A more powerful approach to creating interactions between controls is to use Control Processing. In this case, Controls are wired to Control Processor devices - a Controls master node is wired to the input node of a Control Processor, and one or more Control slave nodes are wired to the output node of a Control Processor. Control Processing supports the linking and processing of Control settings by position, string and value. (Recall that direct Control-to-Control wiring only supports position-based linking.)

There are two types of Control Processor devices: Control Operator and Control Scripter.

Appendix A

Internal latencies and alignment

In This Appendix

Internal latencies and alignment

Here is some valuable information regarding latencies in NION. These are necessary in calculating how much alignment is necessary.

Latencies in a NION system are as follows:

- IO Card serial port receive: 1 vector
- IO Card serial port transmit: 1 sample
- DSP processing: 1 vector
- DSP to DSP transfer (within a NioNode): 1 vector
- XDAB transfer (NioNode to NioNode): 3 vectors
- IO Card ADC: 34 samples @ 48/96 KHz
- IO Card DAC: 37 samples @ 48 KHz, 20 samples @ 96 KHz

With that said, it's not always so easy to manually count in your head the latency of a particular signal relative to another. For instance, in the design below, the align must be set to 4 vectors (32 samples):

- DSP A processing (sine)
- transfer A to B
- DSP B processing (Gain)
- transfer B to A
- DSP A processing (2x1 mixer)

You can think of it as counting up the vectors between the 5 stages I listed. It is easier to create your design and check the delay report to see what manual delays you need to insert.

Appendix B

Front panel and GPIO external control

In This Appendix

Front Panel Control

Controls can be made accessible through a NioNode hardware front panel interface by control wiring them to NioNode Frontpanel Control Devices.

GPIO Control

A GPIO, or General Purpose Input Output, connector is provided on the rear panel of each NioNode. In addition to the fault relay pin signals, there are 20 configurable pins available for a variety of uses.

GPIO Device

The use of the NioNode hardware general purpose input output pins for control begins with configuring one or more GPIO Devices in the Design. The GPIO Device manages the pins found on the hardware GPIO connector on the back of a NioNode.

The Device Properties of the GPIO Device presents a selection for each of 20 hardware pins. Configuring these pins for various functions will create the appropriate Controls in the controls page of the Device. The designer then uses Control Wiring to interconnect these Controls to other Controls in the Design.

Here are the possible settings for the GPIO pins:

- unused
- raw
- digital in
- digital out
- analog in –1k 12V
- analog in –10k 12V
- analog in –10k 24V
- analog in –1k self-powered
- analog in –10k self-powered
- digital rotary encoder
- high current out
- PWM output.

GPIO Hardware Reference

The NioNode GPIO connector is a standard Female DB-25. There are three sections to the GPIO: General Input/Output Pins, High Current Outputs, and the Fault Relay Contacts.

General Input/Output Pins (GIO)

There are 16 GIO pins. Each pin can be independently configured as a Digital Input, Digital Output, or Analog Input.

- Digital Input - Accepts TTL level inputs, where a low is less than 0.8 volts and a high is greater than 2 volts.
- Digital Output - Outputs an LVTTL signal, where a low is 0 volts and a high is 3.3 volts. As an output, this pin cannot sink or source very much current. There is not enough current to drive an LED, but there is sufficient current to switch a transistor.
- Analog Input - Accepts an analog signal from 0 to 24 volts with 12 bits of precision.

The general purpose I/O pins are heavily filtered and protected. They can withstand being shorted to a 36 volt supply without sustaining any damage.

Additionally, the 16 digital inputs can be fed into eight quadrature rotary decoders.

High Current Outputs (HCO)

There are 4 HCO pins. Each switchable output can supply +11.5 volts (nominal) at 0.50 amps. There are internal protection diodes so these outputs can directly drive relays and other inductive loads. A self-resetting fuse also protects each output.

The high current outputs can only sink a very small amount of current.

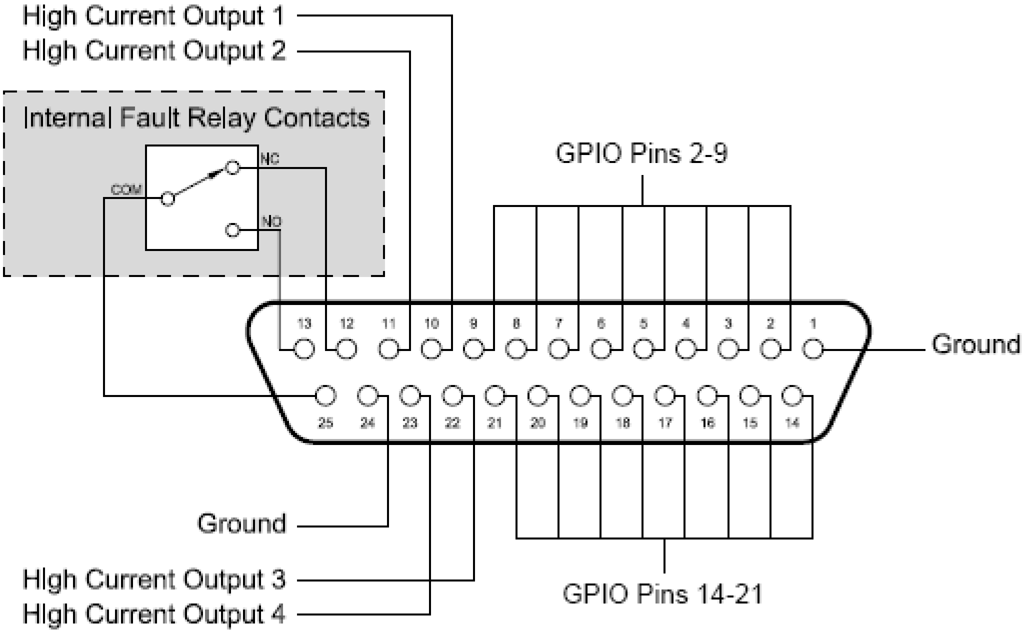
There is no dedicated power pin on the GPIO connector. If a dedicated power pin is required, then a high current pin can be switched on to provide power.

Fault Relay Contacts

The internal fault relay trips in the event of a power failure or any number of hardware or software fault detection events. Normally Open (closed on fault condition or power off) and Normally Closed (closed when no fault condition) contacts are provided. These are electrically isolated from the rest of the system.

GPIO pin assignment

The illustration below shows the pin assignments for the GPIO system. If you are using the GPIO-25, use the screened pin numbers to identify the functions available for the associated captive wire terminal.



Pin	**Function**	**Pin**	**Function**
1	Ground	14	Configurable GPIO
2	Configurable GPIO	15	Configurable GPIO
3	Configurable GPIO	16	Configurable GPIO
4	Configurable GPIO	17	Configurable GPIO
5	Configurable GPIO	18	Configurable GPIO
6	Configurable GPIO	19	Configurable GPIO
7	Configurable GPIO	20	Configurable GPIO
8	Configurable GPIO	21	Configurable GPIO
9	Configurable GPIO	22	High current output 3
10	High current output 1	23	High current output 4
11	High current output 2	24	Ground
12	Fault Relay N.C.	25	Fault Relay Common
13	Fault Relay N.O.		

Appendix C

Python control scripting

In This Appendix

Python control scripting

Python, used in the Control Scripter device, is an open source scripting language that has been ported to run on many platforms. The home page for Python is *http://www.python.org/* (http://www.python.org/). A free, Windows-based Python development package, called ActivePython, is available from ActiveState at *http://www.activestate.com/* (http://www.activestate.com/).

In Nion, Python functions can be authored by the designer to implement custom control interactions. A script function will be run whenever any of the Controls wired to the input(s) of the Control Scripter Device referencing it changes. Scripts currently run within a limited environment for safety reasons –i.e. they do not have access to operating system level APIs such as the file system, networking, etc. Scripts do, of course, have the ability to get and set Control settings.

Four pre-defined variables are passed into the script function of a Control Scripter Device - inputs, outputs, message and state. Variables inputs and outputs are Python vectors referencing the Controls wired to the Device's input and output nodes. So, inputs[0] returns a reference to the Control wired to the first input node, outputs[2] returns a reference to the Control wired to the third output node, etc.

Functions for an input Control are:

- position_get() - returns position in float format
- position_set(value) - sets position in float format
- value_get() - returns value in numeric format
- value_set(value) - sets value in numeric format
- string_get() - returns value in string format
- string_set(value) - sets value in string format
- changed_get() - returns true if the input Control has changed since the last time the script function was run.

Functions for an output Control are:

- position_set(value) - sets position (value must be float or int)
- value_set(value) - sets value (value must be float or int)
- string_set(value) - sets value (value must be string)

Variable message is a text style control, displayed in the Control Scripter Device, which has the following function:

- string_set(value) - sets value in string format

Variable state is a persistent variable that defaults to an integer with a value of 0.

Here is an example of an averaging function that uses message and state:

```
result = 0

state = state +1

message.string_set( "this function has been called " + str( state ) + "
times" )

# for each input, get the position and add it to result

for n in inputs :

  result += n.position_get()

# set output 0 to the average value of the inputs

outputs[0].position_set( result / len(inputs) )
```

Note: The '#' character starts a comment line in Python.

Note: Line indentation is how block scoping is done.

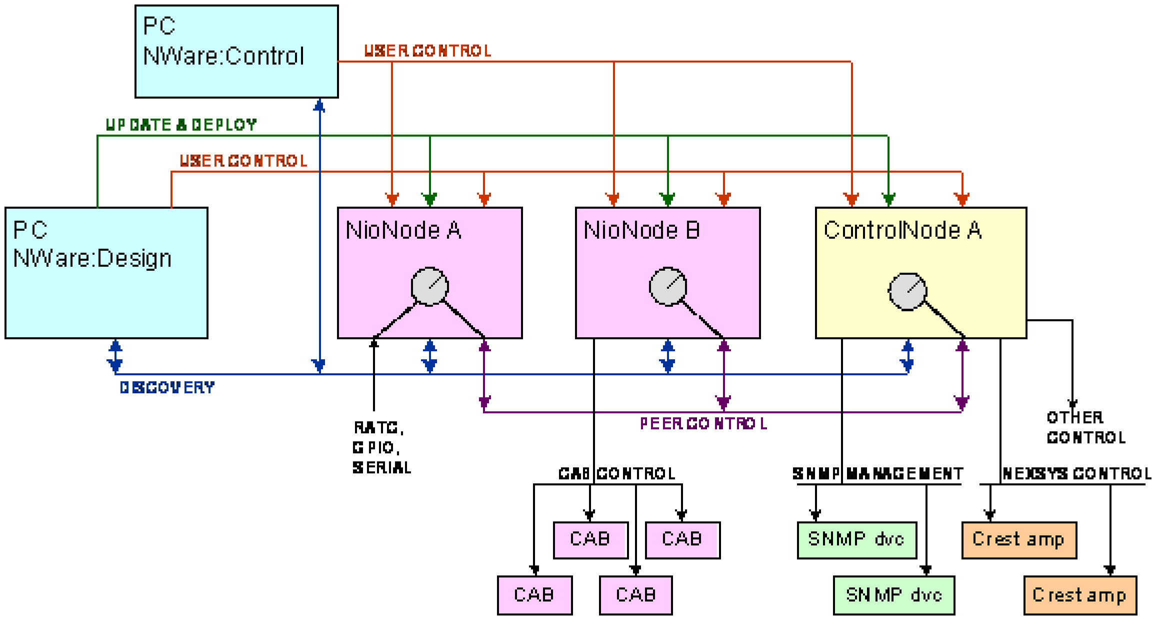
Appendix D

Understanding Nion communications

In This Appendix

Introduction

There are many patterns of communication active in a Nion system. This section gives a brief overview. Not all communication and control protocols are listed.



Discovery

A network discovery method is required to enable NWare hosts and Nion targets (NioNodes and ControlNodes) to discover one another on the network. On the PC-based NWare hosts, the service that accomplishes this is called PANDAD.EXE, and has the label "Target discovery". PANDAD.EXE is started the when the first copy of NWare is started, and is shut down when the last copy of NWare is shut down.

Update and deploy

This is the communication channel through which the NWare Design program can update target firmware and deploy compiled Projects to the targets.

User control

The communication channel through which NWare Design and NWare Control accomplishes knob twiddling and meter reading in a running Project.

Peer control

The communication channel through which controls are linked between target nodes. This communication is automatic and is not directly specified by the Project designer. For instance, if you link two controls together that happen to be on different target nodes, peer control is used to make this work.

In addition, any Control that is given a Control Alias, which enables it to be externally controlled via RATC, can be controlled by a client program (such as Telnet) connecting to any target node –you don't need to know which node the control actually resides on. This is accomplished behind the scenes through Peer Control.

Cab control

CABs assigned to a NioNode are controlled by that NioNode using this control protocol. This protocol involves both CAB discovery and parameter exchange. It is not SNMP-based.

SNMP management

ControlNodes (which only perform control processing and not audio processing), manage SNMP devices such as CobraNet devices, Ethernet switches and UPSs (uninterruptible power supplies).

NexSys control

Control of NexSys compatible amplifiers.

Other control

Other miscellaneous control capabilities will be added to the ControlNodes as the need arises.

Appendix E

Keyboard accelerators

In This Appendix

File menu

New	Ctrl+N
Open	Ctrl+O
Save	Ctrl+S
Save As	Ctrl+Alt+S
Deploy	F9
Emulate	Ctrl+F9
Compile	Shift+F9

Edit menu

Undo	Ctrl+Z Tip: 100 undo levels are supported.
Redo	Ctrl+Shift+Z
Cut	Ctrl+X
Copy	Ctrl+C
Select Position / Paste	Ctrl+V
Duplicate	Ctrl+D
Duplicate Special...	Ctrl+Shift+D
Delete	Delete / Del
Select All	Ctrl+A
Find...	Ctrl+F

Align

Align Left	Ctrl+Shift+L
Align Top	Ctrl+Shift+T
Align Right	Ctrl+Shift+R
Align Bottom	Ctrl+Shift+B

Pack

Pack Left	Ctrl+L
Pack Top	Ctrl+T
Pack Right	Ctrl+R
Pack Bottom	Ctrl+B

Arrange

Bring To Front	Ctrl+]
Send To Back	Ctrl+[

View menu

Show/Hide Output Frame	F10
Show/Hide Utility Frame	F11
Show/Hide Status Bar	F12

Zoom

Zoom Out	Ctrl+PageUp
Zoom Normal	Ctrl+home
Zoom In	Ctrl+PageDown
Zoom Fit All	Ctrl+0 (zero) **Note:** The zero key on the numeric keypad cannot be used to select this function.

Mode menu

Edit Mode	Ctrl+E
Wire mode	Ctrl+W
Gesture Mode	Ctrl+G

Paint Mode	Ctrl+P

Tools menu

Control Group	Ctrl+I
Control Ungroup	Ctrl+Shift+I
Lock	Ctrl+Alt+L
Unlock All	Ctrl+Alt+U
Replace Text	Ctrl+8
Insert waypoint	Ctrl+Shift+W
Change Flyoff Direction	Tab
Open Object	Enter
Display Object Properties	Alt+Enter
Display Block Properties	Alt+Shift+Enter
Device Placement...	Ctrl+Shift+Enter

Expression labeling

Expression Label Flyoff	Ctrl+Shift+F
Expression Label Name	Ctrl+Shift+E
Expression Label Alias	Ctrl+Shift+X

Control Nodes

Set Master Node for control	Ctrl+Shift+M
Set Slave Node for control	Ctrl+Shift+S
Set Peer Node for control	Ctrl+Shift+P

Windows menu

Next Window	Ctrl+Alt+N
Previous Window	Ctrl+Shift+N

Next Page	Ctrl+Tab
Previous Page	Ctrl+Shift+Tab