# QNX® Neutrino® Realtime Operating System

## Phindows™ Connectivity
### *User's Guide*

*For QNX® Neutrino® 6.2 or later*

Printed under license by:

**QNX Software Systems Co.**
175 Terence Matthews Crescent
Kanata, Ontario
K2M 1W8
Canada
Voice: +1 613 591-0931
Fax: +1 613 591-3579
Email: **info@qnx.com**
Web: **http://www.qnx.com/**

### Publishing history

Electronic edition published 2005.

### Technical support options

To obtain technical support for any QNX product, visit the **Technical Support** section in the **Services** area on our website (**www.qnx.com**). You'll find a wide range of support options, including our free web-based **Developer Support Center**.

# Contents

# *About This Guide*

Phindows is a connectivity tool that lets you use Microsoft Windows platforms to connect to and interact with graphical Photon applications running on a remote Neutrino computer.

# Starting Phindows

When you start Phindows, it displays a Connect dialog where you can specify the type of connection (TCP/IP or direct-connect serial), compression type, and cache settings. Various connection options are available, but the defaults usually work well.

Once you've chosen the connection parameters, click Apply to cause Phindows to connect with these parameters. To record these settings permanently before making the connection, click Apply & Save. These saved settings become the defaults for all future Phindows sessions.

If you request a TCP/IP connection, you must also specify the Internet address of the Neutrino computer you're connecting to (e.g. **`198.53.31.1`**), or the hostname. If the remote computer has been configured properly, you should see a Photon login prompt, at which point you're connected and running Photon.

If you request a serial connection, you must specify the COM port (e.g. COM1 or COM2). If you don't specify a baud rate, Phindows uses the current Windows default settings. With a serial connection, Phindows initially acts as a simple text terminal that lets you type commands directly to the modem (e.g. ATDT1-613-591-0934). Once connected, log into Neutrino and then issue the command:

On QNX 4: **`/usr/photon/bin/phrelay`**

On QNX Neutrino:

**`/usr/bin/phrelay -x`**

This command causes Phindows to drop out of text-terminal mode and begin acting as a Photon graphical terminal. A Photon login screen should appear at this point.

## Configuring for TCP/IP use

If you're using TCP/IP, there are several configuration issues that you need to check before you can use Phindows. The configuration should already be set correctly; if not, make the changes described below.

First, the remote QNX host must have TCP/IP installed and running. In addition, **inetd** must be running with the following additional items added to the indicated TCP/IP configuration files:

In **/etc/inetd.conf**, add the line:

On QNX 4:

```
phrelay stream tcp nowait root /usr/photon/bin/phrelay phrelay
```

On QNX Neutrino:

```
phrelay stream tcp nowait root /usr/bin/phrelay phrelay -x
```

In **/etc/services**, add the line:

```
phrelay 4868/tcp
```

These lines are already present in the configuration files, but they're commented out. Just remove the number sign (**#**) to add these entries.

These two entries cause **inetd** to listen for incoming requests to establish a new Photon session. When a request is detected (from a remote Phindows client in our case), **inetd** automatically establishes a full TCP/IP connection and launches **phrelay** on that connection. Phindows is then fully connected to the local machine.

For more information about **inetd**, **/etc/inetd.conf**, and **phrelay**, see the QNX Neutrino *Utilities Reference*.

All that remains now is to configure the Windows client to support TCP/IP. The exact way to do this depends on the type of Windows platform you're using, what services have already been installed on that client, and whether you're using Ethernet or dialup networking over SLIP/PPP (see the Microsoft documentation for help with this).

## Data-compression options

You can use the Phindows Connect dialog to specify one of these data-compression options:

Byte-Pair Encoding (BPE)

Performs the best data compression and is recommended for all serial connections. Although BPE is inexpensive to decode (both in memory and CPU usage), it does require a fair amount of CPU on the host QNX machine. When link speeds are low (such as with modem links), the extra cost in CPU on the sending side is well worth the tradeoff.

Run Length Limited (RLL)

Provides good data compression. Although not as highly compressed as BPE, RLL encoding is both quick to decode and encode. We recommend RLL encoding for network TCP/IP connections because it places less CPU load on the QNX host, yet provides fair compression.

None With data compression turned off, Photon draw events are transmitted as-is. This might make sense if the connection bandwidth is VERY high (say 100Mb Ethernet), making the extra processing time involved in data compression an unnecessary step.

## Data-caching options

Phindows makes very extensive use of data caching techniques to minimize the amount of data that needs to be transmitted from the host to the client machine. Thanks to this intensive data caching, Phindows is usable even on modem-link speeds as low as 9600 baud.

Most large static data objects in Photon are tagged with a unique 32-bit ID, or *tag*. Tagged objects include bitmap data, image data, and color palettes. These objects are generally tagged when first created by the Photon program developers. This process is accomplished automatically by the Photon development tools, so the program developer is, for the most part, not even aware that this is taking place.

Photon passes these tags along with the data objects as they flow through the Photon event space. The end result is that Photon graphics drivers (such as **phrelay**) almost always have available a small, unique, precalculated tag to identify the larger Photon graphical objects.

The host side of a Phindows connection, **phrelay**, has the job of transmitting the graphical Photon events over a communications link to a Phindows client. Phindows and **phrelay** use a private protocol that allows these large graphical objects to be sent only once to the remote client. All subsequent references to that data object are made by passing the small tag, knowing that the Phindows client has a local copy of the entire data object. Over time, commonly encountered objects (such as icons and backdrops) all become cached.

Phindows not only caches these tagged data objects in local memory, but also spills least-recently used objects to disk automatically and saves all cached objects to disk when you close Phindows. The next time that you start Phindows, it preloads its in-memory cache from the most recently encountered disk cache and informs the remote **phrelay** session which objects it already knows about. This means that after a graphical object is seen once within Phindows, it never, in general, has to be transmitted again, even in subsequent Phindows sessions.

The Phindows Connect dialog lets you tune the size of the in-memory data cache and the maximum amount of disk storage to reserve for most recently encountered objects. The default values of 4096K for RAM cache and 20M for disk cache are usually adequate.

## Using Windows fonts

In normal operation when connected to a remote host, phindows receives all text as bitmaps which have already been rendered by the Neutrino font manager. This mode of operation is best in most situations, but at very slow connection speeds you can improve performance by configuring Phindows to use local Windows fonts.

Phindows uses local Windows **.ttf** (TrueType) and **.phf** fonts rather than font bitmaps rendered by **phrelay**, if it finds them in its

font directory. This feature is available in **phrelay** 6.3 or greater and Phindows 2.13 or greater.

To use this feature, you need to:

1. Copy the **.phf** and **.ttf** font files to the **font/** subdirectory used by Phindows. Look in the **%SYSTEMROOT\phindows.ini** file to see the path to the Phindows directory, which contains this subdirectory. If no path is listed in this file, look in the directory where the **phindows.exe** executable resides.

2. Install the copied **.ttf** fonts in Windows using the Fonts control panel. This procedure depends on the version of Windows you are running. See the Windows online help for more information on installing fonts.

You can check which fonts are being picked up from the Neutrino or Windows side by starting **phrelay** with the **- V** and **-D** command-line options. For example, you can change the **/etc/inetd.conf** file to start **phrelay** as follows: **phrelay -x -V -Ddbg_log.txt**.

Then restart **phrelay** (for example, with **slay -s HUP inetd**) to get debug output sent to **/dbg_log.txt**. With the **-V**, debug lines for all text rendered using a font present at the Phindows end are recorded. To see which fonts are still being rendered by **phrelay** and sent as bitmaps, use the **-VVV** option, which produces more verbose debug information. In the resulting debug output look for the string **render opts**, which is output every time **phrelay** calls *PfRenderText()*. The line before this one gives the font filename, and a few lines after is an ASCII representation of the rendered text bitmap.

## Draw buffer size errors

When running applications that display a lot of text, it is possible to exceed the maximum draw buffer size. In this case, the window title displays the error: “[Error: Large Draw Buffers]”. For more information, see Draw buffer limits in **phrelay**.

# Command-line options

**PHINDOWS.EXE** supports several command-line options. You typically add these parameters to the command when you create the icon or shortcut to launch Phindows. The options are:

**-b***baud*[,*commbaud*]

The effective link speed. If you use a direct serial connection, then the comm port's baud rate is set to this value as well, unless you give a specific baud rate (*commbaud*). If you don't specify a baud rate, Phindows uses the Windows default.

**-h***height* Initial window height (default: 460).

**-H***t1*[,*t2*[,*t3*]] Specify the mouse holdoff times (default: *t1*=1.2*9600/baud, *t2*=*t1*/2, *t3*=*t1*/4):

- *t1* — holdoff time for normal mouse motion.
- *t2* — holdoff time when the a mouse button is pressed.
- *t3* — holdoff time when a drag cursor is being moved.

**-i***igrp* Connect to a specific Photon input group (default: 1).

**-K***key* Specify a private key for data encryption.

**-k** Start in kiosk (full-screen) mode.

**-m***commport* Direct connect serial port (default: **com1**)

Setting this option turns on RLL compression and CRC error checking by default. You can turn it off with the **-o** option.

**-N***number* Set the number of messages buffered to allow write-ahead draws to **Phindows**. The actual number used is the lesser of this option and the **-b**

option of **phrelay**. The default value is **20**. Use a lower value to save memory on the Neutrino host running phrelay at the expense of throughput, or, if memory is not an issue, a higher value to gain some additional throughput performance. Adjusting this setting has the most effect when end-to-end response time is slow compared with throughput, such as over a modem or when there are many network hops between the local and remote ends.

**-n***node_or_photon_resource*

Connect to an already-running Photon on the specified node (see below).

**-o** *options* Options:

- 0 — no compression.
- 1 — BPE (Byte-Pair Encoding) compression.
- 2 — RLL (Run Length Limited) compression.
- 8 — Use CRC (Cyclic Redundancy Check) error checking.

Combine options by addition, e.g. to specify BPE and CRC, select **9**. If the selected baud rate requires it, compression will be automatically selected unless you specify **0**.

The command-line options **-t** and **-m** turn on RLL compression by default. The **-m** also turns CRC error checking on. To adjust these side-effects you can use the **-o** option afterwards. For example, to run with no compression when specifying a TCP/IP connection from the command line, you would type: **phindows -t***host* **-o0**.

**-O**[**O**|**o**] Disable (**O**) or enable (**o**) offscreen context support. This option is enabled by default.

**-P***palette_file* Use a non-default color palette file (e.g. %QNX_TARGET%\usr\photon\palette\grey.pal).

**-s***service* Request that this Photon service be automatically started.

**-t***host_name* [:*port*]

TCP/IP address (and optional port) or host name of QNX host (e.g. 198.53.31.1, 198.53.31.1:4869, or myhost:4869).

Setting this option turns on RLL compression by default. You can turn it off with the **-o** option.

**-U***userid*[:*password*]

Initially log into this QNX userid (used with **-s**).

**-u** Unlocked mode (allows independent browsing of Photon session).

**-W"***Title***"** Window title (default: Phindows). The title must be quoted, and can't contain any spaces.

**-w***width* Initial window width (default: 620).

**-x***offset* Create a Photon region at this x offset (default: 0).

**-y***offset* Create a Photon region at this y offset (default: 0).

If you don't specify either **-t** or **-m**, then Phindows pops up the Connect dialog asking for communications parameters.

## Dittoing remote QNX Photon sessions

The normal mode of using Phindows (without the **-n** option) causes a private session of Photon to be started on the QNX machine you've connected to. You can use Photon's Jump Gates (on QNX 4) and Ditto facilities to interact with other Photon sessions on that same QNX machine (or any other QNX machine in the same QNX network), but your Photon session is otherwise independent of everyone else's.

The **-n** command-line option lets a Phindows client see and interact with any Photon session already running on one of the QNX machines in the same QNX network you've connected to. An exact copy of the remote Photon user's screen is displayed on your Windows desktop. In addition, you can use your mouse and keyboard to interact directly with that remote Photon session. You can, in effect, take over the remote screen as if you were sitting there.

## Remote support example

For example, if you want to provide support to a remote QNX site that's running Photon, dial up and log into that QNX machine using Phindows with a command line similar to:

```
phindows.exe -mcom2 -n/dev/photon
```

Log in and start **phrelay** as follows:

On QNX 4: **/usr/photon/bin/phrelay**

On QNX Neutrino:

```
/usr/bin/phrelay -x
```

At this point, **phrelay** is informed that a connection to an already-existing Photon session called **/dev/photon** is requested (i.e. the Photon running on the local console). The **phrelay** command creates a graphics region to overlay the entire console graphics region, so that both your mouse and the remote mouse can control the same cursor. You can then share that remote desktop with the remote user.

This type of remote connectivity is convenient for a variety of purposes, including:

- remote diagnostics
- remote monitoring
- remote technical support

- training.

Of course, there's no need to have a human actually sitting at the remote console. Nor is there a need even to have a physical screen and keyboard at the remote site. You can use Photon connectivity to control remote sites that are completely unstaffed. This not only saves on travel costs involved in remote diagnostics and maintenance, but also saves in hardware costs at the remote site (since you can eliminate the cost of keyboards and display screens).

## Connecting to a remote Photon session

Suppose you need to look at a Photon application running on some QNX machine other than the one with the modem you're connecting to. Not a problem. You can use the **-n** option to specify the full QNX pathname of any Photon session on that remote QNX network. So if you're dialing into one QNX node, but want to look at and interact with Photon running on another node's console, just use:

Targeting QNX 4:

**phindows.exe -n//***node_number***/dev/photon**

Targeting QNX Neutrino:

**phindows.exe -n/net/***node_name***/dev/photon**

When you connect to the QNX network (initially logged into the first node), start:

On QNX 4: **/usr/photon/bin/phrelay**

On QNX Neutrino:

**/usr/bin/phrelay -x**

The **phrelay** program recognizes that you really want to be connected to a Photon on the second node, so it automatically migrates itself to that node, but continues to use the modem you dialed in on (i.e. the modem on the first node). Since QNX is an inherently distributed operating system, this happens seamlessly and efficiently.

## Starting Photon sessions on other QNX nodes

Suppose the QNX machine you log into has lots of modems, but not a lot of spare CPU or memory. You want to start up a private Photon session and discover that node 2 has lots of spare resources (in QNX, any node will do). You can specify this with the following command:

Targeting QNX 4 only:

```
phindows.exe -n//2/dev/ph+
```

## Using a non-standard color palette

If you're connecting to a QNX machine that's using a color palette other than default.pal, you can specify the palette file using the **-P** command line option. The Windows display must be set to 256 colors (8-bit color) for this option to work.

For example, if the QNX machine you are connecting to uses the **grey.pal** palette file, use the following command-line option:

```
phindows.exe -P%QNX_HOST%\usr\photon\palette\grey.pal
```

## Spanning a single Photon session across multiple screens

You can use both Phindows and **phditto** (which comes standard with Photon) to stretch a single Photon space across multiple desktops. Some (or all) of these desktops can be QNX computers, some (or all) of these desktops can be Windows desktops, and some (or all) can be X workstations as well.

The **-x**, **-y**, **-h**, and **-w** options are provided in both **phditto** and Phindows to make this possible. Here's how they work:

Suppose you have a QNX machine running Photon on a console in 1024x768 resolution. You now want to stretch the Photon space to include the 1024x768 Windows screen you've placed immediately to the right of the QNX screen (creating a 2048x768 Photon desktop). You simply need to start Phindows on the Windows machine with:

```
phindows.exe -x1024 -n/dev/photon
```

When the connection is made (TCP/IP works well), you'll notice that the Photon desktop manager, which is normally on the bottom of the Photon screen, stretches to include the bottom of the right-hand MS-Windows screen as well. You can now take either mouse, start up new applications, drag them from screen to screen, and otherwise take advantage of your increased workspace.

With careful use of the **-x** and **-y** (and **-h** and **-w**) options, you can create many interesting combinations (an array of monitors forming one huge Photon display, multimedia presentations, etc.)

## Sharing a Photon session (workgroup computing)

The normal mode of Dittoing someone else's Photon session (specifying **-n**) is to have a single cursor that can be controlled by either the local or remote user. Similarly, keystrokes from either keyboard can be entered into the application with keyboard focus, which is normally what you want for a remote training or debugging session.

But sometimes you may want the remote user to carry on working without necessarily being affected by what you're doing. You can do this by specifying the **-u** option on the Phindows command line. In this mode, your mouse, keyboard, and display are completely independent of the other user's console. Both of you see two cursors (your cursor is solid, while the other person's cursor is dimmed or ghosted), but the two cursors behave independently.

In Photon terminology, you're running in your own private *input group*. You're free to roam around the Photon space without affecting the other user. Your mouse clicks and keystrokes are directed to whatever application has input focus for your input group. You can therefore start up new applications anywhere in the Photon space (probably in a different Photon console to be polite). Unless the other user chooses to roam over to the part of the desktop you're working in, he or she is otherwise unaffected by your presence there.

There's no built-in limit to how many concurrent users can share the same workspace in this manner, although it's doubtful that you'd practically want to have more than two at any one time. It's far more likely that if multiple users are connected to a single QNX machine, they each run in their own private Photon workspace (the default behavior of Phindows) and communicate with each other using the built-in Photon connectivity facilities, such as Jump Gates and **msgpad** (QNX 4 only), and **phditto**. A single QNX machine can easily support dozens of such Phindows clients, provided it's fast enough and has enough RAM.

## Enabling offscreen context support

Offscreen context support is enabled by default, though you can turn it off with the **-OO** option.

Offscreen context support has some limitations:

- For a given Photon session only one display receives offscreen draws. This means that if you use the **-n** option of Phindows to ditto an existing Photon session, offscreen draws are passed only to the Phindows display leaving the original display unchanged.
- The support of the offscreen contexts via phrelay is limited to uncomplicated cases where applications do regular draws followed by periodic blits from offscreen to the main display. This means:
  - The PdGetOffscreenContextPtr() function isn't supported.
  - The creation of offscreen locks isn't implemented.
- Using Phindows to connect to a Photon session where offscreen contexts currently exist or were previously orphaned will cause the software to attempt to re-create and re-draw those contexts. The accuracy that results depends on how well the applications currently running handle invalidation of their offscreen contexts.
- Offscreen context support is disabled when you connect to remote hosts running older versions of **phrelay**.

## Using predefined Photon services

The **-s** command-line option simplifies the task of creating shortcuts to Photon applications within the Windows desktop.

By using the **-s** option, you can create an icon or shortcut on the Windows desktop that starts a Photon application automatically (within a private Phindows session). With proper specification of the remote Window manager options, it's possible to make that Photon application look like it's a native Windows application.

When **phrelay** runs on the QNX host machine, it looks up the Photon service specified with the **-s** option in a configuration file (**/etc/config/phrelay[.***node***]**). If a matching service is found, **phrelay** launches the specified Photon command instead of the default Photon desktop. You can specify optional Window Manager options, but the default mode is to start the remote Photon application such that it looks and behaves as if it were a native Windows application.

You typically use the **-U** option along with the **-s** option to specify a QNX userid to use when running the remote Photon command. If you don't give a userid, and the **phrelay** service doesn't specify a default one, Photon pops up a login dialog requesting the QNX userid before proceeding. By specifying a userid with the **-U** option, you can avoid this login dialog.

For example, if you create a Windows shortcut as follows:

**phindows -t***x.x.x.x* **-svpoker -Ujoe**

where the IP address *x.x.x.x* specifies the TCP/IP address of the Neutrino host, and the **phrelay** configuration file on the host (**/etc/config/phrelay**) contains the following line:

**vpoker % /usr/photon/bin/vpoker**

then **joe** could directly launch a Photon **vpoker** session (running as QNX userid **joe**) on his Windows desktop by simply clicking on the icon or shortcut.

# **phrelay** configuration file format

See **phrelay** for a description of the configuration file format.