

# Wind River® VxWorks® 653 Platform

RELEASE NOTES

2.2

---

Copyright © 2007 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, the Wind River logo, Tornado, and VxWorks are registered trademarks 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

<b>1</b>	<b>Overview .....</b>	<b>1</b>
1.1	Introduction .....	1
1.2	Features .....	2
	VxWorks 653 .....	2
	VxWorks 653 Configuration and Build Tools .....	2
	Wind River DO-178B Network Stack for VxWorks 653 .....	2
	Wind River Workbench .....	3
	GNU Compiler Collection .....	3
	Workbench Plug-in for On-chip Debugging .....	3
1.3	Package Contents .....	4
1.4	Before You Install .....	4
<b>2</b>	<b>Changes in This Release .....</b>	<b>5</b>
2.1	Introduction .....	5
2.2	Enhancements .....	6
	BSPs .....	6
	VxWorks 653 .....	6
	VxWorks 653 Configuration and Build Tools .....	6
	Wind River DO-178B Network Stack for VxWorks 653 .....	7
	Wind River Workbench .....	7
	Wind River Workbench for On-Chip Debugging .....	7

2.3	Unsupported Features .....	7
<b>3</b>	<b>System Requirements .....</b>	<b>9</b>
3.1	Introduction .....	9
3.2	Host Requirements .....	9
3.3	Target Requirements .....	11
<b>4</b>	<b>Known Problems .....</b>	<b>13</b>
4.1	Introduction .....	13
4.2	Usage Caveats .....	14
	Setting Environment Variables (Solaris) .....	14
	Wind River Registry (Windows) .....	15
4.3	Known Problems .....	15
<b>5</b>	<b>GCC .....</b>	<b>17</b>
5.1	Introduction .....	17
5.2	Changes in This Release .....	18
5.3	Usage Caveats .....	18
	5.3.1 Documentation .....	18
	5.3.2 Supported Options .....	18
	5.3.3 Unsupported Features .....	20
5.4	Known Problems .....	22

<b>6</b>	<b>VxWorks 653 .....</b>	<b>23</b>
6.1	Introduction .....	23
6.2	Changes in This Release .....	24
6.2.1	Enhancements .....	24
	Application Multiplexed I/O .....	24
	ARINC 653 Supplement 2, Part 1 .....	24
	Break on Data Access with Value .....	24
	C++ Cert Applications .....	25
	Configuration Record Binary Files .....	25
	Direct-Access Ports .....	25
	Hardware Support .....	25
	Partition Direct-Access Ports .....	25
6.2.2	API Changes .....	26
	Port Monitoring Routines .....	26
	VAL Routines .....	26
6.2.3	Fixed Problems .....	26
6.2.4	Unsupported Features .....	26
6.3	Usage Caveats .....	27
6.4	Known Problems .....	27
6.5	Documentation Errata .....	27
	target.nr for the hpcNet8641 BSP .....	27
	VxWorks 653 Programmer's Guide .....	28
<b>7</b>	<b>VxWorks 653 Configuration and Build Tools .....</b>	<b>31</b>
7.1	Introduction .....	31
7.2	Changes in This Release .....	32
7.2.1	Enhancements .....	32
	Compiler Options in Makefiles .....	32
	Configuration and Build .....	32
	Manuals .....	34
	Shared Libraries and Applications .....	34

	Shared Library Interfaces .....	34
	.sym Files .....	34
	Unused XML Attributes .....	35
	VerIMAX .....	35
7.2.2	Fixed Problems .....	35
<b>7.3</b>	<b>Known Problems .....</b>	<b>35</b>
	hello-ace and Schema Location .....	35
	hello-version Build Error (Windows) .....	36
<b>7.4</b>	<b>Migration Information .....</b>	<b>36</b>
	Build Settings .....	36
	Generating .sym Files .....	37
	Makefiles .....	38
	Namespace Identifiers .....	38
	Shared Data Regions and Access Rights .....	38
	Shared Libraries and Applications .....	39
	Shared Libraries for Partition OSs .....	44
<b>7.5</b>	<b>Documentation Errata .....</b>	<b>44</b>
	VxWorks 653 Configuration and Build Guide .....	44
<b>8</b>	<b>Wind River DO-178B Network Stack for VxWorks 653 .....</b>	<b>45</b>
<b>8.1</b>	<b>Introduction .....</b>	<b>45</b>
<b>8.2</b>	<b>Usage Caveats .....</b>	<b>46</b>
	Priority of Polled Ethernet Driver .....	46
	SO_USELOOPBACK Socket Option .....	46
<b>8.3</b>	<b>Known Problems .....</b>	<b>46</b>
<b>8.4</b>	<b>Documentation Errata .....</b>	<b>47</b>

<b>9</b>	<b>Wind River Workbench .....</b>	<b>49</b>
9.1	Introduction .....	49
9.2	Changes in This Release .....	50
9.2.1	Enhancements .....	50
	Debugger .....	50
	Host Shell (WindSh) .....	50
	Launch Configuration .....	51
	Static Analysis .....	51
	Target Manager .....	52
	Using Workbench in an Existing Eclipse Environment .....	52
	Views .....	52
	Wind River System Viewer .....	53
9.2.2	Fixed Problems .....	53
9.3	Usage Caveats .....	53
	Host-Related Issues (Windows) .....	53
	WindSh Protection-Domain Commands for VxWorks 653 .....	53
	Workbench Issues .....	54
9.4	Known Problems .....	61
	Configuration and Build .....	61
	Debugging Issues .....	61
	Editor Issues .....	63
	Properties View Issues .....	64
	Target Manager Issues .....	64
	Workbench Issues .....	65
9.5	Documentation Errata .....	68
<b>10</b>	<b>Wind River Workbench for On-Chip Debugging .....</b>	<b>69</b>
10.1	Introduction .....	69
10.2	Unsupported Features .....	70
	BSPs .....	70
	diab Compiler .....	70
	Level of Support .....	70

	Standalone Projects .....	70
	Tutorials .....	70
10.3	Usage Caveats .....	71
	Boot ROM Filenames .....	71
	BSP-Specific Information .....	71
	Examples .....	71
	Pathnames .....	71
	Customer Services .....	73



# 1

## Overview

1.1	Introduction	1
1.2	Features	2
1.3	Package Contents	4
1.4	Before You Install	4

### 1.1 Introduction

These release notes are for Wind River VxWorks 653 Platform 2.2, which is a platform for developing safety-critical software applications that can be certified to Level A of the DO-178B avionics software standard.



---

**NOTE:** Wind River VxWorks 653 Platform (VxWorks 653 Platform) is the name of what was previously called Wind River Platform for Safety Critical ARINC 653 (Platform SC ARINC 653). The names of products that make up the Platform have not changed.

---

## 1.2 Features

VxWorks 653 Platform 2.2 includes the following:

### **VxWorks 653**

VxWorks 653 2.2 is a real-time operating system that complies with ARINC 653 and its APEX (Application Executive) API. VxWorks 653 has the following features:

- spatial and temporal partitioning
- ability to configure subsets that can be certified to Level A of the DO-178B avionics software standard
- ARINC 653 port mechanism for interpartition communication
- cold and warm restart of partitions
- warm restart of the core OS
- health monitoring
- support for Ada (through a Wind River partner) and C/C++ languages for ARINC 653 (APEX) and POSIX APIs
- tools to monitor performance, port usage, and memory usage

### **VxWorks 653 Configuration and Build Tools**

Configuration and build tools configure VxWorks 653 systems and build them for your target hardware.

### **Wind River DO-178B Network Stack for VxWorks 653**

The optional Wind River DO-178B Network Stack for VxWorks 653 1.0 (DO-178B Network Stack) must be ordered separately. It includes the following:

- UDP/IPv4 network stack that is certifiable to Level A of the DO-178B avionics software standard
- compatibility with the core OS of VxWorks 653
- build-time configuration using the VxWorks 653 configuration tools

- ability to add support for various protocols
- various debugging utilities

### **Wind River Workbench**

Wind River Workbench 2.6.1 (Workbench) development suite helps you establish and manage host-target communication, and run, debug, and monitor VxWorks 653 systems.

### **GNU Compiler Collection**

The GNU Compiler Collection (GCC) version 3.3.2 supports C and C++ development.

### **Workbench Plug-in for On-chip Debugging**

The optional Workbench plug-in for on-chip debugging must be ordered separately. It adds the following products to Workbench:

- Wind River ICE SX emulator
- Wind River Probe emulator
- Wind River Trace tool

The products can be used to bring up boards, program flash, and test. This emulation system is effective during the entire development process, even before board-level peripherals are stable. The system is useful for board designers as they create new target boards, write device drivers for them, and write BSPs.

## 1.3 Package Contents

The *Installation Kit*, which accompanies your shipment, includes:

- One DVD with images for all the products you ordered. If you ordered the DO-178B Network Stack, it is included in the VxWorks 653 image. If you ordered the Workbench plug-in for on-chip debugging, it is included in the Workbench image.
- One CD with the *Wind River VxWorks 653 Platform Getting Started*.
- Printed documentation, depending on your license:
  - *Developer Install Guide*
  - *License Administrator Install Guide*

After you install, other documentation is available from online help. For a list of documentation and how to access it, see *Wind River VxWorks 653 Platform Getting Started*.

## 1.4 Before You Install

Before you install, go to the Wind River Online Support Web site for software patches, documentation updates, lists of fixed problems, or other information that may apply. For information on accessing the Online Support Web site, see [Customer Services](#), p.73.

# 2

## *Changes in This Release*

- 2.1 Introduction 5
- 2.2 Enhancements 6
- 2.3 Unsupported Features 7

### **2.1 Introduction**

This chapter provides an overview of major features that are new to VxWorks 653 Platform 2.2 and major items that are no longer available. Comparison is with respect to the previous 2.x release: Wind River Platform for Safety Critical ARINC 653 2.1.

## 2.2 Enhancements

The Platform now includes the following new features for the following parts of the Platform.

### BSPs

The following new BSPs are supported:

- hpcNet8641 (MPC8641D, single core only)
- wrSbc8349E
- wrSbc8560 (MPC8560)
- wrSbcPowerQuiccII (MPC8270)

### VxWorks 653

VxWorks 653 adds supports for the following:

- ARINC 653 Supplement 2, Part 1, *Required Services*. (It previously supported Supplement 1.)
- Port mapping (as defined by ARINC 653), including the ability to have pseudo-ports in the core OS (direct access and not) and pseudo-ports in partitions. Port mapping supports communication outside the ARINC module.
- Application multiplexed I/O.
- Ability to create C++ applications that are certifiable to Level A of the DO-178B avionics software standard.

For details, see [6. VxWorks 653](#).

### VxWorks 653 Configuration and Build Tools

The XML-based configuration and build tools are improved. They now provide a partitioned facility that lets you configure and build a VxWorks 653 system without the complete set of pieces. The tools also now let you build and test applications independently of each other. In addition, use cases for a configuration and build reference process are now in the installation and are fully documented. For details, see [7. VxWorks 653 Configuration and Build Tools](#).

### **Wind River DO-178B Network Stack for VxWorks 653**

The Wind River DO-178B Network Stack for VxWorks 653 is a brand-new product and is ordered separately for the Platform. It is a UDP/IPv4 stack that is certifiable to Level A of the DO-178B avionics software standard.

For details, see [8. Wind River DO-178B Network Stack for VxWorks 653](#).

### **Wind River Workbench**

Workbench 2.6.1 adds supports for the following:

- debugging enhancement to support break on data access with value through a WindSh command
- additional WindSh commands that are qualified

For details, see [9. Wind River Workbench](#).

### **Wind River Workbench for On-Chip Debugging**

Through a separately orderable Workbench plug-in, VxWorks 653 Platform now supports kernel-level and single-partition on-chip debugging. For details, see [10. Wind River Workbench for On-Chip Debugging](#).

## **2.3 Unsupported Features**

Because the associated hardware is obsolete, the following BSPs are no longer supported:

- mv5100
- wrSbc7447





# 3

## *System Requirements*

<a href="#">3.1</a>	<a href="#">Introduction</a>	<a href="#">9</a>
<a href="#">3.2</a>	<a href="#">Host Requirements</a>	<a href="#">9</a>
<a href="#">3.3</a>	<a href="#">Target Requirements</a>	<a href="#">11</a>

### **3.1 Introduction**

This chapter lists minimum requirements for running VxWorks 653 Platform when the host and target are separate computers.

### **3.2 Host Requirements**

[Table 3-1](#) lists minimum host requirements. The requirements do not take into account other programs that may be installed or running on the host.

Table 3-1 Minimum Host Requirements

	Windows Hosts	Solaris Hosts
OS	Windows 2000 Professional with service pack 4 Windows XP Professional with service pack 1	Solaris 8 (32-bit mode only) Solaris 9 (32-bit mode only) (Both with run-time patches as Sun recommends at <a href="http://www.sun.com">www.sun.com</a> )
CPU	Intel Pentium-class processor, 1.2 GHz	Blade 100, 500 MHz workstation
RAM	512 MB (1 GB recommended)	512 MB (1 GB recommended)
Disk space for the recommended (default) installation (See <a href="#">Disk Space</a> , p.11)	1.1 GB	1.9 GB
Browser	Any standards-compliant Web browser	Any standards-compliant Web browser
Access rights	Administrator	User
TCP/IP installed and enabled (See <a href="#">TCP/IP</a> , p.11)	Yes	(Installed and enabled by default)
Window manager	N/A	CDE
DVD drive or network access (for installation)	Yes	Yes
Active Internet connection (for installation)	Yes	Yes

**Disk Space**

When you calculate the disk space you need for an installation other than the recommended (default), include space for the following:

- additional products (or parts of products) offered at installation time and their source code if applicable
- source code for contributed packages
- your own applications
- your development code

**TCP/IP**

Because Workbench and some of its associated tools use TCP/IP to communicate with each other, TCP/IP is required even if the host is not connected to a network, but is connected to the target through a serial connection.

## **3.3 Target Requirements**

The following are required for a target:

- Supported target board.  
VxWorks 653 Platform supports the PowerPC architecture. The following BSPs are supported:
  - hpcNet8641 (single core only)
  - simpc
  - wrSbc750gx
  - wrSbc7457
  - wrSbc834x
  - wrSbc8560
  - wrSbcPowerQuiccII
- To support a RAM payload for a VxWorks 653 module with 255 partitions: 1 GB of RAM.
- Ethernet cable or null crossover cable (for initial board setup).
- RS-232 null modem cable (for initial board setup).



# 4

## Known Problems

- [4.1 Introduction 13](#)
- [4.2 Usage Caveats 14](#)
- [4.3 Known Problems 15](#)

### 4.1 Introduction

This chapter outlines known problems, usage caveats, and documentation errata for using VxWorks 653 Platform as a whole.

**For Similar Info About This Part of  
the Platform:**      **See:**

---

GNU compiler collection	<a href="#">5. GCC</a>
VxWorks 653	<a href="#">6. VxWorks 653</a>
VxWorks 653 configuration and build tools	<a href="#">7. VxWorks 653 Configuration and Build Tools</a>
Wind River DO-178B Network Stack for VxWorks 653	<a href="#">8. Wind River DO-178B Network Stack for VxWorks 653</a>

For Similar Info About This Part of the Platform:	See:
Workbench	<a href="#">9. Wind River Workbench</a>
Workbench plug-in for on-chip debugging	<a href="#">Wind River Workbench On-Chip Debugging Release Notes, 2.6.1</a>
	<a href="#">10. Wind River Workbench for On-Chip Debugging</a>

For the latest list of known problems, go to:  
<https://portal.windriver.com/windsurf/index.html>  
and select the Wind River VxWorks 653 Platform product.

## 4.2 Usage Caveats

### Setting Environment Variables (Solaris)

To set up the command-line environment on Solaris, run the following command from your installation directory:

```
wrenv.sh -p vxworks653-2.2
```

However, on some Solaris hosts, local environment variables override the values that **wrenv** defines. To avoid this problem, enter the following command from your installation directory:

```
eval `./wrenv.solaris -p vxworks653-2.2 -o print_env -f csh`
```

To start Workbench before your environment is set up, do the following:

1. Open an xterm window.
2. Change to your installation directory.
3. Run **startWorkbench.sh**.

## Wind River Registry (Windows)

If you install VxWorks 653 Platform on a Windows host while logged on with administrator rights (as is recommended) and then start Workbench while logged on with restricted rights, the Wind River Registry displays a message similar to the following and then shuts down:

```
wtxregd (wtxregd@VAN-CINTO): Thu Jun 14 11:42:13 2007
Wind River Registry version: 4.0.7.12
WTX Library version: 4.0.7.12
Wind River Registry initializing ... succeeded.
Using database file c:/2.2/.wind/wtxregd.MY-PC
Error: Wind River Registry restoring data base from file ... failed.
Error: Can not delete file :
c:/2.2/.wind/wtxregd.MY-PC (Permission denied)
Wind River Registry will exit
```

If the Wind River Registry is not running, you cannot use Workbench to access a target. As well, this issue causes read- and write-permission errors in the current workspace. A typical error message is as follows:

```
workspace in use
```

To avoid these problems, start the Wind River Registry before you start Workbench. To do so, from your program list, select in this order:

- **Wind River > Wind River Workbench 2.6 > Registry**
- **Wind River > Wind River Workbench 2.6 > Wind River Workbench 2.6**

## 4.3 Known Problems

### VxWorks 653 Configuration and Build

The VxWorks 653 configuration and build system is not compatible with the Workbench configuration and build system.

Although **VxWorks 653** is listed as a project type in the Workbench **New Project** wizard, selecting it opens an error dialog box. If you continue, you will not be able to create a project. To configure and build a VxWorks 653 system you must use the command-line tools that accompany VxWorks 653.





# 5

## ***GCC***

*(GNU Compiler Collection)*

<b>5.1</b>	<b>Introduction</b>	<b>17</b>
<b>5.2</b>	<b>Changes in This Release</b>	<b>18</b>
<b>5.3</b>	<b>Usage Caveats</b>	<b>18</b>
<b>5.4</b>	<b>Known Problems</b>	<b>22</b>

## 5.1 Introduction

GCC (GNU Compiler Collection) includes C and C++ compilers, linker, assembler, and utilities. The following are included on the installation DVD:

- C/C++ compilers version 3.3.2 (in the GCC image)
- **binutils** version 2.14 (in the GCC image)
- **make** version 3.80 (in the Workbench image)

Source code for the tools is in the GCC image, but is not installed.

This release supports GCC for the host platforms and target architectures listed in [3. System Requirements](#).

## 5.2 Changes in This Release

VxWorks 653 Platform 2.2 includes the same version of GCC that accompanied Platform SC ARINC 653 2.1, but with patches that add these compiler options:

- **-mvthreads**

Use this option to compile all shared libraries and applications, including those based on COIL.

- **-membedded**

Use this option to compile all C++ cert applications (use in addition to the **-mvthreads** option). The **-membedded** option removes support for exceptions and Run Time Type Information (RTTI).

For more information, see [Compiler Options in Makefiles](#), p.32

## 5.3 Usage Caveats

### 5.3.1 Documentation

The most extensive information on GCC is available from the Free Software Foundation (FSF) at <http://gcc.gnu.org>, which you are encouraged to consult. However, some of the information may not apply to VxWorks 653 Platform, and the Platform does not support all GCC features. For this reason, read these release notes with care, especially [5.3.2 Supported Options](#), p.18.

### 5.3.2 Supported Options

The following options are supported for VxWorks 653 in general. However, some options are not supported for C++ applications that are certifiable to Level A of the DO-178B avionics software standard. For more information, see the *VxWorks 653 Programmer's Guide: Developing C++ Applications*.

## C++

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

## Debug Information

- -g0 .. -g3

## Diagnostics and Lint

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

## Miscellaneous

- -ansi
- -fdollars-in-identifiers
- -fpipe
- -mvthreads
- -pedantic
- -save-temps
- -x assembler-with-cpp

## Optimization and Memory

- -fcombine-regs
- -fcommon
- -fdefer-pop
- -finline-limit=*n*
- -fno-builtin

- **-fno-common**
- **-fno-defer-pop**
- **-fno-unroll-loops**
- **-fomit-frame-pointer**
- **-fpic**
- **-fschedule-insn**
- **-fstrength-reduce**
- **-funroll-all-loops**
- **-funroll-loops**
- **-fvolatile** (C only)
- **-fvolatile-global**
- **-fvolatile-static**
- **-O**
- **-O0 .. -O4**
- **-Os**
- **--param max-unrolled-insns=//**

#### **Preprocessor**

- **-M**
- **-MM**

#### **Type**

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

### **5.3.3 Unsupported Features**

VxWorks 653 Platform does not support the following GCC features.

#### **-A- Switch**

The preprocessor does not recognize the **-A-** switch used alone. **-A-** followed by an assertion is supported.

#### **C99**

The compiler does not provide full C99 support. Portions of the supplied standard libraries that depend on unsupported C99 features (such as complex data types) have been removed.

**-fvolatile**

The **-fvolatile** flag is not supported for C++. Use the **volatile** keyword instead.

**Multiline String Literals**

The preprocessor does not accept multiline string literals.

**Multiple typedefs**

The C compiler does not support multiple **typedefs**. For example, the following code produces an error:

```
typedef int myint;  
typedef int myint;
```

**Naming Types**

The C compiler does not support the naming-types extension:

(**typedef foo = bar**)

Instead, use **typeof**, as in:

**typedef typeof(bar) foo**

**PowerPC AltiVec Usage**

VxWorks 653 does not support AltiVec.

**Prefixed Underscore**

For the PowerPC, the compiler does not add the underscore prefix to symbols. In other words, **symbol** is not equivalent to **\_symbol** as it is for some other architectures.

**Small Data Area**

VxWorks 653 does not support a small data area in kernel mode. When compiling a kernel-mode project, do not specify **-msdata**.

**-traditional**

The **-traditional** C compiler option is not supported. (Traditional preprocessing remains available.)

**weak and alias Attributes**

The **weak** and **alias** attributes are not supported.

## 5.4 Known Problems

This section lists some known problems with GCC. For the latest list, see the Online Support Web site.

### **Make and .bat Files**

If you include a **.bat** file in a makefile, make does not wait for the **.bat** file to finish running before processing the next command in the makefile. Do not run **.bat** files in makefiles.

# 6

## *VxWorks 653*

- 6.1 Introduction 23
- 6.2 Changes in This Release 24
- 6.3 Usage Caveats 27
- 6.4 Known Problems 27
- 6.5 Documentation Errata 27

### **6.1 Introduction**

VxWorks 653 is an RTOS that supports creating safety-critical applications that conform to the ARINC 653 specification and that can be certified to Level A of the DO-178B avionics software standard.

## 6.2 Changes in This Release

This section outlines the changes in VxWorks 653 2.2 compared to VxWorks 653 2.1.

### 6.2.1 Enhancements

VxWorks 653 2.2 includes the following enhancements and new features.

#### Application Multiplexed I/O

Application multiplexed I/O lets you provide input to and view output from multiple partitions over a single serial connection.

#### ARINC 653 Supplement 2, Part 1

VxWorks 653 now complies with ARINC 653, Supplement 2, Part 1, *Required Services*. (It previously complied with Supplement 1.)

#### Break on Data Access with Value

As you debug, using the target shell (or the WindSh host shell), you can set a break point on a particular data address. You can now qualify it with a value. The breakpoint is hit only if the value at that address changes to the specified value. The new commands are:

- **bhv** (for the core OS)
- **pdbhv** (for partitions)

The following routine has been added to VxWorks 653 to support this feature:

- **wtxHwBreakpointValueCheckAdd()**

For details, see the reference entry for the **wtx** C library.



## C++ Cert Applications

You can now create C++ applications that are certifiable to Level A of the DO-178B avionics software standard.

## Configuration Record Binary Files

Configuration record binary files (**configRecord.bin** and **configRecord.reloc**) now include a checksum that you can use to determine whether the files have been corrupted. For information, see [Verifying Configuration Record Binary Files](#), p.28.

## Direct-Access Ports

VxWorks 653 now supports direct-access ports. A direct-access port is a type of pseudo-port (in the core OS) that does not use software buffering. Buffering support is assumed to be provided by the communications hardware.

## Hardware Support

VxWorks 653 adds support for the following hardware:

- hpcNet8641 (single core only)
- wrSbc8349E
- wrSbc8560 (MPC8560)
- wrSbcPowerQuiccII (MPC8270)

## Partition Direct-Access Ports

VxWorks 653 now supports partition direct-access ports. A partition direct-access port is a type of direct-access port that resides in a partition. This type of port can communicate only with a local port in the application resident in the partition. It lets an application communicate outside the module using a port driver in the partition OS. Partition direct-access ports use APEX port routines.

## 6.2.2 API Changes

### Port Monitoring Routines

The synopsis for the core OS **portMonitor( )** routine in **apexPortLib** has changed to let sampling ports be monitored, where only queuing ports could be monitored previously.

### VAL Routines

User-supplied code that runs in the core OS of VxWorks 653 can now use the following VAL routines:

- **valPseudoInt( )**
- **valShow( )**
- **valShowInit( )**

## 6.2.3 Fixed Problems

For the latest list of VxWorks 653 problems that have been fixed, see the Online Support Web site.

## 6.2.4 Unsupported Features

Because the associated hardware is obsolete, VxWorks 653 no longer supports the following BSPs:

- **mv5100**
- **wrSbc7447**

## 6.3 Usage Caveats

The following are caveats for using VxWorks 653.

### Ada Code That Calls C Routines

Calls from Ada to C routines must comply with the PowerPC EABI standard.

### Data Access in Shared Libraries

Access to data in shared libraries is by accessor functions only. This is true for shared libraries supplied by VxWorks 653 and for user-supplied libraries.

6

## 6.4 Known Problems

For the latest list of known problems, see the Online Support Web site.

## 6.5 Documentation Errata

This section lists errata for documentation associated with VxWorks 653. For the latest list, see the Online Support Web site.

### **target.nr for the hpcNet8641 BSP**

The **target.nr** file for the hpcNet8641 BSP is located here in your installation:

**vxworks653-2.2/target/config/hpcNet8641**

In the file, *Flashing the Boot ROM Image Using Workbench*, step e), states the following and specifies an incorrect address (shown below in *italics*):

e) Converting the bootrom.hex file to bootrom.bin.

Select the Add/Remove tab in the flash programmer. Click “convert file” and navigate to the boot loader project you created previously e.g. WindRiver\workspace\hpcNet8641BootProj\bootrom.hex. Select the project. The start address should be 0x0 and the end address should be set to 0xffffffff. Click “convert and add” to convert the file.

At this point, the file is added to the list. Click on the start address entry (should be 0x0) and change it to 0xffff0000. The file is now ready for programming.

In the last paragraph, the start address entry should be changed to 0xffff00100.

## VxWorks 653 Programmer's Guide

The *VxWorks 653 Programmer's Guide* should include the following information.

### Supervisor-Level Driver Information

Chapter 7. *Programming in the Core OS: Setting up Communication with Other Modules*, should include the following information for the supervisor-level driver.

The **availableRtn( )** routine and the **canAcceptMore** flag are essentially the same.

The **availableRtn( )** routine is called only during a partition switch. The return value is used to keep track of which pseudo-ports are available for use. The routine returns **TRUE** if the pseudo-port is capable of performing the read or write operation (the source port has room to send or the destination port has at least one message available for reading).

The **canAcceptMore** flag is returned on every read or write operation to give immediate feedback to the calling read or write routines as to whether a subsequent operation will succeed. The driver sets the flag to **TRUE** if the pseudo-port is capable of performing the read or write operation (the source port has room to send or the destination port has at least one message available for reading).

### Verifying Configuration Record Binary Files

A CCITT CRC-32 checksum at the end of the configuration record binary files (**configRecord.bin** and **configRecord.reloc**) lets you determine whether the binary file was corrupted after the configuration and build tools generated the file.

The checksum is appended with the most-significant byte first. The checksum is useful if you are writing your own loader.

To determine whether **configRecord.bin** or **configRecord.reloc** is corrupted, use the **addToCrc32( )** routine below to perform a byte-wise checksum over the entire file, including its checksum. For the first byte's initial checksum, use 0xFFFFFFFF. If the resulting checksum is zero, the file is not corrupted.

#### **addToCrc32( )**

```
unsigned long addToCrc32(unsigned long crc, unsigned char byte)
{
    unsigned char i;

    crc ^= (byte << 24);

    for(i=0;i<8;i++)
    {
        if(crc&0x80000000)
        {
            crc = (crc << 1) ^ 0x04C11DB7;
        }
        else
        {
            crc <<= 1;
        }
    }

    return crc;
}
```



# 7

## *VxWorks 653 Configuration and Build Tools*

- [7.1 Introduction 31](#)
- [7.2 Changes in This Release 32](#)
- [7.3 Known Problems 35](#)
- [7.4 Migration Information 36](#)
- [7.5 Documentation Errata 44](#)

### **7.1 Introduction**

The VxWorks 653 configuration and build system is a partitioned facility in that its command-line tools let you configure and build a VxWorks 653 system without the complete set of pieces. They also let you build and test applications independently of each other. Configuration is based on XML files.

## 7.2 Changes in This Release

This section outlines the changes to the VxWorks 653 configuration and build facility compared to the facility for VxWorks 653 2.1.

### 7.2.1 Enhancements

#### Compiler Options in Makefiles

If you use the makefiles that accompany VxWorks 653, **Makefile.vars** sets up the appropriate compiler options for compiling shared libraries, applications, and C++ cert applications.

However, if you choose not to use the VxWorks 653-supplied make rules in your makefiles, you need to specify compiler options as follows:

- For all shared libraries and applications (including ones based on COIL), use **-mvthreads**.
- For C++ cert applications, use **-mvthreads** and **-membedded**. The **-membedded** option removes support for exceptions and Run Time Type Information (RTTI).

#### Configuration and Build

The XML-based configuration of VxWorks 653 is substantially redesigned.

##### Build Settings

Build settings that were previously specified in XML configuration files are now specified in makefiles. For migration information, see [Build Settings](#), p.36.

##### Ports and Port Names

Ports are now configured with partitions and no longer with the core OS. Port names are no longer case-sensitive.

##### Reference Process Use Cases

The configuration and build tools now include a set of example reference use cases for configuring and building all the major components of a VxWorks 653 system.



They are installed with the Platform and described in the *VxWorks 653 Configuration and Build Guide*.

### **Shared Data Regions and Access Rights**

Access rights for a shared data region used to apply to all the partitions that attach to the region. You now configure specific access rights for each partition that attaches to the region. For migration information, see [Shared Data Regions and Access Rights](#), p.38.

### **User Extensions to Core OS and Partition Configuration Records**

The core OS and partition configuration records each include two fields for user extensions (**user1** and **user2**), which you were unable to change from their default value of 0. You can now configure them through the following XML attributes at configuration and build time:

- `/Module/CoreOS/CoreOSDescription/KernelConfiguration/@user1`
- `/Module/CoreOS/CoreOSDescription/KernelConfiguration/@user2`
- `/Module/Partitions/Partition/PartitionDescription/Settings/@user1`
- `/Module/Partitions/Partition/PartitionDescription/Settings/@user2`

### **XML Document Types, XML Schemas, and Namespace Identifiers**

The VxWorks 653 configuration and build system now uses a number of different XML files. Each XML file is based on an XML document type defined in one of two XML schemas. The schemas are:

- VxWorks653 Configuration Schema, which has an XML namespace identifier of:  
“`http://www.windriver.com/vxWorks653/ConfigRecord`”
- VxWorks653 Shared Library API schema, which has an XML namespace identifier of:  
“`http://www.windriver.com/vxWorks653/SharedLibraryAPI`”

## Manuals

The *Wind River Platform for Safety Critical ARINC 653 Configuration Reference*, 2.1 is now called the *VxWorks 653 Configuration and Build Reference*, 2.2.

The *VxWorks 653 Configuration and Build Reference* now includes component-reference information, such as names of CDF files (for kernel components), dependencies, exclusions, library filenames (for partition components), permitted location, category, and parameter information.

Configuration and build information that was in the *Wind River Workbench User's Guide (VxWorks 653 Version)*, 2.4, is now in the brand-new *VxWorks 653 Configuration and Build Guide*, 2.2. This new manual describes the new reference process for configuring and building a system.

## Shared Libraries and Applications

Parameters for shared libraries (including parameters for partition OSs) and applications that were previously set through **prj** commands are now set in configlettes or by command-line compiler options. For migration information on configuring shared libraries and applications, see [Shared Libraries and Applications](#), p.39. For migration information on configuring a partition OS shared library, see [Shared Libraries for Partition OSs](#), p.44.

## Shared Library Interfaces

Shared library interfaces, which were previously set by **.def** files, are now set in XML interface definition files. In addition, the XML files can now define multiple versions of interfaces for shared libraries.

## .sym Files

Previously, **.sym** files were generated automatically. Now you need to add rules to the makefiles for building **.sym** files if either of the following is true:

- The core OS includes the **INCLUDE\_NET\_SYM\_TBL** component.
- You want to use Workbench to connect to a target (the Workbench target server reads **.sym** files when it starts).

For information on generating **.sym** files, see [Generating .sym Files](#), p.37.

### Unused XML Attributes

XML schema attributes that are not used in the configuration process have been removed from the schemas.

### VerIMaX

The VerIMaX XML processing and validation tool has been enhanced. Also, there is now a command-line option for specifying an output directory. The default is the current directory. For details see the VerIMaX usage message.

7

### 7.2.2 Fixed Problems

For the latest list of problems fixed in the configuration and build facility, see the Online Support Web site.

## 7.3 Known Problems

This section lists some known problems with VxWorks 653 configuration and build tools. For the latest list, see the Online Support Web site.

### hello-ace and Schema Location

The following file in your installation:

**target/reference/helloWorld/moduleOS/hello-ace/ace/hello-ace.xml**

uses the wrong value for the **schemaLocation** attribute.

The example builds and runs correctly. However, an XML-aware editor will warn that **hello-ace.xml** is not schema-compliant. To eliminate the warning, change **Application.xsd** to **CoreOS.xsd** for the **schemaLocation** attribute.

### hello-version Build Error (Windows)

The hello-version use case creates a shared library with more than one version of its interface. Building the use case on Windows hosts causes a build error.

To make the use case build successfully, modify **Makefile.sl** in the following location in your installation:

**target/reference/helloWorld/sharedLibrary/hello-version/sl/**

In the following lines:

```
xmlgen --linkage --arch $(TOOLARCH) -j "version1" --output-stubs $@ $<  
xmlgen --linkage --arch $(TOOLARCH) -j "version2" --output-stubs $@ $<
```

remove all the double quotation marks.

## 7.4 Migration Information

Because the VxWorks 653 configuration and build process has changed so much, in addition to noting the following migration issues, Wind River recommends that you closely consult the *VxWorks 653 Configuration and Build Guide* as you migrate from VxWorks 653 2.1. Pay particular attention to the reference-process chapter, which includes a quick-start tutorial and use cases to follow as a starting point for building each part of your VxWorks 653 system.

### Build Settings

The following build-related attributes have been removed from the XML schema and are now set in makefiles as described below.

#### BuildQualifier Attribute

For the core OS, set the build spec with the following command:

```
prj projBuildSet -prj myKernelDir buildSpec
```

For shared libraries and applications, the **vpath** directive in the makefile determines which binary components to use. Set the directive as follows:

- For debug build specs, use the following:

```
vpath %.o $(WIND_BASE)/target/vThreads/lib/obj$(CPU)$(TOOL)vx
```

- For cert build specs, use the following:

```
vpath %.o $(WIND_BASE)/target/vThreads/lib/obj$(CPU)$(TOOL)cert
```

### EntryPoint Attribute

For an application, set the **USER\_APPL\_INIT** parameter in **vxMain.c** to the application's initialization routine.

---

**CAUTION:** Enclose the command-line option in double quotation marks, as in the following example:

```
CFLAGS_EXTRA = -D "USER_APPL_INIT=usrAppInit()"
```

Doing so overcomes the problem that would result in shells where parentheses have special meaning.

---

### SourceModule and SourcePath Attributes

For shared libraries and applications, add the module's filename (including its path) to the dependency list for the corresponding **.sm** file.

For example, the following line in a previous XML configuration file:

```
<Description SourcePath="objdir" SourceModule="app.o" />
```

becomes this in the makefile:

```
app.sm: vxMain.o ... objdir/app.o
```

### Generating .sym Files

If the core OS includes the **INCLUDE\_NET\_SYM\_TBL** component or you use Workbench to connect to a target, system integrators need to add rules to their makefiles to generate **.sym** files.

To generate **.sym** files for a downloadable (netboot) image, add the following lines to the makefile:

```
SYM_FILES = $(shell $(XMLGEN_FILES) --sym $(XML_FILE))  
net: $(SYM_FILES)
```

To generate **.sym** files for a RAM payload image, add the following lines to the makefile:

```
SYM_FILES = $(shell $(XMLGEN_FILES) --sym $(XML_FILE))
COREOS_SYM_FILE = $(shell $(XMLGEN_FILES) --sym --only-coreos $(XML_FILE))
ram: sms_ramPayload.sym boot.txt $(SYM_FILES)
sms_ramPayload.sym: $(COREOS_SYM_FILE)
    $(CP) $< $@
```

## Makefiles

Because of a problem with make (it does not wait for a **.bat** file in a makefile to finish running), if you wrote your own makefiles to use the VxWorks 653 2.1 **prj.bat**, **prjCreate.bat**, or **xmlgen.bat** files (or modified supplied makefiles to use these files), use the VxWorks 653 2.2 **prj**, **prjCreate**, or **xmlgen** utilities instead.

## Namespace Identifiers

Because the XML namespace identifiers have changed, VerIMaX generates a namespace error if you do not change them to the following:

- VxWorks653 Configuration Schema has an XML namespace identifier of:  
"http://www.windriver.com/vxWorks653/ConfigRecord"
- VxWorks653 Shared Library API schema has an XML namespace identifier of:  
"http://www.windriver.com/vxWorks653/SharedLibraryAPI"

## Shared Data Regions and Access Rights

Since you now configure specific access rights for each partition that attaches to a shared data region (rather than have access rights the same for all the partitions that attach to the region), you need to change your XML configuration document. The **UserAccess** attribute has been moved to each partition's **SharedDataRgn** element as shown in bold:

```
<Partitions>
  <Partition
    Name="firstPartitionThirtyCharacters"
    Id="1"
    Type="APP_PARTITION">
    <PartitionDescription>
    <Application NameRef="firstPartition"/>
    <SharedDataRegion
      NameRef="sdRgn1"
      UserAccess="READ_WRITE"/>
    ...
```

In this manner, each partition that attaches to “sdRgn1” defines its own **UserAccess** permissions.

## Shared Libraries and Applications

This section describes the relationship between now-obsolete CDF files for shared libraries and applications and their new configlettes (.c files), parameters, and binary components (.o files).

### Locating Binary Components

Binary components are in the following location in your installation:

**target/vThreads/lib/objCpuTypeToolTypeSubset**

For example in:

**target/vThreads/lib/objPPC604gnucert**

### Locating Configlettes

Configlettes are in the following location in your installation:

**target/vThreads/config/comps/src**

### Setting Parameters

You can set a parameter in either of the following ways:

- Modify the configlette in which the parameter is declared.
- Use a command-line option to the C compiler **-Dparameter=value**. For example:

```
CFLAGS_EXTRA=-DROOT_STACK_SIZE=50000
```

**CFLAGS\_EXTRA** puts the flag on the command line for all .c files. If you want to adjust the flag for a single .c file, see the procedure in [Shared Libraries for Partition OSs](#), p.44.

## Replacing Obsolete Components with Binaries, Configlettes, and Parameters

This section lists obsolete components (**INCLUDE\_\***) and the current binary components (.o files), configlettes (.c files), and parameters that correspond to them.

## **INCLUDE\_AMIO**

Replace with:

- **vThreadsAmioComponent.o**
- **usrAmio.c**

Parameters:

- **MAMIO\_DEVICE\_NAME**
- **PAMIO\_DEVICE\_NAME**
- **PAMIO\_OPTIONS**

## **INCLUDE\_AMIO\_REDIRECT**

Replace with:

- **usrAmioRedirect.c**

## **INCLUDE\_APEX**

Replace with:

- **apexComponent.o**

## **INCLUDE\_APEX\_MINIMAL**

Replace with:

- **apexMinimalComponent.o**

## **INCLUDE\_COIL**

Replace with:

- **coilComponent.o**
- **sslMain.c**

In addition, the developer of the shared library must define **INCLUDE\_COIL** as follows when compiling **sslMain.c**:

```
CFLAGS_EXTRA=-DINCLUDE_COIL
```

## **INCLUDE\_COIL\_PPS**

Replace with:

- **coilPpsComponent.o**



**INCLUDE\_COIL\_STDLIB**

Replace with:

- **coilStdlibComponent.o**

**INCLUDE\_CONFIG\_RECORD**

See [INCLUDE\\_VTHREADS](#), p.42.

**INCLUDE\_CPLUS**

Replace with:

- **vThreadsCplusComponent.o**

**INCLUDE\_CPLUS\_LIBRARY**

Replace with:

- **vThreadsCplusLibraryComponent.o**

**INCLUDE\_EXC\_FORMAT**

See [INCLUDE\\_VTHREADS](#), p.42.

**INCLUDE\_HM**

See [INCLUDE\\_VTHREADS](#), p.42.

**INCLUDE\_HM\_SHOW**

Replace with:

- **hmShow.o**

**INCLUDE\_POSIX**

Replace with:

- **vThreadsPosixComponent.o**
- **vThreadsPosixInit.c**

Parameters:

- **AIO\_TASK\_PRIORITY**
- **AIO\_TASK\_STACK\_SIZE**
- **MAX\_AIO\_SYS\_TASKS**
- **MAX\_LIO\_CALLS**
- **MQ\_HASH\_SIZE**

- NUM\_SIGNAL\_QUEUES

#### **INCLUDE\_POSIX\_SHOW**

Replace with:

- vThreadsPosixShowComponent.o

#### **INCLUDE\_SYM\_TBL\_INIT**

Replace with:

- usrSymTbl.c

Parameters:

- SYM\_TBL\_HASH\_SIZE\_LOG2

#### **INCLUDE\_TIME\_MONITOR\_SHOW**

Replace with:

- timeMonitorShowLib.o

#### **INCLUDE\_VTHREADS**

Replace the following:

- INCLUDE\_CONFIG\_RECORD
- INCLUDE\_EXC\_FORMAT
- INCLUDE\_HM
- INCLUDE\_VTHREADS

with:

- vThreadsComponent.o
- sslMain.c

Parameters:

- ENV\_VAR\_USE\_HOOKS
- INCLUDE\_CONSTANT\_RDY\_Q
- INT\_LOCK\_LEVEL
- ISR\_STACK\_SIZE
- MAX\_LOG\_MSGS
- MAX\_NUMBER\_OF\_ACCU
- NUM\_DRIVERS
- NUM\_FILES
- NUM\_STACK\_GUARD\_PAGES
- ROOT\_STACK\_SIZE

- `SELECT_SERVER_QSIZE`
- `TIME_MONITOR_ENABLE`

**INCLUDE\_VTHREADS\_DATA**

Replace with:

- `vThreadsDataInit.c`

**INCLUDE\_VTHREADS\_PPS**

Replace with:

- `vThreadsPpsLib.o`

**INCLUDE\_VTHREADS\_SHELL**

Replace with:

- `vThreadsShellComponent.o`

**INCLUDE\_VTHREADS\_SHELL\_START**

Replace with:

- `vThreadsShell.c`

Parameters:

- `SHELL_STACK_SIZE`

**INCLUDE\_VTHREADS\_SHOW**

Replace with:

- `vThreadsShowComponent.o`

**INCLUDE\_WINDVIEW**

Replace with:

- `vThreadsWindViewComponent.o`

## Shared Libraries for Partition OSs

To configure a parameter for a partition OS shared library, perform the following steps:

1. Find the configlette that uses a particular CDF parameter (for example **ROOT\_STACK\_SIZE**) by searching for the parameter in the following location in your installation:

**target/vThreads/config/comps/src**

In this example, the configlette is **vThreadsConfig.c**.

2. Use the **CFLAGS\_EXTRA** make variable to declare a command-line option to set the parameter. In this example, since **vThreadsConfig.c** is included in **sslMain.c** by a **#include** statement, you need to add the following lines:

```
CFLAGS_EXTRA = $(CFLAGS_*)  
CFLAGS_sslMain = -DROOT_STACK_SIZE=40000
```

The first line above uses make's automatic variable (**\$(\*)**) to set **CFLAGS\_EXTRA** to a target-specific make macro. The second line sets the compiler option to define the C macro only when compiling **sslMain.c**.

You can use this technique to add compiler flags to an individual source file, for example, if an application **.c** file needs one or more special parameters.

## 7.5 Documentation Errata

This section lists errata in VxWorks 653 configuration and build documentation. For the latest list of documentation errata, see the Online Support Web site.

### VxWorks 653 Configuration and Build Guide

In section 22.7.3 *Shared Library Versioning*, in the section *Shared Library Makefile*, references to **"version1"** and **"version2"** should be **version1** and **version2**. (Double quotation marks cause a build error on Windows hosts).

# 8

## *Wind River DO-178B Network Stack for VxWorks 653*

- 8.1 Introduction 45
- 8.2 Usage Caveats 46
- 8.3 Known Problems 46
- 8.4 Documentation Errata 47

### **8.1 Introduction**

The optional Wind River DO-178B Network Stack for VxWorks 653 (DO-178B Network Stack) must be ordered separately. It is a UDP/IPv4 network stack for the core OS of VxWorks 653. It is a brand-new Wind River product.

## 8.2 Usage Caveats

This section lists caveats for using the DO-178B Network Stack.

### Priority of Polled Ethernet Driver

The polled Ethernet driver that must be used with the DO-178B Network Stack should run at a priority lower than the partitions. This ensures that the driver runs only during the idle (SPARE) time in the ARINC schedule.

### SO\_USELOOPBACK Socket Option

The DO-178B Network Stack does not support or use the **SO\_USELOOPBACK** socket option, and the sockets API reference appropriately does not mention it. However, if the option is used with **setsockopt()** or **getsockopt()**, rather than return the expected -1 (failure), the routine returns 0 (success), and the option is set and read.

## 8.3 Known Problems

For the latest list of known problems, see the Online Support Web site.

## 8.4 Documentation Errata

This section lists errata for documentation associated with the DO-178B Network Stack. For the latest list, see the Online Support Web site.

### Wind River DO-178B Network Stack for VxWorks 653 Guide

In chapter 1 *Overview*, in the section *Additional Wind River Documentation*, *VxWorks 653 vThreads API Reference* should be *VxWorks 653 Partition OS API Reference*.

References to supervisor-mode and user-mode drivers should be supervisor-level and user-level drivers.

In section 3.7 *Configuring Network Interfaces*, references to `TSTK_ETHERNET_n` should be `TSTACK_ETHERNET_n`.

In section 3.12: *Specifying the Build Spec*, the two command-line examples are reversed. The section should read:

To set the build spec for debugging, add the following to the core OS project makefile (for example):

```
prj projBuildSet -prj myKernelDir PPC604gnu.debug
```

To set the build spec for a cert image, add the following:

```
prj parojBuildSet -prj myKernelDir PPC604gnu.cert
```

In appendix A *Socket Options*, the following option is missing:

#### **SO\_SNDTIMEO**

Send timeout, that is, the maximum number of seconds and milliseconds an output routine waits until it completes. If it returns before all data is sent, it returns the short count (or `EWOULDBLOCK` if no data was sent).

Allowable values:

- 0 (routine waits forever)
- positive integers (the routine waits for the specified time)

Default value: 0

### Wind River DO-178B Network Stack for VxWorks 653 Sockets API Reference

The sockets API reference describes some features that the DO-178B Network Stack does not support.

Since TCP and IPv6 are not supported, disregard references to them.

The following **SOL\_SOCKET** socket options are not supported, so disregard references to them:

- **SO\_ACCEPTCONN**
- **SO\_DEBUG**
- **SO\_DONTROUTE**
- **SO\_KEEPALIVE**
- **SO\_LINGER**

If any is used with **setsockopt( )** or **getsockopt( )**, the routine returns the expected result of -1 (failure) and sets **errno** to **ENOPROTOOPT**.

The following **IPPROTO\_IP** options are not supported, so disregard references to them:

- **IP\_DONTFRAG**
- **IP\_HDRINCL**
- **IP\_TOS**
- **IP\_TTL**



# 9

## *Wind River Workbench*

- 9.1 Introduction 49
- 9.2 Changes in This Release 50
- 9.3 Usage Caveats 53
- 9.4 Known Problems 61
- 9.5 Documentation Errata 68

### 9.1 Introduction

Wind River Workbench (Workbench) is an Eclipse-based development suite that lets you establish and manage host-target communications and develop, debug, and monitor VxWorks 653 systems running on target hardware or a simulator.

## 9.2 Changes in This Release

The following sections outline changes in Workbench 2.6.1 compared to Workbench 2.4, which was the release that accompanied the previous release (2.1) of VxWorks 653 Platform, then called Wind River Platform for Safety Critical ARINC 653.

Workbench 2.6.1 is based on Eclipse 3.2.2. For information about Eclipse and new features in release 3.2.2, see:

<http://www.eclipse.org/eclipse/>

### 9.2.1 Enhancements

#### Debugger

##### **Adding Debug Information for a Module Loaded by a Custom Loader**

You can now specify an object file for the debugger to use when object code is loaded on the target using a custom loader.

##### **Configuring Debug Settings for Custom Editors**

In addition to specifying which editor should be used for particular file types, you can edit an Eclipse extension point to allow other features that were formerly associated with only the Workbench editor (such as double-clicking in the gutter to set breakpoints, run to line, and watch) to be associated with your preferred editor.

##### **Improved Performance for Large Applications**

Single-stepping time for large applications and executable reading performance for large applications have been improved.

#### Host Shell (WindSh)

As you debug, using the WindSh host shell (or the target shell), you can set a break point on a particular data address. You can now qualify it with a value. The

breakpoint is hit only if the value at that address changes to the specified value. The new commands are:

- **bhv** (for the core OS)
- **pdbhv** (for partitions)

This functionality is command line only. It is not available in the GUI-based debugger.

You can now specify scripts to play when trace events are hit. For details, see *Wind River Workbench Host Shell User's Guide: Eventpoint Scripting*.

The host shell supports additional GDB commands that are specific to Wind River. They are listed in *Wind River Workbench Host Shell User's Guide: Using the GDB Interpreter*.

Users of SingleStep can now convert SingleStep scripts to Tcl scripts that can be run in the host shell. For details, see *Wind River Workbench Host Shell User's Guide: SingleStep Compatibility*.

9

## Launch Configuration

### Command-Line Launches

The **wrrws\_update** script has an additional option (**-r**) that lets you run a launch configuration from the command line. Execution is always in run mode.

### Launch Control

You can now create a launch that consists of a sequence of your launch configurations, each one of which is then considered a sub-launch. Pre-launch, post-launch, and error-condition commands are also supported for each sub-launch.

## Static Analysis

### Include Search Path Wizard Improved

The **Include Search Path** wizard has been improved to be more intuitive and self-explanatory. Also, you can now copy search paths to the clipboard, so that you can paste them into a makefile. All operations run in a background thread to make the wizard more responsive.

### Symbol Browser Supports Sorting

The contents of the Symbol Browser can now be sorted by clicking a table column heading. Clicking the heading a second time changes the sort direction.

## Target Manager

### Attaching the Host Shell to a Running Process or Task

Using the **Attach Host Shell** option from the Target Manager context menu, you can now attach the host shell to a running process, thread, or kernel task. The host shell is started in GDB mode.

You can also attach the host shell to a process or task from the command line using the **-attach** *threadID* command, where *threadID* is the debugger thread ID.

### OCD and GDB Serial Connections Automatically Attach the Debugger

Connecting to on-chip debugging or serial-connection types for the GNU debugger now attaches the debugger and switches to the **Device Debug** perspective automatically.

## Using Workbench in an Existing Eclipse Environment

You can now register Workbench into an existing Eclipse environment from the Workbench **Help** menu, rather than running a script from the command line. Workbench will work with Eclipse's C/C++ Development Tooling (CDT) and other tools you already have installed.

## Views

### Eclipse Console

Workbench now supports the **Eclipse Console** view with virtual I/O (VIO) features that let you monitor the standard output and error output of your applications and enter standard input.

### Kernel Objects Viewer

The **Kernel Objects Viewer** view has been enhanced to display which tasks are associated with which executable. In addition, the display is faster because the view fetches information only when you request it.

### Retriever and Search

The **Retriever** view has been integrated into the **Search** view. Choosing a search scope is now more flexible, in that you can specify projects, files, folders, or arbitrary resource selections.

### Wind River System Viewer

The Wind River System Viewer 4.9 supports VxWorks 653 2.2. Other than that, there are no functional changes specific to VxWorks 653.

### 9.2.2 Fixed Problems

For the latest list of problems fixed in Workbench, see the Online Support Web site.

## 9.3 Usage Caveats

### Host-Related Issues (Windows)

#### Defragment Hard Disks to Decrease Workbench Startup Time

If Workbench takes a minute or more to open on Windows, the cause might be a fragmented hard disk. Defragmenting your hard disk could reduce Workbench startup time by half or more.

### WindSh Protection-Domain Commands for VxWorks 653

The WindSh protection-domain commands for VxWorks 653 are available in task-mode debugging only. They are not available in system-mode debugging. Protection-domain commands are described in the list of unqualified WindSh commands in the *Wind River VxWorks 653 Shell* reference. They start with **pd**.

## Workbench Issues

### Accessing Online Support from the Workbench Welcome Page

#### On Windows

If Internet Explorer is installed on your computer, you must enable cookie support for the Internet Explorer ActiveX control before you can access the Wind River Online Support Web site from the Workbench Welcome page. If your host computer is configured not to accept cookies for other sites, follow these steps to override the configuration:

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

You can now access the Online Support login page.

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

#### On Solaris

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

### Dragging and Dropping in Workbench

When dragging information from the Editor and dropping it into the **Watch** view or other data views, press the **Ctrl** key to ensure that you are copying rather than cutting and pasting.

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

### Increasing the Limit on Open Files in the Editor

By default, Workbench limits you to having six files open in the Editor. If you open a seventh file, Workbench closes one of the existing open files.

To increase the number of files you can have open, select:

**Window > Preferences > General > Editors**

Increase the number in the **Number of opened editors before closing** field.

If you do not want Workbench to close files for you, clear the **Close editors automatically** check box.

If you want to keep open only those files 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.

### Increasing Virtual Memory

Building large applications can cause Java to run out of memory. To avoid this, increase the memory for the Java virtual machine to 512 MB or higher as follows:

#### From the Command Line

On Windows:

From a shell, type the following:

```
C:\> cd installDir\workbench-2.x\wrwb\platform\eclipse\x86-win32\bin
C:\> .\wrwb.exe -vmargs -Xmx512m
```

On Solaris:

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

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

#### From the Windows Start Menu

1. Select **Start > Programs > Wind River > Workbench 2.6**, then right-click **Wind River Workbench 2.6** and select **Properties**.
2. From the **Shortcut** tab, move the cursor to the end of the command in the **Target** text box.
3. After **wrwb.exe**, type:  
  
    -vmargs -Xmx512m
4. Click **OK**.

Each time you use the **Start** menu to start Workbench, you 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.

### Make System Support

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

### Multiple Users and Installations

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



---

**NOTE:** Each user should work in a user-specific workspace. Sharing workspaces is not possible. To specify a workspace, do one of the following:

- Use the **-data** startup option.
  - During startup, select a workspace in the **Workspace Selection** dialog box.
  - In Workbench, select **File > Switch Workspace**.
- 

The following considerations apply.

#### Single User with a Single Installation

This configuration presents no special installation or use considerations, assuming 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 in *Single User with a Single Installation*, p.56, 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. Solaris root users performing multiple installations should refer to the **umask** and **chmod** manual pages.



### Multiple Users with a Single Installation

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

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

### Single User with Multiple Installations

If a single user installs Workbench more than once, it is important that the configuration area:

- **%USERPROFILE%\workbench-2.6** (Windows)
- **\$HOME/workbench-2.6** (Solaris)

does not get corrupted.

Different versions of Workbench do 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 the team features of the standard Eclipse installation as documented in the *Eclipse Workbench User Guide: Tools, Working in the team environment with CVS* in online help.

### Renaming Projects (Windows)

Even with an open connection and the debugger attached to a build target output file of a project, you can rename the project, as long as the file size does not exceed the setting in this field:

#### Maximum host-side Symbol File Size to load fully into In-Memory Cache

(See **Window > Preferences > Target Manager > Debug Server Settings > Symbol File Handling Settings**.)

## Running Workbench Remotely

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

In the **Remote Connections** dialog box, the **Remote Workspace Location** field 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.

## Turning off Static Analysis for Large Projects

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

To turn off static analysis do the following:

### For existing projects

Select **Project > Properties > Static Analysis > Basic Configuration**.

Clear **Enable analysis**.

### For new projects

Select **Window > Preferences > Static Analysis**.

Clear **Automatically analyze projects added to the workspace**.

### By project

If you prefer not to turn off static analysis for all new projects, you can decide whether analysis should be performed for any particular project as you create it. In the **New Project** wizard, click **Next** until you reach the **Static Analysis** page, then configure if and how Workbench should analyze the code of the new project.

## 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 run the script in the proper environment for your operating system.

### On Windows

1. Open a VxWorks 653 Development Shell by selecting:  
**Start > Programs > Wind River > VxWorks 653 2.2 Development Shell**

Starting the shell sets up the appropriate environment variables.

2. Create and save a batch file with a descriptive name (such as **showhelp.bat**) and the following content. Replace *installDir* with your Workbench installation directory path and *2.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-2.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-2.x\wrwb\platform\eclipse -noexec
pause
```

3. Run the new batch file in the shell. It may take a minute or two for the browser to open.
4. When you are finished browsing, close the browser, then press CTRL+C in the shell. When asked if you want to stop the batch file, press Y for yes.

### On Solaris

1. In a terminal window, create and save a script with a descriptive name (such as **showhelp.sh**) and the following content. Replace *installDir* with your Workbench installation directory path, and *2.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-2.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-2.x/wrwb/platform/eclipse -noexec pause
```



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

2. Save the script, then make it executable:
3. Copy the script to your installation directory.
4. Change to your Workbench installation directory.
5. Run the **wrenv** command to initialize the appropriate environment variables:

```
$ ./wrenv.sh -p workbench-2.6
```

6. Run the new script in the shell. It may take a minute or two for the browser to open.

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

7. When you are finished browsing, close the browser by entering **CTRL+C** in the terminal window. When asked if you want to stop the batch file, press **Y** for yes.

### Viewing Individual Wind River Documents in a Browser

It is possible to view a single document at a time using the HTML files that are in:

*installDir*\docs\extensions\eclipse\plugins

Workbench does not need to be running, and you do not need to run a batch file.

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.

To view, for example, the *Wind River Workbench User's Guide, 2.6.1 (VxWorks 653 Version)*, enter the following into your browser, substituting your Workbench installation path for *installDir*:

```
installDir/docs/extensions/eclipse/plugins/com.windriver.ide.doc.wr_workbench_vxw
orks_653/wr_workbench_vxworks_653_users_guide_2.6.1/html/index.html
```

### Viewing Workbench Help and Other Documentation in Netscape (Solaris)

On Solaris, the default Workbench browser for documentation is Mozilla. To view documentation in Netscape, do the following:

1. In 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, type simply:

```
netscape
```

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

You can now view the documentation in Netscape, either context-sensitive help from Workbench itself (by pressing the **Help** button and selecting a link from the infopop), or by navigating to it from:

**Help > Help Contents > Wind River Documentation**

### Workspace Location

You will face significant performance problems unless you place your workspace:

- on a local file system
- outside a ClearCase dynamic view (if you are using ClearCase)

## 9.4 Known Problems

This section lists known problems with Workbench. For the latest list, see the Online Support Web site.

9

### Configuration and Build

The VxWorks 653 configuration and build system is not compatible with the Workbench configuration and build system.

Although **VxWorks 653** is listed as a project type in the Workbench **New Project** wizard, selecting it opens an error dialog box. If you continue, you will not be able to create a project. To configure and build a VxWorks 653 system you must use the command-line tools that accompany VxWorks 653.

### Debugging Issues

#### 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 size accommodates debugging about 300 tasks with short names. If you use longer task names or more than 300 tasks, the Target Manager, the shell, and the debugger fail.

To avoid this problem, set **WDB\_GOPHER\_TAPE\_NB** to a higher value (such as 20). For information on this parameter, see the *VxWorks 653 Configuration and Build Reference*.

## Data Views

Depending on the nature of a variable or expression, changing its value in data views (such as the **Watch** view) may return an error.

For example, if the expression refers to a write-only register, a write succeeds, but a read always fails.

## Debugger Views

### Memory View

**Memory** view features not implemented in Workbench include:

- back
- forward
- compare
- scroll (with mouse)

### Run Control

The step return command is not fully working for recursive routines. For example, consider the following simple recursive routine:

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

If you call **factorial()** with an argument of 6, then perform some **Step** operations until **factorial()** is called with an argument of 5, the call stack contains two routine 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 routine, which is **factorial(6)**.

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

## Default Eclipse Debugger Views

Many of the default Eclipse views were designed to be used synchronously and were not designed to be used with a real-time target. Therefore, the Workbench debugger supports only the **Debug** and **Breakpoint** Eclipse debugger views.

All other views in the **Debug** perspective (or accessed from the **Window > Show View > Other > Debug** list) are not supported and are not updated. Instead, use the corresponding views in the **Device Debug** perspective (or accessed from the **Window > Show View > Other > Device Debug** list).

## Symbol Browser Debug Symbols

Displaying debug symbols in the Symbol Browser can slow the loading of symbols when a symbol-reload operation or reset and download operation is performed. To accelerate the operations, clear **Show Debugger Symbols**.

## Target Server and ICE Connection Conflicts

Although the Target Manager lets you connect the ICE and target server to a single board and use both connection types simultaneously, you must set the target-server timeouts to longer default values to stop the target server from automatically disconnecting when the OCD connection stops the target.

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

## Editor Issues

### Printing in Workbench (Windows)

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

### Source Code Editor Preferences

Workbench comes with its own source code editor that provides productivity-enhancing features for device-software developers. Preferences settings for this editor are located in:

**Window > Preferences > General > Editors > Wind River Workbench Editor**

All settings on the **Editors**, **Annotations**, and **Quick Diff** screens also apply to the Workbench Editor.

However, settings on the **Text Editor** screen do not apply. The text editor is used only for files that cannot be opened in the Workbench Editor.

## Properties View Issues

### Name Information Not Displayed for Parameter Values

Eclipse does not display **Name** information for parameters and parameter values that default to values defined by macros.

### Quotes Not Preserved in Parameter Values

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

For example, to include "ppp", type:

```
" \"ppp\" "
```

## Target Manager Issues

### Cannot Connect Registry or Debug Server

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 as follows:

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

### 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 Target Manager display. Do not use this feature when there are more than 100 contexts on the target.



### Workbench Requires a Host Name to Be Set

Workbench cannot connect to the target registry or debug when the host name of the local computer is set to **localhost**.

If you cannot change the local host name, you can work around the problem by setting the **-host** option in this file in your installation:

**workbench-2.x/foundation/version/resource/wtxregd/wtxregd.conf**

## Workbench Issues

### 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.

### Long Delay When Displaying Source Code for Dwarf1 Files in Assembly Mode

If you view Dwarf1 files in source mode and then switch to assembly mode, the assembly code is displayed immediately, but Workbench may take almost a minute to display the corresponding source code.

### Non-English Character Restrictions

If you use spaces (blank characters) or non-English-language characters in project names, errors result. In Windows, non-English-language characters are not supported for user names.

### Online Update

You cannot use this Eclipse feature to update the Eclipse platform or Workbench:

**Help > Software Updates > Find and Install**

However, you can use the feature to update any third-party plug-in that you may have added to your installation.

To update Workbench, download and install patches from the Wind River Online Support site.

### Problem Interpreting Unquoted Numbers as Arguments

When launch configurations are created, underlying debugger components interpret an unquoted number in an **Arguments** field as an address value. The

corresponding argument value is 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").

### 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-2.6** directory in your home directory. The directory is recreated when Workbench restarts.

### Workbench Font Size Problem with Exceed 8

Running Workbench with Exceed 8 causes font size problems. Workbench runs correctly with Exceed 7.1 and Exceed 9.0.

### Workbench Reports fork Problems (Solaris)

On Solaris, 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 runtime has the limitation that external processes are started using **fork**, which requires that enough free memory (including swap space) be 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 is allocated. If the maximum heap size is set too high, the child process cannot be started and Workbench operations fails.

Make sure that:

- Your computer is configured with enough swap space.
- Workbench is started with a reasonable heap size, but not unnecessarily high. Choose a heap size that is always lower than half the freely available memory (physical and swap). Every started process reduces the available memory.

### Workbench sh.exe Hangs During Automated Build (Windows)

If a make process hangs on Windows, check all involved makefiles for any complex pipe ("|" operator) usage. Due to a general problem with the way Windows handles pipes, 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.

### Workbench Startup Reports “An Error Has Occurred”

If Workbench starts with the message **An Error Has Occurred**, start it with the **-clean** option. On Windows, modify the shortcut that starts Workbench to **wrb.exe -clean**. On Solaris, start Workbench with:

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

If that does not help, remove the Workbench configuration directory:

- **\$HOME/.workbench-2.x** (Solaris)
- **%USERPROFILE%\workbench-2.x** (Windows)

If neither method helps, use a new workspace.

### Workspace-in-Use Errors

Several issues can cause Workbench to display this 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.
- Workbench may not have unlocked the workspace during an abnormal exit. To unlock it, kill the remaining Workbench and Java processes.
- 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 solutions, use another JVM argument to start Workbench:

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



---

**CAUTION:** The above command lets multiple users open the same workspace, which might result in unrecoverable lost data.

---

### 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, navigate to your Workbench install directory, then delete the *workspaceDir*/**.metadata/.lock** file.

## 9.5 Documentation Errata

For the latest list of documentation errata, see the Online Support Web site.

# 10

## *Wind River Workbench for On-Chip Debugging*

(OCD)

<b>10.1 Introduction</b>	<b>69</b>
<b>10.2 Unsupported Features</b>	<b>70</b>
<b>10.3 Usage Caveats</b>	<b>71</b>

### **10.1 Introduction**

The optional Wind River Workbench for On-Chip Debugging (OCD) must be ordered separately. It is a new addition for VxWorks 653 Platform.

For release-notes information on OCD in general, see the *Wind River Workbench On-Chip Debugging Release Notes*, 2.6.1. Chapter 4 of that document is specific to VxWorks 653 Platform. However, additional information related to OCD and VxWorks 653 Platform has become available since that document was written.

This chapter is a modification of chapter 4. The additional information includes clarifications, additional detail, and the use of the current product name of VxWorks 653 Platform (rather than Wind River Platform for Aerospace & Defense).

## 10.2 Unsupported Features

VxWorks 653 Platform does not support the following OCD features.

### **BSPs**

VxWorks 653 Platform provides OCD support for only the following:

- PPC 750GX
- PPC 7457

### **diab Compiler**

VxWorks 653 Platform does not support the diab compiler, which OCD does support.

### **Level of Support**

VxWorks 653 Platform provides only kernel-level and single-partition OCD support.

### **Standalone Projects**

VxWorks 653 Platform does not support standalone projects, which OCD does support.

### **Tutorials**

Tutorials in the OCD documentation for creating projects do not work with VxWorks 653 Platform.

## 10.3 Usage Caveats

This section describes information that VxWorks 653 Platform users need to be aware of when reading the OCD documentation.

### Boot ROM Filenames

VxWorks 653 boot ROM filenames are:

- **bootApp\_romCompressed.bin** (not **bootrom.bin**)
- **bootApp\_romCompressed.hex** (not **bootrom.hex**)

### BSP-Specific Information

VxWorks 653 BSPs record board-specific information (such as switch and jumper settings) in files named **target.nr** (not **target.ref**). The files are located here in your installation:

**vxworks653-2.2/target/config/yourTargetBoard/target.nr**

### Examples

Most of the examples are specific to VxWorks. Some steps are different for VxWorks 653. For example, if you are told to select **Wind River VxWorks 6.x Target Server Connection** in the **New Connection** wizard, select the appropriate VxWorks 653 target server connection.

### Pathnames

Pathnames are specific to VxWorks. Many are different for VxWorks 653. For example, VxWorks 653 boot ROMs are located here in your installation:

**vxworks653-2.2/target/proj/yourTarget\_bootApp/buildSpec\_romCompressed**





## 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)	<a href="mailto:support@windriver.com">support@windriver.com</a>
Europe, Africa, Middle East	+(00) 800-4977-4977 (toll-free)	<a href="mailto:support-EC@windriver.com">support-EC@windriver.com</a>
Japan	81-3-5778-6001 (direct)	<a href="mailto:support-jp@windriver.com">support-jp@windriver.com</a>

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.