QNX® Momentics®
Development Suite

Welcome to QNX Momentics

For Windows®, Linux®, Solaris™, and QNX® Neutrino® hosts

© 2006, QNX Software Systems GmbH & Co. KG.
Contents

About This Guide ix
  Typographical conventions xi
  Note to Windows users xii
  Welcome to QNX Momentics xiii
  What’s in this guide? xiv

1 What is QNX Momentics? 1
  Choice, tools, source, and help 3
  QNX Momentics at a glance 3
  Getting started before you have your target 5
    x86 (PC) target 6
    x86 non-BIOS target 6
    Any supported CPU family 7
  A word about licensing 8
  Migrating from QNX 4 to QNX Neutrino 9

2 Installation and Setup 11
  Supported hosts 13
  Installing and removing QNX Momentics 13
  What do I do if my installation fails? 14
  What did I just install? 14
  What’s on my desktop? 17
  Upgrading your software 18
  Using source code 19
  Running QNX Neutrino self-hosted 19
Mixing a self-hosted machine with other hosts 19
Can different versions of QNX Momentics coexist? 20
Running QNX Neutrino on a target machine 20

3 How to Get Help 23

Overview of the documentation 25
   Integrated Development Environment (Professional Edition only) 25
   QNX Neutrino Realtime Operating System 25
   Photon microGUI 26
   BSPs, DDKs, and specialty tools 27
   Dinkum C and C++ 28

Viewing the documentation 28
   Within the IDE 28
   Outside of the IDE 29
   The Photon Helpviewer 29
   Keyword indexes 31

What should I read first? 31
   During development (on your host) 31
   At runtime (on your target) 33

Where key features are documented 33
Related reading 42
   On QNX Neutrino 42
   On POSIX 42
   On TCP/IP 43

Getting quick help with the use command 43
Hover help in the IDE 44
Other sources of technical help 44
   Visit http://www.qnx.com 44
   Eclipse consortium 45
   Related sites 45
   Try the newsgroups 45

Support plans 46
List of Figures

QNX Momentics is the development environment on your host for the QNX Neutrino RTOS running on your target. xiii
The host-related directory structure. 15
The target-related directory structure. 16
Getting help in the IDE. 29
Photon Helpviewer. 30
Typographical conventions

Throughout this manual, we use certain typographical conventions to distinguish technical terms. In general, the conventions we use conform to those found in IEEE POSIX publications. The following table summarizes our conventions:

<table>
<thead>
<tr>
<th>Reference</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code examples</td>
<td><code>if ( stream == NULL )</code></td>
</tr>
<tr>
<td>Command options</td>
<td><code>-lR</code></td>
</tr>
<tr>
<td>Commands</td>
<td><code>make</code></td>
</tr>
<tr>
<td>Environment variables</td>
<td><code>PATH</code></td>
</tr>
<tr>
<td>File and pathnames</td>
<td><code>/dev/null</code></td>
</tr>
<tr>
<td>Function names</td>
<td><code>exit()</code></td>
</tr>
<tr>
<td>Keyboard chords</td>
<td><code>Ctrl – Alt – Delete</code></td>
</tr>
<tr>
<td>Keyboard input</td>
<td><code>something you type</code></td>
</tr>
<tr>
<td>Keyboard keys</td>
<td><code>Enter</code></td>
</tr>
<tr>
<td>Program output</td>
<td><code>login:</code></td>
</tr>
<tr>
<td>Programming constants</td>
<td><code>NULL</code></td>
</tr>
<tr>
<td>Programming data types</td>
<td><code>unsigned short</code></td>
</tr>
<tr>
<td>Programming literals</td>
<td><code>0xFF, &quot;message string&quot;</code></td>
</tr>
<tr>
<td>Variable names</td>
<td><code>stdin</code></td>
</tr>
<tr>
<td>User-interface components</td>
<td><code>Cancel</code></td>
</tr>
</tbody>
</table>

We format single-step instructions like this:

➤ To reload the current page, press Ctrl – R.

We use an arrow (→) in directions for accessing menu items, like this:
You’ll find the Other... menu item under Perspective→Show View.

We use notes, cautions, and warnings to highlight important messages:

Notes point out something important or useful.

CAUTION: Cautions tell you about commands or procedures that may have unwanted or undesirable side effects.

WARNING: Warnings tell you about commands or procedures that could be dangerous to your files, your hardware, or even yourself.

Note to Windows users

In our documentation, we use a forward slash (/) as a delimiter in all pathnames, including those pointing to Windows files.

We also generally follow POSIX/UNIX filesystem conventions.
Welcome to QNX Momentics

Thank you for choosing the QNX Momentics development suite. You have everything you need to build and maintain your QNX Neutrino-based embedded system: a comprehensive set of integrated development tools, lots of in-depth documentation, as well as powerful diagnostics and optimization tools for your system once it’s up and running on your target.

Board support packages (BSPs), driver development kits (DDKs), and technology development kits (TDKs) are available separately; you can download them from our website, http://www.qnx.com/.

Trusted and proven in countless embedded systems, QNX Neutrino has a growing reputation as the world’s most reliable RTOS. We now invite you to explore the advanced tools that QNX Momentics adds to QNX Neutrino.

QNX Momentics is the development environment on your host for the QNX Neutrino RTOS running on your target.
What’s in this guide?

The Welcome to QNX Momentics guide is intended to introduce you to the QNX Momentics development suite and help you start developing applications for QNX Neutrino. This guide is organized around these main topics:

- What is QNX Momentics?
- How do I install and set up my system?
- How do I get help?

The following table may help you quickly find information in this guide:

<table>
<thead>
<tr>
<th>To:</th>
<th>Go to:</th>
</tr>
</thead>
<tbody>
<tr>
<td>See a list of this product’s components</td>
<td>“QNX Momentics at a glance” in What is QNX Momentics?</td>
</tr>
<tr>
<td>Determine which document to look in for information about key features</td>
<td>“Where key features are documented” in How to Get Help</td>
</tr>
<tr>
<td>Understand the license agreement</td>
<td>“A word about licensing” in What is QNX Momentics?</td>
</tr>
<tr>
<td>Use a virtual machine</td>
<td>“Getting started before you have your target” in What is QNX Momentics?</td>
</tr>
<tr>
<td>Install QNX Momentics on your host machine or remove it</td>
<td>“Installing and removing QNX Momentics” in Installation and Setup</td>
</tr>
<tr>
<td>Find out where components are installed on your host machine</td>
<td>“What did I just install?” in Installation and Setup</td>
</tr>
<tr>
<td>Upgrade your software</td>
<td>“Upgrading your software” in Installation and Setup</td>
</tr>
</tbody>
</table>

continued...
### What's in this guide?

<table>
<thead>
<tr>
<th>To:</th>
<th>Go to:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run the QNX Neutrino RTOS</td>
<td>“Running QNX Neutrino on a target machine” in Installation and Setup</td>
</tr>
<tr>
<td>on a target machine</td>
<td></td>
</tr>
<tr>
<td>Get started with the</td>
<td>How to Get Help</td>
</tr>
<tr>
<td>documentation</td>
<td></td>
</tr>
<tr>
<td>Get technical support</td>
<td>How to Get Help</td>
</tr>
<tr>
<td>Find the meaning of terms used in</td>
<td>Glossary</td>
</tr>
<tr>
<td>QNX documentation</td>
<td></td>
</tr>
</tbody>
</table>

If you’ve already developed applications with QNX Momentics 6.2.1, see the 6.3.0 and 6.2.1 Compatibility notes. You can find the latest version of this document on our website: [http://www.qnx.com/](http://www.qnx.com/). Log into your myQNX account, then go to the Download Center.
Chapter 1

What is QNX Momentics?

In this chapter...

Choice, tools, source, and help 3
QNX Momentics at a glance 3
Getting started before you have your target 5
A word about licensing 8
Migrating from QNX 4 to QNX Neutrino 9
Choice, tools, source, and help

As a complete package designed for embedded systems developers, the QNX Momentics development suite gives you everything you need at every stage of your product-development cycle:

Choice    Choice of host (Windows, Linux, Solaris, QNX Neutrino), choice of target (ARM, MIPS, PowerPC, SH-4, XScale, x86), and choice of development language (C, C++, Embedded C++, Java).

Tools     Code development, editors, source control, compilers, libraries, profilers, analyzers, optimizers, etc.

Help      Documentation, newsgroups, technical support programs, etc.

Full source code for numerous startup programs, IPLs, device drivers, etc. is available from our website.

QNX Momentics at a glance

There are two versions of QNX Momentics: the Standard Edition (SE) and the Professional Edition (PE). PE includes everything that’s in SE, as well as our Integrated Development Environment. For Windows hosts, PE also includes the Rational ROSE Adaptation Layer as well as Phindows (“Photon in Windows”).

Here are the main parts of your QNX Momentics suite:

QNX Neutrino RTOS

The whole point of it all. If Neutrino is the “engine” that will empower the embedded system you’re developing, then QNX Momentics is the “factory” where you modify your engine as well as build, test, and finish your vehicles.
Integrated Development Environment

If you have PE, this is your toolbox. The IDE’s task-oriented interface helps you quickly set up your project, choose your programming language, choose a target processor, compile your code, connect to your target, transfer your application to your target, run it, debug it, profile it, and fine-tune it.

Command-line tools

If you aren’t using the IDE, you can use command-line tools to develop applications. For example, you can use qcc to compile and link, and mkifs to create an OS image.

Libraries

ANSI C, POSIX, Dinkum C++ (full and embedded), GNU C++ (x86 only), graphics, widgets, compression, etc.

Documentation

How-to guides, references, context-sensitive help, and technotes. See the chapter How to Get Help to help you find your way through the documentation.

If you’ve installed the Standard or Professional Edition, you can download these components from our website after logging into your myQNX account:

Binary Board Support Packages

Binaries and step-by-step instructions to help you get Neutrino and your applications running on specific evaluation boards.

Driver Development Kits

Full source and detailed documentation to help you write your own drivers for various devices: audio, graphics, input (mice, keyboards, etc.), network, and USB.

Note that the QNX Momentics development suite includes the DDK documentation.
These components are available separately:

**Source Board Support Packages**
Source code for the BSPs.

**Technology Development Kits**
Kits that augment the base Neutrino OS platform in QNX Momentics 6.3.0 with specialized, value-added technologies. This packaging helps you control costs for systems that you build with QNX Momentics.

The TDKs currently include:
- Adaptive Partitioning
- Advanced Graphics (formerly 3D Graphics)
- Critical Process Monitoring
- Extended Networking
- Flash Filesystem & Embedding
- Multimedia
- Multicore (formerly Symmetric Multiprocessing)
- Web Browser

For more information about these components, contact your sales representative.

### Getting started *before* you have your target

Assuming you plan to develop a Neutrino-based embedded system of some kind (e.g. a vehicle telematics system, a router, a medical imaging device), you can start developing your application, even before you have your target hardware.

And if you haven’t yet decided on the CPU family for your target (e.g. PowerPC, XScale), you can still begin developing your application. As a rule, with QNX Momentics you write your application code once, then compile it however many times you need for whatever

---

© 2006, QNX Software Systems GmbH & Co. KG.   Getting started *before* you have your target

November 3, 2006   Chapter 1 • What is QNX Momentics? 5
Getting started before you have your target

supported targets you’re using. As you’ll see if you plan to use the IDE, you can build the very same project for any of our supported processors with just a couple of mouse clicks.

If you’re developing on a self-hosted Neutrino system, you can run your application directly on your development machine.

Consider the following scenarios:

x86 (PC) target

If your final target hardware will be a PC in some shape or form (e.g. a PC/104 SBC module), you can simply connect your host to any spare PC and use that as your temporary target. You should expect to make very few changes (if any) in your application when the time comes to move to your final target.

And if you don’t happen to have a spare PC, you could use a virtual machine — products such as Workstation by VMware (http://www.vmware.com) can emulate a separate, complete hardware environment right on your desktop. You can get a free, time-limited evaluation of this product from the vendor.

Once you’ve set up a virtual machine (VM) on your host, you can then use it as your target — you can run, test, and debug your Neutrino applications on the VM as if it were an actual machine connected to your host.

Virtual machines probably don’t support hard realtime.

x86 non-BIOS target

In this case, the target computer (e.g. an SC400) isn’t a PC. It doesn’t have a BIOS, so it won’t boot and load Neutrino in the same way as a PC. In order to run Neutrino on such a target, you need an appropriate BSP for your particular hardware, including an IPL that could set up the processor and load Neutrino properly.
Getting started before you have your target

We’re continually adding new BSPs to our extensive selection; for a list, see the Products area of our website, http://www.qnx.com. If you don’t see a BSP for your target hardware, contact us.

You can develop your application without target hardware, because much of your code will be hardware-independent. For device drivers and other hardware-specific areas of your system, you may be able to use a virtual platform tool that accurately emulates your specific embedded system hardware.

For instance, Virtio (http://www.virtio.com) makes virtual platforms that emulate various processors and peripherals, right down to the CPU’s instruction set. You can download a free time-limited eval copy from the vendor’s website.

Any supported CPU family

Whatever your target hardware, you can still go a long way down your development path using QNX Momentics right now. Develop all your application-level components first, leaving device drivers and other hardware-specific details until you have your target hardware.

As mentioned earlier, you might want to consider Virtio if you need a functionally accurate virtual prototype of your embedded platform before you have the actual hardware itself.

Platform-related issues

Whatever the specific CPU family, your application code will, in most cases, run unchanged when you go to your final target. But you may need to be aware of a few platform-related issues (e.g. endian differences, CPU speeds, I/O addressing, alignment of data structures, synchronization on SMP vs uniprocessor systems, etc.).

Fortunately, QNX Momentics includes several header files and convenience functions that will help you anticipate and easily handle such problems.
A word about licensing

For more information on these and other platform-related issues, see Freedom from Hardware and Platform Dependencies in the Neutrino Programmer's Guide.

A word about licensing

Here are some general questions about licensing. For more information, visit the Legal area of http://www.qnx.com, where you’ll find details on our licensing model, types of licenses, etc., or email licensing@qnx.com.

Where can I find my EULA?

You’ll find your Momentics End User License Agreement on your CD and also in the Legal area of our website, http://www.qnx.com

If you have a source license, you’ll also find your agreement on our website.

I’ve installed QNX Momentics on my workstation. Can I also make a copy for my laptop?

Certainly. Our licensing policy is per seat, not per machine. You may use QNX Momentics on your primary workstation as well as on a laptop or on a PC at your home, provided that you use only one system at any given time.

What are floating licenses?

These allow a group of developers to share a small set of licenses; a fixed number of developers can alternately use any given license. Note that only one developer may use a license at any given time.

As an OEM, how do I get a runtime license?

To obtain a runtime license (which governs how you may distribute certain Neutrino runtime files as part of your product), please contact your QNX sales representative.
Can I get a Neutrino runtime system?

You can purchase a Neutrino Runtime CD, which includes runtime binaries and data, but no development tools or source; for more information, please contact your QNX sales representative.

Migrating from QNX 4 to QNX Neutrino

If you’re migrating from QNX 4 to Neutrino, you should download the migration kit from the Download Center area in our website, http://www.qnx.com, after logging into your myQNX account. This free kit includes:

- a QNX 4 archive that contains a source-analysis tool to help you identify trouble spots in your code
- a Neutrino package that contains documentation and a migration library to bridge many of the differences between QNX 4 and Neutrino
Chapter 2
Installation and Setup

In this chapter . . .

Supported hosts 13
Installing and removing QNX Momentics 13
What do I do if my installation fails? 14
What did I just install? 14
What’s on my desktop? 17
Upgrading your software 18
Using source code 19
Running QNX Neutrino self-hosted 19
Mixing a self-hosted machine with other hosts 19
Can different versions of QNX Momentics coexist? 20
Running QNX Neutrino on a target machine 20
Supported hosts

To install and use QNX Momentics, you’ll need a desktop machine running one of the following for your host development environment:

- Windows NT, 2000, or XP
- Linux Red Hat 8, 9, or Enterprise WS 3 or 4
- Solaris 7, 8, or 9
- QNX Neutrino 6.3

We’re deprecating support for the following:

- Linux Red Hat 8 and 9
- Solaris 7
- Windows NT

QNX Momentics 6.3.0 is the last release that you can install on these hosts.

For particular system requirements (e.g. amount of disk space you’ll need to install), see the installation note that came with your software.

Installing and removing QNX Momentics

QNX Momentics uses InstallShield on all hosts except Neutrino (which uses tarballs and shell scripts) to help you install and remove software. For specific instructions, see the Installation Note that was shipped with your software.

If you want to develop in Neutrino (self-hosted), you must install the Neutrino runtime system, then the QNX Momentics development suite. They’re both on the same CD. For other hosts (e.g. Windows or Linux), you simply need to install QNX Momentics.
What do I do if my installation fails?

If for some reason (e.g. hardware incompatibility) your QNX Momentics system doesn’t install properly, you should:

1. Review the requirements and instructions given in the *Installation Note* that was shipped with your software.

2. If you can access the Internet, check the lists of supported hardware posted on our website ([http://www.qnx.com](http://www.qnx.com)).

3. Check the troubleshooting sections in the Neutrino *User’s Guide*.

4. Contact us for help, or post questions in one of our newsgroups. For more information, see the How to Get Help chapter.

What did I just install?

QNX Momentics software is organized around these two main areas:

Host-related
All your libraries, executables, etc., designed to run on your host system (e.g. Windows).

Target-related
All CPU-specific components, as well as certain common things (e.g. the documentation) that you can use on any target system.

The **QNX_HOST** environment variable identifies the directory that holds the host-related components:
What did I just install?

You choose the path 

$QNX\_HOST

e tc

qcc system ntoarm, ntoppc, ntosh, ntox86

usr

bin lib qde share

The host-specific development tools for target platforms

The host-related directory structure.

The **QNX\_TARGET** environment variable identifies the directory that holds the target-related components:
Neutrino also uses these environment variables to locate files on the host machine:

**QNX_CONFIGURATION**

The location of the configuration files and licenses for QNX Momentics.

**MAKEFLAGS**

The location of included *.mk files.

Here’s where some of the key components are installed:

<table>
<thead>
<tr>
<th>Component</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buildfiles</td>
<td>$QNX_TARGET/platform/boot/build/brandname.build</td>
</tr>
</tbody>
</table>

*continued...*
<table>
<thead>
<tr>
<th>Component</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command-line utilities</td>
<td>For the host: $\text{QNX}<em>{\text{HOST}}/\text{usr/bin}$ and $\text{QNX}</em>{\text{HOST}}/\text{platform/bin}$</td>
</tr>
<tr>
<td></td>
<td>For the target: $\text{QNX}<em>{\text{TARGET}}/\text{bin}$ and $\text{QNX}</em>{\text{TARGET}}/\text{platform/bin}$ and $\text{QNX}_{\text{TARGET}}/\text{platform/sbin}$</td>
</tr>
<tr>
<td>Device drivers (binaries)</td>
<td>$\text{QNX}_{\text{TARGET}}/\text{platform/sbin}$</td>
</tr>
<tr>
<td>Device drivers (DLLs)</td>
<td>$\text{QNX}_{\text{TARGET}}/\text{platform/lib/dll}$</td>
</tr>
<tr>
<td>Filesystems</td>
<td>$\text{QNX}_{\text{TARGET}}/\text{platform/sbin}$</td>
</tr>
<tr>
<td>GUI-related</td>
<td>$\text{QNX}_{\text{TARGET}}/\text{usr/photon}$</td>
</tr>
<tr>
<td>Shared libraries</td>
<td>$\text{QNX}_{\text{TARGET}}/\text{platform/lib}$</td>
</tr>
<tr>
<td>System header files</td>
<td>$\text{QNX}_{\text{TARGET}}/\text{usr/include}$</td>
</tr>
<tr>
<td>Documentation</td>
<td>$\text{QNX}_{\text{TARGET}}/\text{usr/help/product}$</td>
</tr>
</tbody>
</table>

If you install any BSPs or DDKs, they’re installed under $\text{QNX}_{\text{TARGET}}$. The IDE (in the QNX Momentics Professional Edition) creates a workspace directory in your home directory (Linux, Solaris, and Neutrino) or in \texttt{C:\QNX630} (Windows).

For information about the directory structure on a Neutrino runtime system, see “Where everything is stored” in the Working with Files chapter of the Neutrino User’s Guide.

**What’s on my desktop?**

How you access the components of QNX Momentics depends on your host machine:

**Windows** You can start the IDE by clicking its icon on the desktop:
or by choosing Start→QNX Momentics IDE. The Start→All Programs→QNX Momentics 6.3.0 menu lets you add or activate licenses, configure your machine to build for a specific version of QNX Momentics, view the documentation, run Phindows, and start the IDE.

Linux The Start→Programming menu lets you add or activate licenses, view the documentation, and start the IDE.

Solaris Use a terminal window. For example, to start the IDE, run qde; to view the documentation, point a browser at $QNX_TARGET/usr/help/product/momentics/index.html.

Neutrino The Launch→QNX Momentics 6.3.0 menu lets you start the IDE; the Launch→Configure menu lets you add and activate licenses. The Help item in the Launch menu and on the shelf starts the Helpviewer, where you’ll find the documentation.

Upgrading your software

Here’s the general procedure for updating a version of QNX Momentics or other components you’ve purchased:

1  Go to the QNX Software Systems website (http://www.qnx.com) and log into your myQNX account. If you don’t already have a myQNX account, please register now.

2  Follow the instructions for registering your product. You’ll need the Product Registration serial number and password from your License Certificate.

3  Go to the Download Center and click Products & Updates.

4  Select the type of product (e.g. “QNX Momentics”).

5  Select the appropriate version of the product you have (e.g. “QNX Momentics 6.3.x”).
In the next step, you’ll download a file. Don’t download it into a directory whose path contains spaces. For example, don’t download the file into 
C:\Documents and Settings\my_userid\Desktop.

6 Download the appropriate file and follow the instructions.

If you installed an evaluation copy of QNX Momentics, and you now want to upgrade to a permanent copy, contact your sales representative. When you’ve received a commercial QNX License Certificate, you can add a commercial license without reinstalling. For more information, see the installation notes.

Using source code

You’ll probably want to use some sort of version-control system to manage and track changes to the software that you develop. For an introduction to CVS (Concurrent Versions System), see Using CVS in the Neutrino User’s Guide, and Managing Source Code in the IDE User’s Guide.

Running QNX Neutrino self-hosted

You can develop software on a self-hosted QNX Neutrino system. For more information on working with Neutrino, see the Neutrino User’s Guide; for information on developing software on Neutrino, see the Neutrino Programmer’s Guide and the IDE User’s Guide.

Mixing a self-hosted machine with other hosts

If you have a Neutrino host, you can communicate with other hosts in various ways:

- You can access resources — such as files, directories, and processes — on other Neutrino machines as if the resources were
Can different versions of QNX Momentics coexist?

You can install more than one version of QNX Momentics on your machine. You can use the IDE or the `qconfig` utility to set up your environment to build for the correct version. On Windows hosts, use `QWinCfg`, a graphical front end for `qconfig` that you can launch from the Start menu.

Coexistence of 6.3.0 and 6.2.1 is supported only on Windows and Solaris hosts.

For more information, see the Compiling and Debugging chapter of the Neutrino Programmer’s Guide and the IDE Concepts chapter of the IDE User’s Guide.

Running QNX Neutrino on a target machine

Neutrino is well suited to embedded systems. For information about creating OS images, downloading them to your target hardware, and running your software, see:

- the BSP documentation for your particular target (see the Start or Launch menu on your host)
- Building Embedded Systems
- IDE User’s Guide
You don’t always need to have the hardware to run your software; for more information, see “Getting started before you have your target” in the What is QNX Momentics? chapter in this guide.
In this chapter...

Overview of the documentation 25
Viewing the documentation 28
What should I read first? 31
Where key features are documented 33
Related reading 42
Getting quick help with the use command 43
Hover help in the IDE 44
Other sources of technical help 44
Support plans 46
Training 46
Custom engineering and consulting 46
Overview of the documentation

In QNX Momentics, the online documents are in HTML, which you can access either from within or outside of the IDE.

You can download PDF versions of the documentation from our website (http://www.qnx.com), once you’ve logged into your myQNX account. Printed books are also available.

To help you find your way around the QNX Momentics documentation set, we’ve provided a documentation roadmap.

While the roadmap works best online because of all its links, here’s a hardcopy version that can still help you know where to look for information.

The complete QNX Momentics documentation set contains the following books, arranged here under each main component:

Integrated Development Environment (Professional Edition only)

IDE User’s Guide

Describes the QNX Momentics Integrated Development Environment, how to set up and start using the tools to build Neutrino-based target systems, etc.

QNX Neutrino Realtime Operating System

System Architecture

Describes the concepts and architecture of the QNX Neutrino microkernel, resource managers, processes, threads, message-passing services, and more.

QNX Neutrino User’s Guide

Explains how to interact with a running Neutrino system. Covers both Photon and text-mode interfaces, as well as various system-administration topics.
Overview of the documentation

QNX Neutrino Programmer’s Guide
Tells you how to get started writing programs, including interrupt handlers, resource managers, etc.

Building Embedded Systems
Tells you how to get the OS running on your target embedded system, write an IPL, customize a startup program, etc.

Utilities Reference
Describes the Neutrino configuration files, utilities, and manager processes you’ll use during development and at runtime. See the Utilities Summary chapter in the reference for a listing of all the entries by category.

QNX Neutrino Library Reference
Describes the C library data types and functions, including POSIX threads, kernel calls, resource manager functions, etc. See the Summary of Functions chapter in the reference for a listing of all the functions by category.

Technical Notes
Deals with a series of topics (e.g. IP tunneling) that aren’t covered in the basic documentation set.

Photon microGUI

Photon Programmer’s Guide
Gives you a hands-on tour of the Photon Application Builder (PhAB). You’ll learn how to quickly assemble a GUI from predefined widgets, link the GUI to an application, and generate C source to bring the GUI to life.

Multimedia Developer’s Guide
Describes the multimedia architecture for developers who wish to write multimedia plugins.

Photon Library Reference
Provides concise descriptions of Photon’s and PhAB’s global data structures and functions.
Overview of the documentation

Widget Reference

Contains guidelines for programming widgets, along with concise descriptions of all global data structures, resources, and convenience functions associated with widgets. It also gives you practical examples of how to use Photon widgets and widget functions.

Building Custom Widgets

Explains how to create a custom widget and how to bind it into PhAB. If you need a widget whose features extend the standard capabilities of the Photon widget library, this guide is for you.

BSPs, DDKs, and specialty tools

BSP guides Describe how to get Neutrino running on your target board. You’ll find a separate guide for each BSP that you’ve installed. We support boards in these processor families: ARM/XScale, MIPS, PowerPC, SH-4, and x86. You can download binaries for the BSPs from our website, http://www.qnx.com/; for information about BSP source, contact your sales representative.

DDK guides Describe how to write drivers for QNX Neutrino. You’ll find a separate DDK guide for audio, character, graphics, input, network, and Universal Serial Bus (USB) devices. You can download the DDKs from our website, but QNX Momentics includes the documentation.

System Analysis Toolkit User’s Guide

Describes how to use the SAT with our instrumented microkernel. You can log every communication and state change within the microkernel, including interrupts, all parameters/return values from kernel calls, and scheduling decisions, resulting in a deeper and more detailed analysis of system elements. You can even perform kernel-level diagnostics remotely.
Photon Multilingual Input

Tells you how to input Chinese, Japanese, and Korean characters in Photon.

Phindows Connectivity

Tells you how to access Photon from a Windows machine.

TDKs

The release notes (see ${QNX\_TARGET}/etc/readme/tdk) for each TDK indicate the relevant documentation.

**Dinkum C and C++**

*Dinkum C++ Library*

A conforming implementation of the Standard C++ library.

*Dinkum C99 Library*

A conforming implementation of the Standard C library, as revised in 1999.

*Dinkum EC++ Library*

A conforming implementation of the Embedded C++ library as specified by the Embedded C++ Technical Committee.

**Viewing the documentation**

**Within the IDE**

Click **Help→Help Contents**. There you’ll find several booksets listed, with *QNX Momentics Professional Edition* at the top of the list. The other documents listed, such as the *Workbench User Guide* and *JDT Plug-in Developer Guide*, pertain to the Eclipse platform and its various plugins.
Getting help in the IDE.

Note that the IDE’s internal help system has a built-in search facility. For details, see “Help system” in the IDE Concepts chapter of the IDE User’s Guide.

Outside of the IDE

Point your browser at:

$QNX\_TARGET/usr/help/product/momentics/bookset.html

You can also reach the HTML documentation from the Start or Launch menu, depending on the host:

Windows

Programs→QNX Momentics 6.3.0→Documentation

Linux

Programming→QNX Momentics 6.3.0 Documentation

Neutrino

Help or QNX Momentics 6.3.0→Documentation

The Photon Helpviewer

If you’re using Neutrino self-hosted, you can access the documentation via our native Helpviewer.
Photon Helpviewer.

To open the Helpviewer, click the **Help** button in the Applications group on the shelf or select **Help** from the right-click menu on the desktop:
What should I read first?

For more information, see “Getting help with the Helpviewer” in the Using the Photon microGUI chapter of the Neutrino User’s Guide, as well as the entry for helpviewer in the Utilities Reference.

Keyword indexes

Nearly every book in the QNX Momentics documentation set has its own keyword index. At the top and bottom of the online documents, you’ll find an icon that’s linked to the keyword index file (keywords-all.html):

Index

What should I read first?

Many people simply don’t read manuals cover to cover. They often browse or search the documentation for a specific topic, usually for information on how to do a certain task. But if you want to approach your tasks with enough knowledge to work effectively, it’s a good idea to start with the System Architecture guide; it will help you understand Neutrino’s unique features, particularly its message-based interprocess communication (IPC) and microkernel architecture.

Once you know how Neutrino works, you’ll then want to know how to work with it. Which document you’ll need next depends on when you need it.

During development (on your host)

Most of the documents in the bookset are geared towards developing your Neutrino-based applications. Of these development books, some are how-to guides, some are reference works.
Some books are useful during development as well as at runtime. For instance, the *Utilities Reference* is a comprehensive document that includes descriptions of both “development” utilities (e.g. `make`), which you wouldn’t normally use on your target, as well as “runtime” programs (e.g. `devc-sersci`), which you would run only on your target.

Here are the main how-to guides and their corresponding reference books:

<table>
<thead>
<tr>
<th>How-to guide:</th>
<th>Companion reference:</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDE <em>User’s Guide</em></td>
<td><em>Utilities Reference</em> (for underlying command-line utilities, such as <em>make</em>); QNX Neutrino <em>Library Reference</em> for API</td>
</tr>
<tr>
<td>QNX Neutrino <em>User’s Guide</em></td>
<td><em>Utilities Reference</em></td>
</tr>
<tr>
<td>QNX Neutrino <em>Programmer’s Guide</em></td>
<td>QNX Neutrino <em>Library Reference</em></td>
</tr>
<tr>
<td>Photon <em>Programmer’s Guide</em></td>
<td>Photon <em>Library Reference</em>; Photon <em>Widget Reference</em></td>
</tr>
<tr>
<td><em>Building Embedded Systems</em></td>
<td><em>Utilities Reference</em>; QNX Neutrino <em>Library Reference</em></td>
</tr>
</tbody>
</table>

If you’ve already developed applications with QNX Momentics 6.2.1, see the 6.3.0 and 6.2.1 *Compatibility* notes. You can find the latest version of this document on our website: [http://www.qnx.com/](http://www.qnx.com/). Log into your *myQNX* account, then go to the Download Center.
At runtime (on your target)

The most runtime-oriented document in your bookset is the QNX Neutrino User’s Guide, which describes how to use and interact with a running Neutrino system. The book covers both Photon and text-mode interfaces, as well as various system-administration topics.

Other runtime-oriented documents include the Utilities Reference and the IDE User’s Guide (for information on diagnostic tools you’d use on a running system).

Where key features are documented

The following list may help you learn which document to look in when you want information on certain key features or components of the OS and tools.

See the System Architecture guide for information on almost every topic in this list.

Adaptive partitioning
   Adaptive Partitioning TDK User’s Guide

Asymmetric multiprocessing (AMP)
   Multicore Processing in the System Architecture guide

Audio support
   Audio Developer’s Guide (for writing audio applications); Audio DDK (for writing audio device drivers); io-audio manager and the deva-* drivers in the Utilities Reference.

Bound multiprocessing (BMP)
   Multicore Processing in the System Architecture guide; the Multicore TDK User’s Guide; procnto* in the Utilities Reference.

BSPs
   Each BSP that you install includes documentation; see the BSP installation notes.
Where key features are documented

Buildfiles

Making an OS Image in Building Embedded Systems; mkifs in the Utilities Reference.

CD-ROM

Working with Filesystems in the QNX Neutrino User’s Guide; cam-cdrom.so in the Utilities Reference.

Channels

Channel*, Msg*, and Connect* functions in the QNX Neutrino Library Reference; Writing a Resource Manager in the QNX Neutrino Programmer’s Guide.

Compiling

Developing C/C++ Programs in the IDE User’s Guide; Compiling and Debugging in the QNX Neutrino Programmer’s Guide; make and qcc in the Utilities Reference.

Compression

Making an OS Image in Building Embedded Systems; deflate in the Utilities Reference.

CPU time, sharing among competing processes

Adaptive Partitioning TDK User’s Guide

Data server


DDKs

Driver Development Kits (DDKs). QNX Momentics includes the documentation, but if you want the DDKs, you must download them from our website.

Debugger

Debugging Programs in the IDE User’s Guide; Compiling and Debugging in the QNX Neutrino Programmer’s Guide; gdb and other debugging utilities in the Utilities Reference.

Device management

devctl() in the QNX Neutrino Library Reference; Writing a Resource Manager in the QNX Neutrino
Where key features are documented

Programmer’s Guide; dev* drivers in the Utilities Reference.

Editors

Embedded systems
Building Embedded Systems.

Endian issues
Freedom from Hardware and Platform Dependencies in the QNX Neutrino Programmer’s Guide.

Environment, configuring

Environment variables
Commonly Used Environment Variables appendix in the Utilities Reference; environ() in the QNX Neutrino Library Reference; Using the Photon microGUI and Configuring Your Environment chapters in the QNX Neutrino User’s Guide.

Event handling (OS)
MsgDeliverEvent() in the QNX Neutrino Library Reference; Writing an Interrupt Handler in the QNX Neutrino Programmer’s Guide.

Event handling (Photon)

File handling
creat() and related functions via its “See also” section in the QNX Neutrino Library Reference; basename, cat, etc. in the Utilities Reference
Where key features are documented

(see “File and directory manipulation” in the Utilities Summary).

Filesystems    Working with Filesystems in the QNX Neutrino User’s Guide.


Graphics       Graphics DDK; devg-* drivers in the Utilities Reference.

Hardware       Connecting Hardware in the QNX Neutrino User’s Guide.

HID (human input devices)

hidview and devh-* drivers in the Utilities Reference.

Helpviewer (Photon)

Using the Photon microGUI in the QNX Neutrino User’s Guide; helpviewer in the Utilities Reference.

I/O management

iofunc*, Interrupt*, and open() and related functions via open()’s “See also” section in the QNX Neutrino Library Reference; Writing a Resource Manager in the QNX Neutrino Programmer’s Guide.

IPC (interprocess communication)

pthread_mutex*, SyncMutex*, sem.*, SyncSem*, pthread_cond*, and SyncCondvar* in the QNX Neutrino Library Reference. See also “Message Passing,” below.

IPL (Initial Program Loader)

Writing an IPL Program in Building Embedded Systems.
Images (OS) Making an OS Image in *Building Embedded Systems*; **mkifs** in the *Utilities Reference*.


Input *Input DDK*; **devh-** and **devi-** drivers in the *Utilities Reference*.

Instrumented kernel System Analysis Toolkit *User’s Guide*.

Interrupt handling

*InterruptAttach()* and related functions in its “See also” section in the QNX Neutrino *Library Reference*.

Kernel calls *Channel*-, *Clock*-, *Connect*-, *Debug*-, *Interrupt*-, *Msg*-, *Sched*-, *Signal*-, *Sync*-, *Thread*-, *Timer*-, and *TraceEvent()* in the QNX Neutrino *Library Reference*.

Keyboard support Using the Command Line in the QNX Neutrino *User’s Guide*; **mkkbd** and related utilities in its “See also” section in the *Utilities Reference*.


Linker Compiling and Debugging in the QNX Neutrino *Programmer’s Guide*; **qcc** and related utilities in its “See also” section in the *Utilities Reference*.

**Makefile** structure Developing C/C++ Programs in the IDE *User’s Guide*; Conventions for Makefiles and Directories in the QNX Neutrino *Programmer’s Guide*.  

November 3, 2006
Where key features are documented

Memory management


Message passing  *Msg*, *Connect*, and *Channel* in QNX Neutrino Library Reference. See also “IPC,” above.

Message queues  *mq_* functions in the QNX Neutrino Library Reference; *mqueue* and *mq* in the Utilities Reference.

Multicore systems

Multicore Processing in the System Architecture guide; the Multicore TDK User’s Guide; *procnto* in the Utilities Reference.

Native networking (Qnet)

See “Transparent distributed processing,” below.

Network drivers  *Network DDK*; *devn-* drivers in the Utilities Reference.

Package filesystem

Working with Filesystems in the QNX Neutrino User’s Guide; *fs-pkg* in the Utilities Reference.

Permissions (on files, directories)

Managing User Accounts and Working with Files chapters in the QNX Neutrino User’s Guide; *chmod*(), *stat*(), and *umask*() in the QNX Neutrino Library Reference; *chmod* in the Utilities Reference.
Power management


Process manager procnto* in Utilities Reference; procmgr_* in the QNX Neutrino Library Reference.


Profiler (application)

Proﬁling an Application in the IDE User’s Guide.

Profiler (system)

Analyzing Your System with Kernel Tracing in the IDE User’s Guide.

Pulses

MsgSendPulse(), MsgReceivePulse(), and pulse_* in the QNX Neutrino Library Reference; “Handling private messages and pulses” in the chapter on writing resource managers in the QNX Neutrino Programmer’s Guide.

QoS (Quality of Service)

“Quality of Service and multiple paths” in the Native Networking (Qnet) chapter in System Architecture; netmgr ndtostr() in the QNX Neutrino Library Reference.

RAM disk

io-blk.so (using ramdisk=size option), devb-ram, and devf-ram in the Utilities Reference.

Realtime scheduling

Sched* functions and sched_param structure in the QNX Neutrino Library Reference.
Where key features are documented

Resource management

Writing a Resource Manager in the QNX Neutrino Programmer’s Guide; resmgr_* and iofunc_* in the QNX Neutrino Library Reference.

Resources, sharing among competing processes

Adaptive Partitioning TDK User’s Guide

Symmetric multiprocessing (SMP)

Multicore Processing in the System Architecture guide; the Multicore TDK User’s Guide; procnto* in the Utilities Reference.

Self-hosted development

Compiling and Debugging in the QNX Neutrino Programmer’s Guide.

Signal handling

sigaction() and related functions via its “See also” section in the QNX Neutrino Library Reference.

Startup programs

Customizing Image Startup Programs in Building Embedded Systems; startup-* in the Utilities Reference.

Synchronization

Sync* functions in the QNX Neutrino Library Reference.

System information

pidin and sin in the Utilities Reference; Analyzing Your System with Kernel Tracing in the IDE User’s Guide; Fine-Tuning Your System in the QNX Neutrino User’s Guide.

Target agent (qconn)

qconn in the Utilities Reference; IDE Concepts in the IDE User’s Guide.
Where key features are documented

Threads
- `pthread_*` and `Thread_*` in the QNX Neutrino Library Reference; Programming Overview in the QNX Neutrino Programmer’s Guide.

Time
- Configuring Your Environment in the QNX Neutrino User’s Guide; Clock*, `time()`, and related functions via their “See also” sections in the QNX Neutrino Library Reference.

Timestamps (for files)
- `touch` in the Utilities Reference; `utime()` and related functions via its “See also” section in the QNX Neutrino Library Reference.

Timers
- `alarm()`, Timer*, and `timer_*` in the QNX Neutrino Library Reference.

Transparent distributed processing

USB
- Connecting Hardware in the QNX Neutrino User’s Guide; `devu-*`, `devh-usb.so`, `devi-hid`, and `io-usb` in the Utilities Reference; USB DDK.

Unicode

Version control
Related reading

Web server (**slinger**)

Setting Up an Embedded Web Server in the QNX Neutrino User’s Guide; **slinger** in the Utilities Reference.

XIP (execute in place)

Building OS and Flash Images in the IDE User’s Guide; Writing an IPL Program in Building Embedded Systems.

Related reading

**On QNX Neutrino**


  You can get this book from QNX Software Systems; the online documentation for QNX Momentics includes some sample chapters.


  You can get this book from the publisher; for more information, see [http://www.parse.com](http://www.parse.com).

**On POSIX**

The latest POSIX standards documents are available online here:

For an up-to-date status of the many POSIX drafts/standards documents, see the PASC (Portable Applications Standards Committee of the IEEE Computer Society) report at http://pasc.opengroup.org/standing/sd11.html.

In addition to the POSIX standards themselves, you might find the following books useful:


On TCP/IP


Some of the advanced API features mentioned in these TCP/IP books might not be supported.

Getting quick help with the **use** command

Similar to the UNIX **man** command, Neutrino presents a simple usage message for each command-line utility. At the OS system prompt, type:
use utility_name

and you’ll see a brief description as well as the command-line options for that utility.

Hover help in the IDE

When working on your projects in the IDE’s editor, whenever you hover your mouse pointer over a function name in a line of code, you’ll see a popup text box containing that function’s purpose and synopsis (provided that it’s a documented Neutrino function).

For details, see the chapter on C/C++ development in the IDE User’s Guide.

Other sources of technical help

The first place to look for help is in the QNX Momentics documentation, but if you still have problems, there are several other avenues of help:

- http://www.qnx.com
- Eclipse consortium
- related sites
- newsgroups

Visit http://www.qnx.com

The QNX Software Systems website (http://www.qnx.com) offers help on using QNX Momentics through such facilities as:

- detailed lists of supported hardware
- Developer Support Center, which has many technical articles and other resources
- Legal area, which gives details on product licensing
• Partners area, which includes lists of distributors and resellers around the world
• and more

If you log into your myQNX account, you can register your products and support plans and gain access to protected areas of the website. The Developer Support Center and Download Center areas have additional links and content. You might also be given access to public and private newsgroups; see “Try the newsgroups,” below.

**Eclipse consortium**

For general help with Eclipse, the open platform for our IDE, visit the Eclipse consortium website (http://www.eclipse.org). There you’ll find valuable support in the form of newsgroups, mailing lists, articles, and more.

**Related sites**

Neutrino users form an active community worldwide. Listed below (in alphabetical order) are just a few of the community websites devoted to the OS:

• http://qnx.org.ru
• http://www.openqnx.com
• http://www.osnews.com
• http://www.qnxzone.com

For the current list of community resources, see the Developer Support Center area of our website, http://www.qnx.com.

**Try the newsgroups**

The public support forums (qdn.public.* and comp.os.qnx) are a very useful way to get help. Several newsgroups are available from our news server:

Members of the Neutrino user community at large as well as our own developers and staff frequent these newsgroups daily.

Your support plan might also give you access to private newsgroups via news://nntp.qnx.com.

Support plans

You can access a wide range of support resources, depending on the particular support plan that you purchase. We offer two main levels of support:

Standard Includes Help Center access (phone, fax, or email), online account access, product updates, and bug reporting/tracking.

Priority Comes in three varieties: Bronze, Silver, and Gold. In addition to the Standard services, Priority services range from private email addresses and newsgroups to training and even design review.

For more information about our technical support offerings, including email addresses and telephone numbers, please see the Services section of our website (http://www.qnx.com).

Training

QNX training services offers many hands-on courses at your choice of location (QNX headquarters, various training centers around the world, or your site). For details, including the current training schedule, see the Training area in the Services section of our website (http://www.qnx.com).

Custom engineering and consulting

Depending on the nature of your particular project, you may choose to engage us to help in areas such as:
- custom BSPs
- device driver development
- hardware troubleshooting and integration
- application development
- migration/porting services

For more information, contact your local sales representative.
BSP

Board Support Package — a set of software components (IPL, startup code, drivers, buildfiles, as well as the QNX Neutrino RTOS) and documentation intended to help you get Neutrino up and running on a particular evaluation board.

buildfile

A “control” file that specifies the particular startup program, environment variables, drivers, host-target communications, etc. that will be used to generate an OS image. The file provides instructions to a utility program such as mkifs, which generates OS images. In the context of the IDE, the buildfile is sometimes called a target file.

Development License

Development using QNX Software Systems products is governed by the terms of the QNX End User License Agreement (“EULA”). Some source code is licensed under the EULAs; source may also be available under the QNX Source Code License or Open Source Licenses. For more information, see the Legal area of our website, http://www.qnx.com/.

Distribution License

Distribution of QNX Software Systems products is governed by the terms of the QNX Runtime License Agreement or the QNX OEM License Agreement. For more information, see the Legal area of our website, http://www.qnx.com/.

DDK

Driver Development Kit — a set of source code and documentation intended to help you write your own drivers for various devices: audio, graphics, input (mice, keyboards, etc.), network, and USB.

Eclipse

Name of a tools project and platform developed by an open consortium of vendors (Eclipse.org), including QNX Software Systems.
The IDE within the QNX Momentics suite consists of a set of special plugins integrated into the standard Eclipse framework.

**EULA**

End User License Agreement.

**IDE**

Integrated Development Environment — in QNX Momentics, the IDE is based on the Eclipse framework, and includes many highly integrated tools for code development and system analysis.

IDE also stands for Integrated Drive Electronics, a standard interface used between a computer’s bus and disk storage devices.

**IPL**

Initial Program Loader — the software component responsible for setting up the machine into a usable state, such that the **startup code** can then perform further initializations.

**Momentics**

Product name for the development suite created by QNX Software Systems for its Neutrino RTOS.

**Neutrino**

Product name of the RTOS created by QNX Software Systems.

**OCL**

Open Community License.

**PE**


**QCL**

QNX Community License.
QNX

Name of an earlier-generation RTOS created by QNX Software Systems. Also, short form of the company’s name.

QoS

Quality of Service — a policy (e.g. loadbalance) used to connect nodes in a network in order to ensure highly dependable transmission. QoS is an issue that often arises in high-availability (HA) networks as well as realtime control systems.

QSS

QNX Software Systems.

RTOS

Realtime operating system.

Runtime License

See the Legal area of our website, http://www.qnx.com/.

SE


startup code

The software component that gains control after the IPL code has performed the minimum necessary amount of initialization. After gathering information about the system, the startup code transfers control to the OS.

TDK

Technology Development Kit — a kit that augments the base Neutrino OS platform in QNX Momentics 6.3.0 with specialized, value-added technology.
Index

A
account center (myQNX) 18, 25, 45
AMP (asymmetric multiprocessing) 33
architecture, system 25
audio 33

B
BMP (bound multiprocessing) 33
BSPs (Board Support Packages) 4–6, 17, 27
buildfiles 16, 34
Building Custom Widgets 27
Building Embedded Systems 26
Butenhof, David R. 43
channels 34
Chinese characters 28
CIFS (Common Interface File System) 20
coeistence of multiple versions of QNX Momentics 20
command-line utilities 16, 32
documentation 26
usage messages for 43
compiling 34
compression 34
configuration files 16
documentation 26
conventions, typographical xi
custom engineering 46
CVS (Concurrent Versions System) 19

C
C functions 26, 28
C++ 28
CD-ROMs 34
data servers 34
data types 26
DDKs (Driver Development Kits) 4, 17, 27

November 3, 2006
debugger 34
desktop 17
device drivers 16, 35
Dinkum 28
distributed processing 20, 41
distributors 45
documentation
   HTML 25
      knowing which to read first 31
      overview of 25
   PDF 25
   printed 25
   roadmap 25
   third-party 42
via the Photon Helpviewer 30
via the Start menu 29
which to read for key features of the OS 33
within the IDE 28
DOS filesystems 20
Driver Development Kits (DDKs) 4, 17, 27

QNX_HOST 14
QNX_TARGET 15
environment, configuring 35
evaluation copy, upgrading 19
event handling 35
execute in place (XIP) 42

F
 FAX 46
files
   configuration 16, 26
   handling 36
   timestamps 41
filesystems 16, 36
   mounting DOS and Linux 20
floating licenses 8
fonts 36
functions, C 26

G
Gallmeister, Bill O. 43
Getting Started with QNX Neutrino 2 42
graphics 36, 37
GUI 16. See also Photon
developing 26

H
hardware
connecting 36
supported 44
header files 16
help
via the Photon Helpviewer 30
via the Start menu 29
within the IDE 28
Helpviewer (Photon) 30, 36
HID (human input devices) 36
hosts
choice of 3
devolution on xiii
host-related components 14
supported 13
hover help 44
how-to guides 32
Hunt, Craig 43

I
I/O management 36
IDE (Integrated Development
Environment) 4
Help system 28
starting 17
User’s Guide 25
workspace directory 17
images 37
index in online documentation 31
installation, general information
about 13
InstallShield 13
instrumented kernel 27, 37
international characters 28, 41
interrupt handling 37
IPC (interprocess
communication) 36
IPL (initial program loader) 36

J
Japanese characters 28

K
kernel
calls 37
instrumented 27, 37
keyboards 37
Korean characters 28
Krten, Robert 42

L
libraries 4, 16, 37
Library Reference
Photon 26
QNX Neutrino 26
licenses 8, 16
linker 37
Linux 13
filesystems 20

M
Makefile structure 37
MAKEFLAGS  16
man command  43
memory management  38
menus  17
message passing  38
message queues  38
migrating from QNX 4 to QNX Neutrino  9
mkifs  4
multicore systems  38
multilingual input  28
Multimedia Developer’s Guide  26
myQNX account center  18, 25, 45

N
network drivers  38
networking, transparent (native)  20, 41
newsgroups  45
NFS (Network File System)  20

P
packages  38
pathname delimiter in QNX
  Momentics documentation xii
PDF documents  25
PE (Professional Edition)  3
permissions  38
Phindows  28
Photon
  Building Custom Widgets  27

Helpviewer  30
Library Reference  26
multilingual input  28
Multimedia Developer’s Guide  26
Programmer’s Guide  26
Widget Reference  27
Windows, accessing from  28
platform-specific issues  7
POSIX
docs available online  42
link to site for up-to-date status  42
POSIX.4: Programming for the Real World  43
power management  39
printed books  25
process manager  39
processes  39
Professional Edition (PE)  3
professional services  46
profiler  39
Programmer’s Guide
  Photon  26
  QNX Neutrino  26
Programming with POSIX
  Threads  43
pulses  39

Q
qcc  4
qconfig  20
qconn  40
qde  18
Qnet 20, 41
QNX 4, migrating from 9
QNX Cookbook: Recipes for Programmers 42
QNX Momentics
about xiii
coexistence of multiple versions 20
End User License Agreement (EULA) 8
evaluation copy, upgrading 19
features of 3
parts of 3
system requirements 13
upgrading 18
QNX Neutrino
Building Embedded Systems 26
community websites 45
Library Reference 26
Programmer’s Guide 26
self-hosted 13, 19
System Architecture 25
Technical Notes 26
User’s Guide 25, 33
Utilities Reference 26
QNX Software Systems website 44
QNX_CONFIGURATION 16
QNX_HOST 14
QNX_TARGET 15
QoS (Quality of Service) 39
QWinCfg 20

R
RAM disk 39
realtime scheduling 39
reference books 32
resellers 45
resource management 40
runtime
CD 9
directory structure 17
documentation 33
self-hosted development 13

S
scheduling 39
SE (Standard Edition) 3
self-hosted development 13, 19, 40
documentation 29
signal handling 40
slinger 42
SMP (symmetric multiprocessing) 40
software
developing 26
managing 19, 41
updates 18
Solaris 13
source code 3
Standard Edition (SE) 3
startup programs 40
Stevens, W. R. 43
support plans 46
synchronization 40
system
administration 33
Index

header files 16
information 40
requirements 13
System Analysis Toolkit 27
System Architecture 25

T

targets
agent (qconn) 40
developing for 3, 20, 26
running on xiii, 5
target-related components 14
TCP/IP 20
TCP/IP Illustrated, Volume 1: The Protocols 43
TCP/IP Illustrated, Volume 2: The Implementation 43
TCP/IP Network Administration 43
TDKs (Technology Development Kits) 5, 28
Technical Notes 26
technical support 46
public forums 46
Technology Development Kits (TDKs) 5, 28
telephone, support via 46
threads 41
time 41
timers 41
timestamps 41
tools 3
training 46
transparent distributed processing 20, 41
typographical conventions xi

U

Unicode 41
UNIX Network Programming,
Volume 1, Second Edition:
Networking APIs: Sockets and XTI 43
USB (Universal Serial Bus) 41
use command 43
User’s Guide
IDE 25
QNX Neutrino 25, 33
utilities 16, 32
Utilities Reference 26

V

version control 41
Virtio 7
virtual machine 6
Virtual PC 6
VMware 6

W

web server (slinger) 42
websites
Eclipse consortium 45
QNX Neutrino community 45
QNX Software Systems 44
Index

Widget Reference  27
Windows  13
    accessing Photon from  28
workspace directory  17
Workstation  6

X

XIP (execute in place)  42