WIND RIVER

Wind River Linux 3.0

As project resources and time frames shrink, developers must leverage cost-effective, leading-edge technologies to keep up with market requirements. Meanwhile, advances in Linux are creating emerging opportunities to use open source solutions for a variety of projects. Linux is the ideal platform for building embedded devices and is gaining rapid adoption, given the flexibility, innovation, performance, and total cost of ownership advantages of the open source model.

Wind River Linux 3.0 is a commercial-grade Linux solution for embedded device development. The platform contains a fully tested, validated, and supported Linux distribution based on Linux 2.6.27 kernel technology. Wind River Linux offers the optimal combination of integrated and validated open source software with advanced features, optimized for specific target device markets.

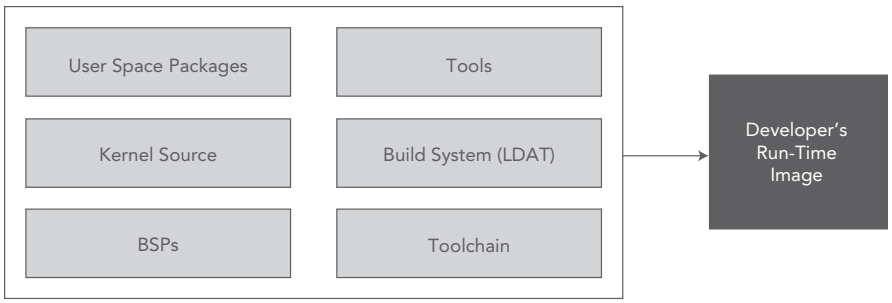

Figure 1: Major components of Wind River Linux

Wind River Linux 3.0 is the latest release to meet the demands of embedded device developers and is delivered in an optimized platform design to address the unique needs of developers of devices for markets such as aerospace and defense, networking, industrial and medical devices, and consumer electronics.

Figure 1 is an overview of the components of Wind River Linux. The product consists of source code and a build system that generates an optimized run-time image suitable for embedded devices. The components of the product are referenced by the developer to create a defined run-time image.

Wind River Linux contains the following components:

- **Application packages:** Hundreds of software packages that operate in protected Linux user mode
- **Kernel source:** The 2.6.27 Linux kernel with many fixes and feature enhancements
- **Board support packages (BSPs):** Hardware enablement components
- **Tools:** Software development tools, including the award winning Eclipse-based Wind River Workbench development suite

Table of Contents

- **Build system:** The Wind River Linux Distribution Assembly Tool, which is used by the developer to compile and assemble these components
- **Toolchain:** The cross compiler based on the GNU Compiler Collection (GCC)

Wind River Linux 3.0 now comes with all the features and hardware support of the previous platform editions in one single product and replaces the previous Linux editions: Wind River General Purpose Platform, Wind River Platform for Consumer Devices, and Wind River Platform for Network Equipment. This provides customers with more flexibility and reduces purchasing and licensing complexity. Wind River Linux 3.0 comes with a set of preconfigured system profiles that make it easy to jump-start device-specific development and allow device manufacturers to bring devices to market easier, faster, and more cost-effectively than ever before. The following are the profiles:

- Consumer premise equipment
- Industrial equipment
- Mobile multimedia device
- Carrier Grade Linux
- Enhanced Carrier Grade Linux
- Limited Platform for Network Equipment

Benefits of the Wind River Linux Solution

The following are some of the competitive advantages of Wind River Linux:

- Commercial product quality with extensive testing and quality assurance (QA), reliable service packs, and security patches with standard software product life cycle support
- Extensive hardware and software ecosystem support
- Lower costs, by eliminating the burden of building, supporting, and maintaining your own Linux distribution, allowing you to focus on differentiating applications rather than on maintaining Linux itself
- Reduced complexity of present and future projects, by leveraging the Wind River Linux cross-build system and layers development methodology
- Rich tools and development-environment support based on the Eclipse framework
- Compliance with industry standards such as Carrier Grade Linux (CGL) to meet market-specific needs
- Delivery of advanced functionality/capabilities with guaranteed real-time performance, advanced networking stacks, and enhanced multicore support
- Predictable roadmap for long-term product planning
- Complete and detailed documentation of software package license information to help ensure compliance with legal and regulatory requirements

The Wind River Linux Distribution Assembly Tool

The Wind River Linux Distribution Assembly Tool (LDAT) is a build system to cross-compile and assemble components for run-time images. LDAT is licensed under the GNU General Public License, version 2.

Typical Use Cases

LDAT commonly addresses the use cases shown in Figure 2.

Project Creation

Users start by creating a project using LDAT. They reference configuration information such as the hardware target, kernel type, profiles, and pointers to other custom software. This creates a project. LDAT uses autoconf to generate a configured build directory. There are a large number of options to select the board, kernel configuration, user space configuration, and so on. The profile option may be used to automatically select the kernel and user space configuration based on the selected BSP and profile combination. The core layers are selected automatically based on the LDAT configuration directory (toolchain version, kernel version, etc.) unless overridden by user arguments.

Additional layers may be included either at the user's request or automatically by other layers. The configure script searches for layers specified without an absolute or relative path.

The configured project directory is itself a layer that can provide modified versions of packages or tools, change configuration information, and include other layers. The project directory can resynchronize these external layers. This allows for the user to include software being actively developed externally.

Development

Users can use tools such as Wind River Workbench to add packages, make changes, debug, and compile software. LDAT creates file systems in a multitude of configurable ways. Nevertheless, it may be the case that in the field, the original equipment manufacturer (OEM), one of the subsequent integrators or developers, or even the final users will want to modify the file system composition. To this end, all the user space components, regardless of their origin, open source trees, source RPMs, or customer trees, are packaged as binary RPMs, and they can be added or deleted or updated at will using the target bound RPM binaries, which also can be included to the file system.

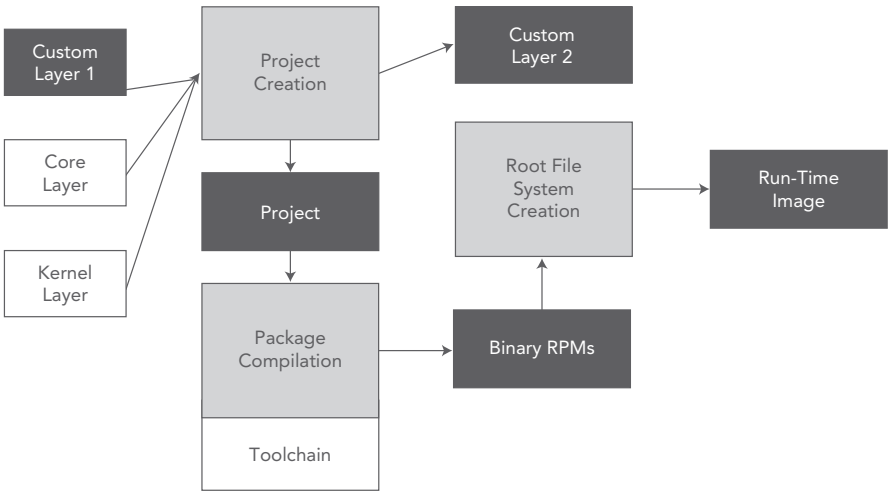

Figure 2: Typical use cases enabled with Wind River Linux

Package Building

Users compile packages from source packages to binary RPMs. This uses the Wind River toolchain. The LDAT cross-build environment uses a simple front-end Makefile fragment and, for SRPM packages, a modified spec file to provide the information required to make the package conform to LDAT and make it cross-buildable. If source code changes to the package are also required, patches are either applied once the package is extracted or during the normal prep stage of the RPM build. Patches (and the spec file) can be provided with the package in the layer and via templates included in the configuration if configuration-specific modifications are required. Most open source packages can be imported with minimal changes and are immediately buildable with LDAT.

Both package formats support a pristine source model. This means the original upstream open source code is provided, plus incremental patches that each implement one major feature. This allows for maximum transparency of source code.

LDAT understands dependencies so that packages are rebuilt when their dependents are changed. To determine whether a package is valid for the current configuration, LDAT uses a checksum value calculated for the current configuration and compares it with the version stored in the RPM. The configuration checksum includes the sources and patches for the package or tool, the LDAT Makefile fragment for the package, any additional configuration files, toolchain flags, and so on, along with the configuration checksums of any other package this package depends on. When the checksum doesn't match, the prebuilt version is ignored and the package will be rebuilt (as will any packages that depend on it).

To build packages, the build system calls the cross-toolchains with minimal performance overhead. The ability to use prebuilt versions of most or all host tools and target packages with checksums to ensure they are valid, along with the ability to update the build configuration with layer changes and to update the target file system without reassembling it from scratch, provides quick turnaround for development builds.

Package building also generates a sysroot to ensure that the right development libraries are being used. This can be exported to other users.

File System Creation

Users create a tuned root file system and run-time image. Once all of the packages are built, the target file system is assembled from the RPMs. If the file system has previously been assembled in this build directory, it can either be rebuilt from scratch or updated in place by uninstalling and reinstalling RPMs that have been changed, added, or removed.

Footprint optimization tools are included to reduce resources.

Exporting a Layer

Users can also create a custom layer for reuse for other development groups or other projects. After the system is built, the *make export-layer* command can be used to generate a layer that incorporates any local changes to the configuration, packages, or tools. The *make install-prebuilt* target can be used to place prebuilt copies of the host tools, target packages, and kernel into either a new or existing layer for use by subsequent builds.

Other Features of LDAT

Pseudo

To allow package builds and file system creation without root privileges, Wind River Linux supplies a tool that provides limited emulation of root access, preserving user ownership, modes, and device nodes correctly for the eventual target file system.

QEMU

Wind River Linux ships with a hardware simulator built in. It simulates ARM, PowerPC, x86, and MIPS hardware. This is integrated tightly with LDAT so that launching a test version in simulated hardware is trivial.

Layers

LDAT uses a hierarchical "layer" structure where each layer may provide anything from a simple configuration setting or a single package to something more complicated such as completely replacing the kernel or augmenting toolchains, host tools, and so on.

The layers provided with Wind River Linux include the "core" layer (called wrll-wrlinux), which supplies the user space packages and core configuration information, the kernel layer for the kernel sources, a host-tools layer for tools that run on the build host, and a toolchain layer that contains the toolchains for cross-development. Add-ons for ecosystem products are typically provided as layers that may augment or modify the default configuration. User-defined layers may provide local changes and prebuilt binaries to speed development.

Contact Wind River for more information.

Server Install

A possible output of LDAT is a bootable DVD image to boot an x86 machine that will then launch an interactive installer. The user can then use this tool to install the generated run-time image on that machine's hard drive.

Toolchain

The Wind River Linux toolchain is a core component of the Wind River Linux build system, providing a number of features specifically suited to embedded development. This section outlines some of the key toolchain features that make embedded development easier and more productive. The toolchain is based on open source components, such as the

GNU Compiler Collection (GCC version 4.3.2), GNU binutils version 2.18.50, and the GNU debugger (gdb version 6.6). Note that toolchain version numbers can be misleading because Wind River makes so many changes to the upstream releases. All these changes are transparent to the developer.

The Wind River Linux toolchain supports ARM, x86, MIPS, PowerPC, and SPARC architectures.

Toolchain Vendor

The Wind River Linux toolchain is a customized distribution of Code Sourcery's Sourcery G++ distribution. Code Sourcery maintains the toolchain, providing support from some of the major contributors to the core toolchain components. Wind River's toolchain has full support for a broad range of target architectures and several host systems. Code Sourcery's toolchain expertise provides confidence that bugs will be fixed quickly and correctly.

C Library

To enhance support for embedded systems in the C library, Wind River has collaborated with other key vendors to found eGlibc, an embedded-oriented distribution of the GNU C library, and continues to sponsor the eGlibc project. The eGlibc project provides superior support for non-x86 architectures such as ARM, PowerPC, and MIPS and has a number of features to make it easier to build with a smaller footprint for target systems. Wind River also has uClibc support, suitable for small footprint targets.

Validation

Code Sourcery runs a large set of toolchain tests, including publicly available tests and proprietary tests. All Wind River Linux toolchain changes are validated, no matter how minor the changes appear to be. Wind River performs additional testing and validation upon receiving a new toolchain. Validation is performed for all supported multilibs for each architecture. This testing procedure ensures that regressions are caught before the compiler ships to a customer.

Custom Features

The Wind River Linux toolchain has a number of features not found in corresponding versions of the upstream compilers and sometimes not available in upstream at all. Vendor-specific updates are included in the toolchain as well as a few features that were added to address specific embedded development concerns. Some such features include architecture- and chip-specific optimizations, support for new opcodes, and improved kernel debugging support in gdb. Finally, all of these features are tested and validated, ensuring reliable behavior even when using custom features.

Multilib Support

The Wind River Linux build system supports simultaneous use of two different CPU types (within the same general family) on a single target; for instance, a target could be configured with both 32-bit and 64-bit binaries, allowing per-package choices of space/speed trade-offs.

Prebuilt Multilibs

The Wind River Linux toolchain provides prebuilt library components, including the C library, for a large variety of multilibs. Each prebuilt library is fully validated as part of the validation process. Prebuilt libraries reduce compilation time while providing extra security and certainty. For each multilib, the toolchain contains configuration files to set up the compiler and other tools to produce compatible code for that multilib. Additional CPU-specific optimizations are available for a broad range of CPUs and configurations.

Toolchain Wrappers

Wind River Linux uses toolchain wrapper scripts that simplify cross development. The project configuration process sets up toolchain wrappers for each CPU type used in the project. These wrappers combine target sysroot configuration and toolchain compilation options to provide seamless building for a target CPU. Users can then use the wrapper program as a substitute for GCC rather than trying to embed needed compiler options in package build processes. The preconfigured CPU templates provide the right combination of options to get good results out of each CPU.

Toolchain Building

The Wind River Linux toolchain is distributed and fully supported as a collection of prebuilt binaries. However, source for every toolchain component is provided, along with configuration data and build scripts to rebuild the toolchain completely from source. Each toolchain component is distributed as a combination of a specific upstream source release or snapshot plus any additional patches provided by Code Sourcery or Wind River. The toolchain build process automates the task of bootstrapping a new cross-compile toolchain, reducing hundreds of commands and configuration steps to a simple "make toolchain."

Toolchain Export

When a toolchain has been configured for a project on a Linux host, it is possible to export that toolchain as an archive that can be used as a toolchain on another host. This can help with distributed development environments.

License Management

The Wind River Linux toolchain provides a license management feature. Customers who want to keep an eye on their compiler usage or who have site license agreements can use the provided license management software, which is fully integrated into the toolchain. If the license management software is not found, the toolchain produces diagnostic messages but runs with full functionality and full performance.

Wind River Linux Development Tools and Wind River Workbench

The Eclipse-based Wind River Workbench development suite offers deep capability throughout the development process in a single integrated environment, with complete platform integration and tools for debugging, code analysis, advanced visualization, root-cause analysis, and test.

Analysis Tools

Workbench and Wind River Linux offer a number of analysis tools available to the developer. Some are enhanced versions of open source tools related to profiling and memory usage, and some are specifically developed by Wind River:

- Performance analysis: The Wind River Workbench Performance Profiler analyzes how a CPU is spending its cycles by providing a detailed function-by-function analysis that shows how individual routines within the processes consume those cycles. This feature is based on the open source tool oprofile, with additional visualization and integration in Workbench.
- Memory analysis: Wind River Workbench Memory Analyzer is a dynamic memory analysis tool that helps prevent and fix such problems as memory leaks, excessive number of memory allocations, and excessive memory allocation sizes. The memory analyzer uses the open source tool mpatrol, with additional visualization in Workbench.
- Boot-time analysis: This uses the ftrace tool to provide lightweight function tracing and includes dynamic ftrace and early-ftrace for boot-time analysis.
- Code execution coverage: The code coverage analyzer feature of Wind River Workbench determines the percentage of source code executed by your software test case and points to the sections of code that have not been fully tested.
- System viewer: Wind River System Viewer supports visualization of multi-core systems; per-core filter and search facilities; the recording of a number of custom events, which use a printf-like format string; graphical and tabular representations of various types of log file analyses, such as CPU usage (aggregate and per core), system load, and per-core ready and running states. System Viewer also supports a host-driven upload method for log files, resulting in log transfer without interference from task CPU use. It also allows for transfer of multiple logs, plus transfer without requiring you to call target functions.

Package Management

Wind River provides several tools to examine the file system's package list, examine package-level dependencies, perform safe package addition or removal based on those dependencies, and perform file-level examination and control of the file system contents:

- Package lists and snapshots: The user space file system is built up from discrete packages, from open source, user source, and virtual packages from custom content. Workbench provides a way to control that package list, to explore different package combinations, and to preserve safe combinations as the user explores the combinations.
- Dependency tracking: Workbench allows the user to visualize the (deep) forward and reverse dependencies and to add or remove packages, knowing that the dependencies are reported and managed.
- Direct package updates to target: Workbench facilitates RPM management on the target as on a regular Linux host. Packages can be developed and compiled and then pushed to the running target for fast turnaround debugging using incremental updates.

Development Tools

Wind River provides several tools to examine and directly control the file system content below the package level. There are also tools to import new open source packages, import new patches, and directly examine a package's patch tree:

- File system layout: The user can directly see the final file system content and directly remove or add files at a fine-grain level below the courser package-level dependencies, allowing direct control of the file system footprint. Wind River also provides tools to discover and visualize which files are touched during a target run.
- Package import tools: Workbench has a feature to handle most of the initial package importation and cross-compilation setup, to help speed up the adoption of new open source packages into a user's project.
- Patch import and export tools: Workbench enables the user to view the patch tree directly and patch files for both source RPM and regular packages. It also provides tools to help import and resolve new patches and to export user changes as new portable patches into a layer directory.
- Export layer: This tool can automatically export many changes made in a project into a new portable Wind River Linux layer. This includes package list changes, file system trimming, kernel configure changes, and new local packages.

Run-Time Features

The following are features of the run-time image that runs on the target.

Kernel Development

The 2.6.27 Linux kernel forms the basis of Wind River Linux's kernel. Wind River adds many features and bug fixes to this kernel, and this specific kernel source configuration is tested and supported.

Kernel Changes

The Wind River kernel adds to the kernel.org 2.6.27 base by importing and validating changes from the following categories:

- mainline: The feature set of the Linux kernel from kernel.org; extended or validated features in particular configurations and applications, by Wind River
- external: Features imported from another external source and merged into the Wind River kernel
- internal: Features that are in layers or merged into the kernel, developed by Wind River
- fixes: Bug fixes for drivers; features in mainline or external projects

These features are tested individually, merged, and then tested as a complete system. This includes stress and use case testing and ensures that the features are stable individually, integrate with Wind River tools, and form a solid base for deployment tuning.

Kernel Presentation

The Wind River kernel is presented to the developer via a fully patched, history-clean Git repository (see http://git-scm.com/ for more information on Git). This stores the selected features, board support, and configurations extensively tested by Wind River. Presenting the Wind River kernel in this manner allows the end user to leverage community best practices to seamlessly manage the development, build, and debug cycles.

The build system generates a flat tree from this Git tree that contains the specific features required for the target kernel's use. Storing the source code in Git enables users to more easily understand what changes have been made to the kernel and why. Wind River uses a combination of tags and branches to assist in delineating between the various added features.

The workflow of the Wind River kernel follows the recognized community best practices. In particular, the kernel as shipped with the product should be considered an "upstream source" and viewed as a series of historical/documented modifications (commits) to the kernel. These modifications to the kernel represent the development and stabilization done by the Wind River kernel development teams.

Contact Wind River for more information about kernel development workflow with Wind River Linux.

Browsing Changes

Wind River Linux's kernel development methodology simplifies the following use cases for browsing and understanding kernel code:

- Showing changes, e.g., "What changes were made to foo.c?"
- Showing foo.diff
- Showing groups of changes, e.g., "Show me only the LTTng patches."
- Comparing branches, e.g., "What's different between the ixm27 and imx31 BSPs?"
- Completing annotation for all changes, e.g., "Where did feature X come from and why is it there?"
- Showing standard commit IDs, e.g., "I see a kernel change on another tree; is this included in my tree?"

Kernel Features

The kernel forms the basis of many features in Wind River Linux's run-time.

Kernel Styles

The code base of the Wind River Linux kernel supports many features that are available for specific applications but not necessarily suitable for all. Wind River provides predefined kernel styles that are specific to these applications. The kernel styles shipped with the product are described here:

- standard: The standard kernel represents a common feature and technology base for all other kernel types. It is Wind River's goal to include all possible features in this single kernel type, making it suitable for many applications and a jumping-off point for more specific kernel implementations. When it isn't possible for a feature to coexist at either compile time or run-time, it is merged into a specific kernel type.
- small: The small kernel is built on top of the standard kernel and is targeted for smaller devices, where functionality such as boot time, memory footprint, or power management may be of primary concern. As a result, it is configured with fewer features, includes boot-time features, and is normally coupled with smaller root file systems.
- cgl: This kernel is targeted at networking equipment. This kernel includes upgrades such as shelf management, security, fault tolerance, threaded interrupt request lines (IRQs), and crash analysis that are not available in the other kernel types. This kernel is a suitable jumping-off point for high availability solutions.
- ecgl: This is a superset of CGL and contains additional features and configuration for specific networking situations.
- preempt_rt: The preempt_rt or "real-time preemption" kernel implements real-time capabilities for specific hardware.
- Real-time core: The rtcore kernel type includes the proprietary real-time core patches and creates a "guaranteed real-time" kernel with a defined programming interface. Wind River Real-Time Core for Linux is an add-on product, described later in this document.

Networking Subsystem

The following are specific networking subsystem features:

- TCP/IP v4
- IPv6, MIPv6
- IPsec
- Stream Control Transmission Protocol (SCTP)
- VLAN tagging
- Transparent Interprocess Communication (TIPC)
- Network block device (NBD)
- cgroups and controllers: Control group support adds support for grouping sets of processes together, for use with process control subsystems such as Cpusets, CFS, memory controls, and device isolation. It includes net traffic controller, memrlimit controller, dm-ioband bio_tracking, and group scheduling controllers.
- Other RFCs (contact Wind River for details)

Security

The following are specific kernel security features:

- BSD jail (bsdjail): The Linux port of the FreeBSD "jail" facility provides the ability to partition the operating system environment while maintaining the simplicity of the UNIX chroot model.
- Simplified Mandatory Access Control Kernel Support (SMACK): This lightweight implementation of name-based security labels is useful for providing Mandatory Access Control (MAC) without a full SELinux policy.

Debugging and Profiling

The following are specific debugging features:

- oprofile: Kernel.org oprofile enhanced with tracing through the syscall boundary
- ftrace: Lightweight function tracing, includes dynamic ftrace (backport from 2.6.29) and early-ftrace for boot-time measurement enhancements
- ptrace: Process trace, single step, multi-threaded trace support
- kprobes: Kernel address trapping
- KGDB: Kernel debug support over serial, Ethernet, and console
- lockdep: Lock dependency checking and analysis
- wrnote: ELF image annotation for core dumb debug

- On-chip debugging
- Linux Trace Toolkit (LTTng): Extensible, lightweight kernel instrumentation for tracing program execution and debugging parallel and real-time behavior
- latency top: Latency visualization support
- Boot-time reduction: Enhancements for measuring and streamlining boot time
- Footprint reduction: Kernel configuration and modifications to limit the runtime kernel footprint
- kmemcheck: Kernel memory checking and leak detection

File Systems

The following are kernel file system features:

- Boot technologies: ramdisk, execute in place (XIP), kernel libc support for boot environments (klibc), initial ramfs support (initramfs), fastboot (asynchronous boot/init)
- Flash file systems: yaffs, yaffs2, jffs, advanced XIP file system (axfs)
- Logical Volume Manager (LVM and LVM2)
- RAID
- Network file systems: NFS, smb
- Disk file systems: ext2, ext3, FAT, VFAT
- Other file systems: Stackable unification file system (unionfs), file system for large device scalability (logfs), compressed read-only file system (squashfs), compressed ROM file system (cramfs)
- Revoke: revokeat() system call for inode-based revocation

Input and Output

The following are input and output (IO) features of the kernel:

- I/O splice: A system call that copies data between a file descriptor and a pipe, or between a pipe and user space, without actually copying the data
- User space I/O: Drivers that allow programs easy access to kernel interrupts and memory locations; used for user mode drivers
- Asynchronous I/O

Real-Time and Deterministic Scheduling Behavior

The following are specific real-time and deterministic scheduling behavior features:

- Wind River Real-Time Core for Linux: See "Optional Add-on Products" on page 9.
- preempt_rt: Complete preemption (real-time) reduces the scheduling latency of the kernel by replacing almost every spinlock used by the kernel with preemptible mutexes and thus making all but the most critical kernel code involuntarily preemptible
- Voluntary kernel preemption (desktop): Reduces kernel latency by adding more explicit preemption points to kernel code
- No forced preemption (none): Traditional Linux preemption model
- Robust priority inheritance mutex: Robust and priority inheritance support for user space mutexes
- High resolution timers (HRT)
- Dynamic tick support (NOHZ): Timer interrupts will only trigger on an as-needed basis both when the system is busy and when the system is idle

Hardware Support

The following are highlights of hardware features:

- Multiarchitecture: Five architectures (MIPS, ARM, SPARC, x86, PowerPC) and eight subarchitectures (MIPS, MIPS64, ARM, SPARC64, x86, x86-64, PowerPC, PPC64)
- SMP/AMP/multicore: SMP safety of drivers and core kernel functionality
- CPU isolation (cpuisol)
- CPU hotplug
- IEEE float: IEEE floating point conformance for PowerPC processors supporting Signal Processing Extensions (SPE)
- SEC (Talitos Freescale Security Engine): Hardware acceleration for PowerQuicc E processors
- Peripherals: Device drivers for peripherals such as audio, Ethernet, GPIO, SDIO, SCSI, MTD, serial, framebuffer, VGA (graphics), keyboard, USB (gadget, host, OTG), touch screen, PATA/SATA, sound, PMEM, wireless (Wi-Fi), Bluetooth, MTP

Other Features

- kexec: A system call that provides the ability to shut down the current kernel and start another without rebooting hardware
- kdump: Kernel crash dump
- Kernel virtual machine (KVM): OS virtualization for certain architectures
- Clock API: Wind River interface for manipulating clock sources and data sampling

Networking Features

These highlights describe features relevant to network equipment. They may overlap with the kernel features described previously.

System Black Box

Taking cues from the aviation industry, Wind River Linux's persistent memory framework (PMEM) provides a system black box acting much like the combined flight data recorder (FDR) and cockpit voice recorder (CVR). Scheduler decision history, logs of all exceptions, panic and console logs, kernel log messages, system reset and reboot logging, Linux Trace Toolkit (a set of kernel patches and supporting user space tools to control tracing) logs, even end-user defined events can all be logged to dedicated nonvolatile memory, external memory, peripherals, or even protected segments of normal system RAM. This enables faster recovery and better system uptimes by allowing all necessary debug information to be preserved by the PMEM driver in these protected regions of memory for later analysis while allowing the system to reboot and re-enter service immediately.

Transparent Interprocess Communication Protocol

As a major contributor and one of the maintainers of the TIPC project, Wind River actively develops this cross-platform, high-speed communications technology aimed specifically at clustered computing environments. TIPC is a communications protocol that provides developers with an extremely flexible means of creating distributed, cooperative applications that may migrate as required throughout the cluster seamlessly. Wind River continues to invest in TIPC, and Wind River Linux remains up-to-date with developments in the TIPC project.

Security

Originally developed by the National Security Agency (NSA), SELinux remains the premier method of ensuring a flexible and trusted computing environment. SELinux is both a Linux Security Module (LSM)—a piece of the kernel that arbitrates access to all system resources based on the system policy—and a collection of supporting user space tools for

developing, applying, enforcing, auditing, and debugging the security policy used by the LSM. Wind River Linux includes three levels of security out of the box for SELinux-enabled configurations and all the tools necessary to customize or develop new policies from the ground up.

Additionally, Wind River Linux includes advanced, preemptive security technologies such as run-time stack and buffer overflow protection and a suite of tools that together provide a complete intrusion detection and prevention system. The Wind River Linux kernel also includes the PaX patch set implementing least-privilege protections for memory pages or memory segments as well as GRSecurity, further kernel patches that build upon PaX and implement a trusted execution model, role-based access control, detailed system accounting logs, and fine-grained privilege separation.

Further user containment features that resist all known chroot-jail attacks protect your system even when deployed in a hostile environment or with an unknown user base.

Carrier Grade and Network Equipment Provider Ready

The first Linux distribution registered as fully compliant with the Carrier Grade Linux 4.0 specification, Wind River Linux 3.0 offers expanded support of CGL features including better support for clustered computing environments and better support for developing and deploying highly available systems.

Wind River Linux also meets the SCOPE Alliance Linux profile and addresses key SCOPE Alliance gaps. These include persistent shared memory with the system black box, coherent user and kernel tracing framework with LTTng, run-time analysis tools, and common command-line tools such as strace and ltrace for doing system call and library tracing. The following are additional SCOPE Alliance gaps that are addressed by Wind River Linux:

- Coarse resource enforcement: Wind River Linux's group scheduling and resource controllers allow memory and scheduling limits to be enforced on a group basis rather than simple per-process or per-object.
- Layer 2 Tunneling Protocol (L2TP) support
- File access tracing: Linux kernel features such as inotify as well as Wind River Linux features such as GRSecurity provide extensive logging and notification options for monitoring file access and recording system events.

With support for both standalone and clustered systems as well as the PICMG Advanced Telecommunications Computing Architecture (ATCA) specification, Wind River Linux provides highly available solutions for devices at both the network core and edge.

Advanced Network Equipment Features

With Wind River–developed proven technologies such as the virtual management controller, it becomes easy to develop complex applications that can run on a variety of hardware using common intelligent platform management interface (IPMI) commands for health monitoring even on systems without a baseboard management controller (BMC). The addition of the application monitoring and migration feature (memmon) allows complex applications to transparently migrate between systems to ensure zero service interruption even in the case of a scheduled outage. Additional system engineering tools such as flexible out of memory (OOM) killer behavior, Ethernet link bonding, and statistics gathering and reporting on a per-socket and per-interface basis allow designers to engineer their systems to the absolute maximum capabilities of the hardware. This is combined with Wind River's support for error detection and correction (EDAC) on new chipsets.

Network-Based Storage Solutions

Integrating technologies such as the distributed replicated block device (DRBD), multiple redundant communication paths to external storage over fiber channel links, ATA over Ethernet, the Oracle Cluster File System version 2 (OCFS2), and Internet Small Computer System Interface (iSCSI), Wind River Linux provides functionality for centralized logging servers, centralized billing and accounting servers, and share file system servers.

Hardware Support: Board Support Packages

Wind River Linux board support packages (BSPs) are hardware-enablement components that contain elements such as drivers and settings needed to make Wind River Linux support specific hardware.

BSPs are separable configuration components that can be created and added to Wind River Linux at any time. In addition to the BSPs Wind River Linux ships with, Wind River continues to add boards according to customer demand and hardware availability. Such additional BSPs are available via Wind River's Online Support website to customers under an active platform subscription. Also, Wind River Services can create customer-specific BSPs for hardware that is not covered by existing ones.

A typical BSP includes board-specific configuration files that overwrite or add configuration options defined by the common platform templates. Additional kernel patches included in the BSP can add new device drivers or apply necessary changes to existing Linux code. BSPs can also contain additional user space components or other files.

Wind River has validated proper operation of the Linux run-time for each supported reference board. The supported features are board-specific and depend on availability and maturity of the code in the open source community.

The product ships with more than 40 BSPs covering the following target processors:

- ARM
- Intel x86
- MIPS
- PowerPC
- UltraSPARC T2

BSPs are also created and shipped asynchronously, after the product is released. For this reason, the list of BSPs is not static. Contact Wind River to get an up-to-date supported BSP list with detailed descriptions of supported peripherals.

Applications

Wind River Linux provides more than 500 integrated user space application packages. They implement functionality typical of an embedded Linux run-time.

The Wind River build system (LDAT) generates binary RPMs from these sources. LDAT can then use these to generate a root file system.

Origins and Porting

A variety of open source projects forms the origins of the user space code base. About 100 packages are based on traditionally prepackaged trees containing source code, configuration scripts, and Makefiles or Makefiles precursors (i.e., a classic package format). The remaining 400 have source RPMs as their base.

Wind River patches these upstream sources for integration and bug fixing. These packages generally contain the following types of patches:

- Cross compilation: Many packages are expected to be compiled on x86 architectures for x86 architectures. This often means host libraries can be referenced or linked in.
- Multilib: This ensures that packages can be built for both 32- and 64-bit targets.
- Other defects: Wind River ensures that packages are properly integrated together.

The Wind River Linux Distribution Assembly Tool, LDAT, will access the ported components, pass the appropriate cross-compilation parameters, and create a file system matching the target's architecture and the kernel's configuration and including the features needed.

Customers can add their own user space packages using the same mechanism. Instructions for this are included in product documentation.

Lists of Packages

It is easiest to consider the package list in terms of categories used. The following is a list:

- Administration
- Basic C support
- Booting and startup
- Daemons
- Databases
- Debugging
- Devices
- DirectFB
- File systems
- File transforms
- Graphics
- Hardware
- Kernel
- Languages
- Middleware
- Multimedia
- Networking
- Shells and scripting
- Security
- SELinux
- Setup
- Sound
- System
- Host tools
- Test
- Utilities
- Various
- Wind River instrumentation
- X Server

A full list of the package names can be found at the end of this document in "Appendix A: Package Summary by Category." For a full list of package details (source package names, binary package names, versions, licenses, etc.), contact Wind River.

Profiles

Wind River also includes several profiles that define preassembled root file systems and kernels for specific functionality. The following are the profiles shipped:

- Consumer Premise Equipment
- Industrial Equipment
- Mobile Multimedia Device
- CGL, Carrier Grade Linux
- ECGL, Enhanced Carrier Grade Linux
- Limited Platform for Network Equipment

Optional Add-on Products

Wind River provides other products to implement functionality not available in the base product.

Wind River Real-Time Core for Linux

Wind River Real-Time Core for Linux and Wind River Linux provide device manufacturers with a mature, proven technology for developing complex, next-generation, Linux-based applications that require guaranteed hard real-time with microsecond-level interrupt and scheduling latency. Wind River Real-Time Core for Linux employs a simple real-time executive that runs the non-real-time Linux kernel as its lowest priority task and routes interrupts to the Linux kernel scheduler through a virtual interrupt layer.

All interrupts are initially handled by Wind River Real-Time Core and are passed to the standard Linux kernel only when there are no real-time tasks to run. Real-time applications are implemented as standard POSIX threads and receive interrupts immediately, resulting in near hardware-threshold speeds for interrupt processing. Wind River Real-Time Core and user space tasks communicate through lock-free queues and shared memory. From the application programmer's point of view, real-time queues look very much like standard Linux character devices, accessed via POSIX read/write/open/ioctl system calls. Shared memory is currently accessed via the POSIX mmap calls.

Additionally Wind River Real-Time Core for Linux includes the user space real-time module that enables programmers to add a memory protection layer to their real-time threads. This protection adds a level of insurance that no real-time thread will bring the entire system down should one ever fail. The user space real-time module also adds a greater programming language support to Wind River Real-Time Core for Linux for those developers who prefer to program in C++ or FORTRAN.

Also included as part of Wind River Real-Time Core for Linux is a real-time networking module based on Wind River Advanced Networking Technologies. This real-time networking module ensures that your network applications send and respond to network packets in a fully deterministic fashion. A brief list of network protocols supported includes TCP, UDP, virtual routing, VLAN, and telnet.

Wind River Real-Time Core benefits include the following:

- Guaranteed hard real-time response times regardless of system load
- Wind River Real-Time Core applications implemented as standard POSIX threads
- Largely untouched Linux kernel, maintaining standard interfaces that do not interfere with kernel, silicon vendor, community, or user developed patches
- Real-time thread memory protection utilizing user space real-time
- Fully deterministic real-time networking support

IPL Cantata++ for Wind River Workbench

IPL Cantata++ for Wind River Workbench (formerly Unit Tester), now available for Wind River Linux, is a set of tools that allows developers greater efficiency in completing unit testing, integration testing, and code coverage analysis on the tests. The integration of Cantata++ with the Wind River Workbench development suite places these capabilities within easy reach. Cantata++ increases software quality, decreases time-to-market, and reduces support costs through better, faster, more automated testing in the development life cycle.

Wind River Workbench On-Chip Debugging

In the early stages of hardware and software development, a robust connection to the microprocessor through its run-control port is essential. Wind River Workbench provides connectivity between the host development environment and the target device via the JTAG or on-chip debugging interface of the microprocessor that resides on the device.

The on-chip debugging interface of most microprocessors enables full control of the microprocessor itself, access to core and peripheral registers, and access to on-chip switch fabrics and memory controllers, along with access to external buses and many devices attached directly to the bus. In addition, some microprocessors support either internal or external trace buffers, allowing developers to capture information regarding the exact code that ran on the target and when.

On-chip debugging provides developers with complete system-level control of their environments at all times, enabling more efficient and effective hardware bring-up, firmware development, and device driver and BSP generation. Specifically for Linux development, Workbench On-Chip Debugging provides visibility into hardware and software interactions for kernel and kernel modules and enables development and debug of user space applications. The JTAG-based debug capability is a useful alternative to agent-based debugging in applications where serial, Ethernet, or USB interfaces are not available or in environments where agent instrumentation of the operating system is not desired.

The Wind River Debugger provided with Wind River platforms can be enabled for on-chip debugging. This capability, along with Wind River ICE, Wind River Trace, and Wind River Probe hardware, provides access to significant additional capability within Workbench.

For more information, visit http://www.windriver.com/products/workbench.

Wind River Test Management

Wind River Test Management is a scalable system that links device development and test teams with a collaborative suite of applications for efficient system testing and defect resolution. The system leverages a unique, dynamic instrumentation technology to measure code coverage, profile performance, and diagnose and repair the system at run-time. The product is designed to manage multiple devices under test at multiple lab locations, maximizing resource utilization and accelerating the testing process.

Benefits of Wind River Test Management 3.1 include the following:

- Higher quality, faster time-to-market, lower cost
- More testing, more often
- Faster defect resolution
- Management of progress, quality, and resources
- Benefit to both development and QA
- Powerful sensorpoint technology
- Open, scalable architecture
- Broad platform support

For more information, visit http://www.windriver.com/products/test_management/.

Wind River Network Management

Wind River Network Management includes advanced SNMP, CLI, and web-based management interface development tools. The Network Management products are designed and implemented as cross-platform and validated on both VxWorks and Wind River Linux. This makes it easy for developers to implement management interfaces for both VxWorks and Wind River Linux–based devices or to migrate from VxWorks to Wind River Linux.

The Wind River Network Management SDK includes a standards-based implementation of SNMP, consisting of SNMP v1/v2c/v3 and AgentX support as well as a scalable, unified, small-footprint management framework to create web-based, CLI-based, or custom management interfaces to manage networked elements. The scalable framework consists of a management backplane that acts as a conduit for data handling between management interfaces (consumers) and manageable elements (producers); it can have any type of consumer and any type of producer.

Wind River Network Management SDK 3.2 includes the following:

- Wind River SNMP 10.3
- Wind River CLI 4.7.1, Wind River Web Server 4.7.1, and Wind River MIBway 4.7.1
- Web and CLI-based network management interfaces
- Standalone web server: HTTP and HTTPS
- Integration with SNMP via MIBway
- Wind River Management Integration Tool (Windows host support only)
- Management Configuration Editor

Wind River SNMP

The Simple Network Management Protocol (SNMP) is designed to facilitate management and configuration of networked devices. Wind River SNMP is a highly portable, memory-efficient, and standards-compliant implementation of SNMP specifically designed for original equipment manufacturers (OEMs) and system integrators who require full compliance with SNMP standards in a fast, small SNMP agent. This complete solution for integrated SNMP design and implementation includes a full MIB development platform. It is composed of SNMP v1/v2c/v3 and AgentX.

Features of Wind River SNMP 10.3 include the following:

- Bilingual SNMP agent supports SNMPv1/v2c protocols
- Asynchronous support
- SNMPv3 security features
- SNMP notifications
- "Target" and "notify" MIBs
- SNMP proxy
- SNMP v1/v2/v3 coexistence
- AgentX module

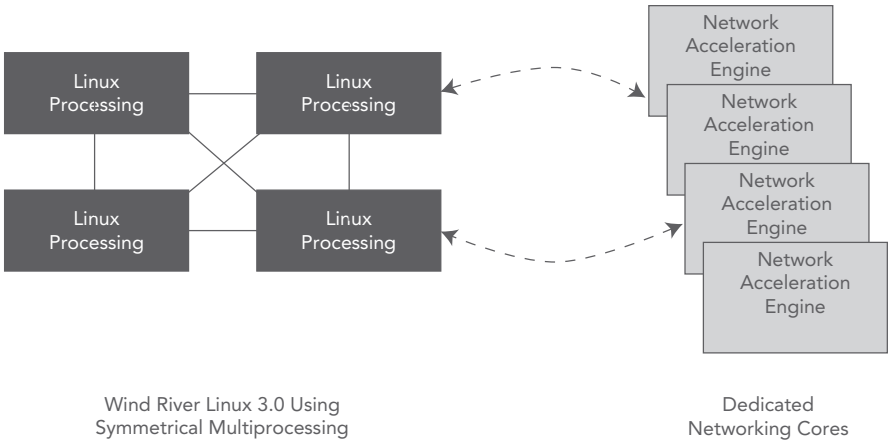

Figure 3: Network acceleration with Wind River Linux 3.0

- MIB compiler
- Compact, interoperable, standards-based configuration
- Integration and validation with Wind River Advanced Networking Technologies (Interpeak-based)
- Portable design and implementation
- A new API to support SNMPv3 INFORM PDU operations

Wind River Advanced Networking Technologies

Wind River networking add-on products are delivered in ANSI-compliant C source code, with ready-to-run examples on Wind River Linux. The optional Wind River Advanced Networking Technologies addresses networking, security, and mobility requirements in markets such as wireless infrastructure, network infrastructure, and consumer devices. In particular, IPv6, routing, wireless, and mobility technologies enable device and equipment manufacturers to deliver products to market quickly while offering the latest networking and security capabilities on Wind River Linux.

Multicore silicon has brought unprecedented cost, power, and space efficiencies to networking equipment. Advanced Networking Technologies 6.7 for Wind River Linux is designed to optimize multicore network throughput by utilizing individual cores to process networking tasks. Candidates for fast path processing include IPv4/IPv6 forwarding, IPsec, virtual routing, VLAN tagging, NAT, tunneling, and access control list filtering. By using dedicated processing cores for these tasks, operating system overhead can be greatly reduced. This yields gigabit wire speed throughput for packet forwarding and frees up more processing resources for Linux kernel tasks and applications.

The following diagram illustrates the division of labor for network processing using Advanced Networking Technologies 6.7.

The number of network acceleration engines (NAEs) is specified at build time based on system requirements. Wind River Linux 3.0 initializes these cores, reserves them as NAEs, and does not schedule any other tasks to run on them. Linux then "gets out of the way" and lets the NAEs run independently to process network traffic. Packets in the fast path are processed using a run to completion model and do not touch the Linux kernel. Any packets that are not processed on the NAE, including locally destined traffic, are passed from the NAE to the Linux kernel. This is defined as "normal path" processing.

Testing and Validation

Wind River is committed to providing quality products for both proprietary and open-source-based technologies. Our quality policies include formal code inspections, peer reviews, project reviews, program audits, and traceable requirements change management. Wind River Linux was created following a methodical process to thoroughly test key features on every supported reference configuration (defined by development host, kernel and package configurations, and supported board).

Wind River has developed a robust, scalable, and automated build and test infrastructure with more than 4,000 test cases and 301,336 test runs. This infrastructure supports many processor architectures and uses a combination of commercial, open source, and proprietary tests, including LTP Core, LTP Network, LSB, TAHI, and Open POSIX. Wind River uses coverage tools, such as gcov and lcov, to optimize test development and close gaps in existing test suites.

Automated testing packages for Wind River Linux 3.0 include the following:

Test Suite	Description
Automated Boot Login Test	This tests the booting process of any target architecture for a given kernel and rootfs. The process is completely automated for a set of targets, which helps in determining the boot sanity of the target.
CD Sanity Test	This automation suite covers CD installation on a new release, followed by building the rootfs for various target combinations using prebuilt RPMs. It boots the target with the prebuilt kernel and rootfs and executes KGDB and user-mode tests on the target, then reports the results to the database.
Linux Test Project (LTP)	This test suite validates the reliability, robustness, and stability of Linux kernel and its network components.
Open HPI	This is the Open Hardware Platform Interface (Open HPI) conformance test.
Open POSIX	This test suite is for POSIX 2001 APIs not tied to specific implementations. It provides conformance, functional, and stress testing, with an initial focus on threads, clocks and timers, signals, message queues, and semaphores.
Perl_test	This tests the Perl package.
RT Feature Testing: LMBENCH Realfeel	Real-Time Feature Testing tests performance. LMBENCH is used to measure I/O of the kernel. Realfeel tests scheduler behavior.
Saftest	This tests the Open HPI package.

In addition to automated testing, significant manual testing—including feature testing, Workbench testing, and complete system testing—for Wind River Linux 3.0 has been completed.

Test Suite	Description
3.0 Bug Fix Testing	The bugs fixed during various release cycles were tested for the fixes, then closed.
3.0 New Feature Testing	• Kernel test: Kernel Feature Test Package (kftp), feature matrix tests • User space test: User space packages sanity test—442 user space packages sanity tests; user space packages functionality test: 151 packages' test script • LDAT test: Build system functionality test • Toolchain test: Toolchain sanity test and new feature test • Nmap: Network exploration tool and security scanner • Netperf: Networking performance test tool • CGL compliance testing: More than 120 P1 requirements tested • Kexec and core-dump, server-based install: CGL features tested • TAHI IPv6 conformance tested • ecgl-tools: ecgl kernel test tools, provided by Nortel • LSB: Linux Standard Base Compliance test packages on sun_cp3020 – cgl -glibc_cgl • iozone: File system performance test tool • Nbnch-bite: CPU calculation performance • Footprint test: Static kernel footprint—kernel image • Static rootfs footprint: rootfs size • Dynamic footprint: Run-time system footprint • Boot-time test: Measurement of PCD board boot-up time; implemented by a tool provided in Workbench • Coverity: Static code analysis software
Board-Specific Testing	Along with new BSPs added for release, there are board-specific tests mentioned in the board readme. These tests were executed as part of regression testing on Wind River Linux 3.0.
Documentation Testing	Documentation for Wind River Linux 2.0 was tested to make sure all steps are properly recorded.
Host OS Testing	Installation testing was done on various host OSes supported for Wind River Linux 3.0 as well as sample application build and debugging from Workbench and platform build.
HRT Regression Testing	HRT features for previous releases were tested for regression on the supported platforms.
Install Bundle Testing	Installation testing was done for various product structure bundles.
Nessus Report	A Nessus report for Intel sun_cp3220 and fsl_8548cds (glibc_cgl, cgl) was done.
Pre-PRT Testing	Some user scenarios for Workbench, Wind River Run-Time Analysis Tools, and the build system were tested on supported hosts and platforms, as done by the PRT.
Regression Testing Async BSPs	We tested the Async BSPs (IMX31) for regression (Workbench and other platform testing).
Regression Testing Sync BSPs	We tested the regular BSPs for regression (Workbench and other platform testing).
RT Regression Testing	Regression testing features for previous releases were tested for regression on the supported platforms.
SNMP Testing	SNMP tests were performed.
Stress Testing	Stress tests were performed.
Usability Testing	Wind River Linux 3.0 and Workbench 3.1 usability testing is based on the usability testing document.
Use Case Testing	The use cases for Workbench, run-time analysis tools, and build system were tested on supported hosts and platforms.
Workbench Integration Testing	Wind River tests the feature integration of Workbench with Wind Manage, System Viewer, and run-time analysis tools.

Partner Ecosystem

Wind River's world-class partner ecosystem ensures tight integration between our core technologies and those of the premier hardware and software companies we've chosen to build out our solutions. Our partners help extend the capabilities of Wind River Linux by offering out-of-the-box integration and support for key technologies in a number of fast-moving markets. Our team is trained to troubleshoot partner technologies in use with Wind River products, making ours the best-supported ecosystem in the Device Software Optimization (DSO) industry.

The Wind River Partner Ecosystem is constantly expanding. Contact us for more details or visit http://www.windriver.com/partners/.

Alignment with the Open Source Community

Wind River has a long history of working with and contributing to the open source community. We were one of the early adopters of the GNU compiler and debugging technologies for use with our VxWorks real-time operating system and Tornado cross-development environment, and we contributed bug fixes and improvements to these projects.

We continue our commitment to enable our customers to successfully leverage open source in their development of next-generation devices through our active involvement in a number of organizations.

Carrier Grade Linux

The Linux Foundation CGL working group is developing the Carrier Grade Linux specification and driving the adoption of Linux in the network infrastructure industry.

Consumer Electronics Linux Forum

CELF drives adoption of Linux in the consumer electronics industry.

Eclipse Forum

The Eclipse Forum works to improve and enhance the core Eclipse framework and also enables Wind River Workbench to be integrated with a wide range of complementary Eclipse plug-ins for software development.

LiMo Foundation

The LiMo Foundation was established to develop the Foundation Platform, a Linux-based, open, mobile communication device software platform. The Foundation's purpose is to advance the creation, evolution, promotion, and support of the Foundation Platform and to cultivate an ecosystem of complementary products, capabilities, and services.

Mobile Linux Initiative

The Linux Foundation Mobile Linux workgroup is focused on accelerating the adoption of Linux in the rapidly growing mobile device market.

Open Handset Alliance

The Open Handset Alliance is a group of mobile and technology leaders who share the vision for building a better mobile phone and are committed to changing the mobile experience for consumers through the efforts of an open mobile ecosystem.

SCOPE Alliance

SCOPE is an industry alliance committed to accelerating the deployment of carrier-grade base platforms for service provider applications.

Service Availability Forum

The SAF is an organization developing APIs and specifications for high availability of hardware and software applications. Its specifications include the Hardware Platform Interface (HPI) specification and the Application Interface Specification (AIS).

OpenSAF Foundation

OpenSAF is an open source project established to develop a base platform middleware consistent with Service Availability Forum (SA Forum) specifications, under the LGPLv2.1 license. The OpenSAF Foundation was established by leading communications and enterprise computing companies to facilitate the OpenSAF Project and to accelerate the adoption of the OpenSAF code base in commercial products.

TIPC Project

The TIPC Project provides an open solution for communication and messaging.

Professional Services

Wind River Professional Services, a CMMI Level 3–certified organization, enables you to reduce risk and focus on development activities that add value and differentiate design. As part of our comprehensive solutions, Wind River offers a Linux Services Practice, with focused offerings that help you meet strict market deadlines while keeping development costs down. Our experienced team delivers device software expertise that solves key development challenges and directly contributes to your company's success. Backed by our commercial-grade project methodology, Wind River Professional Services include device design, Linux BSP and driver optimization, software system and middleware integration, and legacy application and infrastructure migration.

Education Services

Education is fundamentally connected not only to individual performance but also to the success of a project or an entire company. Lack of product knowledge can translate into longer development schedules, poor quality, and higher costs. The ability to learn—and to convert that learning into improved performance—creates extraordinary value for individuals, teams, and organizations. To help your team achieve that result, Wind River offers flexible approaches to delivering product education that best fit your time, budget, and skills development requirements.

Personalized Learning Program

Wind River offers a unique solution to minimize the short-term productivity drop associated with the process of adopting new device software technology and to optimize the long-term return on investment in a new device software platform. The Wind River Personalized Learning Program delivers the right education required by individual learners to accomplish their jobs. The program identifies work-related skill gaps, generates development plans, materials, and learning events to address these skill gaps, and quantifies the impact of the development activities for each individual user.

This programmatic, focused, and project-friendly approach to skills development results in a significant increase in the personal productivity of your teams, improved efficiency in the processes they employ, and faster adoption of the technology you have purchased. Personalized Learning Programs deliver improved business performance—customers have reported a return on investment ranging from 18 percent to 80 percent over a traditional training approach. Consult your local Wind River sales representative for more information on Personalized Learning Programs.

Support Services

Wind River Customer Support, a certified Service Capability and Performance (SCP) organization, provides support for Wind River Linux platforms. Your subscription to Wind River Linux includes full maintenance and support, delivered through Wind River's Online Support website and our worldwide support team. Wind River Support includes the development suite and cross-toolchain, Linux kernel, and the reference root file system, as validated on supported boards and development host operating systems. While under subscription, customers receive both maintenance updates and major upgrades.

Technical Support

Wind River works with every customer to help you solve technical support problems. We may not be able to support every configuration of hardware and software that a customer may have selected, but we will do everything we can to provide support. Linux Technical

Support on modified or unsupported configurations is best-effort-based. Wind River Customer Support will try to reproduce the problem on a supported configuration. If the problem can be validated, we will provide a fix that will be tested on a supported configuration. Wind River Professional Services can provide support for boards or host operating system versions that are not supported by the standard product, as well as for customized versions of the source code or additional nonstandard packages.

Customer Support will provide bug fixes following the process outlined in Wind River's Customer Support User's Guide (CSUG), available at http://www.windriver.com/support/resources/csug.pdf.

If appropriate, Wind River will submit changes in open source code to the open source project maintainer for inclusion in a future release of the open source package. Wind River will maintain changes until a new version from the open source project is available and can be released for Wind River Linux.

Customers with a valid support or subscription agreement are eligible for all respective updates free of charge. If customers cannot update to a new version but need critical parts of the update applied to an older version of the product, Wind River Professional Services can be engaged to backport the required functionality on a case-by-case basis.

Visit Wind River Online Support (OLS) for fast access to product manuals, downloadable software, and other problem-solving resources. OLS offers a comprehensive knowledge base with a robust search feature for locating product information and manuals by keyword, author, published date, document type, language, and solution category. OLS also provides new BSPs, updates to existing packages, patches, manuals, the latest errata, and other announcements about Wind River Linux. Wind River will also provide new contributed Linux packages through our support website. These packages have been contributed by the open source community and are prebuilt and tested with Wind River Linux.

Additional support features, including proactive email alerts covering particular technologies, platforms, or product patches and technical tips for common problems, are available for all customers on subscription. OLS visitors can also access a community of developers to discuss their issues and experiences.

Appendix A: Package Summary by Category

If you are interested in more detailed package information, contact Wind River.

Administration	eel, evlog, fmconf, fmldc, hildon-control-panel, hildon-libs, libglade2, libhildonfm, libhildonhelp, libhildonmime, memstat, monit, openais, quota
Basic C support	binutils, boost, glibc, libaio, libatomic_ops, libcap, libdrm, libgcc, libstdcxx, outo, prelink, wrs_kernheaders
Booting and startup	Bootpc, grub, silo, yaboot
Daemons	acpid, audit, crontabs, daemontools, esound, fam, iozone, iscsi-initiator-utils, mcelog, ORBit2, pcsc-lite, pulseaudio, quagga, samba, vixie-cron, vsftpd, xinetd
Databases	hwdata, libtermcap, mysql, openldap, postgresql, python-ldap, sqlite, unixODBC
Debugging	eventlog, gdb, kexec-tools, libevent, logrotate, ltrace, oprofile, smartmontools, strace, sysklogd, syslog-ng, sysstat, watchdog
Devices	ccid, device-mapper, device-mapper-multipath, eject, ethtool, ipmitool, ipmiutil, kbd, libfakekey, libusb, lm_sensors, makedev, mingetty, minicom, nbd, openipmi, parted, pciutils, scsidev, setserial, udev, usbutils, vblade
DirectFB	dfbtutorials, directfb, directfb_headers
File systems	acl, attr, dmapi, drbd-tools, e2fsprogs, filesystem, fuse, gphotofs, hfsutils, installsw, lsof, lvm2, mdadm, mobile-basic-flash, mtd-utils, mtpfs, rdist, rsync, samhain, xfsdump, xfsprogs, yaffs2
File transforms	bzip2, cpio, gzip, libid3tag, libidn, lzo, shared-mime-info, tar, unzip, zip, zlib
Graphics	atk, cairo, cairo, claws-mail, fbset, fontconfig, freetype, gail, glib2, gnome-desktop, gnome-icon-theme, gnome-keyring, gnome-menus, gnome-mime-data, gnome-panel, gnome-vfs2, gtk, hicolor-icon-theme, libart-lgpl, libgnome, libgnomecanvas, libgnomeui, libgsf, libjpeg, libmatchbox, libpng, librsvg, libtiff, libvisual, libwnck, matchbox-keyboard, matchbox-window-manager, metacity, nautilus, ncurses, pango, pycairo, pygobject, pygtk, sdl, SDL_image, SDL_mixer, SDL_ttf, tslib
Hardware	hal, hal-info, RT71W_Firmware
Kernel	hotplug, intercept_proprietary, kvm, liboil, linux_filter_proprietary, mac-fdisk, pth, rtcore-docs, small-module-init-tools
Languages	gnome-python2, python-gst, ruby
Middleware	ace, dbus, dbus-glib, dbus-python, gypsy, libbonobo, libbonoboui, libIDL, libsoup, paste, startup-notification
Multimedia	cdparanoia, GConf2, gst-plugins-bad, gst-plugins-base, gst-plugins-farsight, gst-plugins-good, gst-plugins-ugly, gstreamer, ial, ImageMagick, libcdio, libexif, libgphoto2, libgweather, libksba, libmms, libmtp, libogg, libtheora, neverball, setmixer, sox, timidity, xulrunner
Networking	agent-proxy, aoetools, apache-ssl, atftp, bind, bluez-libs, boa, curl, dhcp, ecgl-tools, gtkhtml, heartbeat, ifenslave, inetutils, iproute, iptables, iputils, klibc, libnet, libnl, libpcap, librds, lksctp-tools, lrzsz, mailx, midbrowser, mipv6-daemon-umip, netcat, net-snmp, net-tools, nfs-utils, nfs-utils-lib, nspr, portmap, ppp, pyca, radvd, rdate, rsh, sendmail, socat, Socket-CAN, tcpdump, telnet, tipc_demo, tipc-utils, tnftp, traceroute, tunctl, usagi-tool, vlan, wget
Shells and scripting	bash, busybox, expect, gawk, grep, less, microperl, mutagen, pcre, perl, perl_tests, perl-Convert-ASN1, perl-LDAP, perl-XML-Parser, python, python-imaging, sed, tcl, tcsh, xerces

Appendix A (cont.)

Security	beecrypt, cracklib, ecryptfs-utils, freeradius, gnupg2, gnutls, gradm, ipsec-tools, keynote, keyutils, krb5, libassuan, libgcrypt, libgpg-error, librmisec, libsepol, logcheck, nss, opendiameter, openssh, openssh-sftp-only, openssl, pam, pam_passwdqc, passwd, shadow-utils, sudo, tcp_wrappers
SELinux	libselinux, libsemanage, mcstrans, policycoreutils, pyetree, refpolicy, refpolicy-strict, sepolgen, setools
Setup	initscripts, libuser, linux, mm, module-init-tools, procps, psmisc, rng-tools, setup, sysvinit
Sound	alsa-lib, alsa-plugins, alsa-utils, audiofile, lame, libmad, libvorbis, samplerate, sndfile
System	bluez-utils, checkpolicy, crackerjack, fm, hdparm, ipmi-test, ldcskt, ltp-full, netperf, ocfs2-tools, oncpu, openhpi, osso-gwconnect, pinentry, pmem, posixtestsuite, robust-tests, saftest, screen, simple_exec_open, simple_exec_proprietary, unionfs, wireless-tools, wpa_supplicant
Host tools	chkconfig, db4, elfutils, expat, flex, libtool, libxml2, neon, paxctl, rpm
Test	application_args_proprietary, crypto_proprietary, cyclictest, hello_proprietary, lmbench, low_latency_mem_proprietary, m4, mailbox_proprietary, named_block_proprietary, perl-net-telnet, queue_proprietary, traffic_gen_proprietary, uart_proprietary, wifitest, xreg, xts, zebra
Utilities	at, bc, bootlogger, bridge-utils, coreutils, desktop-file-utils, diffutils, file, findutils, gettext, gmp, gnome-doc-utils, gtk-doc, make, mhash, mktemp, mpatrol, newt, ntp, popt, readline, slang, sysfsutils, syslinux, time, timezone, ustr, util-linux, util-linux, which
Various	cyrus-sasl, ed, ElectricFence, freeglut, galculator, gdbm, gpm, mce-dev, vim
Wind River instrumentation	wbagent-ptrace, wr-coverageagent, wr-opagent, wrproxy, wrsv-ltt
X Server	hildon-desktop, hildon-theme-mobile-basic, hildon-thumbnail, libdmx, libfontenc, libICE, libosso, libSM, libX11, libXau, libXaw, libxcb, libXcomposite, libXcursor,, libXdamage, libXdmcp, libXevie, libXext, libXfixes, libXfont, libXfontcache, libXft, libXi, libXinerama, libxkbfile, libxkbui, libxklavier, libXmu, libXpm, libXrandr, libXrender, libXres, libxslt, libXt, libXTrap, libXtst, libXv, libXvMC, libXxf86dga, libXxf86misc, libXxf86vm, marquee-plugins, mesa, mesa-demos, sapwood, ttf-arphic-ukai, ttf-arphic-uming, ttf-dejavu, ttf-kochi, ttf-unfonts-core, xcb-proto, xgtk, xkeyboard-config, xorg-x11-apps, xorg-x11-drv-i810, xorg-x11-drv-keyboard, xorg-x11-drv-mouse, xorg-x11-drv-vesa, xorg-x11-drv-void, xorg-x11-filesystem xorg-x11-fonts, xorg-x11-font-utils, xorg-x11-proto-devel, xorg-x11-server, xorg-x11-server-utils, xorg-x11-twm, xorg-x11-util-macros, xorg-x11-utils, xorg-x11-xauth, xorg-x11-xbitmaps, xorg-x11-xinit, xorg-x11-xkb-utils, xorg-x11-xtrans-devel, xrestop, xterm

Appendix B: Supported Target Boards

Supported and unsupported boards can be found in the Wind River Linux Platforms area of the Wind River Online Support website: http://www.windriver.com/products/bsp_web/.

Appendix C: Supported Development Hosts

The following table contains a complete list of supported development hosts with the necessary updates. It lists which hosts support the Application Developer package only and which hosts also support the Platform Developer package.

Supported Host	Architecture	Platform Developer	Application Developer
Windows XP Professional	x86 32-bit		✓
Windows Vista Business	x86 32-bit		✓
Windows Vista Enterprise	x86 32-bit		✓
Fedora 9	x86 64-bit	✓	✓
Red Hat Enterprise Linux Workstation 5	x86 32-bit	✓	✓
Red Hat Enterprise Linux Desktop with Workstation 5	x86 32-bit, x86 64-bit	✓	✓
SUSE Linux/openSUSE 11	x86 32-bit	✓	✓
Novell SUSE Linux Enterprise Desktop 10	x86 32-bit x86 64-bit	✓	✓
Ubuntu Desktop Edition 8.0.4	x86 64-bit	✓	✓
Sun Solaris 9 (Update 9/05, GTK only)	SPARC 32-bit		✓
Sun Solaris 10	SPARC 32-bit		✓

Note that although development may be possible on other Linux distributions and versions, Wind River has not certified the product on them.

For more details on features of Wind River Linux, contact Wind River or visit http://www.windriver.com/linux/.

WIND RIVER

Wind River is the global leader in Device Software Optimization (DSO). We enable companies to develop, run, and manage device software better, faster, at lower cost, and more reliably. www.windriver.com