Codesigned Virtual Machines

Seminar Virtualisierung

Philipp Kirchhofer
philipp.kirchhofer@student.kit.edu

Institut für Technische Informatik
Lehrstuhl für Rechnerarchitektur
Universität Karlsruhe (TH)

13. Juli 2009

Gliederung

Codesign & Virtuelle Maschine

Einführung

Hardware/Software Codesign:

Gleichzeitiger Entwurf von Hardware und Software

- Hohe Flexibilität
- Hoher personeller und planerischer Aufwand

In Verbindung mit einer Virtuellen Maschine:

- Unterstützung alter Befehlssatzarchitekturen (Investitionsschutz)
- Erhöhung der Freiheitsgrade beim Design

Verknüpfung

Codesigned Virtual Machine $\rightarrow$ Codesign + Virtuelle Maschine

Gründe für Codesigned Virtual Machines

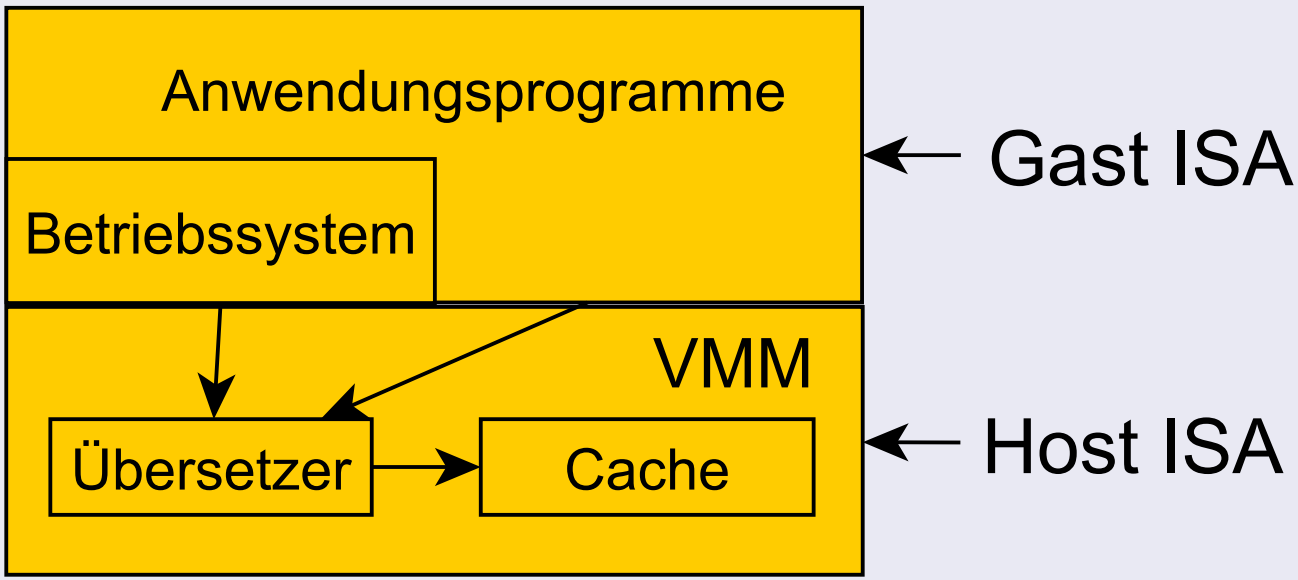
- Hohe Flexibilität
- Verbesserte Performance
- Einfaches Hardware-Design
- Energieeffizienz

Gliederung

Virtual Machine Monitor

Zentrale Funktionen der Codesigned Virtual Machine

- Umwandlung von Instruktionen (Gast ISA $\rightarrow$ Host ISA)
- Behandlung von Ausnahmen und Unterbrechungen



Virtual Machine Monitor

- in der Host ISA codiert
- läuft direkt auf der Hardware

Register Abbildung

Abbildung Register Gast ISA $\rightarrow$ Register Host ISA

Mehr Host ISA Register vorhanden als für die Ausführung des Gast ISA Codes benötigt

Weitere Verwendung:

- Optimierung der Ausführung des Gast Codes
- Schattenregister
- VM Verwaltung

Virtual Machine Monitor

Speicher Abbildung

Aufteilung des realen Systemspeichers in disjunkte Teile:

- Reservierter Speicher:
  - Code und Daten der VMM Software
  - Übersetzungscache
  - Verwaltungsinformationen
- Konventioneller Speicher:
  - Betriebssystem
  - Anwendungen

Übersetzungscache

Programme oft zeitlich und örtlich lokal:
$\rightarrow$ Caching übersetzter Programmteile

Checkpointing

Situation

- Bei Codesigned Virtual Machines häufig blockweise Programmabarbeitung
- Realisierung von Ausnahmesemantiken

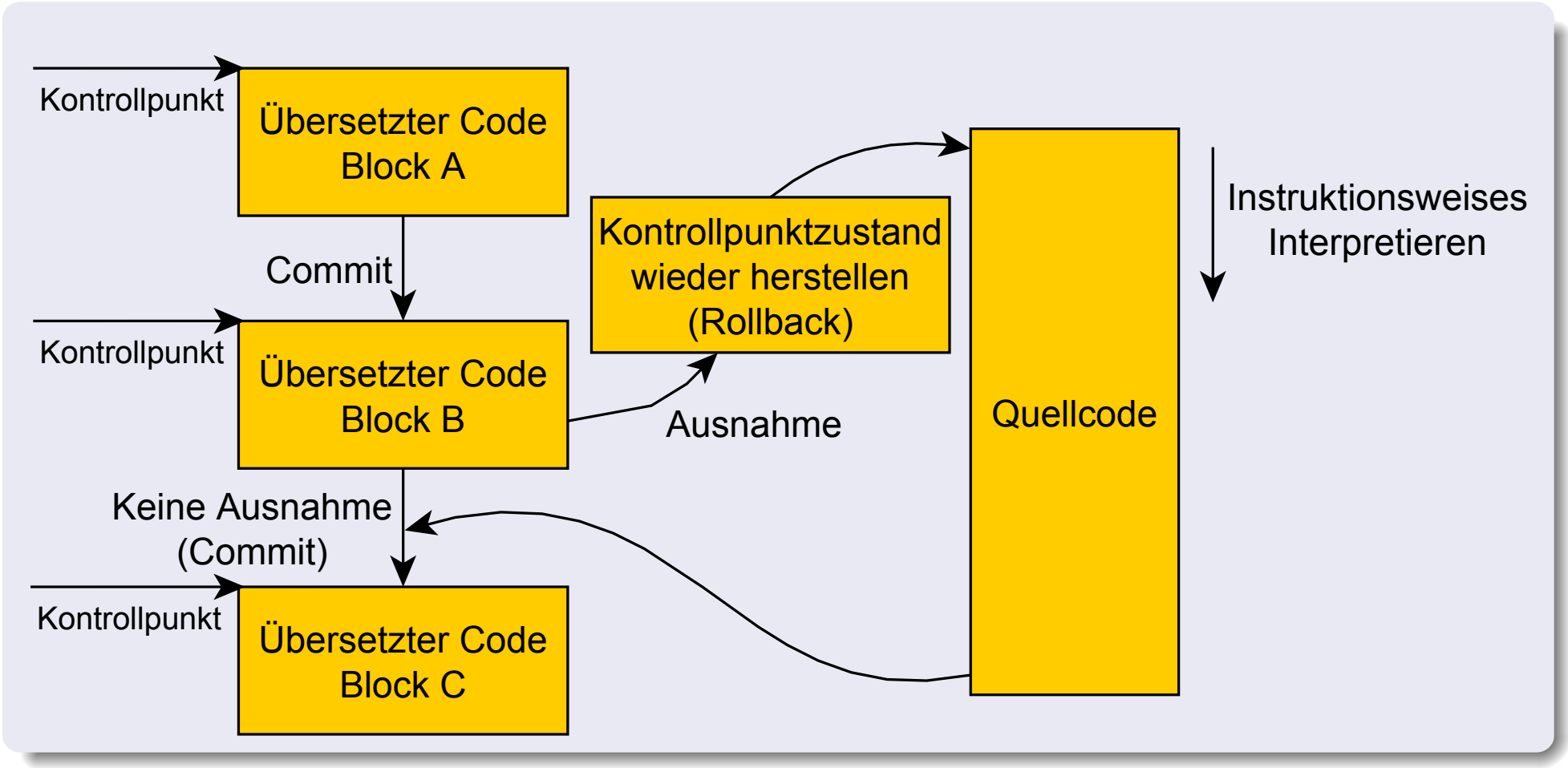
Problem

Fehlerbehandlung kann erst nach vollständiger Ausführung eines Blockes durchgeführt werden

Lösung

Checkpointing durch Einsatz von Schattenregistern

Checkpointing



Gliederung

Transmeta Entstehungsgeschichte

Transmeta

- 1995 in Santa Clara, Kalifornien gegründet
- CEO David Ditzel (Co-Autor RISC)
- VLIW Ansatz
- Entwickelte Prozessorreihen Crusoe und Efficeon

Zielsetzung beim Design der Prozessoren

- Hohe Flexibilität
- Verbesserte Performance
- $\rightarrow$ Einfaches Hardware-Design
- $\rightarrow$ Energieeffizienz

Code Morphing Software

Code Morphing Software

- Transmetas Bezeichnung für Virtual Machine Monitor
- Umsetzung IA-32 $\rightarrow$ VLIW
- Gast ISA prinzipiell beliebig
  - Entscheidung für IA-32 aufgrund weiter Verbreitung
  - Prototyp PowerPC
  - Änderung durch Austausch von Code Morphing Software möglich

Transmeta Interna

Zerlegung

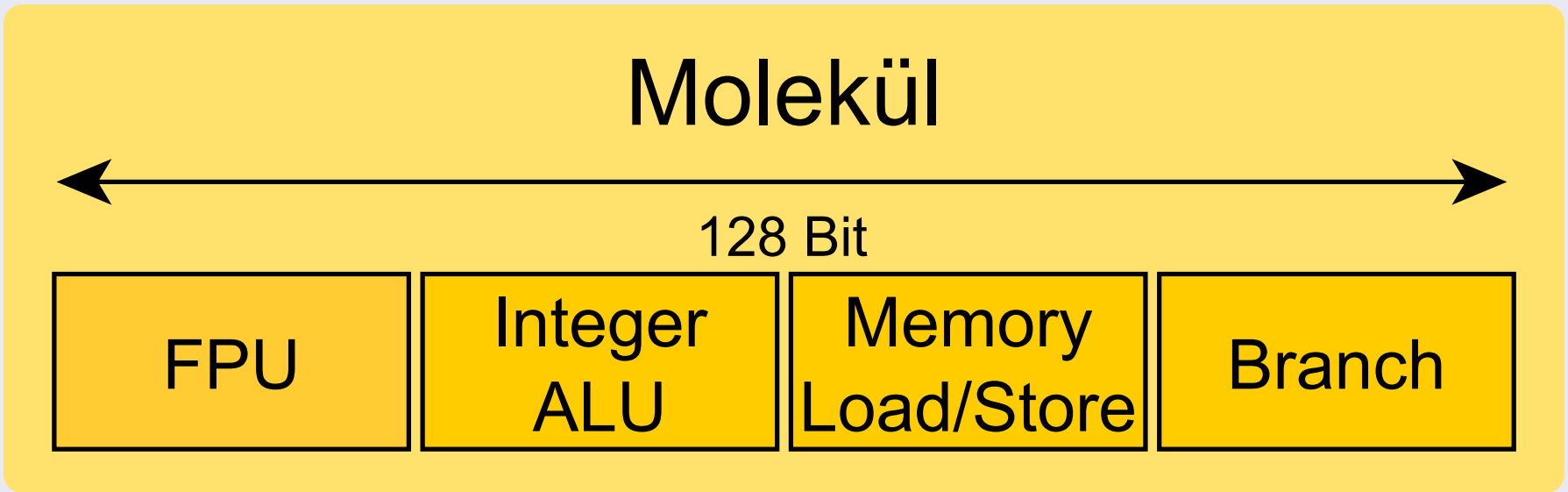
- Zerlegung IA-32 Instruktion $\rightarrow$ Atome
- Atome $\rightarrow$ 128 Bit VLIW Moleküle

Verarbeitung

- Molekül kann Atome aus verschiedenen IA-32 Instruktionen enthalten
- Parallele Ausführung aller Atome eines Moleküls
- Feste Zuordnung Atomposition $\rightarrow$ Funktionseinheit

Transmeta Interna

Aufbau eines Transmeta Crusoe Moleküls



Transmeta Interna

Vorteile

- Einfache Dekodierungs- und Verteilungslogik
- Keine komplexe Out-of-Order Hardware

Performance-Problem

Atomplätze nicht vollständig belegt

$\Rightarrow$ NOPs einfügen

$\Rightarrow$ Erhebliche Reduktion des Durchsatzes

MegaProto/E Cluster

Einsatz von Codesigned Virtual Machines

Neue Möglichkeiten für den Bau von energieeffizienten Prozessoren

Idee

Hohe Performance durch Nutzung von Niedrigenergieprozessoren bei hoher Packungsdichte

Einordnung

- Hohe Flexibilität
- $\rightarrow$ Verbesserte Performance
- Einfaches Hardware-Design
- $\rightarrow$ Energieeffizienz

MegaProto/E Cluster

Cluster Einheit

Interna

- 16 Transmeta Efficeon
- 1 Transmeta Crusoe
- Temperatur $< 40^{\circ}$ C
- 1,3 bis 3,7 fach schneller als Dual-Xeon bei vergleichbarer Leistungsaufnahme

Peak Performance / Power

Crusoe TM5800	124 MFlops / W
Efficeon TM8820	666.7 MFlops / W
Intel Dual-Xeon	71,76 MFlops / W

IBM System i

Einsatz von Codesigned Virtual Machines

Hoch abstrahierte Gast ISA

IBM System i

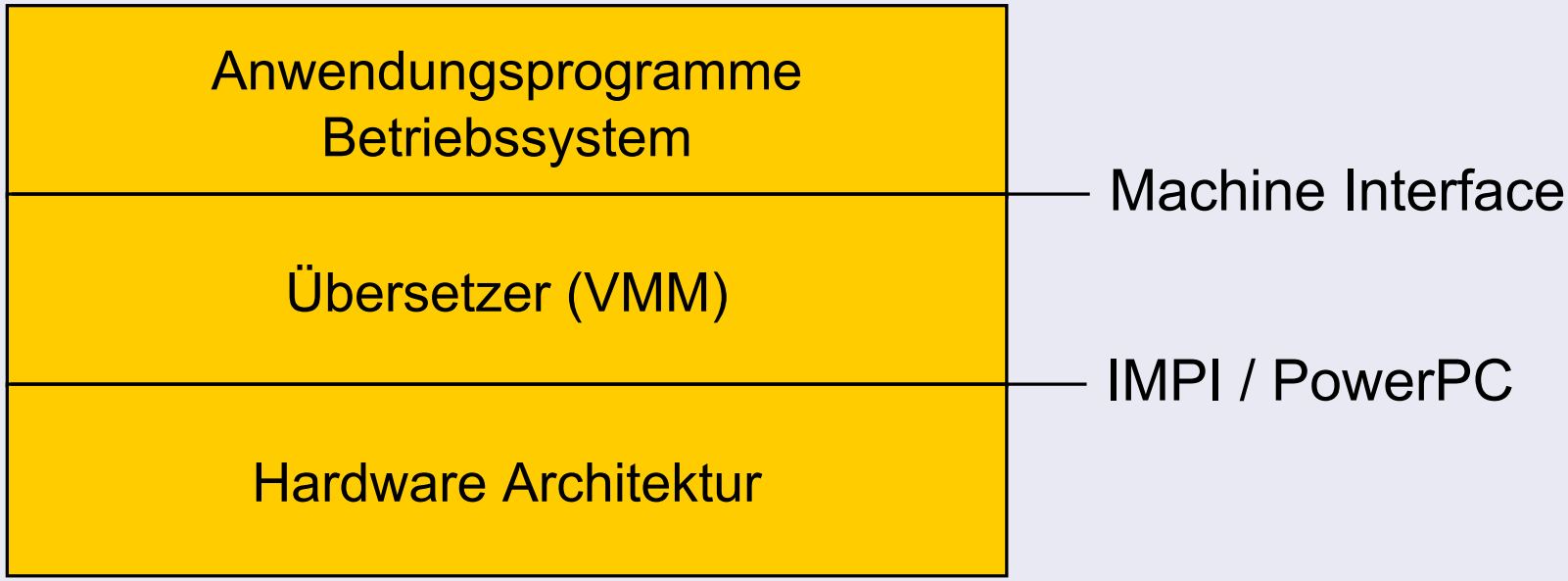
- 1986 entwickelt
- Midrange-Rechner
- Architekturunabhängigkeit
- Einfaches Software Design

Einordnung

- $\rightarrow$ Hohe Flexibilität
- Verbesserte Performance
- Einfaches Hardware-Design
- Energieeffizienz

IBM System i

IBM System i Architektur



Machine Interface

- Umfangreicher objektorientierter Befehlssatz
- Zukunftssicherheit (128 Bit Adressraum)
- Einstufiges Speichersystem

Ausblick

Ausblick

IBM System i	Komplette Architekturwechsel durchgeführt
Transmeta Crusoe / Efficeon	Kommerziell nicht erfolgreich
	Technischer Pionier

Weitere Produkte im Consumer Bereich nicht absehbar.

Vielen Dank für die Aufmerksamkeit!

Fragen?

Code Morphing Software