WIND™

WIND RIVER® LINUX

GETTING STARTED

7.0

Copyright Notice

Corporate Headquarters

Wind River
500 Wind River Way
Alameda, CA 94501-1153
U.S.A.
Toll free (U.S.A.): 800-545-WIND
Telephone: 510-748-4100
Facsimile: 510-749-2010

For additional contact information, see the Wind River Web site:

www.windriver.com

For information on how to contact Customer Support, see:

www.windriver.com/support

22 May 2015

Contents

1

Overview

Wind River Linux

Wind River Linux is a software development environment that creates optimized Linux distributions for embedded devices.

Wind River Linux is based on the Yocto Project implementation of the OpenEmbedded Core (OE-Core) metadata project. The Yocto Project uses build recipes and configuration files to define the core platform project image and the applications and functionality it provides.

Wind River Linux builds on this core functionality and adds Wind River-specific extensions, tools, and services to facilitate the rapid development of embedded Linux platforms. This support includes:

- straightforward platform project configuration, build, and deployment that simplifies Yocto Project development
- a range of popular BSPs to support most embedded hardware platforms
- an enhanced command-line-interface (CLI) to the system
- developer-specific layer for platform project development and management
- platform project portability to copy or move platform projects, or create a stand-alone platform project
- package revision management through the implementation of the package revision server.
- a custom USB image tool for platform project images

Wind River Linux supports all Yocto Project build commands, but also offers simplified configure and build commands based on the Wind River Linux LDAT build system. This functionality greatly reduces your development time.

Wind River Linux provides development environments for a number of host platforms and supports a large and ever-growing set of targets, or the platform hardware you are creating your embedded system for. For details on which development hosts are supported, refer to the release notes. For supported target boards, refer to Wind River Online Support.

The build system consists of a complete development environment that includes a comprehensive set of standard Linux run-time components, both as binary and source packages. It also includes cross-development tools that can be used to configure and build customized run-time systems and applications for a range of commercial-off-the-shelf (COTS) hardware.

Wind River supports boards according to customer demand. Please contact Wind River if yours is not yet officially supported.

Wind River Workbench is included as part of Wind River Linux to provide a robust development and debugging environment.

For more information about Wind River Linux, see *http://www.windriver.com/products*.

Embedded Development Environment Overview

Embedded systems are typically configured, programmed and built on a host system and then deployed to a target system.

If you have a physical target system, or board, you will test your platform and application on that target. Your target might have a network interface and on-board tools through which you can deploy software.

You may not have the actual target available at all times, so it may be more convenient to use a target simulator on your host. Wind River Linux provides simulated target development with QEMU and Wind River Simics, in addition to supporting a range of hardware target board support packages (BSPs). Refer to the *Wind River Linux Release Notes* for information on supported BSPs.

This section describes typical development environments that are supported by Wind River Linux.

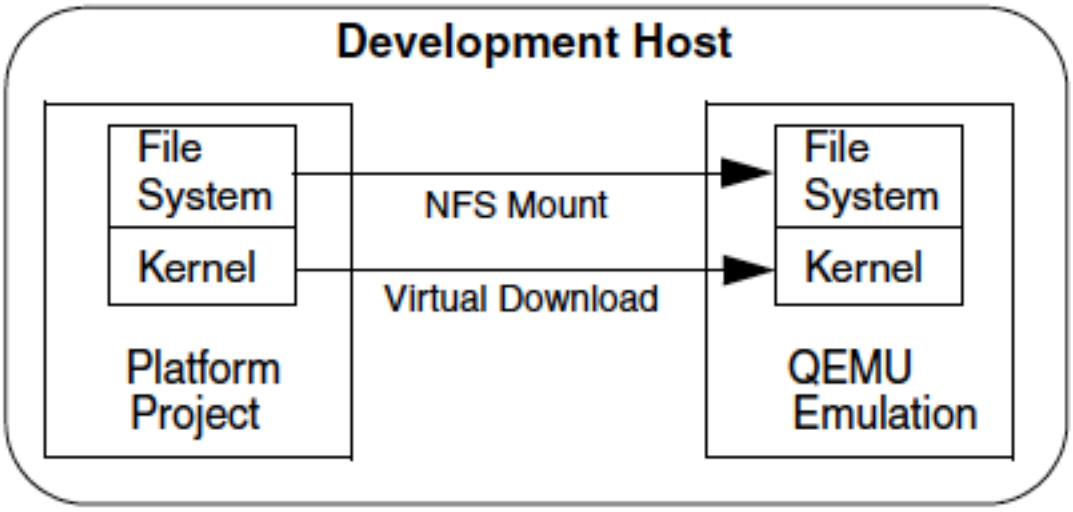
The QEMU Development Environment

QEMU (Quick Emulator) provides a simulated deployment environment for testing platform projects and applications.

QEMU supports many development boards (refer to your Release Notes for a list of supported boards) and does not require actual target board hardware or networking preliminaries. QEMU deployment offers a suitable environment for application development and architectural level validation. User-space and kernel binaries are compatible with the real hardware.

QEMU provides a means to rapidly test and debug platform projects and applications using Workbench or the command-line.

The following figure illustrates a typical working environment for developing Wind River Linux for a target board by using an emulation of that board.



The emulated target is passed the name of the kernel file as a parameter (-kernel) and NFS-mounts the root file system from the development host at boot time. The difference is that the target is a simulation running on the development host, not external hardware.

For details on the use of QEMU, refer to the *Wind River Linux User's Guide*. For open source information on QEMU, see *http://wiki.qemu.org*.

The Simics Development Environment

Wind River Simics is a fast, functionally-accurate, full system simulator.

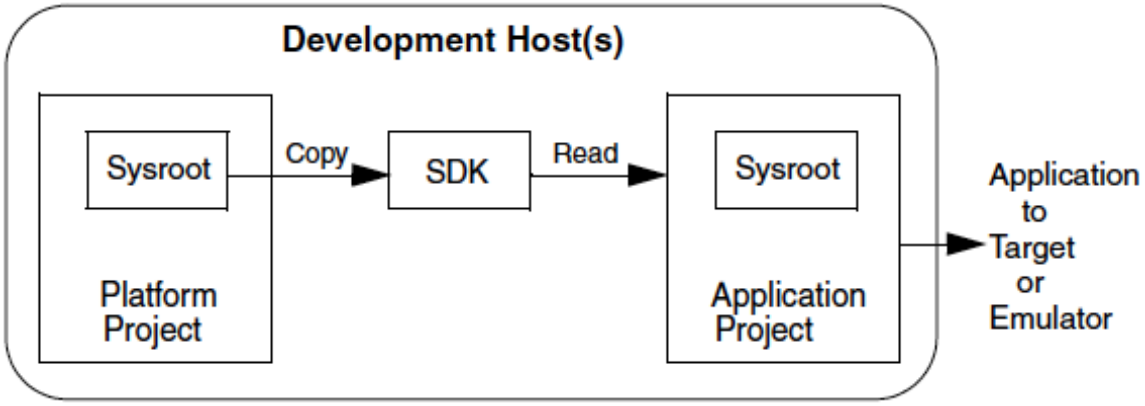
Simics creates a high-performance virtual environment in which any electronic system – from a single board to complex, heterogeneous, multi-board, multi-processor, multi-core systems – can be defined, developed and deployed.

For basic information on using Simics with Wind River Linux, see: *WindRiver Linux User's Guide*.

To purchase Simics, contact your Wind River sales representative.

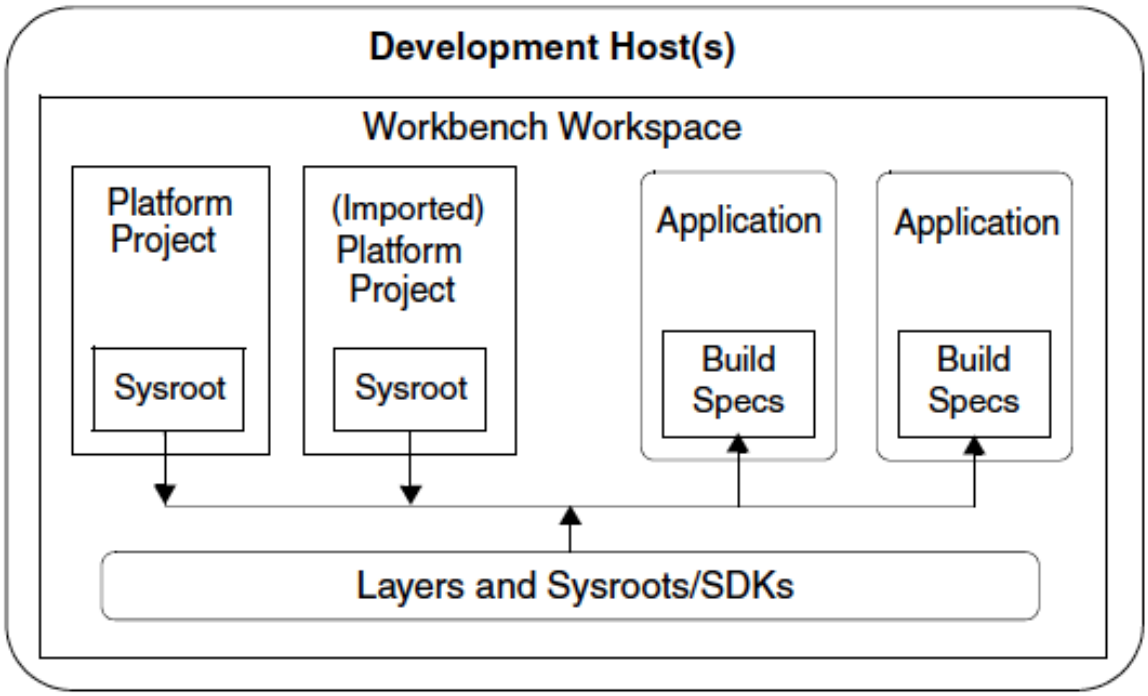
The Application Development Environment

The following figure illustrates a typical working environment for developing a Wind River Linux application for a target board. A platform developer provides the application developer with a sysroot (see *Wind River Linux Build System Concepts* on page 8).



The application developer can use Workbench or command-line tools. In either case, the developer creates an application project and associates the project with the platform project sysroot. After building (cross-compiling) the application, it can be deployed to the emulated or physical target for testing and/or debugging.

The following figure illustrates the benefits of using Workbench for application development. With Workbench, each platform project and/or application, whether created with Workbench or imported to the Workbench workspace, automatically shares their inner sysroots with all Workbench applications. Additionally, resources from layers, sysroots, and SDKs in the development environment are also available. This shared-resource approach helps simplify and streamline application development.



Wind River Linux Build System Concepts

Understanding the build system and terminology prepares you to develop platform and application projects with Wind River Linux.

Whether you are using command-line tools or Wind River Workbench, you need to understand the concepts listed below in order to develop Wind River Linux platforms or applications. See the *Wind River Linux User's Guide* for more information.

Platform Project

A platform project is a configured and built Wind River Linux system contained in a directory that you create. Once a project is configured and built, the resulting platform project directory contains:

- the Linux kernel to be deployed to the embedded target

- the target file system, including packages, executables, and other files, which are located in the **/export/dist** folder of the directory of the platform project.
- files required by the build system such as makefiles and configuration (**.conf**) files
- layers required by the project, as well as the local layer of the project for use in application development and custom package development

A platform project developer configures the project, defining the BSP, Linux kernel, and type of root file system. In addition, configuration options can include layers, packages, kernel configuration, and templates. Once a platform project is configured appropriately, it is built, resulting in a deployable Wind River Linux kernel and file system.

See the *Glossary* on page 15 for definitions of these terms.

Optional Add-on Products

Wind River Linux supports optional add-on products that extend its capabilities to meet your development needs.

This version of Wind River Linux supports the following add-on products:

Wind River Simics System Simulator

Wind River Simics is a fast, functionally-accurate, full system simulator. Simics creates a high-performance virtual environment in which any electronic system – from a single board to complex, heterogeneous, multi-board, multi-processor, multicore systems – can be defined, developed and deployed.

Simics enables companies to adopt new approaches to the product development life cycle resulting in dramatic reduction in project risks, time to market, and development costs while also improving product quality and engineering efficiency. Simics allows engineering, integration and test teams to use approaches and techniques that are simply not possible on physical hardware.

To purchase Wind River Simics, contact your Wind River sales representative.

2

Workflows

Platform Project Workflow

The platform project defines the Wind River Linux system image you will use as a base to add your applications to.

Development of a platform project consists of the steps listed below. Workbench and command-line tutorials that provide step-by-step instructions are available in the Knowledge Library at *https://knowledge.windriver.com*.

1. Install the product and any updates. This typically includes installing Wind River Linux, Wind River Workbench, and optional products. For information on product installation, see the Wind River product installation and licensing guides at *http://www.windriver.com/licensing/documents/*.
2. Create and configure a Wind River Linux platform project. A platform project provides a deployable Linux platform for your development needs. To create a platform project, you can use Workbench or command-line tools. This includes:
   - creating a project directory and setting environment variables,
   - configuring the platform project by selecting an appropriate board support package (BSP), kernel, and root file system,
   - optionally specifying additional userspace packages, layers, or templates to include in the platform project.
3. Build the Wind River Linux platform project. Building a platform project results in a kernel and file system that are ready for deployment to a target or an emulator. You can build the project using Workbench or command-line tools. The Wind River Linux build system can also output the appropriate configuration of development tools with which to develop applications for that platform. This can include analysis tools such as **gdb** and **mpatrol**, or any package you specify, to aid in project development.

4. Tailor the platform. This may include adding applications and packages, and/or modifying the kernel
5. Deploy the platform project. Use Workbench or the command-line to deploy the new platform to a target.
6. Test and debug the platform. Use Workbench or command-line tools to test and debug platform executables. Depending on how you configure the platform project, you may be able to test and debug directly on the target. Otherwise, you can do so from the host. For more information, see *Wind River Linux User's Guide*.

Application Development Workflow

There are common steps to take in application development for a platform project image.

Application development typically consists of the following steps:

1. Create and build an application.

   Workbench provides tools for managing application files and editing software, or you can use command-line tools.
2. Deploy the platform project.

   You can use Workbench to deploy a platform project, including the application executable, to a simulated or real target. You can also do so using command-line tools if you prefer.
3. Run or debug the application.

   You can use Workbench to remotely execute or debug the target application from the host. Using command-line tools, you can open a terminal to the simulated or hardware target and type a command to run the application. How you debug the application depends on how the platform project was configured. The platform project may include tools so that you can debug the application directly on the target. Otherwise, you can remotely debug the application on the development host using tools such as **gdb**.

For information about how to accomplish the above steps using Workbench and the command-line, see: *Wind River Linux 7 Workbench Tutorials* and *Wind River Linux 7 Command-Line Tutorials*.

3

Documentation

Wind River Linux Documentation

Understanding the full range of documentation available is invaluable to helping you find the information you need to work effectively in Wind River Linux.

All Wind River Systems documentation is available in the Knowledge Library *https://knowledge.windriver.com*.

Additionally, much of the documentation is available through the start menu of the installation host, for example under **Applications** > **Wind River** > **Documentation**in the Gnome desktop for Linux.

Most of the documentation is available online as PDFs or HTML accessible through Wind River Workbench online help. Links to the PDF files are available by selecting **Wind River** > **Documentation** from your operating system start menu.

From a Workbench installation you can view the documentation in a Web browser locally (**Help** > **Help Contents**).

The documentation is also available below your installation directory (called `installDir`) through the command line as follows:

- PDF Versions—To access the PDF, point your PDF reader to the ***.pdf** file, for example:

  ***installDir*/docs/extensions/eclipse/plugins/com.windriver.ide.doc.wr_linux_7/wind_river_linux_users_guide_70/wind_river_linux_users_guide_70.pdf**
- HTML Versions—To access the HTML, point your web browser to the **index.html** file, for example:

  ***installDir*/docs/extensions/eclipse/plugins/com.windriver.ide.doc.wr_linux_7/wind_river_linux_users_guide_70/html/index.html**

External Documentation

Use external documentation to enhance your developer knowledge and capabilities. The following table lists open source documentation related to the Yocto Project and Wind River Linux:

Table 1 **Wind River Linux External Documentation**

External Information source	Location
The Yocto Project	Online: *http://www.yoctoproject.org*
BitBake User Manual	Online: *http://www.yoctoproject.org/docs/1.7/bitbake-user-manual/bitbake-user-manual.html*
OpenEmbedded Core (OE-Core)	Online: *http://www.openembedded.org/wiki/OpenEmbedded-Core*
QEMU	Online: *http://wiki.qemu.org*
GNU Toolchain Documentation	***installDir*/wrlinux-7/layers/binary-toolchain-4.9-3/share/doc/wrs-linux-arm-wrs-linux-gnueabi**—for ARM-based target systems ***installDir*/wrlinux-7/layers/binary-toolchain-4.9-3/share/doc/wrs-linux-aarch64-wrs-linux-gnu**—for 64-bit ARM-based target systems ***installDir*/wrlinux-7/layers/binary-toolchain-4.9-3/share/doc/wrs-linux-i686-wrs-linux-gnu**—for x86-based target systems ***installDir*/wrlinux-7/layers/binary-toolchain-4.9-3/share/doc/wrs-linux-mips-wrs-linux-gnu**—for MIPS-based target systems
GNU Toolchain Documentation	***installDir*/wrlinux-7/layers/binary-toolchain-4.9-3/share/doc/wrs-linux-arm-wrs-linux-gnueabi**—for ARM-based target systems ***installDir*/wrlinux-7/layers/binary-toolchain-4.9-3/share/doc/wrs-linux-aarch64-wrs-linux-gnu**—for 64-bit ARM-based target systems ***installDir*/wrlinux-7/layers/binary-toolchain-4.9-3/share/doc/wrs-linux-i686-wrs-linux-gnu**—for x86-based target systems ***installDir*/wrlinux-7/layers/binary-toolchain-4.9-3/share/doc/wrs-linux-mips-wrs-linux-gnu**—for MIPS-based target systems

Appendix A

Glossary of Terms

Glossary

Application Project

An application project contains the source files, makefile, and other items required to develop an embedded application, and typically resides in a directory, similar to a platform project. If you are using Workbench, Workbench provides built-in tools for editing, running, debugging and measuring applications on embedded targets. A configured and built Wind River Linux platform project includes a local layer for the developers' application projects and their related recipes.

BitBake

BitBake is the build tool used to parse metadata, generate a list of tasks and then execute them. A typical metadata is a BitBake recipe which is used to specify how to fetch, unpack, patch, and compile an application package.

BitBake Recipe

A recipe is an application- or package-specific configuration file that provides information on the source, revision, license, and compile/install functions. Each application or package that you include in your platform project build requires its own recipe file.

Board Support Package (BSP)

A collection of packages that allow Wind River Linux to run on a given hardware platform board.

Build Spec

In Workbench, a build spec is associated with a sysroot. You associate your application project with a build spec so that Workbench can use the correct sysroot.

Kernel Module

A kernel module is an object file with code specific to the target system hardware. This file extends the running base kernel to support the features of the board for the target operating system. When you build a target file system with Wind River Linux, the kernel module is located in the **/export/images** folder of the project directory.

Layer

A directory whose contents are used to customize a platform project. It can contain templates and other items, such as board support packages (BSPs), applications, and files. Wind River Linux provides layers for customizing your platform projects, and also adds a default **/layers/local** directory to your configured and built platform project directory for your application projects.

Package

A packaging unit of user space content, including possible user content.

Perspective

Since Workbench is based on Eclipse, it makes use of the Eclipse concept of perspective. A perspective is a collection of views that are associated with a task. For example, the device debug perspective includes editor, debugger, and remote connection views. The Performance Profiler perspective includes tools for measuring target software performance. See the *Wind River Workbench User's Guide* for additional information on perspectives.

SDK

A software development kit that contains the exported sysroot of a project, toolchain, and emulators for stand-alone development away from the parent project.

Sysroot

A sysroot identifies the correct versions of tools required to build an application for a given Wind River platform project. It includes the libraries and header files necessary to compile and link an application so it will run on the target. The platform project developer typically exports a sysroot for application developers once the project has been built. Workbench uses the sysroot in order to correctly cross-compile an application.

Template

A template is a collection of configuration files (fragments) that enable specific features in a platform project image. Templates are contained within layers, and may be specified at project configure and build, or included by default from the build system.