TECHNISCHE UNIVERSITÄT DRESDEN

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

Java-Bytecode-Prozessor SHAP

Output'08

Martin Zabel (*martin.zabel@tu-dresden.de*)
Thomas B. Preußer, Peter Reichel
Professur für VLSI-Entwurfssysteme, Diagnostik und Architektur
Institut für Technische Informatik
http://vlsi-eda.inf.tu-dresden.de

Dresden, 18.04.2008

Motivation

Objekt-orientierte Programmierung ist vorteilhaft für die schnelle und einfache Entwicklung komplexer Applikationen. Java ist sehr populär:

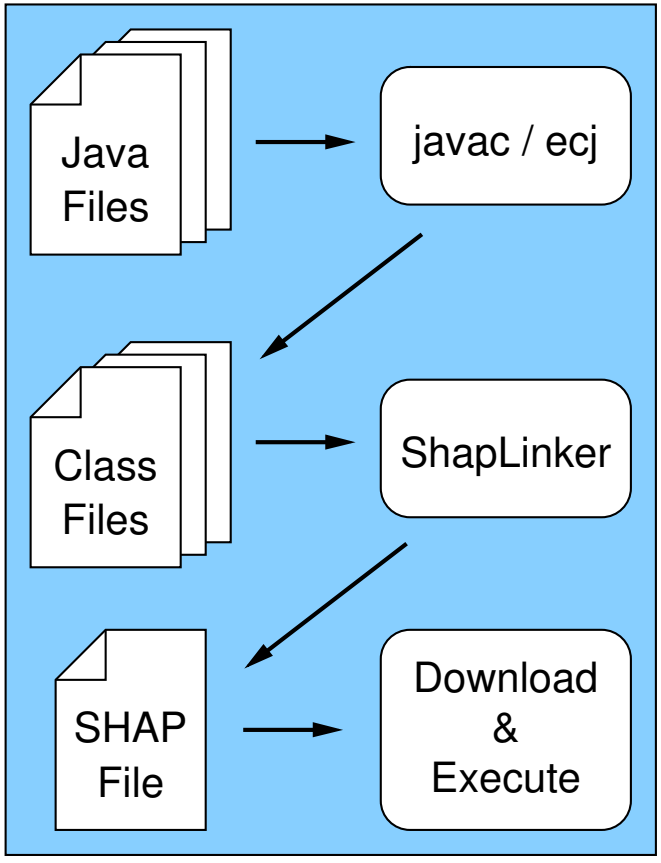
- Definition der JVM addressiert Portabilität und Sicherheit.
- Implementierungen der JVM für viele Plattformen verfügbar.
- Kompakter Java-Bytecode bedingt geringen Speicherbedarf und kurze Download-Zeiten.

Features von SHAP

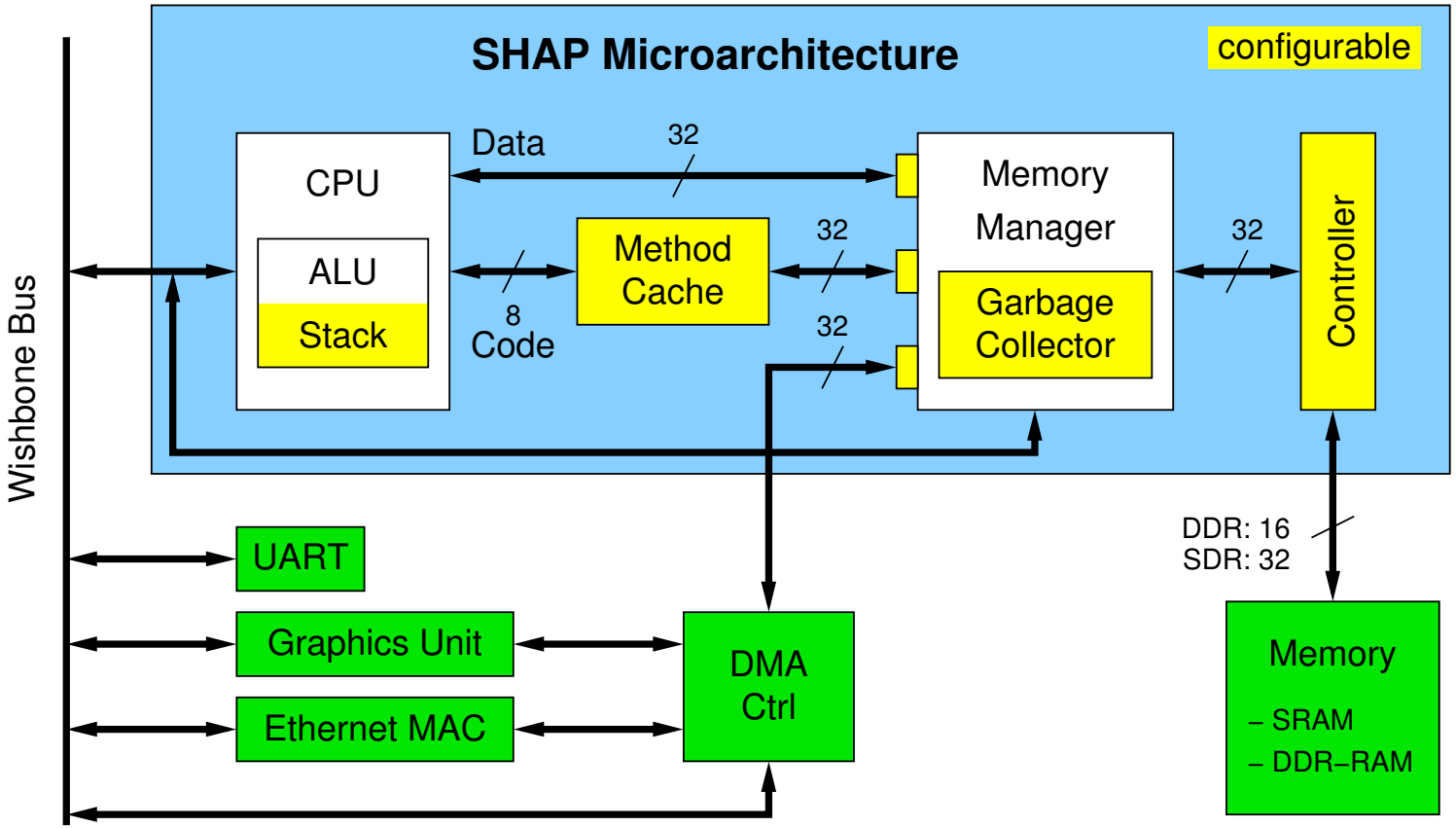
- Eingebette Java-Mikroarchitektur für sicherheitsrelevante, echtzeitfähige und mehrfädige Applikationen ohne untergeordnetes Betriebssystem.
- **Neuartige Konzepte für Echtzeit-Unterstützung:**
  - integriertes Stack- und Thread-Management,
  - nebenläufiger, CPU-unabhängiger Garbage Collector,
  - preemptives Round-Robin-Scheduling.
- **Vielfältige Unterstützung von objekt-orientierten Konzepten:**
  - automatisches Heap-Management, strukturierte Ausnahmebehandlung, Mehrfachvererbung durch Interfaces,
- Anwendungsbereiche: Automatisierungstechnik, Sensorik, Netzwerktechnik, ...
- API: „Connected Limited Device Configuration“

SHAP-Linker

- Statische Auflösung von Verweisen.
- Ersetzung von Native-Methoden durch Spezial-Bytecodes.
- Einfügen von Bytecodes für Interface-Handling. [1]
- Bytecode-Optimierung.
- Constant-Pool Elimination.
- Aggressive Inlining.
- Dynamisches Nachladen von Applikationen prinzipiell möglich.

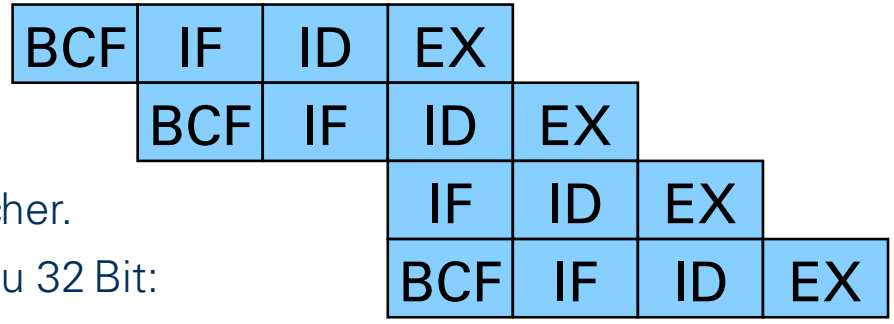


SHAP-Mikroarchitektur [2]



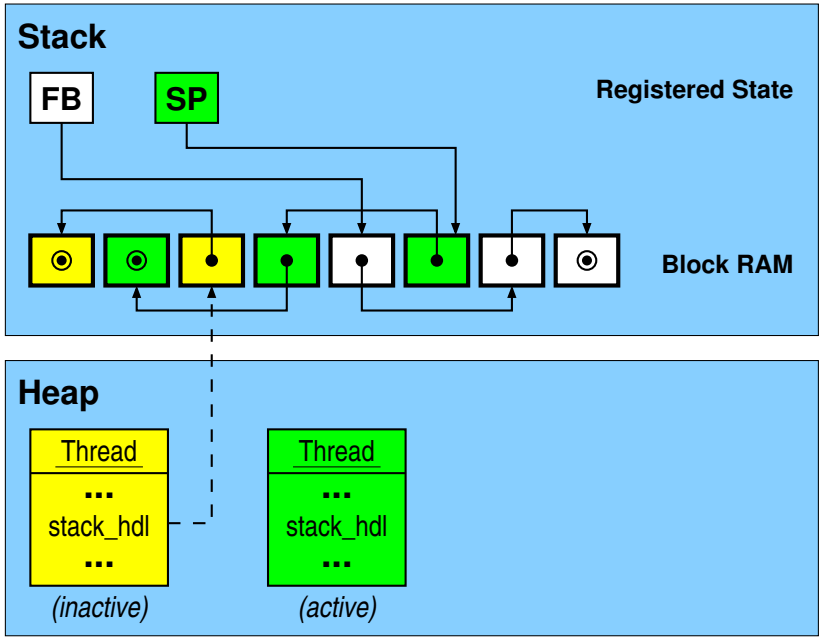
CPU [3]

- Abarbeitung der Java-Bytecodes durch Mikroprogramme.
- 4-stufige Pipeline:
- 57 Mikrocode-Befehle mit 9-Bit-Befehlsformat
- Stack-orientiert, kein Registerspeicher.
- Mikrokode-Datenspeicher, Werte zu 32 Bit:
  - 16 Variablen,
  - 48 Konstanten (an Stelle von Direktwerten),
  - 64 „Sprünge“.
- Preemptives Round-Robin-Scheduling.
- Wishbone Master.



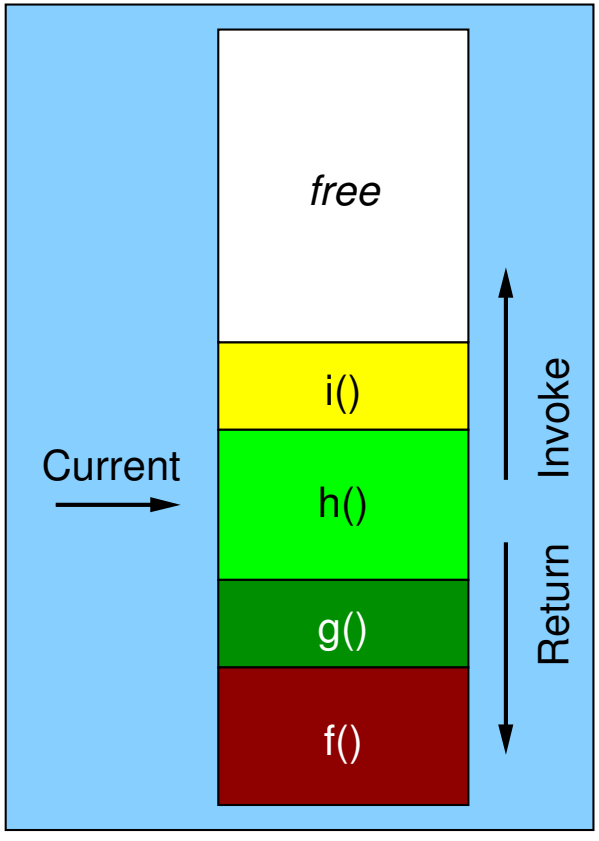
Multi-Threaded Stack [4]

- On-Chip-Speicher.
- Unterteilung des Speichers in Blöcke gleicher Größe.
- Verkettung dieser Blöcke zu einem Stack bzw. zur Freiliste.
- Eigenständiger Kontextwechsel.
- Management der Methoden-Frames.
- *TOS* und *NOS* in Register.
- *Beispiel:* 8 KByte Stack, bis zu 32 Threads



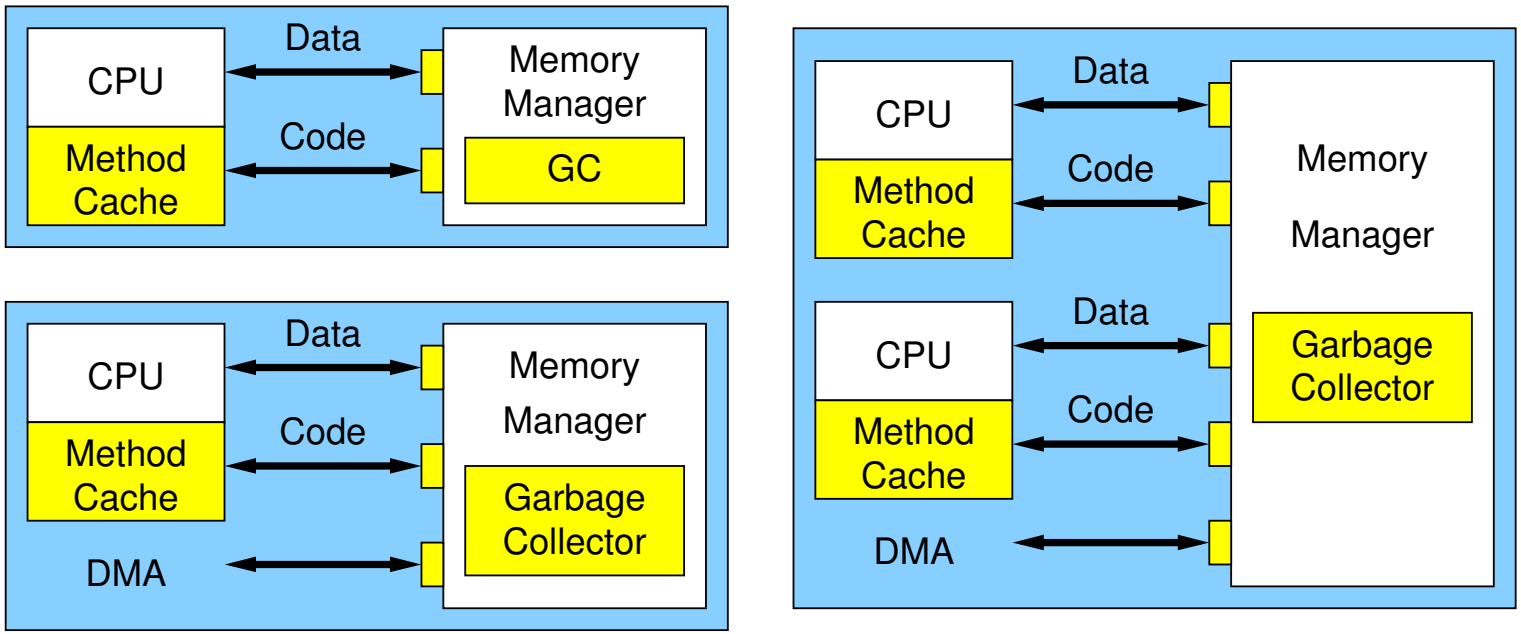
Method Cache [5]

- Zur Zeit: Caching der kompletten Methode vor deren Ausführung.
- In Zukunft: Smart Prefetch Caching.
- Stack-basierte Organisation:
  - 3-fach assoziativ: Return, Recursive Invoke, Re-Invoke.
  - Verwaltungsdaten ebenfalls im Cache-Speicher abgelegt.
  - 90% Trefferrate typisch.
- Ring-Puffer.
- Optional: Parallele „Stacks“ durch mehrere Slices.
- *Beispiel:* 2 KByte



Multi-Port Memory Manager [6]

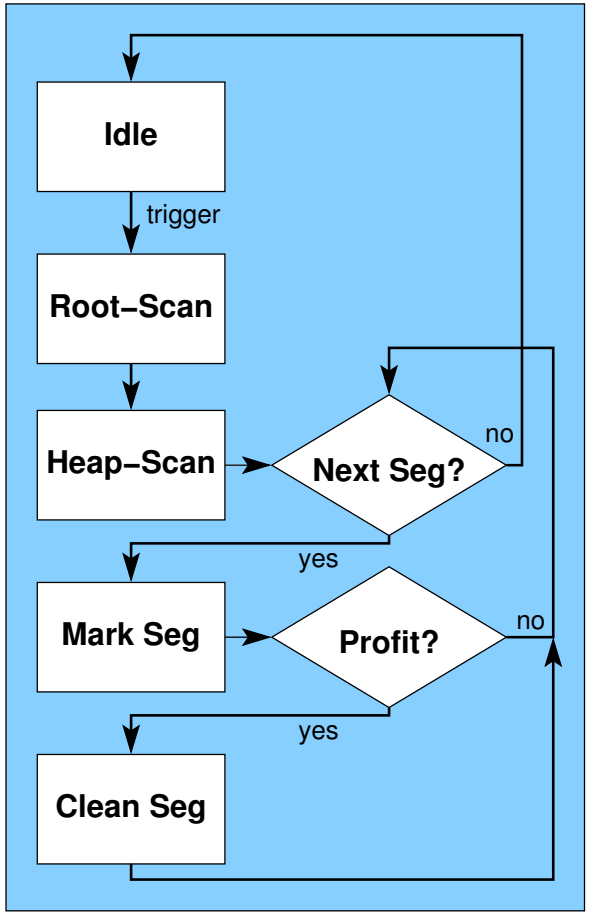
- Kapselt die Heap-Verwaltung.
- Mehrere Ports für verschiedene Systemkonfigurationen.
- Pipelined Reads für hohen Durchsatz.



Garbage Collector [7]

- Nebenläufig und CPU-unabhängig.
- Stack-Monitoring.
- Segmentierung des Heaps:
  - Schnelle Allokation in konstanter, vorhersagbarer Zeit.
  - Segmentfreigabe durch Objekt-verschiebung; sofern gewinnbringend.
- Scannen / Verschieben von Objekten ist unterbrechbar.

  ⇒ kurze Latenz für Zugriffe durch CPU/Method-Cache/DMA,
  ⇒ automatische Umlenkung von Zugriffen.
- GC-Policy zur Laufzeit konfigurierbar.



Aktuelle Projekte

Im Bereich Hardware:

- Erweiterung des GC auf schwache Referenzen,
- Multi-CPU-SHAP,
- IPv4-Preprozessor,
- USB-Controller,
- DDR-SDRAM-Controller mit ECC.

Im Bereich Software:

- Spiel „SkyRoads“ mit 3D-Engine.
- SLIP-Protokoll.

Interesse?

`http://shap.inf.tu-dresden.de`

Themen sowohl im Bereich **Hardware**...

- Hardware-unterstützer Scheduler,
- Fortgeschrittene Methoden-Cache-Organisation,
- Objekt-Cache,
- Controller für I/O und Speicher.

...als auch im Bereich **Software**:

- Diverse Applikationen.
- Portierung einer .NET-Laufzeitumgebung,
- Serialisierung von Objekten,
- Dynamisches Nachladen von Applikationen,
- TCP/IP-Stack für geg. Ethernet-Controller.

Beleg Diplom Komplexpraktikum

Literatur

[1] PREUSSER, T. B.; ZABEL, M.; SPALLEK, R. G.:
Enabling Constant-Time Interface Method Dispatch in Embedded Java Processors.
In: *The 5th International Workshop on Java Technologies for Real-time and Embedded Systems - JTRES 2007*, 2007

[2] ZABEL, M.; PREUSSER, T. B.; REICHEL, P.; SPALLEK, R. G.:
Secure, Real-Time and Multi-Threaded General-Purpose Embedded Java Microarchitecture.
In: *Proceedings of the 10th Euromicro Conference on Digital System Design Architectures, Methods and Tools (DSD 2007)*, IEEE Press, August/2007

[3] ZABEL, M.; PREUSSER, T. B.; REICHEL, P.; SPALLEK, R. G.:
SHAP — Secure Hardware Agent Platform.
In: *Dresdner Arbeitstagung Schaltungs- und Systementwurf DASS'2007*.
Bergstr. 70, D-01069 Dresden: TUDpress, Verlag der Wissenschaften GmbH, Mai/2007. – ISBN 978–3–940046–28–4

[4] PREUSSER, T. B.; ZABEL, M.; REICHEL, P.:
The SHAP Microarchitecture and Java Virtual Machine.
Fakultät Informatik, Technische Universität Dresden, Forschungsbericht TUD-FI07-02, 2007. –
ISSN 1430–211X

[5] PREUSSER, T. B.; ZABEL, M.; SPALLEK, R. G.:
Bump-Pointer Method Caching for Embedded Java Processors.
In: *The 5th International Workshop on Java Technologies for Real-time and Embedded Systems - JTRES 2007*, 2007

[6] ZABEL, M.; REICHEL, P.; SPALLEK, R. G.:
Multi-Port-Speichermanager für die Java-Plattform SHAP.
In: *Dresdner Arbeitstagung Schaltungs- und Systementwurf DASS'2008*, To be published.

[7] REICHEL, P.:
Entwurf und Implementierung verschiedener Garbage-Collector-Strategien für die Java-Plattform SHAP, Technische Universität Dresden, Großer Beleg, 2007