

Wind River® General Purpose Platform, VxWorks® Edition

RELEASE NOTES

3.6

Copyright © 2010 Wind River Systems, Inc.

All rights reserved. No part of this publication may be reproduced or transmitted in any form or by any means without the prior written permission of Wind River Systems, Inc.

Wind River, Tornado, and VxWorks are registered trademarks of Wind River Systems, Inc. The Wind River logo is a trademark of Wind River Systems, Inc. Any third-party trademarks referenced are the property of their respective owners. For further information regarding Wind River trademarks, please see:

<http://www.windriver.com/company/terms/trademark.html>

This product may include software licensed to Wind River by third parties. Relevant notices (if any) are provided in your product installation at the following location:

installDir\product_name\3rd_party_licensor_notice.pdf.

Wind River may refer to third-party documentation by listing publications or providing links to third-party Web sites for informational purposes. Wind River accepts no responsibility for the information provided in such third-party documentation.

Corporate Headquarters

Wind River Systems, Inc.
500 Wind River Way
Alameda, CA 94501-1153
U.S.A.

toll free (U.S.): (800) 545-WIND
telephone: (510) 748-4100
facsimile: (510) 749-2010

For additional contact information, please visit the Wind River URL:

<http://www.windriver.com>

For information on how to contact Customer Support, please visit the following URL:

<http://www.windriver.com/support>

Contents

1	Overview	1
1.1	Introduction	1
1.2	Features	2
1.3	Installation and Licensing	3
1.4	Configuration	4
1.5	Migration and Backward Compatibility	4
1.6	Latest Release Information	4
2	Changes in This Release	5
2.1	Introduction	5
2.2	Enhancements	6
2.2.1	Debugger Enhancements	6
2.2.2	Compiler Enhancements	6
2.2.3	Networking and Middleware Enhancements	7
2.2.4	Operating System Enhancements	9
2.2.5	Development Environment Enhancements	11

2.2.6	BSP Enhancements	13
2.2.7	Documentation Enhancements	13
2.3	Deprecated Features	14
2.4	Unsupported Features	15
3	System Requirements	17
3.1	Introduction	17
3.2	Host	17
3.2.1	System Requirements	18
	Windows Host	18
	Solaris Host	18
	Linux Host	19
3.3	Target	20
3.3.1	System Requirements	20
3.3.2	Supported Hardware and Software	20
	Architectures	21
	Hardware	24
4	Known Problems	25
4.1	Introduction	25
4.2	Known Problems	26
	Existing Wind River Product Installations	26
	Generating Includes in Downloadable Kernel Modules	27
	TrueFFS Support on the ColdFire Architecture	27
4.3	Usage Caveats	28
4.4	Documentation Errata	32

5	Wind River Workbench	37
5.1	Introduction	37
5.2	Changes in This Release	38
5.2.1	Enhancements	39
	Wind River Workbench	39
	Wind River Run-Time Analysis Tools	47
	Wind River System Viewer	49
5.2.2	Fixed Problems	50
5.2.3	Deprecated Features	50
	Standard Managed Build	50
5.2.4	Unsupported Features	51
5.3	Usage Caveats	51
	Workbench Issues	51
	Project and Build Issues	60
	Wind River Run-Time Analysis Tools	60
	Windows Host-Related Issues	61
	Linux Host-Related Issues	61
	Solaris Host-Related Issues	62
5.4	Known Problems	62
	Workbench Issues	62
	Project Issues	67
	Remote Systems View Issues	67
	Build Issues	68
	Build Properties Issues	69
	Debugging Issues	69
	Indexer Issues	72
	Properties View Issues	73
	Run-Time Analysis Tools Issues	73
	System Viewer Issues	73
5.5	Documentation Errata	74

6	VxWorks	75
6.1	Introduction	75
6.2	Changes in This Release	76
6.2.1	Enhancements	76
	VxWorks SMP	76
	Distributed Shared Memory	77
	Scheduler Extension: taskRotate()	77
	Read/Write Semaphore	78
	Disable Stack Filling	78
	Task-Specific Variables with __thread Local Storage	78
	Small VxWorks Configuration Profiles	79
	I/O System	79
	HRFS File System	80
	dosFs File System	80
	Flash File System Support with TrueFFS	81
	SDA Support for Power Architecture	81
	Kernel Configuration	81
	Boot Loader	82
	Kernel Shell	82
	Kernel Object Module Loader	82
	WDB Target Agent	82
	BSPs	83
	pcPentium-Based BSP Configuration	83
	Drivers for VxWorks SMP	84
	New APIs	86
	Compilers	87
6.2.2	API Changes	87
6.2.3	Fixed Problems	91
6.2.4	Deprecated Features	91
6.2.5	Unsupported Features	94
6.3	Usage Caveats	95
6.4	Known Problems	98
6.5	Documentation Errata	99

7	Wind River VxWorks Simulator	101
7.1	Introduction	101
7.2	Changes in This Release	102
7.2.1	Enhancements	102
7.2.2	Fixed Problems	102
7.3	Known Problems	102
7.4	Documentation Errata	103
8	Wind River Compiler	105
8.1	Introduction	105
8.2	Changes in This Release	106
8.2.1	Enhancements	106
8.2.2	Fixed Problems	108
8.2.3	Unsupported Features	108
8.3	Supported Hardware and Software	109
8.4	Usage Caveats	109
	C and C++ Compilers	109
	Linker	110
	Far Relative Addressing and VLE	110
8.5	Known Problems	111
8.6	Documentation Errata	112
9	Wind River GNU Compiler	113
9.1	Introduction	113
9.2	Changes in This Release	114
9.2.1	Enhancements	114

9.2.2	Fixed Problems	114
9.2.3	Unsupported Features	114
9.3	Usage Caveats	115
9.4	Known Problems	118
9.5	Documentation Errata	119
10	Wind River Network Stack	121
10.1	Introduction	121
10.2	Changes in This Release	122
10.2.1	API Changes	124
10.2.2	Fixed Problems	124
10.2.3	Deprecated Features	124
10.3	Usage Caveats	125
10.4	Known Problems	126
10.5	Documentation Errata	126
11	Wind River PPP	127
11.1	Introduction	127
11.2	Changes in This Release	128
11.2.1	Enhancements	128
11.2.2	Fixed Problems	128
11.3	Known Problems	128
11.4	Documentation Errata	128
	API Reference	129

12	Wind River TIPC	131
12.1	Introduction	131
12.2	Changes in This Release	132
12.2.1	Enhancements	132
	Distributed Shared Memory (DSHM)	132
	Footprint Reduction	133
	TIPC Native API	133
	Test Suite	133
	Expanded Networking Capabilities	134
	Build Component for the TIPC Sample Application	134
	New tipcConfig Options	134
12.2.2	API Changes	135
	New TIPC Native API	135
	Changed Parameter Requirement for shutdown() Routine	135
12.2.3	Fixed Problems	135
12.3	Known Problems	135
12.4	Documentation Errata	136
13	Wind River USB	137
13.1	Introduction	137
13.2	Changes in This Release	138
13.2.1	Enhancements	138
13.2.2	API Changes	138
13.2.3	Fixed Problems	138
13.2.4	Unsupported Features	138
13.3	Supported Hardware and Software	139
	USB Peripheral Stack BSPs	140
13.3.1	Tested Devices	140
	USB Host Stack Devices	140

USB Peripheral Stack Controllers 142

13.4 Usage Caveats 143

13.5 Known Problems 143

13.6 Documentation Errata 143

1

Overview

1.1	Introduction	1
1.2	Features	2
1.3	Installation and Licensing	3
1.4	Configuration	4
1.5	Migration and Backward Compatibility	4
1.6	Latest Release Information	4

1.1 Introduction

Wind River General Purpose Platform, VxWorks Edition, 3.6 is the latest release of the VxWorks operating system, Wind River Workbench, the Wind River Compiler, the Wind River GNU Compiler, and other middleware and host tools. Important features of this release include the following:

- backward-compatible OS
- symmetric multiprocessing (SMP) capability
- highly scalable, OS-independent network stack
- updated networking products

- state-of-the-art memory protection including real-time processes (RTPs), error detection and reporting, and interprocess communication (IPC)
- architecture support for ARM, ARM/XScale, ColdFire, IA-32, MIPS, Power Architecture, and SuperH
- middleware products integrated with VxWorks and Wind River Workbench

1.2 Features

The General Purpose Platform includes a set of development tools and software run-time products, at the core of which is the VxWorks real-time operating system. The General Purpose Platform extends VxWorks by adding networking and communications products, as well as an enhanced set of host tools including compilers, static analysis, run-time analysis, and debugging tools.

The General Purpose Platform includes the following products:

- VxWorks 6.6, a high-performance, memory protected real-time operating system with support for symmetric multiprocessing (SMP).
- Wind River VxWorks Simulator 6.6, the VxWorks host-based target simulator.
- Wind River Workbench 3.0, a development suite that facilitates managing and building projects, establishing and managing host-target communication, and running, debugging, and monitoring VxWorks applications. Wind River Workbench also includes the following:
 - Wind River System Viewer, a software logic analyzer that provides graphical representations of the dynamic interactions of elements of the system.
 - Wind River Run-Time Analysis Tools (formerly called ScopeTools), including:
 - Wind River Memory Analyzer (formerly MemScope)
 - Wind River Performance Profiler (formerly ProfileScope)
 - Wind River Data Monitor (formerly StethoScope)

- Wind River Code Coverage Analyzer (formerly CoverageScope)
 - Wind River Function Tracer (formerly TraceScope)
- Wind River Compiler 5.6, a compiler for C and C++ development.
- Wind River GNU Compiler 4.1.2, a compiler for C and C++ development.
- Wind River Network Stack 6.6, a full-featured IPv4/IPv6 dual stack, specifically designed for next-generation device software applications. Wind River Network Stack is an ideal choice for devices that require a combination of networking features and high performance. With its small and scalable memory footprint, Wind River Network Stack is well suited for resource- and cost-constrained devices.
- Wind River PPP 6.6, which allows for the establishment, authentication, and control of Point-to-Point Protocol (PPP) connections and PPP over Ethernet (PPPoE) implementations. PPP provides a standard method for transporting multi-protocol datagrams over point-to-point links. PPPoE provides the ability to connect a network of hosts over a simple bridging access device.
- Wind River TIPC 1.7, an implementation of the Transparent Inter Process Communication (TIPC) protocol. This product is a port of the open-source implementation on Linux.
- Wind River USB 2.4, including both a Universal Serial Bus (USB) host stack and a USB peripheral stack. The USB host stack enables you to write your own USB host controller driver. The USB peripheral stack enables you to write your own USB target application or target controller driver.

1.3 Installation and Licensing

For information on installing your Platform and configuring your product licenses, see the Wind River Site Configuration Setup Guides. They are accessible from the following URL:

<http://www.windriver.com/licensing>

1.4 Configuration

Configuration information for all products included in this release is available in *Wind River General Purpose Platform, VxWorks Edition Getting Started*.

1.5 Migration and Backward Compatibility

Information on migrating your existing installation and applications for use with the General Purpose Platform is available in the *Wind River General Purpose Platform, VxWorks Edition Migration Guide*.

1.6 Latest Release Information

The latest information on this release can be found in the Wind River General Purpose Platform, VxWorks Edition area of the Wind River Online Support Web site:

<http://www.windriver.com/support/>

This site includes links to topics such as known problems, fixed problems, documentation, and patches.



NOTE: Wind River strongly recommends that you visit the Online Support Web site before installing or using this product. The Online Support Web site may include important software patches or other critical information regarding this release.

For information on accessing the Wind River Online Support Web site, see *Customer Services*, p.145.

2

Changes in This Release

- 2.1 Introduction 5
- 2.2 Enhancements 6
- 2.3 Deprecated Features 14
- 2.4 Unsupported Features 15

2.1 Introduction

This release of the General Purpose Platform includes the introduction of VxWorks SMP, as well as substantial updates to VxWorks, the Wind River Network Stack, Wind River Workbench, and several middleware products. This chapter provides an overview of new features as well as information on items that are no longer available. For more detailed information on changes to the products included in your Platform, see the individual chapter on that product in this document.

2.2 Enhancements

This release of the General Purpose Platform includes expanded architecture support and improvements to the development suite and operating system, including feature updates, defect fixes, and documentation enhancements.

The following sections describe important enhancements in this release.

2.2.1 Debugger Enhancements

The debugger has been improved, and now includes support for the following:

- SMP task mode and system mode debugging for simulators and real targets.
- Wind River Compiler 5.6 and Wind River GNU Compiler 4.1.
- Debugging of VxWorks 5.5.x targets on Solaris hosts.
- Accessing thread-local storage (TLS) variables.
- New CPU architectures, including:
 - ARM Thumb-2
 - Cavium OCTEON
 - ColdFire v3
 - MIPS 74kf and 24kf
 - RMI XLR
- Linux 64-bit native host application debugging for 64-bit Red Hat Workstation 5 hosts.
- Expression evaluation improvements including **w_char*** casting and pointer to array casting.

For more information on enhancements to the debugger, see [Debugger](#), p.41.

2.2.2 Compiler Enhancements

Wind River Compiler

The Wind River Compiler 5.6 includes the following enhancements:

- Support for the **__builtin_prefetch** and **__builtin_expect** intrinsic functions.
- Support for the **__thread** keyword to allocate thread-local storage.
- Support for the ARM Thumb-2 architecture.

- Enhanced behavior for **-Xkill-opt**.
- A new **OPTIONAL** section type specifier for the linker.
- Support for 64-bit bit-fields.
- Support for enabling certain MIPS instructions.
- New error message documentation.

For more information on enhancements to the Wind River Compiler, see [8. Wind River Compiler](#).

Wind River GNU Compiler

The Wind River GNU Compiler 4.1.2 provides improved optimization and language support, new target architecture support, bug fixes, and other enhancements.

For more information on enhancements to the Wind River GNU Compiler, see [9. Wind River GNU Compiler](#).

2.2.3 Networking and Middleware Enhancements

Wind River Network Stack

This release of the Wind River Network Stack includes enhancements in the following functional areas:

- SMP support
- IPv6-only build option
- Mobile IP
- DHCP
- FTP
- 802.1Q VLAN
- Quality of Service (QoS) – Differentiated Services (Diffserv)
- MIB support



NOTE: If you are programming in a uniprocessor system, the **tNetTask** task is now called **tNet0**. If you are using VxWorks SMP, multiple instances are available and are named **tNet*n***. For more information, see the *Network Drivers* chapter of the *VxWorks Device Driver Developer's Guide, Volume 2: Writing Class-Specific Device Drivers*.

For more information on enhancements to the Wind River Network Stack, see [10. Wind River Network Stack](#).

TIPC

Wind River TIPC 1.7 includes the following enhancements:

- Support for distributed shared memory (DSHM).
- Build options for reducing TIPC's footprint.
- Introduction of a new API, the TIPC native API, that can provide both footprint reduction and faster performance.
- A new test suite for debugging TIPC applications.
- Expanded networking capabilities through support for multiple zones and clusters.
- A new build component for including the sample TIPC application of the previous release in a VxWorks kernel image.
- New command options for the **tipcConfig** utility.

For more information on enhancements to Wind River TIPC, see [12. Wind River TIPC](#).

Wind River USB

Wind River USB 2.4 includes enhancements to support VxBus device drivers and a bi-directional USB headset peripheral device.

For more information on enhancements to Wind River USB, see [13. Wind River USB](#).

2.2.4 Operating System Enhancements

VxWorks 6.6 includes the following enhancements:

- **VxWorks SMP**

SMP is a configuration of VxWorks designed to provide symmetric multiprocessing, providing the same distinguishing RTOS characteristics of performance and determinism. For more information on SMP, see [6. VxWorks](#).

The WDB target agent provides support for VxWorks SMP. This support includes task mode and system mode debugging, allowing users to debug kernel tasks, RTP tasks and interrupt handlers.

This release includes new BSPs and drivers that support SMP. For more information on these BSPs and drivers, see [Supported BSPs](#), p.22 and [Drivers for VxWorks SMP](#), p.84.

- **Distributed shared memory**

The VxWorks distributed shared memory (DSHM) facility is a middleware subsystem that allows multiple services to communicate over different types of buses that support shared-memory communication.

For more information on DSHM, see [6. VxWorks](#) and [12. Wind River TIPC](#).

- **Read/write semaphore**

Read/write semaphores provide enhanced performance for applications that can effectively make use of differentiation between read access to a resource, and write access to a resource (as with SMP applications).

- **Small VxWorks configuration profiles**

Basic I/O support has been added to the VxWorks `PROFILE_BASIC_KERNEL` small (source-scalable) configuration profile.

- **I/O system**

The component `INCLUDE_IO` has been split to allow more precise configuration of the I/O system.

- **HRFS file system**

The Highly Reliable File System (HRFS) now provides configurable transaction points, which allow for finer control of how and when transaction points are set, to enable better performance.

- **dosFs file system**

For this release, the cache has been modified to allow for fine-tuning of the cache and improved performance.

- **Flash file system support with TrueFFS**

With this release, TrueFFS automatically disables FAT monitoring and sets the start sector to 1.

- **SDA support for Power Architecture**

Small data area (SDA) support has been implemented for the VxWorks kernel programming environment with the Power Architecture (PowerPC). SDA is designed to take advantage of base plus displacement addressing mode, which provides a more memory-efficient way of accessing a variable and better performance.

- **Kernel configuration**

The **vxprj** VxWorks configuration utility provides support for VxWorks SMP and for asymmetric multiprocessor (AMP) projects.

- **Boot loader**

The boot loaders provided with this release can be used for VxWorks UP, SMP, and AMP projects. There is no special boot loader for symmetric multiprocessing or asymmetric multiprocessing.

- **Kernel shell**

The VxWorks kernel shell now provides commands that allow you to save the shell history to, and load it from, a file. The kernel shell has also been enhanced with new commands for networking technologies.

- **Kernel object module loader**

The kernel object module loader and the host loader now include an option for loading fully linked modules.

- **Additional VxWorks functionality**

VxWorks now offers added functionality in the following areas:

- disabling of stack filling (which can be used to speed up system boot time and task initialization times)
- **semExchange()**
- **taskRotate()** scheduler extension
- TLS support

For more information on enhancements to VxWorks, see [6. VxWorks](#).

2.2.5 Development Environment Enhancements

Wind River Workbench 3.0 includes the following enhancements:

- **Adoption of Eclipse CDT**

In this release, Workbench has adopted several Eclipse projects, including the C/C++ Development Toolkit 4.0 (CDT). This change replaces some Workbench tools with their Eclipse counterparts:

- CDT Indexer for static analysis.
- Workbench Editor replaced by CDT C/C++ Editor.
- Project Navigator replaced by Eclipse Project Explorer.
- Target Manager replaced by Eclipse Remote Systems view.

- **SMP support**

In this release, Workbench provides support for the optional VxWorks SMP product.

- **Project and build systems**

The project and build systems have been enhanced for ease of use and flexibility, simpler migration, and improved performance.

- **Static analysis**

This release includes the following enhancements in static analysis:

- The two modes of static analysis have been separated.
- Importing a project to a different workspace also imports the shared symbol data.
- You can now configure indexer-specific exclusion filters using regular expressions.
- You can now reindex a subset of files in a project, updating only those files that have been modified, or all files.

- **View enhancements**

This release of Workbench includes several improvements to views:

- To improve multi-core configuration and control, the Debug view now has a **Link Cores/Unlink Cores** toggle button.
- The Terminal view now includes a **Toggle Command Input Field** button, which opens a text inset field that allows full editing capabilities inside Workbench.
- Upon startup, Workbench displays a new Getting Started Resources view containing links to the Wind River Online Support site, Workbench documentation, and other useful resources.
- The RSS Feed view now provides quick links to Workbench patches and other critical information.
- Several Workbench views have been replaced by Eclipse views.

Wind River Run-Time Analysis Tools

- The entire suite of Wind River Run-Time Analysis Tools (formerly ScopeTools) was renamed in this release.
- For the following tools, the previous standalone window version has been eliminated. The tools have been integrated to run from within Workbench. For some tools, this integration required major functional changes. For more information, see [Wind River Run-Time Analysis Tools](#), p.47.
 - Code Coverage Analyzer
 - Memory Analyzer
 - Performance Profiler

Wind River System Viewer

Support for multi-core systems has been added in the Log Viewer and the new Analysis Suite views.

For more information on enhancements to Workbench, see [5. Wind River Workbench](#).

2.2.6 BSP Enhancements

This release includes the following new BSPs:

- idp945
- ads8540
- cav_cn3xxx_mipsi64r2sf
- hpc2-7448
- hpcNet8641
- lite5200b
- m5329evb
- malta74kf
- mrvl_db88f5181_mmu
- mv3100
- mv6100
- wrSbc8548
- wrSbc8641d

2.2.7 Documentation Enhancements

Accessing Documentation on Windows

Windows users can now access Wind River documentation by selecting **Start > Programs > Wind River > Documentation > *productName***.

VxWorks Device Driver Developer's Guide

The *VxWorks Device Driver Developer's Guide* was previously contained in a single volume. It now consists of the following volumes:

- *VxWorks Device Driver Developer's Guide, Volume 1: Fundamentals of Writing Device Drivers*
- *VxWorks Device Driver Developer's Guide, Volume 2: Writing Class-Specific Device Drivers*
- *VxWorks Device Driver Developer's Guide, Volume 3: Legacy Drivers and Migration*

New Error Message Documentation

A new manual, *Wind River Compiler Error Messages*, is included with this release. It lists several hundred new and previously undocumented error messages.

C++, C99 Libraries Documentation

Documentation for the C++ and C99 libraries can now be found online at https://portal.windriver.com/noAuth/wr_compiler_docs. Note that nonstandard C++ and C99 headers are not supported.

Data Monitor Documentation

The Windows and UNIX versions of the former StethoScope user's guides have been merged into one document, the *Wind River Workbench Data Monitor User's Guide*.

2.3 Deprecated Features

This section lists features that have been deprecated for the Wind River General Purpose Platform, VxWorks Edition as a whole. For detailed information on deprecated features for a specific product, see the product-specific chapter in this document.

IPCOM Routines

The routines `ipcom_run_cmd()` and `ipmcp_cmd()` are deprecated.

VxWorks

The following features have been deprecated in VxWorks 6.6:

- configuration and build using **config.h**
- the **vxWorks.st** image type
- the **taskVarLib** and **tlsLib** task-specific variables
- VxWorks routines for accessing VxMP objects

BSPs

The following BSPs are no longer available on the VxWorks installation media and will not be supported after this release of VxWorks. That is, they will not be updated and shipped with future releases of the product.

- **lite5200**
- **mds8360**

These BSPs are available for download from the Online Support Web site at the following URL. Select your release from the list provided.

https://portal.windriver.com/products/bsp_web/bsp_platform.html

For a list of currently supported BSPs, see *Supported BSPs*, p.22.

2.4 **Unsupported Features**

This section lists features that were available in the Wind River General Purpose Platform, VxWorks Edition, 3.5 and are not available in the Wind River General Purpose Platform, VxWorks Edition, 3.6. For detailed information on unsupported features for a specific product, see the product-specific chapter in this document.

Non-VxBus Drivers

As of this release of VxWorks, non-VxBus implementations of the host controller drivers are unsupported.

BSPs

The following BSPs are no longer supported. That is, they have not been updated for this release and are no longer shipped with the product.

- **ebony**
- **mv2400**
- **mv2700**
- **wrPpmc440gp**

3

System Requirements

[3.1 Introduction 17](#)

[3.2 Host 17](#)

[3.3 Target 20](#)

3.1 Introduction

This chapter lists the minimum requirements for running the General Purpose Platform in a development environment where the host and target are separate computers.

3.2 Host

The host is the computer on which you do your development work. This section lists the minimum requirements for running the General Purpose Platform in the standard configuration.

3.2.1 System Requirements

These system requirements are for the General Purpose Platform alone; they do not take into consideration any other software you are running on the host computer.

Windows Host

- Windows XP Professional (Service Pack 2), Windows Vista Business, or Windows Vista Enterprise.
- Administrator rights.
- Intel Pentium 4 class computer with a 2 GHz processor, or a computer with higher performance.
- 1 GB of RAM (2 GB of RAM is recommended for larger projects).
- 3.8 GB disk space (for a typical installation). When calculating the amount of disk space needed, be sure to reserve space for your own applications and development.
- A local DVD-ROM drive or access to a network for installation.
- A current version of a standards-compliant Web browser.
- TCP/IP must be installed on the host system, even if it is being used as a standalone PC with a serial connection to the target.
- A network interface card for debugging over a network (recommended).
- An active Internet connection is recommended during initial installation to access patches, documentation, and other important information from the Wind River Online Support Web site. (See [1.6 Latest Release Information](#), p.4.)

Solaris Host

- Sun Solaris 9 or 10.
- A Blade 150 workstation with a 500 MHz processor, or a workstation with higher performance.
- 1 GB of RAM.

- 3.8 GB of disk space (for a typical installation). When calculating the amount of disk space needed, be sure to reserve space for your own applications and development.
- A local DVD-ROM drive or access to a network for installation.
- A current version of a standards-compliant Web browser.
- CDE Window Manager (recommended).
- An active Internet connection is recommended during initial installation to access patches, documentation, and other important information from the Wind River Online Support Web site. (See [1.6 Latest Release Information](#), p.4.)

Linux Host

- One of the following host operating systems:
 - Red Hat Enterprise Linux Workstation 4.0, Update 5
 - Red Hat Enterprise Linux Desktop with Workstation option 5 (32- or 64-bit)
 - SUSE Linux/openSUSE 10.2
 - Novell SUSE Linux Enterprise Desktop 10 (Service Pack 1)
 - Fedora 7¹
- GNOME Window Manager.
- Intel Pentium 4 class computer with a 1 GHz processor, or a computer with higher performance.
- 1 GB of RAM.
- 3.8 GB of disk space (for a typical installation). When calculating the amount of disk space needed, be sure to reserve space for your own applications and development.
- A local DVD-ROM drive or access to a network for installation.
- TCP/IP must be installed on the host system.
- A network interface card for debugging over a network (recommended).
- A current version of a standards-compliant Web browser.

1. GNOME terminal is required for Fedora Core 7.

- An active Internet connection is recommended during initial installation to access patches, documentation, and other important information from the Wind River Online Support Web site. (See [1.6 Latest Release Information](#), p.4.)

3.3 Target

The target is the computer for which you are developing. This section lists the minimum requirements for targets running VxWorks.

3.3.1 System Requirements

Your target system must consist of either the Wind River VxWorks Simulator or a supported target board with the following:

- 8 MB of RAM (more if a RAM disk is used). All supported reference boards from Wind River include at least 8 MB of memory.
- Space for a file system on the hard disk, RAM disk, flash memory, or floppy disk.
- An Ethernet cable or null crossover cable (needed for initial board setup).
- An RS-232 null modem cable (needed for initial board setup).
- A floppy disk drive or network installation card for installation of a file system.
- A keyboard (recommended for configuration if using network booting).
- A monitor (recommended for configuration if using network booting).

3.3.2 Supported Hardware and Software

This section lists additional hardware and software that can be used on a target running VxWorks and addressed by the tools included with the General Purpose Platform.

Architectures

The following architectures are supported in this release:

- ARM
 - ARM Architecture 4
 - ARM Architecture 5
 - ARM Architecture 6
 - ARM/XScale
- ColdFire
 - ColdFire V3
 - ColdFire V4e
- IA-32
 - P5 (Pentium)
 - P6 (PentiumPro, Pentium II, Pentium III)
 - P7 (Pentium IV)
 - Pentium M
 - Xeon LV²
 - Core Duo²
- MIPS
 - Broadcom BCM3300
 - Broadcom SiByte SB1a³
 - Cavium OCTEON CN3xxx²
 - MTI 4K
 - MTI 5K
 - MTI 24K
 - MTI 74K
 - NEC VR5500
 - PMC-Sierra RM9k
 - Toshiba TX4938
- Power Architecture (PowerPC)
 - Freescale E200
 - Freescale E300
 - Freescale E500
 - Freescale E600³
 - IBM/AMCC 405
 - IBM/AMCC 440
 - IBM 970

2. Supports symmetric multiprocessing (SMP).

3. Supports symmetric multiprocessing (SMP) and asymmetric multiprocessing (AMP).

- SuperH
 - SH-4
 - SH-4A
- Wind River VxWorks Simulator

Supported BSPs

The following BSPs are supported in this release.



NOTE: BSPs that are not available on the physical media shipped with your Platform are available from the following URL on the Wind River Online Support Web site: https://portal.windriver.com/products/bsp_web/bsp_platform.html.

ARM

- integrator920t
- integrator926ejs
- integrator1136jfs
- mrvl_db88f5181

ARM/XScale

- ixdp425
- ixdp465

ColdFire

- m54x5evb
- m5329evb

IA-32

- idp945
- pcPentium
- pcPentium2
- pcPentium3
- pcPentium4

MIPS

- cav_cn3xxx_mipsi64r2sf
- malta4kc
- malta4kec
- malta5kc

- malta5kf
- malta24kc
- malta24kec
- malta74kf
- rbt4938
- rh5500
- sb1250a
- sb1250e
- sb1480
- yosemiteRm9k

Power Architecture (PowerPC)

- ads827x
- ads860
- ads88x
- ads834x
- ads8540
- ads8560
- amcc440ep
- cds85xx
- cds8548
- hpcNet8641
- hpc2-7448
- lite5200b
- mcpn805
- mv3100
- mv5100
- mv5500
- mv6100
- ocotea
- sp7447
- walnut
- wrSbc405gp
- wrSbc834x
- wrSbc8540
- wrSbc8548
- wrSbc8560
- wrSbc8641d
- wrSbcPowerQuiccII

SuperH

- **ms7751Rse**
- **lanbic7770**
- **sdk7780**

Wind River VxWorks Simulator

- **linux**
- **simpc**
- **solaris**

For the latest General Purpose Platform information, including architecture and BSP updates, see the following Web site:

<http://www.windriver.com/support>

Additionally, you can subscribe to Wind River's RSS feed to receive information about hardware support on the General Purpose Platform. This RSS feed is available within Workbench's RSS Feed view (or through a third-party RSS reader).

Hardware

For a list of supported hardware for your target architecture, see the BSP **target.ref** file, generally located at *installDir/vxworks-6.6/target/config/bspName*.

4

Known Problems

- 4.1 Introduction 25
- 4.2 Known Problems 26
- 4.3 Usage Caveats 28
- 4.4 Documentation Errata 32

4.1 Introduction

This chapter lists known problems, usage caveats, and documentation errata for the General Purpose Platform as a whole. For a list of issues for a specific product, see the product-specific chapter in this document.

The most current information for this release is available from the Wind River Online Support Web site (see [1.6 Latest Release Information](#), p.4).

4.2 Known Problems

This section highlights known issues that may have a significant impact on installation and your initial experience with this product. The current status of these issues is available online. Defect numbers, where applicable, are given in parentheses after the description of the issue.

Existing Wind River Product Installations

Pre-Production Installations

If you have a pre-production release of VxWorks 6.6 or Wind River Workbench 3.0 installed, you must remove the installed software before installing the official release version. You do not need to remove your workspace directory. For details on how to uninstall the product, see the Wind River Site Configuration Setup Guides, available from the following URL:

http://www.windriver.com/support/site_configuration

Stopping Existing Registry Services

If you have any other Wind River products installed, check to see if the registry service is running before starting a VxWorks 6.6 session, and stop any existing Tornado registry service. If an existing registry service is running, it prevents you from connecting to a target. For example, you will not be able to start a target server, a Wind River VxWorks Simulator session, or the Wind River System Viewer.

On Linux or Solaris:

1. To check if a registry is running, from a terminal window type the following:

```
% ps -auwx | grep wtxregd
```

2. If **wtxregd** is running, note its process ID and type the following:

```
% kill processid
```

On Windows:

On a Windows host, to stop the Tornado registry service or to determine whether a service is running, do the following:

1. Select **Start > Settings > Control Panel**.
2. In the **Control Panel** window, double-click **Administrative Tools**.

3. In the **Administrative Tools** window, double-click **Services**.
4. Look for a **Tornado Registry** entry in the **Services** window. If there is a **Tornado Registry** entry, stop the service.

Or, to locate manually started services, do the following:

1. Look in the system tray of your Windows desktop and locate the icon that resembles the Wind River logo.
2. Right-click the icon to stop the service.

Wind River recommends that you set your Tornado registry services to **Manual** startup instead of **Automatic** to prevent the Tornado registry from starting up automatically when you boot your computer. You can change the startup type by right-clicking the **Tornado Registry** entry in the **Services** window and selecting **Properties**. Change the startup type by selecting **Manual** from the drop-down list in the **Startup type** field.

In addition to stopping any previous Wind River registry services, you must remove or clear the **WIND_REGISTRY** environment variable from your host system.

Before starting a command-line development session, start the Wind River Registry manually. Select
Start > Programs > Wind River > Wind River Workbench 3.0 > Registry.

Generating Includes in Downloadable Kernel Modules

When creating or importing a VxWorks downloadable kernel module (DKM) project, include paths are set up as absolute paths. When you build your project for the first time and select **Generate Includes...**, the resultant environment variables are incorrect. To correct the paths, replace the absolute paths with **WIND_BASE**. (WIND00102743)

TrueFFS Support on the ColdFire Architecture

The ColdFire **m54x5evb** BSP included with this release does not provide support for TrueFFS. However, TrueFFS can be used with the ColdFire architecture if you provide the necessary support in your custom BSP. For more information on adding TrueFFS support, see the *VxWorks Kernel Programmer's Guide: Flash File System Support with TrueFFS* and the *Non-Volatile RAM Drivers* chapter of the

VxWorks Device Driver Developer's Guide, Volume 2: Writing Class-Specific Device Drivers.

4.3 Usage Caveats

This section describes usage issues that you should be aware of when working with the General Purpose Platform. For a list of usage caveats for a specific product, see the product-specific chapter in this document.

Application Binaries

Application binaries (kernel or RTP) compiled with earlier versions of VxWorks 6.x are not compatible with VxWorks 6.6. All applications must be recompiled with VxWorks 6.6.

Multiple Users and Installations of Workbench

Different configurations are possible for the number of Workbench users and the number of Workbench installations on a single host. For information on the considerations for each configuration, see [5. Wind River Workbench](#).



NOTE: Each user should work in a user-specific workspace. Sharing of workspaces is not possible. For more information, see [5. Wind River Workbench](#).

Workbench Workspace Location

Wind River strongly recommends placing your workspace on a local file system. Remote file systems can cause significant performance problems. If you use IBM ClearCase, place the workspace outside a dynamic view, otherwise you will face significant performance problems.

Supported Make System

The Workbench make system is based on features of the GNU Make utility (**gmake**). Other make systems are not supported.

Scalable Projects Require Source

In order to build scalable projects, you must have the VxWorks source code product. The source code is available for purchase as an add-on to the General Purpose Platform; see your Wind River sales representative for details.

tNetTask

If you are programming in a uniprocessor system, the **tNetTask** task is now called **tNet0**. If you are using VxWorks SMP, multiple instances are available and are named **tNet*n***. For more information, see the *Network Drivers* chapter of the *VxWorks Device Driver Developer's Guide, Volume 2: Writing Class-Specific Device Drivers*.

Cavium BSP

To use the new Cavium OCTEON BSP (**cav_cn3xxx_mipsi64r2sf**), you must first obtain the Cavium SDK directly from Cavium Networks. Information is available at the following URL:

http://cavium.com/processor_octeon_software_develop_kit.html

Further, the Cavium BSP does not support the Wind River Compiler. To build this BSP, use the GNU compiler.

ColdFire BSPs

The ColdFire BSPs (**m54x5evb** and **m5329evb**) do not support the GNU compiler. To build these BSPs, use the Wind River Compiler.

IP Prefix

The prefix "IP" appears throughout the code base and in many component names and macros. These instances are artifacts of Wind River's acquisition of Interpeak AB in 2006, and the subsequent integration of their technology into the General Purpose Platform.

Target Server File System: TSFS

The Target Server File System (TSFS) is not designed for use with large files (whether application executables or other files), and performance may suffer when they are greater than 50 KB. For large files, use FTP or NFS instead of TSFS.

BSP Reference Instructions

The **target.ref** files for some BSPs have instructions describing how to write a boot loader image into the target flash memory. The instructions may contain references to legacy tools, such as visionCLICK, visionICE, or Tornado. Their equivalent counterparts are Wind River Workbench for On-Chip Debugging, Wind River ICE, and Wind River Workbench, respectively. Most of the parameters, such as **start address**, **offset** and **bias**, can be applied directly to the new tools as well as the old ones. Make adjustments as necessary when following the instructions.

Treatment of Invalid Compiler Command-Line Options

In most cases, passing an unrecognized option flag to the tools generates a warning. This behavior, however, is not completely consistent. In some cases, no warning is generated; in other cases, such an invalid option may cause compilation to stop.

Limited Support for RTA Profiling

VxWorks supports one compiler option, **-Xrtc**, for generating Run-Time Analysis (RTA) profiling information. Other compiler profiling options, such as **-Xprof**, are not supported in VxWorks.

For **-Xrtc**, all mask options are supported except *mask* = 0x4.

Backward Compatibility

VxWorks and OS-related run-time components are backward compatible. When recompiled, socket-based applications are generally backward compatible. Other applications may need to be migrated. For more information on backward compatibility and migration, see the *Wind River General Purpose Platform, VxWorks Edition Migration Guide*.

BSP Migration

EXTRA_INCLUDE defines with the following format should not be used in BSP makefiles:

```
EXTRA_INCLUDE = $(path)
```

The **EXTRA_INCLUDE** define may already be defined coming into your makefile. To add directories to your include path, use the following:

```
EXTRA_INCLUDE += $(path)
```


Floating-Point Variables

Some code makes use of floating-point variables. Any task that uses code that makes use of floating-point variables should be spawned using the **VX_FP_TASK** task option.

License Usage Reporting

The license usage reporting tool **wrsReport** must be run from the machine on which it is installed. You cannot run this utility remotely from another machine. For more information about license usage reporting, see the Wind River Site Configuration Setup Guides at the following URL:

http://www.windriver.com/support/site_configuration

Mixing C++ Output from Different Compilers

Because of ABI differences, C++ object files and libraries created by different compilers are not compatible and cannot be linked into a single executable. When building a project, make sure all C++ modules are produced by the same toolchain (Wind River GNU Compiler or Wind River Compiler).

Shell Commands

Your Platform provides many shell commands that work only with the command interpreter. To switch from the C-interpreter to the command-interpreter, enter **cmd** at the shell prompt.

Online Documentation with Multiple Workbench Installations

It is best to install multiple versions of Workbench in different installation directories. If you have multiple installations in the same installation tree:

- Help Contents displays help for only the most recent installation.
- Help Search searches only the most recent installation.
- Context-sensitive help returns results from only the most recent installation.

If you have installed multiple versions of Workbench in the same installation directory, documentation for earlier installations is still available. Use a Web browser to access the **index.html** file (**libIndex.html** or **rtnIndex.html** for API documents) in the lowest level of the documentation directories (below *installDir/docs*). PDF documents for the previous installations, where provided, are also still available in the documentation tree.

Checksum Offloading

The driver interface for hardware checksum offloading has not changed for this release. That is, no changes need to be made to your driver code to support hardware checksum offloading if the driver was implemented for an earlier VxWorks 6.x release. However, checksum offloading support is no longer included in the network stack by default. To enable checksum offloading, you must include the `IPCOM_USE_HW_CHECKSUM` build macro when building your Wind River Network Stack source code. This support can be included by uncommenting the definition for `IPCOM_USE_HW_CHECKSUM` in the `ipcom_pconfig.h` file (located in `installDir/components/ip_net2-6.6/ipcom/port/vxworks/config`).

For more information on building network stack source code, see *Wind River General Purpose Platform, VxWorks Edition Getting Started*. For more information on checksum offloading, see the *END Ethernet Drivers* chapter of the *VxWorks Device Driver Developer's Guide, Volume 3*.

Breakable Tasks

Some tasks are set to be breakable by default. In some cases, this may cause issues when debugging using global breakpoints. To set tasks to unbreakable, use `taskOptionSet()`, which is implemented in `installDir/vxworks-6.6/target/src/wind/taskInfo.c`. (WIND00092231)

4.4 Documentation Errata

This section lists documentation errata for the General Purpose Platform documentation set as a whole. For a list of documentation errata for a specific product, see the product-specific chapter in this document.

Product Uninstallation

The process for uninstalling Wind River products has changed from that described in the Wind River Site Configuration Setup Guides.

When you uninstall the products in your Platform, Wind River recommends that you use the new maintenance tool provided with this release. Unlike previous tools, the new maintenance tool can uninstall many products at the same time.



NOTE: The new maintenance tool uninstalls all Wind River products from this release forward; it does not uninstall products from previous releases.

4

To launch the uninstallation tool, do the following:

- **From the Windows GUI**

Select **Start > All Programs > Wind River > Uninstall and Maintenance > Maintenance Tool**.

- **From the Linux and Solaris GUI**

Select **Applications** (the main menu on the panel) > **Wind River > Uninstall and Maintenance > Maintenance Tool**.

- **From the Windows command line**

On Windows, execute the following commands from a command prompt:

```
% cd installDir\maintenance\mtool
% mtool.exe
```

- **From the Linux command line**

On Linux, execute the following commands from the command shell:

```
% cd installDir/maintenance/mtool
% ./mtool_linux
```

- **From the Solaris command line**

On Solaris, execute the following commands from the command shell:

```
% cd installDir/maintenance/mtool
% ./mtool_solaris
```

Wind River Run-Time Analysis Tools

Documents delivered with your Platform may contain cross-references to ScopeTools products or documentation. The entire suite of Wind River Run-Time Analysis Tools was renamed in this release. The new product names are as follows:

Previous Name	New Name
ScopeTools	Run-Time Analysis Tools
CoverageScope	Code Coverage Analyzer
StethoScope	Data Monitor
TraceScope	Function Tracer
MemScope	Memory Analyzer
ProfileScope	Performance Profiler

Wind River Workbench

- The term *Kernel Editor* has been changed to *Kernel Configuration Editor*. There are various forms used throughout the documentation set distributed with your Platform.
- Some documents delivered with your Platform explain how to locate components in the Kernel Configuration Editor by right-clicking the **Components** tab and selecting **Find Component**. This menu entry, and the dialog it opens, have been renamed **Find**. For details on using **Find**, see the *Wind River Workbench User Interface Reference: Kernel Configuration Editor*. Note that the *Kernel Configuration Editor* chapter was previously called *Kernel Editor View*.
- Documents delivered with your Platform may contain cross-references to Workbench documentation. Some of the Workbench documents have been reorganized as follows, resulting in incorrect cross-references:
 - In the *Wind River Workbench User Interface Reference*, the *Kernel Editor View* chapter has been renamed *Kernel Configuration Editor*.
 - Build console and build properties content has been removed from the *Wind River Workbench User's Guide* and added to the *Wind River Workbench User Interface Reference* in chapters called *Build Console View* and *Build Properties Dialog*.
 - Information about building projects is now located in *Wind River Workbench User's Guide: Building Projects*.

Wind River Network Stack

Documents delivered with your Platform may contain cross-references to the *Wind River Network Stack for VxWorks 6 Programmer's Guide*. The Network Stack documentation has been redistributed into the following guides, resulting in

incorrect cross-references. The cross-references should point to one or more of the following guides:

- *Wind River Network Stack for VxWorks 6 Programmer's Guide, Volume 1: Transport and Network Protocols*
- *Wind River Network Stack for VxWorks 6 Programmer's Guide, Volume 2: Application Protocols*
- *Wind River Network Stack for VxWorks 6 Programmer's Guide, Volume 3: Interfaces and Drivers*

Wind River IPsec and Wind River IKE

Documents delivered with your Platform may contain cross-references to the *Wind River IPsec and IKE for VxWorks 6 Programmer's Guide*. The Wind River IPsec and Wind River IKE documentation is now provided in two separate guides, resulting in incorrect cross-references:

- Information about IPsec is now located in the *Wind River IPsec for VxWorks 6 Programmer's Guide*.
- Information about IKE is now located in the *Wind River IKE for VxWorks 6 Programmer's Guide*.

Directory Paths

The directory paths in some guides that have not been updated for this release are incorrect. In this release, Wind River directory paths generally begin with *installDir/vxworks-6.6*. Some guides may incorrectly indicate this path as *installDir/vxworks-6.1*, *installDir/vxworks-6.2*, *installDir/vxworks-6.3*, *installDir/vxworks-6.4*, or *installDir/vxworks-6.5*.

Release Notes

Some documents delivered with your Platform indicate that these release notes are shipped with your product. This is incorrect. *Wind River VxWorks Platforms Wind River General Purpose Platform, VxWorks Edition Release Notes* are available from the Online Support Web site:

<http://www.windriver.com/support/>

5

Wind River Workbench

- 5.1 Introduction 37
- 5.2 Changes in This Release 38
- 5.3 Usage Caveats 51
- 5.4 Known Problems 62
- 5.5 Documentation Errata 74

5.1 Introduction

Wind River Workbench 3.0 is an Eclipse-based development suite that facilitates creating and building projects, establishing and managing host-target communications, and developing, debugging, and monitoring operating system kernels as well as device software applications running on a simulator or a target.

For tutorials demonstrating many Workbench features, see the *Wind River Workbench User's Guide*.

5.2 Changes in This Release

Workbench is the successor to Tornado, SNIFF+, visionCLICK, SingleStep, and WIND POWER IDE. It incorporates many capabilities that formerly required external tools, such as target management and object browsing.

Wind River has improved and updated Workbench with this release, as described in [5.2.1 Enhancements](#), p.39. Workbench also adopted several Eclipse projects (C/C++ Development Toolkit 4.0, Target Management 2.0, and Device Debugging 0.9), including replacing some proprietary views with their Eclipse counterparts. To see the old views side-by-side with the new views, see the *Wind River Workbench User's Guide: What's New in CDT, DD, and TM*.

Though functionality remains essentially the same, these Eclipse projects bring with them some differences in workflow and user interface.

Change	Look here for details
Adoption of the CDT Indexer for static analysis.	Static Analysis Now Uses the CDT Indexer , p.42
Indexer is configurable, but has limitations.	Indexer-Specific Exclusion Filters , p.43 Reindexing a Subset of Project Files , p.43 Macro Limitations , p.72 C++ Limitations , p.72 Call Tree Limitations , p.73 External APIs Cannot Be Configured , p.73
Debug and static analysis symbols are now displayed separately.	Two Modes of Static Analysis Have Been Separated , p.42
User build arguments field moved.	Build Console Enhancements , p.43
Workbench Editor is now the CDT Editor.	Workbench Editor Replaced by CDT C/C++ Editor , p.43, Ada Editing No Longer Supported , p.66, and ASM Symbols Not Available , p.66
Project Navigator is now the Project Explorer.	Project Navigator Replaced by Eclipse Project Explorer , p.43
Target Manager is now the Remote Systems view.	Target Manager Replaced by Eclipse Remote Systems View , p.44

Change (cont'd)	Look here for details (cont'd)
The Memory, Registers, Watch, and Local Variables views are replaced by Eclipse equivalents.	Several Workbench Views Replaced by Eclipse Views , p.45

5.2.1 Enhancements

5

Wind River Workbench

Workbench 3.0 is based on the Eclipse 3.3 platform release, and supports VxWorks 6.6. Workbench includes the following enhancements.

Project and Build Systems

SMP Support

In this release, Workbench provides support for the optional VxWorks SMP product, but this support is only available for some BSPs.

Workbench includes prebuilt VxWorks images that include SMP support. These images are for evaluation purposes only, and cannot be built. In order to experiment with VxWorks SMP without SMP hardware, SMP images of simulators (VxWorks simulator) for Linux, Solaris, and Windows are provided.

VxWorks Boot Loader Project Is Now VxWorks Boot Loader/BSP Project

VxWorks Boot Loader projects now display a check box that allows you to copy the files for a selected BSP into your project. This gives you the opportunity to modify and build them into a custom BSP without altering the VxWorks installation tree.

New Projects Use Flexible Build System

Newly created projects will use the flexible build system developed in previous versions of Workbench. If necessary, you can create projects that use standard builds by enabling that option in the Build preferences.

Projects Created with Previous Versions of Workbench Are Updated on Import

When projects created with previous versions of Workbench are imported into this version of Workbench, they will be migrated to function properly with the new set

of plug-ins in Workbench 3.0. In addition, all import wizards that previously created standard managed build projects now automatically create flexible managed build projects.

Performance Improved for VxWorks Image Projects (VIP)

Workbench performance has significantly improved when adding or removing many files from a VIP in a single operation.

Build Specs Are Now Shipped with VxWorks, Not Generated After Installation

Build specs are now shipped as part of the VxWorks distribution, rather than being generated after the product is installed.

This means that if you change any makefile fragments in the VxWorks installation, you need to regenerate the cache information before you can use the **Restore Defaults** option in the Preferences (**Window > Preferences -> Wind River -> Build -> Build Properties**) to use the changed settings in your Workbench projects.

To regenerate the cache, use the following commands:

```
% cd installDir/vxworks-6.x/setup
% vx_postinstall.bat (on Windows) or vx_postinstall.sh (on Linux or Solaris)
```

Binary Image Parser

You can now parse binary image files by double-clicking them in the Project Explorer, not just by running **ddump** or **objdump** from the command line. Workbench parses the file using the appropriate tool, and displays the output in the Editor.

Parameters for **ddump** and **objdump** can be modified globally by selecting **Window > Preferences > Binary Parser**, or selectively for a given project by right-clicking it in the Project Explorer and selecting **Properties**.

Target Management

Target Server Is Not Forward-Compatible

To accommodate 64-bit Linux and SMP VxWorks targets, changes were made to the WTX protocol. This makes Workbench 3.0 target servers incompatible with tools shipped with previous versions of Workbench.

Debugger

Debugger Enhancements

The debugger has been improved, and now includes support for the following:

- SMP task mode and system mode debugging for simulators and real targets.
- Wind River Compiler 5.6 and Wind River GNU Compiler 4.1.
- Debugging of VxWorks 5.5.x targets on Solaris hosts.
- Accessing thread-local storage (TLS) variables.
- New CPU architectures including:
 - ARM Thumb-2
 - Cavium OCTEON
 - ColdFire v3
 - MIPS 74kf and 24kf
 - RMI XLR
- Linux 64-bit native host application debugging for 64-bit Red Hat Workstation 5 hosts.
- Expression evaluation improvements including `w_char*` casting and pointer to array casting.

Pin and Clone

Many views, including the Expressions, Registers, Memory, and Build Console, provide a **Pin to Color** option for use while debugging multi-core targets.

If you are connected to a multi-core target, you can use the Debug view to assign a color to each core, and then select **Pin to Color** from the drop-down menu in one of these views to pin the current instance of the view to that color.

For example, for a dual-core target, you can use the **New View Instance** button (some views provide a **New *viewName* View** button instead) to open two instances of the view, then use the **Pin to Color** option to pin one view to core A and the other to core B.

Launch Configuration

Specifying a Working Directory for a Process

While creating a **Process on Target** launch, you can accept the default working directory, or specify a different directory if necessary.

Host Shell

Command and command output logging to a file is supported.

Static Analysis

Static Analysis Now Uses the CDT Indexer

Workbench source code parsing is now done by the CDT Indexer, which is comparable in speed to the Workbench parser it replaces.

The Indexer also provides build output parsing: if **Enable project specific settings** is unchecked, the output of the build is scanned to set up the sources for indexing. Only the sources that are part of the build process will be indexed, and all flags (defines, includes) used by the build will be used for the indexing.

Indexer performance is affected by preferences settings, such as whether indexing of type references is turned on or off. It is on by default, but turning it off speeds up the index. When turned off, it is still possible to navigate typedefs, but C/C++ search for references will not work.

Two Modes of Static Analysis Have Been Separated

In this release the two modes of the Static Analysis Symbol Browser (debug and static analysis) have been separated. Now debug symbols are displayed in the Debug Symbol Browser, and static analysis symbols appear on the **C/C++ Search** tab of the Search view and in the **Navigate > Open Element** dialog.

The Open Element dialog (**SHIFT+F3**) functions in a similar fashion to the static symbol browser that it replaces. Symbol matches are listed as you begin typing, and you can navigate to the desired symbol by clicking on it.

The C/C++ Search Dialog (**CTRL+H**, then select the **C/C++ Search** tab) works in a similar fashion, except that it produce a list of search results in a separate view after you initiate the search. You can then navigate from this view.

Shared Symbol Data Workflow Change

In Workbench 3.0, shared data is copied into the workspace when a project created in a different workspace is imported. After that, the data is treated the same as if it had been indexed by that particular workspace.

The import itself is automatic as long as the data was exported properly using **Export > C/C++ > Team shared index**.

Indexer-Specific Exclusion Filters

It is now possible to configure indexer-specific exclusion filters using regular expressions. Select a project, choose **Properties > C/C++ General > Paths and Symbols > Source/Filters**, then edit the filter data and add a regular expression.

Reindexing a Subset of Project Files

It is possible to initiate the reindex of a subset of files in a project, updating only those files that have been modified, or all files. Updating of modified files happens automatically when they are saved, but can also be initiated for example when a header file is touched.

Views

Build Console Enhancements

The User Build Arguments field, which previously appeared at the top of the Project Navigator and provides a way to temporarily pass additional arguments to the make command, now appears in the Build Console view toolbar. Also included on the toolbar is a **Run Last Build Again** button, with a drop-down list containing previous builds.

CTRL+click on build console entries to format the build output.

Workbench Editor Replaced by CDT C/C++ Editor

The Workbench Editor has been replaced by the CDT C/C++ Editor, which has many of the same features such as code folding, code completion, parameter hinting, and syntax highlighting.

Some new features in the Editor include a quick outline dialog available by pressing **CTRL+O**, and the ability to correct indentation by selecting badly-formatted lines and pressing **CTRL+I**. The Editor context menu contains these entries, as well as many others including one for creating breakpoints directly in the Editor.

However, the syntax coloring is different (it is similar to the Java editor). Coloring is configurable in the **C/C++ > Editor > Syntax Coloring** preferences.

Project Navigator Replaced by Eclipse Project Explorer

The Workbench Project Navigator has been replaced by the Eclipse Project Explorer, which displays projects from other CDT sources alongside your Workbench projects and provides additional access points to sources.

Right-clicking a binary under the **Binaries** node launches and/or debugs that file; double-clicking parses it and displays the output from the binary in the Editor. Double-clicking a file under the **Includes** node displays header files. The **Build Targets** node provides a dialog that displays the detailed setup of each individual build target. For more information about the Project Explorer, see the *C/C++ Development User Guide*, available from the Workbench help system.

The Project Explorer does not support modifying project hierarchies and build target contents by dragging and dropping, as was possible in the Project Navigator. Relationships between projects can be modified by selecting **Project References > Add as Project Reference**.

The Project Navigator can be opened, if desired, by selecting **Window > Show View > Project Navigator** from the Workbench toolbar.

Target Manager Replaced by Eclipse Remote Systems View

The Target Manager has been replaced by the Eclipse Remote Systems view. This view provides a Local node for access to the local file system and shell, and allows you to access remote file systems as if they are local, including remote command lines as well.

Switching to the Remote Systems perspective displays the Remote Systems Details view, which provides additional information about each defined target connection.

For more information about this view, see the *RSE User Guide*, available from the Workbench help system.

Terminal View Enhancements

In its default mode, the Terminal view relies on the connected remote system for text editing. This works fine when the remote system supports any kind of terminal emulation. Simple systems like boot loaders, however, often do not provide good text editing capabilities.

In order to address this for Workbench 3.0, the Terminal view now includes a **Toggle Command Input Field** button, which opens a text inset field at the bottom of the Terminal view. This separate field allows full editing capabilities inside Workbench. The full contents of the text inset field are sent to the target when you press **ENTER**. It also provides a history of previously sent commands, when you press the cursor up or down buttons.

To hide the text inset field, click the **Toggle Command Input Field** button again.

Several Workbench Views Replaced by Eclipse Views

This table shows several Workbench views that have been replaced by their Eclipse counterparts. For more information about these views, see the *C/C++ Development User Guide*, available from the Workbench help system.

Workbench View	Eclipse View	Changes
Watch	Expressions	Drag & drop to this view is supported. Casting pointers is supported. Cannot set radix for a single element. No more Expand All capability.
Local Variables	Variables	Cannot set radix for a single element. No more Expand All capability.
Memory	Memory	Tabs are now monitors. Renderings are different ways to look at the memory; currently only one is installed with Workbench. Find and Replace are available, though slow over large range. SNF import/export are no longer supported. No S-record format for 64-bit import/export. Drag & drop to Memory view is not supported.
Registers	Registers	Now contains OCD extensions. The Properties view was removed; bit fields are now editable in place. The Details pane (at bottom of view) displays register value in each radix.

Kernel Configuration Editor

In Workbench 2.x it was not possible to exclude components from a VxWorks kernel configuration if they became invalid due to external changes of the configuration. In Workbench 3.0, such components can now be excluded in the Kernel Configurator.

Getting Started Resources

Upon startup, Workbench displays a new Getting Started Resources view containing links to the Wind River Online Support site, Workbench documentation, and other useful resources. Clicking some of these links displays content in Workbench, clicking others opens an external browser.



NOTE: Workbench cannot open an external browser on Solaris, so the Getting Started Resources links may not function properly on that host.

RSS Feed View

The RSS Feed view provides quick links to Workbench patches and other critical information.

Preferences

Preferences that formerly were displayed just below the root node (such as **Build**, **Run/Debug**, and **Static Analysis**) are now displayed under the package that provides those preferences.

For example, all Workbench-specific preference settings are now located under a root node called **Wind River**. To view them, select **Window > Preferences > Wind River**. Preferences that come from CDT are displayed under the **C/C++** root node.

Documentation

New Cheat Sheets Available

Commonly used procedures, tips, and other useful information is available by selecting **Help > Cheat Sheets** from the Workbench main toolbar.

Eclipse

Java Development Tools (JDT) Dependency

Previous versions of Workbench included a modified version of JDT, but to better integrate with Eclipse and other third party packages this has been removed. However, some third party plug-ins have a dependency on JDT.

If a plug-in you are interested in requires JDT, you should download it from the Eclipse Download Center at <http://www.eclipse.org>. You can also install it from within Workbench using the Update Manager, as described in *Wind River Workbench User's Guide: Integrating Plug-ins*.

Wind River Run-Time Analysis Tools

This section describes the enhancements and improvements to the Run-Time Analysis Tools suite (formerly ScopeTools) provided with this release.

All Tools

The entire suite of Wind River Run-Time Analysis Tools (formerly ScopeTools) was renamed in this release. The old names and corresponding new names are shown in [Table 5-1](#).

Table 5-1 Renaming of Run-Time Analysis Tools

Previous Name	New Name
ScopeTools	Run-Time Analysis Tools
CoverageScope	Code Coverage Analyzer
StethoScope	Data Monitor
TraceScope	Function Tracer
MemScope	Memory Analyzer
ProfileScope	Performance Profiler

Support is extended to the latest stable version of the 4.x gcc toolchain (**gcc4.1.2**) for all supported architectures, on both VxWorks and Linux targets.

Code Coverage Analyzer

The Code Coverage Analyzer has been integrated to run completely from within Workbench in this release. The previous standalone window version has been eliminated. This required the following major changes:

- Coverage instrumentation integrated into the Workbench project manager.
- Provided settings for coverage on/off and types on a per-project basis.
- Provided manual and automated upload of coverage data for offline viewing.
- Provided dynamic display of coverage data in Workbench.

As a result, the implementation of most functionality has changed.

- The procedure to invoke the coverage instrumentor from Workbench has changed. Previously set in the **Preferences** dialog box, it is now set (per

project) by right-clicking your project node, selecting **Build Options** in the pop-up menu, and selecting **Build with Code Coverage Analyzer** in the next level pop-up menu.

- The procedure previously used to delete data files has been replaced by the following steps:
 - a. Open the **report** node in your project tree in the **Project Explorer** view.
 - b. Right-click the data file you want to delete to open the pop-up menu.
 - c. In the pop-up menu, select **Delete** to delete the file.

This change results from the integration of the Code Coverage Analyzer into Workbench.

- With this release, the previous CoverageScope .prj project concept has been merged into Workbench to become Workbench projects. This means that a .prj project (created with any previous release) can no longer be imported and used by the Workbench 3.0 release. Only Workbench projects will be recognized as having been instrumented with the Code Coverage Analyzer. Be sure to reinstrument and compile your Workbench project before running with the Code Coverage Analyzer if it was created using a previous version of Workbench.

Consult the *Wind River Workbench Code Coverage Analyzer User's Guide* for details.

Data Monitor

The Windows and UNIX versions of the former StethoScope User's Guides have been merged into one document: *Wind River Workbench Data Monitor User's Guide*. Strong attempts were made to preserve clarity and avoid ambiguity while combining similar descriptions wherever it was feasible. However, separate descriptions appear in places where combinations would have yielded unclear understanding. The 731 pages of separate documents were combined into 391 pages.

Function Tracer

A maximum of 32 has been set, and is enforced, for the number of trace points that can be selected.

Consult the *Wind River Workbench Function Tracer User's Guide* for details.

Memory Analyzer

The previous Memory Analyzer standalone window version has been eliminated.

The Time and Fragmentation views, previously available only in the standalone version, are now available as perspective views, along with the other Memory Analyzer views. However, the Fragmentation view is only available for a VxWorks target.

The capacity of the Memory Analyzer tool has been increased using a database that stores allocation and free events from the target. This persistent data storage file is now created by every Memory Analyzer session. Collected data is streamed and saved into the file automatically. This data stream can be displayed in the GUI in real time, or at any later time. This design provides for automatically saving data without having to explicitly select an option, but it also creates an impact on disk usage.

Consult the *Wind River Workbench Memory Analyzer User's Guide* for details.

Performance Profiler

The previous Performance Profiler standalone window version has been eliminated.

Consult the *Wind River Workbench Performance Profiler User's Guide* for details.

VxWorks SMP Mode Support

Support has been added for VxWorks targets running in symmetric multiprocessing (SMP) mode, including MIPS, Power Architecture (PowerPC), and x86 targets.

Wind River System Viewer

Log Viewer

Log Viewer opening time has been greatly reduced.

Support for Multi-Core Systems

New Log Viewer functionality: Visualization of the behavior of multi-core systems, and per-core filter and search facilities.

New Analysis Suite Views: These graphical and tabular representations of various types of log file analyses are accessed from the System Viewer Configuration editor. The views include analyses of CPU usage (aggregate and per core), system load, and per-core ready and running states.

New 'Memory Read' Upload Method

This log file upload method is, unlike the other existing methods, host-driven. The host reads data directly from the target's memory and then processes the log structure. Some advantages are: Log transfer is not affected by task starvation, multiple log transfers are possible, transfer takes place without needing to call target functions.

New Core Dump File Upload Feature

Using the new Memory Read upload method, it is now possible to upload logs from VxWorks core dumps.

Wind River Workbench Projects Are Now Automatically Created for Log Files

This facilitates maintenance and access to existing logs.

5.2.2 Fixed Problems

For a list of problems fixed in Wind River Workbench, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

5.2.3 Deprecated Features

Standard Managed Build

Standard managed builds have been deprecated and replaced by managed builds (formerly known as *flexible managed builds*).

If your build structure is similar to the file system structure, and you prefer standard managed builds, you can enable them by selecting **Preferences > Wind River > Build** and checking the box for **Enable deprecated standard managed build**. Note that you cannot change a project's build type once it is created.

5.2.4 Unsupported Features

Exporting APIs

In previous versions of Wind River Workbench, it was possible to both import external APIs and export APIs when you build a project. API importing is unchanged, but the ability to export APIs is no longer available.

5.3 Usage Caveats

Workbench Issues

Workbench Supports US English Language Only

Workbench supports and is tested only with US English. Support for other languages is not available at this time.

-RedirectIO Flag Is Now Obsolete

In Tornado, you needed to use the **-redirectIO** flag with the Target Server to redirect I/O to the Wind Shell. In Workbench, the **-redirectIO** flag has been made obsolete, and I/O is now redirected to the wtxConsole.

Workbench 2.6.1 Registry Does Not Support Workbench 3.0

The Workbench 3.0 registry will support running Workbench 2.6.1, but the reverse is not true. So if you launch Workbench 2.6.1, then close it and start Workbench 3.0 without shutting down the running registry, an error will appear.

To get around this, always shut down the Workbench 2.6.1 registry before starting Workbench 3.0.

Multiple Users and Installations

Different configurations of the number of Workbench users and the number of Workbench installations on a single host are possible, and the following considerations apply.



NOTE: Each user should work in a user-specific workspace. Sharing of workspaces is not possible. To specify a workspace, either use the **-data** startup option, select a workspace in the Workspace selection dialog during startup, or select **File > Switch Workspace** in Workbench.

Single User with Single Installation

This configuration presents no special installation or use considerations, assuming the users on each host perform their own installation and have standard access permissions to the installation location. No special startup arguments are necessary and the default workspace may be used.

Multiple Users with Multiple Installations

The same conditions apply here as they do for a single user with a single installation, with the exception that a single administrative user often performs the different Workbench installations. In this case it is important that the proper permissions be granted to each user for access to their particular installation. Linux and Solaris root users performing multiple installations should refer to the **umask** and **chmod** manual (**man**) pages.

Multiple Users with a Single Installation

Multiple users can share a single Workbench installation as long as the access rights allow them to read all files of the installation (same group as the user who installed Workbench). Linux and Solaris users should refer to the **umask** and **chmod** manual (**man**) pages.

For performance reasons, it is desirable to have the workspace on a local file system. Some Eclipse-specific data is stored in the user's home directory by default. If this is not desired because of slow network access, use the **-configuration** startup option to redirect this data. Refer to *Eclipse Workbench User Guide: Running Eclipse* in the online help for more information on startup options.

Single User with Multiple Installations

If a single user is installing Workbench more than once, it is important that the configuration area (**%USERPROFILE%\workbench-3.x.build_ID** in Windows and **\$HOME/.workbench-3.x.build_ID** in UNIX) does not get corrupted. Different versions of Workbench will not conflict, but for multiple installations of the same version for the same user, different configuration areas should be specified at startup with the **-configuration** option.

Eclipse Team Features

Workbench supports all the team features of the standard Eclipse installation as documented in the online help supplied with Workbench.

Workspace Location

Wind River strongly recommends placing your workspace on a local file system. Remote file systems can cause significant performance problems. If you use IBM ClearCase, place the workspace outside a dynamic view, otherwise you will face significant performance problems.

Source Build Option for VxWorks Scalability Projects

Use of the source build option with any profiles other than those listed below, or with any BSPs other than those listed below, is not supported. Adding components that are not part of one of the profiles below may cause the system to revert to the binary build, thus negating the scalability benefits of the profiles.

Profiles

- Minimal VxWorks Kernel Profile
- Basic VxWorks Kernel Profile
- Basic VxWorks OS Profile

BSPs

- `integrator1136jfs`
- `wrSbcPowerQuiccII`

Turning off Static Analysis for Large Projects

Static analysis is turned on by default. For large projects (such as kernel build projects) static analysis can slow down Workbench functionality considerably.

To turn off static analysis:

For existing projects:

Right-click the project and select **Properties**, then select **C/C++ General > Indexer**. From the **Select Indexer** drop-down list, select **No Indexer**. Click **OK** to close the dialog.

For new projects:

Select **Window > Preferences** and choose **C/C++ > Indexer**. From the **Select Indexer** drop-down list, select **No Indexer**. Click **OK** to close the dialog.

By project:

If you would prefer not to turn off static analysis for all new projects, you can turn it off for a particular project as you create it. Click **Next** in the New Project wizard until you reach the Indexer page, then select **Enable project specific settings**. From the **Select Indexer** drop-down list, select **No Indexer**, then click **Finish** to create the project.

Supported Make System

The Workbench make system is based on features of the GNU Make utility (**gmake**). Other make systems are not supported.

Increasing Virtual Memory

Building large applications can cause Java to run out of memory. To avoid this, increase the size of memory for the Java virtual machine to 512 MB or higher, either manually (when you launch Workbench from the command line) or by editing a launch command.

From the command line:

On Windows:

From a shell, type the following:

```
C:\> cd installDir\workbench-3.x\wrwb\platform\eclipse  
C:\> .\wrwb-x86-win32.exe -vmargs -Xmx512m
```

On Linux and Solaris:

Use the values as parameters to the **startWorkbench.sh** script:

```
% ./startWorkbench.sh -vmargs -Xmx512m &
```

where 512 corresponds to 512 MB of RAM for the virtual machine.

From the Windows Start menu:

1. Select **Start > Programs > Wind River > Workbench 3.x**, then right-click **Wind River Workbench 3.x** and select **Properties**.
2. From the **Shortcut** tab, move the cursor to the end of the command in the **Target** text box.

3. After `wrb-x86-win32.exe`, type `-vmargs -Xmx512m`.
4. Click **OK**.

Each time you launch Workbench using the **Start** menu, you will allocate the appropriate amount of memory for the Java virtual machine.

From a desktop shortcut:

Right-click the shortcut, select **Properties**, and edit the **Target** command line as described above.

Accessing Online Support from the Workbench Welcome Page

On Windows

If Internet Explorer is installed on your system, you must enable cookie support for the Internet Explorer ActiveX control before you can access the VxWorks 6.x Release Web site from the Workbench Welcome page. If your host machine is configured not to accept cookies for other sites, follow these steps to override:

1. From the Windows **Start** menu, select **Settings > Control Panel > Internet Options**.
2. Select the **Privacy** tab.
3. Under **Web Sites**, click **Edit**.
4. Under **Address of Web site**, type `secure.windriver.com`, click **Allow**, then click **OK**.

You should now be able to access the Online Support login page.

If you have removed Internet Explorer from your system, Workbench opens your default browser and your regular cookie settings apply.

On Linux and Solaris

Workbench opens your default browser. Your regular cookie settings apply.

Viewing Workbench Help and Other Documentation on Solaris

To view Workbench documentation in Netscape on Solaris, you must adjust your Workbench preferences to run Netscape (the default browser is Mozilla):

1. From within Workbench, select **Window > Preferences > Help > Custom Browser Command**, then enter the full path to Netscape:

`/usr/dt/bin/netscape`

If Netscape is already in your path, you can type just **netscape**.

2. Netscape asks for a cookie by default. Accept the internal cookie.

You can now view the Workbench documentation in Netscape, either context-sensitive help from Workbench itself (by pressing the **CTRL+F1** and selecting a link from the help view) or by navigating to it from **Help > Help Contents > Wind River Documentation**.



NOTE: The **Help** button on Solaris keyboards does not open Workbench help due to a problem in Solaris/GTK+. Instead, use **CTRL+F1** to access help.

Using the Help Browser Independently of Workbench

For a full-featured help browser that runs independently of a particular instance of Workbench, create a script with the commands given below, and then start it in the proper environment for your operating system.

On Windows

1. Open a VxWorks Development Shell by selecting **Start > Programs > Wind River > VxWorks 6.x and General Purpose Technologies > VxWorks Development Shell**.



NOTE: Starting this shell sets the appropriate environment variables for you.

2. Create and save a batch file with a descriptive name (such as **showhelp.bat**) with the following content. Replace *installDir* with your Workbench installation directory path and *3.x* and *version* with the appropriate version number.

```
echo off
echo Please close this shell with CTRL+C when you have finished browsing
the documentation.
java -classpath installDir\workbench-3.x\wrwb\platform\eclipse\plugins\org.ec
lipse.help.base_version.jar org.eclipse.help.standalone.Help -command
displayHelp /com.windriver.ide.doc.globals/toc.xml -eclipsehome installDir\wo
rkbench-3.x\wrwb\platform\eclipse -noexec
pause
```
3. Run your new batch file in the shell; it may take a minute or two for the help browser to appear.

4. When you are finished browsing the documentation, close the help browser window, then press **CTRL+C** in the shell. When asked if you want to stop the batch file, press **Y** for yes.
5. At the shell prompt, type **exit** to close the shell.

On Linux/Solaris

1. In a terminal window, create and save a script with a descriptive name (such as **showhelp.sh**) with the following content. Replace *installDir* with your Workbench installation directory path, and *3.x* and *version* with the appropriate version number.

```
#!/bin/sh

echo Please close this shell with CTRL-C when you have finished browsing

exec installDir/jre/version/x86-linux2/bin/java -classpath
installDir/workbench-3.x/wrwb/platform/eclipse/plugins/org.eclipse.help.base_
version.jar org.eclipse.help.standalone.Help -command displayHelp
/com.windriver.ide.doc.globals/toc.xml -eclipsehome
installDir/workbench-3.x/wrwb/platform/eclipse -noexec pause
```



NOTE: The **exec** command shown above is all one line to the final **pause**.

2. Save the script and make it executable:
3. Copy the script to your installation directory:
4. \$ **cp showhelp.sh installDir**
5. Change to your Workbench installation directory.
6. Run the **wrenv** command to initialize your Wind River environment:

```
$ ./wrenv.sh -p workbench-3.x
```

7. Run your new script in the shell; it may take a minute or two for the help browser to appear.

```
$ ./showhelp.sh
Please close this shell when you have finished browsing
<<<press CTRL-C when you are done with the help browser>>>
```

8. When you are finished browsing the documentation, close the help browser by entering **CTRL+C** in the terminal window. When asked if you want to stop the batch file, press **Y** for yes.
9. At the shell prompt, type **exit** to close the shell.

Viewing Individual Wind River Documents in a Browser

It is possible to view a single document at a time using the HTML files in *installDir\docs\extensions\eclipse\plugins*. This does not require that you have Workbench running, nor must you run a batch file.

For example, to view the *Wind River Workbench User's Guide, 3.0 (VxWorks Version)*, enter the following URL into your browser, substituting your Workbench installation path for *installDir*:

```
file://installDir/docs/extensions/eclipse/plugins/com.windriver.ide.doc.wr_workbench_vxworks/wr_workbench_vxworks_users_guide_3.0/html/index.html
```



NOTE: This method does not provide all the features of the help browser provided by Workbench help, such as full-text search, bookmarks, and a fully expandable table of contents for the whole documentation set.

Setting the Number of Open Files in Editor

Workbench allows you to open as many files as you like in the Editor.

To limit the number of files you have open at one time, select **Window > Preferences > General > Editors**, and select **Close editors automatically**. Increase or decrease the number in the **Number of opened editors before closing** field, then click **OK**. This causes Workbench to close one of the open files as soon as you try to open another file that exceeds your set limit.

If you only want to keep those files open that you are particularly interested in, set the **Number of opened editors before closing** field to **1**, causing Workbench to close each file as it opens the next one. For those files you want to keep open, click the **Pin Editor** icon on the main Workbench toolbar. Workbench keeps that file open, and opens a new editor the next time you open a file.

Dragging and Dropping in Workbench

When dragging information from the Editor and dropping it into the Expressions view or other data views, press the **CTRL** key to ensure that you copy the information to the destination rather than cutting and pasting it.

This also applies if you are dragging and dropping the information to an application outside of Workbench.

Coprocessor Support

The build mechanism provided for process-based applications does not support pre-defined compiler options for coprocessors.

In Workbench, these options must be specified in the `CC_ARCH_SPEC` build macro. For more information, see *Wind River Workbench User Interface Reference: Build Properties Dialog*.

From the command line, you can add build options to an existing makefile with the `ADDED_CFLAGS` macro. This example illustrates usage for AltiVec:

```
include $(WIND_USR)/make/rules.rtp
ADDED_CFLAGS += -tPPC7400EV:rtp
```

For the Wind River Compiler, the last `-t` option specified in the makefile overrides any `-t` option specified on the command line. For the GNU compiler, `-fvec-eabi` can be added with the `ADDED_CFLAGS` macro. For coprocessors other than AltiVec, a similar approach can be used to specify the corresponding compilation options.

VxWorks 5.5 Support

For VxWorks 5.5, only task objects are shown in the Kernel Objects view.

VxWorks 5.5 Debugging Using OCD with Workbench

When using Workbench with Tornado 2.x tools over an OCD connection, the Tornado 2.x registry is shut down and only the Workbench registry is accessible.

Renaming Projects on Windows

Projects can be renamed, even with an open connection and the debugger attached to a build target output file of the project, as long as the file size does not exceed the setting in the **Maximum host-side Symbol File Size to load fully into In-Memory Cache** field (see **Window > Preferences > Wind River > Target Management > Debug Server Settings > Symbol File Handling Settings**).

Disabling Startup Code Can Impact Workbench

Workbench launches a background process called the Wind River Registry during **org.eclipse.ui.startup**. The Registry manages multiple connections from Workbench to target boards. The launch process does not block further execution of other components.

Disabling the startup code in Workbench will result in a diminished capacity for Workbench.

This functionality is only executed if you have the Application Development perspective or the Remote Systems view open.

Project and Build Issues

Running Workbench Remotely

Wind River recommends using Virtual Network Computing (VNC) software (<http://www.realvnc.com/>) when running Workbench on a remote Linux or Solaris server. User interface responsiveness is much better compared to the standard remote X access method.

The **Remote Workspace Location** field of the **Remote Connections** dialog must contain the absolute path to the root directory of the workspace, as seen on the remote host. Environment variables are not supported in this field.

Wind River Run-Time Analysis Tools

Changing Timeouts to Accommodate RTP Initialization Task

If you want to run an RTP on your target after starting one of the Analysis tools, the RTP spawn time limit should be set at **120** seconds or greater, and the back end request time limit should be set to **30** seconds. With the target disconnected, edit these values in the **Advanced target server** panel in the **Target Server Options** tab view of the target **Properties** dialog box.



NOTE: If you do not attend to both of these items, the RTP initialization task may not receive sufficient CPU time to complete its execution before the RTP spawn time limit expires and causes the host to stop all tasks running in the RTP.

Configuring a Custom VxWorks Simulator with Networking Support

The Workbench default simulator (Wind River VxWorks Simulator), which uses a **wdbpipe** back end connection (**INCLUDE_WDB_COMM_PIPE** enabled), connects to a VxWorks target but does not successfully connect to any of the Analysis Tools tools using TCP/IP. To connect using TCP/IP, you *must* configure a custom simulator with **INCLUDE_WDB_COMM_END** enabled instead. You will also have to set up a **wrtap** adapter, and install a network daemon.

For detailed information on setting up a VxWorks simulator with full networking support, see the *Wind River VxWorks Simulator User's Guide*.

Windows Host-Related Issues

Defragment Hard Disks to Decrease Workbench Startup Time on Windows

If it takes Workbench a minute or more to open on Windows, this could be because your hard disk is very fragmented. Defragmenting your hard disk could reduce Workbench's startup time by half or more.

Linux Host-Related Issues

Welcome Page Requires Supported Browser

The default Welcome implementation is HTML-based and requires a supported browser in order to work. If no supported browser is found, the Welcome page displays in its Forms-based implementation, which has a different (simpler) appearance. Consult the SWT FAQ (<http://www.eclipse.org/swt/faq.php#browserplatforms>) for supported browsers and instructions on setting up your browser to work with Eclipse.

Workbench Crashes on SuSE Running KDE

A problem exists in the **gtk-qt-engine** system library provided by SuSE/Novell. If this package is present, Workbench may crash on startup. The only workaround is to remove this library. This will have consequences for any program that uses the library.

To remove the library, issue the following command:

```
$ rpm -e gtk-qt-engine
```

Licensing

The licensing system for Wind River Workbench assigns a license based on a combination of a unique machine ID and a unique name. If multiple Linux machines share the same name (such as **localhost.localdomain**), installation and licensing only works on the first machine to install it. To correct this problem, do the following:

1. Select **Start > System Settings > Network** on your Linux machine, and select the **DNS** tab.



NOTE: You are prompted to log in as **root** to view these settings.

2. In the **Hostname** field, replace the existing host name with a unique name for your system.
3. Close the **Network Configuration** dialog.
4. Add your new host name to the list of host names associated with address 127.0.0.1 in the **/etc/hosts** file.
5. Reboot your system so that the new network settings take effect.

Solaris Host-Related Issues

Starting the VxWorks Simulator on Solaris

In order to start the VxWorks simulator on Solaris, the path environment variable must include **/usr/openwin/bin** so that it can find **xterm**. If **xterm** is not in the path, the simulator connection fails.

5.4 Known Problems

This section lists some known problems with Wind River Workbench. For a complete list of known problems in Wind River Workbench, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

Workbench Issues

Help Unavailable for Eclipse Plug-in Registry and Error Log Views

Workbench provides the Plug-in Registry and Error Log views, but it does not include the entire Plug-in Development Environment (PDE) so the Workbench help system cannot display help for either of these views.

To find the documentation for the PDE, see
<http://www.eclipse.org/documentation/>.

Workbench sh.exe Hangs During Automated Build

If a make process hangs on Windows, check all involved Makefiles for any complex pipe ("|" operator) usage. Due to a general problem with Windows' pipe

handling, Wind River recommends that you do not use the “|” operator for combined commands in Makefiles. Instead, split them into explicit commands.

Makefiles created by Workbench managed builds do not contain any “|” operators by default.

Workbench Does Not Pass IPv6 Build Flag to VxWorks Image Project Builds

Information for how libraries are built is not visible when a VxWorks image is built, so even if IPv6 is set to **true** in **config.mk**, Workbench does not pass the IPv6 build flag.

To workaround this problem, add the following to the top of the header:

```
#ifdef INCLUDE_IPCOM_USE_INET6
#define IPCOM_USE_INET6
#endif
```

```
<app includes go here>
```

Problem Interpreting Unquoted Numbers as Arguments

When creating launch configurations, unquoted numbers in **Arguments** fields are interpreted as address values by the underlying debugger components. The corresponding argument value will be read from the memory specified by this address value.

To read a number as a string instead of an address, enclose the number in quotation marks (for example, "1").

Workbench May Crash When Using KDE with BSD Automount Daemon (amd)

When using Workbench on Red Hat Enterprise Linux (RHEL) with KDE, Workbench may crash when you use a native directory dialog to browse to a file system which is mounted by the BSD automount daemon **amd**. (WIND00069984)



NOTE: You will have similar problems when using any system application, such as **gedit**. Wind River recommends that you do not use **amd** with RHEL and KDE since it will crash any standard application.

Undefined Symbols with C++ Applications

You may get undefined symbols when linking or downloading C++ applications. C++ applications must be compiled with the same compiler used to build the VxWorks image.

For example, if the VxWorks image is built with the Wind River Compiler, the C++ applications must also be built with the Wind River Compiler. Likewise, if the VxWorks image is built with the GNU compiler, the C++ applications must be built with the GNU compiler.

Workspace Locks After an Unexpected Reboot

If your host unexpectedly reboots during a Workbench session, the **.lock** file may remain in your workspace. This means that when you restart Workbench, your workspace is locked and you cannot access your projects.

To unlock your workspace, use a shell window to navigate to your Workbench install directory, then delete the *workspaceDir*/.**metadata**/.**lock** file.

Workspace Remains Locked After Forced Quit of Workbench on Linux

If Workbench is not responding and you close the Workbench window, some Linux window managers may not actually kill the Workbench application. The Workbench window may disappear, but the application remains running.

After you click the **Close** icon on the title bar, the window manager may ask if you want to quit the application:

The window Application Development - Wind River Workbench is not responding. Forcing this application to quit will cause you to lose any unsaved changes.

If you choose to quit, Workbench may still be running and you will not be able to open the same workspace. The workaround for this problem is to manually kill the Workbench Java process using the **kill** command.

Workspace in Use Errors

Several issues can cause Workbench to display the error **Workspace in use, choose a different one**.

Possible reasons are:

- Workbench or Eclipse does not have write access to the workspace or **.metadata** folder.
- Another instance of Workbench is running in the same workspace. Only one instance of Workbench can use a workspace at a time.
- The workspace may not have unlocked after an abnormal exit. To unlock it, kill the remaining Workbench and Java processes. On Linux, check which

process still has an open file handle on the file `installDir/workspace/metadata/lock` by using the command `ls -of`.

- Your workspace is on a file system that does not support locking. Several issues in Eclipse and Java prevent the locking mechanism from working properly on some NFS-exported file systems such as HP. In such cases, after trying all the suggested fixes, use another Java Virtual Machine (JVM) argument to start Workbench:

```
./startWorkbench.sh -data /nfs/exported/hp/filesystem -vmargs  
-Dosgi.locking=none
```



NOTE: This will allow multiple users to open the same workspace, which may result in potentially unrecoverable data loss.

Workbench Startup Reports “An Error Has Occurred”

If Workbench starts with the message **An Error Has Occurred**, try starting Workbench with the **-clean** option. On Windows, modify the shortcut that starts Workbench to **rwrb.exe -clean**. On Linux, start Workbench with:

```
$ startWorkbench.sh -clean
```

If that does not help, remove the Workbench configuration directory **\$HOME/.workbench-3.x.build_ID** (Linux) or **%USERPROFILE%\workbench-3.x.build_ID** (Windows).

If neither of these methods works, try using a new workspace.

Workbench Reports fork Problems on Linux and Solaris

On Solaris and Linux, Workbench may report a problem if it fails to execute an external process, for example:

```
java.io.IOException: Not enough space at  
java.lang.UNIXProcess.forkAndExec(Native Method)
```

The Java run time has the limitation that external processes are launched using **fork**, which requires that enough free memory (including swap space) is available. The fork tries to reserve the same amount of memory as the parent process, in this case, the Workbench process. After the successful launch of the process, only the actual required process memory will be allocated. If the maximum heap size is set too high, the child process cannot be started and Workbench operations will fail.

Make sure that:

- Your system is configured with enough swap space.
- Workbench is started with a reasonable heap size, which is not unnecessarily high. Choose a heap size which is always lower than half of the freely available memory (physical and swap). Note that every started process will reduce the available memory.

Resetting Workbench to Its Default Settings

If Workbench crashes, some of your settings could get corrupted, preventing Workbench from restarting properly.

To reset all your settings to their defaults, delete the **.workbench-3.x.build_ID** directory in your home directory. It is recreated when Workbench restarts.

Workbench Font Size Problem with Exceed 8

Running Workbench with Exceed 8 causes font size problems. Workbench appears correctly when running with Exceed 7.1 or Exceed 9.0.

Online Update

It is not possible to use the **Help > Software Updates > Find and Install** feature of Eclipse to update the Eclipse platform or Workbench itself. However, the feature can still be used to update any third party plug-in that you may have added to the installation.

To update Workbench, please download and install available patches from the Wind River Online Support site (see [1.6 Latest Release Information](#), p.4).

Printing in Workbench

On Windows, **File > Print** is enabled only for the Source Editor and the Disassembly view.

Ada Editing No Longer Supported

Because of the switch from the Workbench Editor to the Eclipse Editor, Ada syntax highlighting is no longer available. Debugging still works, but you must use the default text editor for Ada files.

ASM Symbols Not Available

ASM symbols no longer appear in the Outline view when editing an assembly file.

Project Issues

Rebuilding While Debugging

If you want to rebuild a project after a debugger session has started, you may need to disconnect from the debugger (depending on the type of project):

- To rebuild an RTP, the RTP must not be running, either having exited or been terminated from Workbench. You do not need to disconnect from the debugger.
- To rebuild the VxWorks kernel image, you must disconnect from the debugger.
- To rebuild a kernel module, you can either disconnect from the target, or you can make sure the kernel module is not being used by any tasks, right-click the kernel module file, then select **Delete** (dialog prompts you with further options and cautions regarding rebuilding while still connected to the target).

Non-English Character Restrictions

Workbench does not support the use of spaces (blank characters) or non-English language characters in project names. The usage is not prohibited, but it leads to errors.

Non-English language characters are also not supported for user names on Windows.

Remote Systems View Issues

Update of Display Is Slow When Listening to Execution Life Cycle Events

When more than 100 threads are running on the target, selecting **Listen to execution context life cycle events** can severely delay the update of the Remote Systems display. Do not use this feature when there are more than 100 contexts on the target.

Run RTP Exec Path Should Be Checked for Accuracy

Each time you run an RTP by right-clicking the executable and selecting **Run RTP** from the Remote Systems view or the Project Explorer, check the exec path in the **Run RTP** dialog for accuracy. The path that automatically appears may not be correct.

Workbench Requires a Host Name to Be Set

Workbench cannot connect to the target registry or debug when the host name of the local machine is set to **localhost** (which is the default on some Red Hat Linux distributions).

On Red Hat Linux, the host name can be changed in the **System Settings > Network > DNS** tab. If you cannot change the local host name, you can work around the problem by setting the **-host** option in the file *installDir/workbench-3.x/foundation/version/resource/wtxregd/wtxregd.conf*. (SPR 108938, WIND00013729)

Cannot Connect Registry or Debug Server (All Hosts)

When Workbench cannot connect to the Wind River Registry, or the debug server connection fails (logged as **Failed to create Target Control**) check your network configuration.

1. Check the error messages for host names and IP addresses, and do a **ping** at a command prompt to check whether the corresponding host names and addresses are actually reachable. Also, try **nslookup** with the host names to check if the DNS service is configured properly.
2. Try **ping localhost** and **ping 127.0.0.1** to verify that the loopback interface is up and configured properly.

Drag and Drop Not Supported for Wind River Target Connections

It is not possible to drag images from the Project Explorer to Wind River target connections.

Build Issues

Problem Building User Mode Agent on Different Linux Host Version

When building the Linux user mode agent (usermode-agent), it is important to build it on the version upon which you intend to run it.

- When built on a Red Hat Linux Workstation 4 host, the x86-64 agent does not work reliably on a Red Hat Linux Workstation 5 machine.
- When built on a Red Hat Linux Workstation 5 host, the x86-64 agent does not start at all on a Red Hat Linux Workstation 4 machine.

The version of the agent shipped with this release is built for Red Hat Workstation 5. To debug on a Red Hat Workstation 4 host, rebuild the agent on that platform.

Build Properties Issues

Manual Addition of New Build Macros

New build macros you define in the Build Properties of a managed build project are not automatically added to any of the build tools. You must manually add the macro to the command line of the appropriate build tool(s) where appropriate. An **Add to all** button is available in the Build Paths settings which allows you to add a new Include Path to all build specs of the project in one step.

Build Property Adjustments for CPP Demo Kernel Module Sample Project

Launching the cpp demo Kernel Module sample project on an ARM target causes the error **WTX Loader error: relocation offset too large**.

A workaround for this problem is to modify your C++ build tool properties.

- For diab, include **-Xcode-absolute-far**.
 - For gnu, include **-mlong-calls**.
1. Modify these properties by right-clicking on the project, selecting **Properties**, and then selecting the **Build Tool** Tab.
 2. Select **C++ Compiler** in the **Build Tool** pull-down box.
 3. Edit the **Command** field to include the above directive after the **\$(ADDED_CFLAGS)** text.

Note that the *VxWorks Architecture Supplement: Building Applications* states to use the **\$(LONGCALL)** macro to be tool agnostic, but this macro does not resolve. (SPR 109859)

Debugging Issues

Problem Removing Breakpoint on KGDB Target

When a KGDB target is running and you remove a breakpoint that is installed on the target, the debugger can fail to remove that breakpoint if the target is in the process of hitting the breakpoint that you intend to remove.

In this situation, the target continues to stop at this breakpoint although it is no longer present in the Breakpoints view. To fully remove the breakpoint, you must disconnect from the target and then to re-connect to it.

Casting an Address to a Structure Requires Extra Parenthesis

When passing an expression containing parentheses to the debugger, you must add extra parentheses to the expression. For example, if you place a watch on the following expression and expand it, the members get errors:

```
*(GRID *)0x20658
```

However, if you add parentheses around the address, the entry expands correctly.

```
(*(GRID*)0x20658)
```

In order to access members, you must also surround the cast with parentheses:

```
(*(GRID *)0x20658).x
```

```
(WIND00016848)
```

Problem Autodetecting the Correct Processor for MIPS64 Targets

The CPU autodetection mechanism fails to select the correct processor for MIPS64 targets when connected using a WDB agent. Therefore, the user must select the processor.

To select the correct processor, follow these steps:

1. Disconnect the target connection in the Remote Systems view.
2. Right-click the target connection, then select **Properties**.
3. From the **Target Connection** dialog, click **Select** beside the **Processor** field, then select **MIPS64(32-bit addresses) > 5Kf_sa**. Click **OK**, then **OK** again to close the dialog.
4. Connect the target.

Consecutive Launches Using F11

Issuing multiple launch commands of the same application consecutively (using F11) may cause delays in Workbench view updates. This is due to the network traffic required to fetch stack information for each launched application.

Configuring WDB for Debugging a Large Number of Tasks

The WDB agent's default Gopher configuration calls for 14,000 bytes of Gopher tape. This accommodates debugging about 300 tasks with short names, but if you use longer task names or more than 300 tasks, the Remote Systems view, the shell, and the debugger will fail.

To avoid this problem, set **WDB_GOPHER_TAPE_NB** to a higher value (such as 20). This run-time parameter can be located and modified using the Kernel Configuration Editor.

Debugger Link Exception When Processing Symbol Request Return Value

When using Performance Profiler, the debugger occasionally stops resolving symbols after a reconnect. Disconnecting the target server from the target, then reconnecting the target server and Performance Profiler may resolve the error. (WIND00070739)

Target Server and ICE Connection Conflicts

Though the Remote Systems view allows you to connect the ICE and target server to a single board and use both connection types simultaneously, you must set the target server time-outs to longer default values to avoid the target server automatically disconnecting when the OCD connection stops the target.

For information about setting target server time-outs, see *Wind River Workbench User's Guide*.

Debugger Views

Debug View

In GTK+ on Linux, the Debug view icons do not wrap when the view is smaller than the width of the toolbar. So if you do not see all the icons discussed in the *Wind River Workbench User's Guide: Debugging Projects*, expand the Debug view horizontally until all the icons appear on the toolbar.

Run Control

Step return is not fully working for recursive functions. For example, consider the following simple recursive function:

```
int factorial( int x )
{
    if (x > 1)
    {
        int result = x;
        result *= factorial(x-1);
        return result;
    } else
        return 1;
}
```

If the **factorial()** function is called with an argument of 6, then you perform some **Step** operations until **factorial()** is called with an argument of 5, the call stack now contains two function calls: **factorial(6)** and **factorial(5)**.

If you step to the statement **result *= factorial(x-1);** and then request that the debugger perform a **Step Out** operation, you would expect to end up in the calling function, which is **factorial(6)**.

However, since the debugger sets a breakpoint at the statement **return result;** and then performs a **Go** operation, this recursive algorithm actually stops in the **factorial(2)** call. This is because the breakpoint isn't actually reached until the recursive function has completed calling itself all the way down to **factorial(1)** and then started returning values (unwinding the call stack).

Data Views

Changing the value of a variable or expression in data views (such as the Expressions view) may return an error, depending on the nature of the variable or expression.

For example, if the expression refers to a WRITE-ONLY register, a write succeeds but a read always fails. (SPR 103034)

Indexer Issues

Macro Limitations

No macro references appear in the call hierarchy (for MACROs that look like functions).

C++ Limitations

- Template support is limited.
- There are no references to constructors, destructors, or implicit type conversions in the symbol list. However, Declarations and Definitions are in the symbol list.
 - An implicit conversion (sequence) is generated when the type of an argument in a function call does not match the type of the parameter. It can make use of constructors to convert the argument to an object of the expected type.
 - Implicit constructor calls may be needed for the initialization of base classes, automatic variables, or returned values.

- An implicit destructor call is necessary when an automatic variable goes out of scope.

Call Tree Limitations

There are no read/write flags for variables, and polymorphic method calls are not honored.

External APIs Cannot Be Configured

The preferences page for configuring external APIs no longer exists, because these APIs can no longer be configured by the user (they are associated with a project depending on the project type). External APIs are supplied for Wind River-specific project types (such as VxWorks project types).

Properties View Issues

Quotes Not Preserved in Parameter Values

In the Properties view, Eclipse does not preserve embedded quotes in parameter values. As a workaround, use \" before and after the parameter.

For example, to include "ppp", type "\"ppp\"". (SPR 98394)

Run-Time Analysis Tools Issues

WDB_TIPC Connections Not Supported with Run-Time Analysis Tools

The Run-Time Analysis Tools do not support connecting to a target over a **WDB_TIPC** connection. This means that if you are working in an AMP environment, you cannot connect the Analysis Tools to cores that do not support networking over Ethernet, such as core 1 on the Broadcom BCM1480 board.

System Viewer Issues

Continuous Upload from Target not Recommended

Continuous upload from a target with a slow connection link is problematical. It is strongly recommended that you use a different upload method for slow connections.

Out-of-Memory Problems with Large Parameters in the Log

If extremely large parameter payloads are encountered in the log, out-of-memory problems can occur. To prevent this, only the first 1024 bytes of either **STRING** or **BLOB** event parameters are now displayed by default. You can increase or decrease this value by setting the environment variable **WRSV_PARAM_VALUE_MAXBYTES** before you start Workbench; a value of **-1** will display all bytes.

Events in Nested Interrupts Can Generate Warnings in Log Load Report

On some boards, events in nested interrupts may cause warnings to appear in the log load report, and some events will be shown in an interrupt context instead of a task. This can happen if interrupt controller drivers emit several events in the interrupt demultiplexor. In such cases, the number of **INT_ENT** events and **INT_EXIT** events emitted by the interrupt demultiplexor must satisfy:

```
number(INT_EXIT events) == number(INT_ENT events) - 1
```

A workaround has been published on the Wind River online support site; see WIND00046653.

5.5 Documentation Errata

For a detailed list of documentation errata for Wind River Workbench, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

6

VxWorks

6.1	Introduction	75
6.2	Changes in This Release	76
6.3	Usage Caveats	95
6.4	Known Problems	98
6.5	Documentation Errata	99

6.1 Introduction

VxWorks 6.6 is an update to the operating system that provides VxWorks SMP, distributed shared memory (DSHM), and other significant enhancements to the operating system. In addition, it includes new BSPs and defect fixes.

6.2 Changes in This Release

The major features of this release are VxWorks support for symmetric multiprocessing (SMP) and distributed shared memory (DSHM). These features and others are described in the following sections. Enhancements to other run-time facilities (such as compilers and networking facilities) are described in other chapters of the release notes.

6.2.1 Enhancements

VxWorks SMP

VxWorks 6.6 provides support for symmetric multiprocessing (SMP). VxWorks SMP is a configuration of VxWorks designed for symmetric multiprocessing, providing the same distinguishing RTOS characteristics of performance and determinism. The differences between the SMP and uniprocessor (UP) configurations are limited, and strictly related to support for multiprocessing. In VxWorks documentation, the terms *VxWorks SMP* and *VxWorks UP* are used to identify the uniprocessor and symmetric multiprocessing configurations of VxWorks, respectively.

With few exceptions, SMP and UP support in VxWorks share the same API—the difference amounts to only a few routines. There is binary compatibility for both kernel and RTP applications between VxWorks UP and VxWorks SMP (for the same release), as long as the applications are based on the subset of APIs used by VxWorks SMP. Not all uniprocessor APIs are suitable for an SMP system, and are therefore not available for VxWorks SMP. Similarly, SMP-specific APIs are not relevant to a uniprocessor system—but default to appropriate uniprocessor behaviors or have no effect (such as in the case of the spinlock and task locking routines).

VxWorks SMP is designed for symmetric-multiprocessor target hardware. That is, each CPU has uniform access to all memory and all devices. VxWorks SMP can therefore run on targets with multiple single-core processors or with multicore processors, as long as they provide a uniform memory access (UMA) architecture with hardware-managed cache-coherency.

For information about BSP and driver support for VxWorks SMP, see [BSPs](#), p.83 and [Drivers for VxWorks SMP](#), p.84.

For detailed information about VxWorks SMP, including sample programs and migration of VxWorks UP code to VxWorks SMP, see the *VxWorks Kernel Programmer's Guide: VxWorks SMP*.



NOTE: SMP support for VxWorks is available as an optional product. However, default VxWorks simulator SMP system images (including the WDB target agent, kernel shell, object module loader, and so on) are provided with the standard VxWorks installation as an introduction to the product.



NOTE: As part of this release of *Wind River General Purpose Platform, VxWorks Edition*, the optional VxWorks SMP product is for use with VxWorks 6.6 only. It should not be used with previous versions of VxWorks.

6

Distributed Shared Memory

The VxWorks distributed shared memory (DSHM) facility is a middleware subsystem that allows multiple services to communicate over different types of buses that support shared-memory communication. DSHM provides two main features for the services that make use of distributed shared memory: messaging over shared memory, and allocation of shared memory resources to services for their use in writing custom data. DSHM currently provides optional support for TIPC for communication over shared memory media.

Custom services can be developed for use with DSHM, and custom DSHM hardware interfaces can be developed for hardware that is not supported currently by Wind River. Communication over a local bus such as is used by multicore devices is currently supported (that is, drivers have been developed and testing conducted).

For detailed information about DSHM, see the *VxWorks Kernel Programmer's Guide: Distributed Shared Memory*. For information about the current release of TIPC, see [12. Wind River TIPC](#).

Scheduler Extension: `taskRotate()`

The new **`taskRotate()`** routine can be used as an alternative to round-robin scheduling. It allows a program to control sharing of the CPU amongst tasks of the same priority that are ready to run, rather than having the system do so at predetermined equal intervals.

Read/Write Semaphore

Read/write semaphores provide enhanced performance for applications that can effectively make use of differentiation between read access to a resource, and write access to a resource (as with SMP applications). A read/write semaphore can be taken in either read mode or write mode.

A task holding a read/write semaphore in write mode has exclusive access to a resource. On the other hand, a task holding a read/write semaphore in read mode does not have exclusive access. More than one task can take a read/write semaphore in read mode, and gain access to the same resource.

Because it is exclusive, write-mode permits only serial access to a resource, while read-mode allows shared or concurrent access. In a multiprocessor system, multiple tasks (running in different CPUs) can have read-mode access to a resource in a truly concurrent manner. In a uniprocessor system, however, access is shared but the concurrency is virtual. More than one task can have read-mode access to a resource at the same time, but since the tasks do not run simultaneously, access is effectively multiplexed.

Disable Stack Filling

In addition to using the `VX_NO_STACK_FILL` task creation option for individual tasks, the new `VX_GLOBAL_NO_STACK_FILL` system configuration parameter can be used to disable stack filling for all tasks and interrupts in the system.

By default, task and interrupt stacks are filled with 0xEE. Filling stacks is useful during development for debugging with the `checkStack()` routine. It is generally not used in deployed systems because not filling stacks provides better performance during task creation (and at boot time for statically-initialized tasks).

Task-Specific Variables with `__thread` Local Storage

Thread-local storage is a compiler facility that allows for allocation of a variable such that there are unique instances of the variable for each thread (or task, in VxWorks terms).

The `__thread` storage class instructs the compiler to make the defined variable a thread-local variable. This means one instance of the variable is created for every task in the system. A variable is defined as a thread storage class element with the compiler keyword `__thread`.

The **__thread** storage class variables can be used for both UP and SMP configurations of VxWorks, and Wind River recommends its use in both cases as the best method of providing task-specific variables. The **taskVarLib** and the RTP **tlsLib** facilities are maintained primarily for backward compatibility, are not compatible with VxWorks SMP, and their use is not recommended. In addition to being incompatible with VxWorks SMP, the **taskVarLib** and **tlsLib** facilities increase task context switch times. (Also note the change of name from **tlsLib** to **tlsOldLib** with this release; see *Task-Specific Variables: taskVarLib and tlsLib*, p.92.)

Small VxWorks Configuration Profiles

Basic I/O support has been added to the VxWorks **PROFILE_BASIC_KERNEL** small (source-scalable) configuration profile. Also see *I/O System*, p.79 and *BSP Support for Small VxWorks Configuration Profiles*, p.94.

I/O System

The I/O system can now be configured in a more precise manner. The component **INCLUDE_IO** has been split into the following components:

- **INCLUDE_IO_BASIC**—provides basic IO functionality.
- **INCLUDE_IO_FILE_SYSTEM**—provides file system support.
- **INCLUDE_POSIX_DIRLIB**—provides POSIX directory utilities.
- **INCLUDE_POSIX_FS**—provides POSIX file system APIs.
- **INCLUDE_IO_REMOVABLE**—provides support for removable file systems.
- **INCLUDE_IO_POSIX**—Provides POSIX IO support.
- **INCLUDE_IO_RTP**—provides IO support for RTPs.
- **INCLUDE_IO_MISC**—miscellaneous IO functions that are no longer referenced but are provided for backward compatibility.

The component **INCLUDE_IO_SYSTEM** is still provided for backward compatibility. It includes the components listed above by default.

HRFS File System

The Highly Reliable File System (HRFS) now provides configurable transaction points, which allow for finer control of how and when transaction points are set, to improve performance. The **HRFS_DEFAULT_COMMIT_POLICY** and **HRFS_DEFAULT_COMMIT_PERIOD** component configuration parameters are used to statically define the default commit policy and period.

Both kernel and RTP applications can change commit policies at runtime. The following **ioctl()** functions are used to get and set commit policies:

- **FIOCOMMITPOLICYGETFS**
- **FIOCOMMITPOLICYSETFS**
- **FIOCOMMITPERIODGETFS**
- **FIOCOMMITPERIODSETFS**

The commit policy for each volume can be changed using the **ioctl()** function **FIOCOMMITPOLICYSETFS** as the second parameter. The third parameter then specifies the actual commit policy: **FS_COMMIT_POLICY_AUTO**, **FS_COMMIT_POLICY_MANUAL**, or **FS_COMMIT_POLICY_PERIODIC**.

dosFs File System

In previous versions of dosFs, one cache was shared by the FAT, directory entries, and data. For this release, the cache has been modified so that the FAT, directory entries, and the data each have their own cache. This allows for finely tuning the cache and improved performance. The following parameters to **INCLUDE_DOSFS_CACHE** allow for cache configuration:

- **DOSFS_DEFAULT_FAT_CACHE_SIZE** (default 16 KB)
- **DOSFS_DEFAULT_DATA_CACHE_SIZE** (default 128 KB)
- **DOSFS_DEFAULT_DIR_CACHE_SIZE** (default 64 KB)

These parameters replace **DOSFS_DEFAULT_CACHE_SIZE** (which does exist in this release). The settings for a particular cache can be retrieved using **dosFsCacheInfo()** and **dosFsCacheTune()**. Also see [Modified Routines](#), p.90.

Flash File System Support with TrueFFS

With this current release, TrueFFS automatically disables FAT monitoring and sets the start sector to 1. For use with dosFs, this means that the device must be formatted when it is used for the first time.

The performance issues related to using HRFS with a TrueFFS device have been resolved.

6

SDA Support for Power Architecture

Small data area (SDA) support has been implemented for the VxWorks kernel programming environment with the Power Architecture/PowerPC (it is also currently supported in the user-space programming environment and has been in the past). The SDA construct is defined by the PowerPC Embedded Application Binary Interface (EABI) specification. SDA is designed to take advantage of base plus displacement addressing mode, which provides a more memory-efficient way of accessing a variable and better performance.

SDA is not supported in downloaded kernel modules (DKMs).

For information about the impact of SDA support on other features, see [SDA and Loading Kernel Object Modules](#), p.96 and [SDA and Custom Power Architecture BSPs](#), p.96.

Kernel Configuration

The **vxprj** VxWorks configuration utility provides support for VxWorks SMP and for asymmetric multiprocessor (AMP) projects. The new **vxprj** options are as follows:

- **-smp**
- **-amp**

Examples of use are as follows:

```
vxprj create -smp hpcNet8641 diab  
vxprj create -amp hpcNet8641 diab
```



CAUTION: Boot loaders for VxWorks SMP must not be built with the SMP or AMP build options—neither with Workbench nor with **vxprj**.

Boot Loader

The boot loaders provided with this release can be used for VxWorks UP, SMP, and AMP projects. There is no special boot loader for symmetric multiprocessing or asymmetric multiprocessing.

For information about the SMP boot process, see the *VxWorks Kernel Programmer's Guide: VxWorks SMP*.

Kernel Shell

The kernel shell now provides the **histSave()** and **histLoad()** commands for the C interpreter, as well as the **history save** and **history load** commands for the command interpreter. The commands allow you to save the shell history to, and load it from, a file. The commands are provided by the **INCLUDE_SHELL_HISTORY_FILE** component for the C interpreter commands, and by the **INCLUDE_HISTORY_FILE_SHELL_CMD** component for the command interpreter.

The VxWorks kernel shell (as well as the VxWorks host shell) has been enhanced with new commands for networking technologies. Note that the new commands only run in command-interpreter mode. For information about the new commands, see the networking guides provided with this release.

Kernel Object Module Loader

The **LOAD_FULLY_LINKED** option is now available for the kernel object module loader as well as the host loader. It allows for loading fully linked modules (that is, modules without any unresolved symbols or relocations). For more information, see the **loadModuleAt()** shell command reference.

Also see [SDA and Loading Kernel Object Modules](#), p.96, with regard to loading object modules that include SDA.

WDB Target Agent

The WDB target agent provides support for VxWorks SMP. For system mode debugging, this means the following:

- When the system is stopped either by a breakpoint or an exception, all CPUs of the target are stopped.

- The index of the CPU that hit a breakpoint or that got an exception is reported through the notification.

BSPs

The new BSPs provided with this release for VxWorks SMP are as follows:

- **cav_cn3xxx_mipsi64r2sf**—Cavium OCTEON CN3xxx MIPS board
- **hpcNet8641**—Freescale MPC8641D board
- **idp945**—Intel Capell Valley board
- **sb1480_mipsi64**—Broadcom BCM1480 board for MIPS

For information about all the BSPs provided with this release, see [Supported BSPs](#), p.22.

pcPentium-Based BSP Configuration

In this release, the default **pcPentium**-based BSP configuration, including the **pcPentium3** and **pcPentium4**, has changed. The default configuration now includes the popular GEI Ethernet driver in addition to the FEI driver. This means you will no longer need to change the BSP configuration to include support for the GEI driver. In addition, a generous pool size of 100 KB for the parameter **HWMEM_POOL_SIZE** is set as default to better provide development support. You will notice that the footprint size for this BSP has increased due to these changes. However, both the additional driver and the memory pool size can be tuned and scaled back to allow for a smaller footprint. The function **hwMemShow()** can display information about memory consumption in the **HwMemPool** pool.

For more information on memory allocation, see the *Services Available to Drivers* chapter in the *VxWorks Device Driver Developer's Guide, Volume 1*. For more information on **hwMemShow()**, see the reference entry for this routine.

For more information on the GEI Ethernet driver, see the *VxWorks Device Driver Developer's Guide, Volume 2: Network Drivers*.

Drivers for VxWorks SMP

The following drivers are supported for VxWorks SMP:

Bus Controllers

m85xxPci.c—m85xx PCI bus controller
pentiumPci.c—PCI host controller for Pentium CPU
vxbIPiix4Mf.c—iPIIX4 Multifunction PCI device
vxbMsc01Pci.c—PCI bridge on SOC-it system controller

Console

vxbI8042Kbd.c—Intel 8042 keyboard
vxbM6845Vga.c—Motorola 6845 VGA console

END

am79c97xVxbEnd.c—AMD Am79c97x PCnet/PCI
an983VxbEnd.c—Infineon AN983B/BX
fei8255xVxbEnd.c—Intel PRO/100
gei825xxVxbEnd.c—Intel PRO/1000
mvYukonIIVxbEnd.c—Marvell Yukon II
mvYukonVxbEnd.c—Marvell Yukon I
ns8381xVxbEnd.c—National Semiconductor DP83815/6
rtl8139VxbEnd.c—RealTek 8139/8100 10/100
rtl8169VxbEnd.c—RealTek 8139C+/8101E/816x/811x
sbeVxbEnd.c—Broadcom/Sibyte BCM1250
tc3c905VxbEnd.c—3Com 3c905/B/C
tsecVxbEnd.c—Freescale TSEC
vxbEtsecEnd.c—Freescale Enhanced TSEC
vxbSmscLan9118End.c—SMSC LAN9118

Interrupt Controller

vxbEpicIntCtrl.c—Embedded Programmable Interrupt Controller (EPIC)
vxbI8259IntCtrl.c—Intel 8259A Programmable Interrupt Controller
vxbIoApicIntr.c—Intel IO APIC/xAPIC
vxbLoApicIntr.c—local APIC/xAPIC (Advanced PIC)
vxbMipsCavIntCtrl.c—interrupt controller for Cavium CN3xxx family
vxbMipsIntCtrl.c—generic interrupt controller for MIPS CPU
vxbMipsSbIntCtrl.c—interrupt controller for BCM1480
vxbMpApic.c—Intel MP APIC/xAPIC (Advanced PIC)

Media Independent Interface

bcm52xxPhy.c—Broadcom bcm52xx 10/100 ethernet PHY
bcm54xxPhy.c—Broadcom bcm54xx 10/100/1000 ethernet PHY
dm9161Phy.c—Davicom DM9161(A) 10/100 ethernet PHY
geiTbiPhy.c—Intel PRO/1000 fiber TBI ports
genericPhy.c—generic 10/100/1000 ethernet PHY
genericTbiPhy.c—generic TBI interfaces
lxt972Phy.c—Intel LXT972 10/100 ethernet PHY
mdio.c—generic MII MDIO ports
mv88E1113Phy.c—Marvell 88E1113 fiber PHY
mv88E1x11Phy.c—Marvell 88E11xx 10/100/1000 ethernet PHY
rtl8169Phy.c—RealTek integrated 10/100/1000 ethernet PHY
rtl8201Phy.c—RealTek 8201L 10/100 ethernet PHY
tsecMdio.c—TSEC MDIO port
vsc82xxPhy.c—Vitesse/cicada single-chip 10/100/1000 ethernet PHY

Resource

m85xxCCSR.c—Power Architecture (PowerPC) 85xx CCSR resource allocation

SIO

vxNs16550Sio.c—National Semiconductor 16550 Serial

vxOctheonSio.c—OCTEON 16550ish Serial

vxPrimeCellSio.c—ARM AMBA UART

vxSb1DuartSio.c—BCM1250/1480 serial

Storage

vxIntelIchStorage.c—ICH0/1 (82801) ATA/IDE and ATAPI CDROM

vxSI31xxStorage.c—Silicon Image 3132/3124 SATA

Timer

vxCn3xxxTimer.c—timer on Cavium CN3xxx Core

vxI8253Timer.c—Intel 8253 timer

vxIntelTimestamp.c—timestamp on the Intel chip

vxLoApicTimer.c—IoApic timer

vxMc146818Rtc.c—MC146818 real time clock

vxMipsR4KTimer.c—MIPS R4000 on CPU Timer

vxOpenPicTimer.c—OpenPIC timer

vxPpcDecTimer.c—Power Architecture (PowerPC) decrementer

vxSb1Timer.c—timer on BCM Core

USB

All the USB drivers provided by Wind River for this release are supported for VxWorks SMP.

New APIs

For information about new kernel and application APIs, see [6.2.2 API Changes](#), p.87.

Compilers

For information about the compilers provided with this release, see [8. Wind River Compiler](#) and [9. Wind River GNU Compiler](#).

6.2.2 API Changes

This section describes new APIs and modifications to existing APIs.

For information about routines that have been deprecated, see [6.2.4 Deprecated Features](#), p.91.

New Routines

SMP

- `spinLockIsrInit()`
- `spinLockIsrTake()`
- `spinLockIsrGive()`
- `spinLockTaskInit()`
- `spinLockTaskTake()`
- `spinLockTaskGive()`
- `intCpuLock()`
- `intCpuUnlock()`
- `taskCpuLock()`
- `taskCpuUnlock()`
- `vxAtomicAdd()`
- `vxAtomicSub()`
- `vxAtomicInc()`
- `vxAtomicDec()`
- `vxAtomicOr()`
- `vxAtomicXor()`
- `vxAtomicAnd()`
- `vxAtomicNand()`
- `vxAtomicSet()`
- `vxAtomicClear()`
- `vxCas()`
- `taskCpuAffinitySet()`
- `taskCpuAffinityGet()`
- `kernelIsCpuIdle()`
- `kernelIsSystemIdle()`
- `kernelCpuEnable()`

- vxCpuConfiguredGet()
- vxCpuEnabledGet()
- vxCpuIndexGet()
- vxCpuIdGet()

C Macros for SMP

- CPUSET_SET()
- CPUSET_SETALL()
- CPUSET_SETALL_BUT_SELF()
- CPUSET_CLR()
- CPUSET_ZERO()
- CPUSET_ISSET()
- CPUSET_ISZERO()
- CPUSET_ATOMICSET()
- CPUSET_ATOMICCLR()
- VX_MEM_BAR_R()
- VX_MEM_BAR_W()
- VX_MEM_BAR_RW()



NOTE: The APIs provided for VxWorks SMP are also available for VxWorks UP (to assist with code portability). In VxWorks UP, they default to corresponding uniprocessor API behavior—for example, **spinLockTaskTake()** defaults to **taskLock()** behavior—or they perform other appropriate behavior, or have no effect.



NOTE: A small number of routines provided in uniprocessor (UP) VxWorks are not supported in VxWorks SMP because they are not appropriate for multiprocessing systems. For information in this regard, see the *VxWorks Kernel Programmer's Guide: VxWorks SMP*.

Distributed Shared Memory (DSHM)

- dshmMuxHwRegister()
- dshmMuxHwGet()
- dshmMuxHwNodesNumGet()
- dshmMuxHwTasGet()
- dshmMuxHwTasClearGet()
- dshmMuxHwOffToAddr()
- dshmMuxHwAddrToOff()
- dshmMuxHwLocalAddrGet()
- dshmMuxSvcNodeJoin()

- `dshmMuxSvcNodeLeave()`
- `dshmMuxSvcRegister()`
- `dshmMuxSvcObjGet()`
- `dshmMuxSvcObjRelease()`
- `dshmMuxSvcWithdraw()`
- `dshmMuxSvcWithdrawComplete()`
- `dshmMuxMsgSend()`
- `dshmMuxMsgRecv()`
- `dshmMuxMemAlloc()`
- `dshmMuxMemFree()`

C Macros for DSHM

- `DSHM()`
- `DSHM_PTR()`
- `DSHM_CAST()`
- `DSHM_TYPE()`
- `DSHM_BUILD()`
- `DSHM_DST_SET()`
- `DSHM_DAT8_SET()`
- `DSHM_DAT16_SET()`
- `DSHM_DAT32_SET()`
- `DSHM_SVC_GET()`
- `DSHM_SRC_GET()`
- `DSHM_DST_GET()`
- `DSHM_TYP_GET()`
- `DSHM_DAT_GET()`
- `DSHM_DAT8_GET()`
- `DSHM_DAT16_GET()`
- `DSHM_DAT32_GET()`
- `DSHM_MEM_BARRIER()`
- `DSHM_BUILD()`

Read/Write Semaphore

- `semRWCreate()`
- `semRTake()` ¹
- `semWTake()` ¹

1. There is no specific API for giving a read/write semaphore. Use `semGive()` to give a read/write semaphore.

Scheduling

- `taskRotate()`

dosFs File System

- `dosFsCacheInfo()`
- `dosFsCacheTune()`

HRFS File System

- `commit()`

Semaphores

- `semExchange()`

Modified Routines

`checkStack()`

Now displays the interrupt stack for all CPUs in an SMP system.

`spy()`

Now reports the number of ticks spent in kernel, interrupt, idle, and task code for each CPU in an SMP system.

`timex()`

Is now supported for SMP systems.

`dosFsCacheCreate()`

Now takes additional parameters for all three cache types (also see [dosFs File System](#), p.80).

`rtpKill()`

The syntax has changed, but the functionality is the same.

`rtpSigQueue()`

The syntax has changed, but the functionality is the same.

Removed Routines

The `pipe()` routine has been removed. It was implemented previously as a stub routine that returned **ERROR**.

The **semOLib** library has been removed. It provided the following routines:

- `semOLibInit()`
- `semCreate()`

- `semInit()`
- `semOTake()`
- `semClear()`

The `tlsLib` library is not supported for VxWorks SMP, and the routines have been moved to `tlsOldLib`, which has the following routines:

- `tlsKeyCreate()`
- `tlsValueGet()`
- `taskValueSet()`
- `tlsSizeGet()`

In addition, the following individual routines have also been removed:

- `shellInit()`
- `shell()`
- `shellOrigStdSet()`

6.2.3 Fixed Problems

For a list of problems fixed in VxWorks, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

6.2.4 Deprecated Features

Configuration and Build Using `config.h`

Wind River recommends using either Workbench or the `vxprj` command-line facility for configuring and building VxWorks. The legacy method using `bspDir/config.h` and `bspDir/make` has been deprecated for most purposes since VxWorks 6.0, and it *cannot* be used for multiprocessor (SMP and AMP) development.

However, the `config.h` method must still be used for the following:

- Some middleware products (consult the documentation in this regard).
- Boot loaders, when the BSP does not support the `PROFILE_BOOTAPP` configuration profile.

The `config.h` method can also be used for uniprocessor BSP development before CDFs have been implemented.

vxWorks.st Image Type

Building a **vxWorks.st** image type is not supported by Workbench or the **vxprj** command-line tool, and the legacy **config.h** method has been deprecated (see [Configuration and Build Using config.h](#), p.91).

However, a comparable VxWorks image that includes a standalone symbol table can be configured and built with a Workbench VIP project or with **vxprj**. To do so, configure VxWorks with the **INCLUDE_STANDALONE_SYM_TBL** and **INCLUDE_SHELL** components.

In order to prevent the network from starting automatically, add **INCLUDE_NET_INIT_SKIP** as well. The network can then be started from the shell as follows:

```
-> sp usrNetworkInit
```

Note, however, that since **INCLUDE_NET_INIT_SKIP** is incompatible with the WDB agent running **WDB_COMM_END** or **WDB_COMM_NETWORK**, the WDB agent must be excluded, or a non-network back end (such as **INCLUDE_WDB_COMM_SERIAL**) must be used.

To get closer to the legacy configuration of a **vxWorks.st** image, components such as the following can be added as well:

- **INCLUDE_SHOW_ROUTINES**
- **INCLUDE_DEBUG**
- **INCLUDE_UNLOADER**
- **INCLUDE_DISK_UTIL**

Task-Specific Variables: taskVarLib and tlsLib

The **taskVarLib** and **tlsLib** facilities are maintained primarily for backward-compatibility, are not compatible with VxWorks SMP, and their use is not recommended. Note that with this release **tlsLib** is now named **tlsOldLib** (also see [Removed Routines](#), p.90). In addition to being incompatible with VxWorks SMP, the **taskVarLib** and **tlsLib** facilities increase task context-switch times.

The **__thread** storage class variables can be used for both UP and SMP configurations of VxWorks, and Wind River recommends its use in both cases as the best method of providing task-specific variables (see [Task-Specific Variables with __thread Local Storage](#), p.78).

Symbol Routines

The following individual routines have been deprecated and will be removed in a future release:

- **symFindByValue()**
- **symFindByValueAndType()**

Use of Various Kernel APIs with VxMP

This is an advance notice of deprecation of the use of standard VxWorks routines for accessing VxMP objects. For performance reasons, the following APIs will not support VxMP objects in a future release, but will be replaced with routines specifically designed for VxMP:

- **msgQSend()**
- **msgQReceive()**
- **msgQNumMsgs()**
- **msgQShow()**
- **msgQInfoGet()**
- **semGive()**
- **semTake()**
- **semFlush()**
- **semShow()**
- **semInfo()**
- **memPartAlloc()**
- **memPartRealloc()**
- **memPartFree()**
- **memPartFindMax()**
- **memPartOptionsSet()**
- **memPartAddToPool()**

The use of each of these routines with VxMP will be replaced by a VxMP-specific routine, similar to the existing object creation routines **semBSmCreate()**, **semCSmCreate()**, **msgQSmCreate()**, and memory partition creation routine **memPartSmCreate()**. This change is intended to provide performance improvements for both VxMP and the standard routines.

CDF Objects

Use of the following component descriptor file (CDF) objects is deprecated:

- **BSP_STUB**
- **EXCLUDES**
- **MACRO_NEST**

BSPs

Several BSPs have been deprecated in this release. For more information, see [BSPs](#), p.15.

6.2.5 Unsupported Features

This section describes features that are not provided in this release.

Configuration and Build of VxWorks SMP Using config.h

VxWorks SMP cannot be configured and built with the legacy method involving **config.h** and **make**. Also see [Configuration and Build Using config.h](#), p.91.

dosFs File System

The **DOSFS_DEFAULT_CACHE_SIZE** configuration parameter has been replaced by the following three parameters:

- **DOSFS_DEFAULT_DATA_CACHE_SIZE**
- **DOSFS_DEFAULT_DIR_CACHE_SIZE**
- **DOSFS_DEFAULT_FAT_CACHE_SIZE**

Also see [dosFs File System](#), p.80.

BSP Support for Small VxWorks Configuration Profiles

The sb1250 BSP does not support small VxWorks (source-scalable) configuration profiles for this release. The profiles in question are **PROFILE_MINIMAL_KERNEL**, **PROFILE_BASIC_KERNEL**, and **PROFILE_BASIC_OS**.

For this release, the small VxWorks profiles are provided for **wrSbcPowerQuiccII** (for Power Architecture) and **integrator1136jfs** (for ARM).

Routines

See [Removed Routines](#), p.90.

TaskLib.c

The global variable **taskPriRangeCheck** has been removed from **taskLib.c**. The variable was previously used to skip priority range checking. The default was **TRUE**. Now the code automatically checks for the range, and does not have the option of not doing so. This option was undocumented.

vxWorksCommon.h

The following definitions have been removed from **vxWorksCommon.h**:

```
/* network configuration parameters */

#define   INET           /* include internet protocols */
#undef    BSD           /* remove any previous definition */
#define   BSD           44 /* BSD 4.4 like OS */
#define   BSDDEBUG      /* turn on debug */
#define   GATEWAY        /* tables to be initialized for
gateway routing
*/
```

6

Boot Loader Shell Command

The boot loader shell command **s** (lower-case 's'), which was used to turn the CPU system controller **ON (1)** or **OFF (0)** (for boards on which the system controller can be enabled by software) is no longer supported.

6.3 Usage Caveats

This section lists usage caveats for VxWorks.

timexLib and VxWorks SMP

Note that while **timexLib** can avoid precision errors by auto-calibrating itself and doing several calls of the functions being monitored, it suffers from the lack of scheduling management during the calls. The tasks can move between CPUs while the measurements take place. Depending on how often this occurs, this is likely to have an impact the precision of the measurement.

Configuration and Build

See [Configuration and Build Using config.h](#), p.91.

Boot Loaders

Do not build boot loaders for symmetric multiprocessor (SMP) and asymmetric multiprocessor (AMP) configurations of VxWorks with the SMP or AMP build option—neither with Workbench nor with **vxprj**. The same boot loaders are used for uniprocessor (UP), SMP, and AMP, configurations of VxWorks.

Scalable Projects Require Source

In order to build scalable projects, you must have the VxWorks source code product. If you are unsure about whether you have VxWorks source code, please contact your Wind River sales representative.

SDA and Loading Kernel Object Modules

If a kernel module is built with SDA, the loader will not load it, but generates an error message. The error messages for the kernel object module loader and the host loader (respectively), are as follows:

- `S_loadLib_SDA_NOT_SUPPORTED`
- `WTX_ERR_LOADER_SDA_NOT_SUPPORTED`

Workbench also displays the following error message: “WTX Loader Error: dynamic loading of modules with SDA (Small Data Area) sections is not supported; check your build rules and make sure your module does not contain any SDA section or relocation. The loader cannot perform SDA relocation.”

Modules with SDA should be statically linked with the kernel. For more information about VxWorks SDA support, see [SDA Support for Power Architecture](#), p.81.

The Wind River Compiler (diab) assembler flag `-Xwarn-use-greg` can be used to generate the following warning if code accesses the SDA reserved registers:

```
Xwarn-use-greg=0x2004
```

In addition, the `SDA_DISABLE` makefile variable can be used to disable SDA, as follows:

```
SDA_DISABLE=TRUE
```

For information about SDA support, see [SDA Support for Power Architecture](#), p.81.

SDA and Custom Power Architecture BSPs

The VxWorks kernel initialization process now initializes Power Architecture (PowerPC) SDA/SDA2 base registers. If a custom BSP invokes a C function from within `_sysInit()`—that is, before the invocation of `usrInit()`—the SDA/SDA2 base registers need to be initialized prior to calling the C function. They should be initialized as follows:

```
lis    r2, HI(_SDA2_BASE_)
ori    r2, r2, LO(_SDA2_BASE_)

lis    r13, HI(_SDA_BASE_)
ori    r13, r13, LO(_SDA_BASE_)
```

Note that the Wind River Compiler (diab) assembler flag **-Xwarn-use-greg** can be used to generate the following warning if code accesses the SDA reserved registers:

```
Xwarn-use-greg=0x2004
```

For information about SDA support, see [SDA Support for Power Architecture](#), p.81.

Limited Support for RTA Profiling

VxWorks supports one compiler option, **-Xrtc**, for generating Run-Time Analysis (RTA) profiling information. Other compiler profiling options, such as **-Xprof**, are not supported in VxWorks.

For **-Xrtc**, all mask options are supported except *mask* = 0x4.

Restrictions on tt()

The **tt()** stack trace shell command does not work consistently for the following architectures:

- ARM/XScale
- MIPS
- PPC
- SH

tt() should not be used with these architectures.

Cavium BSP

To use the new Cavium OCTEON BSP (**cav_cn3xxx_mipsi64r2sf**), you must first download the Cavium SDK from the following URL:

http://cavium.com/processor_octeon_software_develop_kit.html

Further, the Cavium BSP does not support the Wind River Compiler. To build this BSP, use the GNU compiler.

ColdFire BSPs

The ColdFire BSPs (**m54x5evb** and **m5329evb**) do not support the GNU compiler. To build these BSPs, use the Wind River Compiler.

Target Server File System: TSFS

The Target Server File System (TSFS) is not designed for use with large files (whether application executables or other files), and performance may suffer when they are greater than 50 KB. For large files, use FTP or NFS instead of TSFS.

Boot Loader M and N Commands

The **M** command is a replacement for the **N** command, which is maintained for backward compatibility purposes. The commands are provided with the **INCLUDE_BOOT_ETH_MAC_HANDLER** and **INCLUDE_BOOT_ETH_ADR_SET**, respectively. Do not use both components in the same configuration of VxWorks. For information about which of the two commands is supported for a given BSP, consult the BSP reference.

In addition, do not use boot loaders configured with the **INCLUDE_BOOT_ETH_MAC_HANDLER** component to boot VxWorks images produced with any release prior to VxWorks 6.5.

BSP Reference Instructions

The **target.ref** files for some BSPs have instructions describing how to write a boot loader image into the target flash memory. The instructions may contain references to legacy tools, such as visionCLICK, visionICE, or Tornado. Their equivalent counterparts are Wind River Workbench for On-Chip Debugging, Wind River ICE, and Wind River Workbench, respectively. Most of the parameters, such as **start address**, **offset** and **bias**, can be applied directly to the new tools as well as the old ones. Make adjustments as necessary when following the instructions.

6.4 Known Problems

This section lists some known problems with VxWorks. For a complete list of known problems in VxWorks, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

Power Management Not Supported on Power Architecture (PowerPC)

In SMP configuration, power management (the **INCLUDE_CPU_PWR_MGMT** and **INCLUDE_CPU_LIGHT_PWR_MGR** components) is not supported on the Power Architecture (PowerPC). Enabling this support can potentially cause board lock-ups with heavy interrupt rates. (WIND00104713)

In UP configuration, with power management included (the **INCLUDE_CPU_PWR_MGMT** and **INCLUDE_CPU_LIGHT_PWR_MGR** components) and with heavy interrupt traffic, the **hpcNet8641** and **wrSbc8641d**

Power Architecture BSPs may encounter board lock-ups. To work around this issue, disable power management for the BSP. (WIND00111630)

ColdFire and TrueFFS

The ColdFire **m54x5evb** BSP included with this release does not provide support for TrueFFS.

6

6.5 Documentation Errata

This section lists some documentation errata for documents associated with VxWorks. For a detailed list of documentation errata for VxWorks, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

Limited Support for RTA Profiling

The *VxWorks Application Programmer's Guide* and *VxWorks Kernel Programmer's Guide* describe RTEC compiler support for VxWorks. They do not indicate that other RTA features are not supported.

CDF Naming Convention

The *VxWorks Kernel Programmer's Guide: Kernel* contains the following instructions on Component Description Files (CDFs):

- All VxBus driver component names are of the form **INCLUDE_driverType_driverName** (for example, **INCLUDE_SIO_NS16550**).

This text should read as follows:

- All VxBus driver component names are of the form **DRV_driverType_driverName** (for example, **DRV_SIO_NS16550**).

(WIND00101961)

Transmitting Zero-Length Buffers

Section 6.2.9 *netBufLib: Transferring Data with MBlks* of the *VxWorks Device Driver Developer's Guide, Volume 2: Writing Class-Specific Device Drivers* should include the following note after the second paragraph:



NOTE: The network stack may pass zero-length **mBlk** chains. If the device your driver supports attempts to transmit zero-length buffers, your driver must ensure that zero-length data packets are not transmitted.

DSHM Demo Requires Modification of DSHM_MAX_SERVICES Parameter

The **DSHM_MAX_SERVICES** parameter of the **DSHM_MUX** component specifies the maximum number of DSHM services that can run on a single board. By default, **DSHM_MAX_SERVICES** is set to 2. Any user-created services must be added to this number. The DSHM demo contains a user-created DSHM service. As a result, if you do not run any other user-created DSHM services, to run the demo, you must increase **DSHM_MAX_SERVICES** to 3.

Before Running the DSHM Demo, Start Both Cores

The DSHM demo runs on two cores. Both cores must be running before you start the demo.

Wind River VxWorks Simulator

7.1 Introduction	101
7.2 Changes in This Release	102
7.3 Known Problems	102
7.4 Documentation Errata	103

7.1 Introduction

The Wind River VxWorks Simulator 6.6 is a simulated VxWorks target for use as a prototyping and test-bed environment. For external applications needing to interact with a VxWorks target, the capabilities of a VxWorks simulator instance are identical to those of a VxWorks system running on target hardware. A VxWorks simulator instance supports the standard VxWorks applications, such as the network applications and the target and host VxWorks shells. Building those applications into a VxWorks simulator instance is no different than building them into any VxWorks cross-development environment using a standard BSP.

7.2 Changes in This Release

7.2.1 Enhancements

This release of the Wind River VxWorks Simulator includes support for SMP (symmetric multiprocessing) as well as improvements to the Windows user interface. For details, see the *Wind River VxWorks Simulator User's Guide*.

Wind River VxWorks Simulator supports running VxWorks SMP on a single-processor host, and can run any number of VxWorks processors on any number of host processors. The best performance is achieved when the number of VxWorks processors is equal to the number of host processors.



NOTE: SMP support for VxWorks is available as an optional product. However, default SMP system images (including the WDB target agent, kernel shell, object module loader, and so on) are provided with the standard VxWorks installation as an introduction to the product.

Documentation Enhancements

The *Wind River VxWorks Simulator User's Guide* has been updated for this release to reflect support for SMP. In addition, several errors in the documentation have been corrected, and the networking tutorials now include instructions on how to run an instance of the VxWorks simulator on the local network. There are also instructions for installing the host connection driver (the WRTAP driver) on Windows Vista hosts.

7.2.2 Fixed Problems

For a list of problems fixed in Wind River VxWorks Simulator, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

7.3 Known Problems

For a complete list of known problems in Wind River VxWorks Simulator, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

7.4 Documentation Errata

This section lists some documentation errata for documents associated with Wind River VxWorks Simulator. For a detailed list of documentation errata for Wind River VxWorks Simulator, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

Wind River VxWorks Simulator User's Guide

The *Wind River VxWorks Simulator User's Guide* currently contains the following text on setting up the network daemon:

The remainder of this section tells you how to set up a VxWorks simulator network daemon. Using the VxWorks simulator network daemon, you can link same-host VxWorks simulator instances into simulated subnets. By default, these internal subnets do not communicate with the host. However, included with the VxWorks simulator is a simulated network drive—a host adapter interface—that you can use to give the host system an address on the simulated subnet.

These instructions are correct, but they do not make it sufficiently explicit that if you want to include the host machine on the simulated subnet, you must start the host connection driver (WRTAP) before you start the VxWorks simulator network daemon. (WIND00112051)

8

Wind River Compiler

- 8.1 Introduction 105
- 8.2 Changes in This Release 106
- 8.3 Supported Hardware and Software 109
- 8.4 Usage Caveats 109
- 8.5 Known Problems 111
- 8.6 Documentation Errata 112

8.1 Introduction

The Wind River Compiler is a complete toolkit for embedded application development, including C and C++ compilers, assemblers, linkers, utilities, and standard libraries for a variety of target CPU architectures. The compiler version shipped with this release is 5.6.0.

8.2 Changes in This Release

All components in a toolchain release (compiler, assembler, linker, and utilities) have the same version number, even when a particular component is unchanged. Changes are further organized by language and target:

- Changes to the C or C++ compiler apply to all targets unless otherwise noted.
- Changes to the C compiler also apply to C++, unless otherwise noted.
- Changes marked with a target name, such as ARM or Power Architecture (PowerPC), apply to all languages unless otherwise noted.

8.2.1 Enhancements

Support for `__builtin_prefetch` and `__builtin_expect`

The compiler now supports the `__builtin_prefetch` and `__builtin_expect` intrinsic functions.

Support for `__thread` Keyword Added

The compiler now supports the use of the `__thread` keyword to allocate thread-local storage. Such variables will have a unique instance for every thread.

Support for Thumb-2 Architecture Added

The Wind River Compiler now supports the ARM Thumb-2 architecture. The Thumb-2 target may be selected for compilation with the `-tARMT2LS`, `-tARMT2LN`, or `-tARMT2LV` options.



NOTE: For the ARM7 architecture, both VFP and Thumb-2 have been tested separately; however, they have not been tested jointly.

Enhanced Behavior for `-Xkill-opt` (`-Xkill-opt=1`)

A new mask, 0x1, for the `-Xkill-opt` option, disables various optimizations. These include:

- Several registerization optimizations designed to reduce read and write operations, including: registerization of memory references across loops; registerization of function memory references; and registerization of loop-invariant loads.

- Loop unswitching (loop splitting).
- Tail merging (merges common code out of **if/then/else** and **switch** statements).
- Optimization for the `__builtin_expect` intrinsic function.

New OPTIONAL Section Type Specifier for the Linker

You may now designate a section as **OPTIONAL** in a linker command language file. An **OPTIONAL** section will be discarded by the linker if it is empty. For example, specifying the following means that the `.text1` section will be created only if it will not be empty, and, if it is created, it inherits its type from its contents:

```
.text1 ( OPTIONAL ) :
```

Support for 64-Bit Bit-Fields

The Wind River Compiler now supports 64-bit bit-fields (for example, they may be used in variables of type **long long**).

Support for Enabling Certain MIPS Instructions

Three options now enable code generation of certain MIPS instructions for any MIPS processor.

- When **-Xmips-mad** is specified, the compiler can generate the MIPS instructions **msub**, **msubu**, **mad** and **madu**.
- When **-Xmips-movc** is specified, the MIPS conditional move instructions **movn** and **movz** can be generated.
- When **-Xmips-mul** is specified, the compiler can generate the MIPS **mul** instruction.

Support for ColdFire Processors

See [8.2.3 Unsupported Features](#), p.108.

New Error Message Documentation

A new manual, *Wind River Compiler Error Messages*, is now included with this release. It lists several hundred new and previously undocumented error messages.

C++, C99 Libraries Documentation

Documentation for the C++ and C99 libraries may now be found online at https://portal.windriver.com/noAuth/wr_compiler_docs. Note that nonstandard C++ and C99 headers are not supported.

8.2.2 Fixed Problems

For a list of problems fixed in the Wind River Compiler, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

8.2.3 Unsupported Features

HP-UX

The HP-UX operating system is no longer supported.

WindISS for x86 (Pentium)

The Wind River Instruction Set Simulator does not yet support simulation of x86 targets.

Library Support Restrictions

While the compiler supports the **wchar_t** type, in most environments the libraries do not support locales, wide- or multibyte-character functions, or the **long double** type. (Some VxWorks files may include stubs for unsupported wide-character functions.) For user-mode (RTP) VxWorks projects, the libraries support wide-character and multibyte functions.

ColdFire Processor Support

Direct support for the ColdFire processors 5445x, 5221x, 5223x, 51QEx, and 5222x is not in this release. However, passing the **-Xcpu-all** flag to the assembler will allow all machine instructions for those processors to be recognized.



CAUTION: Use of the **-Xcpu-all** flag will allow *any* ColdFire or 68000 family instruction to be recognized. Care should be taken to not use any instruction that is not supported by the processor that is being targeted.

8.3 Supported Hardware and Software

Intel-Based Solaris Systems

Although the Wind River Compiler may run on Intel-based Solaris systems, it is only supported for SPARC-based machines. No functionality is promised for Intel-based Solaris computers

8.4 Usage Caveats

C and C++ Compilers

Treatment of Invalid Command-Line Options

In most cases, passing an unrecognized option flag to the tools generates a warning. This behavior, however, is not completely consistent. In some cases, no warning is generated; in other cases, such an invalid option may cause compilation to stop.

-Xmismatch-warning Overrides -e Option

-Xmismatch-warning and **-Xmismatch-warning=2** override the **-e** option (used for changing the severity of a message). If either form of **-Xmismatch-warning** is used, mismatched types will only produce a warning, even if **-e** is used to increase the severity level of the diagnostic.

#pragma weak Usage

Using **#pragma weak** may occasionally produce unforeseen behavior. Two instances are worth mentioning:

- **#pragma weak** is incompatible with local data area (LDA) allocation; using **#pragma weak** with **-Xlocal-data-area** or **-Xlocal-data-area-static-only** enabled will produce a warning and temporarily disable LDA.
- While a symbol may be defined in more than one module as long as at most one of the definitions is global while the rest (or all) are weak, the linker resolves references to the first instance of the symbol it encounters. Consider the following scenario. Function **foo()** uses **x**, which is declared weak in

library 1 and global in library 2. If library 1 is searched first, the weak version of *x* will be used. On the other hand, if library 2 is subsequently linked (because, for example, another function uses it), then the global version of *x* will replace the weak version.

Limited Support for RTA Profiling

Only one compiler option for generating Run-Time Analysis profiling information is supported in VxWorks. That option is **-Xrtc**. Other compiler profiling options, such as **-Xprof**, are not supported in VxWorks.

For **-Xrtc**, all mask options are supported except *mask* = 0x4.

Linker

Warnings for Duplicate Definitions

When SDA is enabled, the compiler changes the linkage of some symbols from **COMMON** to **BSS** data. Because of this, the linker may issue a warning if a symbol (such as **int foo**) is defined in more than one place (e.g., **sysLib.c** and **usrConfig.c**), as may be the case with custom BSPs. Although such warnings may not indicate a significant problem, nonetheless it's a good idea to declare the symbol **extern** (e.g., **extern int foo**) in the header file and define the symbol in one C file only.

Far Relative Addressing and VLE

The following applies to Power Architecture code using the VLE (Variable Length Encoding) instruction set.

Programs compiled to use far (32-bit) relative addressing, either for code or data (for example, programs compiled with **-Xcode-far-relative** or **-Xdata-far-relative**), must explicitly reference the symbols **_SDA_BASE** and **_SDA2_BASE**. If these symbols are not referenced anywhere in the program, the linker will generate incorrect code. Specifically, it will try to use "absolute SDA," in which **r0** is used as a base register to indicate a base location of zero. (See the user's guide section on ELF Relocation Information for more on absolute SDA.)

This is not a problem for non-VLE code, where **r0** is interpreted as zero; in contrast, in VLE mode, **r0** is interpreted as the *contents* of **r0**.

8.5 Known Problems

This section lists some known problems with the Wind River Compiler. For a complete list of known problems in the Wind River Compiler, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

Casting in Inline Assembly Code

The compiler fails to sign-extend certain arguments after a cast in inline assembly routines. To fix this problem, use the **-Xold-inline-asm-casting** option.

Multiple -# Options

If two or three **-#** options are present on the command line, the compiler acts as if the **-##** or **-###** option is present, respectively. (**-##** displays subprogram invocation lines with arguments but does not execute any subprogram; **-###** is like **-##** but quotes arguments.)

C++ Trigonometric Functions Do Not Support Complex-Number Objects

Passing instances of the complex-number class template (defined in **complex.h**) to trigonometric functions such as **cos** leads to unresolved symbols.

Using **setjmp()** and **longjmp()** in Mixed ARM and Thumb Code

If you enable interworking (**-Xinterwork**) to compile mixed ARM and Thumb code, there is a limitation to the use of **setjmp()** and **longjmp()**. The **longjmp()** routine does not support switching between 32-bit and 16-bit mode; if you call **setjmp()** in Thumb mode and make a corresponding call to **longjmp()** in ARM mode, the result will be unpredictable. (However, the reverse procedure—calling **setjmp()** in ARM mode and **longjmp()** in Thumb mode—yields correct behavior.) A workaround is to define a routine like the **__common_long_jmp()** shown below and compile it with **-Xinterwork** enabled.

```
void __common_long_jmp()
{
    return;
}
```

Incorrect Type-Extension of Arguments for 68K Targets

For 68K targets, when a character or short integer is passed to a function, the argument is normally extended to 32 bits. But if the function prototype is present and the parameter is explicitly declared in the prototype as a character or short integer, this extension should not occur. In cases where the prototype is available

but appears in a different file from the function call, however, the compiler (incorrectly) performs the extension anyway.

8.6 Documentation Errata

This section lists some documentation errata for documents associated with the Wind River Compiler. For a detailed list of documentation errata for the Wind River Compiler, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

x86 Instruction Set Simulation

Because WindISS does not currently simulate the x86 instruction set, some example programs cannot be compiled and run as documented for x86 targets.

9

Wind River GNU Compiler

- 9.1 Introduction 113
- 9.2 Changes in This Release 114
- 9.3 Usage Caveats 115
- 9.4 Known Problems 118
- 9.5 Documentation Errata 119

9.1 Introduction

The Wind River GNU Compiler comprises C and C++ compilers, linker, assembler, and utilities. The compiler in this distribution is GCC version 4.1.2. The utilities include **binutils** version 2.17 and **make** version 3.80.

Source code for the tools is available on the product CD.

This release supports the GNU tools for the host platforms and target architectures listed in 3. *System Requirements*. Further, Wind River ships and supports only the C and C++ compilers, although other languages are included in the GNU Compiler Collection.

The most extensive information on GCC is available from the Free Software Foundation (FSF) at <http://gcc.gnu.org/releases.html>, which developers are

encouraged to consult. Some of the information on the FSF Web site, however, may not apply to Wind River products, and Wind River does not support all features of GCC. For this reason, you should read these release notes, including *Supported Options*, p. 115, with care.

9.2 Changes in This Release

9.2.1 Enhancements

GCC 4.1.2 provides improved optimization and language support, new target architecture support, bug fixes, and other enhancements. For information about recent GCC improvements, see the FSF Web site at the following URL:

<http://gcc.gnu.org/releases.html>

9.2.2 Fixed Problems

For lists of problems fixed in recent GCC releases, see the FSF Web site at the following URL:

<http://gcc.gnu.org/releases.html>

9.2.3 Unsupported Features

Pentium Flags

The `-mcpu` flag has been replaced by `-mtune`.

Specs File

The specs file has always been built into the driver, but GCC 3.3.2 and earlier also had a copy in the file system. Because the driver reads the copy in the file system if it exists, having this file there made the compiler slower. This copy has now been removed to speed up the compiler.

If the specs file is required for any reason, **gcc -dumpspecs** can be used to generate it. Modifications can then be made to this file. If a specs file exists, the driver will read it and use it instead of the built-in file.

9.3 Usage Caveats

Documentation

Manuals for the GNU tools are based on manuals published by the FSF and may not include Wind River extensions such as command-line switches for RTPs and shared libraries.

Supported Options

The following options have been tested for use in VxWorks development. Some additional architecture-specific options are documented in the *VxWorks Architecture Supplement*.

Preprocessor

-M	-MM
----	-----

Type

-fsigned-char	-funsigned-char
---------------	-----------------

Optimization and Memory

-fcombine-regs	-fomit-frame-pointer	-fvolatile-static
-fcommon	-fpic	-O
-fdefer-pop	-fschedule-insn	-O0
-finline-limit= <i>n</i>	-fstrength-reduce	-O1
-fno-builtin	-funroll-all-loops	-O2
-fno-common	-funroll-loops	-O3
-fno-defer-pop	-fvolatile (C only)	-O4
-fno-unroll-loops	-fvolatile-global	-Os
	--param max-unrolled-insns= <i>n</i>	

Debug Info

-g0	-g2
-g1	-g3

Diagnostics

-w	-Wstrict-prototypes
-Wall	-Wunused
-Werror-implicit-function-declaration	

C++

-fexceptions	-fno-for-scope	-fpermissive
-ffor-scope	-fno-implicit-templates	-frtti
-fimplicit-templates	-fno-rtti	-fweak
-fno-exceptions	-fno-weak	

Miscellaneous

-ansi	-maltivec	-pedantic
-fdollars-in-identifiers	-mrtp	-save-temps
-fpipe	-non-static	-x assembler-with-cpp

Checking for Null Format Arguments

Checking for null format arguments has been decoupled from the rest of the format checking mechanism. Programs that use the **format** attribute may regain this functionality by using the new **nonnull** function attribute. For all functions for which GCC has a built-in **format** attribute, an appropriate built-in **nonnull** attribute is also applied.

Zero-Initialized Variables in .bss

GCC automatically places zero-initialized variables in the **.bss** section on some operating systems, including VxWorks. To disable this optimization, use the **-fno-zero-initialized-in-bss** option.

Strict Aliasing

-fstrict-aliasing is now part of **-O2** and higher optimization levels. This allows the compiler to assume the strictest aliasing rules applicable to the language being compiled; for C and C++, it activates optimizations based on expression types. These optimizations may break old, non-compliant code. If a previously working program fails to recompile with newer versions of GCC, try **-fno-strict-aliasing**.

Enumerations and the ABI

Enumerations are now properly promoted to **int** in function parameters and function returns. Normally this difference is not visible, but when **-fshort-enums** is enabled, it entails an ABI change.

C99

The compiler does not provide full C99 support.

Small Data Area (Power Architecture and MIPS)

VxWorks does not support a small data area in kernel mode or, for MIPS targets, in user mode. When compiling a kernel-mode Power Architecture (PowerPC) project, or any MIPS project, do not specify **-msdata**.

Software Floating Point for Power Architecture Targets

If you build for Power Architecture targets with software floating-point support, it is not sufficient to simply specify (for example) **-mcpu=860** on the command line. You must also specify **-msoft-float**, which tells the driver to link against the correct libraries.

Altivec __vector Type

To use **__vector** in your code, you must first specify **#include <altivec.h>**. (This was not required by GCC 2.96+.)

ARMv6 Preprocessor

The **-march=armv6** option has not been fully tested. It should be avoided or used with caution.

ARM Flags

The abbreviated flags listed in [Table 9-1](#) are recommended in kernel mode and must be used in RTP mode:

Table 9-1 ARM Flags

Abbreviated Flag	Equivalent Old Flag
-t4	-mapcs-32 -mlittle-endian -march=armv4
-t4be	-mapcs-32 -mbig-endian -march=armv4

Table 9-1 **ARM Flags** (cont'd)

Abbreviated Flag	Equivalent Old Flag
-t4t	-mthumb -mthumb-interwork -mlittle-endian -march=armv4t
-t4tbe	-mthumb -mthumb-interwork -mbig-endian -march=armv4t
-t5	-mapcs-32 -mlittle-endian -march=armv5
-t5be	-mapcs-32 -mbig-endian -march=armv5
-t5t	-mthumb -mthumb-interwork -mlittle-endian -march=armv5
-t5tbe	-mthumb -mthumb-interwork -mbig-endian -march=armv5
-txscale	-mapcs-32 -mlittle-endian -mcpu=xscale
-txscalebe	-mapcs-32 -mbig-endian -mcpu=xscale

9.4 Known Problems

This section lists some known problems with the Wind River GNU Compiler. For a complete list of known problems in the Wind River GNU Compiler, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

System Failure During Builds

Crashes have been reported during compilation, especially under Windows 2000. These failures result from faulty memory and are not consistently reproducible. For information and suggestions, see <http://www.bitwizard.nl/sig11/>.

9.5 Documentation Errata

For a detailed list of documentation errata for the Wind River GNU Compiler, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

10

Wind River Network Stack

10.1 Introduction	121
10.2 Changes in This Release	122
10.3 Usage Caveats	125
10.4 Known Problems	126
10.5 Documentation Errata	126

10.1 Introduction

The Wind River Network Stack is an IPv4/IPv6 dual stack.

When Wind River Network Stack 3.1 was replaced by a new design and implementation of the network stack—Wind River Network Stack 6.5—many Workbench components, configuration parameters, and API routines were replaced. Several shell commands were also added. Changes between the 6.5 and 6.6 releases include new features and enhancements, but nothing as extensive as the previous reimplementations of the stack. For more information, see the migration guide for your Platform.

10.2 Changes in This Release

This release of the Wind River Network Stack includes enhancements in the following functional areas:

SMP

VxWorks 6.6 introduces facilities that support symmetric multiprocessing (SMP). However, the Wind River Network Stack does not support ingress filtering in SMP builds.

IPv6-Only Build Option

You can now build the network stack as an IPv6-only stack. The previous configurations of IPv4-only or IPv4/IPv6 are still available.

Mobile IP

- Integration with WiMAX
- Security enhancements:
 - RADIUS
 - Diameter¹
 - MOBIKE (IKE mobility in conjunction with Wind River IKE)
 - Enhanced support for AAA (authorization, authentication, and accounting)

DHCP

The DHCPv6 client and server now allow authentication. (Authentication was not available in the 6.5 release.)

FTP

The hard-coded FTP access account with user name "ftp" and password "interpeak" that, by default, had read/write access to the FTP server in the 6.5 version of the Wind River Network Stack has been removed in this release. The read-only account with user name "anonymous" (without a password) still exists by default.

1. Authentication is supported; accounting is not.

FTP6

The `FTPS_ROOT_DIR` parameter now has the default value of "" (an empty string). In the previous release, the default value was "/" (root).

802.1Q VLAN

1.

You can now use the “compact naming convention” to describe the VLAN pseudo-interface.
2.

The MUX-L2 is now integrated into the stack as an optional component of Wind River VxWorks Platforms.

Quality of Service (QoS)—Differentiated Services (Diffserv)

Support for the Differential Services (Diffserv) model of QoS has been added as a built-in feature of the dual-mode IPv4/IPv6 stack.

MIB Support

Table 11 shows the RFCs that specify the networking-related MIBs that are supported in versions 6.5 and 6.6 of the Wind River Network Stack. The MIBs in RFCs 2011, 2465, and 2466 are now supported in their RFC 4293 forms.

Table 11 MIBs Supported by Wind River Network Stack Versions 6.5 and 6.6

MIB	6.5	6.6
RFC 2011	●	obsolete
RFC 2465	●	obsolete
RFC 2466	●	obsolete
RFC 2863	●	●
RFC 4022	●	●
RFC 4113	●	●
RFC 4292	●	●
RFC 4293		●

For Further Information

For a detailed description of this release of the Wind River Network Stack, see the three Wind River Network Stack programmer's guides:

- *Wind River Network Stack for VxWorks 6 Programmer's Guide, Volume 1: Transport and Network Protocols*
- *Wind River Network Stack for VxWorks 6 Programmer's Guide, Volume 2: Application Protocols*
- *Wind River Network Stack for VxWorks 6 Programmer's Guide, Volume 3: Interfaces and Drivers*

10.2.1 API Changes

IPCOM Routines

The routines `ipcom_run_cmd()` and `ipmcp_cmd()` are deprecated.

10.2.2 Fixed Problems

For a list of problems fixed in the Wind River Network Stack 6.6, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

10.2.3 Deprecated Features

IPCOM Routines

The routines `ipcom_run_cmd()` and `ipmcp_cmd()` are deprecated.

10.3 Usage Caveats

Optimizing Stack Performance

Make the following configuration adjustments to optimize the performance of the network stack:

- To improve forwarding performance with numerous flows, increase the value of `IPNET_IPV4_CACHE_ORDER` or `IPNET_IPV6_CACHE_ORDER` in *installDir/components/ip_net2-6.6/ipnet2/config/ipnet_config.h*. A flow is defined by the tuple {ingress interface index, source address, destination address, IP protocol, TOS} for IPv4 and {ingress interface index, source address, destination address, IP protocol, traffic class, flow label} for IPv6. You must rebuild the networking code libraries after making this change.
- To improve forwarding performance when Wind River NAT and Wind River Firewall are not used, disable them both. Disable the firewall by setting `COMPONENT_FIREWALL` to false in *installDir/vxworks-6.6/config/feature set/config.mk*. Disable NAT by undefining `IPNET_USE_NAT` in *installDir/components/ip_net2-6.6/ipnet2/config/ipnet_config.h*. You must rebuild the networking code libraries after making these changes.
- To improve general networking performance, disable System Viewer support by undefining `INCLUDE_WINDVIEW` in the BSP's `config.h` file.
- To improve general networking performance, raise the number of packet pool buffers by editing the packet pool configuration parameters in Workbench (See `NUM_POOL_1`, `SIZE_POOL_1`, `NUM_POOL_2`, `SIZE_POOL_2`, and so on.)
- If you will not use IPv6, disable it in *installDir/vxworks-6.6/config/feature set/config.mk* and rebuild the networking code libraries.

tNetTask

tNet0. If you are using VxWorks SMP, multiple instances are available and are named **tNet*n***. For more information, see the *Network Drivers* chapter of the *VxWorks Device Driver Developer's Guide, Volume 2: Writing Class-Specific Device Drivers*.

inet_aton Return Values

The return values of `inet_aton` are not compatible with some other BSD socket implementations. `inet_aton` returns OK for success or ERROR for failure, whereas `vxworks.h` defines OK as 0 and ERROR as (-1).

Like other BSD implementations, the internal API `ipcom_inet_aton` returns -1 for success and 0 for failure. However, to maintain backward compatibility for VxWorks socket applications, these values have been reversed by the public wrapper code `inet_aton`.

10.4 Known Problems

For a complete list of known problems in the Wind River Network Stack, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

10.5 Documentation Errata

This section lists some documentation errata for documents associated with the Wind River Network Stack. For a detailed list of documentation errata for the network stack, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

Wind River Network Stack for VxWorks 6 Programmer's Guide, Volume 2: Application Protocols, 6.6

Volume 2 of the programmer's guide incorrectly lists the default value for the `FTPS_ROOT_DIR` parameter as "/" (root). The correct default value is "" (an empty string).

11

Wind River PPP

11.1 Introduction	127
11.2 Changes in This Release	128
11.3 Known Problems	128
11.4 Documentation Errata	128

11.1 Introduction

Wind River PPP 6.6 allows for the establishment, authentication, and control of Point-to-Point Protocol (PPP) connections and PPP over Ethernet (PPPoE) implementations.

The Point-to-Point Protocol provides a standard method for transporting multi-protocol datagrams over point-to-point links. PPP is designed for simple links that transport packets between two peers.

PPP over Ethernet provides the ability to connect a network of hosts over a simple bridging access device to a remote Access Concentrator. In this model, each host utilizes its own PPP stack, so each user is presented with a familiar user interface.

11.2 Changes in This Release

There are no functional changes in this release, which moves Wind River PPP to a new release of VxWorks.

11.2.1 Enhancements

There are no functional enhancements in this release.

11.2.2 Fixed Problems

For a list of problems fixed in Wind River PPP, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

11.3 Known Problems

This section lists some known problems with Wind River PPP. For a complete list of known problems in Wind River PPP, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

RIP Table Not Updated with PPPoE Connection on Windows Vista

On Windows Vista, if you have included PPP, PPPoE, and RIP in your image, the RIP table is not updated when a PPPoE connection is made. (WIND00109416).

11.4 Documentation Errata

This section lists some documentation errata for documents associated with Wind River PPP. For a detailed list of documentation errata for Wind River PPP, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

API Reference

ipppp_config()

In the API reference entry for this routine, the description of the sixth argument is incorrect. The type is **int**, but the accepted values are zero or more of the **IP_FC_FLAGS_***xxx* flags. The **IP_FC_FLAGS_***xxx* should be **or**:ed (using bitwise or "**|**"). (WIND00112177)

Paths in Examples

Some examples may contain the path **ip_net2-6.5**. The path should be **ip_net-6.6**. (WIND00112177)

12

Wind River TIPC

12.1 Introduction	131
12.2 Changes in This Release	132
12.3 Known Problems	135
12.4 Documentation Errata	136

12.1 Introduction

The Transparent Inter Process Communication (TIPC) protocol is a transport protocol that allows socket-based applications to communicate easily and efficiently in both single-node environments and environments containing clusters of nodes. TIPC was originally developed at Ericsson in the 1990s.

Wind River TIPC 1.7 is a port of the open-source Linux 1.7.5 implementation of TIPC (see <http://tipc.sourceforge.net>). This allows the Wind River TIPC and open-source Linux implementations to interoperate within a network.

12.2 Changes in This Release

Wind River TIPC 1.7 adds the following enhancements to the previous release:

- Support for distributed shared memory (DSHM).
- Build options for reducing TIPC's footprint.
- The introduction of a new API, the TIPC native API, that can provide both footprint reduction and faster performance.
- A new test suite for debugging TIPC applications.
- Expanded networking capabilities through support for multiple zones and clusters.
- A new build component for including the sample TIPC application of the previous release in a VxWorks kernel image.
- New command options for the **tipcConfig** utility.

12.2.1 Enhancements

Distributed Shared Memory (DSHM)

For BSPs that support DSHM, a new TIPC configuration option allows you to use DSHM for communication between TIPC nodes. Currently, only the following BSPs support DSHM:

- **hpcNet8641**
- **linux**
- **sb1250**
- **sb1480**
- **simpc**
- **solaris**
- **wrSbc8641d**

For more information, see the *Communication Using Distributed Shared Memory (DSHM)* section of the *Wind River TIPC for VxWorks 6 Programmer's Guide, 1.7: Building VxWorks to Include Wind River TIPC*.

Footprint Reduction

TIPC provides a set of new build parameters that make it possible to reduce TIPC's footprint by excluding any of the following features from a build:

- debugging
 - system messages
 - the **tipcConfig** utility and TIPC show routines
 - the TIPC socket API
- (In this case, you must use the new TIPC native API; see [TIPC Native API](#), p.133.)

TIPC Native API

There is a new API, the TIPC *native* API, that you can use in place of the TIPC socket API. The advantages of the new API are that it has a smaller footprint than the socket API and can improve throughput and response latencies. The disadvantage of the TIPC native API is that it is very different from the standard socket API and must be learned from scratch. The TIPC native API is only available in kernel applications. RTPs must use the TIPC socket API.

The socket API and the native API are not mutually exclusive. If the TIPC socket API is included in a build, you can use both the socket API and the native API in a kernel application.

Note that the TIPC native API has not been finalized by the TIPC Working Group of the Multicore Association (see <http://www.multicore-association.org>) and is still subject to change.

Test Suite

A new TIPC build component provides access to a suite of tests that you can use for debugging. The following tests are available:

- 0 Execute test 1 through 15, in sequence.
- 1 Create a non-reliable, connectionless socket (**SOCK_DGRAM**).
- 2 Create a reliable, connectionless socket (**SOCK_RDM**).
- 3 Create a reliable, connection-oriented socket (**SOCK_STREAM**).
- 4 Create a reliable, connection-oriented socket (**SOCK_SEQPACKET**).
- 5 Shut down a **SOCK_STREAM** connection.
- 6 Shut down a **SOCK_SEQPACKET** connection.

- 7 Test message-size limits using **SOCK_RDM**.
- 8 Test sending of **TIPC_IMPORTANCE** levels with a message (see the reference page for **getsockopt()**).
- 9 Test sending TIPC socket options with a message (see the reference pages for **getsockopt()** and **setsockopt()**).
- 10 Test sending of header ancillary data with **SOCK_SEQPACKET** (reliable, connection-oriented).
- 11 Test sending of header ancillary data with **SOCK_RDM** (reliable, connectionless).
- 12 Test multicast using **SOCK_RDM**.
- 13 Test sending fragmented message using **SOCK_RDM**.
- 14 Test sending large messages (over 66000 Bytes) using **SOCK_STREAM**.
- 15 Test **sendto()** and **recvfrom()** socket routines using **SOCK_RDM**.

Expanded Networking Capabilities

The previous release of Wind River TIPC supported only a single zone and a single cluster. The current release supports multiple zones and multiple clusters.

Build Component for the TIPC Sample Application

The previous release of Wind River TIPC made available a sample application illustrating the use of the Wind River TIPC API. The application is a demonstration program that simulates a store with items for sale and customers who enter the store to purchase items. The current release provides a new build component for the sample application. If you include the build component in a build, the code for the inventory simulation is included as part of your project and compiled as part of your VxWorks image project (VIP).

New tipcConfig Options

Command options have been added to the **tipcConfig** utility:

- **-s**
Displays the current TIPC release number.
- **-V**
Displays the current version of the **tipcConfig** utility.

12.2.2 API Changes

New TIPC Native API

There is a new TIPC *native* API that can be used instead of the TIPC socket API in kernel applications (see [TIPC Native API](#), p.133).

Changed Parameter Requirement for shutdown() Routine

The syntax of the **shutdown()** routine in the TIPC socket API is:

```
STATUS shutdown
(
    int sd,          /* identifies the socket to shut down */
    int how          /* function code */
)
```

In previous releases, only the complete shutdown of a socket was supported and the **how** parameter was ignored. In the current release, only a complete shutdown is supported, but you must now enter the following value for the **how** parameter:

SHUT_RDWR

Any other value for the parameter results in an error.

12.2.3 Fixed Problems

For a list of problems fixed in Wind River TIPC, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

12.3 Known Problems

For a complete list of known problems in Wind River TIPC, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

12.4 Documentation Errata

For a detailed list of documentation errata for Wind River TIPC, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

13

Wind River USB

13.1 Introduction	137
13.2 Changes in This Release	138
13.3 Supported Hardware and Software	139
13.4 Usage Caveats	143
13.5 Known Problems	143
13.6 Documentation Errata	143

13.1 Introduction

The Universal Serial Bus Revision 2.0 (USB 2.0) provides hosts and devices with a versatile channel for communication at low, moderate, and high speeds.

Wind River USB 2.4 is composed of two parts—a USB host stack, which you can use to write your own USB host class driver, and a USB peripheral stack, which allows you to write your own USB target application or USB target controller driver. The USB peripheral stack is the component that interprets and responds to the commands sent by a USB host.

13.2 Changes in This Release

USB 2.4 includes support for VxBus and a bi-directional headset.

13.2.1 Enhancements

With this release of VxWorks, the USB host controller drivers are fully integrated with VxBus.

13.2.2 API Changes

With this release of VxWorks, the host controller drivers were extensively modified to support VxBus. The new API for the host controller drivers is documented in the API reference. This is of interest primarily to engineers involved in BSP porting or in new host controller driver development.

The USB API used to interface to the USB stack is unchanged. This allows class drivers and USB class applications developed for previous releases of VxWorks to be used without modification on this release of Wind River USB.

The API for the USB peripheral stack is unchanged.

13.2.3 Fixed Problems

For a list of problems fixed in Wind River USB, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

13.2.4 Unsupported Features

As of this release of VxWorks, non-VxBus implementations of the host controller drivers are unsupported.

13.3 Supported Hardware and Software

This section lists additional hardware and software that can be used with Wind River USB.

Wind River USB has been modified to work with VxWorks SMP on SMP-capable boards, as listed in [Table 13-1](#).

Table 13-1 **SMP-Capable Boards and BSPs**

Target Architecture	BSP
Broadcom BCM1480 board for MIPS	sb1480_mipsi64
Freescale MPC8641D board	hpcNet8641
Intel Capell Valley board	idp945

[Table 13-2](#) lists the reference boards that the USB host stack supports.

Table 13-2 **Host Stack Supported BSPs**

Target Architecture	BSP
ARM/XScale	ixdp425
ColdFire	m54x5evb
Intel Architecture	pcPentium
	pcPentium2
	pcPentium3
	pcPentium4
MIPS	malta4kc_mips32sf
Power Architecture (PowerPC)	wrSbcPowerQuiccII
SuperH (SH4)	ms7751Rse

USB Peripheral Stack BSPs

Table 13-3 lists the reference boards that the USB peripheral stack supports.

Table 13-3 Peripheral Stack Supported BSPs

Target Architecture	BSP
ARM/XScale	ixdp425
Intel Architecture	pcPentium
	pcPentium2
	pcPentium3
	pcPentium4
MIPS	malta4kc_mips32sf
Power Architecture (PowerPC)	wrSbcPowerQuiccII
SuperH (SH4)	ms7751Rse

13.3.1 Tested Devices

This section lists the devices on which the USB host stack and USB peripheral stack have been tested.

USB Host Stack Devices

The USB host stack has been tested with the following devices:

USB Headset Devices

- GNN GN-9350
- Plantronics CS50-USB
- Logitech 350

USB Host Controllers

- Intel PIIX-3 (UHCI)
- VIA Technology VT6202 (UHCI and EHCI)
- SIIG PCI-to-USB Add-In Card (OPTi FireLink 82C861 with OHCI)

- SIIG USB 2.0 Adapter (NEC 720100 Chipset with EHCI and OHCI)
- SIIG USB 2.0 PCI Add-In Card (NEC 720101 Chipset with EHCI and OHCI)
- USB 2.0 PCI Add-In Card (Philips ISP1561 Chipset with EHCI and OHCI)

USB Hubs

- SIIG USB MiniHub 4000P (Model No. US2228)
- Belkin Hi-Speed USB Pocket Hub (Model No. F5U217)
- D-Link Model No.: DUB-H4

USB Keyboards

- Belkin USB Keyboard
- NavTech USB Keyboard
- QTRONIX Scorpius 980NPIUS/980
- Memorex MX3300 Multimedia Keyboard
- Zebronics USB keyboard Model No: K10
- Dell USB Keyboard Model No: SK - 8125
- iBall USB Keyboard Model No: EZ9900

USB Mice

- Belkin Mouse (Model No. F8E201-USB)
- Kensington Mouse-in-a-Box USB (Model #MOSUU B)
- Microsoft Wheel Mouse
- Microsoft Basic Optical Mouse

USB Mass Storage Class Devices**CBI Implementation**

- IBM USB Portable Diskette Device
- VST Technologies USB Floppy Drive
- Sony USB Floppy Disk Driver Model No: MPF88E

Bulk-Only Implementation

- Transcend JetFlash 256 MB Model No.TS256MJF2B
- TwinMOS Mobile Disk Z4
- EagleTec 128MB
- Kingston 256MB Data traveler flash disk
- iBall USB 2.0 256MB
- Frontech Genius Driver USB 1.1 Flash Disk 256 MB Model No. JIL-1210
- Lexar Jump Driver

- Zion 256 MB USB Flash Disk
- Western Digital HardDrive (Firewire/USB 2.0 Combo) WD2000B006-RNN
- Maxtor 120 GB Hard Driver (Firewire/USB 2.0)
- Iomega HDD 20GB USB 2.0 Portable Hard Drive Model No: IPHDUSB2
- Card Drive USB Multiple Card Reader/Writer
- M-Systems FlashOnChip
- Fuji 256 MB Flash-on-Chip
- Universal Smart Drive (Bulk-Only Implementation)
- Sony 64MB USB 2.0 USM64A
- Fujitsu-Siemens Memorybird PD-256 flash disk
- MyFlash 128MB
- Pretec 128MB flash disk
- Ur-Disk USB 1.1 Key 256MB

USB Network Devices

- Belkin 10/100 Ethernet Adapter F5D5050 (PegasusII-ADM Tek AM8511 chip)
- SOHOware 10/100 USB Network Adapter NUB110 (PegasusII- AM8511 chip)

USB Speaker Devices

- iMic USB audio interface with normal speaker
- Sony SRS-T100PC USB Active Speaker System

USB Printer Devices

- Epson Stylus C86 Inkjet USB Printer

USB Peripheral Stack Controllers

The USB peripheral stack has been tested using the following USB peripheral controllers:

- Philips PDIUSBBD 12 evaluation kit
(Uses ISA, supported only on Pentium.)
- Philips ISP1582 evaluation kit
(Uses PCI, supported on the architectures listed in [Table 13-2](#), except for ARM/XScale and Power Architecture.)
- Netchip NET2280
(Uses PCI, supported on the architectures listed in [Table 13-3](#).)

13.4 Usage Caveats

Non-VxBus-Compliant BSPs

With this release of Wind River USB, support for non-VxBus-compliant BSPs is discontinued. Wind River USB now runs exclusively on VxBus-compliant BSPs. For information on migrating your BSP to be VxBus-compliant, see the *VxWorks BSP Developer's Guide*.

13.5 Known Problems

For a complete list of known problems in Wind River USB, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

13.6 Documentation Errata

For a detailed list of documentation errata for Wind River USB, visit the Online Support Web site (see [1.6 Latest Release Information](#), p.4).

CUSTOMER SERVICES

Wind River is committed to meeting the needs of its customers. As part of that commitment, Wind River provides a variety of services, including training courses and contact with customer support engineers, along with a Web site containing the latest advisories, FAQ lists, known problem lists, and other information resources.

Customer Support

For customers holding a maintenance contract, Wind River offers direct contact with a staff of technical support engineers experienced in Wind River products. A full description of the Customer Support program is provided in the *Customer Support User's Guide* available at the following Web site:

<http://www.windriver.com/support>

The *Customer Support User's Guide* describes the services that Customer Support can provide, including assistance with installation problems, product software, documentation, and service errors.

You can reach Customer Support using either e-mail or telephone as follows:

Location	Phone	E-mail
North and South America, Asia/Pacific (outside Japan)	800-872-4977 (toll-free)	support@windriver.com
Europe, Africa, Middle East	+(00) 800-4977-4977 (toll-free)	support-EC@windriver.com
Japan	81-3-5778-6001 (direct)	support-jp@windriver.com

For more detailed contact information, including contact information specific to your products, please see the Support Web site shown above.

Wind River Online Support

Wind River Customer Services also provides Wind River Online Support, an online service available under the Support Web site. This is a basic service to all Wind River customers and includes advisories, online manuals, and a list of training courses and schedules. For maintenance contract holders, Online Support also provides access to additional services, including known problems lists, available patches, answers to frequently asked questions, and demo code.