

**VxWorks<sup>®</sup>**  
*Release Notes*

*5.2 Beta*

---



**VxWorks<sup>®</sup>**  
*Release Notes*

*5.2 Beta*

---

---

Copyright © 1984 – 1994 Wind River Systems, Inc.

ALL RIGHTS RESERVED. No part of this publication may be copied in any form, by photocopy, microfilm, retrieval system, or by any other means now known or hereafter invented without the prior written permission of Wind River Systems, Inc.

VxWorks and Wind River Systems are registered trademarks of Wind River Systems, Inc. VxGDB, VxSim, VxMon, VxMP, VxVMI, MicroWorks,  $\mu$ Works, WindC++, WindC++ Gateway, *wind*, windX, WindNet, WindPower, WindView, and the Wind River Systems logo are trademarks of Wind River Systems, Inc.

All other trademarks used in this document are the property of their respective owners.

**Corporate Headquarters**

Wind River Systems, Inc.  
1010 Atlantic Avenue  
Alameda, CA 94501-1153  
USA

toll free (US): 800/545-WIND  
telephone: 510/748-4100  
facsimile: 510/814-2010

**Europe**

Wind River Systems S.A.R.L.  
27, Avenue de la Baltique  
Bâtiment B4, LP 739  
Z.A. de Courtaboeuf  
91962 Les Ulis Cédex  
FRANCE

telephone: 33-1-69-07-78-78  
facsimile: 33-1-69-07-08-26

**Japan**

Wind River Systems Japan  
Pola Ebisu Bldg. 11F  
3-9-19 Higashi  
Shibuya-ku  
Tokyo 150  
JAPAN

telephone: 81-3-5467-5900  
facsimile: 81-3-5467-5877

**TECHNICAL SUPPORT**

**North America**

telephone: 800/872-4977  
e-mail: support@wrs.com  
Please give your *Support ID Number* (SID)  
when contacting Technical Support.

**Europe and Japan**

Please contact your distributor to  
determine how to reach your technical  
support organization.

## Contents

<b>1. Introduction</b> .....	1
<b>2. Documentation Profile</b> .....	2
<b>3. Supported Platforms</b> .....	3
<b>4. Installing VxWorks</b> .....	4
<b>5. New Features</b> .....	5
5.1 POSIX Interfaces .....	5
5.2 Asynchronous I/O .....	6
5.3 NFS Server .....	6
5.4 New Socket Interface: Zbufs .....	7
5.5 Compressed SLIP (CSLIP) .....	7
5.6 Ping .....	7
5.7 MIB-2 Interface Libraries .....	7
5.8 New Optional Product: WindNet SNMP .....	7
<b>6. Porting Applications from VxWorks 5.1.1</b> .....	8
6.1 BSPs .....	8
6.1.1 New BSP: mv177 .....	9
6.1.2 New BSP: mv162lc .....	9
6.1.3 New BSP: ss5 .....	9
6.1.4 BSP for FORCE CPU-2CE .....	10
6.1.5 BSPs for MIPS R3000 .....	10
6.1.6 BSPs for i386/i486 .....	10

6.2	Boot ROMs .....	11
6.3	Recompiling (for MIPS R3000, or for POSIX Timers) .....	11
6.4	POSIX Clocks and Timers .....	12
<b>7.</b>	<b>Porting Applications from Older VxWorks Releases .....</b>	<b>12</b>
<b>8.</b>	<b>Problems Fixed in VxWorks 5.2 .....</b>	<b>13</b>
<b>9.</b>	<b>Known Problems in VxWorks 5.2 .....</b>	<b>56</b>
<b>Appendix A:</b>	<b>New Routines .....</b>	<b>77</b>



*Release Notes*

# VxWorks

*Release 5.2 Beta*

---

## 1. Introduction

Release 5.2 of VxWorks offers new features, support for a new target architecture, and corrections of software and documentation errors in previous releases. For the most part, applications developed under VxWorks 5.1.1 are fully compatible with VxWorks 5.2.

- **New Features**

VxWorks 5.2 brings the following new features into the standard VxWorks release:

- POSIX real-time extensions
- NFS Server capability
- CSLIP
- the *zbuf socket interface* for faster sockets
- a ***ping()*** utility

Moreover, the standard VxWorks release includes new libraries for network control (the *MIB-2 interface*) to integrate a new optional product: WindNet SNMP.

These enhancements are described in §5. *New Features*.

- **New Target**

VxWorks 5.2 supports the MC68060 as a target architecture, in addition to the broad range of targets already supported by VxWorks.

The supported combinations of target architecture, host environment, and cross-development tools are described in §3. *Supported Platforms*.

- **Bug Fixes**

VxWorks 5.2 includes corrections of problems discovered since Release 5.1.1. These corrections are described in §8. *Problems Fixed in VxWorks 5.2.*

- **Compatibility**

For the most part, applications developed under VxWorks 5.1.1 are fully compatible with VxWorks 5.2; special considerations are discussed in §6. *Porting Applications from VxWorks 5.1.1.* For information on upgrading applications from earlier releases, see §7. *Porting Applications from Older VxWorks Releases.*

Subroutines introduced or modified in Release 5.2 are listed in *Appendix A. New Routines.*

- **Installation**

VxWorks 5.2 introduces a few changes to the installation procedures. Installation is now described in a separate, comprehensive *Wind River Products Installation Guide.* These changes are summarized in §4. *Installing VxWorks.*

---

## 2. Documentation Profile

The VxWorks manuals for VxWorks 5.2 have been revised and slightly restructured.

The documentation set for VxWorks 5.2 includes the following components:

- *VxWorks Release Notes, 5.2* (this document)
- *Wind River Products Installation Guide (UNIX Version)*
- *VxWorks Programmer's Guide, 5.2*
- *VxWorks Reference Manual, 5.2* (not printed for Beta release)
- UNIX-compatible online reference (man pages)
- BSP supplements (for most BSPs, available as online man pages only)

The following Wind River Systems documentation describes auxiliary products:

- *GNU ToolKit User's Guide*
- *VxGDB User's Guide*

A new manual, the *Wind River Products Installation Guide*, consolidates information about installing VxWorks, optional products, and Wind River Systems development tools.

The *VxWorks Reference Manual* now includes entries for device drivers and associated subroutines, which were formerly supplied separately as the *VxWorks Driver Supplement*.

### 3. Supported Platforms

Table 1 shows which development hosts support which target architectures, and the toolkit that supports each combination.

Table 1. Supported Platforms with Tools Available

Target Architecture	Host					
	Sun-4 (Solaris 2)	Sun-4 (SunOS 4)	HP 9000/400 (HPUX 8.05)	HP 9000/700 (HPUX 9.0)	SGI Iris (IRIX 5.2)	IBM RS6000 (AIX 3.2)
Motorola MC680x0	GNU 2.2.3.1	GNU 2.2.3.1	GNU 2.2.3.1	GNU 2.2.3.1	GNU 2.2.3.1	GNU 2.2.3.1
Motorola CPU32	GNU 2.2.3.1	GNU 2.2.3.1	GNU 2.2.3.1	GNU 2.2.3.1		
SPARC[lite]	GNU 2.2.3.1	GNU 2.2.3.1				
Intel i960	GNU/960 (GNU 2.4)	GNU/960 (GNU 2.4)	GNU/960 (GNU 2.4)	GNU/960 (GNU 2.4)		GNU/960 (GNU 2.4)
Intel i386/i486	GNU 2.2.3.1	GNU 2.2.3.1				
AMD Am29K	GNU 2.2.3.2	GNU 2.2.3.2				
MIPS R3000/4000		SDE-MIPS (GNU 2.5.7)				

Distribution tapes for Sun-4 systems include binaries for both supported host operating systems (SunOS 4.1.x and Solaris 2.x) on a single tape. However, note that MIPS targets are not yet supported from hosts running Solaris 2.

---

**NOTE:** Certain host systems or host OS levels (listed below), are no longer supported in VxWorks 5.2. The last VxWorks release supported for these systems is VxWorks 5.1.1.

---

The following host systems are not supported at all for VxWorks 5.2:

- Sun-3 - Not supported for any SunOS level.
- DECstation - Not supported for any Ultrix level.

The following host systems are not supported with older operating system levels:

- Sun-4 - SunOS releases older than 4.1 are no longer supported; you must upgrade to SunOS 4.1.x, or to Solaris 2.x.
- HP 9000/300,400 - HP/UX 7.03 is no longer supported; you must upgrade to HP/UX 8.05.
- HP 9000/700 - HP/UX 8.05 is no longer supported; you must upgrade to HP/UX 9.0.
- SGI Iris - UNIX SVr4.0.5 is no longer supported; you must upgrade to Irix 5.2.

---

## 4. Installing VxWorks

The installation procedures for VxWorks 5.2 are somewhat different from earlier releases. See the new *Wind River Products Installation Guide* for the complete software installation instructions.

The new procedure permits including updates to software-only drivers on the VxWorks object tape rather than with each separate Board Support Package (BSP).

---

**NOTE:** The installation instructions do not include information on *configuring* VxWorks and its supporting products, because configuration (unlike installation) is important to all VxWorks developers—not just those performing the installation. Configuration instructions are in *§2. Getting Started* in the *VxWorks Programmer's Guide, 5.2*.

---

Information about setting up hardware, host network configuration, alternative configurations, and troubleshooting is also available in *§2. Getting Started* in the *VxWorks Programmer's Guide, 5.2*. If you are about to use VxWorks for the first time, be sure to consult this chapter.

---

**Warning (i386/i486):** For the Beta release only, you must set the environment variable `LD_LIBRARY_PATH` in order to use VxGDB. Set it (assuming installation in `/usr/vw`) to:

```
/usr/vw/bin/host:$LD_LIBRARY_PATH
```

---

## 5. New Features

VxWorks 5.2 introduces Asynchronous I/O (AIO), NFS Server capability, faster socket calls through the new *zbuf* socket interface, CSLIP, and a **ping** utility. The new AIO facilities are POSIX-compliant. This release also includes alternate POSIX-compliant interfaces for other OS services.

### 5.1 POSIX Interfaces

VxWorks 5.2 provides most of the interfaces specified in the POSIX standard for Real-Time Extensions (1003.1b, formerly called 1003.4). By using the POSIX interfaces carefully, you can make your applications more portable among operating systems that conform to POSIX 1003.1b.

The new or modified libraries shown in *Table 2* provide POSIX-compliant subroutines.

Table 2. **POSIX-Compliant Subroutines**

Feature	Library	Status
asynchronous I/O	<b>aioPxLib</b>	new
semaphores	<b>semPxLib</b>	new
message queues	<b>mqPxLib</b>	new
memory management (page locking)	<b>mmanPxLib</b>	new
queued signals	<b>sigLib</b>	modified
scheduling	<b>schedPxLib</b>	new
clocks and timers	<b>clockLib, timerLib</b>	modified

Release 5.2 brings VxWorks closer to conformance with the POSIX standard in the following ways:

- The new facilities for AIO comply with POSIX. See §5.2 *Asynchronous I/O* for a discussion of these new capabilities.
- VxWorks now includes alternate POSIX-compliant interface routines for facilities that were already available in previous releases: counting semaphores, message queues, signals, page locking, and scheduling control. See §3. *Basic OS* in the *VxWorks Programmer's Guide, 5.2*, for descriptions of both VxWorks-specific and POSIX interfaces to these features.
- This release provides updates to the POSIX clock and timer routines, and a file-truncation routine.

VxWorks 5.2 complies extensively with the POSIX 1003.1b standard, but the following POSIX routines are not yet included:

<code>aio_fsync()</code>	<code>mmap()</code>	<code>shm_open()</code>
<code>fchmod()</code>	<code>munmap()</code>	<code>shm_unlink()</code>
<code>fsync()</code>	<code>mprotect()</code>	
<code>fdatasync()</code>	<code>msync()</code>	

## 5.2 Asynchronous I/O

Asynchronous Input/Output (AIO) is the ability to perform input and output operations concurrently with a task's other activities. AIO enables you to decouple I/O operations from a task's internal processing whenever the two are logically independent. The VxWorks AIO implementation complies with the POSIX 1003.1b standard.

For a full description, see §4.6 *Asynchronous Input/Output* in the *VxWorks Programmer's Guide, 5.2*.

## 5.3 NFS Server

VxWorks networking facilities in previous releases included only *clients* for NFS (the Network File System), allowing you to mount remote file systems on your VxWorks targets. This release also provides an NFS *server* for VxWorks, allowing you to export dosFs file systems to other machines from VxWorks.

For details about NFS servers in VxWorks, see §6.3.3.3 *Allowing Remote Access to VxWorks Files through NFS* in the *VxWorks Programmer's Guide, 5.2*.

## 5.4 New Socket Interface: Zbufs

The *zbuf* socket interface increases performance by allowing applications to read from and write to UNIX BSD 4.3 sockets without having to copy between user buffers and network buffers. The TCP subset of this new interface is sometimes called “zero-copy TCP.” To take advantage of the faster socket calls, your applications must use a special interface for buffer manipulation and socket layer operations. See §6.2.6 *The Zbuf Socket Interface* in the *VxWorks Programmer’s Guide, 5.2*.

## 5.5 Compressed SLIP (CSLIP)

VxWorks 5.2 provides two versions of the Serial Line Interface Protocol, which permits IP networking over serial-line connections. The original SLIP, as provided in earlier VxWorks releases, is still available; but you can now choose instead to use compressed SLIP (CSLIP), a variation that writes packet headers more efficiently. See §6.7 *Serial Line Internet Protocol (SLIP and CSLIP)* in the *VxWorks Programmer’s Guide, 5.2*.

## 5.6 Ping

The UNIX **ping** command is useful to test network connections. VxWorks now includes a **ping()** call as well. See §6.3.8 *Testing Network Connections* in the *VxWorks Programmer’s Guide, 5.2*.

## 5.7 MIB-2 Interface Libraries

VxWorks 5.2 includes a library of new network information interfaces to support the WindNet SNMP optional product. Although these interfaces were designed for SNMP, they may also be useful for inspecting or changing network parameters directly. For details, see the reference documentation for **m2Lib** and the associated libraries **m2cmpLib**, **m2IfLib**, **m2IpLib**, **m2SysLib**, **M2TcpLib**, and **m2UdpLib**.

## 5.8 New Optional Product: WindNet SNMP

WindNet SNMP is an optional product that provides VxWorks with SNMP (Simple Network Management Protocol) capabilities. SNMP is a network management protocol that enables network devices, called *agents*, to be monitored, controlled, and configured remotely from a network management station.

Using this optional product, a VxWorks target can become an SNMP agent, allowing the target to be managed and configured remotely via SNMP.

WindNet SNMP is extensible: in addition to the base functionality, your applications can make extensions to the Management Information Base (MIB) of the SNMP agent. By making extensions to the MIB, you can include information that is specific to each application or to its environment.

WindNet SNMP is documented in a supplementary manual: *WindNet SNMP VxWorks Optional Component Supplement*.

Contact your Wind River sales representative for availability information, or telephone the number listed on the back cover for your national Wind River Systems organization.

---

## 6. Porting Applications from VxWorks 5.1.1

There are very few special considerations for porting existing applications from VxWorks 5.1.1.

If you are working with the i386/i486 architectures, use the revised BSPs for release 5.2. See §6.1.6 *BSPs for i386/i486*.

If you are porting MIPS R3000 applications from 5.1.1 to 5.2, note that the supported toolkit is now the GNU-based SDE-MIPS toolkit. Due to this change, the object module format for MIPS R3000 (and R4000) is ELF rather than ECOFF; you may need to recompile, depending on what BSP you were using with VxWorks 5.1.1. See §6.1.5 *BSPs for MIPS R3000*.

If your applications call *timer\_create()*, see §6.4 *POSIX Clocks and Timers*.

For information about porting directly from VxWorks 5.1 or from older releases, see §7. *Porting Applications from Older VxWorks Releases*.

### 6.1 BSPs

VxWorks 5.2 does not include changes to most Board Support Packages (BSPs). Wind River Systems enhances BSPs on an independent schedule, as needed. Except for the cases described below, BSPs that supported VxWorks 5.1.1 continue to work with VxWorks 5.2.

However, the installation procedure for all BSPs is slightly different. BSPs are now installed with the VxWorks **installOption** script, rather than with the UNIX **tar** utility. After you install VxWorks 5.2, you must re-install any BSPs you were using with VxWorks 5.1.1. See the *Wind River Products Installation Guide* for detailed installation instructions.

For detailed information about any BSP and the associated target system, see the target-specific manual entry (for example, execute **vxman mv177**, using the **vxman** alias suggested in §2.2.3 *VxWorks man Pages* of the *VxWorks Programmer's Guide, 5.2*.)

#### 6.1.1 New BSP: mv177

The mv177 BSP supports the Motorola MVME177 target board, based on the MC68060 CPU. For information about VxWorks on the MC68060, see *Appendix C. Motorola MC680x0* of the *VxWorks Programmer's Guide, 5.2*.

---

**NOTE:** The overall configuration file **vw/config/all/configAll.h** should enable hardware floating point by default for the MC68060, but this automatic configuration is missing in VxWorks 5.2-Beta. As a temporary workaround, please add the following definitions in **vw/config/mv177/config.h**:

```
#define INCLUDE_MC68881
#define INCLUDE_HW_FP
```

---

#### 6.1.2 New BSP: mv162lc

The mv162lc BSP supports the Motorola MVME162 target board with a Motorola LC68040 CPU. The LC68040 has no floating-point unit (FPU).

#### 6.1.3 New BSP: ss5

The ss5 BSP serves as a BSP for the following systems:

- FORCE CPU-3CE
- FORCE CPU-5CE
- SPARCstation LX
- SPARCstation 5

The same boot ROM works on all four targets.

---

**NOTE:** The ss5 BSP replaces the force3e and sunec BSPs, which are now obsolete.

---

#### 6.1.4 BSP for FORCE CPU-2CE

The sun2ce BSP, for FORCE CPU-2CE systems, has been revised to support VxWorks boot ROMs.

#### 6.1.5 BSPs for MIPS R3000

Two versions of the MIPS R3000 BSPs were available with VxWorks 5.1.1. The earlier version used ECOFF object module format, and required either the **sunmips** or the **sgimips** toolchain (depending on the development host). This version displays the version identifier **5.1.1-R3000** when booting; it is *not compatible* with VxWorks 5.2.

The current version of BSPs for MIPS-based targets supports both the R3000 and the R4000, and uses ELF object module format. These BSPs require the **sde-mips** toolchain. The current R3000/R4000 BSPs display the version identifier **5.1.1-R4000** when booting. These BSPs support ELF object module format, which is now standard for VxWorks on MIPS architectures.

For more information about VxWorks on MIPS architectures, see *Appendix E. MIPS R3000, R4000* of the *VxWorks Programmer's Guide, 5.2*.

#### 6.1.6 BSPs for i386/i486

The BSPs for i386 and i486 architectures were revised for VxWorks 5.2. For more information about VxWorks on i386/i486 architectures, see *Appendix H. Intel i386/i486* in the *VxWorks Programmer's Guide, 5.2*. The revised i386/i486 BSPs provide the following new features:

- A new Ethernet driver for the Intel EtherExpress32. Specify **ei** as the boot device to boot over Ethernet using this driver.
- A new printer driver for the LPT port. See the reference manual entry for **lptDrv**.
- New functions in the disk drivers for “raw” I/O. Use these functions for maximum throughput, bypassing the file system. See the reference manual entries for **fdRawio()** (for raw I/O to floppy disks, with the **nec765Fd** driver) and **ideRawio()** (for raw I/O to IDE hard drives, with **ideDrv**).
- The boot diskette now includes the standalone configuration of VxWorks (**vxWorks.st**). The default boot parameters now boot this image, so that you can begin using VxWorks immediately, even without an Ethernet card.

The default boot device specification is `fd=0,0`, and the default boot file name is `/fd0/vxWorks.st`.

The new i386/i486 BSPs also include the following changes:

- The first and the second arguments of three routines, `fdDevCreate()`, `vxsys()`, and `usrFdConfig()`, were reversed. For all three, the first argument is now the drive number, and the second argument is the diskette type.
- The order of the two parameters for the `fd` boot device in the boot prompt was also reversed to follow suit. To use this boot device, specify `fd=drv, typ` (where `drv` is the drive number and `typ` is the diskette type).

The following limitations apply to the new i386/i486 BSPs:

- The Double Space feature in DOS6.0 or greater is not compatible with either the floppy device driver or the IDE hard disk driver.
- The `eex` and `ei` Ethernet drivers do not support automatic detection of the connector configuration in EEPROM. (The `eex` driver supports the Intel EtherExpress ISA Ethernet driver, and the `ei` driver supports the Intel EtherExpress32 EISA Ethernet driver).

## 6.2 Boot ROMs

VxWorks 5.2 is compatible with all VxWorks 5.1 boot ROMs, except where new BSPs are required (see §6.1 BSPs above).

## 6.3 Recompiling (for MIPS R3000, or for POSIX Timers)

You should not have to recompile 5.1.1 applications, with the following exceptions:

- If your code uses the POSIX routine `timer_create()`, you must edit the subroutine call, then recompile. See §6.4 POSIX Clocks and Timers.
- If your target is a MIPS R3000, and if you used the `sunmips` or `sgimips` toolsets with release 5.1.1, you must recompile. The currently supported toolset is the SDE-MIPS toolset, which produces ELF format object modules. ELF is now the standard VxWorks object format for both the R3000 and the R4000.

## 6.4 POSIX Clocks and Timers

VxWorks 5.1 and 5.1.1 included an earlier version of the POSIX standard clock and timer routines. Because that standard changed in the final version of POSIX 1003.1b, some of the routines controlling clocks and timers were modified to remain compliant with the changed standard. Compliance with the new standard required the following changes:

- Several routines previously took a **clock\_t** as the first argument. The name of this type has changed; it is now called **clockid\_t** (defined in **time.h**). Because these routines always take the constant **CLOCK\_REALTIME** as the first argument, this change should not affect application code. The routines affected are **clock\_getres()**, **clock\_setres()**, **clock\_gettime()**, and **clock\_settime()**.
- The include file for clocks and timers is now **time.h**; it used to be **timer.h**. This component release includes a new **timer.h** that includes **time.h** for backwards compatibility.
- The arguments and return value of **timer\_create()** have changed. The routine is now declared as follows:

```
int timer_create
(
    clockid_t      clock_id, /* clock ID (CLOCK_REALTIME) */
    struct sigevent * evp,    /* user event handler      */
    timer_t *      pTimer    /* ptr to return value     */
)
```

The **timer\_create()** routine previously returned a pointer to the new timer. It now returns **OK** (0) on success, **ERROR** (-1) on failure. A pointer to the new timer is now stored in **pTimer**.

See §3.5 *POSIX Clocks and Timers* in the *VxWorks Programmer's Guide, 5.2* for a full description of these routines.

---

## 7. Porting Applications from Older VxWorks Releases

No changes other than those described in §6. *Porting Applications from VxWorks 5.1.1* are required to move applications from VxWorks 5.1. However, there were some improvements in VxWorks 5.1.1 that avoid certain problems present in release 5.1; if your code included workarounds for these problems, you may be able to simplify some parts of your applications. These improvements are listed in *VxWorks Release Notes, 5.1.1*.

Changes *are* required when moving your VxWorks application from release 5.0.x to newer releases; consult *VxWorks 5.1 Release Notes* for details, if you have not yet made this transition.

In general, if you skip one or more VxWorks releases when you install an upgrade, we strongly recommend that you consult the release notes for the intervening releases.

---

## 8. Problems Fixed in VxWorks 5.2

The following problems, reported in earlier VxWorks releases, have been fixed in VxWorks 5.2 (or in those BSPs released with it):

---

*SPR:* **1147**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* Programmer's Guide claims that one can exclude the network facilities by undefining the options `INCLUDE_NETWORK`, `INCLUDE_NET_INIT`, and `INCLUDE_NET_SYM_TBL`. This is not the case. The code still resides in the image. What options really "exclude" the network facilities.

*STATUS:* Fixed.

---

*SPR:* **1228**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* Documentation of `sockLib` is incorrect.

*STATUS:* Fixed.

---

*SPR:* **1493**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* Customer like to see support for **ping()** call in VxWorks. It will be very helpful for debugging network.

*STATUS:* Fixed.

---

*SPR:* **1552**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* When the **SO\_DEBUG** option is set the default message printed out is:

tcp tracing enabled : use INCLUDE\_TCP\_DEBUG

**INCLUDE\_TCP\_DEBUG** is never used/defined in any code although there is a **tcp\_debug.c** module. It appears that **INCLUDE\_TCP\_DEBUG** was meant to call/pull in **tcpTraceInit()** from **tcp\_debug.c**. This needs to be researched and added to **usrConfig** if necessary or the default trace routine **tcpTraceStub()** needs to be modified to remove the reference to **INCLUDE\_TCP\_DEBUG**.

*STATUS:* Fixed.

---

*SPR:* **1941**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* Enhancement request to make **rloginSocket** a globally-defined symbol. It's often very useful value, but only currently available if the local kernel symbols are loaded.

*STATUS:* Fixed.

---

*SPR:* **2019**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* If a driver does not provided an *xxRead()*, calls to *read()* return zero. They should instead return **ERROR** (-1) and set an appropriate **errno**.

*STATUS:* Fixed.

---

*SPR:* **2020**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* If a driver does not supply a *xxWrite()* routine, calls to *write()* return *nBytes* as if the write succeeded.

*STATUS:* Fixed.

---

*SPR:* **2050**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* If a new rlogin session to VxWorks is started while a previous rlogin session is in the process of being terminated, data corruption can occur.

*STATUS:* Fixed.

---

*SPR:* **2222**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* **fiolib.h** still has a function prototype for **itob()** though the routine doesn't exist anymore.

*STATUS:* Fixed.

---

*SPR:* **2321**

*ARCH:* SPARC

*BOARD:* sun2ce

*HOST:* All

*DESCRIPTION:* The board diagram for the Force CPU-2CE is missing from the file **target.nr**.

*STATUS:* Fixed.

---

*SPR:* **2323**

*ARCH:* SPARC

*BOARD:* sun2ce

*HOST:* All

*DESCRIPTION:* On the Force CPU-2ce, a board with 16Mbytes of memory cannot boot when the instruction and/or data cache are on.

*STATUS:* Fixed.

---

*SPR:* **2263**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* The dosFs file system erroneously prohibits creation of a directory whose name uses the filename extension (e.g. **SOMETHIN.DIR**). Regular MS-DOS systems allow this.

*STATUS:* Fixed.

---

*SPR:* **2285**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* The man page for **memPartLib** says the include file is **memLib.h**; however, this does not contain the function prototype for **malloc()**, which is actually in **stdlib.h**.

*STATUS:* Fixed.

---

*SPR:* **2334**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* The procedure to guide the user through the modification of the system error table (**statSymTbl**) contains a flaw. Invoking **vw/src/usr/Makefile** is supposed to create the file **vw/src/usr/statTbl.c**, however, if the file **statTbl.c** already exists (e.g., from a previous **make**), then it does not get updated.

*STATUS:* Fixed.

---

SPR: **2351**

ARCH: All

BOARD: All

HOST: All

DESCRIPTION: **bootpLib** man page includes the following:

SEE ALSO **bootpdLib**, **bootLib**, **RFC 951**, **RFC 1048**,

However, **bootpdLib** doesn't exist.

STATUS: Fixed.

---

SPR: **2353**

ARCH: All

BOARD: All

HOST: All

DESCRIPTION: The function **slipInit()** is being called with the wrong peer address. If the slip board is booting from a host on another subnet, then the peer should be the gateway host, not the boot host.

STATUS: Fixed.

---

SPR: **2357**

ARCH: Intel 960

BOARD: 80960

HOST: All

DESCRIPTION: **tcp.h** needs underscore added to **BYTE\_ORDER**, **LITTLE\_ENDIAN**, **BIG\_ENDIAN**.

STATUS: Fixed.

---

---

*SPR:* **2366**

*ARCH:* Intel 960

*BOARD:* 80960

*HOST:* All

*DESCRIPTION:* For the Intel 960 product the **d()** display function in the bootrom displays data in memory with incorrect byte order.

*STATUS:* Fixed.

---

*SPR:* **2369**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* Documentation for **loadModuleAt()** uses old loader flags. The new flags are:  
**LOAD\_NO\_SYMBOLS**  
**LOAD\_LOCAL\_SYMBOLS**  
**LOAD\_GLOBAL\_SYMBOLS**  
**LOAD\_ALL\_SYMBOLS**  
**HIDDEN\_MODULE**

*STATUS:* Fixed.

---

*SPR:* **2387**

*ARCH:* All

*BOARD:* All

*HOST:* All

Single step does not show fp registers when a task is spawned with **VX\_FP\_TASK** options on.

*STATUS:* Fixed.

---

SPR: **2391**

ARCH: All

BOARD: All

HOST: All

DESCRIPTION: The man pages for **printf()**, **sprintf()**, **vsprintf()**, and **sscanf()** list **ioLib.h** as the include file where the function prototypes are found; however, it should actually be **stdio.h**.

STATUS: Fixed.

---

SPR: **2423**

ARCH: SPARC

BOARD: sunec

HOST: All

DESCRIPTION: The function **vmContextCreate()** fails due to bug in **mmuSparcLib.c**.  
NOTE: this is for part MMU-100089-0003, the special version of mmu support for the sunec BSP 1.0 Beta.

The sequence of events is:

- (1) **mmuBufBlockAlloc()** is called—formats read-only page.
- (2) **mmuLstGet()** allocates a Region Table from that page.
- (3) Global Region Table is copied to the newly allocated Table. This table is not writable at this time (due to 1 above).

STATUS: Fixed.

---

SPR: **2444**

ARCH: All

BOARD: All

HOST: All

DESCRIPTION: The manual page for **tyLib** mentions the **ioctl()** function **FIOBUFSET** (Reference Manual, p.195). The name of this function is actually **FIORBUFSET**.

---

An additional **ioctl()** function **FIOWBUFSET** is not mentioned. This function sets the size of the send buffer.

STATUS: Fixed.

---

SPR: **2484**

ARCH: All

BOARD: All

HOST: All

DESCRIPTION: In the 5.1 Reference Manual, page 1-192, and the online manual page for **tyLib**, it says:

DESCRIPTION: **tyIRd()** - interrupt-level routine to deliver an input character  
**tyITx()** - interrupt-level routine to get the next output character

These lines should be:

**tyIRd()** - interrupt-level routine to get an input character  
**tyITx()** - interrupt-level routine to deliver the next output character

STATUS: Fixed.

---

SPR: **2486**

ARCH: All

BOARD: All

HOST: All

DESCRIPTION: Including **vw/h/time.h** and re-compiling yields an undefined reference to the function **clocks\_per\_sec()**; this routine does not exist in VxWorks.

STATUS: Fixed.

---

SPR: **2487**

ARCH: All

BOARD: All

HOST: All

DESCRIPTION: Documentation of how the shell deals with data (data structures) is incomplete.

STATUS: Fixed.

---

---

*SPR:* **2491**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* The manual entry for **cacheLib** in VxWorks 5.1 Reference Manual (Pg 1-30, line no 5) and the online man pages says:

“pointers in the cacheLib structure. For ...”

It should say:

“pointers in the **CACHE\_LIB** structure. For ...”

*STATUS:* Fixed.

---

*SPR:* **2493**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* The function **bcmp()** returns incorrect values. Instead of returning -1, 0, or -1 (as documented), it returns a negative integer, 0, or a positive integer.

*STATUS:* Fixed.

---

*SPR:* **2502**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* Many of the manual entries for subroutines listed in the **mathALib** man page fail to list the include file: **math.h**.

*STATUS:* Fixed.

---

---

<i>SPR:</i>	<b>2505</b>
<i>ARCH:</i>	SPARC
<i>BOARD:</i>	All
<i>HOST:</i>	All
<i>DESCRIPTION:</i>	The reference “basic” and “full” MMU support assumes “pass-through” mode when entering <b><i>mmuSparcLibInit()</i></b> . This means that the library does not support hardware implementations that use the entire 36-bit address space, or situations where “translate” mode is initiated prior to the initializing call to <b><i>mmuSparcLibInit()</i></b> .
<i>STATUS:</i>	Fixed.

---

<i>SPR:</i>	<b>2512</b>
<i>ARCH:</i>	All
<i>BOARD:</i>	All
<i>HOST:</i>	All
<i>DESCRIPTION:</i>	The manual entry for <b><i>fprintf()</i></b> under the list item “ # - The result is to be converted to an ...” mistakenly says:  For e, E, f, g, and g conversions...  The second “g” should be changed to “G”.
<i>STATUS:</i>	Fixed.

---

<i>SPR:</i>	<b>2513</b>
<i>HOST:</i>	All
<i>DESCRIPTION:</i>	The text under NOTE SPARC in the <b><i>intVecSet()</i></b> documentation is not clear (in the 5.1 Reference manual, p. 2-159).  The clause “...and puts it into the trap table entry corresponding to the specified vector” appears at the end of item (4) of a list that describes what the code generated by <b><i>intVecSet()</i></b> does.  However, that clause isn’t part of the activity described at (4); it refers to where <b><i>intVecSet()</i></b> places its generated code.

This clause should be formatted on a separate paragraph from the list of what the generated code does.

*STATUS:* Fixed.

---

*SPR:* **2520**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* The paragraph starting with “When either loading method...” in the manual entry for **loadModuleAt()** should be removed (VxWorks Reference Manual, p. 2-188). It applied to 5.0 but is no longer true for later versions.

*STATUS:* Fixed.

---

*SPR:* **2546**

*ARCH:* All

*BOARD:* 68360

*HOST:* HP 9000/700

*DESCRIPTION:* The doc for **hex.c** mentions **-v** (output vector info), and **-p** (PC vector), and **-s** (SP vector), but it doesn't mention what PC and SP will default to if **-v** is used and no values are given for **-p** and **-s**.

*STATUS:* Fixed.

---

*SPR:* **2547**

*ARCH:* SPARC

*BOARD:* All

*HOST:* Sun4

*DESCRIPTION:* Registers **g5**, **g6**, **g7** are reserved for kernel use; thus, the user should not use these registers; this is not documented.

*STATUS:* Fixed.

---

*SPR:* **2548**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* The ***flush()*** routine does not call the corresponding driver flush routines for the specified file descriptor.

*STATUS:* Fixed.

---

*SPR:* **2562**

*ARCH:* All  
*BOARD:* All  
*HOST:* Sun4

*DESCRIPTION:* The ***sscanf()*** routine does not always work correctly for [] type syntax.

*STATUS:* Fixed.

---

*SPR:* **2566**

*ARCH:* Intel 960  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* The i960 cannot be booted using the EXOS-202 ethernet controller, due to byte ordering problems.

*STATUS:* Fixed.

---

*SPR:* **2567**

*ARCH:* All

*BOARD:* N/A

*HOST:* All

*DESCRIPTION:* In the driver for the Intel 82596 ethernet controller chip, the ***eiWatchDog()*** routine uses a pointer before it has been initialized. As a result, transmit and receive timeouts may cause bus errors or other serious problems.

*STATUS:* Fixed.

---

*SPR:* **2576**

*ARCH:* SPARC

*BOARD:* mb930

*HOST:* Solaris

*DESCRIPTION:* The VxGDB demo program in ***src/demo/rdb*** is not correctly compiled for the SPARC*lite* processor. The source debugging information is not included, because the ***-g*** flag is not passed to the ***gcc*** compiler.

*STATUS:* Fixed.

---

*SPR:* **2580**

*ARCH:* Motorola 680x0 CPU32

*BOARD:* 68000

*HOST:* All

*DESCRIPTION:* The Motorola 68K software floating point library has a problem with the ***atan2()*** function. The arguments to the function are reversed.

*STATUS:* Fixed.

---

*SPR:* **2587**

*ARCH:* All  
*BOARD:* mv167  
*HOST:* Sun3 Sun4

*DESCRIPTION:* In sec 5.2.3 of the *VxWorks Programmer's Guide, 5.1* we cover booting from a local DOS file system. Although the example uses `/sd0/` as the boot device, we don't mention the caveat that the dosFs file system name *must* follow the convention of `/name/`—there must be leading and trailing slashes around the device name or the boot ROMs won't be able to build the correct device.

Note: this is just for SCSI boot; DOS-drive style names work fine programmatically and from the shell (e.g., **A:**). The exception is for SCSI booting only.

*STATUS:* Fixed.

---

*SPR:* **2599**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* It is not stated explicitly in the manuals that the virtual address, physical address, and length fields in a `PHYS_MEM_DESC` object are subject to the following limitation:

Each entry must be a multiple of `VM_PAGE_SIZE` (found in `configAll.h`).

Moreover, although section 6.8.1.7 *On-Board and Off-board Options* of the Network chapter in the Programmer's Guide mentions that off-board memory should often be non-cacheable, section 8.3 of the Programmer's Guide should repeat or cross-reference this information.

*STATUS:* Fixed.

---

*SPR:* **2609**

*ARCH:* Motorola 68040

*BOARD:* All

*HOST:* All

*DESCRIPTION:* The 68LC040 and 68EC040 processors do not have any hardware support for floating point operations. They require the pure software emulation libraries, as used with 68020 processors which lack math coprocessors.

*STATUS:* Fixed.

---

*SPR:* **2639**

*ARCH:* SPARC

*BOARD:* All

*HOST:* All

*DESCRIPTION:* Incorrect signal sequences in **sigfaulttable[]**.

*STATUS:* Fixed.

---

*SPR:* **2647**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* On p.109 of the Programmer's Guide, the discussion of **nanosleep()** implies that its resolution is in nanoseconds; however, it does not have such precision because it uses **taskDelay()**. The precision of **nanosleep()** is actually limited by the system clock rate.

*STATUS:* Fixed.

---

*SPR:* **2648**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* The **semLib.h** header file contains an ANSI prototype for **semCreate()**, but there is no corresponding non-ANSI declaration

*STATUS:* Fixed.

---

*SPR:* **2649**

*ARCH:* Motorola 680x0  
*BOARD:* 68000  
*HOST:* Sun4

*DESCRIPTION:* Under some circumstances, the GNU compiler may generate a library callout to a routine names **fixunssfsi()**. The VxWorks library does not contain such a routine.

*STATUS:* Fixed.

---

*SPR:* **2651**

*ARCH:* Intel 960  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* Under some conditions the **bcopy()** routine will perform buffer manipulations incorrectly. For backward copying cases, the alignment (qword, triple word, double word, word, or byte) is not determined correctly, sometimes causing an incorrect buffer manipulation.

Also, the pointer decrementing for the quadword backward copying is incorrect. The pointers are decremented by 22 decimal (0x16) bytes rather than 16 decimal.

*STATUS:* Fixed.

---

SPR: **2653**

ARCH: SPARC

BOARD: All

HOST: All

DESCRIPTION: The **intConnect()** call does not write enable the vector table before attempting to write to it, resulting in a failed operation.

STATUS: Fixed.

---

SPR: **2657**

ARCH: All

BOARD: All

HOST: All

DESCRIPTION: The following information about using the **ld()** command is not documented.

When a **.o** module is loaded using netDrv, there is not sufficient memory to hold the entire file at the time of loading; thus, the file is copied over the network before being read to the memory. Because of this, at the time of loading, you need memory space equal to twice the size of the module being loaded. The above situation does not happen when using NFS.

Example: If **file1.o** is 2MB, you need at least 4MB of available memory if you are using netDrv.

STATUS: Fixed.

---

SPR: **2666**

ARCH: All

BOARD: All

HOST: All

DESCRIPTION: **ledLib** does not deal with null characters correctly and you can overwrite memory by typing:

a^@bbbbbb *hold down any key until you get a bus error*

STATUS: Fixed.

---

---

*SPR:* **2673**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* The POSIX ***timer\_gettime()*** function returns the incorrect value. It always returns the time when the timer was started, rather than the remaining time.

*STATUS:* Fixed.

---

*SPR:* **2675**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* The prototype of ***xdr\_u\_char()*** is incorrect.

```
xdr_u_char(XDR *xds, char *cp);
```

should be

```
xdr_u_char(XDR *xds, u_char *ucp);
```

*STATUS:* Fixed.

---

*SPR:* **2690**

*ARCH:* Motorola 68000  
*BOARD:* All  
*HOST:* Sun4

*DESCRIPTION:* Using ***longjmp()*** in a signal handler can cause internal data structures to get corrupted

*STATUS:* Fixed.

---

SPR: **2705**

ARCH: All  
BOARD: All  
HOST: All

DESCRIPTION: In the documentation for **tyLib**, there is no indication on how to set up the parity, stop, and data bits. For example only the baud rate can be set with **FIOBAUDRATE**. It should be mentioned that for setting up the others, the user needs to manipulate **tyCodrv.c**, since they are not supported.

STATUS: Fixed.

---

SPR: **2706**

ARCH: All  
BOARD: All  
HOST: All

DESCRIPTION: When “inet on ethernet” of the boot parameter doesn’t have netmask set, the VxWorks boot ROM will try to get netmask by using **icmpMaskGet()**. The routine **icmpMaskGet()** should set the ip address to **ac\_ipaddr** but it doesn’t.

STATUS: Fixed.

---

SPR: **2744**

ARCH: All  
BOARD: All  
HOST: All

DESCRIPTION: Calling **closedir()** with an invalid directory pointer results in an attempt to free an invalid memory block.

STATUS: Fixed.

---

*SPR:* **2813**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* We do not specify a range of **errno** values that customers can use. Some want to define their own **errno** values, but are concerned with conflicts with WRS future use.

*STATUS:* Fixed.

---

*SPR:* **2847**

*ARCH:* SPARC  
*BOARD:* sun2ce  
*HOST:* Sun4

*DESCRIPTION:* For the Force CPU-2CE boards with over 16 megabytes of memory will not boot properly.

*STATUS:* Fixed.

---

*SPR:* **2908**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* If **memShow()** comes across an invalid block it incorrectly keeps the memory partition semaphore.

*STATUS:* Fixed.

---

*SPR:* **2949**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* The example output of ***dosFsConfigShow()*** in the *VxWorks Programmer's Guide*, section 5.2.2.6 does not show the fields for "available space" and "max avail. contig space".

*STATUS:* Fixed.

---

*SPR:* **2950**

*ARCH:* SPARC

*BOARD:* sparc

*HOST:* All

*DESCRIPTION:* The *VxWorks Programmer's Guide 5.1*, Table 9-2 on p.359 gives example invocations for the SPARC and SPARClite architectures using both the **-O** and **-O2** options. This is redundant. The **-O2** flag includes the optimization provided by **-O**.

*STATUS:* Fixed.

---

*SPR:* **2966**

*ARCH:* All

*BOARD:* 80386

*HOST:* All

*DESCRIPTION:* When ***msgQShow()*** is called with a level of 0, if there are tasks delayed on this message queue it is possible to have a data access exception.

*STATUS:* Fixed.

---

<i>SPR:</i>	<b>3005</b>
<i>ARCH:</i>	All
<i>BOARD:</i>	All
<i>HOST:</i>	All
<i>DESCRIPTION:</i>	<b>reld()</b> can cause memory to be overwritten.
<i>STATUS:</i>	Fixed.

---

<i>SPR:</i>	<b>3033</b>
<i>ARCH:</i>	All
<i>BOARD:</i>	All
<i>HOST:</i>	All
<i>DESCRIPTION:</i>	<p>The system runs a number of tasks, part are with <b>VX_FP</b> and part are without <b>VX_FP</b> option. Some tasks with and without <b>VX_FP</b> are created and deleted frequently. The tasks delete themselves by returning from the main function rather than being deleted by another task.</p> <p>It may take a while until the function <b>fppSwapHook()</b> calls <b>fppSave()</b> with a <b>NULL</b> pointer (<b>pFppContext</b>) thereby overwriting the interrupt vectors in low memory.</p> <p>This problem is due to the fpp switching optimization in <b>fppSwapHook()</b> which uses the local variable <b>_pLastFpTcb</b> to save the previous FP task that was running prior to context switch. In this case, that last task has just deleted itself, and its TCB is partially reclaimed, thus the <b>pFppContext</b> pointer is zeroed.</p> <p>The function <b>fppSwapHook()</b> fails to check for this condition and calls <b>fppSave()</b> with the <b>NULL</b> argument.</p>
<i>STATUS:</i>	Fixed.

---

<i>SPR:</i>	<b>3094</b>
<i>ARCH:</i>	All
<i>BOARD:</i>	All
<i>HOST:</i>	All
<i>DESCRIPTION:</i>	There is a bug in the macro <b>bcopy_from_mbufs()</b> in <b>mbuf.h</b> . It does not work correctly when the data width is anything but 0 or 1. This should only affect custom-

ers writing their own network drivers. There is a fix available through tech support.

*STATUS:* Fixed.

---

*SPR:* **3103**

*ARCH:* SPARC

*BOARD:* sparc

*HOST:* All

*DESCRIPTION:* On the frc3ce, uninitialized interrupts cause tNetTask to suspend, and further generation of interrupts causes further problems.

*STATUS:* Fixed.

---

*SPR:* **3115**

*ARCH:* SPARC

*BOARD:* All

*HOST:* All

*DESCRIPTION:* The SPARC and SPARClite versions of the *wind* kernel do not properly empty the "work queue." The work queue contains kernel requests which were made when the kernel was already busy doing other operations. As a result of not properly emptying the queue, the execution of these deferred kernel operations may be delayed unacceptably.

A patch for this problem is available from Wind River Technical Support, part number VXO-100622-0000.

*STATUS:* Fixed.

---

*SPR:* **3148**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* The IMPLEMENTATION section of reference manual for **rpcLib** is incorrect. It says:

The task-specific data structures are automatically deleted when the task exits.

One needs to free the memory manually using task hooks.

*STATUS:* Fixed.

---

*SPR:* **3176**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* The file `usrExtra.c` incorrectly checks `INCLUDE_CPLUS` and `INCLUDE_CPLUS_MIN`.

*STATUS:* Fixed.

---

*SPR:* **3177**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* The macro `MCLGET()` uses uninitialized pointers. Drivers that allocate mbuf clusters directly using `copy_to_mbufs()` (e.g. `if_sn.c!`) end up with whatever was previously in the mbuf as the entry in `m_freeRtn` thus possibly causing a system crash.

*STATUS:* Fixed.

---

*SPR:* **3188**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* The routine `ifloctlCall()` sets a bogus `errno` value.

*STATUS:* Fixed.

---

---

*SPR:* **3199**  
*ARCH:* Motorola 68030 68040  
*BOARD:* All  
*HOST:* Sun4  
*DESCRIPTION:* One cannot remove symbols built into a **vxWorks.st** standalone kernel without receiving a memPartFree error.  
*STATUS:* Fixed.

---

---

*SPR:* **3229**  
*ARCH:* Intel 960  
*BOARD:* 80960  
*HOST:* Sun4  
*DESCRIPTION:* **loadCoffLib** for i960 is unable to deal with multiple text sections. The common object file format(COFF) allows only 65535 line-number entries and 65535 relocation entries for any section within an object file. The linker creates additional sections whenever adding another input section to an existing output section, making the output section too big. The linker splits no single input section containing excess entries. A patch is available from support.  
*STATUS:* Fixed.

---

---

*SPR:* **3230**  
*ARCH:* Intel 960  
*BOARD:* All  
*HOST:* HP 9000/300 & 400  
*DESCRIPTION:* Appendix E.5 on page 486 of the VxWorks 5.1 Programmer's Guide refers to the file **vw/h/i960/iv960.h**. This file is actually **vw/h/i960/ivI960.h**.  
*STATUS:* Fixed.

---

---

*SPR:* **3238**

*ARCH:* All  
*BOARD:* All  
*HOST:* Sun4

*DESCRIPTION:* Paragraph 6.3.10 *Configuration of MBUFs* is not totally accurate. The section that discusses **memPartitionSize** states "...If this field is not large enough to accommodate the number of buffers specified in **initialAlloc**, the root task will suspend itself and print an error message on the console."

In **usrNetwork.c** we set **memPartitionSize** to 0.

Note, **memPartitionSize** is only used if **memPartition** is not NULL.

*STATUS:* Fixed.

---

*SPR:* **3260**

*ARCH:* Motorola 680x0  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* When using the **-msoft-float** option with **gcc68k**, if a floating point value less than zero is converted to an integer, the value is incorrectly rounded downwards (i.e. towards negative infinity) instead of towards zero. For example, -3.5 is rounded to -4, not the correct value of -3.

A patch which corrects this problem is available from Wind River Technical Support.

*STATUS:* Fixed.

---

*SPR:* **3269**

*ARCH:* MIPS R3000  
*BOARD:* hkv3500  
*HOST:* N/A

*DESCRIPTION:* When an RPC task exits by a mechanism that causes the **taskDeleteHook** to execute outside the context of the task, an access to an uninitialised pointer may oc-

cur. With R3000 BSPs, this may be evidenced by a "TLB Load Error." A patch for the SGI toolset is available from support.

STATUS: Fixed.

---

SPR: **3274**

ARCH: All

BOARD: All

HOST: All

DESCRIPTION: An attempt to **rlogin** to an engaged VxWorks target should indicate the current user. In that way it's possible to find the user. Instead of:

```
delaware:pat(vw.511.lance/config/frc40) 115> rlogin t44-160
Sorry, this system is engaged.
Connection closed.
```

It should be something like:

```
Sorry, the system is currently engaged by user: pat
Connection closed.
```

STATUS: Fixed.

---

SPR: **3309**

ARCH: All

BOARD: All

HOST: All

DESCRIPTION: A description of the following two issues should be added to the *VxWorks Programmer's Guide*.

- (1) VxWorks symbols cannot be removed in a VxWorks standalone image.
- (2) VxWorks local symbols are not in the symbol table in a VxWorks standalone image.

STATUS: Fixed.

---

SPR: **3334**

ARCH: Intel 960  
BOARD: All  
HOST: All

DESCRIPTION: The routine `ifMaskGet("ei0", &mask)` returns an error:  
  
uninitialized NMI: Bus Error: 80960 bus owner  
ip : 0x3b428 pcw: 0xd86088ff acw: 0xffff7efc

STATUS: Fixed.

---

SPR: **3363**

ARCH: SPARC  
BOARD: All  
HOST: All

DESCRIPTION: In the file `vw/h/vxLib.h`, `vxMemProbeAsi()` does not have any prototypes.

STATUS: Fixed.

---

SPR: **3395**

ARCH: All  
BOARD: All  
HOST: All

DESCRIPTION: Add a routine (i.e. `routeNetAdd()`) for adding a destination that is a subnet.

STATUS: Fixed.

---

SPR: **3399**

ARCH: SPARC  
BOARD: sun2ce  
HOST: Sun4

DESCRIPTION: Chap 10 of the Programmer's Guide doesn't mention `bootrom_uncmp`.

STATUS: Fixed.

---

---

**SPR:** 3427  
**ARCH:** All  
**BOARD:** All  
**HOST:** All  
**DESCRIPTION:** The programmer's guide should more clearly document the issue of using " " from the shell. This allocates memory that is not freed until the target is rebooted.  
**STATUS:** Fixed.

---

**SPR:** 3435  
**ARCH:** All  
**BOARD:** All  
**HOST:** N/A  
**DESCRIPTION:** Section 7.4.4 of the *VxWorks Programmer's Guide, 5.1* is incorrect.  
(1) **INCLUDE\_SH\_OBJ** should be **INCLUDE\_SM\_OBJ**.  
(2) The directions for setting up **config.h** are confusing. Most of the time nothing will have to be done in **config.h**.  
**STATUS:** Fixed.

---

**SPR:** 3441  
**ARCH:** Motorola 68040  
**BOARD:** 68040  
**HOST:** All  
**DESCRIPTION:** The sequence of events in the mmu initialization leaves some of the pages containing translation table data structures writable. This leaves the mmu's translation tables vulnerable to inadvertent corruption.  
**STATUS:** Fixed.

---

---

*SPR:* **3446**

*BOARD:* All  
*HOST:* Sun4 Solaris

*DESCRIPTION:* The **rpcLib** reference documentation (man page) should not refer to the obsolete library **dbxLib**, which is no longer part of VxWorks.

*STATUS:* Fixed.

---

*SPR:* **3450**

*ARCH:* Motorola 68030 68040  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* The **mmuLib** library obtains memory from the system memory manager via **memalign()** to contain the mmu's translation tables. This memory is allocated a page at a time on page boundaries. Unfortunately, in the current memory management scheme, the memory manager is not able to allocate these pages contiguously. Building large translation tables (e.g., when mapping large portions of virtual memory) causes excessive fragmentation of the system memory pool.

*STATUS:* Fixed.

---

*SPR:* **3461**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* In **/vw/h/usrConfig.h**, the function **printConfig()** is declared as an **extern**, even though it has been removed from **/config/all/usrConfig.c** and is not referenced anywhere.

*STATUS:* Fixed.

---

**SPR:** 3467  
**ARCH:** All  
**BOARD:** All  
**HOST:** N/A  
**DESCRIPTION:** The routine **copy()** in **usrLib.c** does not work correctly when the source file does not exist.  
**STATUS:** Fixed.

---

**SPR:** 3480  
**ARCH:** Intel 486  
**BOARD:** epc4  
**HOST:** All  
**DESCRIPTION:** Part 2, the *Reference Manual Supplement*, of the document *VxWorks for Intel i386/i486 Architecture Supplement, 5.1.1* only includes subroutine man pages for the PC386/486 version of the BSP.  
The correct reference documentation for the RadiSys EPC4 is somewhat different; the differences are significant for anything involving VME. The correct EPC4 man pages are available online.  
**STATUS:** Fixed.

---

**SPR:** 3490  
**ARCH:** All  
**BOARD:** All  
**HOST:** All  
**DESCRIPTION:** In the man page for **ansiTime()**, under the “ZONE & Daylight Saving” heading, the example for specifying the **TIMEZONE** environment variable is not in the correct form. The example is given as:  
`UTC:UTC:0:040102:100102`  
when it should be:  
`UTC::0:040102:100102`

---

(i.e. "UTC" should appear only once).

*STATUS:* Fixed.

---

*SPR:* **3515**

*ARCH:* Intel X86

*BOARD:* 80386

*HOST:* Sun4

*DESCRIPTION:* When trying to use a local disk, *ideDrv()* returns an error.

*STATUS:* Fixed.

---

*SPR:* **3516**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* The TCP mss is set off the mbuf cluster size. Thus, 1024 (1078 with IP header) is the maximum TCP packet that the target sends on the ethernet. This does not take advantage of the full enet bandwidth.

*STATUS:* Fixed.

---

*SPR:* **3518**

*ARCH:* Intel X86

*BOARD:* pc486

*HOST:* Sun4

*DESCRIPTION:* 3Com's etherLink III doesn't work with Gateway 2000 DX2-66.

*STATUS:* Fixed.

---

<i>SPR:</i>	<b>3536</b>
<i>ARCH:</i>	All
<i>BOARD:</i>	All
<i>HOST:</i>	Sun4
	The routine <b><i>getcwd()</i></b> is not prototyped anywhere. The prototype should be in <b><i>unistd.h</i></b> .
<i>STATUS:</i>	Fixed.

---

<i>SPR:</i>	<b>3538</b>
<i>HOST:</i>	Sun3 Sun4 Solaris
<i>DESCRIPTION:</i>	The functions in <b><i>symLib</i></b> use a parameter with the defined type <b><i>SYM_TYPE</i></b> . Possible values for objects of this type are defined in <b><i>a_out.h</i></b> , but not documented. The VxWorks man page for <b><i>symLib</i></b> should describe the range of values for <b><i>SYM_TYPE</i></b> objects.
<i>STATUS:</i>	Fixed.

---

<i>SPR:</i>	<b>3541</b>
<i>ARCH:</i>	Intel 960
<i>BOARD:</i>	All
<i>HOST:</i>	All
<i>DESCRIPTION:</i>	The macros <b><i>LSW()</i></b> and <b><i>MSW()</i></b> are missing from the <b><i>vxWorks.h</i></b> header file.
<i>STATUS:</i>	Fixed.

---

<i>SPR:</i>	<b>3542</b>
<i>ARCH:</i>	All
<i>BOARD:</i>	All
<i>HOST:</i>	All
<i>DESCRIPTION:</i>	The routine <b><i>ctime()</i></b> results in a bus error when values above <b><i>0x7fffffff</i></b> are used.
<i>STATUS:</i>	Fixed.

---

SPR: **3550**

ARCH: All  
BOARD: All  
HOST: All

DESCRIPTION: The routine **connectWithTimeout()** is a useful function... it should be documented.

STATUS: Fixed.

---

SPR: **3551**

ARCH: All  
BOARD: All  
HOST: All

DESCRIPTION: Certain socket options (e.g. snd/rcv timeouts, snd/rcv lowat marks) are not implemented in the WRS stack. An error condition should be returned, signifying that these are not supported options.

STATUS: Fixed.

---

SPR: **3552**

ARCH: All  
BOARD: All  
HOST: All

DESCRIPTION: Several tasks could sleep while waiting for socket buffer space, but **sbwakeup()** (really **wakeup()**) only wakes up one task.

STATUS: Fixed.

---

---

*SPR:* **3557**

*ARCH:* Intel X86

*BOARD:* pc386

*HOST:* Sun4

*DESCRIPTION:* After doing **spy()** and **spyStop()** from the shell, `rlogin host` doesn't work.

*STATUS:* Fixed.

---

*SPR:* **3561**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* The **scanf()** family of functions is supposed to stop processing and return if any single scan fails. However, this is not done; the functions continue processing even after a scan fails.

*STATUS:* Fixed.

---

*SPR:* **3564**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* If a block size is changed with **memPartRealloc()**, the count of the number of allocated blocks is sometimes decremented.

The problem is that if adjacent memory can be used for the new block, the adjacent memory is carved out, and the remainder of the adjacent memory is freed with a call to **memPartFree()**—which causes the allocation stats to be decremented.

*STATUS:* Fixed.

---

---

*SPR:* **3572**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* The ***usrScsiConfig()*** routine contains references to both ***dosFsDevInit()*** and ***rt11FsDevInit()***, which causes both of these libraries to be included whenever SCSI is used.

*STATUS:* Fixed.

---

*SPR:* **3582**

*ARCH:* N/A  
*BOARD:* All  
*HOST:* N/A

*DESCRIPTION:* The interaction between the use of ***intLock()*** and making kernel calls needs to be better documented.

*STATUS:* Fixed.

---

*SPR:* **3632**

*ARCH:* SPARC  
*BOARD:* frc3ce  
*HOST:* Sun4

*DESCRIPTION:* Including **MMU\_FULL** with the frc3ce/sun3ce BSP results in multiple  
vmLib.o(.text+xxx): undefined reference to `'_mmuPhysAddrShifted'`  
messages. A patch is available from support.

*STATUS:* Fixed.

---

*SPR:* **3645**

*ARCH:* Intel X86

*BOARD:* pc386

*HOST:* Sun4

*DESCRIPTION:* Booting from IDE hard disk doesn't work on DOS6.0 format.

*STATUS:* Fixed.

---

*SPR:* **3646**

*ARCH:* Intel X86

*BOARD:* pc486

*HOST:* Sun4

*DESCRIPTION:* The SMC's Ultra Ethernet card has low throughput.

*STATUS:* Fixed.

---

*SPR:* **3695**

*ARCH:* Motorola 680x0

*BOARD:* All

*HOST:* Sun4

*DESCRIPTION:* ***printf()*** with a format of "% .20g" sometimes prints out more than 20 characters after the decimal point.

*STATUS:* Fixed.

---

*SPR:* **3712**

*ARCH:* N/A

*BOARD:* N/A

*HOST:* N/A

*DESCRIPTION:* The following libraries are mentioned in some of the man pages. These libraries are obsolete or superseded by others.

- **pathLib**—in man page for ***iosDevFind()***.
- **strLib**—in man page for **bALib** and **bLib**.

- **bootpdLib**
- **dbxLib**
- **intALib**

STATUS: Fixed.

---

SPR: **3718**

ARCH: All  
BOARD: All  
HOST: All

DESCRIPTION: There is a missing semicolon in the version of the header file **vw/h/private/trig.h**, as supplied with release 5.1.1.

STATUS: Fixed.

---

SPR: **3721**

ARCH: Intel X86  
BOARD: pc386  
HOST: Sun4

DESCRIPTION: If an interrupt occurs right after saving the signal context, the signal context may be destroyed.

STATUS: Fixed.

---

SPR: **3734**

ARCH: Intel 386  
BOARD: 80386  
HOST: Sun4

DESCRIPTION: The manual page for **ideDevCreate()** has an error. **ideDrv** is intended to be used with hard disks not “floppy” disks. The *drive* is a drive number of the hard disk device as defined in the structure table **ideTypes[]** in **sysLib.c**. That table has two entries: the first one is for drive 0, second one is for drive 1.

STATUS: Fixed.

---

SPR: **3737**

ARCH: Intel X86  
BOARD: pc386  
HOST: Sun4

DESCRIPTION: The **sysMemTop()** routine does not check for the size of memory automatically, instead it simply returns (**LOCAL\_MEM\_LOCAL\_ADRS + LOCAL\_MEM\_SIZE**).

STATUS: Fixed.

---

SPR: **3739**

ARCH: N/A  
BOARD: N/A  
HOST: N/A

DESCRIPTION: In the *VxWorks Programmer's Guide, 5.1* page 31, section 2.2.2.2, the way to set the environment variable **PATH** in the C shell is incorrect

Instead of

```
setenv PATH $PATH:/usr/gnu/...
```

(which gives the error "Variable syntax"), the correct command is:

```
setenv PATH ${PATH}:/usr/gnu/...
```

STATUS: Fixed.

---

SPR: **3750**

ARCH: Intel X86  
BOARD: pc386  
HOST: Sun4

DESCRIPTION: Bootrom doesn't reboot when it gets a general protection fault.

```
[VxWorks Boot] d $500000  
500000:  
General Protection Fault  
Program Counter: 0x0000a241  
Status Register: 0x00010287  
Error Code: 0x00420000  
Task: 0x42b71c "tBoot"
```

It is supposed to reboot after this, but it doesn't.

*STATUS:* Fixed.

---

*SPR:* **3758**

*ARCH:* Intel X86

*BOARD:* pc386

*HOST:* Sun4

Booting **vxWorks** from IDE drive causes memPartAlloc error, if the drive has a lot of files.

*STATUS:* Fixed.

---

*SPR:* **3759**

*ARCH:* Intel X86

*BOARD:* pc386

*HOST:* Sun4

*DESCRIPTION:* The keyboard driver doesn't work on some Gateway2000 towers. The keyboard works at the boot prompt (after the reset). However, at the shell prompt (after booting **vxWorks**), the keyboard driver doesn't understand what's typed in. It never returns the prompt back.

*STATUS:* Fixed.

---

*SPR:* **3765**

*ARCH:* Intel X86

*BOARD:* pc386

*HOST:* Sun4

*DESCRIPTION:* The NW bit in control register 0 should not be set.

The Pentium processor user's manual volume 3, page 23-12 says:

CD=1 NW=1 : Cache disabled. Memory consistency not maintained.

CD=1 NW=0 : Cache disabled. Memory consistency maintained.

This affects the **bootrom** image which disables the cache. It doesn't affect **vxWorks** and **vxWorks.st**, because they enable the cache.

STATUS: Fixed.

---

SPR: **3769**

ARCH: All

BOARD: All

HOST: All

The bottom two paragraphs on p. 2-188 in the Reference Manual are first of all, completely different than the actual functionality, and secondly, are extremely difficult to read.

STATUS: Fixed.

---

SPR: **3785**

ARCH: All

BOARD: All

HOST: All

DESCRIPTION: The documentation for **sendto()** should mention that the maximum length of the buffer to be sent depends on the UDP buffer size. This buffer size may be set with **setsockopt()**.

STATUS: Fixed.

---

SPR: **3811**

ARCH: N/A

BOARD: N/A

HOST: All

DESCRIPTION: In Table 3-19 *Show Routines* on p.111 of the *VxWorks Programmer's Guide, 5.1*, **taskHookShow()** is listed as a subroutine. There is no such subroutine. However, the **taskHookShow** library provides three routines that identify create hooks, switch hooks, and delete hooks.

STATUS: Fixed.

---

---

*SPR:* **3828**  
*ARCH:* All  
*BOARD:* All  
*HOST:* Sun4

The **tyLib** manpage, *I/O CONTROL FUNCTIONS* section, *FIOCANCEL* subsection, contains the following text:

```
FIOCANCEL - Cancels a read or write. A task blocked on a
read may have previously started a watchdog routine to
time out read. The watchdog would use this call on the
appropriate file descriptor:
    status = ioctl (fd, FIOCANCE, 0);
```

The statement `ioctl (fd, FIOCANCEL, 0)` is incorrect. The *ioctl()* routine with the function **FIOCANCEL** calls *semTake()* which cannot be done in interrupt context.

*STATUS:* Fixed.

## 9. Known Problems in VxWorks 5.2

The SPRs (Software Problem Reports) listed below are outstanding for VxWorks 5.2 at the time of publication.

This list covers only SPRs affecting VxWorks itself, or affecting the BSPs released together with VxWorks 5.2. Most BSPs required no revision for VxWorks 5.2; see §6.1 BSPs (above) for a list of the BSPs revised with this release.

---

**NOTE:** For the most up-to-date list of known problems in VxWorks, and in all BSPs, contact your Technical Support representative. In North America, call 1-800-872-4977 with your Support ID number. In other countries, contact your national WRS organization (see back cover).

---

---

*SPR:* **1539**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* No warning message is returned when an unsuccessful write to NVRAM is made. An unsuccessful write to NVRAM most commonly occurs when booting VxWorks with a bootline which exceeds the capacity of the on board NVRAM. In this case, VxWorks boots successfully, however, the new bootline is not written to NVRAM and subsequent reboots will revert to the bootline last written to NVRAM.

*WORKAROUND:* Ensure that the bootline does not exceed the value of `BOOT_LINE_SIZE` (in `configAll.h/config.h`).

---

*SPR:* **1930**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* The documentation for `semTake()` does not explain how to differentiate an **ERROR** returned due to timeout vs. some other error.

*WORKAROUND:* None.

---

<i>SPR:</i>	<b>2003</b>
<i>ARCH:</i>	All
<i>BOARD:</i>	All
<i>HOST:</i>	All
<i>DESCRIPTION:</i>	There is no information regarding what sort of performance penalty is imposed by using the MMU/VxVMI facilities, or how they can be tuned to minimize the performance degradation.
<i>WORKAROUND:</i>	None.

---

<i>SPR:</i>	<b>2094</b>
<i>ARCH:</i>	All
<i>BOARD:</i>	All
<i>HOST:</i>	Solaris
<i>DESCRIPTION:</i>	Making <b>vxWorks.st_rom</b> fails on any SVR4 host. <b>Makefile.archtool</b> calls <code>@ \$(RM) \$(IMI) \$(IBR)</code> which evaluates to <code>rm -f</code> with no file on non-960 architectures. SVR4 hosts return an error code 2 when no file name is specified for the <b>rm</b> command.
<i>WORKAROUND:</i>	<ol style="list-style-type: none"><li>(1) <b>Makefile.archtool</b> may be edited to define <code>IBR = foo</code> or <code>IMI = foo</code>.</li><li>(2) <b>MakeSkel</b> may be edited to define <code>IBR = foo</code> or <code>IMI = foo</code>.</li><li>(3) <code>%setenv IBR foo</code> or <code>%setenv IMI foo</code>.</li></ol>

---

<i>SPR:</i>	<b>2238</b>
<i>ARCH:</i>	All
<i>BOARD:</i>	All
<i>HOST:</i>	All
<i>DESCRIPTION:</i>	The <b>cacheDmaMalloc()</b> routine provides cache coherency on a particular board; however, a VME access occurs to this board's local memory can cause a problem, depending on the board. There is no indication in the documentation that a problem could occur.
<i>WORKAROUND:</i>	None.

---

*SPR:* **2320**

*ARCH:* SPARC

*BOARD:* sun2ce

*HOST:* All

*DESCRIPTION:* On the Force CPU-2CE using ^X to reboot a master board causes the slave to reboot as well as the master board.

*WORKAROUND:* None.

---

*SPR:* **2322**

*ARCH:* SPARC

*BOARD:* sun2ce

*HOST:* All

*DESCRIPTION:* On the Force CPU-2Ce, with data and instruction caches turned off, it is reported that a board with 64Mbyte memory cannot boot when `LOCAL_MEM_SIZE` is set to greater than 32Mbyte.

*WORKAROUND:* Available from WRS Technical Support.

---

*SPR:* **2376**

*ARCH:* SPARC

*BOARD:* sun2ce

*HOST:* All

*DESCRIPTION:* On the Force CPU-2CE, the functions `sysIntDisable()` and `sysIntEnable()` do not work for level 4 interrupts.

*WORKAROUND:* Before the `return(OK)` statement, the routines need:

In the `sysIntDisable()` routine:

```
if (intLevel == 4)
    *SUN_IR &= 0xef;
```

In the `sysIntEnable()` routine:

```
if (intLevel == 4)
    *SUN_IR |= 0x10;
```

---

<i>SPR:</i>	<b>2395</b>
<i>ARCH:</i>	All
<i>BOARD:</i>	N/A
<i>HOST:</i>	N/A
<i>DESCRIPTION:</i>	<p>An attempt to single-step into a call to <b><i>bcopy()</i></b> using <b><i>s()</i></b> has the effect of doing a <b><i>c()</i></b>—that is, the task runs until it hits a breakpoint or exits.</p> <p>This is one symptom of a more general problem; the root cause is the invocation of user-callable routines within a section of code in <b><i>dbgLib</i></b> that handles breakpoint processing. On architectures that lack a trace mode (e.g., SPARC, MIPS), the problem is more pronounced, in that it will occur when stepping into such routines as well as when hitting a breakpoint within them. And, on such architectures, the problem will be yet further highlighted by the use of VxGDB, which will step into a routine (invisibly to the user) when single-stepping <i>over</i> a call to the routine.</p> <p>The problem is not architecture-specific, although it will manifest itself in different circumstances on different architectures. On the SPARC, for example, routines that provide runtime support for integer divide and multiply are affected, meaning that the problem can occur when single-stepping through code with no explicit function calls at all.</p> <p>Other routines that are known to cause problems:</p> <p><b><i>cacheClearEntry()</i></b>  <b><i>taskIdSelf()</i></b>  <b><i>taskOptionsSet()</i></b>  <b><i>taskRegsGet()</i></b>  <b><i>taskTcb()</i></b></p>
<i>WORKAROUND:</i>	None.

---

<i>SPR:</i>	<b>2499</b>
<i>ARCH:</i>	All
<i>BOARD:</i>	All
<i>HOST:</i>	All
<i>DESCRIPTION:</i>	<p>The documentation (man pages) for <b><i>memLib</i></b> omit some information. If the <b><i>MEM_ALLOC_ERROR_SUSPEND_FLAG</i></b> flag is set via a call to <b><i>memOptionsSet()</i></b>,</p>

only tasks which are spawned without the **VX\_UNBREAKABLE** flag are suspended following attempts to **malloc()** more memory than is available.

**WORKAROUND:** None.

---

**SPR:** 2529

**ARCH:** All

**BOARD:** All

**HOST:** All

**DESCRIPTION:** The default kernel is compiled with **INCLUDE\_SCSI** defined, but if a board does not have a SCSI chip the boot will hang when **usrScsiConfig()** is called. This is because the status returned by **sysScsiInit()** is not checked.

**WORKAROUND:** In the file **config/all/usrConfig.c**, in the function **usrRoot()** change the following:

```
#ifndef INCLUDE_SCSI
    sysScsiInit ();          /* initialize SCSI controller */
    usrScsiConfig ();       /* configure SCSI peripherals */
#endif /* INCLUDE_SCSI */
```

To the following:

```
#ifndef INCLUDE_SCSI
    if (sysScsiInit () != ERROR) /* initialize SCSI controller */
        usrScsiConfig ();       /* configure SCSI peripherals */
#endif /* INCLUDE_SCSI */
```

---

**SPR:** 2555

**ARCH:** SPARC

**BOARD:** sun2ce

**HOST:** All

**DESCRIPTION:** The instructions for the Force CPU-2CE to program the OPEN BOOT to boot from the flash memory does not work. The space before "loop" should be removed.

**WORKAROUND:** Replace the line:

```
4: 100000 0 do flash-va i + 1@ load-base i + 1! /1 + loop
```

with the line:

```
4: 100000 0 do flash-va i + 1@ load-base i + 1! /1 +loop
```

---

<i>SPR:</i>	<b>2556</b>
<i>ARCH:</i>	All
<i>BOARD:</i>	All
<i>HOST:</i>	All
<i>DESCRIPTION:</i>	When initializing a disk, dosFs does not write 0x55aa to the last two bytes of the boot sector. Some systems check this value to determine if the block is in fact a valid boot sector, so disks initialized under VxWorks cannot be read.
<i>WORKAROUND:</i>	The user can init the disk in the regular way under VxWorks (i.e. use <b>dosFsDevInit()</b> then <b>ioctl(FIODISKINIT)</b> ). Then, using <b>scsiRdSecs()</b> and <b>scsiWrtSecs()</b> (or equivalent low-level read/write calls, if using something other than SCSI), the user can read the first block, modify the buffer to include 0x55aa in the last 2 bytes, then write the block back out.

---

<i>SPR:</i>	<b>2575</b>
<i>ARCH:</i>	All
<i>BOARD:</i>	All
<i>HOST:</i>	All
<i>DESCRIPTION:</i>	To use the system symbol table, you need to use the <b>sysSymTbl</b> variable. This variable is not documented in <b>symLib</b> , and should be.
<i>WORKAROUND:</i>	None.

---

<i>SPR:</i>	<b>2601</b>
<i>ARCH:</i>	Motorola 680x0
<i>BOARD:</i>	All
<i>HOST:</i>	All
<i>DESCRIPTION:</i>	<b>^X</b> does not cause processors 1–n to begin their reboot process while booting from <b>sm</b> as it did for <b>bp</b> in 5.0.2. During boot from <b>sm</b> , <b>^X</b> is apparently disabled.
<i>WORKAROUND:</i>	None.

---

**SPR:** 2608

**ARCH:** All

**BOARD:** All

**HOST:** All

**DESCRIPTION:** There are several routines which are used internally by the debugger (**dbgLib**), rendering them ineligible for normal debugging (e.g. setting breakpoints). The involved routines include **taskIdSelf()**, **taskTcb()**, and floating point multiply and divide callouts generated by the compiler.

**WORKAROUND:** None.

---

**SPR:** 2610

**ARCH:** SPARC

**BOARD:** sun2ce

**HOST:** All

**DESCRIPTION:** With the Force CPU-2CE when using a specified bootline to load the VxWorks image, the VxWorks image then uses the bootline from non-volatile RAM for the path name to try to load the symbol table. This is only when the bootline is specified as:

[VxWorks Boot]: `$ln(0,0)host:/vw/config/sun2ce/vxWorks ...`

If the boot parameters are modified by the change command then the VxWorks image and symbol table boot properly.

**WORKAROUND:** None.

---

**SPR:** 2611

**ARCH:** SPARC

**BOARD:** sun2ce

**HOST:** Sun4

**DESCRIPTION:** With the 5.0 backplane driver enabled and multiple targets running VxWorks, booting the sun2ce with mailbox interrupt enabled (**SM\_INT\_MAILBOX\_1**) sometimes results in bad mailbox interrupts and general communication failure on backplane after booting.

**WORKAROUND:** The fix can be done in one of two ways.

---

- (1) in your **config.h** file change from

```
#define SM_INT_ARG1      VME_AM_USR_SHORT_IO
to
```

```
#define SM_INT_ARG1      VME_AM_SUP_SHORT_IO
```

—OR—

- (2) In your **sysLib.c** file change from

```
STATUS sysBusToLocalAdrs
switch (adrsSpace)
{
    case VME_AM_SUP_SHORT_IO:
        busAdrs = (char *) ((UINT) busAdrs & 0x0000FFFF);
        busAdrs = (char *) (SUN_MMU_A16_D16 | (UINT) busAdrs);
        break;
    ...
    case VME_AM_USR_SHORT_IO:
    default:
        return (ERROR);
        break;
}
```

to

```
STATUS sysBusToLocalAdrs
switch (adrsSpace)
{
    case VME_AM_USR_SHORT_IO:
    case VME_AM_SUP_SHORT_IO:
        busAdrs = (char *) ((UINT) busAdrs & 0x0000FFFF);
        busAdrs = (char *) (SUN_MMU_A16_D16 | (UINT) busAdrs);
        break;
    ...
}
```

---

**SPR:** 2619  
**ARCH:** SPARC  
**BOARD:** sparc  
**HOST:** All  
**DESCRIPTION:** On SPARC targets, during a remote login if there is an exception then VxWorks crashes when handling exceptions.  
**WORKAROUND:** None.

---

**SPR:** 2623  
**ARCH:** Motorola 68040  
**BOARD:** mv167  
**HOST:** All  
**DESCRIPTION:** When using a 68040 based BP master, and **INCLUDE\_BP\_50** (to enable booting a 5.0.x target over the backplane), the BP anchor and memory pool must be allocated in a cache-safe way. As shipped in 5.1 and 5.1.1, it does not. This can prevent a slave from booting, and may cause a bp master's tNetTask to incur a bus error.  
**WORKAROUND:** Available from WRS Technical Support.

---

**SPR:** 2638  
**ARCH:** All  
**BOARD:** All  
**HOST:** All  
**DESCRIPTION:** When displaying configuration data for a disk with over 2 Gigabytes of free space, **dosFsConfigShow()** displays a negative value for number of free bytes.  
**WORKAROUND:** None.

---

*SPR:* **2646**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* The ***ioctl()*** function **FIOSYNC** is not mentioned in the manual entry for **nfsDrv**; however, it is mentioned in the VxWorks Programmers Guide, section 4.6.4.2.

*WORKAROUND:* None.

---

*SPR:* **2650**

*ARCH:* Motorola 68000  
*BOARD:* All  
*HOST:* Sun4

*DESCRIPTION:* The test-and-set (TAS) retry logic in the 68000 and 68010 is incorrect and can result in an infinite loop when handling a bus error.

*WORKAROUND:* A patch is available from Wind River technical support.

---

*SPR:* **2679**

*ARCH:* SPARC  
*BOARD:* sun2ce  
*HOST:* All

*DESCRIPTION:* On the Force CPU-2CE BSP the ***sysLocalToBusAdrs()*** returns an **ERROR** for the **VME\_AM\_SUP\_SHORT\_IO** case, which it should not do. This should be added to the case for the short I/O address space.

*WORKAROUND:* None.

---

*SPR:* **2691**

*ARCH:* SPARC

*BOARD:* sun2ce

*HOST:* All

*DESCRIPTION:* For the Force CPU-2CE BSP, in **memDesc.c** the entries for the A24/D16 space and the A16/D16 space overlap. The size for the A24/D16 should be reduced to 0x30000 from 0x40000.

*WORKAROUND:* None.

---

*SPR:* **2738**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* Performing a **creat()** on a dosFs file system does not immediately create a directory entry for the file. It waits until the file descriptor returned by the **creat()** call has been closed.

*WORKAROUND:* None.

---

*SPR:* **2739**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* If a seek is done past the end of a dosFs file and data is written, the **write()** operation will only write to the end of the cluster containing the file position specified during the seek. In other words, the **write()** will fail to allocate any additional clusters beyond that which was required by the seek itself. The byte count returned from **write()** will indicate the amount of data written, which will be less than that specified. Subsequent **write()** operations from that point will succeed normally.

*WORKAROUND:* None.

---

*SPR:* **3051**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* In the **wrs.an** macros, the definitions of **.bS** and **.bE** contain changes in vertical spacing. Some shell configurations cause man to try to interpret the change in vertical spacing, which results in vertically compressed output.

*WORKAROUND:* Remove **.vs** requests from the definitions of **.bS** and **.bE** in **wrs.an**.

---

*SPR:* **3085**

*ARCH:* Motorola 68040  
*BOARD:* mv167  
*HOST:* All

*DESCRIPTION:* The **fmod.x** instruction on the 68040 does not return accurate results. Other variants of **fmod** work correctly.

This only affects the actual emulation of the **fmod.x** instruction. It does not affect operation of the **fmod()** C library function.

*WORKAROUND:* None.

---

*SPR:* **3092**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* When a task is restarted after an exception occurs, that exception may still be reported.

*WORKAROUND:* None.

---

---

*SPR:* **3128**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* If a new file is created under dosFs, then a seek is performed past EOF and data is written, no disk space is in fact allocated. This is due to a test in **dosFsClustNew()** which compares the new offset with the file size - 1. Since the file size is zero, the size -1 becomes (unsigned) 0xffffffff and the test never fails. So, dosFs assumes the new offset is within the allocated space and does not allocate any new clusters to the file.

*WORKAROUND:* None.

---

*SPR:* **3221**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* When creating a subdirectory via **mkdir()** or **ioctl(FIOMKDIR)**, dosFs does not check that the parent directory is in fact a directory, resulting in a subdirectory that does not have a valid parent.

*WORKAROUND:* None.

---

*SPR:* **3250**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* If the size of the VxWorks image is too large, the loading of the VxWorks system image during the boot-strap process will fail. This problem occurs because the loaded image over writes the boot strap code in the targets memory.

*WORKAROUND:* This problem can be avoided in two ways.

- (1) The maximum size the VxWorks image can be, without the load failure, is **RAM\_HIGH\_ADRS - RAM\_LOW\_ADRS**. These values are defined in the target specific BSP directory in the file **MakeSkel**. One way to avoid this loading problem is to reduce the size of the VxWorks image. This can be

---

accomplished by undefining unneeded options in **configAll.h** and rebuilding VxWorks. If the image cannot be reduced enough then option number two must be used.

- (2) Move the location where the bootstrap code resides in memory. This can be accomplished by increasing the value of the constant **RAM\_HIGH\_ADRS**. This constant must be changed in the in target specific BSP directory in the files **MakeSkel**, **Makefile.cpu**, and **config.h** (it needs to be changed in all locations) and building new bootroms.

---

*SPR:* **3306**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* Using VxWorks 5.1.1 and dosFs, one can create duplicate filenames by renaming a file to an existing filename when performing this from a different directory.

*WORKAROUND:* None.

---

*SPR:* **3314**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* The **tyLib** module is initialized before **selectLib**. When **tyLib** goes to call **\_func\_selWakeupListInit()**, the function pointer is null because **selectLibInit()** has not been called yet. As a result, **select()** will not work properly on **tyLib** devices.

*WORKAROUND:* None.

SPR: **3343**

ARCH: SPARC

BOARD: sun2ce

HOST: All

DESCRIPTION: On the Force CPU-2CE the entry for the S-bus slot 1 in **memDesc.c** is incorrect, and should be changed. The last flag, **VM\_STATE\_DS\_MEM**, should be **VM\_STATE\_DS\_IO**.

WORKAROUND: Edit the entry for the S-bus slot 1 in **memDesc.c** as follows.

Current entry:

```
/* S-bus slot 1 */
{
(void *) (LOCAL_MEM_LOCAL_ADRS + LOCAL_MEM_SIZE),
(void *) 0xFA000000,
0x00040000,
VM_STATE_MASK_VALID|VM_STATE_MASK_WRITABLE|
VM_STATE_MASK_DEVICE_SPACE|VM_STATE_MASK_CACHEABLE,
VM_STATE_VALID|VM_STATE_WRITABLE|VM_STATE_CACHEABLE_NOT|VM_STATE_DS_MEM
},
```

Revised entry:

```
/* S-bus slot 1 */
{
(void *) (LOCAL_MEM_LOCAL_ADRS + LOCAL_MEM_SIZE),
(void *) 0xFA000000,
0x00040000,
VM_STATE_MASK_VALID|VM_STATE_MASK_WRITABLE|
VM_STATE_MASK_DEVICE_SPACE|VM_STATE_MASK_CACHEABLE,
VM_STATE_VALID|VM_STATE_WRITABLE|VM_STATE_CACHEABLE_NOT|VM_STATE_DS_IO
},
```

---

SPR: **3370**

ARCH: Motorola 68040

BOARD: All

HOST: All

DESCRIPTION: The **sysPhysMemDesc[]** tables in 5.1.x BSP's use **VM\_STATE\_CACHEABLE**, which for the 68040 is interpreted as **VM\_STATE\_CACHEABLE\_COPYBACK**. However, many boards only support write-through (as opposed to copyback)

caching. This may result in copyback caching being enabled even though the hardware does not properly support it, causing cache incoherency and data corruption.

This also violates the comment in the **configAll.h** header file which indicates that write-through mode is the default.

**WORKAROUND:** In the **sysLib.c** file for a particular board support package, change the **VM\_STATE\_CACHEABLE** symbols in the **sysPhysMemDesc** table to **VM\_STATE\_CACHEABLE\_WRITETHROUGH**.

---

**SPR:** 3428

**ARCH:** All  
**BOARD:** All  
**HOST:** All

**DESCRIPTION:** When removing a module using **unld()**, all breakpoints in the system are removed.

**WORKAROUND:** None.

---

**SPR:** 3477

**ARCH:** All  
**BOARD:** All  
**HOST:** All

**DESCRIPTION:** If the **malloc()** of the one-cluster I/O buffer for a dosFs file descriptor fails during the **creat()** call, the allocated file descriptor is not freed.

**WORKAROUND:** None.

---

*SPR:* **3579**

*ARCH:* SPARC

*BOARD:* sparc

*HOST:* All

*DESCRIPTION:* On some BSPs, if the SPARC is the master and sm0 memory is on-board, the memory used for sm0 is not being made non-cacheable. (This is also true for smNet.)

*WORKAROUND:* For BSPs that have this problem, turn caching off when using on-board memory for VxMP and smNet.

---

*SPR:* **3595**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* If **clock\_gettime()** is interrupted between its two reads of the system clock, it can return an invalid value.

*WORKAROUND:* None.

---

*SPR:* **3600**

*ARCH:* All

*BOARD:* All

*HOST:* All

*DESCRIPTION:* When a seek is performed on a newly-created (zero-length) dosFs file, and that seek extends beyond the first cluster of the file, the first cluster is not zeroed out. As a result, the first cluster of the file will contain random data.

*WORKAROUND:* None.

---

*SPR:* **3601**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* If a seek followed by a write is performed on a dosFs file, and there is not enough room on the disk to allocated the requested space, an error is returned but space temporarily allocated is not freed.

*WORKAROUND:* None.

---

*SPR:* **3602**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* Under some circumstances the dosFs file system reads a cluster from the disk even if there is no valid data for the current file, causing a lack of performance.

*WORKAROUND:* None.

---

*SPR:* **3604**

*ARCH:* All  
*BOARD:* All  
*HOST:* All

*DESCRIPTION:* During initialization of a dosFs disk volume, using *ioctl(FIODISKINIT)*, it is possible for two FAT table images to be allocated in memory at once time. Since each image may be very large, this can artificially reduce memory resources temporarily.

*WORKAROUND:* None.

---

**SPR:** 3613

**ARCH:** All

**BOARD:** All

**HOST:** All

**DESCRIPTION:** When using standard MS-DOS filenames, dosFs does not permit the creation of filenames starting with "." although standard MS-DOS does.

**WORKAROUND:** None.

---

---

**SPR:** 3626

**ARCH:** All

**BOARD:** All

**HOST:** All

**DESCRIPTION:** When renaming dosFs files to a filename which already exists, if the target file is in the same directory as the file being renamed, an error occurs. However, if the target file is in a different directory, the rename succeeds.

**WORKAROUND:** None.

---

---

**SPR:** 3885

**ARCH:** All

**BOARD:** All

**HOST:** All

**DESCRIPTION:** VxWorks BSPs released before 5.2 may fail when trying to build VxWorks boot ROMs in a VxWorks 5.2 tree. Although the v1.0 BSPs are compatible with VxWorks 5.2, the image size of the 5.2 release is larger than the previous releases of VxWorks. The increased size may cause the boot ROM image to be larger than the declared ROM size, so that an error results when building a new ROM image.

**WORKAROUND:** If the ROM size is insufficient to accomodate the boot ROM image, you may either:

- Increase the **ROM\_SIZE** value both in **config/target/config.h** and in **config/target/Makefile.cpuTool**, and then rebuild the boot ROM image. This assumes that the target board supports a larger ROM size; consult the target board manual for more information.

---

- Edit **config/target/config.h** to undefine an optional module, thereby generating a smaller boot ROM image.

Likely candidates for removal include: extra network interface devices, device drivers not used in the booting process, file systems not being used, the shared memory network, tftp client, SCSI, and so on.

*SPR:* **3907**

*ARCH:* Motorola 68060

*BOARD:* mv177

*HOST:* All

*DESCRIPTION:* The macro **INCLUDE\_HW\_FP** should be automatically defined for this architecture in **configAll.h**, by including the MC68060 CPU in the following conditional test:

```
#if (CPU==MC68020 || CPU==MC68040 || CPU==CPU32 || CPU==MC68060)
#define INCLUDE_MC68881 /* MC68881/2 (68040) floating pt coprocessor */
#define INCLUDE_HW_FP /* potential hardware fp support */
...
```

*WORKAROUND:* Insert the following definitions in **config/mv177/config.h**:

```
#define INCLUDE_MC68881
#define INCLUDE_HW_FP
```



**New Routines**

The following table lists new routines in VxWorks 5.2, as well as routines whose calling interface has changed since VxWorks 5.1.1. The marker † indicates previously existing routines that were changed. See the online man pages or the *VxWorks Reference Manual* for detailed reference documentation.

	<b><i>aioShow()</i></b>	show AIO requests
	<b><i>aioSysInit()</i></b>	initialize the AIO system driver
POSIX	<b><i>aio_cancel()</i></b>	cancel an asynchronous I/O request
POSIX	<b><i>aio_error()</i></b>	retrieve error status of asynchronous I/O operation
POSIX	<b><i>aio_fsync()</i></b>	asynchronous file synchronization
POSIX	<b><i>aio_read()</i></b>	initiate an asynchronous read
POSIX	<b><i>aio_return()</i></b>	retrieve return status of asynchronous I/O operation
POSIX	<b><i>aio_suspend()</i></b>	wait for asynchronous I/O request(s)
POSIX	<b><i>aio_write()</i></b>	initiate an asynchronous write
POSIX	† <b><i>clock_getres()</i></b>	return the clock resolution
POSIX	† <b><i>clock_gettime()</i></b>	return the current value for the clock
	† <b><i>clock_setres()</i></b>	set the clock resolution
POSIX	† <b><i>clock_settime()</i></b>	set the clock to a specified value
	<b><i>connectWithTimeout()</i></b>	attempt a connection over a socket for a fixed duration
	† <b><i>fdDevCreate()</i></b>	create a device for a floppy disk
	<b><i>fdRawio()</i></b>	raw floppy-disk I/O access
POSIX	<b><i>ftruncate()</i></b>	truncate a file
	<b><i>ideRawio()</i></b>	raw IDE disk I/O access
POSIX	<b><i>lio_listio()</i></b>	initiate a list of asynchronous I/O requests
	<b><i>lptAutofeed()</i></b>	enable or disable the autofeed mode
	<b><i>lptDevCreate()</i></b>	create a device for an LPT port
	<b><i>lptDrv()</i></b>	initialize the LPT driver
	<b><i>lptShow()</i></b>	show LPT statistics
	<b><i>m2Delete()</i></b>	delete all the MIB-II library groups
	<b><i>m2IcmpDelete()</i></b>	delete all resources used to access the ICMP group
	<b><i>m2IcmpGroupInfoGet()</i></b>	get the MIB-II ICMP-group global variables
	<b><i>m2IcmpInit()</i></b>	initialize MIB-II ICMP-group access

	<b>m2IfDelete()</b>	delete all resources used to access the interface group
	<b>m2IfGroupInfoGet()</b>	get the MIB-II interface-group scalar variables
	<b>m2IfInit()</b>	initialize MIB-II interface-group routines
	<b>m2IfTblEntryGet()</b>	get a MIB-II interface-group table entry
	<b>m2IfTblEntrySet()</b>	set the state of a MIB-II interface entry to UP or DOWN
	<b>m2Init()</b>	initialize the SNMP MIB-2 library
	<b>m2IpAddrTblEntryGet()</b>	get an IP MIB-II address entry
	<b>m2IpAtransTblEntryGet()</b>	get a MIB-II ARP table entry
	<b>m2IpAtransTblEntrySet()</b>	add, modify or delete a MIB-II ARP entry
	<b>m2IpDelete()</b>	delete all resources used to access the IP group
	<b>m2IpGroupInfoGet()</b>	get the MIB-II IP-group scalar variables
	<b>m2IpGroupInfoSet()</b>	set MIB-II IP-group variables to new values
	<b>m2IpInit()</b>	initialize MIB-II IP-group access
	<b>m2IpRouteTblEntryGet()</b>	get a MIB-2 ipRouteTable entry
	<b>m2IpRouteTblEntrySet()</b>	set a MIB-II ipRouteTable entry
	<b>m2SysDelete()</b>	delete resources used to access the MIB-II system group
	<b>m2SysGroupInfoGet()</b>	get system-group MIB-II variables
	<b>m2SysGroupInfoSet()</b>	set system-group MIB-II variables to new values
	<b>m2SysInit()</b>	initialize MIB-II system-group routines
	<b>m2TcpConnEntryGet()</b>	get a MIB-II TCP connection table entry
	<b>m2TcpConnEntrySet()</b>	set a TCP connection to the closed state
	<b>m2TcpDelete()</b>	delete all resources used to access the TCP group
	<b>m2TcpGroupInfoGet()</b>	get MIB-II TCP-group scalar variables
	<b>m2TcpInit()</b>	initialize MIB-II TCP-group access
	<b>m2UdpDelete()</b>	delete all resources used to access the UDP group
	<b>m2UdpGroupInfoGet()</b>	get MIB-II UDP-group scalar variables
	<b>m2UdpInit()</b>	initialize MIB-II UDP-group access
	<b>m2UdpTblEntryGet()</b>	get an UDP MIB-II entry from the UDP list of listeners
	<b>memPartInfoGet()</b>	Return partition statistic information
POSIX	<b>mlock()</b>	lock specified pages into memory
POSIX	<b>mlockall()</b>	lock all pages used by process into memory
	<b>mountdInit()</b>	initialize the mount daemon
	<b>mqPxLibInit()</b>	initialize the POSIX message queue library
	<b>mqPxShowInit()</b>	initialize POSIX message queue show facility
POSIX	<b>mq_close()</b>	close a message queue
POSIX	<b>mq_getattr()</b>	get message queue attributes
POSIX	<b>mq_notify()</b>	notify task that a message is available on a queue
POSIX	<b>mq_open()</b>	open a message queue

POSIX	<b><i>mq_receive()</i></b>	receive a message from a message queue
POSIX	<b><i>mq_send()</i></b>	send a message to a message queue
POSIX	<b><i>mq_setattr()</i></b>	set message queue attributes
POSIX	<b><i>mq_unlink()</i></b>	remove a message queue
POSIX	<b><i>munlock()</i></b>	unlock specified pages
POSIX	<b><i>munlockall()</i></b>	unlock all pages used by process
	<b><i>nfsDevInfoGet()</i></b>	read configuration information from the requested NFS device
	<b><i>nfsDevListGet()</i></b>	look for all the NFS devices in the system
	<b><i>nfsExport()</i></b>	specify a file system to be NFS exported
	<b><i>nfsUnexport()</i></b>	remove a file system from the list of exported file systems
	<b><i>nfsdInit()</i></b>	initialize the NFS server
	<b><i>nfsdStatusGet()</i></b>	get the status of the NFS server
	<b><i>nfsdStatusShow()</i></b>	show the status of the NFS server
	<b><i>ping()</i></b>	test that a remote host is reachable
	<b><i>pingLibInit()</i></b>	initialize the <b><i>ping()</i></b> utility
	<b><i>routeNetAdd()</i></b>	add a route to a destination that is a network
POSIX	<b><i>sched_get_priority_max()</i></b>	get the maximum priority
POSIX	<b><i>sched_get_priority_min()</i></b>	get the minimum priority
POSIX	<b><i>sched_getparam()</i></b>	get the scheduling parameters for a specified task
POSIX	<b><i>sched_getscheduler()</i></b>	get the current scheduling policy
POSIX	<b><i>sched_rr_get_interval()</i></b>	get time slice
POSIX	<b><i>sched_setparam()</i></b>	set a task's priority
POSIX	<b><i>sched_setscheduler()</i></b>	set scheduling policy and scheduling parameters
POSIX	<b><i>sched_yield()</i></b>	relinquish the CPU
	<b><i>semPxlLibInit()</i></b>	initialize POSIX semaphore support
	<b><i>semPxShowInit()</i></b>	initialize POSIX semaphore show facility
POSIX	<b><i>sem_close()</i></b>	close a named semaphore
POSIX	<b><i>sem_destroy()</i></b>	destroy an unnamed semaphore
POSIX	<b><i>sem_getvalue()</i></b>	get the value of a semaphore
POSIX	<b><i>sem_init()</i></b>	initialize an unnamed semaphore
POSIX	<b><i>sem_open()</i></b>	initialize/open a named semaphore
POSIX	<b><i>sem_post()</i></b>	unlock (give) a semaphore
POSIX	<b><i>sem_trywait()</i></b>	non-blocking lock (take) of a semaphore
POSIX	<b><i>sem_unlink()</i></b>	remove a named semaphore
POSIX	<b><i>sem_wait()</i></b>	blocking lock (take) of a semaphore
POSIX	<b><i>sigqueue()</i></b>	send a queued signal to a task
	<b><i>sigqueueInit()</i></b>	initialize the queued signal facilities
POSIX	<b><i>sigtimedwait()</i></b>	wait for a signal

POSIX	<b>sigwaitinfo()</b>	wait for real-time signals
	<b>slattach()</b>	publish the <b>sl</b> network interface and initialize the driver
	<b>slipInit()</b>	initialize a SLIP interface
	<b>sys596ChanAtn()</b>	assert Channel Attention signal to 82596
	<b>sys596Init()</b>	prepare LAN board for 82596 initialization
	<b>sys596IntAck()</b>	acknowledge current interrupt from 82596
	<b>sys596IntDisable()</b>	disable interrupt from 82596
	<b>sys596IntEnable()</b>	enable interrupt from 82596
	<b>sys596Port()</b>	issue "PORT" command to 82596
	<b>sysEnetAddrGet()</b>	retrieve net unit's Ethernet address (Intel EtherExpress 32)
	<b>tcpDebugShow()</b>	display debugging information for TCP protocol
POSIX †	<b>timer_create()</b>	allocate a timer using the specified clock as a timing base
†	<b>usrFdConfig()</b>	configure a FD device
†	<b>vxsys()</b>	make a boot diskette (i386/i486)
	<b>zbufCreate()</b>	create an empty zbuf
	<b>zbufCut()</b>	cut bytes from a zbuf
	<b>zbufDelete()</b>	delete a zbuf and free any associated segments
	<b>zbufDup()</b>	duplicate a zbuf
	<b>zbufExtractCopy()</b>	copy data from a zbuf to a buffer
	<b>zbufInsert()</b>	insert a zbuf into another zbuf
	<b>zbufInsertBuf()</b>	create a zbuf segment from a buffer and insert into a zbuf
	<b>zbufInsertCopy()</b>	copy buffer data into a zbuf
	<b>zbufLength()</b>	determine the length in bytes of a zbuf
	<b>zbufSegData()</b>	determine the location of data in a zbuf segment
	<b>zbufSegFind()</b>	find the zbuf segment containing a specified byte location
	<b>zbufSegLength()</b>	determine the length of a zbuf segment
	<b>zbufSegNext()</b>	get the next segment in a zbuf
	<b>zbufSegPrev()</b>	get the previous segment in a zbuf
	<b>zbufSockBufSend()</b>	create a zbuf and send it as data to a TCP socket
	<b>zbufSockBufSendto()</b>	create a zbuf and send it as a message to a UDP socket
	<b>zbufSockLibInit()</b>	initialize the zbuf socket interface library
	<b>zbufSockRecv()</b>	receive zbuf data from a TCP socket
	<b>zbufSockRecvfrom()</b>	receive a zbuf message from a UDP socket
	<b>zbufSockSend()</b>	send zbuf data to a TCP socket
	<b>zbufSockSendto()</b>	send a zbuf message to a UDP socket
	<b>zbufSplit()</b>	split a zbuf into two separate zbufs



**Corporate Headquarters**

Wind River Systems, Inc.  
1010 Atlantic Avenue  
Alameda, CA 94501-1153  
USA

toll free (US): 800/545-WIND  
telephone: 510/748-4100  
facsimile: 510/814-2010

**Europe**

Wind River Systems S.A.R.L.  
27, Avenue de la Baltique  
Bâtiment B4, LP 739  
Z.A. de Courtaboeuf  
91962 Les Ulis Cédex  
FRANCE

telephone: 33-1-69-07-78-78  
facsimile: 33-1-69-07-08-26

Wind River Systems GmbH  
Freisinger Straße 34  
Postfach 1320  
D-85737 Ismaning  
GERMANY

telephone: 49-89-96-09-49-44  
facsimile: 49-89-96-09-49-40

Wind River Systems UK Ltd  
Aston Science Park  
Aston Triangle  
Birmingham B7 4BJ  
UNITED KINGDOM

telephone: 44-121-359-0981  
facsimile: 44-121-628-1889

Wind River Systems Scandinavia  
Marinens Väg 30  
Box 206  
5-136-23 Haninge  
SWEDEN

telephone: 46-87-07-32-20  
facsimile: 46-87-77-30-49

**Japan**

Wind River Systems Japan/Asia-Pacific  
Pola Ebisu Bldg. 11F  
3-9-19 Higashi  
Shibuya-ku  
Tokyo 150  
JAPAN

telephone: 81-3-5467-5900  
facsimile: 81-3-5467-5877