TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Informatik Institut für Technische Informatik, Professur für VLSI-Entwurfssysteme, Diagnostik und Architektur

Belegverteidigung

Studie zum Einsatz eines Network-on-a-Chip für eine Many-Core-Java-Bytecode-Architektur

Tony Müller

Dresden, 11.07.2012

Inhalt

1 Motivation

Begriffe

Many-Core-Architekturen

- Trend zu immer stärkerer Parallelisierung, speziell auf Threadebene
- immer mehr Kerne auf einem Chip
- Weiterentwicklung von Multi-Cores
- keine genaue Definition von „many" → meist deutlich mehr als 10 Kerne je Chip

Networks-on-a-Chip

- Ursprung in SoCs zur Verbindung heterogener Komponenten auf einem Chip
- bekannte Netzwerk-Konzepte auf Chipebene anwenden
- Pakete, Topologie, Routing, Switching, Router etc.

→ NoC zur Verbindung vieler (SHAP-)Kerne nutzen

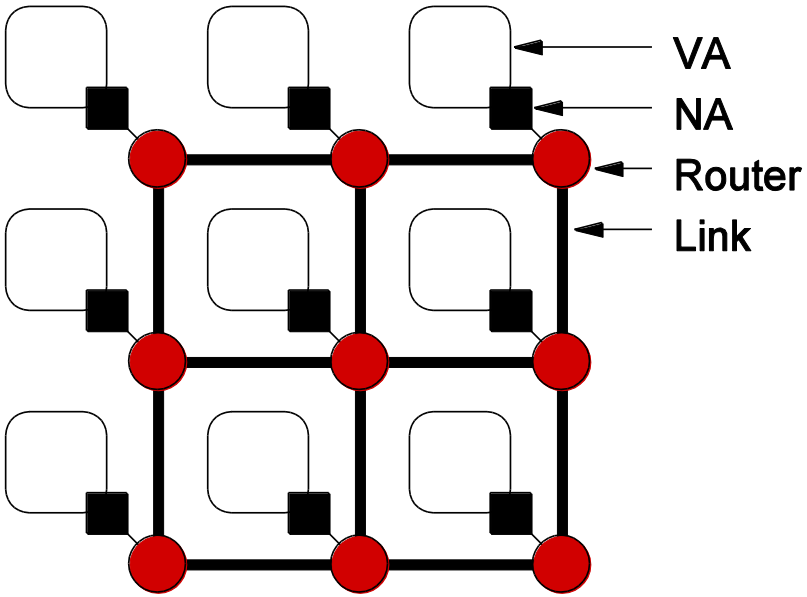
1 Motivation

Aufgabenstellung

- Literaturstudium zu Struktur, Realisierung und Parametrierung von NoCs
- Literaturstudium zum Aufbau von Many-Core-Architekturen
- Identifikation von geeigneten NoC-Konzepten für Many-Core-Architekturen
- Auswahl geeigneter Ansätze für Many-Core-SHAP. Zusammenstellung verschiedener Szenarien und deren möglicher Parametrierung.
- Bewertung der Szenarien bzgl. zu erwartender praktischer Eignung

2 Grundlagen

Networks-on-a-Chip - Komponenten [BDM06] [Bje05]

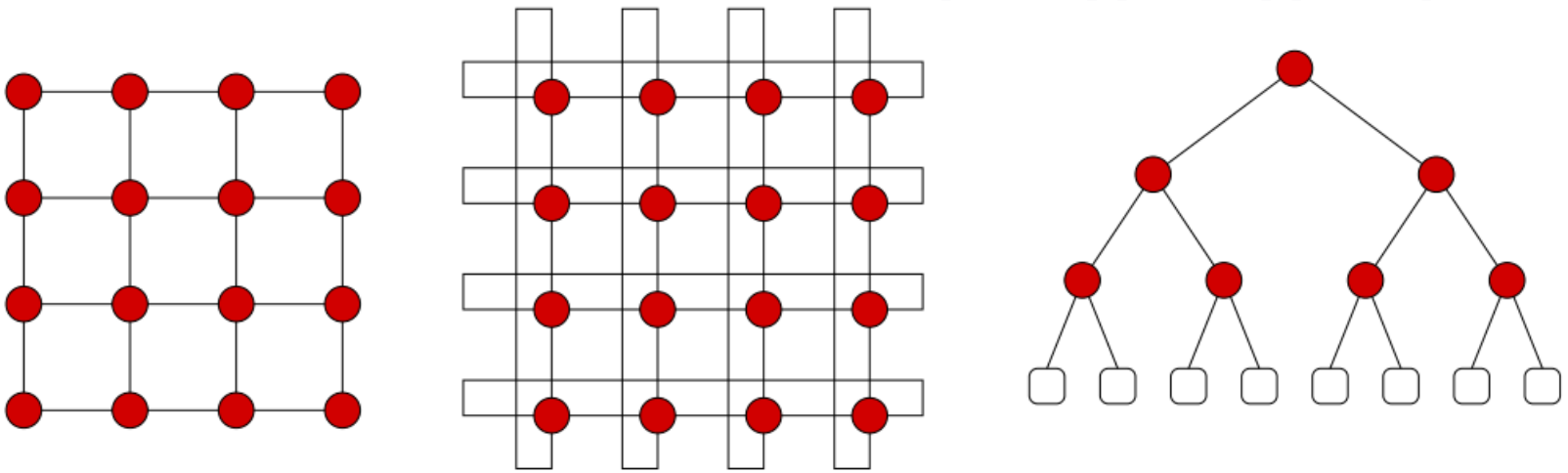


nach [Bje05]

- Router: implementieren Routing- und Switching-Verfahren
- Verarbeitungseinheiten (VA): IP-Cores (Prozessoren, Speicher)
- Links: verbinden Router
- Netzwerkadapter (NA):
  - Schnittstelle zwischen VAs und Netzwerk
  - Packen/Entpacken von Paketen

2 Grundlagen

Networks-on-a-Chip – Topologien [BDM06] [Sam10] [Wik05]



Gitter | Torus | Binärbaum

→ Gittertopologien am häufigsten genutzt

Sonstige Topologien: Ring, Spidergon, irreguläre, hybride

2 Grundlagen

Networks-on-a-Chip – Routing [BDM06] [Sam10] [Wik05]

- Lokalität: Source und Distributed Routing
- Dynamik: statisch und dynamisch
- Zentralisation: lokales und globales Wissen
- Minimalität: minimal und nichtminimal
- Beispiele
  - xy-Routing
  - Deflection-Routing

2 Grundlagen

Networks-on-a-Chip – Switching [BDM06]

- Leitungsvermittlung

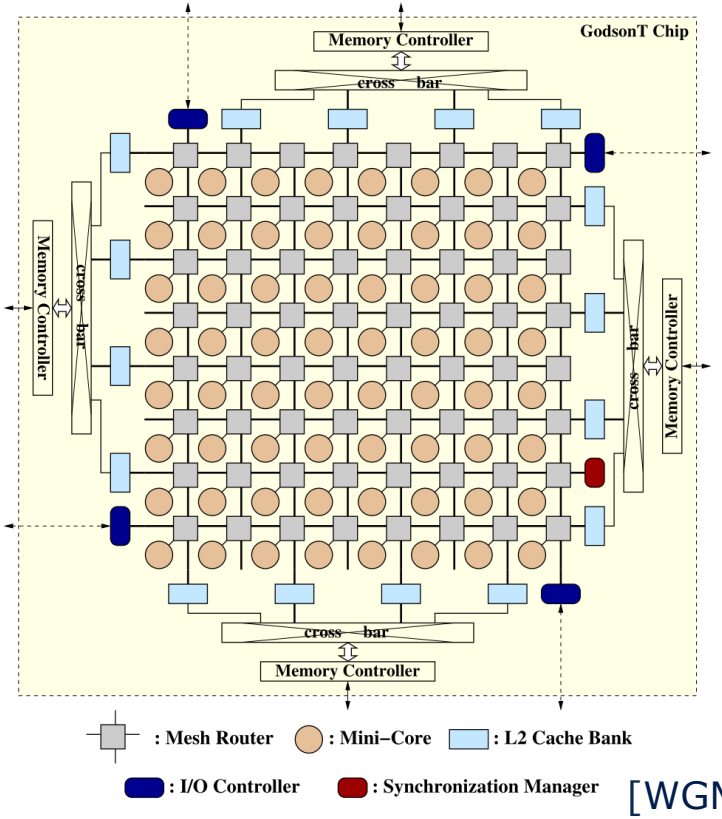
- Paketvermittlung
  - Store-and-Forward (SAF)
  - Virtual-Cut-Through (VCT)
  - Wormhole (WH)

2 Grundlagen

Beispiel: Godson-T-Many-Core-Architektur [FYZ09] [WGM08]

- 64 Kerne
- jeder Kern
  - 16 KB Befehls-Cache
  - 32 KB lokaler Speicher als Daten-Cache oder Scratchpad-Speicher
- L2-Cache an Speichercontrollern

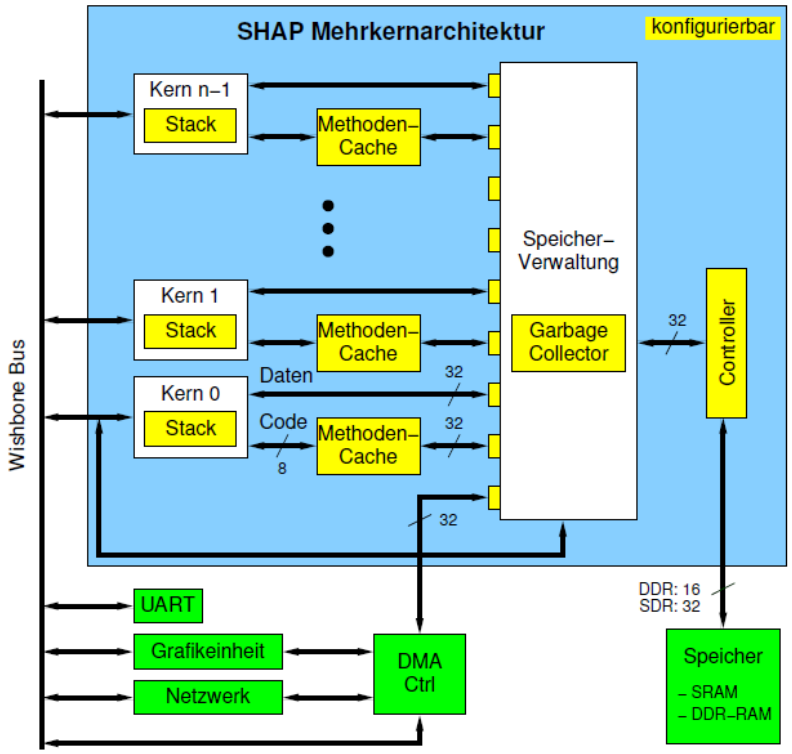
- 8x8 Gittertopologie
- xy-Routing
- Wormhole Switching



[WGM08]

2 Grundlagen

SHAP-Mehrkernarchitektur [Zab12]



[Olu12]

- direkte Ausführung von Java-Bytecode
- objektorientierte Adressierung
- Heap und Bytecode gemeinsam in einem zentralem Speicher
- automatische Speicherverwaltung mit nebenläufigem Garbage Collector
- Kerne über Ports an Speichermanager angeschlossen
- I/O-Anbindung über Wishbone-Bus und DMA

3 NoC-Ansätze für SHAP

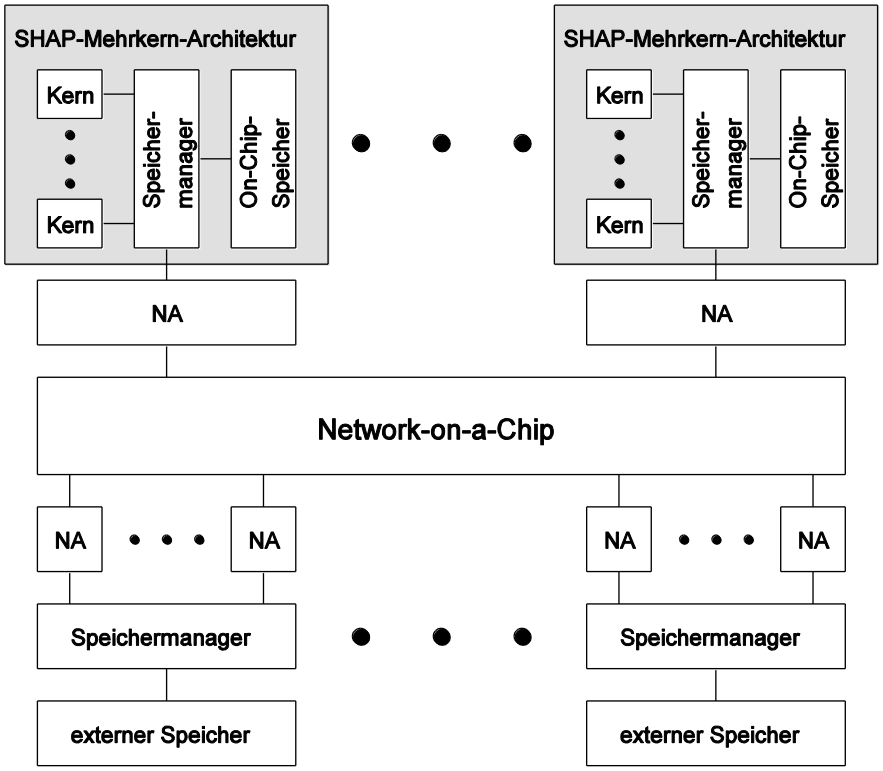
Kommunikationsmöglichkeiten der Kerne

- Speicherkopplung → Kerne nutzen gemeinsamen Adressraum
  - Variante 1: mehrere Speicher mit gemeinsamen Adressraum
  - Variante 2: auf Chip nur Caches

- Nachrichtenkopplung → expliziter Nachrichtenaustausch zwischen Kernen
  - Variante 1: enge Integration in Prozessor-Pipeline
  - Variante 2: über DMA analog Ethernet-Mac

3 NoC-Ansätze für SHAP

Speicherarchitektur



3 NoC-Ansätze für SHAP

Speicherverwaltung (1) - Verwaltungsstrukturen

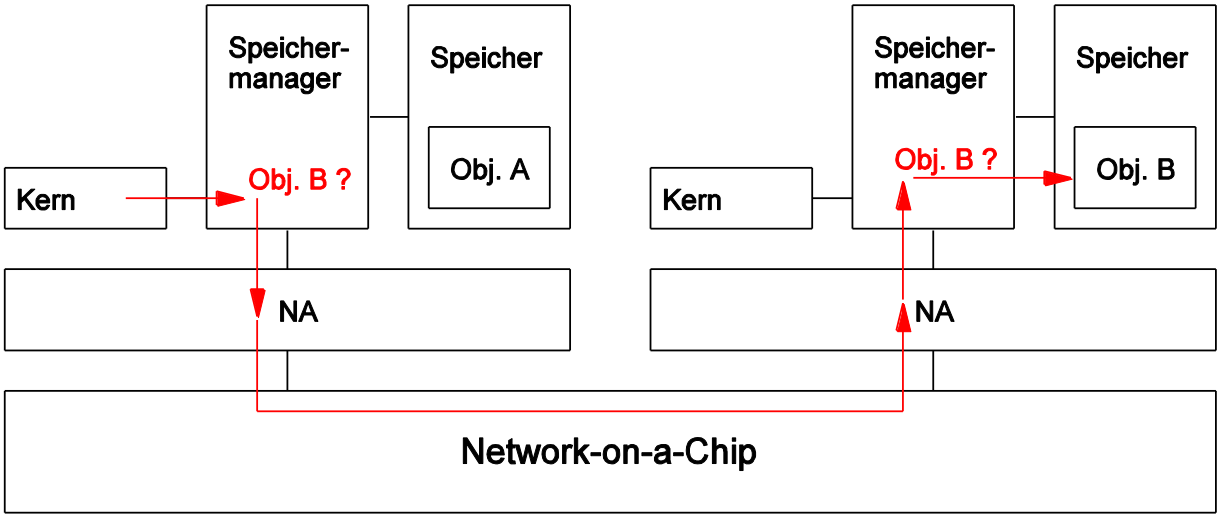
- lokale Objekte:
  - Referenztabelle wie bisher
  - zusätzliche Information: Anzahl eingehender entfernter Referenzen auf ein Objekt (für GC notwendig)

- entfernte Objekte:
  - Referenztabelle mit Indexzugriff nicht möglich
  - Tabelle: Zuordnung Objektreferenz zu entferntem Speicher

3 NoC-Ansätze für SHAP

Speicherverwaltung (2) - Speicherzugriffe

- auf lokale Objekte wie bisher
- auf entfernte Objekte:



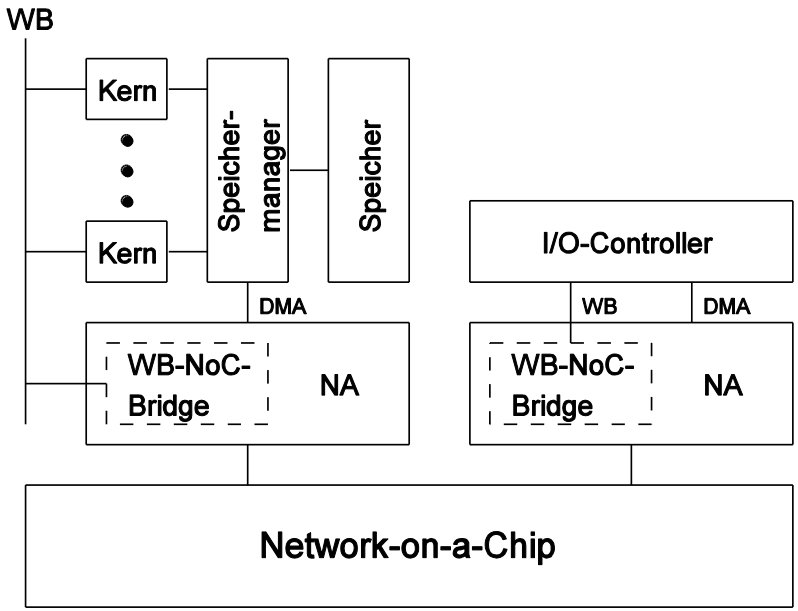
3 NoC-Ansätze für SHAP

Speicherverwaltung (3) – Garbage Collection

- mehrere Speicher → verteilter Garbage Collector notwendig
- neu: entfernte Referenzen auf lokale Objekte sind zu beachten
- Rückgriff auf GC für verteilte objektorientierte Systeme → [GF93]
  - zusätzliche GC-Phase markiert Objekte, auf die entfernte Referenzen existieren
  - dafür wird Anzahl der eingehenden Referenzen für jedes Objekt ausgewertet
  - nur entfernt erreichbare Objekte → Migration?
  - Aktualisierung der Anzahl eingehender entfernter Referenzen
- Kommunikation der GCs:
  - bestehende Daisy-Chain
  - NoC

3 NoC-Ansätze für SHAP

I/O-Anbindung



- Bus-Bridge als Schnittstelle zwischen Wishbone-Bus und NoC
- I/O-Geräte mit Wishbone-Schnittstelle können weiterhin angeschlossen werden
- Problem: Bus blockiert während I/O-Operation
- einfachste Lösung: Bus mit Out-of-Order-Completion

- DMA-Controller in den NAs
- mehrere Speicher → mehrere parallele DMA-Operationen möglich

3 NoC-Ansätze für SHAP

Netzwerkkonzepte

→ für erste Implementierung zunächst möglichst **einfache** Konzepte

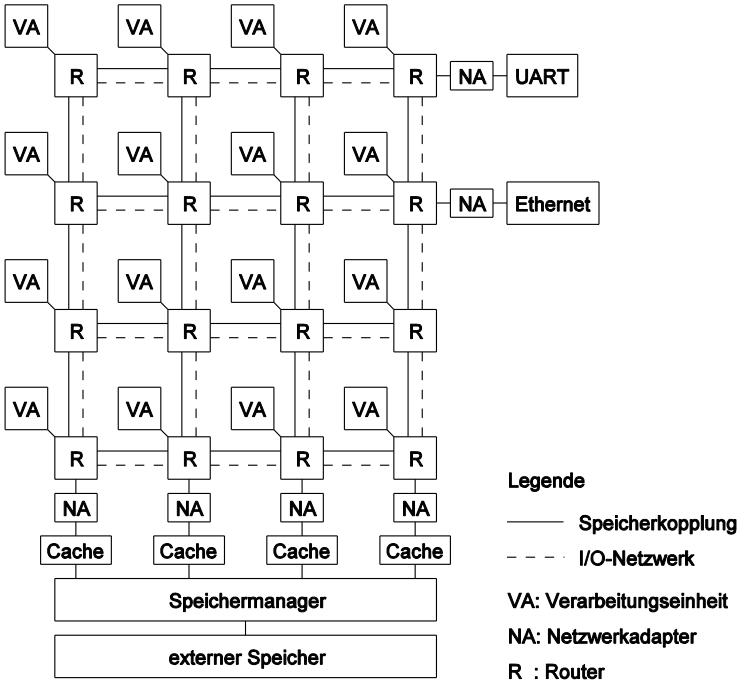
- Topologie: Gitter
- statisches xy-Routing
- Wormhole Switching

  → • Vermeidung von Deadlocks
  • wenig Pufferspeicher

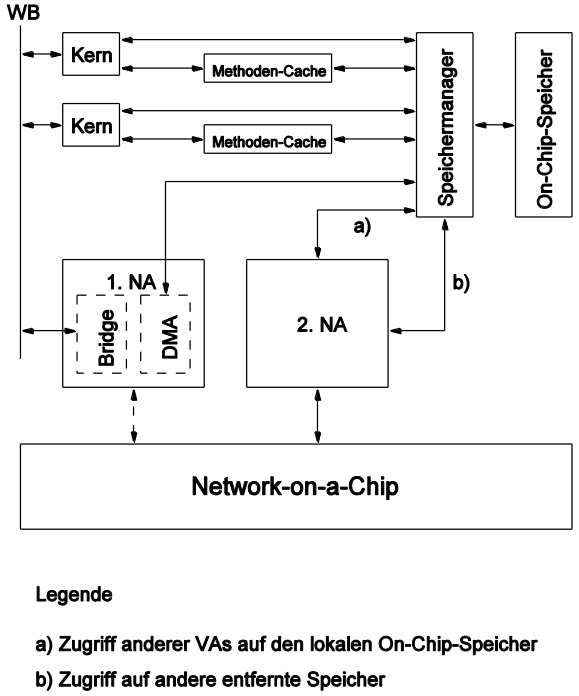
- Netzwerkadapter:
  - 1. NA: I/O: Bus-Bridge, DMA-Controller
  - 2. NA: Unterstützung für Speicherkopplung

3 NoC-Ansätze für SHAP

Beispielkonfiguration



Übersicht



VA-Details

4 Bewertung

Allgemein

- Ziel: möglichst wenige Änderungen am bestehenden System
  → wird erreicht
  → aber dadurch Kompromisse:
  - keine Migration von Objekten in andere Speicher
  - I/O-Bus lange blockiert
  - keine Threadmigration

Vergleich zu anderen Many-Core-Architekturen

- keine andere Java-Many-Core-Architektur bekannt
- Konzept des On-Chip-Speichers vs. Caches (vgl. z.B. IBM-Cell)

5 Zusammenfassung und Ausblick

Ergebnisse

- NoC-Ansätze für SHAP-Many-Core-Architektur gezeigt
- neue Speicherarchitektur
- neue Verwaltungsstrukturen
  → Ablegen von Objekten in verschiedenen Speichern
  → verteilte Garbage Collection

Ausblick

- Ansätze in Implementierung überprüfen
- Kerne über mehrere Boards verteilen
- Nachrichtenkopplung genauer betrachten

Literatur

[BDM06] Benini, Luca; De Micheli, Giovanni: Networks on Chip. Morgan Kaufmann, 2006. – ISBN 978–0–12–370521–1

[Bje05] Bjerregaard, Tobias: The MANGO Clockless Network-on-Chip: Concepts and implementation, Informatics and Mathematical Modelling, Technical University of Denmark, Diss., 2005

[FYZ09] Fan, Dong-Rui; Yuan, Nan; Zhang, Jun-Chao; Zhou, Yong-Bin; Lin, Wei; Song, Feng-Long; Ye, Xiao-Chun; Huang, He; Yu, Lei; Long, Guo-Ping; Zhang, Hao; Liu, Lei: Godson-T: An Efficient Many-Core Architecture for Parallel Program Executions. In: Journal of Computer Science and Technology 24 (2009), S. 1061–1073. – ISSN 1000–9000

Literatur

[GF93] Gupta, Aloke; Fuchs, W. K.: Garbage Collection in a Distributed Object-Oriented System. In: IEEE Transactions on Knowledge and Data Engineering 5 (1993), April, Nr. 2, S. 257–265. – ISSN 1041–4347

[Olu12] Olunczek, Andrej: Leistungssteigerung der Speicherarchitektur des SHAP-Mehrkernprozessors, Fakultät Informatik, Technische Universität Dresden, Diplomarbeit, 2012

[Sam10] Samman, Faizal Arya S.: Microarchitecture and Implementation of Networks-on-Chip with a Flexible Concept for Communication Media Sharing, Fachbereich Elektrotechnik und Informationstechnik, Technische Universität Darmstadt, Diss., 2010

Literatur

[WGM08] Wang, Xu; Gan, Ge; Manzano, J.; Fan, Dongrui; Guo, Shuxu: A Quantitative Study of the On-Chip Network and Memory Hierarchy Design for Many-Core Processor. In: 14th IEEE International Conference on Parallel and Distributed Systems, 2008 (ICPADS ’08). – ISSN 1521–9097, S. 689–696

[Wik05] Wiklund, Daniel: Development and Performance Evaluation of Networks on Chip, Department of Electrical Engineering, Linköping University, Diss., 2005

[Zab12] Zabel, Martin: Effiziente Mehrkernarchitektur für eingebettete Java-Bytecode-Prozessoren, Fakultät Informatik, Technische Universität Dresden, Diss., 2012