# Analyse und Optimierung von fehlertoleranten Eingebetteten Systemen mit gehärteten Prozessoren

Viacheslav Izosimov, Universität Linköping, Schweden
Ilia Polian, Universität Freiburg, Deutschland
Paul Pop, Technische Universität Lyngby, Dänemark
Petru Eles, Universität Linköping, Schweden
Zebo Peng, Universität Linköping, Schweden

## Kurzfassung

Wir stellen einen Ansatz zur Entwurfsoptimierung von fehlertoleranten harten Echtzeitsystemen vor, der Hardware- und Software-Fehlertoleranztechniken kombiniert. Es wird zwischen selektiver Härtung in Hardware und Prozessneuausführungen in Software abgewogen, um benötigte Fehlertoleranz zu geringst möglichen Kosten zu erreichen. Die vorgestellten Entwurfsoptimierungsheuristiken legen die fehlertolerante Architektur und Prozesszuordnung fest, so dass die Systemkosten minimiert, die Deadlines eingehalten und die Zuverlässigkeitsanforderungen erfüllt werden..

## 1 Einleitung

Sicherheitskritische Eingebettete Systeme müssen selbst unter Fehlern kosteneffizient und schnell arbeiten. In dieser Arbeit wird die Toleranz gegen transiente und intermittente Fehler, genannt Soft Errors, behandelt. Wir kombinieren selektive Härtung mit Software-Fehlertoleranz, um die geringst möglichen Systemkosten zu erreichen, ohne die harten Deadlines und Zuverlässigkeitsanforderungen zu verletzen. Wir setzen Prozessneuausführungen ein, um Fehler softwarebasiert zu behandeln. Um die Zuverlässigkeitseigenschaften des Systems zu bewerten, wird auf ein Verfahren zur Berechnung der Systemausfallwahrscheinlichkeit zurückgegriffen, welches. die Redundanzniveaus in Software (maximale Anzahl von Neuausführungen) und in Hardware (Umfang der selektiven Härtung) miteinander verbindet [1].

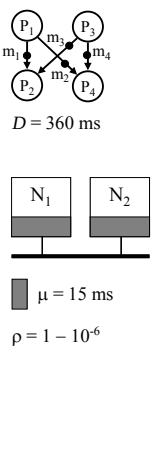
## 2 Systemmodell

Eine Anwendung wird auf Systemebene als eine Menge von gerichteten, azyklischen Graphen modelliert. Die Knoten entsprechen einzelnen (nichtpräemptiven) Prozessen, während die Kanten Abhängigkeitsbeziehungen modellieren. Die Anwendung wird auf eine Menge von Berechnungsknoten abgebildet, die an einen Bus angeschlossen sind und mit Hilfe von Nachrichten über den Bus kommunizieren.

## 3 Fehlertoleranztechniken

Zur Fehlerbehandlung werden zwei Techniken eingesetzt: Prozessneuausführung und selektive Härtung von Berechnungsknoten. Wir nehmen an, dass Fehler während der Prozessausführung auftreten und stets entdeckt werden. In diesem Fall wird der Prozess neu ausgeführt, wobei eine zusätzliche Fehlerbehandlungszeit (Overhead) μ verbraucht wird. Die Fehlererkennungs- und Fehlerbehandlungsmechanismen werden als fehlertolerant angenommen; für die Kommunikation zwischen den Prozessen wird ein fehlertolerantes Protokoll, z.B. TTP [3], unterstellt.

Die Berechnungsknoten stehen in mehreren Versionen mit unterschiedlichen Härtungsgraden ($h$-Versionen) zur Verfügung. Ein größeres $h$ steht für mehr Härtung und folglich höhere Kosten, eine niedrigere Berechnungsgeschwindigkeit und eine geringere Fehlerrate. Formal bezeichnen wir die $h$-Version des Knotens $N_j$ als $N_j^h$, ihre Kosten als $C_j^h$, die Ausführungszeit (WCET) von Prozess $P_i$ auf Knoten $N_j^h$ als $t_{ijh}$, und die Ausfallwahrscheinlichkeit von $P_i$ auf $N_j^h$ als $p_{ijh}$. Abb.1 zeigt eine Beispielanwendung mit vier Prozessen und zwei Berechnungsknoten in drei $h$-Versionen.




| $N_1$ | $h=1$ | | $h=2$ | | $h=3$ | |
|---|---|---|---|---|---|---|
| | $t$ | $p$ | $t$ | $p$ | $t$ | $p$ |
| $P_1$ | 60 | $1.2\cdot10^{-3}$ | 75 | $1.2\cdot10^{-5}$ | 90 | $1.2\cdot10^{-10}$ |
| $P_2$ | 75 | $1.3\cdot10^{-3}$ | 90 | $1.3\cdot10^{-5}$ | 105 | $1.3\cdot10^{-10}$ |
| $P_3$ | 60 | $1.4\cdot10^{-3}$ | 75 | $1.4\cdot10^{-5}$ | 90 | $1.4\cdot10^{-10}$ |
| $P_4$ | 75 | $1.6\cdot10^{-3}$ | 90 | $1.6\cdot10^{-5}$ | 105 | $1.6\cdot10^{-10}$ |
| *Kosten* | 16 | | 32 | | 64 | |

| $N_2$ | $h=1$ | | $h=2$ | | $h=3$ | |
|---|---|---|---|---|---|---|
| | $t$ | $p$ | $t$ | $p$ | $t$ | $p$ |
| $P_1$ | 50 | $1\cdot10^{-3}$ | 60 | $1\cdot10^{-5}$ | 75 | $1\cdot10^{-10}$ |
| $P_2$ | 65 | $1.2\cdot10^{-3}$ | 75 | $1.2\cdot10^{-5}$ | 90 | $1.2\cdot10^{-10}$ |
| $P_3$ | 50 | $1.2\cdot10^{-3}$ | 60 | $1.2\cdot10^{-5}$ | 75 | $1.2\cdot10^{-10}$ |
| $P_4$ | 65 | $1.3\cdot10^{-3}$ | 75 | $1.3\cdot10^{-5}$ | 90 | $1.3\cdot10^{-10}$ |
| *Kosten* | 20 | | 40 | | 80 | |

**Abb. 1** Anwendungsbeispiel

# 4 Entwurfsstrategie

Für eine Anwendung werden neben Daten aus Abb. 1 die Deadline sowie die Zuverlässigkeitsvorgabe ρ angegeben. ρ entspricht der Wahrscheinlichkeit, dass kein weder auf Hardware- noch auf Softwareebene behandelter Fehler auftritt. Gesucht sind: (1) eine Architektur, also eine Auswahl von Berechnungsknoten; (2) die Festlegung des Härtungsgrades für jeden Knoten; (3) eine Allokation von Prozessen auf die Knoten der gewählten Architektur; (4) die maximal erforderliche Anzahl von Neuausführungen auf jedem Berechnungsknoten; und (5) ein quasistatischer Ablaufplan (Schedule) der Prozesse und der Kommunikation.

Die Berechnung der Systemausfallwahrscheinlichkeit ist in [1] beschrieben. Für jeden Prozess wird die Anzahl von Prozessneuausführungen bestimmt, die notwendig ist, um die Zuverlässigkeitsvorgabe ρ zu erreichen. Diese Größe ist von den Härtungsgraden der Knoten in der Architektur abhängig. Die Optimierungsstrategie besteht aus einer Reihe von Heuristiken. Sie legt iterativ die Härtungsgrade der Knoten fest und bildet einzelne Prozesse auf Knoten ab. Dann wird der Ablaufplanplan berechnet, welcher keine Deadlines verletzt, die Zuverlässigkeitsvorgabe einhält und die Gesamtkosten der Architektur (Summe der Kosten aller Knoten) minimiert.

Abb. 2 zeigt fünf unterschiedliche Architekturen für das Anwendungsbeispiel aus Abb. 1. In Architektur a) sind Prozesse $P_1$ und $P_2$ auf Berechnungsknoten $N_1$ und Prozesse $P_3$ und $P_4$ auf Knoten $N_2$, jeweils mit Härtungsgrad $h = 2$, abgebildet. Nachrichten $m_2$ und $m_3$ werden über den Bus verschickt, während Nachrichten $m_1$ und $m_4$ innerhalb der Knoten übertragen werden. Die Ausfallwahrscheinlichkeitsanalyse ergibt, dass im ungünstigsten Fall eine Neuausführung von Prozessen $P_2$ und $P_3$ notwendig ist; die unterschiedlichen Ausführungen sind in der Abbildung als $P_{2/1}$ und $P_{2/2}$ bzw. $P_{3/1}$ und $P_{3/2}$ dargestellt. Die zusätzliche Fehlerbehandlungszeit μ ist in dunkelgrauer Farbe zwischen den Ausführungen zu sehen. Der Ablaufplan hält die Deadline ein, welche durch die vertikale Linie angedeutet ist. Die Gesamtkosten betragen 72.

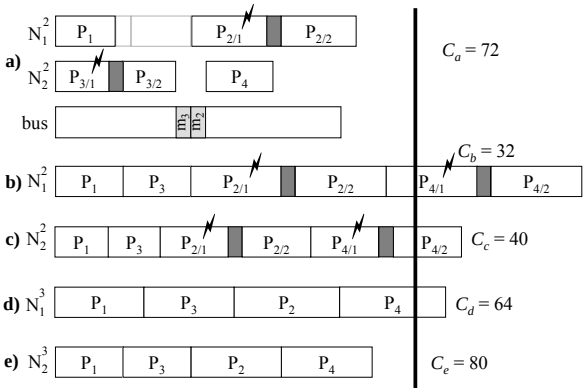



**Abb. 2** Alternative Architekturen: Fehlerbehandlung

Architekturen b) bis e) bestehen aus einem Einzelprozessor. Für Alternativen b) und e) hat die Systemausfallwahrscheinlichkeitsanalyse zwei benötigte Neuausführungen ergeben. Die ungünstigsten Szenarien, in welchen Neuausführungen für die langsamsten Prozesse benötigt werden, verletzen die Deadline. In Architekturen d) und e) werden Knoten mit dem höchsten Härtungsgrad ($h = 3$) eingesetzt; für sie werden keine Neuausführungen benötigt. Obwohl für Architektur e) ein Ablaufplan existiert, übersteigen ihre Kosten die Kosten der Architektur a). Deswegen wird Architektur a) vom Algorithmus gewählt. Die Details der Optimierungsstrategie können aus Platzgründen nicht dargestellt werden.

# 5 Experimentelle Ergebnisse

Wir haben 150 synthetische Anwendungen mit 20 und 40 Prozessen generiert. Durch die Zulassung von fünf unterschiedlichen Härtungsgraden konnte der Anteil der zulässigen Anwendungen um 55% gesteigert werden (als zulässig werden Anwendungen bezeichnet, für welche unsere Optimierungsstrategie einen Ablaufplan generieren kann, der Deadline, Zuverlässigkeitsvorgabe und Kostengrenze nicht überschreitet).

Wir haben unsere Entwurfsstrategie ferner auf ein Vehicle Cruise Controller (CC) angewendet, der aus 32 Prozessen besteht [2]. Die Architektur von CC besteht aus drei Knoten: Electronic Throttle Module (ETM), Anti-lock Braking System (ABS) und Transmission Control Module (TCM). Die Abwägung zwischen Redundanz in Hardware und Software führt zu einer Kostenersparnis von 66%.

# Danksagung

Teile dieser Arbeit wurden von der Swedish Graduate School in Computer Science (CUGS), der ARTES++ Swedish Graduate School in Real-Time Systems und der DFG im Rahmen des Projekts RealTest (Be 1176/15-2) unterstützt.

# Literatur

[1] V. Izosimov. I. Polian, P. Pop, P. Eles, and Z. Peng . “Analysis and optimization of fault-tolerant embedded systems with hardened processors”, Design Automation and Test in Europe Conf. (DATE), pp. 682–687, 2009.

[2] V. Izosimov, “Scheduling and Optimization of Fault-Tolerant Embedded Systems”, Licentiate Thesis No. 1277, Dept. of Computer and Information Science, Linköping University, 2006.

[3] H. Kopetz, G. Bauer, “The Time-Triggered Architecture”, Proc. of the IEEE, 91(1), 112–126, 2003.