Multi-core system architecture for safety-critical control applications

By Gang Li

Nowadays, safety-related systems based on microprocessors are increasing in size and complexity. This leads to some challenges for the development of safety-related systems. The development and certification cost of a safety-related system is increased at an exponential rate as the size and complexity are growing. An application of large size requires increased processing power that maybe a single-core processor cannot provide. Furthermore, current safety-related applications are usually mixed-criticality, and isolation between applications is required for non-interference between them if being implemented on a platform. The arrival of multi-core devices gives safety domains an alternative platform to implement a safety-related system of increased size. As multi-core devices are being widely used in embedded systems and becoming the mainstream, more and more frameworks, methods and tools are available for users. Hence, this motivates this project that concentrates on how to implement safety-related systems on multi-core devices with cost-efficient certification.

Typical existing multi-core architectures do not match current safety-related control applications, since they do not provide isolation between applications with different SILs and deterministic execution behavior. Thus, this dissertation has presented a predictable partitioning multi-core architecture for safety-critical (mixed-criticality) control applications, aiming to address development and certification challenges regarding safety-related systems on multi-core devices. A conceptual partitioning multi-core architecture model has been proposed first, which concentrates on the separation of the platform hardware and software architecture. It provides isolated execution environments named Platform Partition Units (PPU) to applications. Application components executed on a PPU is temporally and spatially isolated from applications on other PPUs. To follow the partitioning multi-core architecture model, specific multi-core hardware and software architectures have been proposed to exploit the natural physical separation between cores. Meanwhile, time-triggered scheduling and hardware supports like Memory Protection Units (MPU) or Memory Management Units (MMU) are used to enable isolation between application components inside a core. Hence, the multi-core architecture can provide two-level isolation: intra-core and inter-core. A real-time separation kernel $\text{HAERTEX}_{\text{safety}}$ and an embedded hypervisor $\text{RodosVisor}_{\text{sc}}$ have been deployed to support the partitioning multi-core architecture. A process-based optimization algorithm has been presented to solve the complicated issue how to search out task mapping and scheduling solutions with the lowest system development cost on potential partitioning multi-core architectures of different number of cores.