

ECHTZEITBETRIEBSSYSTEME AUF MULTICORE

Sicherheit, Virtualisierung und der Einzelprozessor

In den frühen Tagen der Embedded-Software hatten Bausteine in der Regel eine spezielle Funktion, und jedes Ereignis, das asynchron eintrat, wurde von einem Interrupt erfasst. Dadurch war die Programmierung des Systems relativ unkompliziert, obwohl die Software in den Maschinenbefehlen des jeweiligen Prozessors geschrieben wurde. Auch auf moderneren Multifunktionsbausteinen lassen sich Echtzeitbetriebssysteme problemlos einsetzen, doch Mehrkernprozessoren stellen eine große Herausforderung dar.

ROBERT DAY

Die heutigen eingebetteten Systeme sind wahrhaftig multifunktional, und viele entsprechende Funktionen und Geräte werden auf einem einzigen Hardwareteil zusammengefasst. Das bedeutet, dass die eingebettete Software einen großen Teil der Komplexität, Zeiteinteilung, Ereignisverarbeitung und Sicherheit bereitstellen muss. Bisher setzten Softwareentwickler Echtzeitbetriebssysteme (RTOS) für Multifunktionsbausteine ein. Nun aber, da die Anwendungstypen, die auf einem Einzelsystem laufen, immer ungleichartiger werden, und mit der Einführung von Mehrkernprozessoren wird die Bewältigung aller Softwareanforderungen zu einer großen Herausforderung für ein einzelnes RTOS.

Für solch komplexe Systeme gibt es eine Lösung, und eigentlich ist es eine Kombination von Techniken, die bestimmt, wie die Entwicklung eingebetteter Software in vorhersehbarer Zukunft aussehen wird: ein Separation-Kernel und ein Embedded-Hypervisor. Der Separation-Kernel ist eine moderne Version eines Echtzeitbetriebssystems, das fähig ist, Zeit und Speicher für die verschiedenen im System laufenden Softwareanwendungen sicher und zuverlässig zu partitionieren. Die Anwendungen laufen in ihren eigenen virtuellen Maschinen, die voneinander durch den Separation-Kernel und die

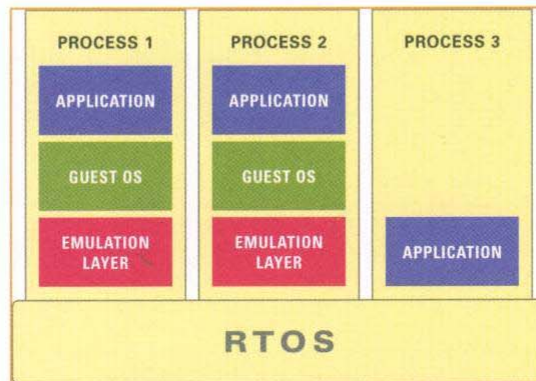


Bild 1: Der Emulationsansatz leidet unter der Ineffizienz mehrerer Schichten und der mangelhaften Kommunikation zwischen den Prozessen

Speicherverwaltungseinheit des Prozessors geschützt sind. Wenn eine Anwendung hinsichtlich Zeit oder Raum hinterher hinkt, laufen die anderen Anwendungen korrekt weiter, und der Betrieb des Systems wird nicht beeinträchtigt. Partitionierte Betriebssysteme kommen bereits seit einiger Zeit in sicherheitskritischen Systemen etwa in der Luftfahrtelektronik zum Einsatz, der Separation-Kernel fügt jedoch zwei neue Komponenten hinzu: Sicherheit und Multiprozessor-Unterstützung.

Anwendungen sicher separieren

In einem sicherheitskritischen System sorgt ein partitioniertes Betriebssystem dafür, dass jeglicher Fehlerzustand auf der jeweiligen Anwendung isoliert bleibt

und keine anderen Anwendungen beschädigen oder kontaminieren kann, die auf derselben Hardware laufen. Gleichzeitig gewährleistet es auf der Grundlage vorbestimmter Prioritäten und Zeitzuweisungen, dass jede Anwendung ihre erforderliche Zeit bekommt, unabhängig davon, was sonst noch passiert. In einem System, das eine Sicherheitsfunktion erfordert, muss der Separation-Kernel tückische Fehler oder Angriffe stoppen können, die durch eine Bruchstelle in einer der Anwendungen in das System eingedrungen sind. Durch die zusätzliche Vielschichtigkeit im Kern kann der Benutzer Sicherheitsrichtlinien für das System festlegen, die Kommunikationsflüsse zwischen bestimmten Anwendungen zulassen und andere ablehnen und so schnell einen Sicherheitsplan des

Systems aufbauen. Dies wurde für Militär- und Regierungssysteme entwickelt, kann aber auch für andere Segmente gelten, die mit möglichen Systemhackern fertig werden oder heikle Informationen wie medizinische oder finanzielle Aufzeichnungen überwachen müssen.

Der Separation-Kernel kann auch bei einer weiteren interessanten Herausforderung helfen, nämlich der Entwicklung von Embedded-Software für Multicore-Systeme. Die traditionellen Echtzeitbetriebssysteme waren gut darin, Funktionen auf einem einzelnen Gerät zu verteilen, da sie alle Ressourcen steuern und ordnen können, die der Einzelprozessor steuern soll. In einem Mehrkernsystem besteht die Notwendigkeit, Ressourcen und Anwendungen zu steuern und zu synchronisieren, die über mehrere Kerne verteilt sind. Zwar kann man durch Ausführen mehrerer Instanzen eines traditionellen Echtzeitbetriebssystems verschiedene Anwendungen über verschiedene Kerne verbreiten, doch dieser Ansatz ist typischerweise etwas eingeschränkt bei der Kommunikation zwischen und der Synchronisation über diese Anwendungen. Ein gut gestalteter Separation-Kernel kann sogar eine Instanz seiner selbst in einem Mehrkernsystem laufen lassen und unter Verwendung der oben beschriebenen Mechanismen zur Partition und sicheren Kommunikation die Partitionierung des Systems über mehrere Kerne ermöglichen sowie Kommunikationswege und Synchronisationsmechanismen zwischen den Anwendungen zulassen, unabhängig davon, auf welchem Kern sie laufen.

Virtuelle Software

Separation-Kernel haben ein hohes Maß an Flexibilität und bieten dennoch einen höheren Grad an Sicherheit als traditionelle eingebettete Echtzeitbetriebssysteme; sie werden wahrscheinlich über eine breite Palette verschiedener Embedded-Anwendungen auf vielen verschiedenen Hardwareplattformen weite Verbreitung finden. Richtig interessant wird es, wenn ein Separation-Kernel mit einem Software-Hypervisor kombiniert wird. Die Hypervisor-Technik,

auch bekannt als Software-Virtualisierung, hat sich in den letzten Jahren in der IT-Industrie weit verbreitet, weil dadurch Benutzer oder IT-Abteilungen viele verschiedene Betriebssysteme und Anwendungen auf bestehenden Hardwareplattformen laufen lassen können. Da Echtzeitausführung und Determinismus nicht so wichtig sind wie eine einfache Handhabung zur Ausführung verschiedener Betriebssystemumgebungen, wurde der traditionelle IT-Hypervisor so entwickelt, dass er alle Merkmale enthielt, jedoch etwas umfangreich und schwerfällig und sicherlich für die meisten Embedded- und Echtzeit-Systeme nicht geeignet war. Ein Embedded-Hypervisor ist jedoch so gestaltet, dass er kleiner und effizienter als seine IT-Vettern ist, jedoch immer noch den Vorteil hat, dass er verschiedene Gast-Betriebssysteme über einem Echtzeitbetriebssystem ausführen kann. Jedoch ist an dieser Stelle eine Klarstellung notwendig, um zwischen Software-Virtualisierung (Hypervisor) und Software-Emulation (Emulator) zu unterscheiden. Ein Hypervisor stellt Verknüpfungen in das zugrunde liegende Echtzeitbetriebssystem her, das die Hardwareausführung steuert und eine Schicht darstellt, über dem ein Gast-Betriebssystem laufen kann. Das Gast-Betriebssystem läuft immer noch auf dem zugrunde liegenden Prozessoraufbau der Zielhardware, aber der Hypervisor steuert, wie diese Anweisungen ausgeführt werden, gleich einer Softwareausführung und Speicherverwaltung. Software-Emulation dagegen ist eine Anwendung, welche die Prozessoranweisungen in Software emuliert, die vom Gast-Betriebssystem angefordert wird. Der Emulationsansatz braucht nicht eng mit dem

ROBERT DAY

ist Vice President
of Marketing
bei LynuxWorks

zugrunde liegenden Echtzeitbetriebssystem verknüpft zu sein; er läuft nur als eine Anwendung auf dem Echtzeitbetriebssystem. Die Nachteile liegen jedoch in einem dramatischen Verlust an Performance und einer mangelhaften Kommunikation zwischen dem Gast-Betriebssystem und dem zugrunde liegenden Separation-Kernel. Dadurch gehen viele Vorzüge des oben beschriebenen Separation-Kernel-Ansatzes verloren. Die Kombination aus Hypervisor und Separation-Kernel gibt der Entwicklung eingebetteter Software eine neue Dimension, weil dadurch mehrere Gast-Betriebssysteme und Anwendungen auf einer Hardware laufen können, sicher getrennt durch ein Echtzeitbetriebssystem. Unter Verwendung von Mehrkernprozessoren ermöglicht ein Separation-Kernel und ein Hypervisor Programmierern, ihr System mit Gast-Betriebssystemen, die über mehrere Kerne verteilt sind, zu partitionieren, jedoch die Kommunikation zwischen ihnen mit einem gemeinsamen Separation-Kernel steuern zu können.

Virtuelle Hardware

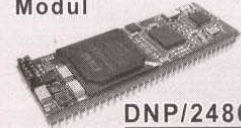
Vor der Beschreibung eines Fallbeispiels für diese Technik im Embedded-Bereich soll noch eine weitere Technik vorgestellt werden: die Hardware-Virtualisierung. Viele neue Mehrkernprozessoren bieten jetzt Hardware-Virtualisierung an, um es Hypervisoren zu ermöglichen, Gast-Betriebssysteme effi-

zienter auszuführen. Obwohl ein großer Teil dieser Technik für die IT-Welt aufgebaut wurde, bietet sie einige Vorzüge für Embedded-Anwender. Weil die Hardware die Software-Virtualisierung unterstützt, zum Beispiel durch die auf Intel-Prozessoren vorhandene VT-D- und VT-X-Technik, kann der Hypervisor viele seiner Anweisungs- und Datenverarbeitungsschritte der Hardware überlassen. Dies steigert die Leistung laufender Gast-Betriebssysteme auf eine fast systemeigene Leistung und beeinträchtigt nicht die Echtzeiteigenschaften der Gastanwendungen, obwohl diese in einer virtualisierten Umgebung laufen. Gleich wie eine Speicherverwaltungseinheit schützt Hardware-Virtualisierung die Gast-Betriebssysteme auch vor störender Beeinflussung anderer Teile des Systems, ohne dass sie sich völlig auf die Softwareverwaltung stützen müssten. Dies erhöht die Sicherheit des Systems, ohne die Echtzeitleistung und den Determinismus zu beeinträchtigen. Ein interessantes Anwendungsbeispiel dieser Technik könnte sich bald in vielen Embedded-Industriezweigen von Industriesteuerungen bis hin zu Systemen in der Medizin, Finanz, dem Militär und Automobilbereich verbreiten. Dieses Beispiel führt ein Echtzeitsystem (vielleicht eine Datensammlung aus einem Sensorensystem in einem Netzwerk) mit einem traditionelleren Betriebssystem auf der Basis einer grafischen Benutzeroberfläche wie Windows oder Linux zusammen. Eine solche Kombination auf einer einzigen Hardwarebasis war immer eine Herausforderung, da beide Systeme einzeln die Steuerung der zugrunde liegenden Hardware angefordert haben, was sowohl die Aspekte der Echtzeitleistung als auch die Sicherheit und Integrität beider Systeme beeinträchtigen konnte. Der Separation-Kernel und der Hypervisor ermöglichen dem Echtzeitbetriebssystem und dem Betriebssystem mit grafischer Benutzeroberfläche, jeweils in ihrer eigenen Partition zu laufen. Jedes arbeitet möglicherweise auf einem eigenen Kern in einem Mehrkernsystem, beide sind voneinander geschützt, lassen aber eine sichere Kommunikation zwischen beiden zu. Da die

Softwareumgebung virtualisiert ist, laufen etwaige Altanwendungen ebenfalls wie zuvor, sodass die Notwendigkeit einer Neukodierung in großer Menge wegfällt. Das Echtzeitbetriebssystem wird in Echtzeit laufen können, weil der zugrunde liegende Separation-Kernel immer noch für eine deterministische Umgebung sorgt, und das Betriebssystem mit grafischer Benutzeroberfläche wird annehmen, dass es die vollständige Kontrolle über den Prozessor hat, und wird ausgeführt, als ob es seine eigene Maschine hätte. Dieses System wird dann in der Lage sein, eine bekannte Benutzerumgebung zur Steuerung und Anzeige der Daten anzubieten, während das Echtzeitbetriebssystem die Informationen ohne Beeinträchtigung sammelt. (mc)

LynuxWorks
Telefon 00 44/20 89 06 95 06
www.lynuxworks.com

Embedded Linux Modul



- 10/100 Mbps Ethernet LAN
- 480 Mbps USB 2.0 Host
- Java 6 Support

www.ssv-embedded.de

Matthias
Mansfeld
Elektronik

Leiterplattenlayout
mit Zukunftsdistanz • Flex, COB, Hochstrom etc.
Neuhardstr. 3 • 85540 Haar
Tel.: 089/462 0093-7, Fax: -8
<http://www.mansfeld-elektronik.de>

POLY RACK
TECH-GROUP

Ihr Spezialist für
individuelle Gehäuse-
lösungen, Systeme und
Kunststofftechnik:
www.polyrack.com

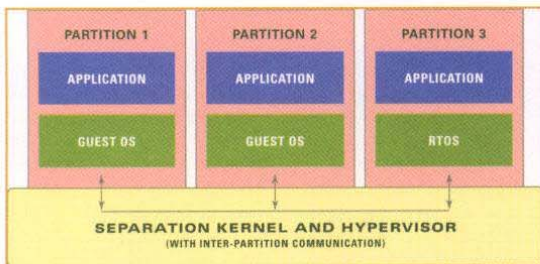


Bild 2: Eine echte Software-Virtualisierung bietet Echtzeitleistung und sichere Kommunikation zwischen den Partitionen