AUTOMOBIL ELEKTRONIK
E/E-Entwicklung für Entscheider

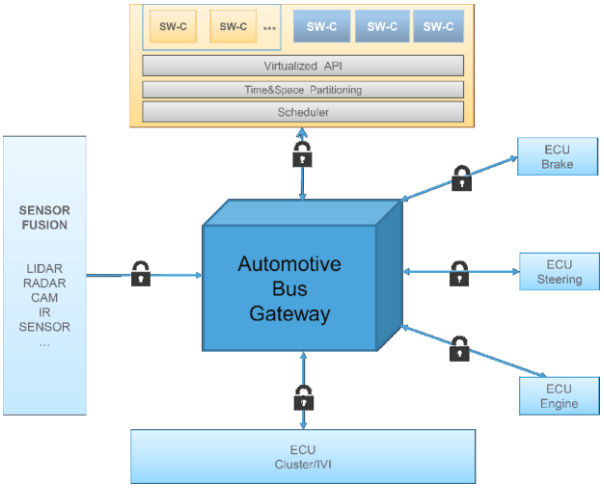
SYSGO EMBEDDING INNOVATIONS

Uniform hardware platforms for all car features

The cost-effective and safe implementation of assistance systems and autonomous cars requires a new approach to development of the control components. The objective is to integrate diverse applications on one hardware platform, but—for safety reasons—to be able to continue operating them fully independently.

An article by Chris Berg, Franz Walkembach

Once upon a time they were just mechanical machines. Today's cars, though, are already based to a large extent on electronic components along with the corresponding software, and the trend towards autonomous driving will undoubtedly intensify this progression. The car of the future will require even more electronics and computing power; on the one hand, to increase safety with the help of driver assistance systems and, on the other hand, to meet the comfort, entertainment and communication requirements of the passengers. The Internet of things (IoT), which lends the devices inside the vehicle the necessary connectivity for this, entails considerable new challenges for safety. The safety aspects are therefore increasingly determining the software design, and the certification standards will soon play a similarly important role in car manufacturing as they already do today in the aeronautical industry.



Based on the separation microkernel, various operating systems and applications can run with strict separation.

Hardware and software systems in cars are historically fragmented. The introduction of electronic systems usually took place completely independently of other systems, which led to an uncontrolled growth in CPUs and controllers, and especially in software. In today's vehicles, between 60 and 100 different CPUs control diverse functions, such as engine control, lighting and the brakes, with their own software applications. These CPUs are also interconnected by way of up to seven different buses. Such complexity increases the development costs and production costs and does not make servicing any easier either.

Multiple hardware platforms therefore also require different development environments and software developers with the respective expertise, which can be a significant cost factor. In addition, it is of course the endeavour of every manufacturer to reduce the hardware costs and shift the functionality increasingly to the software (i.e. the software defined car).

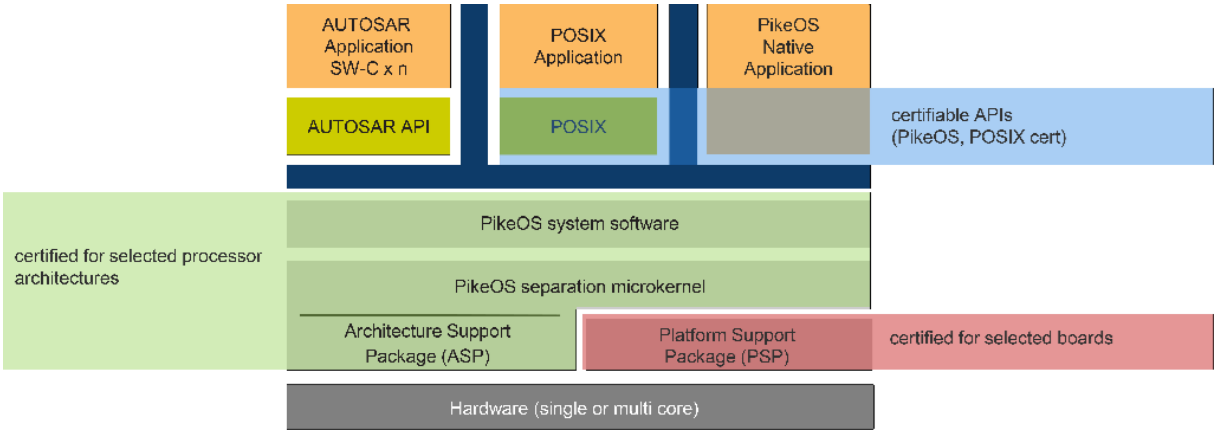
One of the main objectives in the development of the car of the future is therefore the introduction of a uniform platform that controls all car functions.
Despite all problems that the uncontrolled growth in CPUs entails, it does have one big advantage: it separates the individual functions so that no system can be impaired by any errors in another system. Under no circumstances can the audio system affect the brakes in today's cars, because the two are controlled by strictly separated systems. If such diverse systems are migrated to a uniform hardware platform, this separation is no longer guaranteed from the outset and must therefore be achieved in other ways. In aircraft and railway construction this problem has already been largely solved and the approaches used can also be transferred to the car industry.

Hypervisors separate the applications

Many software suppliers use hypervisor technologies to enable several operating systems to be run on just one hardware platform. This is a virtualization technique that uses hardware functions to simultaneously operate guest operating systems. Each of these guest operating systems is provided with a partition that is strictly separated from all of the other operating systems, also called a container, in which it operates independently of all other systems.
Such architecture is obviously suitable for coming closer to the objective of unification of the hardware platforms. In several partitions the hypervisor can host diverse functions that previously required separate CPUs. However, it must be absolutely ensured that the software providing the hypervisor functionality does in fact guarantee strict separation between the partitions. Otherwise, although there is a uniform hardware platform, there may be interactions between critical and non-critical applications, as in the example with the audio system and the brakes. The safety certifications, such as SIL 4 and ISO 26262, then come into play at this point. Such certified hypervisor technologies provide the certainty that the functions in different partitions really are separated from one another as if they were running on different CPUs.

Multi-core CPUs

Another popular approach for the achievement of this objective is the use of multi-core CPUs. Although such CPUs are primarily used for performance reasons, they can also support the required separation of the individual functions. However, the certification of multi-core systems is very complex, many certified systems actually using only one core, which foils the very reason for using multi-core CPUs, the performance. In addition, separation of the functions is therefore not guaranteed. If various functions are bundled into one single piece of software that runs in a Real-Time Operating System (RTOS) on just one CPU core, interference between the functions can occur very easily. The effect of an application on the run-time behaviour of another application can lead to safety problems, for example, the exceeding of time limits in real-time applications. Similarly, timing effects due to the joint use of system resources, for example, cache and memory buses, can lead to hidden information channels that infringe the confidentiality requirements of the application. However, if critical functions are spread over various cores, the required strict separation is guaranteed.
The use of hypervisors on multi-core systems is fundamentally a suitable means to meet the challenges encountered in system design. However, the use of hypervisors alone is not a guarantee for strict separation of the different functions. Most hypervisors are on an RTOS platform, which, as far as its own design is concerned, does not support such separation. Precisely in safety-critical applications, however, it is important that the RTOS is already designed specially for the separate running of different functions, i.e. as far as the design is concerned it is a separation kernel rather than a simple RTOS.



Diverse applications on one hardware platform and a deterministic communication channel enable simple but safe system design.

Separation as the design basis

Such a separation kernel enables spatial and temporal separation between applications and provides the partitions for the execution of user applications. The chronological separation is effected by means of time partitioning, the CPU time being subdivided into time partitions during configuration. The duration, order and repetition period (main time frame) of time partitions can be statically defined by the integrator as a time schedule. A scheduler schedules the time partitions in accordance with the static time schedule. User applications are assigned to one or several such time partitions and are considered for scheduling only if the relevant time partition is active. In this way, the chronological behaviour of a partitioned user application is independent of the rest of the system. Spatial separation is effected by means of resource partitioning, in which the system resources—such as main memory, files, devices, safe communication channels and cores—are partitioned and assigned statically to containers, the so-called resource partitions. The execution of user applications takes place within the context of a resource partition. During execution the separation kernel ensures that an application has guaranteed access to the assigned resources of its resource partition and that applications from other resource partitions cannot use these resources unless sharing is explicitly configured.

Android and Autosar on one hardware unit

Architecture based on a separation kernel also simplifies the integration of functions of differing criticality on one hardware platform. For example, Android-based infotainment functions and critical Autosar applications that can be certified independently of one another—which significantly accelerates certification and reduces the costs of certification—can therefore run in partitions strictly separated from each other. Moreover, in the event of further consolidation, only the new additional applications have to be validated, because they do not affect the previous ones.

KEY DATA

To counteract the uncontrolled growth in CPUs, controllers and software in today's vehicles, manufacturers are placing increased emphasis on CPUs with a separation kernel, on which diverse operating systems can run and which enables spatial and chronological separation between applications. Simple and safe system designs are achievable with such hypervisor technologies.

Especially in mixed environments with relatively poorly safeguarded operating systems, such as Android, a separation kernel can also perform the important security function of making attacks more difficult. This kernel,

which acts as a hypervisor for the different guest operating systems, is in a position of being able to intercept privileged system invocations by the guest systems and to check them first of all for the required permissions prior to their actual execution. While normal desktop operating systems have all device drivers integrated into the kernel, in this way an environment can be created in which only a very small set of services runs in privileged mode in the kernel—for example, scheduling, context switching, process communication, process synchronization and interrupt handling. Device drivers then operate without privileges in normal user mode like any other application code, which significantly reduces the attack surface of the entire system.
(pet)

Contact:
SYSGO AG
Am Pfaffenstein 14
D-55270 Klein-Winternheim
+49 6136 9948-500
info@sysgo.com
www.sysgo.com

