Towards Real-Time Checkpoint/Restore for Migration in L4 Microkernel based Operating Systems

Sebastian Eckl
Chair of Operating Systems
Technische Universitaet Muenchen
Munich, Germany
Email: sebastian.eckl@tum.de

David Werner
Chair of Operating Systems
Technische Universitaet Muenchen
Munich, Germany
Email: david.werner@tum.de

Uwe Baumgarten
Chair of Operating Systems
Technische Universitaet Muenchen
Munich, Germany
Email: baumgaru@tum.de

***Abstract*—Future development in Cooperative Intelligent Transport Systems (C-ITS) and Autonomous Driving will require distributed embedded real-time systems to cope with an ever increasing amount of software-based functionality. Besides traditional CPU-centricity, more and more functionality will also exploit dedicated heterogeneous computing units like embedded GPUs, FPGAs or specific AI/neural network co-processors for acceleration of specific taskloads. Recent ARM-based automotive-grade hardware platforms from Xilinx (Zynq UltraScale+), Renesas (R-Car) or Nvidia (DRIVE AGX) represent this ongoing development. In order to still guarantee high safety standards on common hardware modules, hardware consolidation and system virtualization - a concept already successfully demonstrated in form of integrated modular avionics (IMA) - shall also help future automotive systems in providing both required computing power and guaranteeing safety and security within a mixed-criticality multi-core environment. From software perspective, a proven foundation can hereby be found within separation kernels (e.g. PikeOS), which make use of microkernel design considerations to comparably combine software of different criticality on the same underlying hardware platform. Nevertheless, these existing mixed-critical partitioning approaches lack an adequate degree of flexibility regarding adaptation at runtime. As future automotive development will more and more demand a certain amount of dynamic reconfiguration, e.g. when applying OTA updates or remote installation of software-based functionality, existing techniques will reach their limits.**

Within cloud or data center environments, VM/live migration techniques are already providing a certain degree of runtime adaptation and offer solid mechanisms that allow for fault tolerance/availability or efficient resource management. Building upon existing L4 microkernel-based (virtualization) concepts, we examine the transfer and adaptation of already approved techniques to the automotive environment, by addressing the migration of software components and processes in distributed embedded real-time (operating) systems (RTOS). Besides the provision of a L4 microkernel-based homogeneous run-time environment (RTE), main research is focusing on the development of real-time capable checkpoint/restore mechanisms for snapshot creation and continuation. Prototypical implementation is hereby based on the L4 Fiasco.OC microkernel and the Genode OS Framework. Ongoing development led to the creation of the Real-Time Checkpoint/Restore (RTCR) software component, which is offering two distinct mechanisms:

- **a shared memory based approach (post memory copy), which is able to checkpoint relevant (memory) data at snapshot creation time by explicitly stopping the system for a certain period of time, in order to create a consistent snapshot.**
- **a redundant memory approach (pre memory copy), which is able to intercept each write access call at run-time in order to copy relevant data to a backup area in parallel to program execution. System stopping time may be reduced, as only non-memory related data has to be copied at snapshot creation time.**

Based on the above mentioned mechanisms, further software-based optimizations, like incremental memory checkpointing, read-only memory attachments or copy-on-write have been implemented. As main bottlenecks of the pure software-based implementation, missing parallelization and duration of memory checkpointing could be identified. In order to accelerate the existing solutions, a combination between the RTOS and reconfigurable hardware seemed promisable. Addressing the aspects efficient memory tracing and copying, additional hardware components have been developed with the help of an FPGA, prototypically implemented on the Xilinx Zynq-7000 platform:

- **a hardware component that redirects memory traffic to the FPGA and intercepts memory access in order to distinguish between read and write access calls, trace write calls and write respective data to a redundant backup during run-time.**
- **a hardware component that acts as a co-processor and creates a copy between two memory regions, if told to do so.**

Ongoing work is targeting the lack in parallelization, focusing on efficient exploitation of ARM Cortex-A homogeneous and heterogeneous (ARM big.LITTLE) multi-core architecture as well as ARM Cortex-M based co-processors. Furthermore, checkpoint/restore specific adaptation of CPU components (e.g. the MMU) is examined based on the RISC-V architecture. Additionally, the existing work will also be ported to the seL4 microkernel.

Snapshot creation and migration timing behavior will be evaluated based on a hybrid simulator approach, likewise combining a virtual autonomous driving test environment and physical control devices, executing the operating system.