Software Design Automation für effiziente Multicore-Nutzung

Software-Verteilung unter Leistungs- und Energieeinschränkungen

Maximilian Odendahl, CEO, Silexica GmbH

Die Entwicklung von Software für neuartige Multicore-Systeme stellt zahlreiche Industrien im Embedded High Performance Bereich vor neuartige Herausforderungen. Im Vergleich zu bestehenden Singlecore-Systemen, erfordert der Einsatz moderner Multicore-Systeme neben der funktionalen Verifikation und Singlecore-Optimierung eine optimale Verteilung der Software auf die verschiedenen Prozessorkerne der Zielplattform. Dabei gilt es nicht nur zeitliche und funktionale Vorgaben zu berücksichtigen, sondern idealerweise auch den Leistungs- und Energieverbrauch zu minimieren. Selbst für kleine Anwendungen und Zielsysteme ergibt sich daher eine Vielzahl von möglichen Lösungen, deren Entwicklung und Bewertung bisherige, hauptsächlich manuelle, Entwicklungsansätze an ihre Grenzen bringen. Abhilfe können neuartige Algorithmen und Tools aus dem Bereich der Software Design Automation schaffen. Im Folgenden wird hierzu ein neuartiger Tool-basierter Lösungsansatz näher vorgestellt und die Vorteile anhand eines Anwendungsbeispiels aus der drahtlosen Telekommunikation demonstriert.

Multicore-Programmierung und ihre Herausforderungen

Die Entwicklung von Software für neuartige Multi- und Manycore-Systeme stellt zahlreiche Industrien im Embedded High Performance Bereich vor zusätzliche Herausforderungen. Bedingt wird dies durch die steigende Komplexität der zur Verfügung stehenden Hardwaresysteme, aber auch durch steigende Anforderungen. Hier wird neben hoher Performance für eine Vielzahl von Anwendungsszenarien oft gleichzeitig eine minimale Leistungsaufnahme gefordert. Verschärft wird dieses Problem zusätzlich durch immer kürzer werdende Entwicklungszyklen und gestiegene Sicherheitsanforderungen. Die daraus entstandene „Effizienzlücke“ zwischen den herausragenden Fähigkeiten moderner Hardware und der Fähigkeit, diese effizient durch Software auszunutzen, ist mit der zurzeit üblichen manuellen Systemauslegung und Programmierung nicht wirtschaftlich zu schließen.

Die Herausforderungen sind hierbei neben der Parallelisierung von bestehendem (oft Jahrzehnte alten) C/C++ Code, vor allem eine optimale Verteilung bereits paralleler Software unter Berücksichtigung von zeitlichen und funktionalen Einschränkungen. So besteht z.B. eine Long Term Evolution (LTE) Layer 3 Implementierung aus bis zu 50.000 separaten Tasks, die effizient auf hunderte von heterogenen Prozessorcores (RISCs, DSPs, Hardware-Beschleuniger, …) verteilt werden müssen. Hierbei muss nicht nur die benötigte und verfügbare Rechenleistung berücksichtigt werden, sondern in wachsenden Maß auch die Kommunikation zwischen den Prozessorkernen, die zunehmend zum Flaschenhals für die Systemleistung wird. Um alle zeitlichen und funktionalen Einschränkungen zu erfüllen, ist daher eine gemeinsame Berücksichtigung von Rechenleistung und Kommunikation notwendig.

Lösungsansätze durch Software Design Automation

Für verschiedene Problemstellungen, wie das oben beschriebene Beispiel aus dem Bereich der Telekommunikation, ist eine manuelle Auslegung und Programmierung unter wirtschaftlichen Aspekten bereits heute nicht mehr möglich. Eine neue Generation von automatisierten Entwicklertools löst diese Probleme durch dynamische Datenflussanalyse, automatische Laufzeitabschätzung und automatische Software-Verteilung der einzelnen Tasks. Während

des gesamten Entwicklungsprozesses wird dabei die Zielplattform direkt mit einbezogen, was eine kombinierte Software- und Hardware-Optimierung ermöglicht.

Basierend auf Prozessnetzwerken, die Beziehungen zwischen parallelen Prozessen (Tasks) durch Datenkanäle explizit beschreiben, erfolgt dabei neben einer grundlegenden Anwendungsanalyse (Deadlocks, Tasklaufzeiten, Kanaldurchsätze, …) eine automatisierte Software-Verteilung der Tasks auf eine Zielplattform (siehe Abb. 1). Dabei stehen mehrere Optimierungskriterien (Performance, Speicher, Leistungsaufnahme) zur Verfügung, die unter Berücksichtigung von verschiedenen Anforderungen bzw. Einschränkungen (Latenzen, Durchsätze, Speicher, …) automatisiert iteriert und manuell feingetuned werden können.

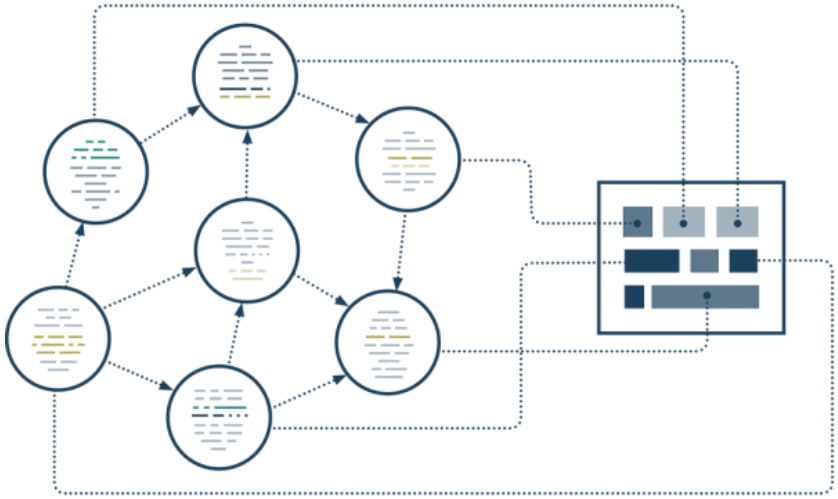

Abb. 1: Beispielhafte Verteilung eines (parallelen) Prozessnetzwerkes (links) auf eine (heterogene) Zielplattform (rechts)

Leistungsaufnahme und Energieverbrauch

Für viele Industrien ist eine reduzierte Leistungsaufnahme und/oder ein geringer Energieverbrauch bereits heute von großer Bedeutung. Beispielhaft hierfür sind neue Systeme in der Telekommunikationsbranche für die nächste Mobilfunkgeneration (5G) sowie aktuelle Systeme aus dem Bereich der automobilen Fahrerassistenzsysteme und des autonomen Fahrens zu nennen.

Die Optimierungskriterien können dabei sehr unterschiedlich sein: Wird bei der Auslegung von Basisstationen im Mobilfunk eine Reduzierung der maximalen Leistungsaufnahme angestrebt, um die bestehende Netz-Infrastruktur weiterhin nutzen zu können, steht im Automotive Bereich die Reduzierung der durchschnittlichen Leistungsaufnahme im Vordergrund um die Fahrzeugreichweite zu erhöhen und gleichzeitig die Hitzeentwicklung zu minimieren.

Für diese beiden Szenarien wurden zwei Algorithmen entwickelt:

1. Algorithmus zur Minimierung der durchschnittlichen Leistungsaufnahme:
   Ziel dieses Algorithmus ist es für jeden Task einer (parallelen) Anwendung den optimalen Prozessor und für jeden Prozessor die optimale Kombination aus Spannung und Frequenz zu bestimmen. Dabei wird iterativ sowohl die statische als auch die dynamische Leistungsaufnahme mittels der auf den Prozessor ausgeführten, Task-spezifischen, Instruktionen berücksichtigt. Weiterhin werden unterschiedliche Frequenz- und Spannungsdomänen automatisch berücksichtigt, um ungültige

Konfigurationen auszuschließen. In jedem Iterationsschritt werden die Tasks unter Berücksichtigung ihrer individuellen Laufzeiten (die zusätzlich noch je nach Prozessor variieren können) möglichst gleichmäßig auf die Plattform verteilt ohne dabei jedoch die Gesamtlaufzeit der Anwendung zu erhöhen. Laufzeitanforderungen (Latenzen, Durchsätze, …) werden in jedem Iterationsschritt separat über eine high-level Scheduler-Simulation verifiziert. Das Ergebnis der Scheduler-Simulation wird im nächsten Iterationsschritt genutzt um die nächste Plattform-Konfiguration auszuwählen.

2. Algorithmus zur Minimierung der maximalen Leistungsaufnahme:
   Ziel ist es ein gegebenes Leistungsbudget (maximale Leistungsaufnahme) während der Ausführung einer Anwendung auf einer Zielplattform nicht zu überschreiten. Dafür werden Tasks basierend auf ihrer Priorität und ihrem Leistungsaufnahmeprofil (Leistungsaufnahme über Zeit) proportional auf verschiedene Prozessoren verteilt. Task Prioritäten werden dynamisch anhand der Zugehörigkeit zum kritischen Pfade der Anwendung bestimmt. Das Leistungsaufnahmeprofil der Prozessoren wird anhand der statischen und dynamischen Leistungsaufnahme für die jeweilige Task Verteilung und die jeweilige Frequenz-/Spannungskonfiguration berechnet (s. o.). Wenn das Leistungsbudget für eine Konfiguration überschritten wird, werden im nächsten Iterationsschritt niedrigere (weniger performante) Frequenz-/Spannungs-Konfigurationen untersucht.

Die hier vorgestellten Algorithmen sind in ähnlicher Form in der Silexica Tool Suite implementiert und kommen in der folgenden Fallstudie zum Einsatz.

Fallstudie Telekommunikation

Durch die steigende Vernetzung und wachsende allgemeine Bedeutung der Informationstechnik im alltäglichen Leben, sind besonders Unternehmen in der Telekommunikationsbranche mit einem rasanten technologischen Fortschritt konfrontiert. Vorhandene Hardware wird mit jedem neuen Standard quasi obsolet und muss aufwändig neuentwickelt werden. Viele bestehende Anwendungen sind nicht für den Einsatz auf Multicore-Prozessoren optimiert. Die hohe Komplexität bestehender Systemarchitekturen macht es daher praktisch unmöglich, das Potential von vorhandenen oder neu entwickelten Systemen zu bewerten, ohne signifikante Zeit und Ressourcen in Hardware- und Software-Design, Herstellung und Prüfung zu investieren.
Durch die Nutzung von speziellen Software-Tools können Hardware-Hersteller hingegen das Potenzial für Einsparungen bei der Leistungsaufnahme und für Performance-Steigerungen bestehender Systeme untersuchen, bevor sie neue Systeme mit unbekannten Ausgang, hohem Risiko und hohen Kosten entwickeln.
Mit Hilfe der SLX Tool Suite (und den oben beschriebenen Algorithmen) konnte ein großer Hardwarehersteller aus der Telekommunikationsbranche die Performance und Leistungsaufnahme für eine bestehende Multiprozessor-Architektur detailliert modellieren und mit Bezug auf verschiedene Stromsparfunktionen analysieren.

Die heterogene Zielarchitektur hat mehr als 100 Prozessorkerne und erlaubt es Spannung und Frequenz weitgehend unabhängig für jeden Prozessor zu variieren.

Abhängig von der maximalen Anwendungslaufzeit konnte bei der Verwendung von dynamischer Spannungs-/Frequenz-Skalierung (DVFS) die maximale Leistungsaufnahme um 67 % reduziert werden. Die Energieeffizienz konnte bei nur geringfügiger Laufzeitänderung

um 24 % verbessert werden, während die mittlere Leistungsaufnahme um bis zu 32 % reduziert werden konnte.

Die Ergebnisse der Untersuchungen zeigen hohes Potenzial für Leistungsverbesserungen und Energieeinsparungen im Vergleich zu Systemen, die ausschließlich mit festen Spannungen und Clock-Gating ausgestattet sind.

Obwohl die hier gezeigten Ergebnisse natürlich nicht 1:1 auf andere Systeme übertragen werden können, so zeigen sie dennoch das Potenzial von Software Design Automation Tools: Eine deutlich reduzierte maximale Leistungsaufnahme ermöglicht Kosteneinsparungen bei der Spannungsversorgung, während die Reduktion der mittleren Leistungsaufnahme den Kühlungsaufwand reduziert und so Kosten einsparen kann.

Zusammenfassung

Die zunehmende Komplexität von modernen Multiprozessorsystemen in Kombination mit verkürzten Entwicklungszyklen und steigenden Anwendungsanforderungen haben zu einer Effizienzlücke geführt: Die effiziente Nutzung moderner Multiprozessorsysteme kann mit manueller Systemauslegung und Programmierung nur beschränkt wirtschaftlich bewerkstelligt werden. Mit steigender Plattform- und Software-Komplexität wird dieses Problem zukünftig sogar noch größer. Software Design Automation Tools können genutzt werden, um diese Lücke zu schließen und selbst komplexe Optimierungsprobleme zu lösen. In der hier gezeigten Fallstudie konnte die durchschnittliche und maximale Leistungsaufnahme für eine Multiprozessorplattform mit mehr als 100 Prozessoren vollautomatisiert signifikant reduziert werden.

Autor

Maximilian Odendahl studierte Computer Engineering an der RWTH Aachen und war nach seinem Abschluss am Institut for Communication Technologies and Embedded Systems (ICE) als Wissenschaftlicher Mitarbeiter/Doktorand mit dem Schwerpunkt der Toolentwicklung für heterogenen Multicore-Systeme tätig. Von Januar 2013 bis Dezember 2014 war er zusätzlich Oberingenieur des ICE tätig. Heute ist er geschäftsführender Gesellschafter der Silexica GmbH und spricht regelmäßig auf Konferenzen.

Kontakt

Internet: http://www.silexica.com
Email: odendahl@silexica.com