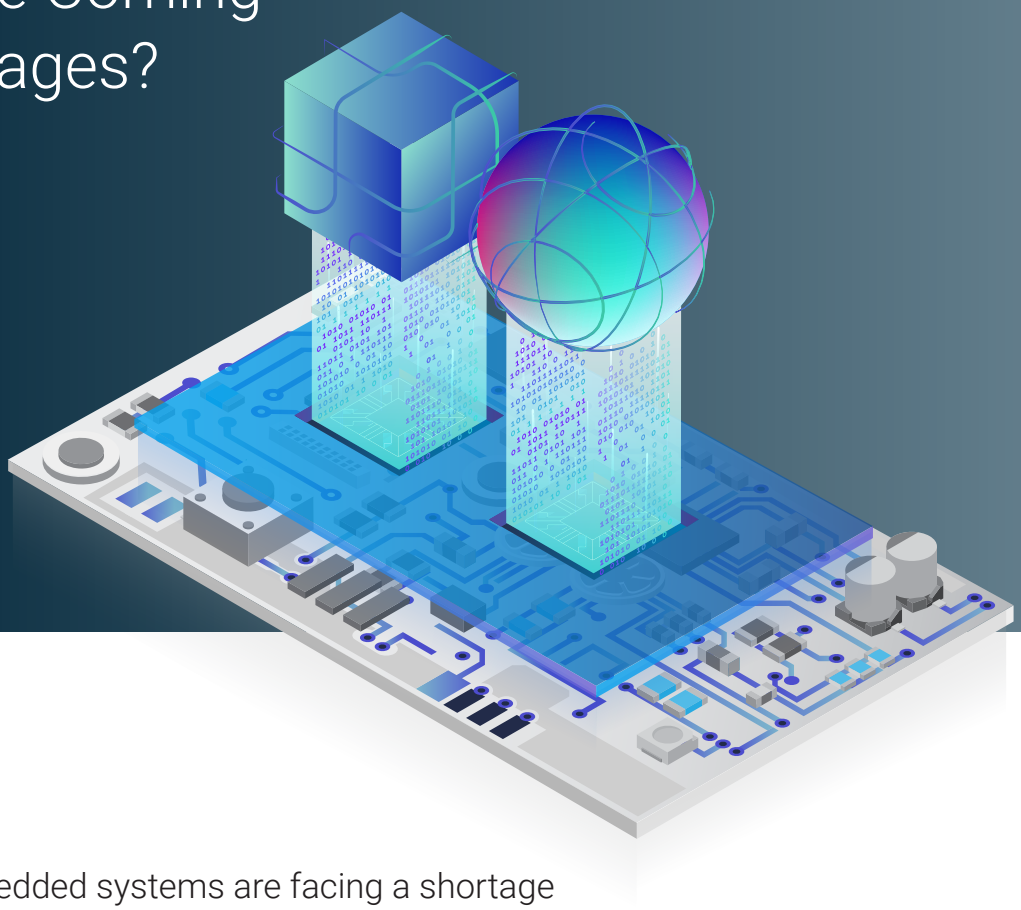
BlackBerry | QNX

Are Hypervisors the Answer to the Coming Silicon Shortages?



Suppliers of embedded systems are facing a shortage of the chips they need for their products, and of the skilled personnel they need to design and build these products.

The chip shortage is in part due to the Covid-19 pandemic disrupting supply chains, but like the shortage of skilled personnel it is also a consequence of the success of embedded systems.

These are now in everything from automobiles to windmills to refrigerators, they are increasingly complex and, even with current slowdowns, demand for them continues to grow in ever larger and more diverse markets.

A coherent hypervisor strategy can both reduce manufacturers' vulnerability to uncertainties in the silicon supply chain and enable skilled personnel to focus on innovation and rapid development of high-value projects.

01 / Dwindling Supplies of Silicon and Talent?

Covid-19 has brought many things, few of them good. In an early 2020 study, published just as the pandemic was firming its grip on economies around the world, VDC Research noted that almost 25% 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. Lack of skilled personnel tops their concerns, ahead even of much-in-the news security!*

A year on, in early 2021, Bloomberg along with other financial and technology publications are sounding the alarm on the domino effect of dwindling chip supplies, attributable at least in part to the pandemic's effect on supply chains. With 40% of the vehicles' value in the electronics, major automakers such as Ford, Volkswagen and Toyota have already closed plants across the world in large part because they can't get the silicon they need, and "The auto industry may lose $61 billion of 2021 sales from chip shortages".

> The auto industry may lose $61 billion of 2021 sales from chip shortages.

Their troubles are just the tip of the iceberg. Bloomberg cites Neil Mawston, an analyst with Strategy Analytics, who estimates that "prices for key smartphone components including chipsets and displays have risen as much as 15% in the past three to six months".

Covid is partly responsible for the chip shortage; it can't be blamed for engineers' concerns about the penury of relevant skills in their organizations. That 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, and the use of these systems in safety-critical environments increasing requirements for safety-certifications. Building a safety-certified cobot, a robot that interacts directly with humans, is a rather more demanding task than making an HVAC vent controller with a programmable logic controller (PLC). To be fair, though, building and maintaining a complex HVAC system like the one implemented in the German Parliament (Bundestag) buildings is a significant undertaking.

* For users of "freely/publicly available open source", this figure rises to 37% of respondents. For those building in-house solutions without an OS it is 36%, but includes a staggering 28.2% for "lack of manpower/technical skills"!

02 / Consolidation, Flexibility and Focus

Attributing responsibility—to a virus, no less! won't create more chips out of thin air, nor will it generate from the same place the skilled personnel needed to build the systems that would use these chips, were they available.

Assuming that 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, a reasonable strategy for dealing with these shortages is to reduce our reliance on the critical resources currently in short supply and likely to remain so for the foreseeable future.

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-a-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.
- **Focus:** unburdening skilled personnel from tasks not critical to the core business so they can apply their knowledge and talents to high value-added work.

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.

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.

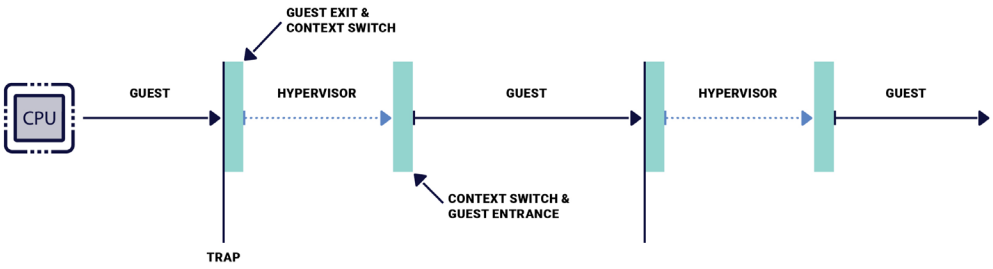


Figure 1. A Lahav Line illustrating a hypervisor guest executing on a CPU core and the hypervisor's intervention

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.

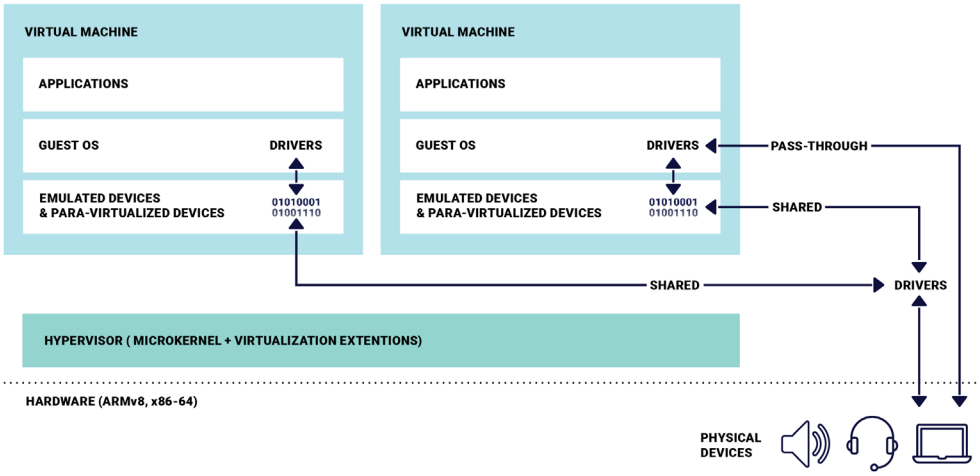


Figure 1. An illustration of a hypervisor with two guests

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 / Focus

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 a medical ventilator 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 dealing with functionally equivalent but not identical chips, and new chip variants and revisions from staff whose time is best spent developing capabilities and features that add value to the product. 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 autonomous vehicles to medical robots, obtaining certification for their products to functional safety standards such as IEC 61508, EN 50128, ISO 26262 or IEC 62304 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.

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 for a safety case.

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 for a safety case: the hypervisor itself has been demonstrated to meet functional safety requirements, and with its virtual machines containing 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.

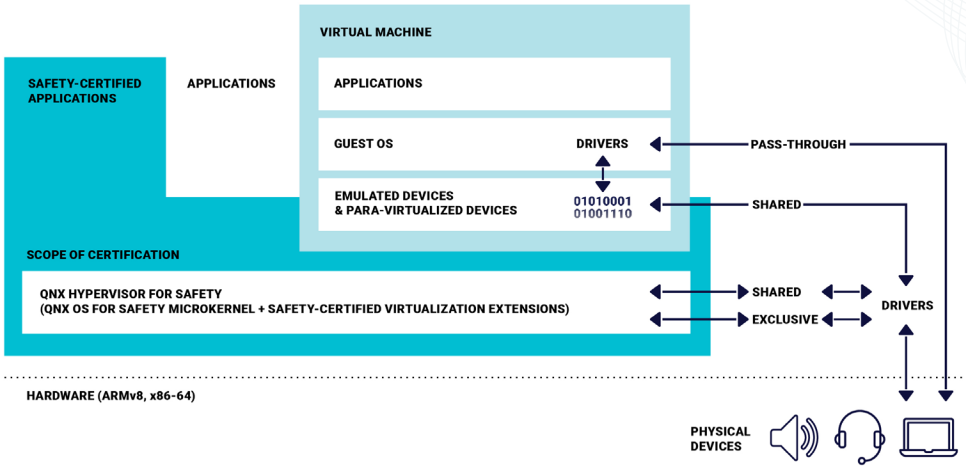


Figure 3. An illustration of a safety-certified hypervisor with one non-safety guest contained and isolated to protect the safety-critical system

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 paravirtualized 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 195 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.