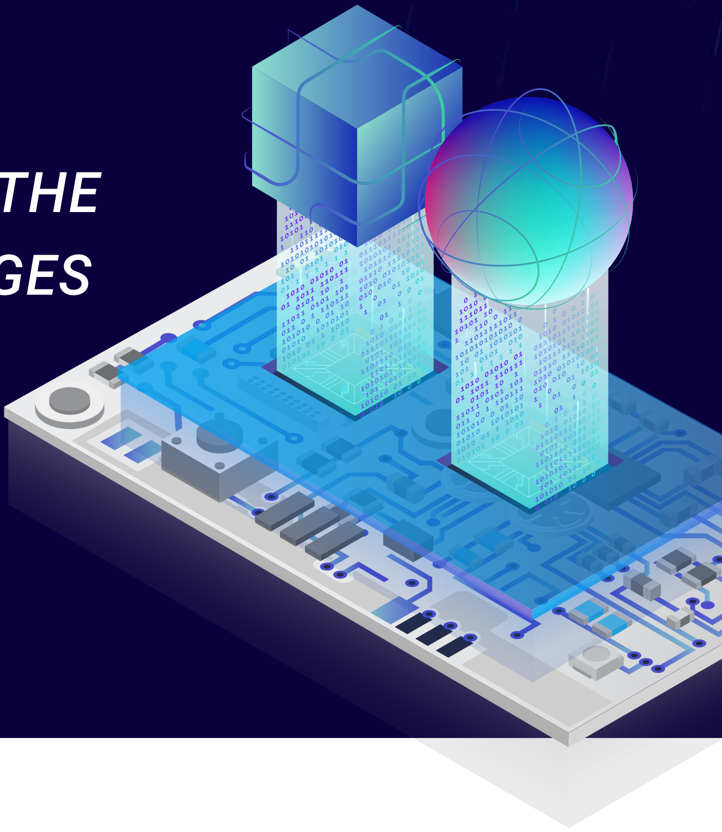


HOW HYPERVISORS REDUCE THE IMPACT OF SILICON SHORTAGES

WHITEPAPER

Two forces have thrown the embedded systems industry into turmoil – the ongoing silicon supply shortage and the lack of skilled personnel necessary to design and build these products.

While much of the silicon problem stems from manufacturers sourcing from the same dwindling supply of semiconductor chips, it is also a consequence of the success of embedded systems. The median demand for chips was up by as much as 17% between 2019 and 2021 but buyers were not seeing similar increases in supply, causing a major mismatch in pipeline.

While finding talented embedded software developers has long been a challenge, the Covid-19 pandemic exacerbated the problem with new projects drying up and manufacturers struggling to implement their planned Industry 4.0 roadmaps.

It is critical for manufacturers to understand how a coherent hypervisor strategy reduces vulnerability to uncertainties in the silicon supply chain and enables existing personnel to focus on innovation and rapid development of high-value projects.

01. DWINDLING SUPPLIES OF SILICON AND TALENT?

Recently, the chip shortage hit center stage in the United States Congress for the second time in less than a year. Repeating its call for $52 billion in funding and incentives for domestic semiconductor production, the Department of Commerce stated that “there is a significant, persistent mismatch in supply and demand for chips, and respondents did not see the problem going away in the next six months.”

With more than 80 percent of global semiconductor production occurring outside the US, this issue affects all industries and regions. The challenges are further complicated by the fact that more buyers want more electronic products that rely on a range of components with varying levels of availability. Industrial IoT products require microchips to run their display drivers and power management ICs while two-thirds of medical technology companies say they use semiconductors in at least 50 percent of their products.

Automakers are among the most impacted by the shortage as cars can have hundreds of microchips controlling everything from windows and door locks to infotainment and advanced driver-assistance systems (ADAS).

Last year, automotive OEMs saw a loss of 11.3 million units as a result of the chip shortage and IHS Markit recently downgraded their production forecast for 2022 due to factors including the sluggish recovery in semiconductor supply and new Covid-19 lockdown measures.

While covid is partly responsible for the chip shortage, it can’t be blamed for engineers’ concerns about the lack of relevant skills in their organizations. In a 2019 study, VDC Research noted that almost 25 percent of engineers working on embedded projects are concerned about their organization’s lack of skilled personnel, their difficulties specifying requirements and a lack of guidance through safety certifications.

This problem has many dimensions, from education to immigration policies to organizations’ hiring and career development practices. It is also largely due to a simple and obvious fact: success.

Just as every single embedded system requires silicon, it also requires skilled personnel: engineers, designers, developers, testers, writers and managers to shepherd it from inception to market, and to maintain it henceforth.

The greater the demand for embedded systems, the greater the demand for the people who build them.

Compounding the problem is the fact that the more complex the system, the greater the demands on workers’ competencies. In addition, the use of these systems in safety-critical environments is increasing requirements for safety-certifications. For example, building a safety-certified vehicle cockpit display is rather more demanding task than making an HVAC vent controller with a programmable logic controller (PLC).

02. CONSOLIDATION, FLEXIBILITY AND INNOVATION

The demand for chips and the skilled people who build with them will only continue to grow in the short term as well as post-Covid, requiring a reasonable strategy to reduce our reliance on these critical resources.

If we apply ourselves to the factors we can directly control, we come to three complementary tactics:

- **Consolidation:** implementing multiple software systems on the same system-on-chip (SoC), thereby reducing the overall number of SoCs needed in an embedded device.
- **Flexibility:** designing the embedded system to facilitate the substitution of functionally equivalent chips, should the preferred silicon become unavailable or overpriced.
- **Innovation:** Reducing developer friction (those processes that place a drag on development work), the costs of initial development and long-term ownership. This allows for the implementation of innovative, market-differentiating functionality.

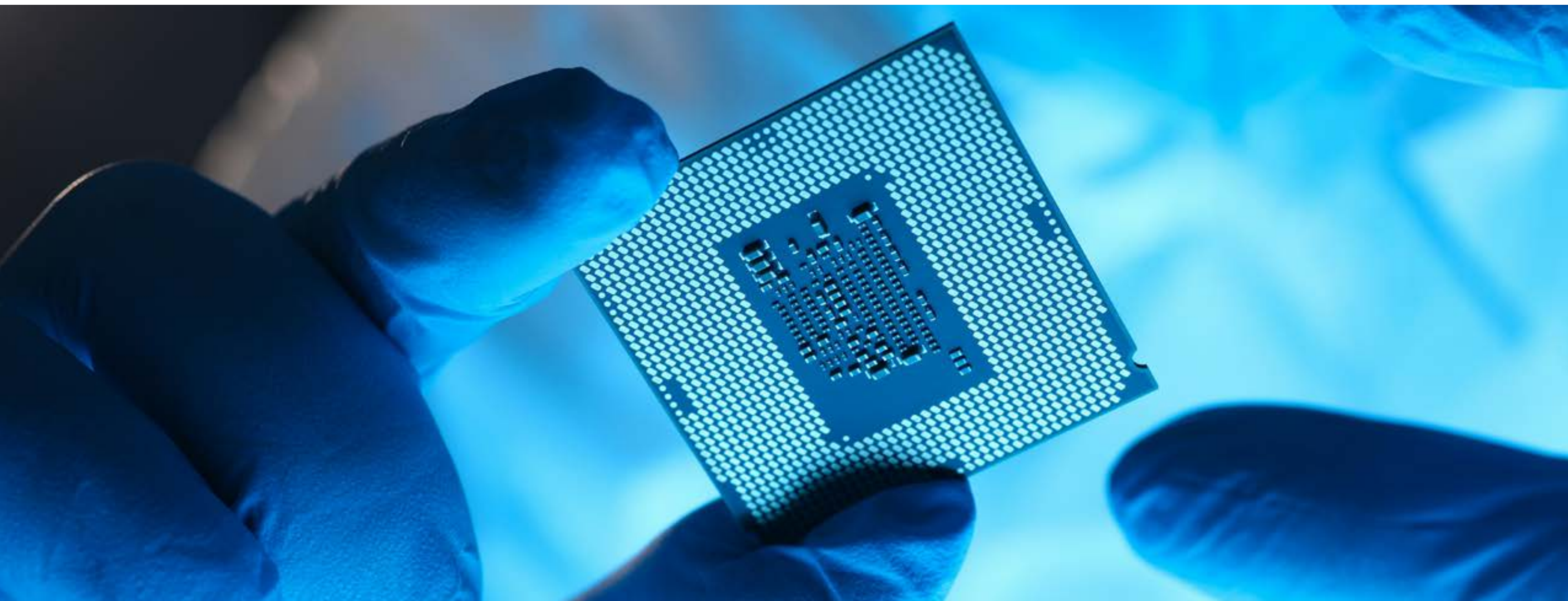
03. MOVING THE BABBAGE-LOVELACE DEMARCATION

Ever since Charles Babbage's Analytical Engine of 1837 and Ada Lovelace's "note G" of 1843, the world's first computer and the world's first computer program, a *de facto* division of responsibilities has persisted in computing.*

In current terms, Babbage designed the hardware and Lovelace wrote the software. To wit, most embedded engineering projects select the silicon; the software comes later, and is expected to adapt to the hardware.

A single strategy offers a path to consolidation, flexibility and focus. This strategy uses a hypervisor to abstract the hardware and move the Babbage-Lovelace demarcation between hardware and software from the actual hardware (or, more accurately, hardware and firmware) up to virtual machines.

* We are excluding rarified research environments and the work of people such as Alan Turing and Tommy Flowers, especially in the early days of electronic computing.



04. WHAT IS A HYPERVISOR?

A hypervisor, also called a virtual machine manager (VMM), is software that abstracts hardware to present virtual machines.

These virtual machines provide environments in which different operating systems (OSs) and their applications can run. An OS and its applications in a hypervisor virtual machine are known as a *guest*.

A hypervisor enables system designers to consolidate diverse guests with different reliability, safety and security requirements on a single SoC, as well as to offload a good deal of the work required to adapt a system to specific chips and even chip revisions onto the hypervisor design and configuration.

In 1974, Popek and Goldberg specified the three essential attributes of a hypervisor:

- **Equivalence:** a hypervisor's virtual machines present a duplicate of the underlying hardware, so that software running in the virtual machine runs as it would directly on the hardware. A guest in a virtual machine mustn't need to know that it is in a virtualized environment.
- **Safety:** the hypervisor controls the hardware, and virtual machines are isolated from the hypervisor and from each other, just as they would be if they were running on separate SoCs.
- **Performance:** software executing in a virtual machine must present no more than a minor decrease in speed compared to the same software running directly on the underlying hardware (bare metal).

Note that a hypervisor is not a machine simulator. Simulators don't let guests execute directly on the hardware, and typically offer performance five to 1000 times slower than hardware. They do not, therefore, meet the Popek/Goldberg requirement for performance.

A hypervisor manages its guests' access to hardware, but the guest executes directly on the hardware, intervening only when the guest issues an instruction that the virtual machine configuration has specified must be handled by the hypervisor (e.g., access a peripheral device owned by the hypervisor). The Lahav Line in the figure below illustrates execution of a guest on hardware.

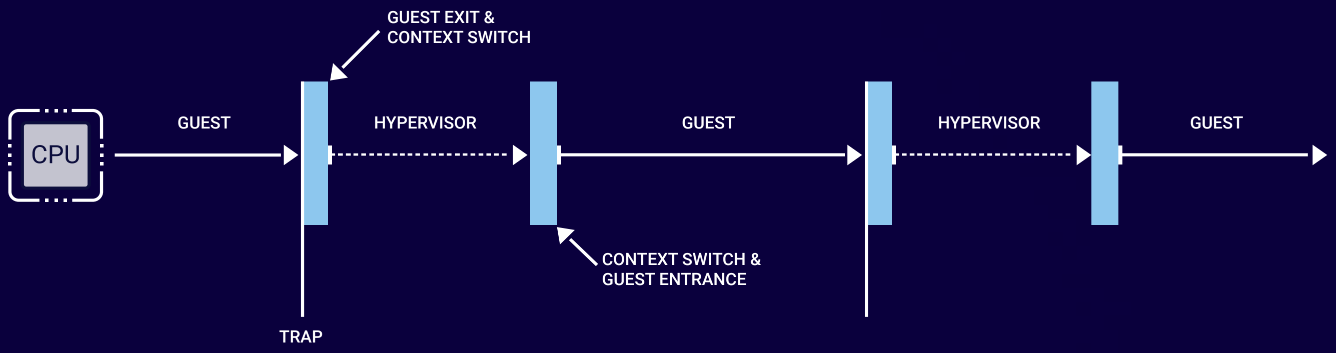


Figure 1:
A Lahav Line illustrating a hypervisor guest executing on a CPU core and the hypervisor's intervention

For more detailed information about hypervisors see our QNX® Hypervisor.

05. CONSOLIDATION

The push for consolidation pre-dates the current chip crisis, though probably not the talent shortage.

Its key drivers are: reduction of device bill of materials (BOM) and total cost of ownership (TCO), and reduction of device weight, power consumption and thermal footprint combined with end-user requirements for both maintaining old systems and implementing new features.

A hypervisor answers these requirements: its most basic function is to abstract the underlying hardware and to present virtual machines in which multiple, diverse systems can run concurrently; for example, a system build on an Android OS and another built on a QNX Neutrino® Real-time Operating System (RTOS), each in its own, separate and isolated virtual machine. In short, hypervisors permit consolidation onto a single SoC of multiple systems that would otherwise require their own, dedicated hardware. This hardware includes, not just the CPU or even the SoC, but also peripheral devices such as displays. The figure below illustrates how two guests with different OSs are implemented in hypervisor virtual machines on a single SoC.

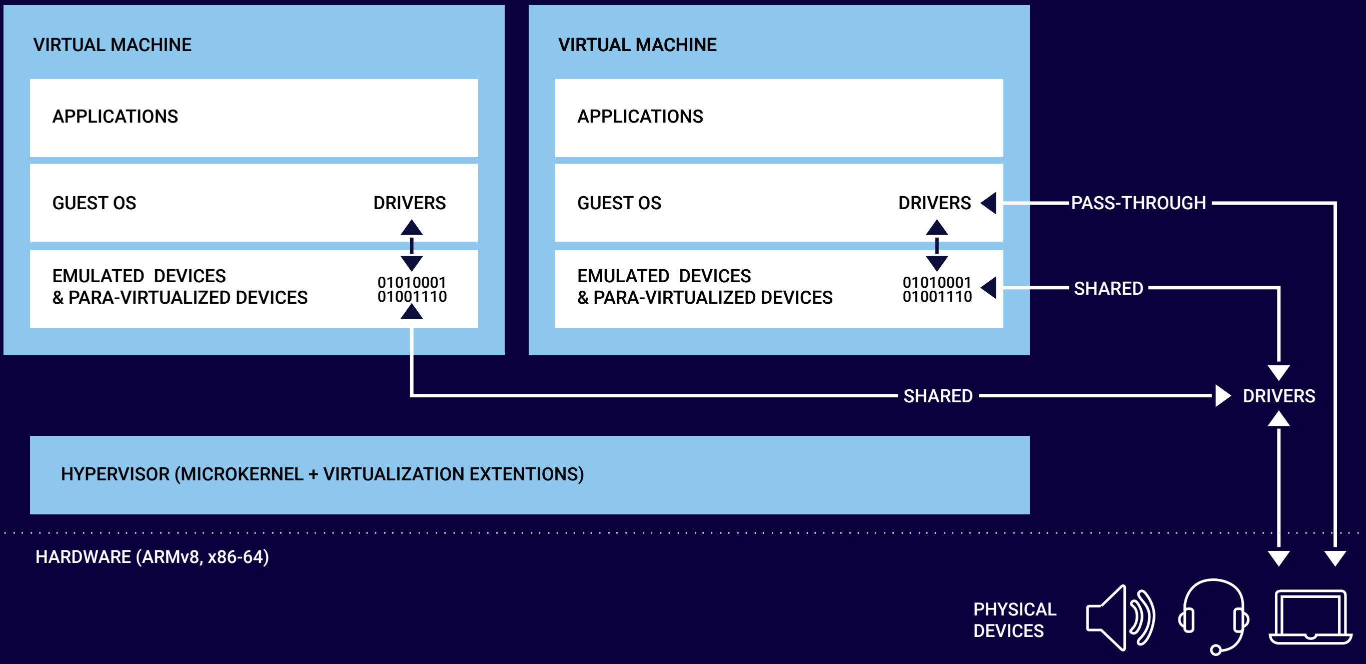


Figure 2:
An illustration of a hypervisor with two guests

06. FLEXIBILITY

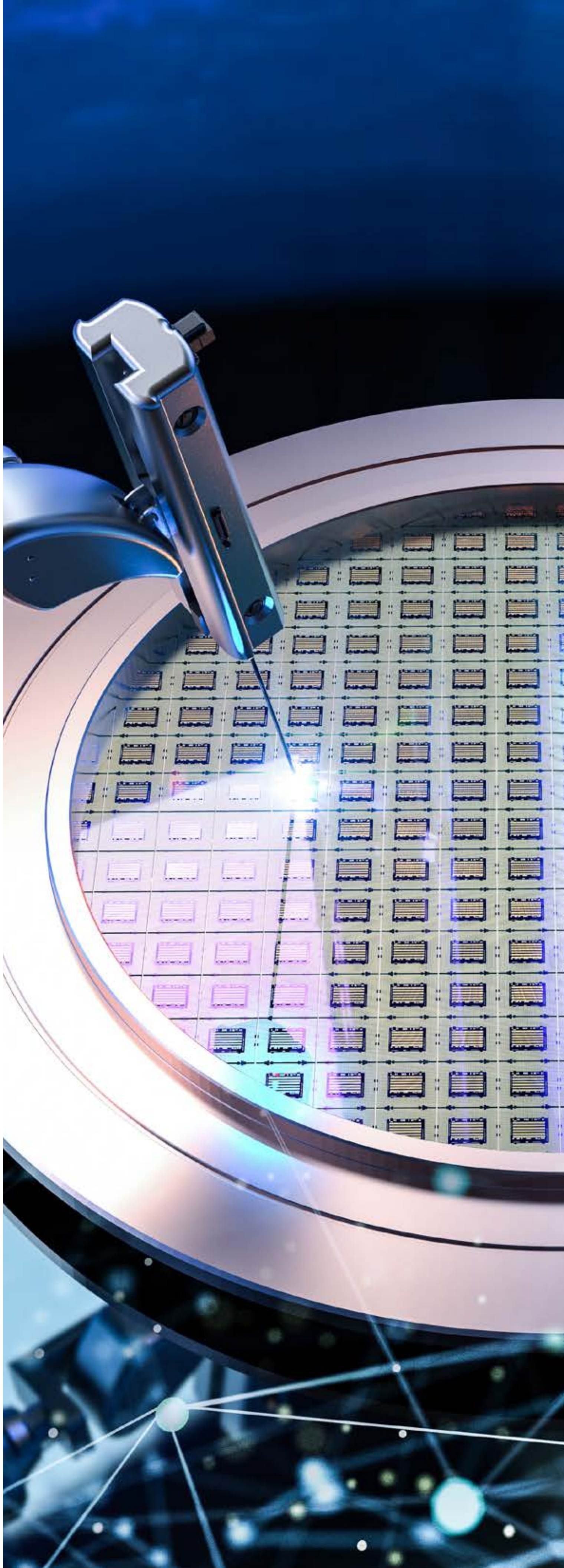
A hypervisor's virtual machines abstract hardware and present stable compute environments in which guests can run.

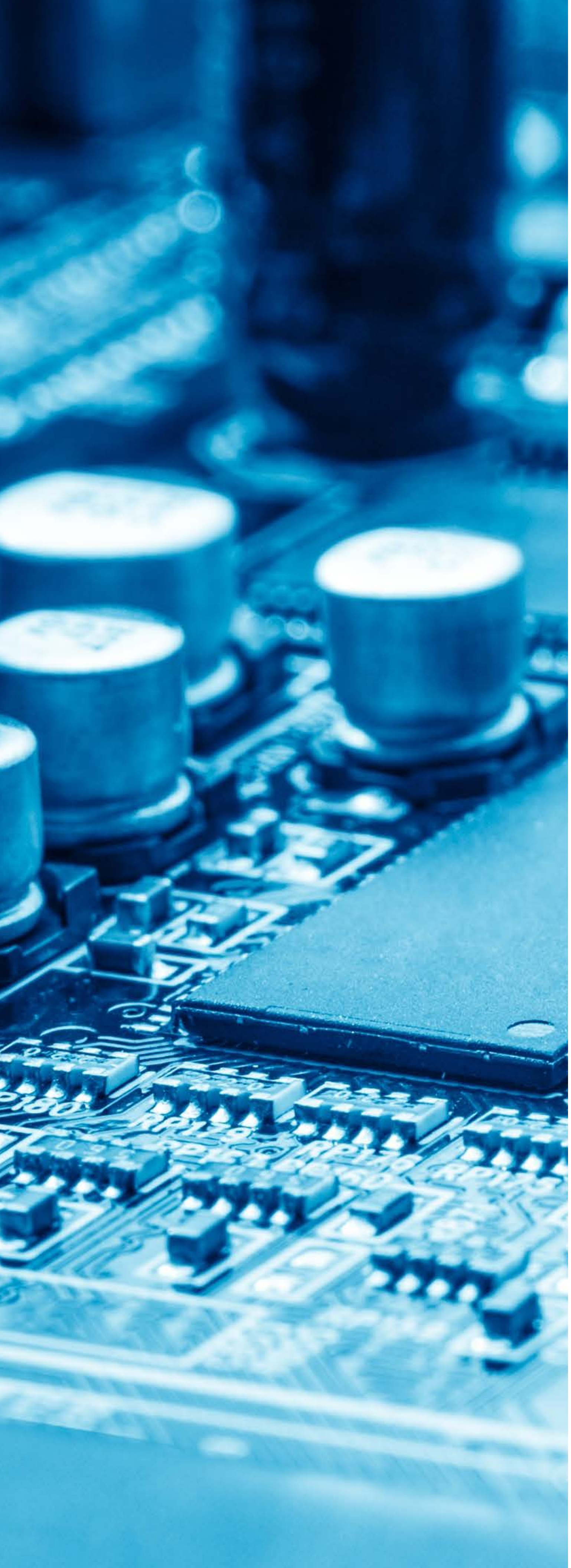
Because guests actually execute directly on hardware rather than in a simulated environment, they must be compiled for the architecture of the underlying SoC. For instance, a guest compiled for an x86 board won't run in a hypervisor virtual machine on an Arm board. Components in the virtual machine (e.g., devices) won't be where the guest expects them to be, and the instruction set won't match.

New systems can be deployed without concern that they will compromise other systems or components running on the same silicon.

Nonetheless, the hypervisor can abstract the hardware sufficiently so as to allow a guest system to run on functionally equivalent SoCs, provided the hosting virtual machine is appropriately configured. In fact, in a hypervisor system, one virtual machine can be configured to host, for example, a 32-bit guest on 64-bit hardware, while another virtual machine hosts a 64-bit guest with a different OS. The hypervisor provides both an environment in which a stable legacy software system can be implemented with no or minimal modification, and an environment where new systems can be deployed without concern that they will compromise other systems or components running on the same silicon.

Thus, the hardware abstraction provided by a hypervisor offers manufacturers more flexibility in their choice of silicon: insurance against chip shortages and price fluctuations, and it allows them to port legacy systems onto new hardware and run them alongside their latest offerings.





07. INNOVATION

With skilled and knowledgeable personnel in short supply, it is critical that their time and talents be spent on activities essential to the business; that is, not on overhead, but on activities that bring in revenue.

For example, an engineer building an automotive infotainment system should no more be tinkering with hardware errata, customizing the OS or even tweaking applications to handle idiosyncrasies introduced by a new SoC revision than she should be repairing lighting fixtures in her lab.

A hypervisor can offload the time-consuming headaches of such 'developer friction', instead enabling innovative and market-differentiating feature development, including the use of cloud-based technologies.

Everything below the virtual machine is below the Babbage-Lovelace demarcation, hence the domain of the hypervisor and its designers.

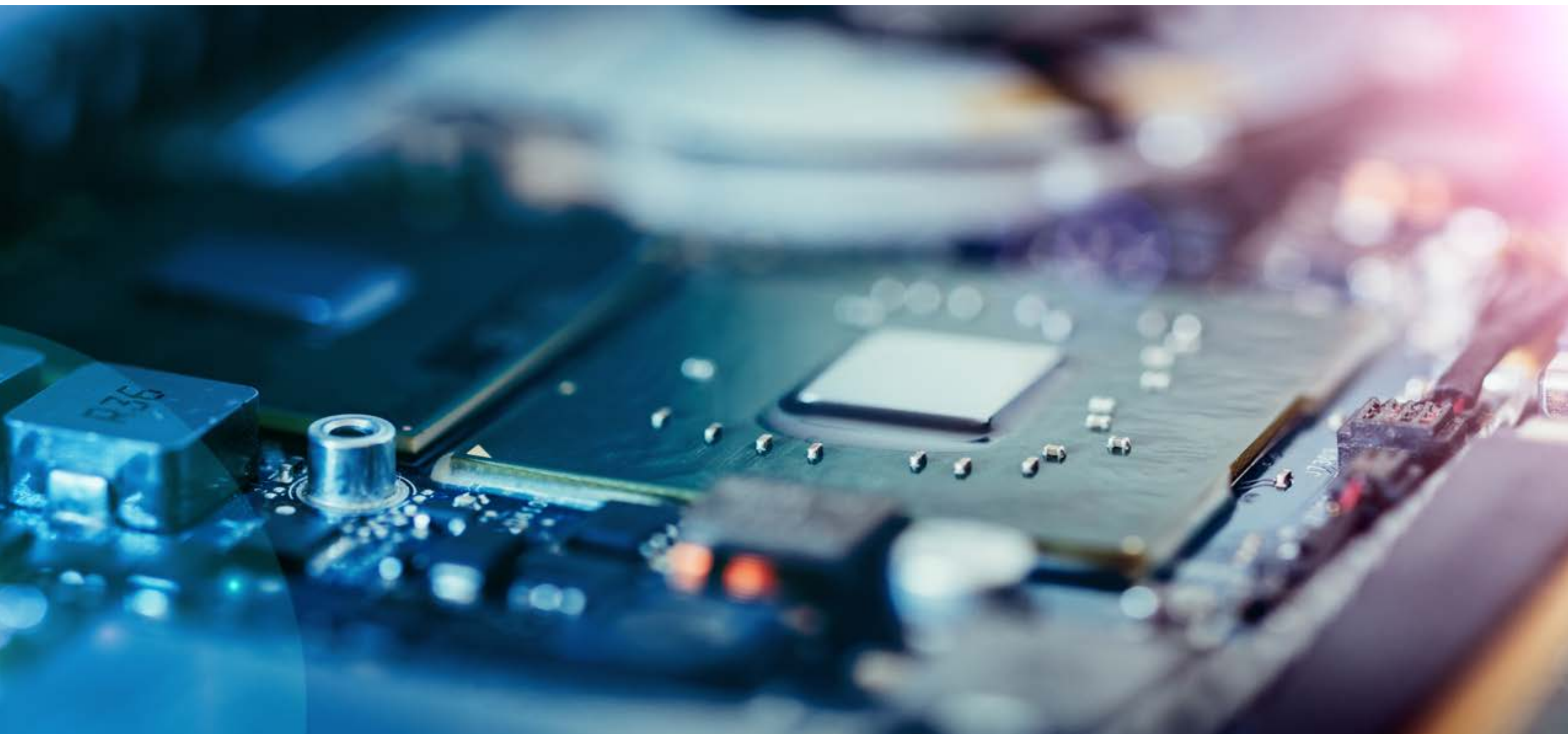
08. SAFETY-CERTIFICATIONS

With the ever-growing demand for embedded systems in safety-critical devices from the Software-Defined Vehicle (SDV) to medical robots, obtaining certification for their products to functional safety standards such as IEC 61508 SIL 3, IEC 62304 Class C, ISO 26262 ASIL D, or EN 50128 is a primary concern of software suppliers and device manufacturers. The role of a hypervisor strategy in safety-certifications therefore warrants special consideration.

Safety-critical systems require isolation from interference, either due to error (bugs in the code) or malicious intent (cyber-attacks). Further, safety-certifications are arduous, time-consuming and expensive; and the greater their scope the greater the expense.

One of the first and most important decisions that we must make when designing a functionally safe system is the scope of the safety-critical system and with that the scope of the certification; that is, what's in and what's out. For example, in a software system running a medical ventilator, the software managing air pressure, the gas mix and the sensors feeding information back to these systems are safety-critical, while software uploading data to the patient's medical records may not be. It would make no sense, then, to spend time and effort ensuring that the data upload software is other than reasonably reliable, as long as we can also ensure that this software cannot interfere with the safety-critical software.

A hypervisor provides an efficient mechanism for separating and isolating safety-critical from non-safety systems, just as it allows the separation and isolation from each other of software systems with different functional safety requirements (e.g., IEC 61508 SIL 2 and IEC 61508 SIL 3). With the different systems contained in their virtual machines, the scope of certifications can easily be limited to just the safety-critical components. With the scope of the certification nicely limited, so too is the time and cost of obtaining it.



If a safety-certified hypervisor with safety-certified virtual machines is available, the task of demonstrating functional safety is further reduced.

Safety-certified components don't guarantee safety-certification or approval for market for the whole system or device, but a safety-certified hypervisor does provide an excellent foundation

If the hypervisor itself has been demonstrated to meet functional safety requirements, and its virtual machines contain non-safety systems, these can be excluded from the certification effort.

The figure below illustrates how a hypervisor can be used to contain one system and isolate the safety-critical system and its applications from interference by the non-safety system.

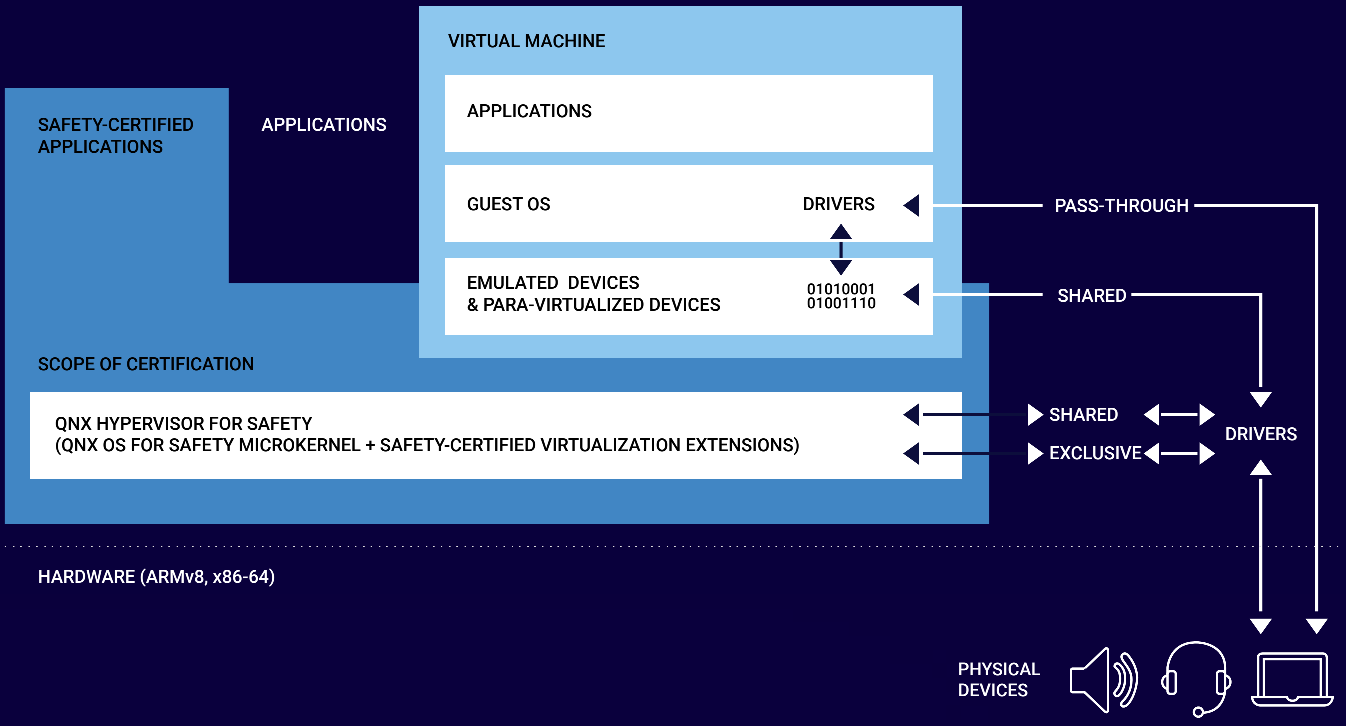

VIRTUAL MACHINE
SAFETY-CERTIFIED APPLICATIONS
APPLICATIONS
APPLICATIONS
GUEST OS
DRIVERS
PASS-THROUGH
EMULATED DEVICES & PARA-VIRTUALIZED DEVICES
01010001
01001110
SHARED
SCOPE OF CERTIFICATION
QNX HYPERVISOR FOR SAFETY
(QNX OS FOR SAFETY MICROKERNEL + SAFETY-CERTIFIED VIRTUALIZATION EXTENSIONS)
SHARED
EXCLUSIVE
DRIVERS
HARDWARE (ARMv8, x86-64)
PHYSICAL DEVICES

Figure 3:
An illustration of a safety-certified hypervisor with one non-safety guest contained and isolated to protect the safety-critical system

09. WHAT A HYPERVISOR STRATEGY OFFERS

The technological benefits of hypervisors are well known. Already in 2013, an article in Embedded Computing Design could rightly claim that "using an embedded hypervisor, developers can have their cake and eat it too". With a hypervisor, developers—more accurately, system designers—can bring legacy software systems onto new silicon, consolidate diverse OSs on the same board, provide the separation and isolation they require for safety certifications, and leverage that same isolation to harden security.

With the exception of consolidation and the reduction in hardware costs it brings, discussions of the business case for hypervisors have tended to be more implicit than explicit. A hypervisor enables delivery of features end customers require; to wit, consumer grade systems such as Android running alongside a safety-critical OS such as the QNX OS for Safety.

That technical questions continue to dominate our thinking about hypervisors is unfortunate. Virtualization for embedded systems has largely been solved. We will see incremental improvements: better virtual devices, more para-virtualized devices, but for both hardware and software we know where we're headed and how to get there.

What has received less attention is how a hypervisor strategy can help a business make better use of available talent while reducing the immediate and long-term costs of meeting customer requirements. The table below presents the four key challenges we've discussed above along with the solutions a hypervisor strategy offers.

Challenge	Hypervisor
Hardware costs, power consumption and thermal footprint	Consolidation: Run multiple, diverse systems on a single SoC, including mixed-criticality systems.
Chip shortages, requiring functionally equivalent substitutes	Flexibility: Reduce time and effort to implement systems on functionally equivalent chips, including porting of legacy code.
Limited supply of talent	Focus: Free up talent to work on high-added value activities.
Increasing requirements for safety-critical systems	Safety-certifications: Limit scope, cost and effort of safety certifications.

In short, a well-reasoned hypervisor strategy can help ensure that an embedded systems supplier is positioned to meet these challenges and seize available opportunities to expand business while keeping the bottom line in check.

About BlackBerry QNX: BlackBerry QNX is a trusted supplier of safe and secure operating systems, hypervisors, frameworks and development tools, and provides expert support and services for building the world's most critical embedded systems. The company's technology is trusted in more than 215 million vehicles and is deployed in embedded systems around the world, across a range of industries including automotive, medical devices, industrial controls, transportation, heavy machinery and robotics. Founded in 1980, BlackBerry QNX is headquartered in Ottawa, Canada and was acquired by BlackBerry in 2010.

BlackBerry® QNX® software and development tools are standards-based and enable companies to adopt a scalable software platform strategy across product lines and business units. The BlackBerry QNX software portfolio, including safety pre-certified products, is purpose-built for embedded systems and scales from single-purpose devices to highly complex systems of mixed criticality. Because we are successful only when you are, you can rely on our support and professional services teams to provide the expertise you need, when you need it—throughout the entire product development life cycle.

QNX-HVsilicon-02-EN(0722)-WP-DF