VxWorks 6.6 SMP

We’re at the beginning of a very important marketplace trend—growth in the use of multiprocessing. More and more devices will be designed using multicore processors to get to the next level of device performance. This document explores multicore technology and the choices and challenges it presents to the device software developer, and how the symmetric multiprocessing (SMP) features in Wind River’s VxWorks platforms enable VxWorks users to move into this exciting new area of technology.

VxWorks SMP is an add-on product to all VxWorks 6.6-based platforms. It is an evolution of the VxWorks RTOS to provide SMP capabilities, not a new operating system. VxWorks SMP leverages multicore processors to achieve true concurrent execution of applications, allowing applications to gain performance through parallelism. Users may purchase it either with the base VxWorks 6.6 platform or any time after. Once installed, users can develop on both VxWorks uniprocessor (UP) and SMP according to their project needs.

The diagram above depicts a VxWorks platform installation with SMP.

What’s New in VxWorks 6.6 SMP?

The VxWorks SMP-enabled platforms allow customers to:

- Deliver higher performance multicore-powered products with reduced risk and development investment
- Speed time-to-market by using the commercially available and supported run-time platforms and developer tools for multiprocessing
VxWorks SMP introduces the following key features:

- **Multitasking**: Allows true concurrent execution of tasks and handling of interrupts. SMP changes the conventional UP paradigm of priority-based preemptive multitasking programming. This is because multiple tasks can run on multiple CPUs while being controlled by a single instance of an operating system. UP multitasking environments are often described as ones in which multiple tasks can run at the same time, but the reality is that the CPU only executes one task at a time, switching from one task to another based on the characteristics of the scheduler and the arrival of interrupts. In an SMP system, concurrent execution is a reality and not an illusion.

- **Concurrent task scheduling**: Provides a priority-based preemptive scheduler that also manages the concurrent execution of tasks on different CPUs.

- **Mutual exclusion**: Provides specialized mechanisms for mutual exclusion between tasks executing and interrupts being received simultaneously on different CPUs. Because SMP systems allow for truly concurrent execution, the uniprocessor mechanisms for disabling (masking) interrupts and for suspending task preemption in order to protect critical regions are inappropriate for—and not available in—an SMP operating system. Enforcing interrupt masking or suspending task preemption across all CPUs would defeat the advantages of concurrent execution and drag multiprocessing performance down towards the level of a UP system. VxWorks SMP therefore provides specialized mechanisms for mutual exclusion between tasks and interrupts. In place of uniprocessor task and interrupt locking routines, VxWorks SMP provides spinlocks, atomic memory operations, and CPU-specific mutual exclusion facilities.

- **CPU affinity**: Provides the ability to assign specific tasks or interrupts to a specific CPU. By default any task can run on any of the CPUs in the system, which generally provides the best load balancing, and interrupts are routed to CPU 0 (the bootstrap CPU). However, this capability may be useful for instances where the designer wishes to assign a task or interrupt to a specific processor resource.

- **Binary and API compatibility**: Maintains substantial commonality between the binary code and the APIs used for both the uniprocessor and SMP configurations. The differences in the APIs are a small number of routines, accounting for APIs not suitable for an SMP system or which are not relevant to a uniprocessor system. This allows applications that use the subset of APIs defined for SMP to have binary compatibility with VxWorks uniprocessor configuration.

- **VxWorks SMP simulation capability**: Allows development of SMP applications to begin without physical hardware. The Wind River VxWorks Simulator provides default SMP system images to develop and test the API's use in the applications. SMP simulators are provided with the standard UP VxWorks installations as an introduction to the SMP product.

- **Broad multicore hardware compatibility**: Provides support for the leading multicore silicon available on the market, giving customers a choice in functionality for their specific device requirements.

The following are supported multicore processors:

- **ARM11 MPCore (ARMv6)**
- **Broadcom BCM1480**
- **Cavium OCTEON CN38XX**
- **Freescale MPC8641D, MPC8572**
- **Dual-Core Intel Xeon processor LV**
- **Intel Core Duo T2400**
- **Raza XLR 732**

### Technology Overview

VxWorks SMP is a configuration of VxWorks designed for symmetric multiprocessing (SMP). It provides the same distinguishing RTOS characteristics of performance, small footprint, high reliability, and determinism as the UP configuration. The differences between the SMP and uniprocessor (UP) configurations are limited, and strictly related to support for multiprocessing.

Multiprocessing systems include two or more processors in a single system. SMP is a variant of multiprocessing technology in which one instance of an operating system controls all processors, and in which memory is shared. SMP differs from asymmetric multiprocessing (AMP) in that an AMP system has a separate instance of an operating system executing on each processor (and each instance may or may not be the same type of operating system).

### Terminology

The terms CPU and processor are often used interchangeably in computer literature. However, it is useful to distinguish between the two for hardware that supports SMP. In this guide, particularly in the context of VxWorks SMP, the terms are used as follows:

- **CPU**: A single processing entity capable of executing program instructions and processing data (also referred to as a core, as in multicore)
- **Processor**: A silicon unit that contains one or more CPUs
- **Multiprocessor**: A single hardware system with two or more CPUs
- **Uniprocessor (UP)**: A silicon unit that contains a single CPU

For example, a dual-core processor has two CPUs. A quad-core processor has four CPUs.
UP code may not always execute properly on an SMP system, and code that has been adapted to execute properly on an SMP system may still not make optimal use of SMP. The following terms are therefore used to clarify the state of code in relation to SMP.

- **SMP-ready**: Runs correctly on an SMP operating system, although it may not make use of more than one CPU (that is, does not take full advantage of concurrent execution for better performance).
- **SMP-optimized**: Runs correctly on an SMP operating system, uses more than one CPU, and takes sufficient advantage of multitasking and concurrent execution to provide performance gains over a UP implementation.

### VxWorks SMP Operating System Features

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

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

### Multitasking

SMP changes the conventional UP paradigm of priority-based preemptive multitasking programming, because it allows true concurrent execution of tasks and handling of interrupts. This is possible because multiple tasks can run on multiple CPUs, while being controlled by a single instance of an operating system.

UP multitasking environments are often described as ones in which multiple tasks can run at the same time, but the reality is that the CPU only executes one task at a time, switching from one task to the other based on the characteristics of the scheduler and the arrival of interrupts. In an SMP system, concurrent execution is a fact and not an illusion.

### Scheduling

VxWorks SMP provides a priority-based preemptive scheduler, like VxWorks UP. In both VxWorks UP and VxWorks SMP, tasks are scheduled—and real-time processes (RTPs) are not. However, the VxWorks SMP scheduler is different from the UP scheduler in that it also manages the concurrent execution of tasks on different CPUs.

### Mutual Exclusion

Because SMP systems allow for truly concurrent execution, the UP mechanisms for disabling (masking) interrupts and for suspending task preemption in order to protect critical regions are inappropriate for—and not available in—an SMP operating system. Enforcing interrupt masking or suspending task preemption across all CPUs would defeat the advantages of truly concurrent execution and drag multiprocessing performance down towards the level of a UP system.

VxWorks SMP therefore provides specialized mechanisms for mutual exclusion between tasks and interrupts executing and being received (respectively) simultaneously on different CPUs. In place of UP task and interrupt locking routines—such as `taskLock()` and `intLock()`—VxWorks SMP provides spinlocks, atomic memory operations, and CPU-specific mutual exclusion facilities.

### CPU Affinity

By default, any task can run on any of the CPUs in the system (which generally provides the best load balancing) and interrupts are routed to CPU 0 (the bootstrap CPU). There are instances, however, in which it is useful to assign specific tasks or interrupts to a specific CPU. VxWorks SMP provides this capability, which is referred to as CPU affinity.

### VxWorks SMP Hardware Requirements

The hardware required for use with VxWorks SMP must consist of symmetric multiprocessors—either multicore processors or hardware systems with multiple single CPUs. The processors must be identical, all memory must be shared between the CPUs (none may be local to a CPU), and all devices must be equally accessible from all CPUs. That is, targets for VxWorks SMP must adhere to the uniform memory access (UMA) architecture.

Regardless of the number of CPUs (typically 2, 4, or 8) in an SMP system, the important characteristics are the same:

- Each CPU accesses the very same physical memory subsystem; there is no memory local to a CPU. This means it is irrelevant which CPU executes code.
- Each CPU has its own memory management unit that allows concurrent execution of tasks with different virtual memory contexts. For example, CPU 0 can execute a task in RTP 1 while CPU 1 executes a task in RTP 2.
- Each CPU has access to all devices. Interrupts from these devices can be routed to any one of the CPUs through a programmable interrupt controller. This means that it is irrelevant which CPU executes interrupt service routines (ISRs) when handling interrupts.
- Tasks and ISRs can be synchronized across CPUs and mutual exclusion can be enforced by using spinlocks.
- Bus snooping logic ensures the data caches between CPUs are always coherent. This means that the operating system does not normally need to perform special data cache operations order to maintain coherent caches. However, this implies that only memory access attributes that allow bus snooping are used in the system. Restrictions in terms of memory access modes allowed in an SMP system, if any, are specific to a hardware architecture.
Comparison of VxWorks SMP and AMP

The features of VxWorks SMP may be highlighted by comparison with the way VxWorks is used in asymmetric multiprocessor (AMP), using the same target hardware in both cases. VxWorks AMP technologies include VxMP, TIPC (over shared memory), and distributed shared memory (DSHM). The relationship between CPUs and basic uses of memory in SMP and AMP systems are illustrated in Figures 2 and 3.

In a symmetric multiprocessing (SMP) configuration, the entire physical memory space is shared between the CPUs. This memory space is used to store a single VxWorks SMP image (text, data, bss, heap). It is also used to store any real-time processes (RTPs) that are created during the lifetime of the system. Because both CPUs can potentially read from, write to, and execute any memory location, any kernel task or user (RTP) task can be executed by either CPU.

In an AMP configuration there is one copy of the VxWorks image in memory for each CPU. Each operating system image can only be accessed by the CPU to which it belongs. It is therefore impossible for CPU 1 to execute kernel tasks residing in VxWorks CPU 0’s memory, or the reverse. The same situation applies for RTPs. An RTP can only be accessed and executed by the instance of VxWorks from which it was started.

In an AMP system some memory is shared, but typically the sharing is restricted to reading and writing data—for example, for passing messages between two instances of VxWorks. Hardware resources are mostly divided between instances of the operating system, so that coordination between CPUs is only required when accessing shared memory.

With an SMP system, both memory and devices are shared between CPUs, which requires coordination within the operating system to prevent concurrent access to shared resources.

Programming for VxWorks SMP

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

However, because of the nature of SMP systems, SMP programming requires special attention to the mechanisms of mutual exclusion, and to design considerations that allow for full exploitation of the capabilities of a multiprocessor system. Also note that VxWorks SMP maintains an idle task for each CPU, and that idle tasks must not be interfered with.

### SMP and Mutual Exclusion

The use of mutual exclusion facilities is one of the critical differences between UP and SMP programming. While some facilities are the same for VxWorks UP and VxWorks SMP, others are necessarily different. In addition, reliance on implicit synchronization techniques—such as relying on task priority instead of explicit locking—does not work in an SMP system.

Unlike UP systems, SMP systems allow for truly concurrent execution, in which multiple tasks may execute, and multiple interrupts may be received and serviced, all at the same time. In most cases, the same mechanisms—semaphores, message queues, and so on—can be used in both UP and SMP systems for mutual exclusion and coordination of tasks.

However, the specialized UP mechanisms for disabling (masking) interrupts and for suspending task preemption in order to protect critical regions are inappropriate for—and not available in—an SMP system. This is because they would defeat the advantages of truly concurrent execution by enforcing masking or preemption across all CPUs, and thus drag a multiprocessing system down towards the performance level of UP system.
The most basic differences for SMP programming therefore have to do with the mechanisms available for mutual exclusion between tasks and interrupts executing and being received (respectively) on different CPUs. In place of UP task and interrupt locking routines—such as taskLock() and intLock()—VxWorks SMP provides the following facilities:

- Spinlocks for tasks and ISRs
- CPU-specific mutual exclusion for tasks and ISRs
- Atomic memory operations
- Memory barriers

As with the UP mechanisms used for protecting critical regions, spinlocks and CPU-specific mutual exclusion facilities should only be used when they are guaranteed to be in effect for very short periods of time. The appropriate use of these facilities is critical to making an application SMP-ready.

**CPU Affinity for Interrupts and Tasks**

By default, any task can run on any of the CPUs in the system (which generally provides the best load balancing) and interrupts are routed to CPU 0. There are cases, however, in which it is useful to assign tasks or interrupts to a specific CPU. VxWorks SMP provides this capability, which is referred to as CPU affinity.

**RTP Applications**

As in VxWorks UP systems, RTP (user-mode) applications have a more limited set of mutual exclusion and synchronization mechanisms available to them than kernel code or kernel applications. In VxWorks SMP, they can make use of semaphores and atomic operations, but not spinlocks, memory barriers, or CPU-specific mutual exclusion mechanisms.

**Spinlocks for Mutual Exclusion and Synchronization**

Spinlocks provide a facility for short-term mutual exclusion and synchronization in an SMP system. Spinlocks must be explicitly taken and released. While semaphores can also be used for mutual exclusion and synchronization, spinlocks are designed for use in situations comparable to those in which taskLock() and intLock() are used in VxWorks UP. Semaphores should be used in an SMP system for the same purposes as in a UP system. (Note that both spinlocks and semaphores provide full memory barriers.)

One of the unique characteristics of VxWorks spinlocks is that they are implemented with algorithms that ensure that they are fair, meaning that they are deterministic in the time between the request and take, and they operate as close to FIFO order as possible.

Note that spinlocks are not available to RTP (user-mode) applications.

**Types of Spinlocks**

VxWorks SMP spinlocks:

- **ISR-callable spinlocks**: Used to address contention between ISRs—or between a task and other tasks and ISRs. They disable (mask) interrupts on the local CPU. When called by tasks, they suspend task preemption on the local CPU as well.

- **Task-only spinlocks**: Used to address contention between tasks (and not ISRs). They suspend task preemption on the local CPU. The local CPU is the one on which the spinlock call is performed.

**Spinlocks as Full Memory Barriers**

VxWorks spinlocks operate as full memory barriers between acquisition and release. A full memory barrier forces both read and write memory access operations to be performed in strict order. The process of updating data structures is therefore fully completed between the time a spinlock is acquired and released.

**Spinlock Behavior and Usage Guidelines**

Unlike the behavior associated with semaphores, a task that attempts to take a spinlock that is already held by another task does not pend. Instead it continues executing, simply spinning in a tight loop waiting for the spinlock to be freed.

The terms spinning and busy waiting—which are both used to describe this activity—provide insight into both the advantages and disadvantages of spinlocks. Because a task (or ISR) continues execution while attempting to take a spinlock, the overhead of rescheduling and context switching can be avoided (which is not the case with a semaphore). On the other hand, spinning does no useful work, and ties up one or more of the CPUs. Spinlocks should therefore only be used when they are likely to be efficient; that is, when they are going to be held for very short periods of time (as with taskLock() and intLock() in a uniprocessor system). If a spinlock is held for a long period of time, the drawbacks are similar to intLock() and taskLock() being held for a long time in VxWorks UP—increased interrupt and task latency.

Acquisition of a spinlock on one CPU does not affect the processing of interrupts or scheduling of tasks on other CPUs. Tasks cannot be deleted while they hold a spinlock.

**Task-Only Spinlocks**

Spinlocks that are used to address contention between tasks alone (and not ISRs) are called task-only spinlocks. These spinlocks disable task preemption on the local CPU while the caller holds the lock (which could otherwise lead to a livelock situation). This prevents the caller from being preempted by other tasks and allows it to execute the critical section that the lock is protecting. Interrupts are not disabled and task preemption on other CPUs is not affected. For VxWorks UP, task-only spinlocks are implemented with the same behavior as the task locking routines taskLock() and taskUnlock().

**CPU-Specific Mutual Exclusion**

VxWorks SMP provides facilities for CPU-specific mutual exclusion, which are for mutual exclusion operations whose scope is entirely restricted to the CPU on which the call is made (the local CPU). These facilities are designed to facilitate porting uniprocessor (UP) code to a symmetric multiprocessing (SMP) system.
**CPU-Specific Mutual Exclusion for Interrupts**

CPU-specific mutual exclusion for interrupts allows for disabling (masking) interrupts on the CPU on which the calling task or ISR is running. For example, if task A, running on CPU 0, performs a local CPU interrupt lock operation, no interrupts can be processed by CPU 0 until the lock is released by task A. Execution of interrupts on other CPUs in the SMP system is not affected. In order to be an effective means of mutual exclusion, therefore, all tasks and ISRs that should participate in the mutual exclusion scenario should have CPU affinity set for the local CPU. Note that some routines should not be used if the calling task or ISR has locked interrupts on the local CPU—similar to the case of holding spinlocks.

**CPU-Specific Mutual Exclusion for Tasks**

CPU-specific mutual exclusion for tasks allows for suspending task preemption on the CPU on which the calling task is running. That is, it provides for local CPU task locking, and effectively prevents any other task from running on the local CPU. For example, task A running on CPU 0 can perform a local CPU task lock operation so that no other task can run on CPU 0 until it releases the lock or makes a blocking call. The calling task is also prevented from migrating to another CPU until the lock is released. Execution on other CPUs in the SMP system is not affected. In order to be an effective means of mutual exclusion, therefore, all tasks that should participate in the mutual exclusion scenario should have CPU affinity set for the local CPU.

**Memory Barriers**

A memory barrier is a class of instructions that cause a CPU to enforce ordering of memory operations. CPUs use optimizations that can cause an out-of-order execution, which causes unpredictable behavior in concurrent programs and device drivers in SMP systems. If code depends on the order in which another CPU writes to memory, a memory barrier should be used. Note that both semaphores and spinlocks provide full memory barriers on their own.

**Atomic Memory Operations**

Atomic operations make use of CPU support for atomically accessing memory. They combine a set of architecture-specific operations into what is effectively a single operation that cannot be interrupted by any other operation on the memory location in question. Atomic operations thereby provide mutual exclusion for a simple set of operations such as incrementing and decrementing variables. Atomic operations can be useful as a simpler alternative to spinlocks, such as for updating a single data element. For example, you can update the next pointer in a singly linked list from NULL to non-NULL (without interrupts locked) using an atomic operation, which allows you to create lockless algorithms. Because the atomic operations are performed on a memory location supplied by the caller, users must ensure the location has memory access attributes and an alignment that allows atomic memory access—otherwise an access exception will occur. Restrictions, if any, are specific to the CPU architecture.

**CPU Affinity**

VxWorks SMP provides facilities for CPU affinity; that is, for assigning specific interrupts or tasks to specific CPUs.

**Task CPU Affinity**

VxWorks SMP provides the ability to assign tasks to a specific CPU, after which the scheduler ensures the tasks are only executed on that CPU. This assignment is referred to as task CPU affinity.

While the default symmetric multiprocessing (SMP) operation in which any task can run on any CPU often provides the best overall load balancing, there are cases in which assigning a specific set of tasks to a specific CPU can be useful. For example, if a CPU is dedicated to signal processing and does no other work, the cache remains filled with the code and data required for that activity. This saves the cost of moving to another CPU—which is incurred even within a single piece of silicon, as the L1 cache is bound to a single CPU, and the L1 must be refilled with new text and data if the task migrates to a different CPU.

**Another example is a case in which profiling an application reveals that some of its tasks are frequently contending for the same spinlock, and a fair amount of execution time is wasted waiting for a spinlock to become available. Overall performance could be improved by setting task CPU affinity such that all tasks involved in spinlock contention run on the same CPU. This would free up more time on other CPUs for other tasks.**

**Interrupt CPU Affinity**

SMP hardware requires programmable interrupt controller devices. VxWorks SMP makes use of this hardware to allow assignment interrupts to a specific CPU. By default, interrupts are routed to the bootstrap CPU (CPU 0).

Interrupt CPU affinity can be useful for load balancing; for example, if there is too much total interrupt traffic for one CPU to handle. It can also be used as an aid in migrating code from VxWorks UP. Runtime assignment of interrupts to a specific CPU occurs at boot time, when the system reads interrupt configuration information from the board support package (BSP). The interrupt controller then receives a command directing that a given interrupt be routed to a specific CPU.

**Developer Tools for Debugging SMP Code**

A multiprocessing platform is of little value unless the software applications that will run on it make effective use of multiprocessing. The Wind River Workbench development environment is based on the Eclipse framework and integrates a broad range of capabilities, as shown in Figure 4. The Workbench...
Development environment and associated tools offer extensive capabilities to meet the challenges of multiprocessing device software development.

Development Challenges and Solutions

Connecting with and Controlling Multiple Processors

A multiprocessing situation requires the ability to physically control multiple processors or processor cores. Wind River Workbench includes a target manager to coordinate multiple processor systems. In order to be useful, debuggers and other developer tools must be able to connect to a processor of interest. In many multiple processor systems, there may be processors that are not on the same network as the developer’s desktop computer, but can be reached through an intermediate processor that does share the network, then the proxy included in the Workbench Debugger can make the connection.

When using on-chip debugging, a single Wind River ICE can be used to control multiple processors or processor cores. Many other on-chip debugging solutions require a separate debugging hardware unit for each processor or core, significantly increasing cost and complexity. Wind River’s support for multiprocessor on-chip debugging also includes synchronized run-control across multiple processors when they are supported by the capabilities of the underlying hardware.

Identifying Overused and Underused Processors

New multiprocessing-specific enhancements have been included in Wind River System Viewer. For example, System Viewer can display the utilization or number of pending tasks for each processor core over time. For SMP systems, the data for all processor core is integrated in a single, time-correlated display. In addition, the VxWorks commands can be used to determine the amount of CPU time used by each task, the amount of time spent at interrupt level and in the kernel, and the amount of time the CPU was idle.

Identifying and Diagnosing Race Conditions

Wind River System Viewer supports the visualization of user-defined events. Developers can identify event sequences that represent possible race conditions by logging appropriate events to a file, then processing the data in that file with a script or program.

Debugging Thread Interaction Across Multiple Processors

The Workbench Debugger works simultaneously on multiple contexts. It assigns each thread its own debug context that includes a source/assembly view, a register window, and a stack backtrace view. Each thread can also be controlled independently using thread-qualified breakpoints that further extend the developers’ ability to debug multiple processor interaction.

Debugging Multiple Processors Without Breakpoints

In AMP systems, it may not be possible to debug certain problems using breakpoints. Stopping one of several processors can cause the system to malfunction or change the system’s behavior so much that it hides the problem that’s being sought. Wind River Workbench Diagnostics enables dynamic interaction with a running application—adding diagnostic code, collecting data, and modifying program execution. Providing these capabilities without requiring the processor to be stopped, or the application code to be rebuilt, makes Workbench Diagnostics a much less intrusive solution for an AMP system than a traditional debugger.

Tracking Data Values Systemwide

Understanding the operation of a multiple processor system often depends on understanding the changes in the state of the system created by the interaction of multiple processors, and having the ability to track data values over time. Wind River Data Monitor (formerly StethoScope) is an easy way to track key data values in an SMP system, or the value of data shared between multiple processors in an AMP system. Workbench target agent and on-chip debugging support for processors with hardware breakpoint capability provide another way to track data values, by setting breakpoints that trigger on data accesses at a specified address.
Identifying the Best Functions to Speed Up
To increase performance by applying parallelism, begin with the parts of your program that consume the most CPU cycles. The Wind River development environment for Linux and VxWorks provides several ways to collect this information. The VxWorks Spy() capability and the top command available in a Linux environment, provide one alternative. Wind River Performance Profiler (formerly ProfileScope) and System Viewer make it easy to identify which functions in an application are consuming the most CPU cycles. In an SMP system, additional threading will help to increase system performance. In an AMP system, these are the functions that may benefit from decomposition or reassignment to a processor with a lighter load.

Uncovering Shared Resource Contention
The rich data collection capabilities of Wind River System Viewer can be used to log information that can later be analyzed to expose a range of resource contention–related issues.

Tracking Interprocessor Communication and Synchronization
In multiprocessor systems, the interaction of processors is at least as important as what happens on any single processor. Spin locks are commonly used to provide synchronization in multiprocessing systems. Spin locks will be supported with VxWorks SMP, and spin-lock activity will be trackable in System Viewer visualization. System Viewer instrumentation of TIPC messages also provides insight into the interaction between processors.

Displaying Static or Dynamic Routing of Interrupts to Processors
The assignment of hardware resources, including interrupts, to specific processors is an important multiprocessing design decision. The ability to query the state of VxWorks kernel objects will allow developers to see how interrupts are assigned to processors. This will apply to static and dynamic assignment.

Technical Specifications

Supported Platforms
VxWorks 6.6-based platforms:
- Wind River General Purpose Platform 3.6
- Wind River Platform for Automotive Devices 3.6
- Wind River Platform for Consumer Devices 3.6
- Wind River Platform for Industrial Devices 3.6
- Wind River Platform for Network Equipment 3.6

Supported Multicore Processors and Boards
- ARM11 MPCore (ARMv6)
- Broadcom BCM1480
- Cavium OCTEON CN38XX
- Freescale MPC8641D, MPC8572
- Dual-Core Intel Xeon processor LV
- Intel Core Duo T2400
- Raza XLR 732

VxWorks Platform Components

Develop: Wind River Workbench 3.0
- Eclipse
  - Eclipse platform
  - C/C++ Development Tooling
  - Target Management/Remote System Explorer
  - Device Debugging
- Project System and Build System
- Editor and Source Code Analyzer
- Index-based global text search-and-replace
- Wind River Compiler and Wind River GNU Compiler
- Debugger and target debug agent
- VxWorks Simulator
- Host Shell and Kernel Shell
- VxWorks Kernel Configurator
- Run-time Analysis Tools
  - System Viewer
  - Performance Profiler (formerly ProfileScope)
  - Memory Analyzer (formerly MemScope)
  - Data Monitor (formerly StethoScope)
  - Code Coverage Analyzer (formerly CoverageScope)
  - Function Tracer (formerly TraceScope)

Run: Wind River VxWorks 6.6
Included in all VxWorks platforms:
- Backward-compatibility with VxWorks 5.5 and all previous versions of VxWorks 6.x
- Memory protection
- Shared libraries and shared memory, distributed shared memory
- Message channels and TIPC
- Error management and power management frameworks
- Processor Abstraction Layer
- Operating system scalability
- Certified POSIX 1003.13-2003 PSE52 conformance, enhanced compliance to PSE 53, PSE 54, and 1003.1 standards
- Networking: TCP, UDP, IPv4/IPv6, PPP
- Routing: Policy-based routing, ECMP
- File systems: dosFs, TrueFFS, Highly Reliable File System (HRFS)
- VxMP
- TIPC

Included in VxWorks industry-specific platforms (see documentation for exact contents of each platform):
- Networking: Mobile IPv4/6, VRRP, MPLS Dataplane
- IPsec and IKE
- Crypto
- Security Libraries
- SSL
- SSH
- RADIUS and DIAMETER Client
- Firewall
- NAT
- OPC
- DCOM
- CAN
- CAN, Web, MiBway
- SNMP
- Learning Bridge
- Media Library
- Web Services

Manage: Wind River Device Management (Optional Add-Ons)
- Wind River Field Diagnostics
- Wind River Lab Diagnostics
Architectures, Hosts, and Board Support Packages

Supported Target Architectures
- ARM MPCore
- BCM SB1a (1480, 1455, 1280, 1255, 1155)
- Cavium OCTEON CN3XXX
- Intel Xeon LV, Core
- PowerPC 8572, 8641d
- Raza XLR 732

Supported Hosts
- Windows XP Professional, Service Pack 2
- Windows Vista (Business and Enterprise)
- Red Hat Enterprise Linux 4, Update 5
- Red Hat Enterprise Linux 5*
- Red Hat Fedora Core 7
- SUSE Desktop Linux 10, Service Pack 1
- SUSE Linux/openSUSE** 10.2
- Solaris 9***
- Solaris 10

* Both 32-bit and 64-bit versions are supported. For 64-bit version, only x86-64 is supported. IA-64 (original Intel Itanium architecture) is not supported.
** SUSE Linux has been renamed to openSUSE.
*** GTK only (Solaris 9, update 9/05)

Board Support Packages
VxWorks 6.6 SMP supports a wide variety of board support packages on the target architectures listed previously. For a complete list of available BSPs, please visit the Board Support Packages section of the Wind River website at www.windriver.com/products/bsp_web/index.html.

Partner Ecosystem
Wind River’s world-class partner ecosystem assures tight integration between our core technologies and those of the premier hardware and software companies we’ve chosen to complement our solutions. Our partners help extend the capabilities of Wind River’s development and run-time platforms by offering out-of-the-box integration and support for key technologies in the fast-moving consumer market. Our customer support team is trained to troubleshoot partner technologies in use with Wind River products, making ours the most comprehensive and best supported partner ecosystem in the DSO industry.

Our hardware partners include:
- ARM
- Broadcom
- Cavium Networks
- Freescale
- Intel
- MIPS
- Raza

Professional Services
Successful completion of a multiprocessing device development effort involves navigating a series of options and developing additional skills and experience. Wind River Professional Services can assist with this process in numerous ways. For example, Professional Services now enables early access to technology currently being developed by Wind River Engineering. Engineering projects include support for a broad range of hardware and software configurations in order to address the needs of many different users. Professional Services may be used to deliver a new capability—in the specific configuration needed on a particular project—well in advance of the scheduled date for a Wind River platform release.

Wind River Professional Services helps customers invest their energy in their applications by enabling them to use Wind River platforms on processors and boards that are not supported by Wind River off the shelf. This accelerates projects, increases project team focus, and is especially helpful in light of the added complexity of multiprocessing hardware support. Often, existing software applications are not ready for use in a multiprocessing environment. For example, software developed and tested in a single-processor environment must typically be modified before it will execute correctly in a multiprocessing system. Even when software runs correctly, changes must often be made before the software will use multiple processors in a way that increases performance. Whether software for a project comes from previous internal development, open source, or a commercial provider, Wind River Professional Services can prepare it to run correctly so it will enhance your multiprocessing system’s performance.

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 your design. As part of our comprehensive DSO solution, Wind River offers industry-specific services practices, 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 our clients’ success.

Backed by our commercial-grade project methodology, Wind River Professional Services include the following:
- Requirements discovery and definition
- BSP and driver optimization
- Software system and middleware integration
- Application and infrastructure development
- Hardware and FPGA design for prototyping or market-ready systems

Typical projects range from two to four man-weeks for driver and BSP implementation, to one man-month to one man-year for hardware design or extensions to an existing software solution, to multi-man-year programs that bring customer concepts to reality through design, creation, and system test and verification.

Installation and Orientation Service
Proper installation and orientation of your VxWorks product means you won’t waste time solving easily avoidable problems before you can begin your next development project. Wind River offers an Installation and Orientation Service to ensure your project starts on time and without hassle by delivering the following:
- Onsite installation: Guided install on your hardware and host platform, along with a sample build process, demonstrations, and examples of customizations
- Hands-on orientation: Architecture, development file system, adding open source packages, porting drivers, and addressing design issues
- Advice: Introduction to Wind River support channels and processes, additional services, project review, and consultation
The Wind River Installation and Orientation Service will expedite your path to productivity, allowing you to rest assured that we have eliminated a common source of user error, and help you realize all of the platform’s potential.

**Education Services**

Education is fundamentally connected not only to individual performance, but also to the success of a project or 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 fits 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 team, improved efficiency in the processes they employ, and faster adoption of the technology you have purchased. The Personalized Learning Program 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 the Personalized Learning Program.

**Public Courses**

Wind River’s public courses are scheduled for your geographical convenience. They are conducted over one to five days, using a mixed lecture and interactive lab classroom format that leverages the experience of Wind River instructors and other course participants. Courses provide a fast, cost-effective way for students to become more productive in Wind River technology.

Benefits of public courses include the following:

- A conceptual introduction that orients students to the subject matter
- A selective examination of the details, focusing on the most commonly used areas, or on areas with which users tend to be least familiar
- Personal guidance and hands-on application of individual tools and course concepts
- The chance to grasp device software concepts, as well as the fundamental issues involved in real-time design
- The knowledge needed to develop device drivers, perform hardware porting, or develop applications
- Answers to specific questions about topics addressed in the course

Please consult your local Wind River sales representative for course schedules and fees.

**Onsite Education**

If you have a large project team or a number of new users, you may benefit from custom onsite education. Instructors will consult with you and, based on the workshop series curriculum, determine which topics should be included and emphasized. This type of education offers an opportunity for one-on-one discussions with our instructors about your specific project needs, technical requirements, and challenges—all in the comfort of your own office.

Advantages of onsite education:

- Your entire team gains a common knowledge base
- Onsite education helps ensure that knowledge and skills will transfer from the classroom to your workplace
- Use of your location saves employees travel expenses and time away from the office

Consult your local Wind River sales representative for further information about onsite education.

**Support Services**

Wind River Customer Support, a Support Center Practices (SCP)—certified organization, provides support for all Wind River VxWorks platforms. Your subscription to VxWorks SMP 6.6 Devices includes full maintenance and support, delivered through Wind River’s Online Support (OLS) website and our worldwide technical support team. While under subscription, customers receive both maintenance updates and major upgrades.

**Support for VxWorks 6.6 SMP**

Visit Wind River Online Support at www.windriver.com/support 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.

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.

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, Wind River 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.


Customers with a valid support or subscription agreement are eligible for all updates and major upgrades to Platform for Consumer Devices, VxWorks...
Edition 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.

If you cannot find the information you need through Online Support, please contact our global support team for access to the industry’s most knowledgeable and experienced support staff.

**North America, South America, and Asia/Pacific**

support@windriver.com
Toll-free tel.: 800-872-4977 (800-USA-4WRS)
Tel.: 510-748-4100
Fax: 510-749-2164
Hours: 6:00 a.m.–5:00 p.m. (Pacific time)

**Japan**

support-jp@windriver.com
Tel.: +81 3 5778 6001
Fax: +81 3 5778 6003
Hours: 9:00 a.m.–5:30 p.m. (local time)

**Europe, the Middle East, and Africa**

support-ec@windriver.com
Toll-free tel.: +800 4977 4977
France tel.: +33 1 64 86 66 66
France fax: +33 1 64 86 66 10
Germany tel.: +49 899 624 45 444
Germany fax: +49 899 624 45 999
Italy tel.: +39 011 2448 411
Italy fax: +39 011 2448 499
Middle East Region tel.: +972 9741 9561
Middle East Region fax: +972 9746 0867
Nordic tel.: +46 8 594 611 20
Nordic fax: +46 8 594 611 49
UK tel.: +44 1793 831 393
UK fax: +44 1793 831 808
Hours: 9:00 a.m.–6:00 p.m. (local time)