Virtualisierung auf Mikrocontrollern

Eingebettete Virtualisierung ermöglicht es, die Ressourcen eines Prozessors mehreren sicher getrennten Anwendungen und Betriebssystemen zuzuordnen. Dies ist ein effektiver Ansatz, um die Leistung von Prozessoren voll auszunutzen und die wachsende Komplexität von softwaredefinierten Funktionen zu bewältigen. Durch eine neue Hypervisorplattform kann sie jetzt auch auf Echtzeitprozessoren und Mikrocontrollern eingesetzt werden.

Fahrzeuge benötigen ständig wachsende Rechenleistung, um softwaredefinierte Funktionen für Infotainment und Connected Services zu betreiben. Diese Funktionen unterstützen den Fahrer, machen das Fahrzeug sicherer und managen Energiequellen. Die Ära, in der jede neue Fahrzeugfunktion die Integration eines neuen Steuergeräts erforderte, ist längst vorbei: Fahrzeughersteller entwerfen neue Architekturen, bei denen viele Softwarefunktionen auf zentraleren, leistungsfähigeren Geräten integriert sind. Sie werden oft als „Domänencontroller" bezeichnet (Bild 1).

Die verschiedenen Funktionen im Auto haben sehr unterschiedliche Anforderungen an die Software- und Hardware-Plattformen, auf denen sie ausgeführt werden. Die Software-Anwendungen benötigen von der Hardware mehr als nur die generische Rechenleistung. Sie nutzen spezialisierte Beschleuniger, um hochauflösende Grafiken zu erzeugen, um Kamerabilder oder Radardaten zu verarbeiten und um Algorithmen für künstliche Intelligenz wie Deep Learning zu betreiben. Darüber hinaus stellen die Fahrzeugfunktionen unterschiedliche Anforderungen an die funktionale Sicherheit (ISO 26262 von „QM" bis „ASIL-D"), Boot-Zeiten und an das Echtzeitverhalten.

Deshalb setzen sich heute die Fahrzeugelektronikarchitekturen aus einer Vielfalt von Prozessorkernen und Hardwarebeschleunigern zusammen. Anwendungsprozessoren, die auf der ARM-Cortex-A-Familie oder der Intel-x86-Architektur basieren, sind für die Ausführung großer Betriebssysteme und Frameworks (einschließlich Linux, Android oder Adaptive-AUTOSAR) ausgelegt. Mikrocontroller und Echtzeitprozessoren hingegen eignen sich für Echtzeitanwendungen. Sie bewältigen hohe Interrupt-Lasten, starten extrem

schnell, erreichen eine sehr hohe Zuverlässigkeit und unterstützen ASIL-Levels, die mit großen Anwendungsprozessoren nur schwer zu erreichen sind. Die Mikrocontroller und Echtzeitprozessoren basieren auf der ARM-Cortex-M- oder Cortex-R-Familie, Renesas RH850, Infineon Tricore und anderen Architekturen. Normalerweise werden sie verwendet, um klassische AUTOSAR-basierte Systeme oder spezialisierte Echtzeitbetriebssysteme auszuführen.

Wegen der unterschiedlichen Anforderungen werden die Softwaremodule der Fahrzeugfunktionen – je nach Aufgabe – verschiedenen Arten von Prozessoren und Beschleunigern zugeordnet. Ein Fahrerassistenzsystem führt zum Beispiel die Bildverarbeitung auf einem Anwendungsprozessor aus, unterstützt von Hardwarebeschleunigern, während die Einhaltung der Sicherheitsziele durch zusatzliche Software auf einem Microcontroller überwacht wird.

Die Halbleiter-Industrie bietet für bestimme Domänencontroller System-on-Chips (SoCs) an, auf denen verschiedene Prozessorkerne zusammen mit Hardware-Beschleunigern enthalten sind. SoCs für Infotainment umfassen Multi-Core-Anwendungsprozessoren (oft basierend auf ARM Cortex-A), leistungsstarke GPUs (Graphic Processing Units) und einen Mikrocontroller (z. B. basierend auf ARM Cortex-M oder Cortex-R) für Sicherheitsfunktionen oder klassisches AUTOSAR. Neue SoCs, optimiert für Domänencontroller mit Fahrerassistenzsystemen, integrieren Anwendungsprozessoren, Echtzeitprozessoren, Mikrocontroller und spezielle Beschleuniger.

Virtualisierung auf Anwendungsprozessoren

Eingebettete Virtualisierung ist eine Technologie, die es ermöglicht, die Ressourcen eines Prozessors in sicher getrennte „virtuelle Maschinen" (VMs) aufzuteilen. Jede VM kann ihr eigenes Betriebssystem („Gastbetriebssystem" genannt), Framework und Anwendungen ausführen. Der „Hypervisor" ist die Software, die die Isolation („Interferenzfreiheit") und kontrollierte Kommunikation zwischen den VMs verwaltet. Anwendungsprozessoren die z. B. der ARMv8-A-Architektur entsprechen, verfügen über integrierte Erweiterungen, die sicherstellen, dass ein Hypervisor sehr effektiv und weitgehend transparent für das Gastbetriebssystem ausgeführt wird. Die Intel-x86-Architektur hat ähnliche Erweiterungen. Darüber hinaus haben viele SoC-Anbieter GPUs ausgewählt oder Systemkomponenten hinzugefügt, die die Virtualisierung der On-Chip-Geräte erleichtern.

Diese Virtualisierungstechnologie ist bereits heute in Serie. Ein wichtiges Beispiel ist der sogenannte „Cockpit Controller", bei dem es sich um einen Domänencontroller handelt, der viele Displays im Auto antreibt und die Infotainment-Funktionalität mit einem digitalen Kombiinstrument auf einer Hardware zusammenführt. In diesem Fall ermöglicht der Hypervisor die Ausführung verschiedener Software-Frameworks (z. B. Android für das Infotainment, Linux für das Kombiinstrument und ein separates Betriebssystem für sicherheitskritische Funktionen) auf einem SoC. Auf diese Weise bietet der Cockpit Controller ein integriertes Fahrerlebnis und ist im Vergleich zu einem Multi-ECU-Ansatz kostengünstiger und flexibler.

Virtualisierung auf Mikrocontrollern?

Die Virtualisierung auf Anwendungsprozessoren in Bereichen wie Infotainment ist das Ergebnis zwei Entwicklungen:

- die Notwendigkeit, Anwendungen mit sehr unterschiedlichen Anforderungen modular auf einem Prozessor zu integrieren.
- die Verfügbarkeit neuer Prozessoren, die es wegen ihrer hohen Rechnerleistung und der Hardware-Erweiterungen ermöglichen, dass Anwendungen effizient auf einem Hypervisor ausgeführt werden können.

Jetzt findet diese Technologie den Weg auch in andere Fahrzeugdomänen, die stärker auf Mikrocontroller und Echtzeitprozessoren angewiesen sind. Denn auch hier müssen die Domänencontroller, die auf einem Mikrocontroller oder einem Echtzeitprozessor laufen, eine »

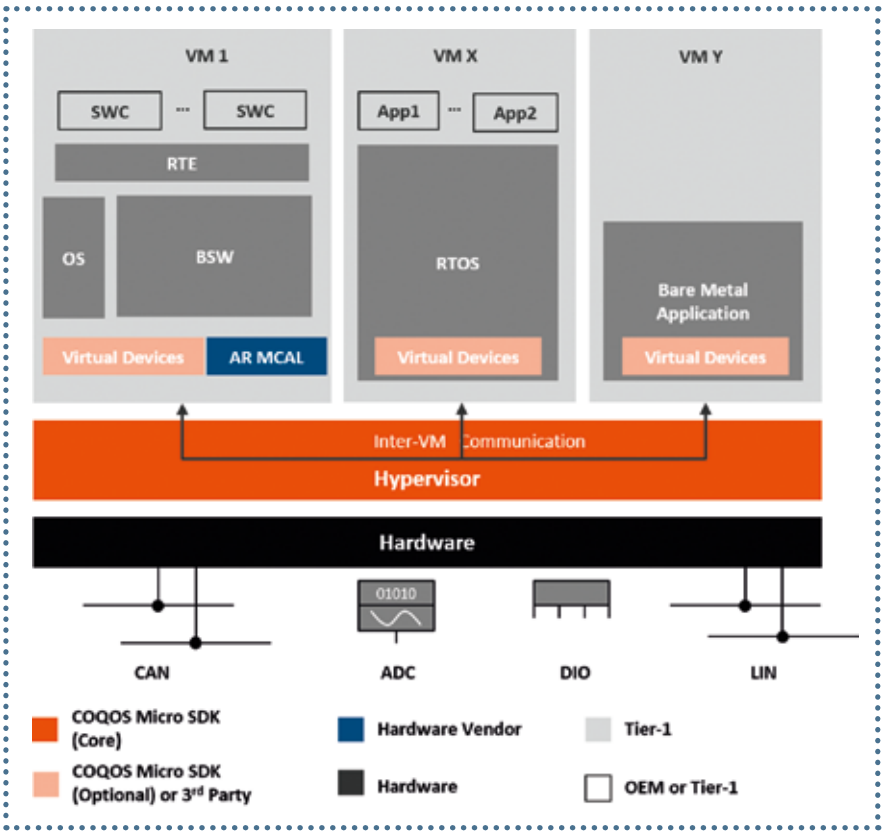

Bild 1: Mehrere virtuelle Maschinen können unterschiedliche Systeme mit verschiedenen AUTOSAR-Implementierungen oder sogar Nicht-AUTOSAR-konforme Software ausführen. (© OpenSynergy)

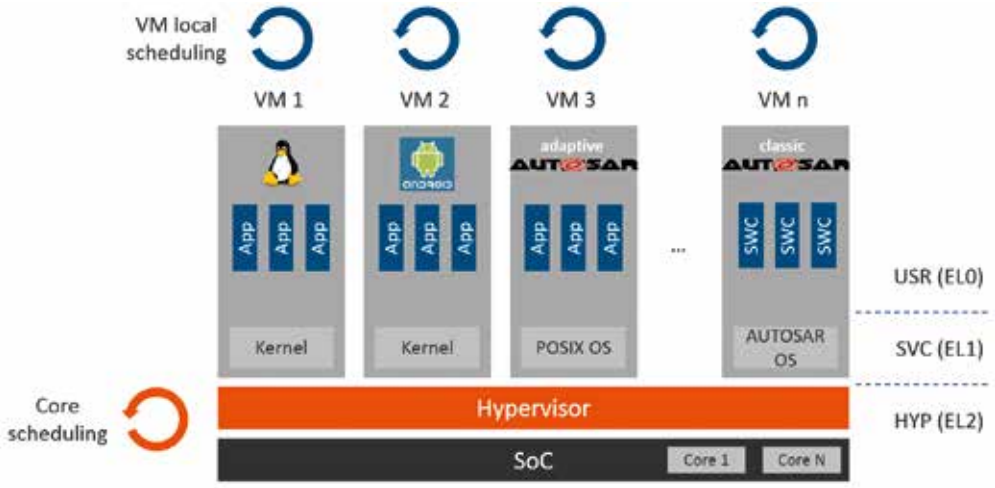

Bild 2: vCPUs-Scheduling-Task: Der Hypervisor ermöglicht es, dass jede virtuelle Maschine die konfigurierte Menge an CPU-Zeit erhält, während der VM-Scheduler die bereitgestellte CPU-Zeit Tasks entsprechend den Prioritäten zuweist. (© OpenSynergy)

immer größere Menge an Software integrieren. Diese Software wird oft nach verschiedenen funktionalen Sicherheitsstufen entwickelt oder von verschiedenen Lieferanten bezogen. Zwischen diesen Systemen muss aber absolute Störungsfreiheit sichergestellt werden. Mit steigendem Software-Umfang muss außerdem die Modularität vom Entwicklungsprozess bis hin zu Software-Updates im Bereich Aftersales reichen. Und es muss möglich sein, eine Softwarefunktion zu aktualisieren, ohne dass andere Funktionen gestört werden und auch ohne dass eine vollständige Neuqualifizierung des gesamten Geräts erforderlich wird.

Beispiel Bodycontroller

Ein konkretes Beispiel findet sich in der Body-Domäne. Ein Bodycontroller führt auf einem einzigen Mikrocontroller sowohl Funktionen aus, die bezüglich ihrer Funktionalität sicherheitskritisch sind (z. B. das Energiemanagement der Body-Domäne) als auch solche, die sicherheitskritisch bezüglich des Angriffsschutzes sind (z. B. das Entsperren des Autos). Dann gibt es noch Funktionen, die keine Sicherheitsanforderungen haben, wie zum Beispiel die Innenraumbeleuchtung. Diese Funktionen müssen unabhängig voneinander auf einem Domänencontroller integriert werden. Und nicht nur das: Es sollte auch möglich sein, ein Software-Update von unkritischen Funktionen (z. B. Innenbeleuchtung) durchzuführen, ohne sicherheitsrelevante Funktionen (z. B. Energiemanagement) zu beeinträchtigen.

Bis zu einem gewissen Grad bietet das klassische AUTOSAR eine solche Trennung an. Es unterstützt die gemeinsame Integration von Softwaresystemen auf unterschiedlichen ASIL-Ebenen, ohne dass ein Hypervisor erforderlich wäre. AUTOSAR nimmt die Isolation der Systeme auf der Betriebssystemebene vor und die darauf ausgeführten Anwendungen bestehen aus einzelnen Softwarekomponenten. Diese Architektur eignet sich jedoch nicht für komplexe Softwaresysteme, denn hier wird die Konfiguration von AUTOSAR unüberschaubar, weil das Verhalten des Betriebssystems und der Dienste in der Basis-Software bei einer reinen AUTOSAR-Architektur zentral definiert werden müssen. Das sprengt die Modularität. Außerdem erfordert das

i Forschungsprojekt ARAMiS II

OpenSynergy bringt sich mit der Entwicklung der Virtualisierungstechnologie COQOS Micro SDK in das bundesweite Forschungsprojekt ARAMiS II ein. ARAMiS II wird vom Bundesministerium für Bildung und Forschung (BMBF) mit rund 15 Millionen Euro gefördert und vom Karlsruher Institut für Technologie (KIT) koordiniert.

Mitglieder sind Hersteller und Zulieferer aus der Automobilbranche, der Luftfahrttechnologie und dem Industriesektor sowie Software-Firmen und Toolhersteller. Anhand von Demonstratoren zeigte das Verbundprojekt ARAMiS bereits im März 2015, dass Mehrkernprozessoren sich grundsätzlich für sicherheitskritische Anwendungen eignen. Daran anknüpfend hat sich seit dem 1. Oktober 2016 ARAMiS II als Anschlussprojekt zum Ziel gesetzt, Entwicklungsprozesse, Methoden, Werkzeuge und Plattformen für die Verwendung von solchen Multicore-Architekturen für sicherheitskritische Systeme zu erarbeiten. So soll z. B. ein Entwicklungsprozess erarbeitet werden, der die Komplexität beherrscht, Safety-by-Design-Eigenschaften ermöglicht und Multicore-spezifische Aspekte bereits früher und konsequenter auf höheren Abstraktionsebenen berücksichtigt. COQOS Micro SDK ermöglicht es den Konsortiummitgliedern, die Ressourcen der Prozessoren zwischen Anwendungen mit unterschiedlicher Kritikalität aufzuteilen.

Am 21. Juni 2018 wurde der Projektstand in Stuttgart der Öffentlichkeit vorgestellt. Die finalen Ergebnisse des Verbundprojekts werden dazu beitragen, Multicore-Prozessoren im industriellen Alltag effizient einzusetzen. Die erarbeiteten Lösungen werden somit auch der weltweiten Automobilindustrie zu Gute kommen.

AUTOSAR-Design, dass alle Anwendungen dem AUTOSAR-Standard – und zwar der gleichen Version von AUTOSAR – entsprechen. Ein weitere Einschränkung bei der Integration unterschiedlicher System nach AUTOSAR ist, dass das resultierende System wegen des AUTOSAR-Entwicklungsprozesses monolithisch ist und somit keine modularen Software-Updates zulässt.

Der Hypervisor fügt eine zusätzliche Ebene der Entkopplung ein und unterstützt damit eine kritische erste Ebene der Trennung in Entwicklung, Konfiguration, Integration und Software-Aktualisierung. Innerhalb einer virtuellen Maschine wird in vielen Fällen ein AUTOSAR-basiertes System (das eine zweite Trennebene bereitstellt) verwendet. Mehrere virtuelle Maschinen können dadurch unterschiedliche Systeme mit verschiedenen AUTOSAR-Implementierungen oder sogar Nicht-AUTOSAR-konforme Software ausführen.

Zusätzlich zu Anforderungen neuer Softwaresysteme, die nicht durch vorhandene Technologien wie die klassischen AUTOSAR-Standards gelöst werden können, verfügen neue Generationen von Mikrocontrollern und Echtzeitprozessoren über integrierte Erweiterungen, die das Ausführen eines Hypervisors sehr effizient machen. Die Hardware-Erweiterungen, die seit vielen Jahren in Anwendungsprozessoren verfügbar sind, werden jetzt auch bei Mikrocontrollern eingeführt. Ein gutes Beispiel ist die ARMv8-R-Architektur. Hier wurde für die Virtualisierung Erweiterungen hinzugefügt, wie z. B. eine zweite MPU (Memory Protection Unit), die vom Hypervisor gesteuert wird. Der ARM-Cortex-R52-Kern verwendet diese Architektur und NXP hat sie für neue Controller wie den NXP S32S übernommen.

OpenSynergy entwickelt seit einigen Jahren eine Variante seines Hypervisors für Mikrocontroller. Dieses Produkt nennt sich COQOS Micro SDK. Es ist der erste Hypervisor, der die Virtualisierungserweiterungen in der ARMv8-R-Architektur nutzt und die nächste Generation von Mikrocontrollern, die auf dieser Architektur basiert, unterstützt.

Wie funktioniert die Virtualisierung auf MCUs?

Die zentrale Komponente des COQOS Micro SDK ist der Hypervisor. Seine wichtigste Aufgabe ist die Sicherstellung der Interferenzfreiheit zwischen virtuellen Maschinen (gemäß ISO 26262) bis zur höchsten Stufe ASIL-D. Dabei sorgt der Hypervisor für eine zeitliche und eine räumliche Trennung von virtuellen Maschinen, indem er eine dedizierte Speicherschutzeinheit (MPU) verwendet. Er ist auch für die Zuordnung exklusiver Speicherbereiche zu einer virtuellen Maschine und für die Zugriffsberchtigung von virtuellen Maschinen auf die Peripheriegeräte zuständig. Falls die Hardware zwei MPUs (pro Kern) enthält, kann das Echtzeitbetriebssystem einer virtuellen Maschine die First-Stage-MPU verwenden, um in-

nerhalb dieser virtuellen Maschine Schutz zu ermöglichen. Für eine vollständige Trennung muss der Speicherraum auch vor Störungen durch andere Busmaster, wie z. B. der DMA-Controller (Direct Memory Access), geschützt werden. Um die zugänglichen Speicherbereiche zu begrenzen, stellen die meisten SoC-Hersteller individuelle Lösungen zur Verfügung.

Wenn mehrere virtuelle Maschinen einen physischen Kern gemeinsam nutzen, kann der Hypervisor unterschiedliche Echtzeit-Scheduling-Strategien verwenden, um zwischen virtuellen Maschinen zu wechseln. Die Sicht der virtuellen Maschinen auf die CPU-Zeit auf einem physischen Kern wird mittels einer sogenannten virtuellen CPU (vCPU) abstrahiert. Das Task-Scheduling ist als hierarchisches Scheduling angelegt. Der Hypervisor ermöglicht es, dass jede virtuelle Maschine die konfigurierte Menge an CPU-Zeit erhält, während der VM-Scheduler die bereitgestellte CPU-Zeit Tasks entsprechend den Prioritäten zuweist (Bild 2).

Ein weiterer wichtiger Aspekt für die zeitliche Trennung ist die Verwaltung von Interrupts. In den meisten Fällen weist der Hypervisor bestimmten virtuellen Maschinen Interrupts zu. Möglicherweise muss der Hypervisor auch einige Interrupts zuerst behandeln und dann die virtuellen Maschinen benachrichtigen. Das trifft z. B. dann zu, wenn sich mehrere virtuelle Maschinen einen physischen Kern teilen.

Die ARMv8-R-Architektur unterstützt eine zusätzliche Berechtigungsebene für den Hypervisor sowie die Virtualisierung von Core Timern. Dies erleichtert die Implementierung von 2-Level-Schedulern. Darüber hinaus unterstützt der generische Interrupt-Controller (GIC) von ARM die Virtualisierung, sodass der Hypervisor Interrupts direkt an virtuelle Maschinen weiterleiten und Interrupts virtualisieren kann. ARM-Virtualisierungserweiterungen erleichtern auch den Kontextwechsel zwischen virtuellen Maschinen.

Darüber hinaus muss der Hypervisor Mittel für eine effiziente und sichere Kommunikation zwischen virtuellen Maschinen bereitstellen. Der Hypervisor kann die vollständigen Kommunikationsmechanismen einkapseln, was konzeptionell ähnlich zu dem Inter-OS-Anwendungs-Kommunikator (IOC) von AUTOSAR ist. Alternativ kann der Hypervisor die grundlegenden Mechanismen bereitstellen, die zum Einrichten eines Kommunikationskanals für virtuelle Maschinen erforderlich sind: gemeinsam genutzter Speicher und ein Benachrichtigungsmechanismus zwischen den virtuellen Maschinen. In diesem Fall führen die virtuellen Maschinen die entsprechenden Kommunikationsmechanismen aus.

Die Virtualisierung ist darüber hinaus ein Schritt in Richtung modularer Software-Updates. Im Gegensatz zu AUTOSAR-OS-Anwendungen können virtuelle Maschinen unabhängig voneinander erstellt werden und der jeweilige Binärcode kann unabhängig von der Hardware aktualisiert werden.

Vorteil gegenüber Nicht-Hypervisor-Methoden?

Der Einsatz von Virtualisierungstechnologie bringt zahlreiche Vorteile für die Integration von Softwaresystemen in das Fahrzeug:

- Die Virtualisierung macht es einfacher, Störungsfreiheit durch zeit- und räumliche Trennung zu erreichen.
- Durch Virtualisierung können unabhängig entwickelte Software-Partitionen auf demselben Steuergerät ausgeführt werden. Die Software-Partitionen können unterschiedliche Software-Stacks verwenden.
- Virtualisierung ermöglicht die Konsolidierung von Software von mehreren Legacy-Steuergeräten in ein neueres, leistungsfähigeres Steuergerät.
- Neue Funktionen und die Einführung von Multi- und Many-Core-Systemen erhöhen die Softwarekomplexität. Die Analyse von Echtzeitverhalten wird schwieriger. Da der Hypervisor einen strikten Timing-Schutz erzwingt, wird eine zeitliche Interferenz zwischen Softwarekomponenten in verschiedenen virtuellen Maschinen vermieden. Dadurch wird ein einfacheres Verständnis des Systems ermöglicht, welches auf der Funktionsebene in Partitionen zerlegt wurde.
- Die Virtualisierung ermöglicht neue Workflows in der Softwareentwicklung, bei denen verschiedene Anbieter Software für verschiedene virtuelle Maschinen parallel entwickeln können. Dies ermöglicht OEMs einen flexibleren Ansatz und senkt die Hardwarekosten.
- Durch unabhängige und modulare Softwareaktualisierungen kann der Aufwand zur Neuqualifizierung von Softwarepartitionen erheblich reduziert werden, insbesondere wenn die Änderungen gering sind.

Fazit

Eingebettete Virtualisierung, eine Technologie, die bereits auf Anwendungsprozessoren in Fahrzeugdomänen wie Konnektivität und Infotainment in der Serie ist, kann jetzt auch für Mikrocontroller und Echtzeitprozessoren eingesetzt werden. Zukünftige Generationen von Mikrocontrollern, wie die, die auf der ARMv8-R-Architektur basieren, haben eingebaute Hardware-Erweiterungen, um die Virtualisierung einfacher und effektiver zu machen. Die Software-Technologie wird verfügbar sein, sobald diese neuen Prozessoren auf den Markt kommen. OpenSynergy bringt sich in das Forschungsprojekt ARAMiS ein und stellt eine Lösung bereit, die der Automobilindustrie weltweit zu Gute kommt. ■ (oe)

» **www.opensynergy.com**

» **www.hanser-automotive.de/6930895**

Hier finden Sie die Download-Version des Beitrags.

Stefaan Sonck Thiebaut ist CEO von OpenSynergy. Der Mitbegründer des Unternehmens ist Ingenieur, hat an der Stanford University (USA) promoviert und hat mehr als 20 Jahre Erfahrung in der Embedded-Software-Entwicklung.