Virtual software platform for the car

On the road to the next paradigm shift with VIRTIO standards

Isaac Trefz, COQOS HypSDK, Berlin

Abstract
Virtualization technology is gaining acceptance in the automotive industry. It makes the development process of software systems more efficient, reduces development and BOM costs, and shortens time-to-market. Especially in the cockpit, manufacturers and their direct suppliers have now been using hypervisor-based systems for several years. This first step towards a new conception of solutions is now experiencing another paradigm shift: the use of open standards. It further enhances the advantages of virtualization. The fully virtualized guest operating systems can be flexibly deployed and reused across all hardware. Only then can the full potential of virtualization technology be fully exploited.

Consolidation of the growing number of functionalities

New driver assistance systems, digital cockpits, and passenger and rear seat displays are in vogue. They require very powerful hardware, but at the same time the growing number of hardware components conflicts with the automotive industry's declared goal of producing lighter and more fuel-efficient vehicles. Virtualization opens up a way to consolidate functions and drastically reduce the number of ECUs. This reduces costs, network complexity and the weight of the vehicle.

A key component of virtualization architectures is the hypervisor. It runs directly on the system-on-chip (SoC) processor and allows multiple virtual machines (VM) with different operating systems to run in parallel. Each VM is isolated from the others. This separation (ISO 26262 calls it "freedom from interference") allows multiple different functionalities to run on a single SoC, even if the systems are assigned different criticality (ASIL levels such as QM, A, B, C and D). Thus, the hypervisor enables the operation of guest operating systems such as Linux, Android, AUTOSAR or other real-time operating systems in separate virtual machines.

The consolidation of functions made possible in this way requires very powerful processors, to which many silicon manufacturers have responded and today offer system-on-chips (SoCs)
that provide such high computing power. On these SoCs are co-processors such as graphics processing units (GPUs), on which a large number of pixels can be driven for multiple displays. Because the hypervisor allows different applications to share the same central processors, maximum utilization of these processors is achieved. This sharing of processors reduces the overall computing power required and, as a result, material and development cost, time and risk.

However, hardware manufacturers offer operating systems with proprietary driver solutions that they deliver in a Board Support Package (BSP). The number and versions of each operating system are limited. Therefore, system developers must ensure that their applications can run on the operating systems supported by the silicon manufacturer in each case. In addition, hardware manufacturers usually only support their SoCs for a limited period of time. This makes it difficult or even impossible to update the software and services in vehicles in the field.

As a result, OEMs and original equipment manufacturers want a different solution: they want to be able to use and update operating systems, which are often tailored to their exact applications, throughout the life of the vehicle. They want to reuse the same architecture and software building blocks across multiple vehicle lines to save even more development time, cost and risk. In detail, this means that different function sets should be reused on software systems and hardware from different manufacturers, in different models, and across multiple vehicle generations.

Manufacturers also want to obtain software from many sources, such as their own internal software houses and third-party suppliers, instead of system software that is usually provided by one supplier.

Moreover, instead of system software, which is usually provided by one supplier, manufacturers want to obtain software from many sources, such as their own internal software houses and third-party suppliers. In addition, over-the-air software updates and device management are a must in all new vehicle systems. This is the only way to fix functionality, safety or security bugs, and vehicle owners have long expected to be able to add new features after purchase. For this, it is important that modular software updates are possible. After all, OEMs don't want to re-qualify the entire device after the update, but only the added or updated application.
Standards-based Virtualization for more flexibility

Virtualization based on open standards offers exactly these desired options. Here, the operating systems use virtual drivers that are standardized and therefore independent of the hypervisor as well as the underlying hardware. These drivers are responsible for the communication between the respective guest operating system and the hardware, e.g. display controller, GPU, sound device, etc.. Usually, one of the operating systems running in a virtual machine serves as the physical device manager. The device manager contains physical drivers that are provided by the silicon vendor for the actual hardware or that are developed or customized by the customer. In addition, there is a customization layer in the device manager that provides the physical drivers via a standardized interface. The operating systems running in other virtual machines use the virtual drivers. They access physical devices through the standardized interface provided by the adaptation layer in the physical device manager. Since the virtual drivers in the other guest VMs rely only on standardized interfaces for hardware access, they are completely independent of hardware-specific device drivers.

If the automotive industry, as a whole, can be convinced to use such virtual drivers, all conforming to the same open standard, they will be free to choose the hardware, software and driver customization layer for their software systems. Reusing existing software systems on other hardware or deploying other software on existing hardware is just as easy. Manufacturers can obtain the adaptation layer between the hardware's physical drivers and the standardized virtual drivers from any company working with the common standard, or even develop or adapt the adaptation layer themselves.

Virtual I/O Devices (VIRTIO) on the way to the automotive industry

The open industry standard, VIRTIO, is very well suited for the decoupling between software and hardware, and has already proven itself in series production. VIRTIO offers a very mature framework for sharing devices with different operating systems running on one or more SoCs that include a hypervisor. Developed in 2008, the standard specifies the interface between virtual machines with access to input/output devices, and interactions between virtual machines and hypervisors. Instead of physical hardware, VIRTIO devices provide abstract device functionality via virtual drivers for guest operating systems. These virtual drivers are
already available in all major operating systems: Linux, Android, BSD and Windows, as well as some proprietary real-time operating systems.

The standard, officially called "OASIS Virtual I/O Devices", is promoted and maintained by the OASIS consortium. It is supported by a large community of developers worldwide and is continuously extended for use in the automotive industry. Thanks to this commitment, new devices that are essential for the automotive industry (e.g. audio, sensors, GPU support for 2D and 3D, USB, CAN, etc.) are currently being virtualized according to the standard.

If software companies in the automotive industry replace their existing devices with VIRTIO-compatible implementations, the gain in flexibility becomes immense. Version 1.2 of these devices will be available for the automotive industry soon:

<table>
<thead>
<tr>
<th>VIRTIO-net</th>
<th>Paravirtualized Ethernet</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIRTIO-input</td>
<td>Input for Touch</td>
</tr>
<tr>
<td>VIRTIO-scmi</td>
<td>Platform clocks and controllers for seamless pass-through of devices</td>
</tr>
<tr>
<td>VIRTIO-snd</td>
<td>Audio</td>
</tr>
<tr>
<td>VIRTIO-blk</td>
<td>Universal Flash Storage</td>
</tr>
<tr>
<td>VIRTIO-gpu</td>
<td>Graphic Unit</td>
</tr>
<tr>
<td>VIRTIO-rpmb</td>
<td>Protected memory area</td>
</tr>
<tr>
<td>VIRTIO-console</td>
<td>Inter-VM communication</td>
</tr>
<tr>
<td>VIRTIO-rng</td>
<td>Random encryption generator</td>
</tr>
<tr>
<td>VIRTIO-vssock</td>
<td>Point-to-point connection between VMs</td>
</tr>
</tbody>
</table>
Because the virtual driver characteristics are similar to Direct Memory Access (DMA), the implementations achieve the same, or nearly the same, performance as hardware-assisted I/O virtualization models. This is because VIRTIO avoids costly memory copy operations. Instead of transferring memory contents, which would be slow and place an excessive load on the system's general-purpose compute cores, VIRTIO transfers references to actual data via virtual queues. For example, to write a large block of data to the physical flash device, the virtual device manager inserts only a small reference to the memory where the data block resides. The physical device manager then transfers the large block of data directly from the guest VM to the flash device controller.

An illustrative example where VIRTIO meets high performance requirements is the sharing of GPUs in a cockpit controller. The cockpit controller controls multiple displays, typically the instrument cluster display and the infotainment display. Sometimes it also includes additional screens such as a head-up display that projects important information for the driver, or screens for passengers in the front or rear of the vehicle. To achieve this, a large number of very different functions are integrated on a system-on-chip: from infotainment applications to driver information systems. All these systems, running on the same virtual platform, can share the available GPU power with minimal performance overhead.

Fig. 1: Example: VIRTIO block device
The new variants of Google’s Android ™ Automotive OS will have a pre-integrated VIRTIO framework. This will allow Android Automotive OS to run on all existing systems that support VIRTIO. The VIRTIO-based virtualization platform also guarantees secure coexistence with other operating systems, such as Linux, an RTOS or other Android instances.

Software-Defined Architecture for a new design approach

In the future, this technology will enable a whole new approach to the design of software-driven in-vehicle functions: software defined architecture (SDA). SDA is used in the cloud and data center space to make cloud applications easier to build. To do this, virtualization hides hardware, system management and orchestration. This approach is also very useful for designing software architectures in cars.

Traditionally, car manufacturers have generally designed their models starting from the individual components of the vehicle. They have also developed the software of a car on the basis of the hardware, or had it developed by suppliers. This led, and still leads, to hardware-specific software stacks and supplier lock-in.

The SDA approach separates the hardware from the application software through the virtualization technology described in this article. An architecture can first be designed that
includes all the desired functions of a system, such as a cockpit. The virtual platform is developed according to this architecture without considering hardware or porting efforts. It is only after the software design has been created that manufacturers decide which hardware to use and which, or how many, SoCs are required to provide the necessary computing power. This makes it easier for manufacturers to use the same software across many vehicle lines with different hardware configurations. The Software Defined Architecture approach thus fulfills existing desires for flexibility: OEMs have freedom of choice in hardware (SoCs, MCUs, GPUs, sensors, storage solutions, and networks) and in software components (operating systems and hypervisors).

Infotainment devices based on Software Defined Architecture and using the VIRTIO standard 1.1 are already in production. Development with more advanced VIRTIO devices, e.g. graphics units on a single SoC, has already started and should enter volume production in the next 1-3 years.

**Conclusion**

This paradigm shift to thinking about vehicle development from a software perspective is a huge step forward for the automotive industry. It is time for manufacturers and suppliers alike to move from proprietary solutions to open standards. This will allow them to create great innovations while saving time and material costs at the same time.