

# Upgrading VME applications with real-time Linux

By Vince Hauber and Andy Podnar

Many VME systems users are looking for ways to increase computational power while retaining their installed VME I/O investment. At the same time, many desire to move to an open source environment without sacrificing the deterministic performance of their current embedded Real-Time Operating System (RTOS). A choice of real-time Linux solutions is available to meet both of these goals.

VME-based systems are widely deployed and remain the workhorse for a large number of time-critical commercial and military applications. With so many installed systems, an ongoing challenge is deciding how to leverage new technologies for benefits such as improved performance and lower total cost of ownership.

A wide range of commercial hardware and software products is currently available to upgrade existing VME applications. Real-time Linux operating systems and tools run natively on COTS VME processor cards and also in hybrid VME systems such as a motherboard-based PCI platform connected to an external VME I/O chassis. Linux drivers are also available for many legacy VME I/O cards.

## Why Linux and open source?

In recent years, industry providers have come to recognize and utilize the advantages of open source solutions. Established open source benefits include improved functionality, compatibility, reliability, time to market, and lower cost of ownership. The Linux operating system has been at the forefront of the open source movement across all technical markets. Platform vendors now offer deterministic versions of Linux with performance approaching traditional embedded operating systems such as VxWorks. Many VME-based applications currently running on a proprietary RTOS can be easily ported to a real-time Linux operating system without replacing any existing VME I/O hardware.

## Real-time Linux at a glance

Many time-critical VME applications require very deterministic or even guaranteed real-time response. In other words, their high-priority transactions, event processing, data acquisition, or simulation tasks must execute accurately and predictably every time with sub-millisecond latency. Failure to do so can result in

mission failure, system damage, or lost revenue. Because of these exacting requirements and the need for interoperability, more businesses are turning to real-time Linux solutions. Examples of commercial real-time Linux offerings are RTLinux from Wind River, LinuxLink from TimeSys, BlueCat Linux from LynuxWorks, Concurrent's RedHawk Linux, which is compatible with Red Hat Enterprise Linux 4, and Novell's SUSE Linux Enterprise Real Time for the SUSE Linux Enterprise 10 environment.

RedHawk Linux and SUSE Linux Enterprise Real Time fully support standard Linux APIs; therefore, most software that runs on Red Hat and SUSE Linux will run on these RTOSs without modification. These real-time distributions support the same level of POSIX compliance as standard Linux and also support POSIX real-time extensions not present in standard Linux. This greatly reduces the effort of porting existing VME applications from other POSIX-compliant operating systems.

Real-time Linux vendors have made a number of key enhancements to standard Linux to get the determinism needed for time-critical VME applications. These enhancements are designed to allow applications to meet their specific timing

needs within a POSIX framework, thus minimizing code changes when porting a VME application to Linux. A summary of real-time Linux enhancements can be found in Table 1.

Linux has been successfully deployed in a wide variety of real-time applications including hardware-in-the-loop engine controller simulation, man-in-the-loop flight training simulation, and avionics equipment testing. To meet critical response requirements, enhanced real-time Linux kernels routinely offer process dispatch latencies less than 100 microseconds. Concurrent's RedHawk and Wind River's RTLinux offer guaranteed worst-case process dispatch latencies including a maximum interrupt response time of approximately 30 microseconds.

## VME hardware upgrade alternatives

VME systems can be upgraded with additional processing power and real-time Linux in one of two ways – a native system or a hybrid system approach. In a native implementation, the real-time Linux operating system runs directly on a VME CPU board collocated in a chassis with its VME I/O. The alternative is a hybrid configuration that utilizes a VME I/O bridge from a motherboard-based PCI-X or PCI Express (PCIe) platform.

Real-time Linux feature	VME application advantage
Processor shielding	Isolates a CPU from nonessential tasks. Allows a CPU to be dedicated to a time-critical task such as responding to specific VME I/O interrupts.
Kernel preemption	Provides a guaranteed, worst-case process dispatch latency time independent of the number of executing processes.
Priority inheritance	Prevents priority inversion. A low-priority process that shares resources with a higher-priority process will not delay the execution of the higher-priority process.
Frequency-based scheduling	Allows processes to be run at a periodic rate in applications that require predictable, cyclic execution.

Table 1

**Native VME**

The native VME alternative is usually the best choice for embedded and/or rugged applications. VME applications can be upgraded by replacing an antiquated VME processor board and its operating system with a VME dual-socket or dual-core SBC running real-time Linux. The benefit of this approach is that it may require only a replacement of a single VME card within the existing chassis.

In addition to porting the application code to the Linux operating system, Linux drivers for each VME I/O device must be integrated. Open source Linux drivers for many I/O devices are readily available from I/O board vendors.

A wide selection of COTS CPU boards is available for implementing native VME upgrades. The most powerful of the latest CPU boards are x86-based. SBCs featuring dual-core Intel Xeon processors and low-power Intel Core 2 Duo chips are currently available from manufacturers such as GE Fanuc, Curtiss-Wright, Mercury, Concurrent Technologies, and Thales. In addition, new VXS (VITA 41)

high-performance switched fabric technology allows multiple CPU cards to be linked to support applications that require multiple computational resources.

**The hybrid alternative**

Some applications, such as simulators, require very high computational power and a level of real-time determinism that can only be achieved by a symmetric multiprocessing system (all CPUs directly addressing a common main memory). A hybrid VME solution is the best choice when the application requires more than four separate processor cores. Server-class Intel and AMD motherboard MP platforms, such as those available from Dell, HP, Supermicro, and Tyan, currently offer more symmetric multiprocessing cores than native VME CPU cards. A hybrid VME solution, as depicted in Figure 1, is often the best choice when CPU performance requirements are very high.

Access to a VME I/O chassis from a motherboard-based server platform is achieved by a PCI-to-VME bridge. A motherboard platform running real-time Linux serves as an application processor

and performs VME I/O via calls to Linux VME device drivers. A cabled VME bridge comprises two cards, one installed in a PCI-X or PCIe slot on the application server and the other installed in a VME slot in the I/O chassis. The major advantage of this approach is that it can leverage the industry's most powerful Xeon and Opteron quad-socket and quad-core servers as the application engine. This high level of compute power is not available in VME SBCs.

VME bridges such as the GE Fanuc/SBS model 618-3 can provide VME system controller functionality and can also act as a VME bus master, alleviating the need for an additional board to provide these services. Also, the 618-3 can act as a VME bus slave so that other VME bus masters can access the host server's PCI bus. Linux drivers are currently available for the 618-3. For systems requiring VME64 transfers, GE Fanuc/SBS also offers the model 810, which has double the throughput rate of the 618-3.

VME I/O bridge configurations have been found to meet most application performance requirements; however, bridge solutions have some identified drawbacks. For example, a latency of one-half to one microsecond may be added to PIO operations across the bridge. Typically, a PCIe-to-VME bridge performs better than a PCI-X-to-VME bridge by reducing the number of interrupts that must be handled across the bus link.

**Linux debugging tools for VME applications**

To efficiently utilize the real-time Linux enhancements described earlier, robust debugging and analysis tools are an essential asset. Time-critical VME applications require debuggers that can handle the complexities of multiple processors, multitask interaction, and multithreading. A wide range of open source Linux debugging tools is available, including the GNU debuggers GDB and DDD, the Linux Trace Toolkit, and SystemTap. In addition to these open tools, commercial, graphical tools such as TotalView from TotalView Technologies and NightStar from Concurrent are available to handle the more complex debugging tasks.

An important characteristic of a good VME debugging tool is nonintrusiveness. A nonintrusive Linux debugger allows a task to run at or near application speed,

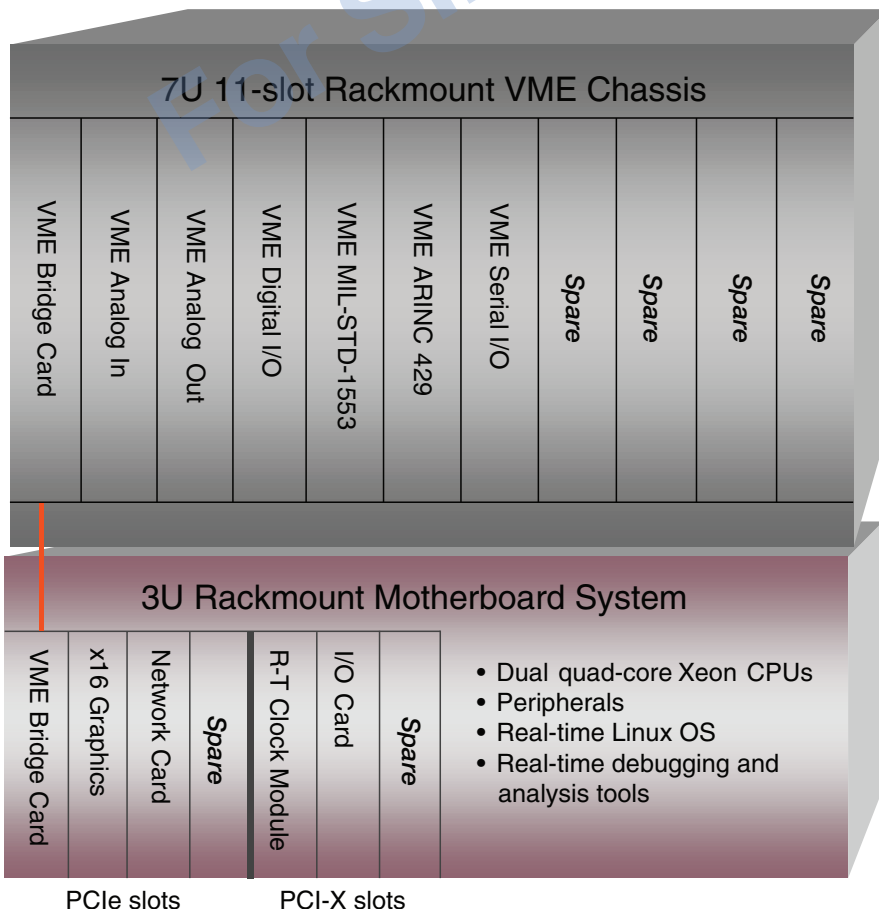


Figure 1

so that the application's behavior is not affected by the debugging process. This is important because debugger intrusion can alter the timing of asynchronous events. This can cause the symptoms of a software problem to change or even disappear during debugging.

Another important debugging feature is real-time kernel event tracing (Figure 2). It is often very valuable to view a runtime trace of both user application and kernel events while an application is running on multiple processors or multiple network-connected VME processor cards. Kernel events include system calls, interrupts, exceptions, and the processor context switches. The ability to simultaneously view both user and kernel events allows

a developer to uncover difficult-to-find, intermittent bugs.

### Upgrade both the CPU and operating system

It is possible to upgrade an existing VME application by installing a new higher-performance VME SBC running a real-time Linux operating system or by bridging to VME I/O from a real-time Linux PCI-based platform. Most existing VME real-time applications can be run by a real-time Linux operating system such as RedHawk Linux or SUSE Linux Enterprise Real Time. These real-time Linux distributions have enhanced kernels that are deterministic and responsive enough to address time-critical requirements. **CS**

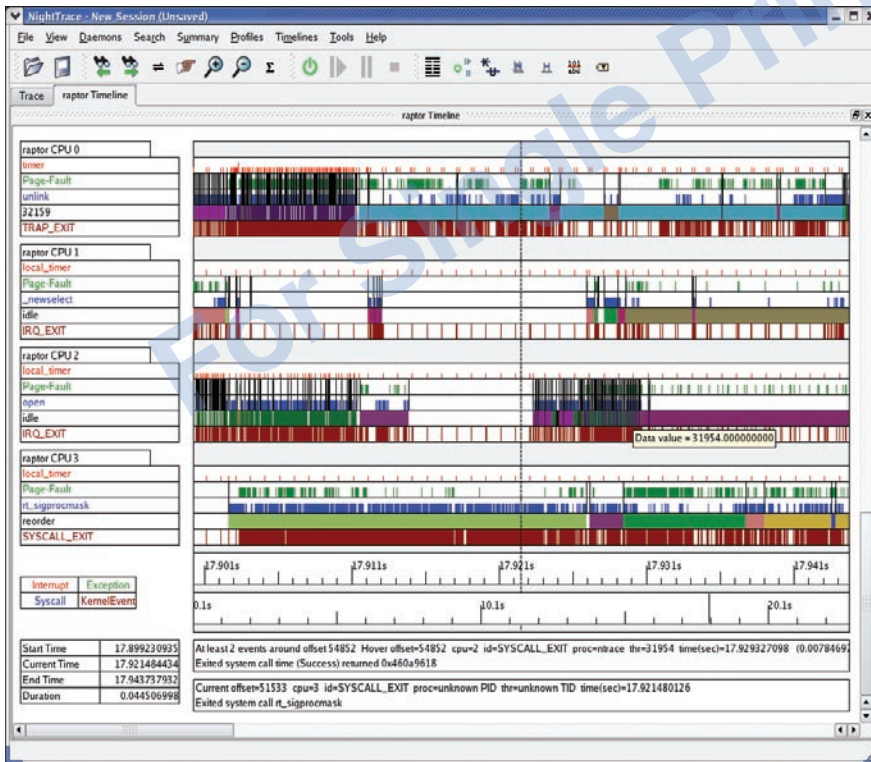


Figure 2



**Vince Hauber**, a senior product manager with Concurrent Computer Corporation, has more than 40 years of experience in real-time system software and platform solutions. His current responsibilities include product management for Concurrent's NightStar advanced Linux debugging and analysis tools, technical sales support, and online business applications development.

To learn more, contact Vince at:

**Concurrent Computer Corporation**  
2881 Gateway Drive  
Pompano Beach, FL 33069  
954-973-5095  
vince.hauber@ccur.com  
www.ccur.com



**Andy Podnar**, a senior field applications engineer with Concurrent Computer Corporation, has more than 20 years of experience in the embedded computing industry. His career includes the design and test of full flight trainers at Link Flight Simulation, avionics integration and test at Honeywell, and applications engineering for Xycom, dSPACE, Applied Microsystems, Motorola Metrowerks, and Concurrent.

To learn more, contact Andy at:

**Concurrent Computer Corporation**  
2881 Gateway Drive  
Pompano Beach, FL 33069  
954-973-5095  
andy.podnar@ccur.com  
www.ccur.com