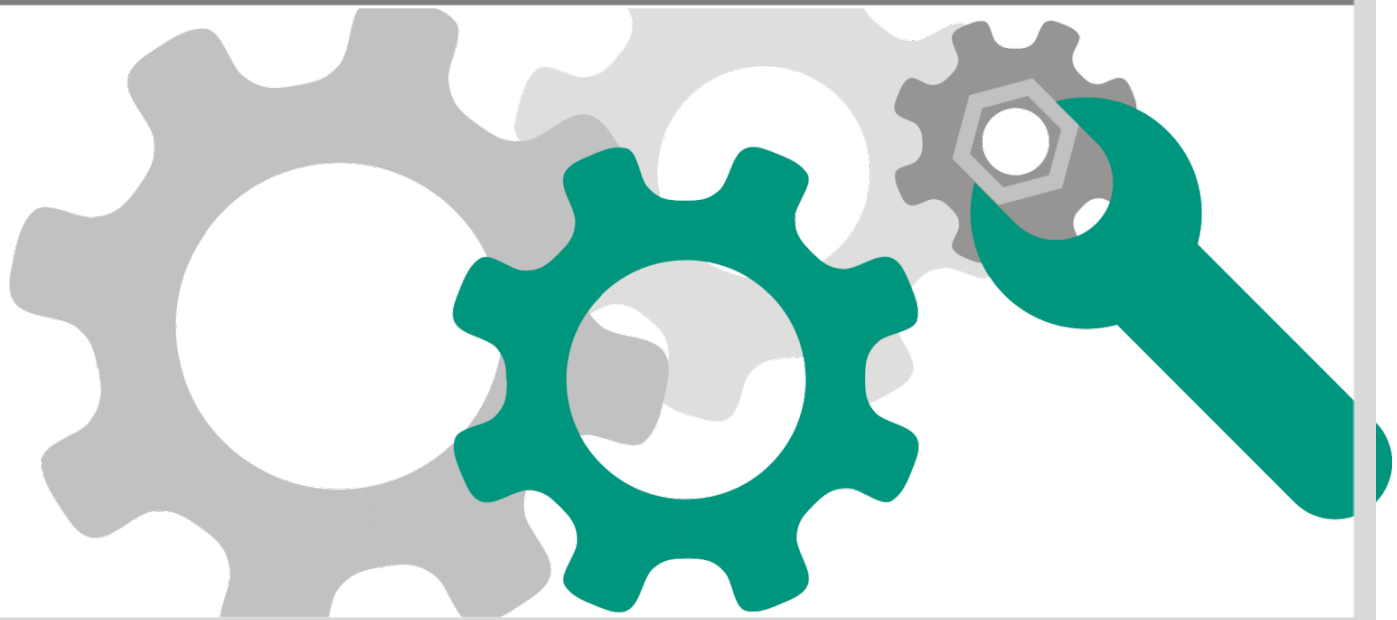
Bereit für Multicore?

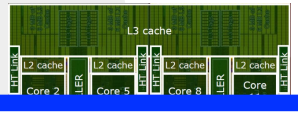
Prof. Dr. Walter F. Tichy

IPD Tichy – Lehrstuhl für Programmiersysteme



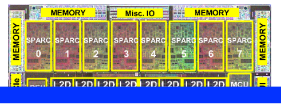
AMD Opteron 12 cores

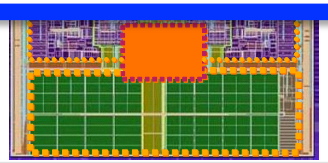
~1.8 Bill. T. on 2x3.46cm^2

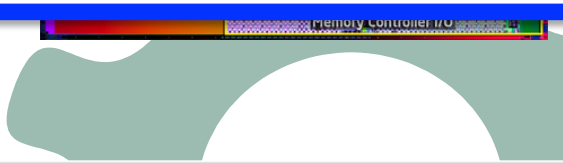


Sun Niagara3 16 cores

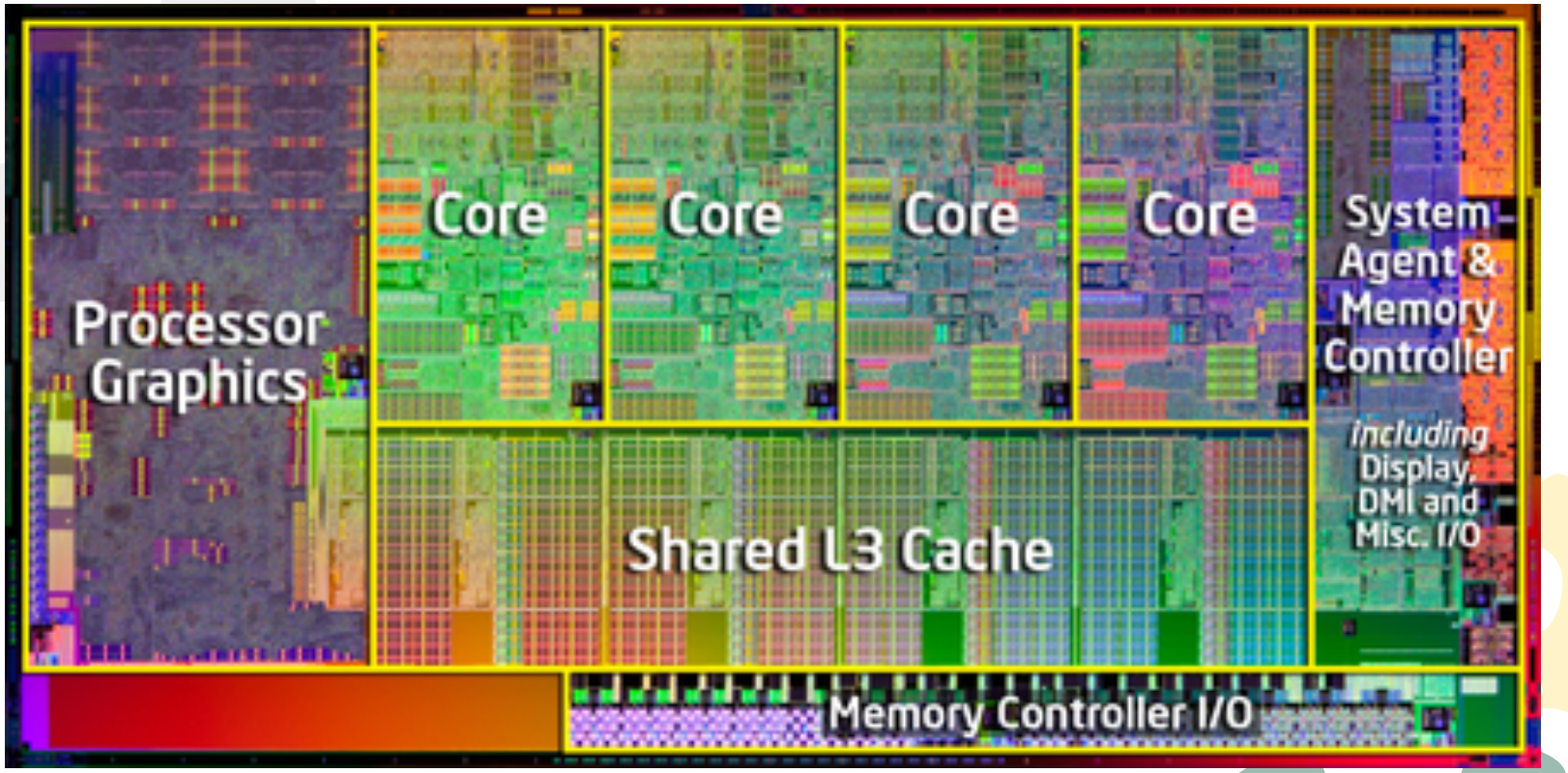
~1 Bill. T. on 3.7cm^2







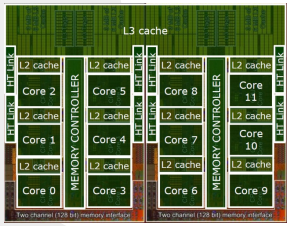
2011 Intel Sandy Bridge



2011: 4 CPUs, 6 graphics Execution Units
Später: 8 CPUs, 12 graphics Execution Units

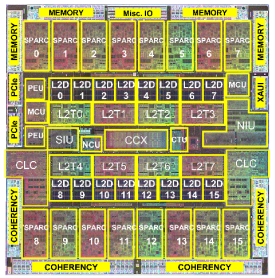
AMD Opteron 12 cores

~1.8 Bill. T. on 2x3.46cm²



Sun Niagara3 16 cores

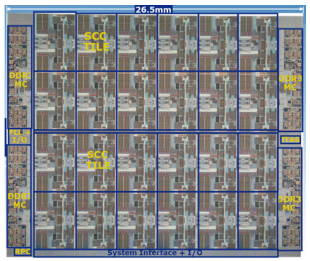
~1 Bill. T. on 3.7cm²



Karlsruhe Institute of Technology

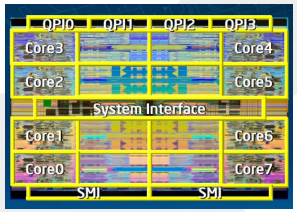
Intel SCC 48 cores

~1.3 Bill. T. on 5.6 cm²



Intel 8 cores

~2.3 Bill. T. on 6.8cm²

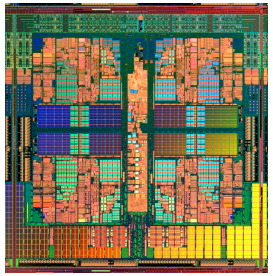


Was fehlt hier?

Software!

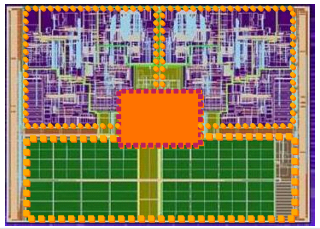
Intel 4 cores

~582 Mio. T on 2.86cm²



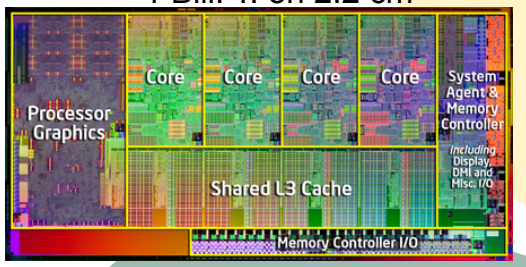
Intel 2 cores

~167 Mio. T. on 1.1cm²

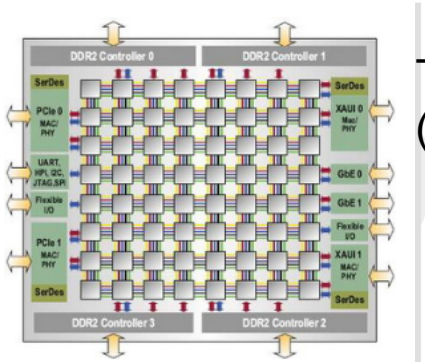


Intel Sandy Bridge 4+6 cores

~1 Bill. T. on 2.2 cm²

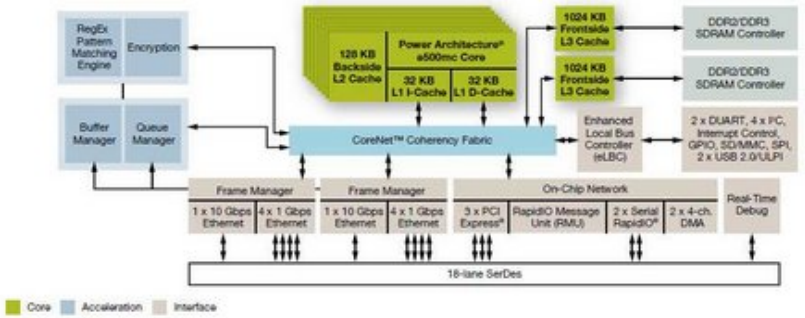


Eingebettete Mehrkerner

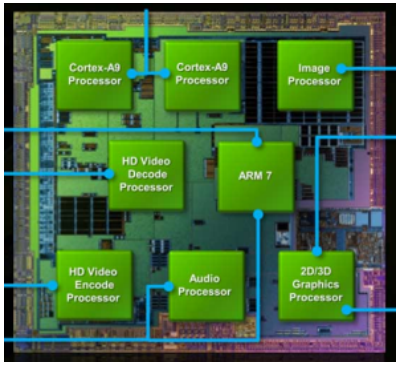


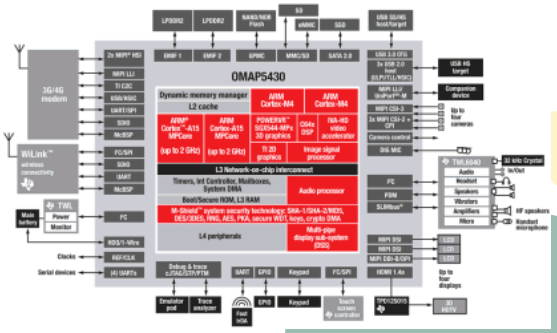
Tilera Tile Pro64
(64 Kerne)

Freescale QorIQ P4080
(8 Power Kerne)



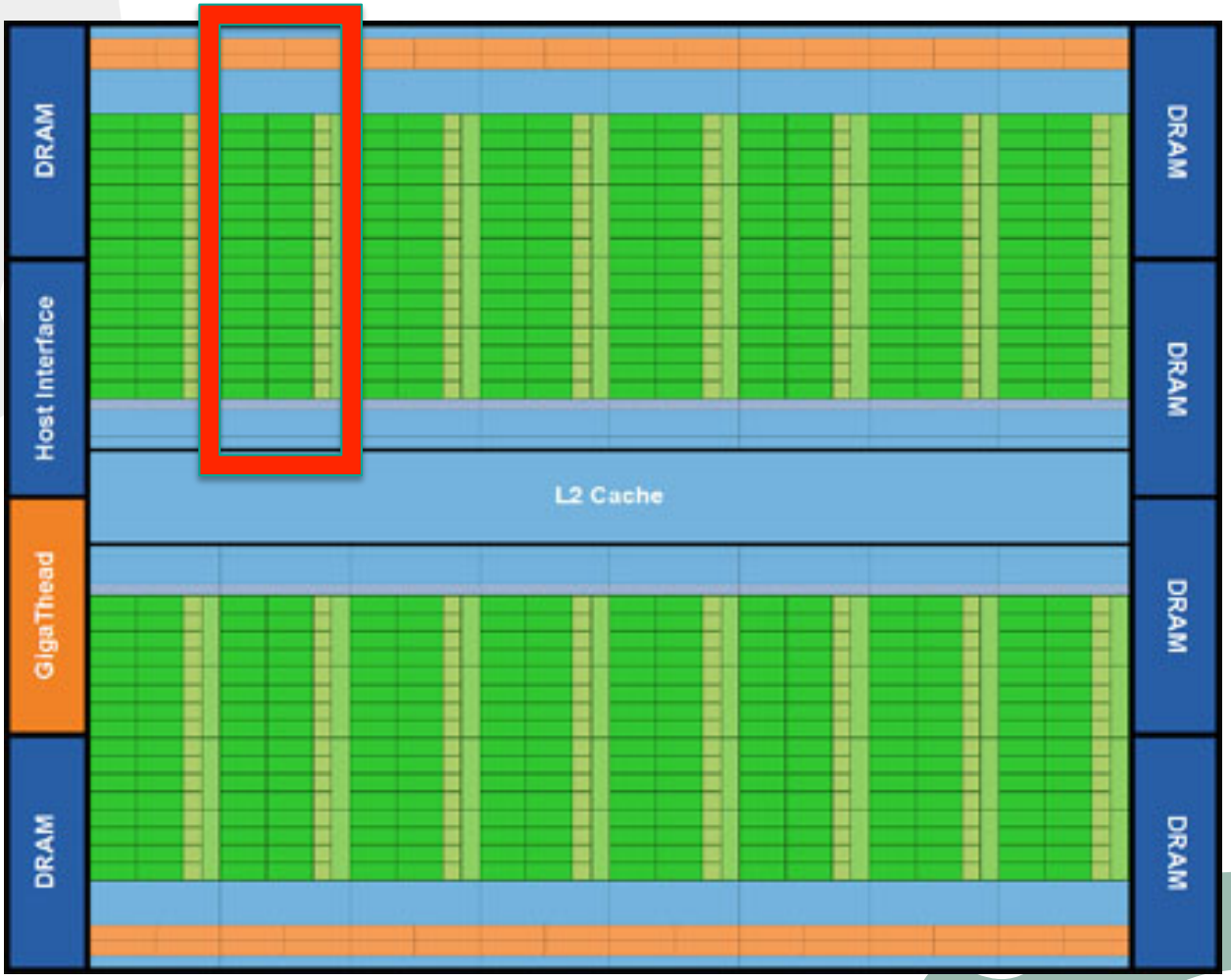
NVIDIA Tegra 2
(2 ARM Kerne + GPU)



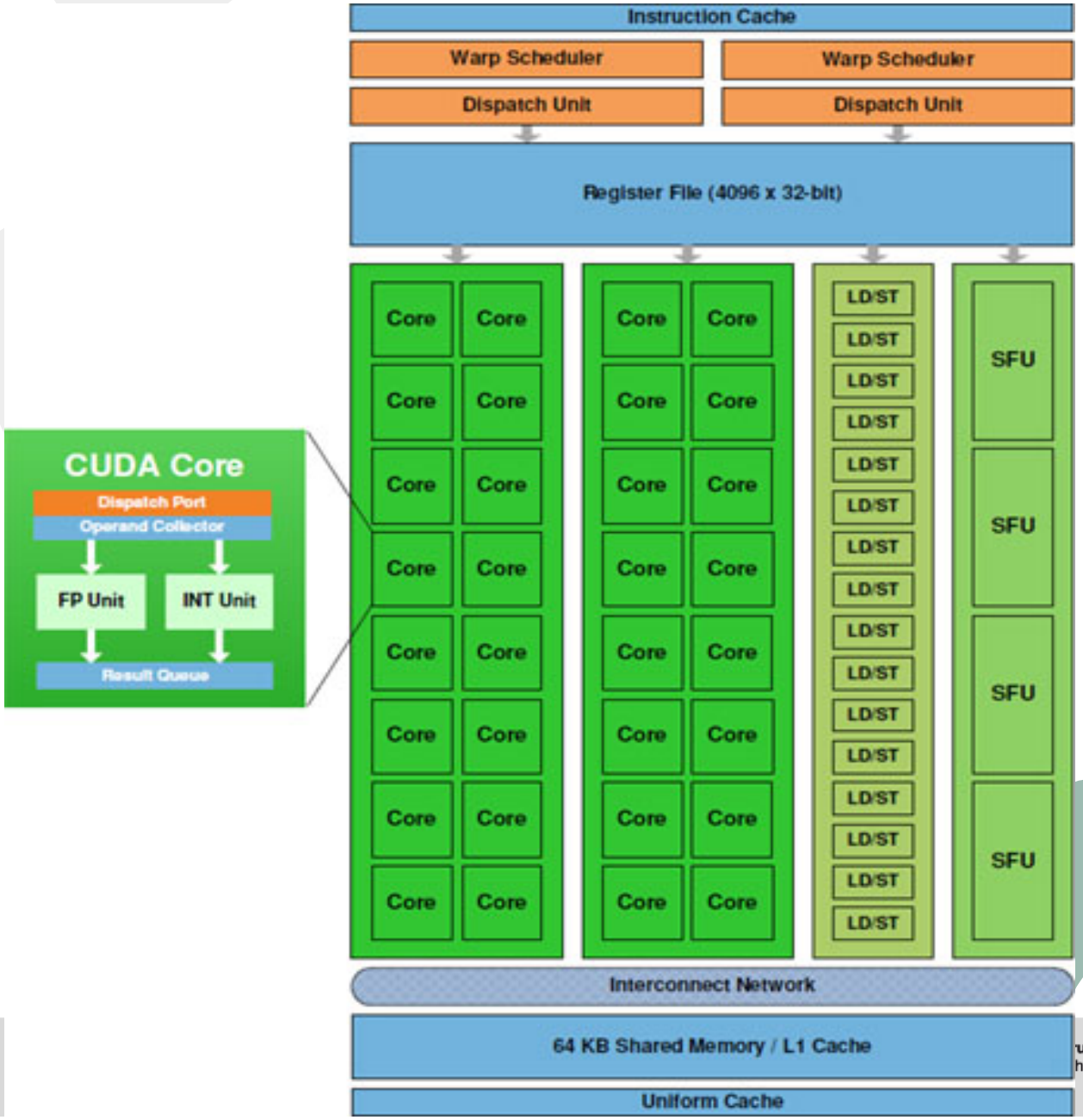


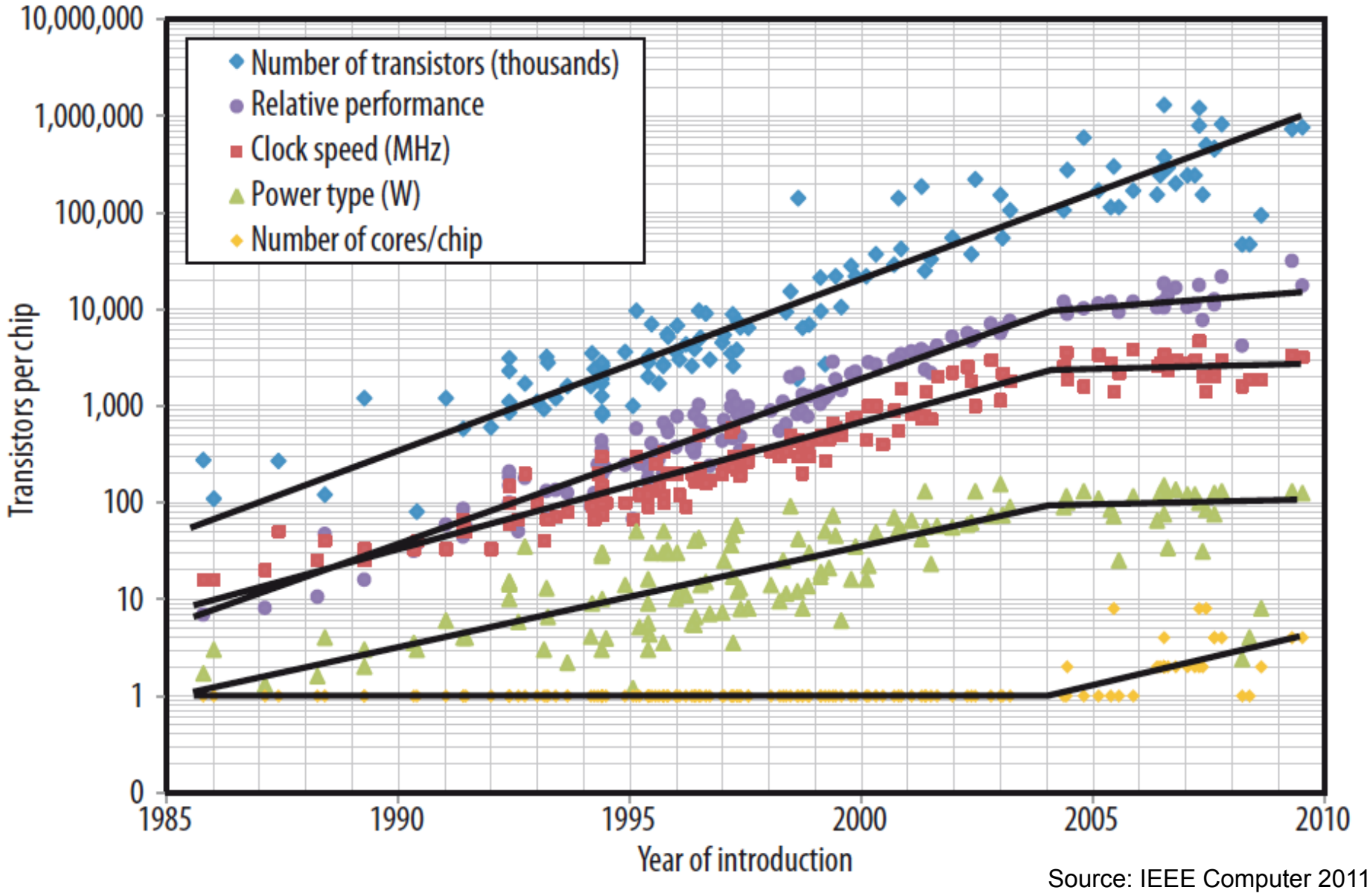
Texas Instruments OMAP 5
(2 ARM Cortex-M4 + 2 ARM Cortex-A15 + GPU)

Nvidia Fermi mit 512 Prozessoren



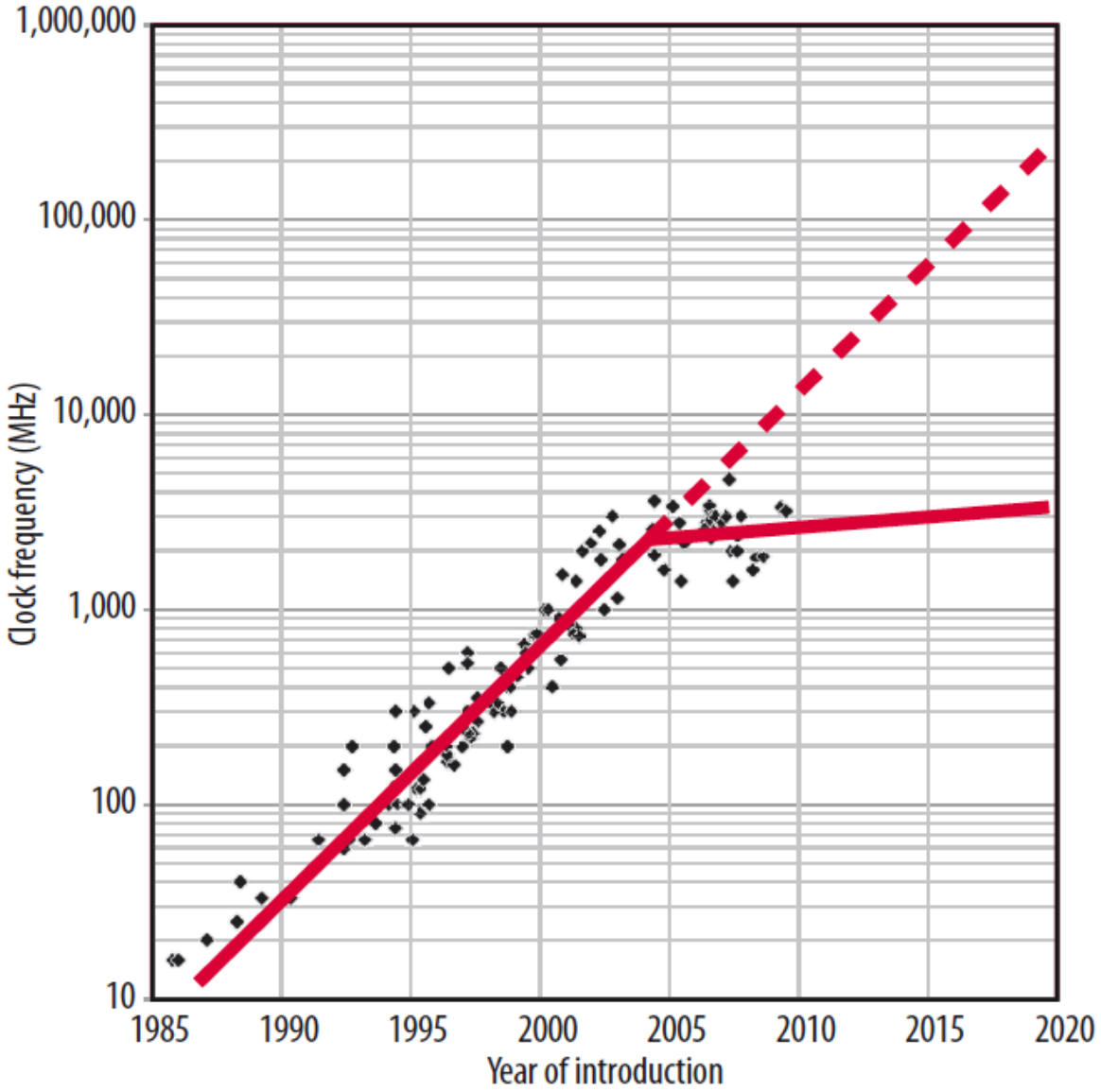
Eine Spalte des Fermi mit 32 Prozessoren





Source: IEEE Computer 2011

Figure 1. Transistors, frequency, power, performance, and processor cores over time. The original Moore's law projection of increasing transistors per chip remains unabated even as performance has stalled.

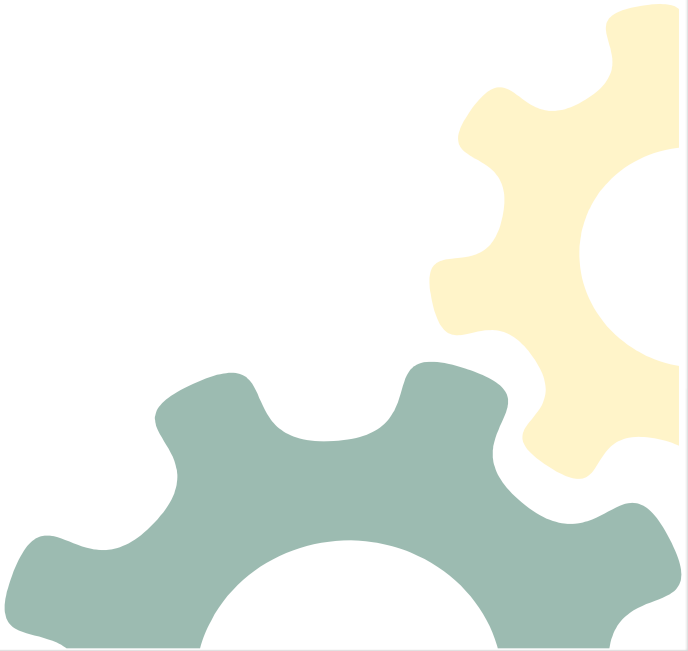


Source: IEEE Computer 2011

Figure 2. Historical growth in single-processor performance and a forecast of processor performance to 2020, based on the ITRS roadmap. A dashed line represents expectations if single-processor performance had continued its historical trend.

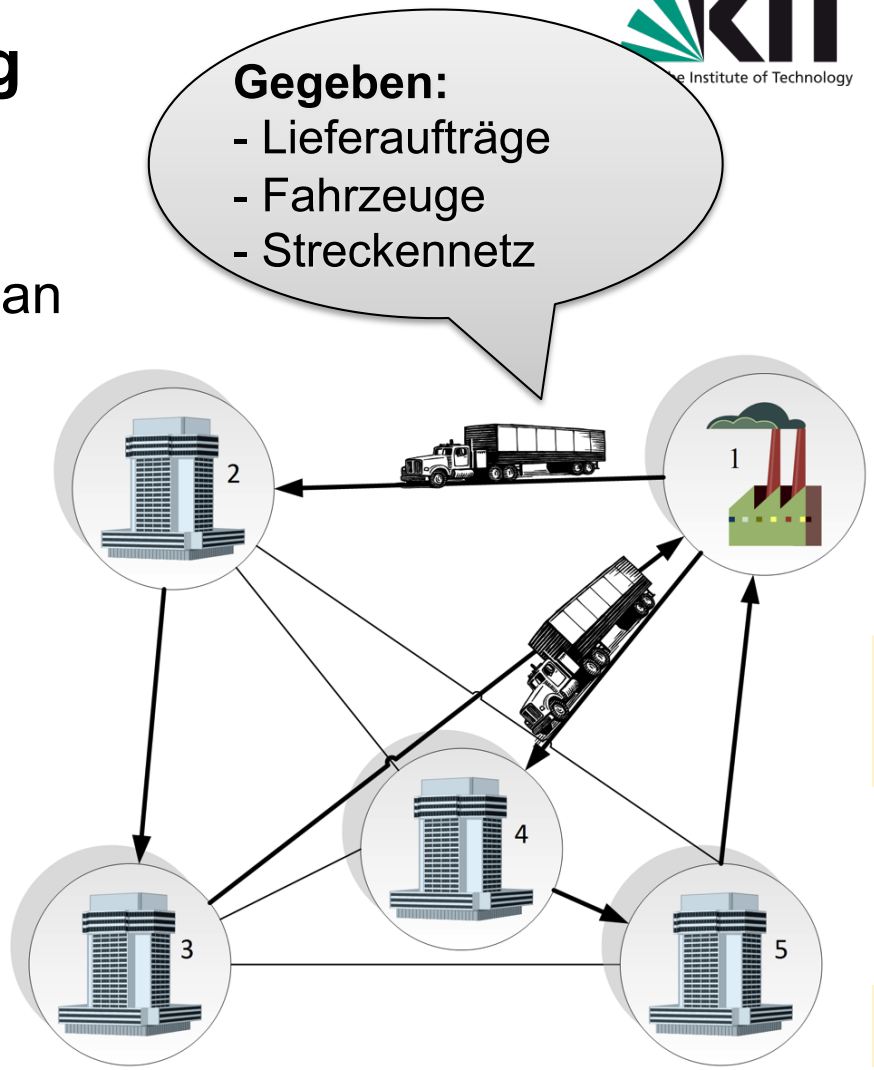
Chance 1: Leistungshungrige Anwendungen

- Mögliche Gebiete
  - Cloudrechner
  - Videoverarbeitung
  - Spiele
  - Datenintensive Anwendungen
  - Optimierungsaufgaben
  - Intelligente Assistenzfunktionen



Beispiel: Transportoptimierung

- Diplomarbeit bei SAP
  - Ziel: kostenoptimaler Transportplan
- Fragestellungen
  - Welche Aufträge?
  - Auf welchen Fahrzeugen?
  - Über welche Strecken?



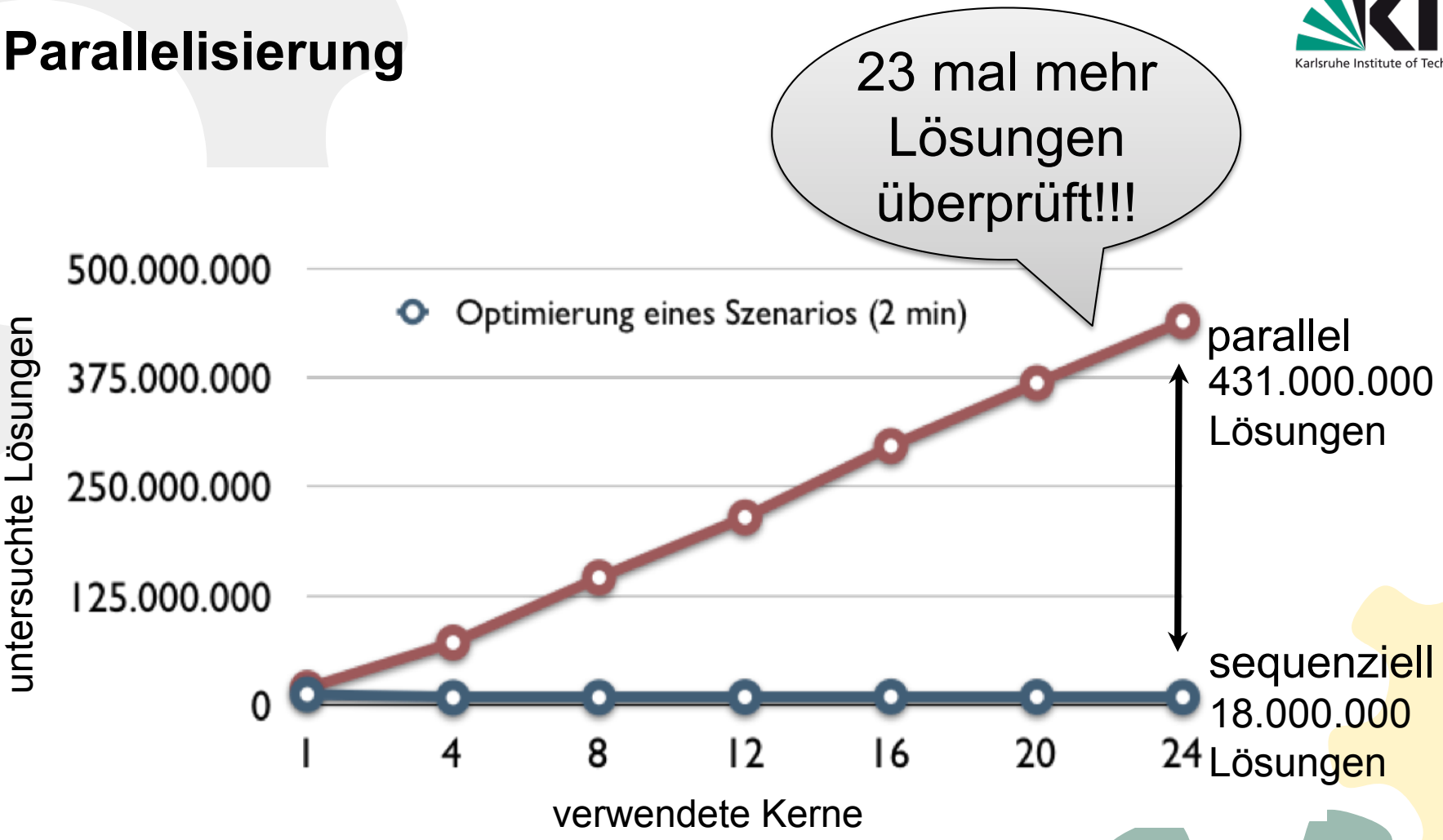
Verallgemeinerung des Problems des Handlungsreisenden

Parallelisierung

- Reale Transportszenarien
- Noch zu berücksichtigen
  - Ruhezeiten der Fahrer, Ladezeiten, Anhänger, mit/ ohne Kühlung, Fähren, Schiffe....
- Sequenzielle Lösung
  - Evolutionärer Algorithmus
  - Mehrere Stunden Rechenzeit für gute Lösungen nötig!

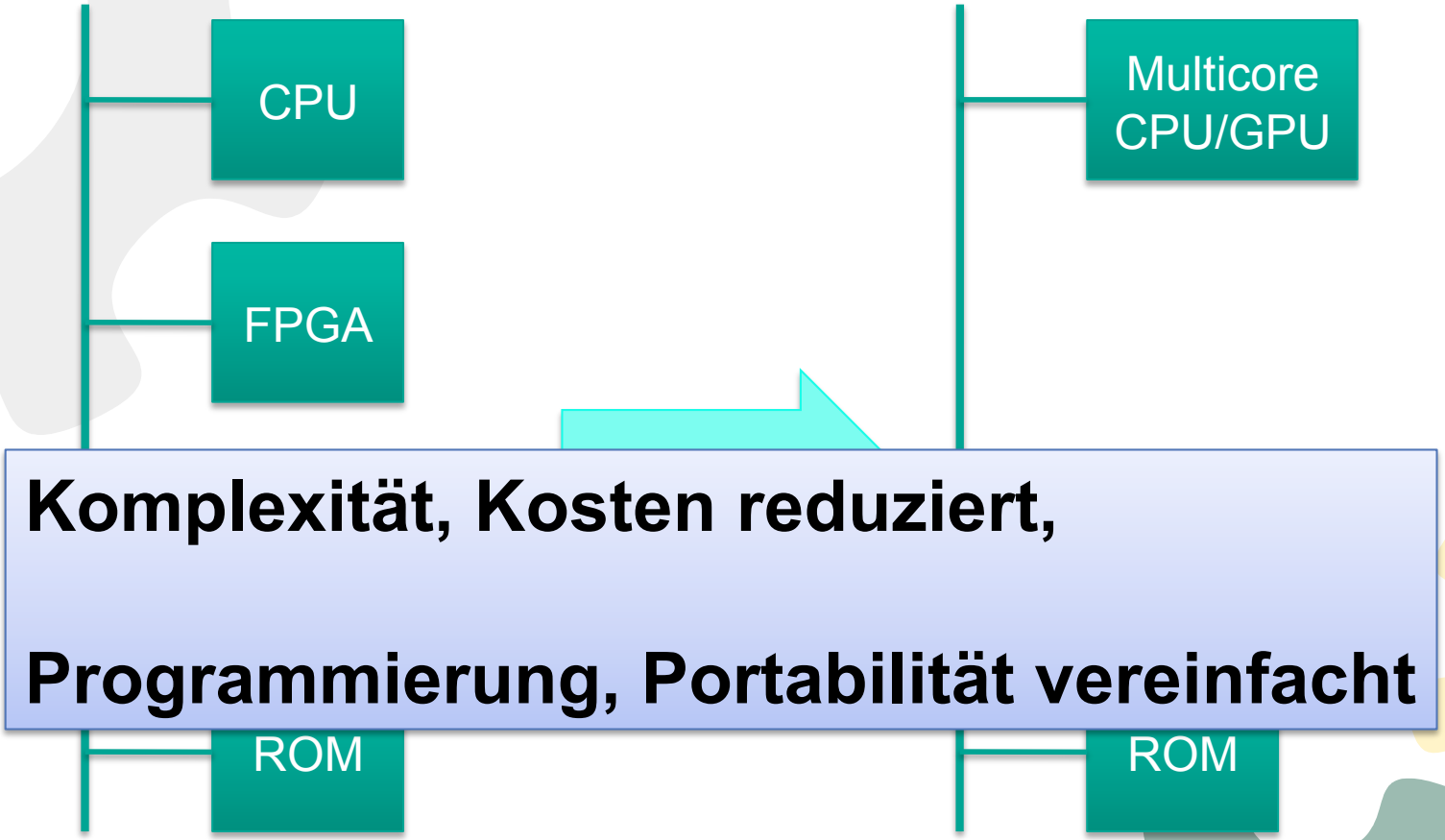
Szenarien	1	2	3
Aufträge	804	1177	7040
Ladedimensionen	3	2	4
Aufladestationen	1	1	3
Abladestationen	31	559	1872
Zwischenstationen	0	5	0
Fahrzeuge	281	680	2011
Fahrzeugtypen	7	3	10
Zeitfenster		1	64

Parallelisierung



Auf Rechner mit 4 Intel Dunnington Chips
(4 * 6 Prozessor-Kerne)

Chance 2: Ersetze FPGAs durch Mehrkerner



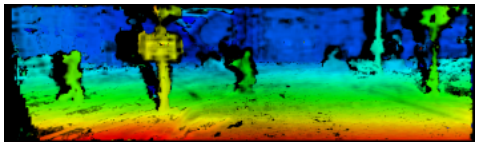
Komplexität, Kosten reduziert,

Programmierung, Portabilität vereinfacht

Beispiel Bildverarbeitung im Fahrzeug

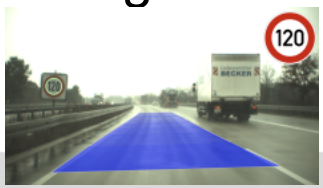
- Strategisches Forschungsfeld Multicore am FZI
- Bildverarbeitung im Automobil-Bereich bislang auf FPGA
  - Tiefenkartenerstellung aus Stereobildern





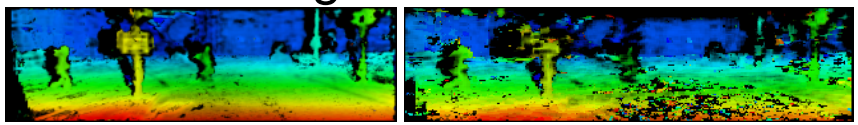
  - Szenensegmentierung (Optischer Fluss)


  - Fahrbahnschätzung und Verkehrszeichenerkennung

Beispiel Bildverarbeitung

- Ziele
  - Standard Prozessoren als Alternative zu FPGAs
  - Konsolidierung
  - Qualitätsverbesserung durch höhere Auflösung
    - Beispiel 1400x400 vs. 400x100

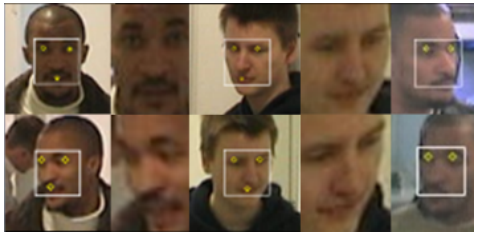


  - Neuartige Bildsensoren
    - Beispiel 360° Kamera (Auflösung 3500x1750)

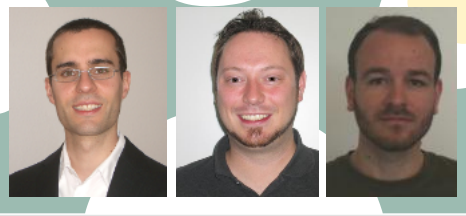
- Bisherige Arbeiten
  - Parallelisierung der Algorithmen
  - Plattformvergleich zwischen x86 CPU/GPU und Tilera
  - Ausführungsstrategien
    - Lastverteilung zwischen CPU und GPU auf x86
    - Zuordnung von Prozessen zu Kernen auf Tilera
- Zukünftige Arbeiten
  - Komplexere Szenarien untersuchen
    - Tiefenkarte + Segmentierung + Fahrbahnschätzung auf einem System

Bildverarbeitung in der Überwachung

- Parallelisierung von Gesichtserkennung



- Fragestellungen
  - Bessere Erkennungsleistung
    - Höher aufgelöste Kamerabilder
    - Parallelbetrieb unterschiedlicher Algorithmen
  - Kosteneffizienz
    - Verarbeitung mehrere Videoströme auf einem System



Chance 3: Ausgliederung von Monitor-Funktionen

- Zuverlässige eingebettete Systeme müssen sich selbst permanent überprüfen
  - Überprüfung auf/ Bereinigung von Speicherfehler
  - Prozessor-Überprüfung
  - ROM-check
  - Höherwertige Überprüfungen
- Diese Überprüfungen können auf einem oder mehreren CPUs durchgeführt werden, ohne die eigentliche Steuerung zu beinträchtigen.
- Eliminiert Schwankungen in der Antwortzeit

Wie wird das alles programmiert?

- Fäden und Synchronisationsmittel:
  - Pthreads,
  - Java threads, etc.
- Biblio
  - In
  - M
  - Pa
  - O
- Spra
  - OpenMP (Fork/Join)
  - Nvidias CUDA (für GPU)
  - KITs Xjava (stromorientiert)

Pankratius, Adl-Tabatabai, Tichy:
Fundamentals of Multicore Software Development,
Taylor & Francis, geplant 2011

Wie wird programmiert?

- Z.B. über stromorientierte Programmerweiterungen
  - Beispiel: Datenkompression in **XJava**
  - Operator „=>“ verknüpft Stufen eines Fließbands (engl. *pipeline*), wie in UNIX

```
compress(File in, File out) {
read(in)  =>  compress()  =>  write(out);
}
```

Schreibt Blöcke in Datei

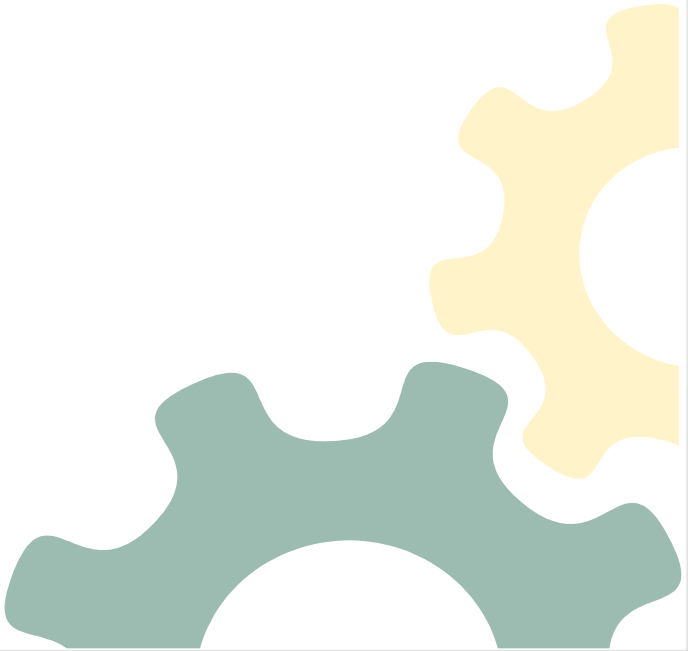
Gepufferter Strom

Gepufferter Strom

Liest Datei ein und gibt Blöcke aus

Liest Blöcke, komprimiert sie

Fragen?



Leistungsoptimierung für Multicore

- Arbeitsbereich: Offline Auto-Tuning
  - Beispiel: Indizierung bei eines Desktopsuche

Zu implementierende Methoden

- `Crawl():List<string>`
- `ParseAlgo1(string s):ParseResult`
- `ParseAlgo2(string s):ParseResult`
- `UpdateIndex(ParseResult p):Index`
- `CreateIndexFile(Index i):void`

Gruppe Leistungsoptimierung für Multicore

```
TunablePipeline MyDesktopSearch
[source:AC_Crawl;sink:AC_CreateIndexFile]
{
    TunableAlternative
    {
        AC_ParseAlgo2
        AC_ParseAlgo1
    },
    AC_UpdateIndex
}
```