

The logo for WIND, featuring the word "WIND" in a bold, sans-serif font with a trademark symbol, set against a white rectangular background.

WIND™

Enabling the Migration to Software-Defined Platforms for Critical Infrastructure

Paul Parkinson

Director, Field Engineering, Aerospace & Defense, EMEA

WHEN IT MATTERS, IT RUNS ON WIND RIVER

EXECUTIVE SUMMARY

Embedded systems used in critical infrastructure are currently undergoing a dramatic evolution. Historically, they had varying levels of network connectivity, performed fixed functions, and may have undergone manual software updates in the field as part of long in-service lifetimes. Now the advent of ubiquitous network connectivity has accelerated innovation of embedded systems at the edge of the network, known as edge devices. This has driven demand for enhanced application capabilities that perform traditional automation and control functions while adding greater intelligence and the ability to support more dynamic behavior.

There is a growing need across multiple sectors for devices to be able to support the greater intelligence required to transition from automated to autonomous systems. This is driving technical requirements for open standards-based software-defined architectures to enable consolidation of multiple applications, including those at different levels of criticality, onto common computing platforms. This approach enables application migration, portability, and interoperability, to avoid being locked into proprietary architectures.

[Wind River® Helix™ Virtualization Platform](#) addresses these common requirements by providing a flexible virtualization platform that supports open standards-based software-defined architectures. It is based on proven software technologies, enabling it to meet the safety certification requirements of aerospace, automotive, defense, industrial, and medical markets.

TABLE OF CONTENTS

| | |
|---|---|
| Executive Summary | 2 |
| Evolution of Embedded Systems from Automation Toward Autonomy. | 3 |
| Common Market Requirements Leading to a Converged Embedded Virtualization Platform | 4 |
| Evolution of Embedded Virtualization for Critical Infrastructure | 4 |
| Development Tools to Enable Digital Transformation | 7 |
| Summary | 8 |

EVOLUTION OF EMBEDDED SYSTEMS FROM AUTOMATION TOWARD AUTONOMY

Over the last decade, embedded systems have undergone a continual evolution in terms of performance, connectivity, and capability. Historically, at one end of the capability continuum, some embedded devices have performed fixed functions and had long in-service deployment lifetimes. They may have rarely undergone system upgrades that added new functionality or deployed security updates to address the latest security vulnerabilities. Systems without any network connectivity may have needed to be manually updated in the field—a potentially time-consuming, error-prone, and expensive process, depending on the number of devices deployed and level of difficulty of physical access.

Processor performance has increased dramatically for many years, per [Moore's Law](#), and the cost of network connectivity has continued to fall, resulting in a dramatic increase in edge device connectivity. This has accelerated innovation, enabling new application functionality to be deployed more rapidly to edge devices using secure communications sessions over network infrastructure. This foundation is now enabling the development and deployment of the next generation of intelligent devices and the transition from automation to autonomy.

Aerospace and Defense Sector

In the aerospace sector, there has been a very significant trend in avionics design, with a marked shift away from federated avionics architectures using fixed-function line-replaceable units (LRU) to integrated modular avionics (IMA) architectures employing common computing platforms hosting multiple applications at different levels of safety criticality. This transition has been driven in commercial and defense aerospace by a common requirement to significantly reduce the overall size, weight, and power (SWaP) requirements of avionics systems, particularly as the number of federated LRUs have increased to support new avionics functions. The adoption of IMA has resulted in significant reduction in weight, enabling aircraft to fly with reduced fuel load, or with more passengers or payload, respectively. The use of standards-based IMA software architectures to enable interoperability and integration has also had a significant positive impact on both commercial and military programs, reducing the risk of design lock-in, driving innovation, and reducing whole-life costs.

Automotive Sector

The automotive sector has its own set of challenges, due to the pressures of an extremely competitive market and the increasing rate of innovation. This is being driven by market demand for enhanced in-vehicle infotainment systems (IVI), over-the-air (OTA) updates, advanced driver assist systems (ADAS), and, more recently, the goal of autonomous driving. This has resulted in a dramatic increase in the number of processors used in automotive vehicles in the last decade. The deployment of autonomous driving systems will require further significant increases in on-board computing power, including general-purpose graphics processing units (GPGPUs), in order to support artificial intelligence (AI) and machine learning (ML) applications. These automotive systems, along with the associated cabling, have increased the weight of vehicles, which has had a negative effect on vehicle performance, fuel economy, CO₂ emissions, and cost (just as in the aerospace market). As a result, major automotive manufacturers are now using common embedded computing platforms to consolidate applications and reduce SWaP.

Industrial Sector

In the industrial sector, companies are embracing digital transformation, using the Industrial Internet of Things (IIoT) and Industry 4.0 to address business challenges and remain competitive. Smart factories, energy systems, and other critical infrastructure are adopting real-time monitoring capabilities and analytics to measure operational efficiency and are now able to use analytics and predictive maintenance to minimize downtime due to system failures. This digital transformation is due to the addition of network-enabled monitoring capabilities to real-time control systems. Companies are now looking to achieve further efficiencies through workload consolidation on common computing platforms and remote upgrade capabilities.

Medical Sector

The medical sector is currently undergoing a revolution in healthcare, as new technologies are enabling advanced imaging systems, surgical robots, and other critical medical devices. Patient safety is life critical, so these systems have to operate securely and reliably. Current cybersecurity regulations mean that medical device manufacturers must be able to easily update their systems over the air; manual updates would not be cost effective due to the large number of devices and distributed install base.

COMMON MARKET REQUIREMENTS LEADING TO A CONVERGED EMBEDDED VIRTUALIZATION PLATFORM

Although each vertical market faces unique challenges and requirements, at an abstract level there are now similar and even common fundamental capabilities needed in embedded software platforms. In particular, all vertical markets need a safe, secure, and reliable platform that share the same high-level requirements:

- **Mixed-criticality:** A common requirement for a consolidation platform is to host both critical real-time control applications and general-purpose applications. It must isolate safety components and reduce their dependencies on the rest of the platform in order to ensure that the safety-certification costs of the overall platform remain affordable.
- **Open standards:** There is an increasing requirement for each sector to adopt open standards to enable application migration and portability as well as competition, and to prevent design lock-in. The commercial aerospace sector has embraced the [ARINC 653](#) standard for open avionics architectures, while the U.S. government has adopted the [Modular Open Systems Approach \(MOSA\)](#). The [Future Airborne Capability Environment \(FACE™\)](#) technical standard provides an open avionics architecture for military avionics and uses commercial open standards (ARINC 653, [POSIX®](#)). The automotive sector is adopting [Adaptive AUTOSAR](#) (Automotive Open System Architecture) to enable greater adoption of commercial off-the-shelf (COTS) hardware and software and to provide a flexible and scalable framework. In the industrial sector, the [Open Process Automation™ Forum \(OPAF\)](#) is focused on developing a secure, open standards-based, interoperable process control architecture.
- **Reuse and scalability:** Intellectual property and previously developed (even previously certified) applications should be reusable on new embedded computing platforms. This includes using the performance and scalability provided by multi-core processor architectures with hardware virtualization support, while managing and isolating the application developer from the complexity of the underlying hardware architecture. It also requires the ability to deploy new applications on the platform to deliver additional capabilities.

- **Security:** There is an increasing need to ensure security throughout every phase of the operations and lifecycle of a device. This includes design, production, commissioning, deployment, and operation as well as decommissioning at the end of the lifecycle. During deployment, the device needs to operate securely in different phases: it needs to perform secure initialization to verify the integrity of the deployed software, to ensure that it is has not become corrupted or tampered with; it needs to provide secure communications during operations and resilience against cyber-attacks; and it needs to securely store sensitive data in process and data at rest, including when powered down.
- **Support for modern development methods:** Manufacturers in each of the vertical markets are under pressure to deliver new, high-quality software capabilities within shorter time-scales. Many companies now use agile development processes to accelerate change while employing continuous integration and continuous delivery approaches through automation to dramatically reduce their integration and delivery timescale, as part of a DevOps culture to increase responsiveness.

EVOLUTION OF EMBEDDED VIRTUALIZATION FOR CRITICAL INFRASTRUCTURE

During the last decade, advances in processor technologies have had a significant impact on deployed architectures; in particular, the evolution and maturity of processor hardware virtualization support has enabled enterprise and cloud computing platforms to host virtualized applications efficiently and at scale. Hardware virtualization support has also been successfully deployed in embedded systems, particularly in safety-critical avionics systems using [VxWorks® 653 Multi-Core Edition](#), where multiple virtualized applications can be hosted on a common avionics computing platform. This convergence across multiple sectors of hardware virtualization, multi-core processor architecture, and open standards-based software-defined architecture provides the opportunity for a common embedded virtualization platform for critical infrastructure. See Figure 1.

Wind River has addressed this market need by leveraging decades of experience and continuing industry [leadership](#) in safety-critical, security, and embedded hypervisor software technologies to develop Helix Platform.

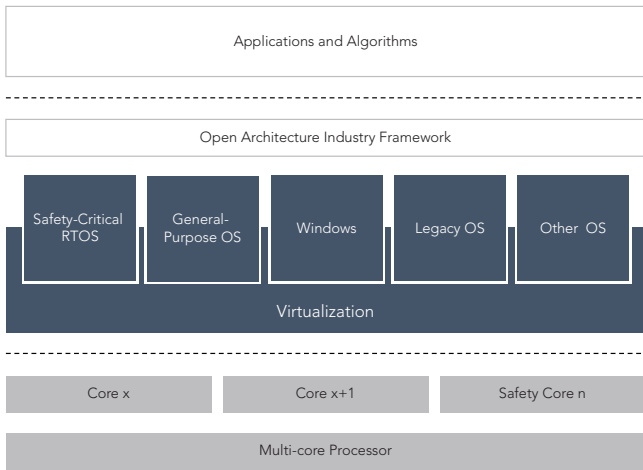


Figure 1. Software-defined architecture for common computing platforms

Hypervisors and Virtual Machines

Helix Platform uses a Type 1 hypervisor (also known as a bare metal hypervisor), as shown in Figure 2, which runs directly on the processor; provides near-native real-time performance using direct interrupts; and provides the scalability, determinism, and small footprint suitable for safety certification. This approach contrasts with Type 2 hypervisors, which focus on abstraction from the underlying physical environment and provide a best-effort performance that is not suitable for hard real-time applications. Additionally, their footprint is too large to undergo safety certification. The Type 1 hypervisor approach provides efficient scalability on multi-core processor architectures, since it can scale to a higher number of cores compared to traditional monolithic and micro-kernel architectures.

The hypervisor uses the processor’s dedicated hypervisor privilege level and full hardware virtualization support, enabling 32-bit and 64-bit guest operating systems and associated applications to run at separate privilege levels inside virtualized environments (commonly known as virtual machines). This includes [VxWorks](#), [Wind River Linux](#), Microsoft® Windows® (64 bit), Android, bare metal, and third-party operating systems, including other flavors of Linux.

The hypervisor also uses the processor’s memory management unit (MMU) to enforce isolation of individual virtual machines. This prevents a guest operating system (OS) and its associated application from making any unauthorized programmed I/O accesses to another virtual machine or privileged system resource;

any attempted unauthorized accesses are reported to the hypervisor. Many modern processors additionally provide direct memory access (DMA) engines to perform efficient transfers of blocks of data between source and destination memory locations, for example between system memory and external I/O devices such as network interfaces. The hypervisor also uses another processor hardware resource, the IOMMU, to ensure that the guest operating system and applications only perform DMA transfers between authorized source and destination addresses. This means that Helix Platform is able to host a virtual machine containing a guest operating system that uses a third-party device driver of unknown pedigree, in the knowledge that an unauthorized DMA access will be detected and prevented by the IOMMU. This isolation capability provides benefits in terms of safety and security, and it supports the consolidation of multiple applications onto a

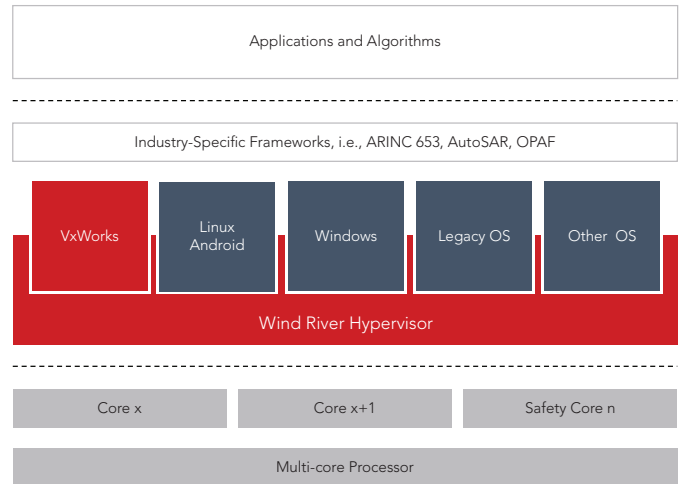


Figure 2. Helix Platform architecture

common embedded processing platform.

These hardware virtualization and isolation capabilities enable Helix Platform to host real-time and safety-critical VxWorks applications alongside embedded Linux applications and general-purpose operating systems, such as Microsoft Windows and other third-party and legacy operating systems. This enables previously developed software to be rehosted, preserving investment in existing intellectual property and serving as an asset bridge between existing deployed systems and next-generation systems, thereby reducing the technical risk when upgrading in-service systems.



Inter-partition Communication

A consolidation platform also needs to support communication between individual virtual machines, and Helix Platform provides a number of different inter-partition communication methods:

- Shared memory and VNICs (Virtual Network Interface Controllers) enable communication between VxWorks, Linux, and Windows guest operating systems as well as other third-party guest operating systems.
- Safe IPC is a controlled shared memory implementation to enable communication among multiple guest operating systems, potentially running at different levels of criticality.

Static and Dynamic Configuration

Helix Platform builds on the capabilities of Wind River safe and secure platforms, which have a world-class certification track record under multiple certification standards across different industry sectors. Helix Platform has also evolved the proven code base to support a broader range of use cases, including both static and dynamic behavior, as shown in Figure 3.



Figure 3. Static and dynamic configuration

Static configuration is used for systems that contain a safety-critical application, enabling system resources to be allocated at system configuration and build time and then used in a predictable and deterministic manner at runtime. The robust partitioning described earlier means that this configuration can support multiple safety-critical applications running in different partitions, or mixed-criticality systems comprising multiple applications at different integrity levels. The Independent Build, Link, and Load (IBLL) approach, pioneered by Wind River in VxWorks 653, enables applications to be configured, built, linked, and loaded independently. This enables individual applications to be updated independently through the platform lifecycle without impacting other components, enabling incremental certification and dramatically reducing whole-life costs for the platform. For more details on IBLL, see the Wind River

white paper, [Safety-Critical Software Development for Integrated Modular Avionics](#).

Dynamic configuration is used for systems that require greater flexibility in terms of configuration of applications and system resources, and that may need to dynamically change system configuration and runtime operation to handle an increase in service requirements or in response to the external environment or mode of operation. This approach enables a wide range of guest operating systems and applications to be hosted, including those that have less stringent requirements for determinism than a safety-critical guest operating system that uses a static configuration.

Scheduling

Helix Platform also supports two distinct types of scheduling in order to support a broad range of application use cases:

Priority preemptive scheduling: This scheduling type uses threads running in the hypervisor, with one hyper-thread per guest operating system per core. This approach enables an individual guest operating system to be scheduled on a priority-based, preemptive basis (as shown in Figure 4) without affecting the scheduling within an individual guest operating system, which uses its own local scheduling policy. The scheduling model enables applications with dynamic behavior and event-driven systems to be hosted on Helix Platform.

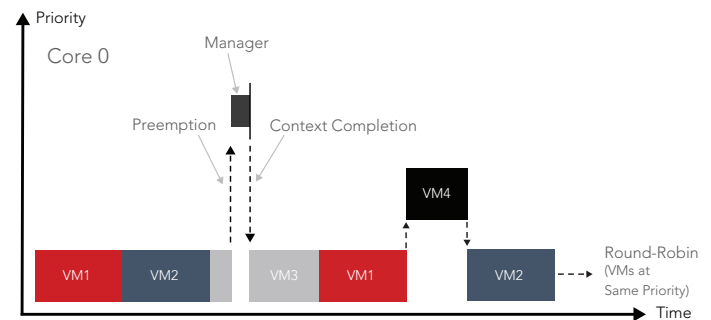


Figure 4. Priority preemptive scheduling

Frame scheduling: This form of scheduling uses a system schedule of multiple individual timeslots of fixed durations that define how long individual guest operating systems and their associated applications can execute. The frame scheduler can be used to ensure that a general-purpose application does not overrun its allocation time period and adversely impact the execution of a



safety-critical application hosted on the same platform (as shown in Figure 5). Helix Platform allows the schedules for individual cores to be synchronized. The frame scheduler also enables multiple schedules to be defined at system configuration and build time and allows the transition from one schedule to another through the use of a dedicated hypervisor call, known as a hypercall, by a trusted guest operating system partition. This support for multiple schedules enables use of different operational modes, e.g., initialization mode, operational mode, and maintenance and/or diagnostic mode. The frame scheduling model is widely used in safety-critical avionics systems (using an ARINC 653 scheduler), but this approach is also equally applicable to safety-critical systems in other market sectors needing to host general-purpose and safety-critical applications on the same computing platform.

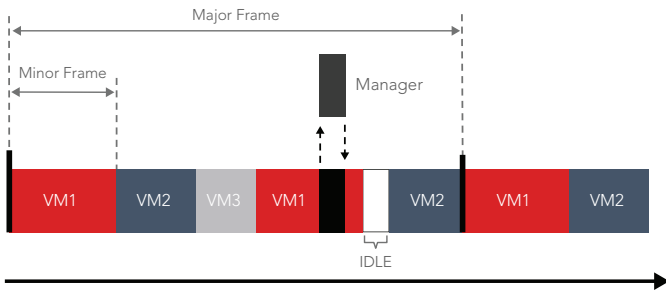


Figure 5. Frame scheduling

Health-Monitoring Framework

A system containing multiple heterogeneous virtual machines needs to be able to monitor, detect, and recover from hardware faults, guest operating system faults within individual virtual machines, and application failures. This requires that a health-monitoring framework be able to isolate faults and prevent failures from propagating. Although these requirements may appear to be straightforward conceptually, they are actually complex and require a sophisticated system-wide implementation to provide continued availability of the platform. Wind River has a proven track record in developing an ARINC 653 health-monitoring (HM) framework for safety-critical IMA applications and has used this expertise in the development of the health-monitoring framework for Helix Platform.

The HM framework handles events that can be either an alarm, representing a fault that needs attention, or a message, providing a notification that can be processed or recorded. The HM

framework uses a hierarchical, table-driven implementation, which enables HM events to be handled at the application level, virtual machine level, or embedded platform level (these are referred to as the process level, partition level, and module level in the aerospace sector). The HM framework also allows HM events to be handled at the level at which they occur or are passed to the next level; e.g., a fault within an individual virtual machine is contained, and the error handling can be routed from the virtual machine’s guest operating system to the hypervisor. This approach enables the system integrator to determine, at the time of system configuration, the course of action to take in the event of specific errors. The HM framework also provides support for cold and warm restarts at the virtual machine and embedded platform levels. This enables individual virtual machines to be restarted without interfering with other virtual machines.

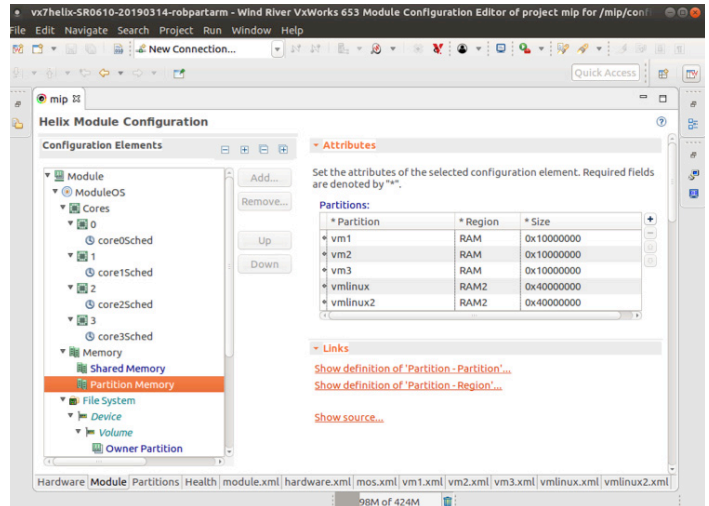


Figure 6. System configuration and application development using Wind River Workbench

DEVELOPMENT TOOLS TO ENABLE DIGITAL TRANSFORMATION

An embedded virtualization platform promises many benefits. To fulfill these promises, sophisticated development toolset support is needed to manage the complexity of system configuration and the building and development of applications for heterogeneous runtime environments.

Wind River has a proven track record in the development of integrated development environments, dynamic visualization tools, and simulation platforms for general-purpose and



safety-critical applications. This expertise has resulted in the latest Wind River Workbench (see Figure 6) development environment for Helix Platform, which provides graphical support for system definition and configuration, including allocation of virtual machines to processor cores, and definition of schedules. Wind River Workbench performs continuous validation of system configuration in the background, providing feedback to developers as they configure their system. It also automates many of the build and configuration steps, reducing the burden on the system integrator and application developer.

The patented Wind River System Viewer (see Figure 7) also provides graphical representation of the system, enabling the developer to view and understand the interaction between system events such as interrupts, guest operating system primitives, VxWorks application tasks, POSIX threads, or ARINC 653 processes.

Helix Platform also uses the modern [Clang/LLVM](#) compiler toolchain. This provides multiple benefits, including support for the latest C and C++ language standards and performance benefits, such as fast compilation times, low memory usage, and faster execution from increased code optimization.

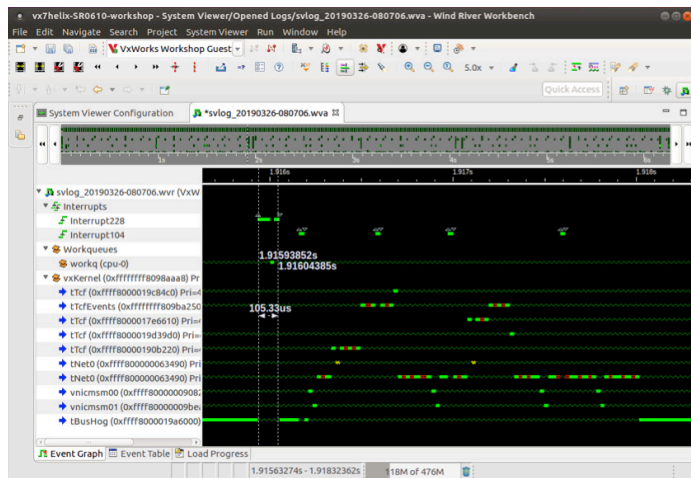


Figure 7. Dynamic visualization using Wind River System Viewer

Applications can be developed using Helix Platform on an embedded target platform or using [Wind River Simics](#). Simics breaks the dependency on hardware availability and enables virtual prototyping to be undertaken earlier in the development lifecycle, reducing costly rework in later stages. Simics also enables many test cases to be run in parallel and at scale, supporting the development of applications for Helix Platform using continuous integration/continuous delivery (CI/CD).

SUMMARY

Legacy embedded systems are facing significant challenges in terms of functionality, maintainability, and obsolescence. Future systems have demanding requirements in terms of software-defined architectures and open standards support. And critical infrastructure systems increasingly need to meet stringent safety certification requirements in their respective vertical markets.

Helix Platform addresses these challenges by providing an embedded virtualization platform that can act as an asset bridge, enabling the consolidation of legacy applications and previously developed intellectual property on a modern, scalable platform. It supports the development of new applications based on open standards using a diverse range of operating environments. It also provides flexibility through static and dynamic configuration options, enabling a broad range of use cases with mixed-criticality to be hosted and deployed using modern development methods and processes.

For more information on Helix Platform, visit www.windriver.com/products/helix-platform.

