Eine kurze Einführung in Rechnerarchitektur und Programmierung von Hochleistungsrechnern als zentrales Werkzeug in der Simulation

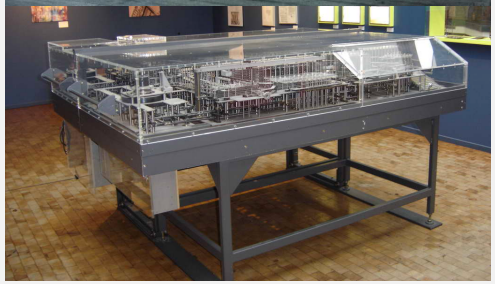
Dr. Jan Eitzinger
Regionales Rechenzentrum (RRZE)
der Universität Erlangen-Nürnberg

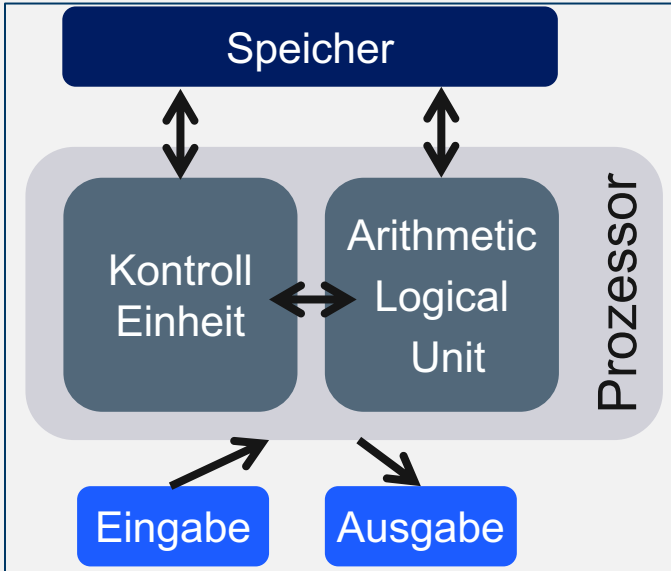
https://hpc.fau.de

On Computable Numbers, with an Application to the Entscheidungsproblem (1936)
Alan Turing

A Symbolic Analysis of Relay and Switching Circuits (1937) **Claude Shannon**

Programmierbarer elektromechanischer Rechner Z1 (1938) **Konrad Zuse**






```
for (int j=0; j<size; j++){
    sum = sum + V[j];
}
```

```
401d08:   f3 0f 58 04 82        addss  xmm0,[rdx + rax * 4]
401d0d:   48 83 c0 01           add    rax,1
401d11:   39 c7                 cmp    edi,eax
401d13:   77 f3                 ja     401d08
```

Instruktions-addressen

Binärcode

Assembler code

- Beschleunigung für relevante Programme
- Was ist technisch umsetzbar?
- Wirtschaftliche Erwägungen

- Takterhöhung
- Parallelisierung
- Spezialisierung

Begriffsklärung

Instruktion – Anweisung bzw. Befehlscode

Adresse – Bezeichnung von Speicherzellen im Hauptspeicher

Register – Direkt mit Rechenwerk verbundener Speicher für Operanden

Assembler – Textdarstellung von binärem Programmcode

(Rechen-)Kern – Bauelement mit mindestens einem Rechenwerk

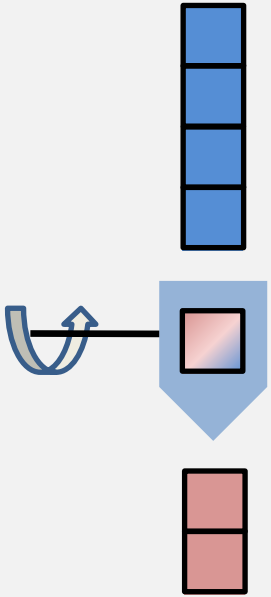
Chip – Ein Stück integrierter Siliziumschaltkreis

Sockel – Steckplatzvorrichtung für Computerprozessoren

(Rechen-)Knoten – Bauelement bestehend aus einer Hauptplatine

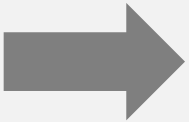
Taktfrequenz – Komponenten laufen mit einem vorgegebenen Takt

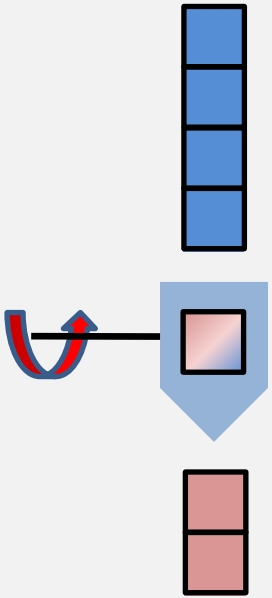
Geschwindigkeitssteigerung durch Takterhöhung



Durchsatz:

1 Einheit pro Sekunde



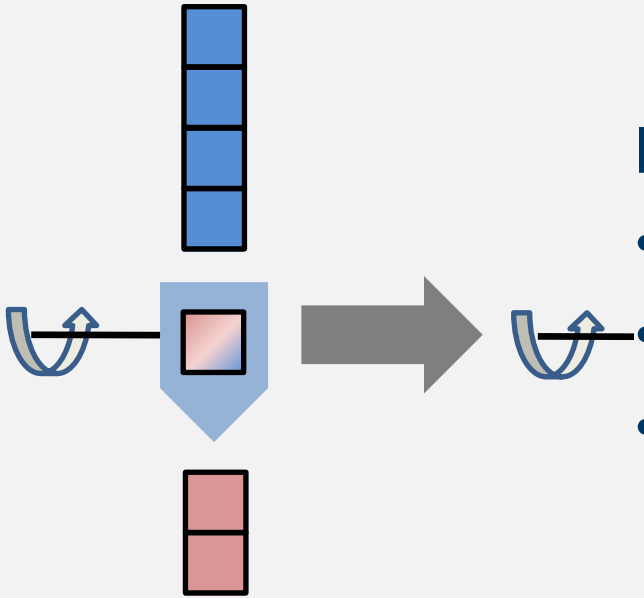


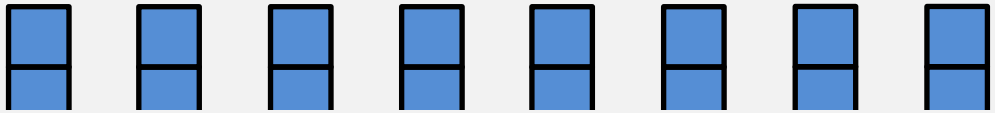
Durchsatz:

4 Einheiten pro Sekunde

Begrenzung: Kühlung physikalisch nicht möglich!

Geschwindigkeitssteigerung durch Parallelisierung





Probleme

- Es muss genug Arbeit vorhanden sein
- Es darf keine Abhängigkeiten geben
- Nutzung meist explizit

Durchsatz:

1 Einheit pro Sekunde

Durchsatz:

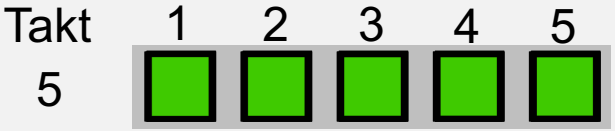
8 Einheiten pro Sekunde

Parallelisierung der Instruktionsausführung

Fließbandprinzip

Instruktionen

Stufen

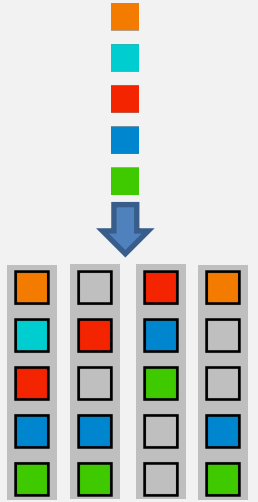


Durchsatz: 1 Instruktion pro Takt

Beschleunigung um Faktor 5

Superskalare Ausführung

4-fach superskalar

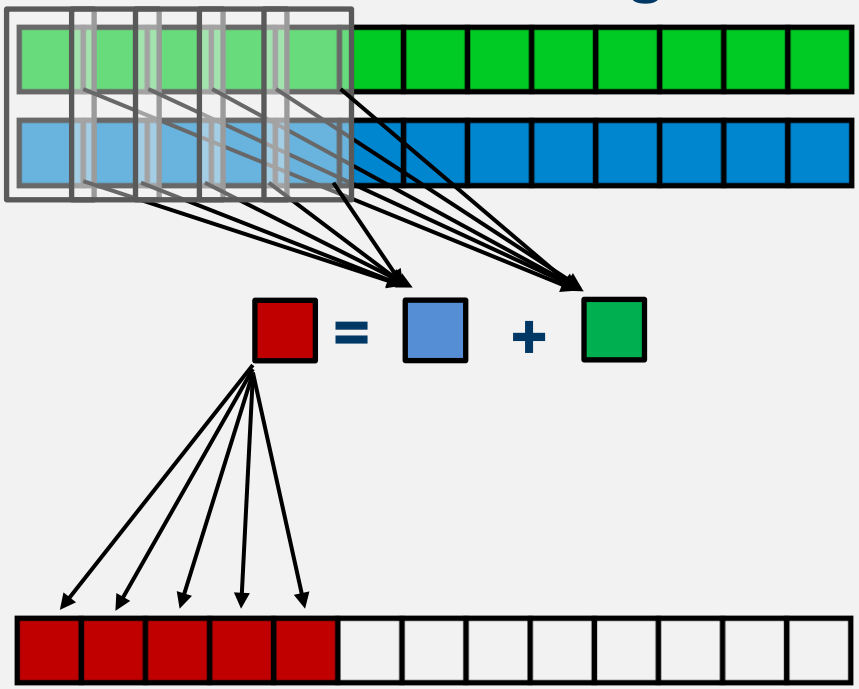


Durchsatz: 4 Instruktionen pro Takt

Datenparallele Ausführungseinheiten (SIMD)

```
for (int j=0; j<size; j++){
    A[j] = B[j] + C[j];
}
```

Skalare Ausführung



Datenparallele Ausführungseinheiten (SIMD)

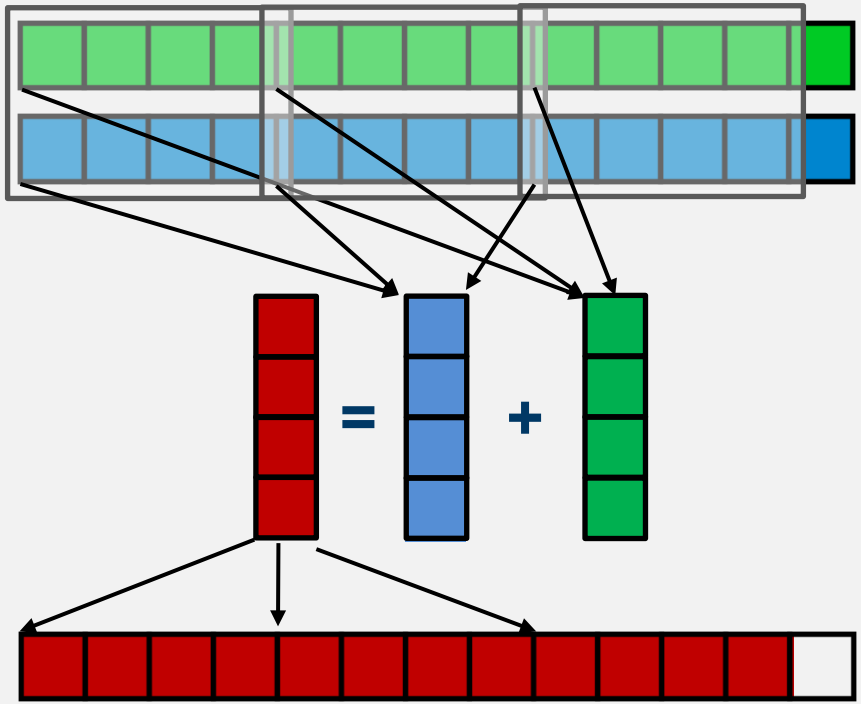
```
for (int j=0; j<size; j++){
    A[j] = B[j] + C[j];
}
```

Breite Register

- 1 Wert

- 4 Werte

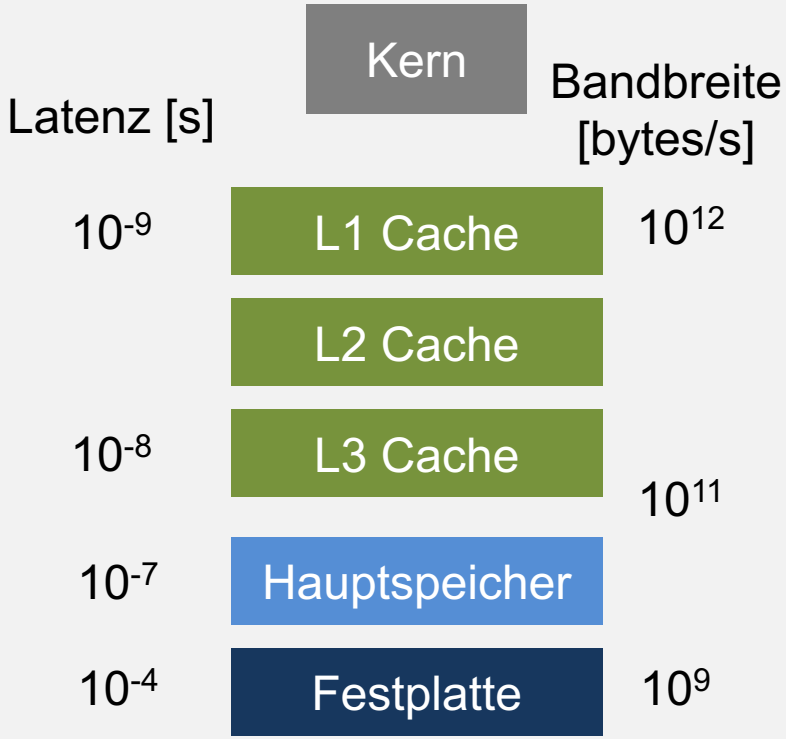
Skalare Ausführung



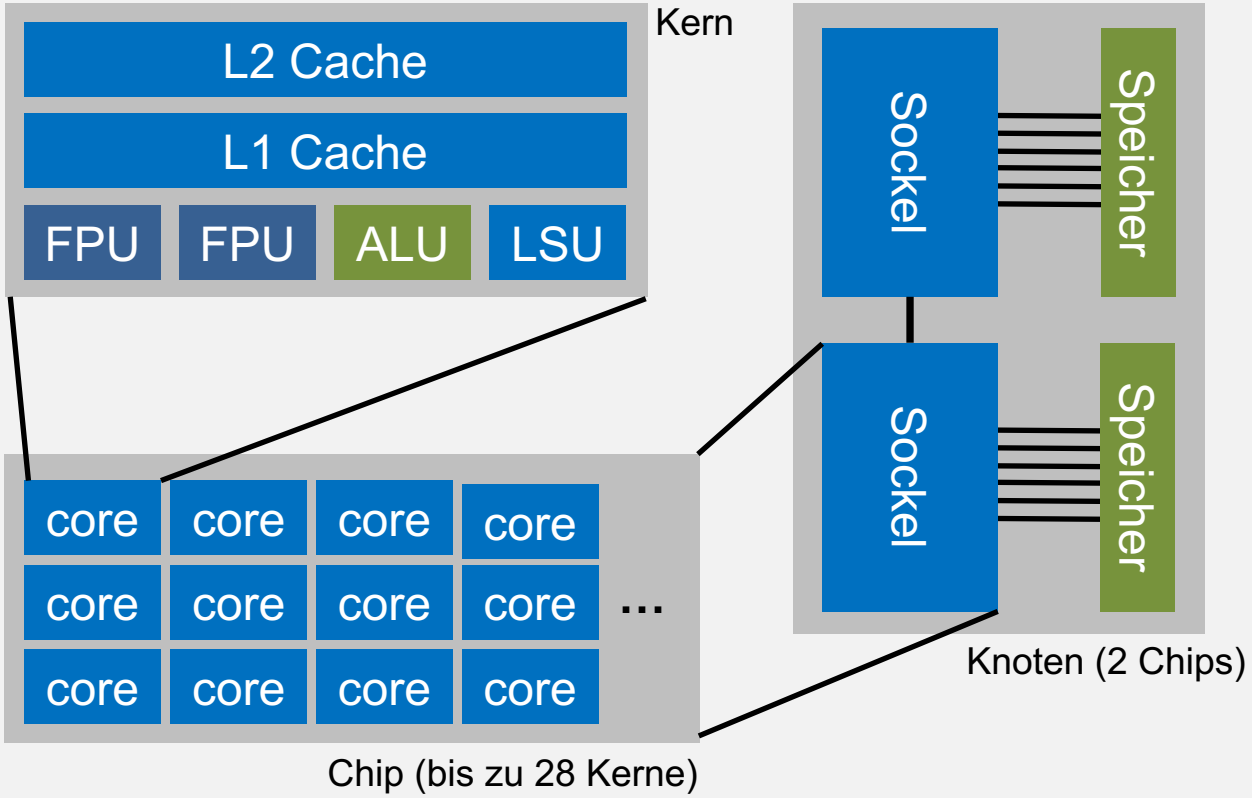
Speicherhierarchie

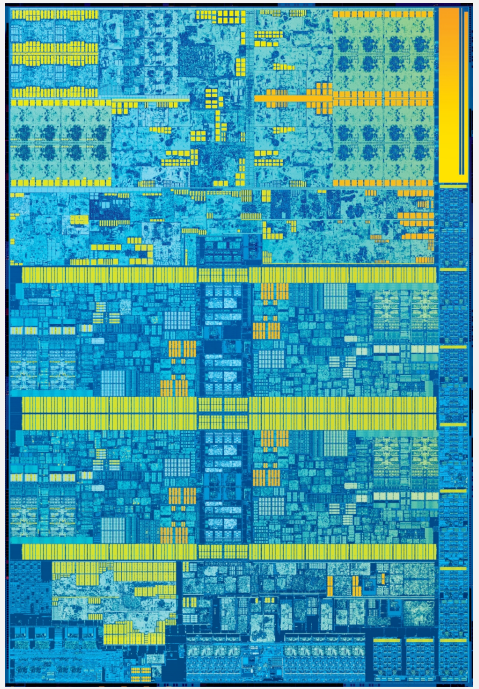
Man kann **entweder** einen *kleinen* und *schnellen* Speicher **oder** einen *großen* und *langsamen* Speicher bauen.

Ziel vieler Optimierungen ist es daher, möglichst viele Daten aus schnellen Speicherebenen zu laden.



Mehrkern-Architekturen





Ca. 8 Mrd. Transistoren auf 500 mm^2

© Intel

Gleitkomma-Rechenleistung pro Knoten

$$P_{Knoten} = n_{Chips} \cdot n_{Kerne} \cdot n_{super}^{FP} \cdot n_{FMA} \cdot n_{SIMD} \cdot f$$

- n_{Chips}: Chips pro Knoten
- n_{Kerne}: Kerne pro Chip
- n_{super}^{FP}: Superskalarität
- n_{FMA}: FMA Faktor
- n_{SIMD}: SIMD Faktor
- f: Takt

Beispiel Intel Xeon Skylake SP 8170

$$P_{Knoten} = 2 \cdot 26 \cdot 2 \cdot 2 \cdot 8 \cdot 2.1 \times 10^9 \frac{\text{Operationen}}{\text{Sekunde}} = \mathbf{3494 \times 10^9} \frac{\text{Operationen}}{\text{Sekunde}}$$

Entwicklung der Intel Xeon Chipleistung

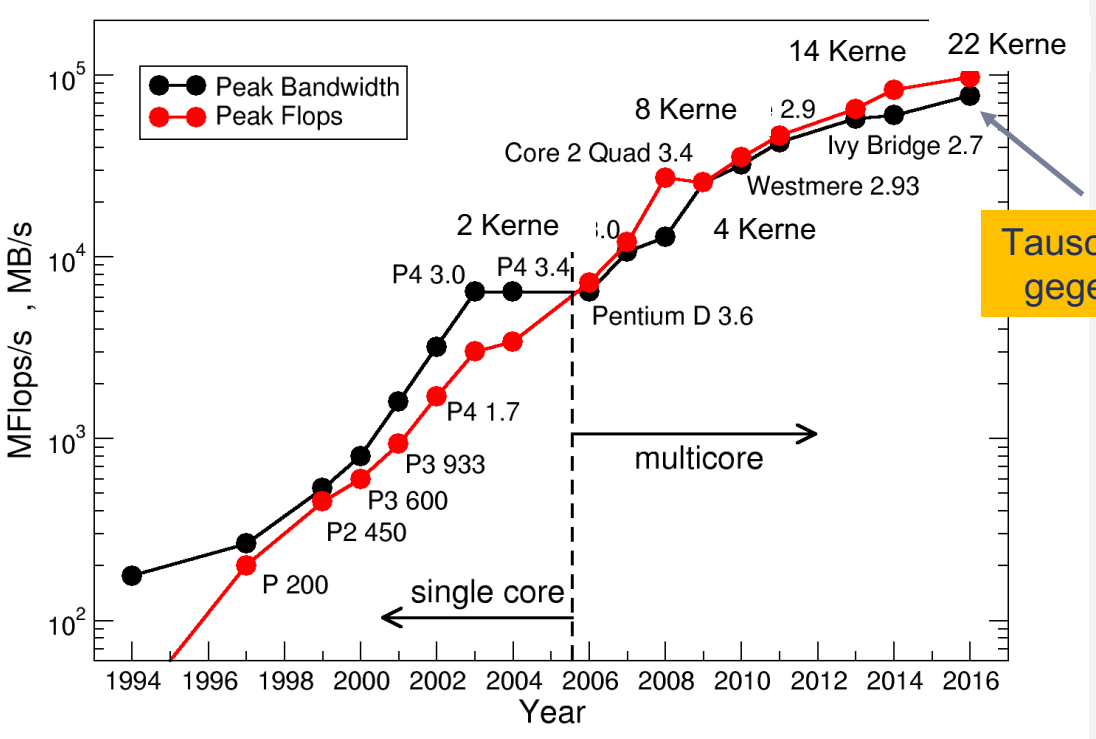
Chipleistung

ohne SIMD

Haswell 120W

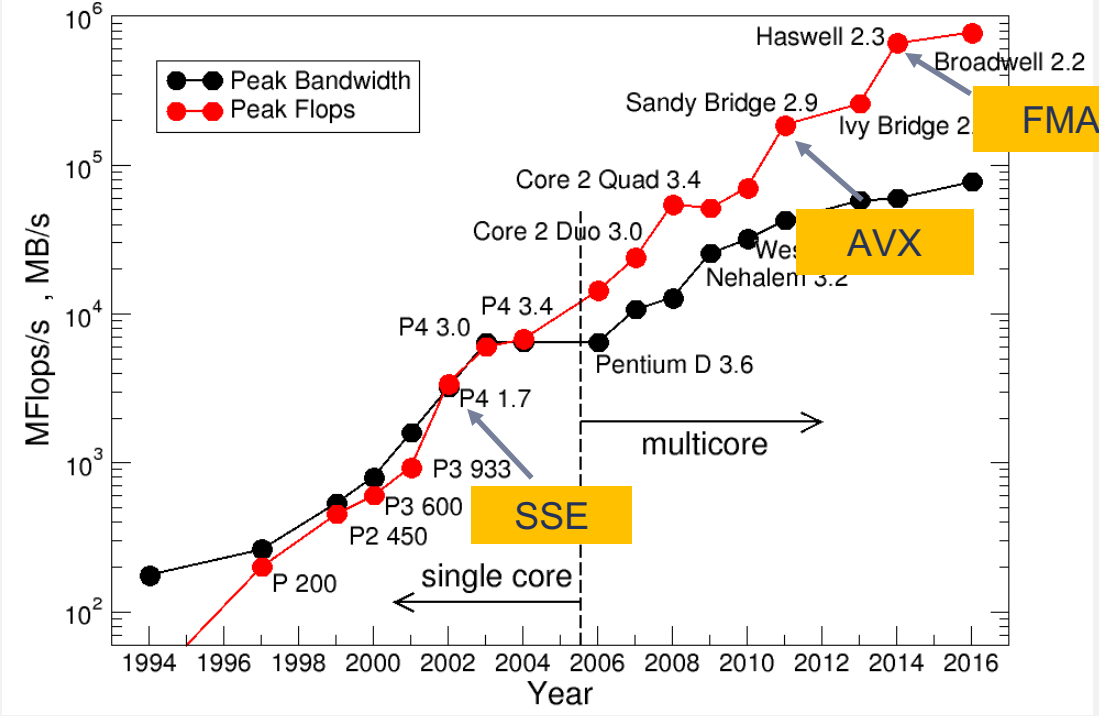
Broadwell 145W

Skylake 165W

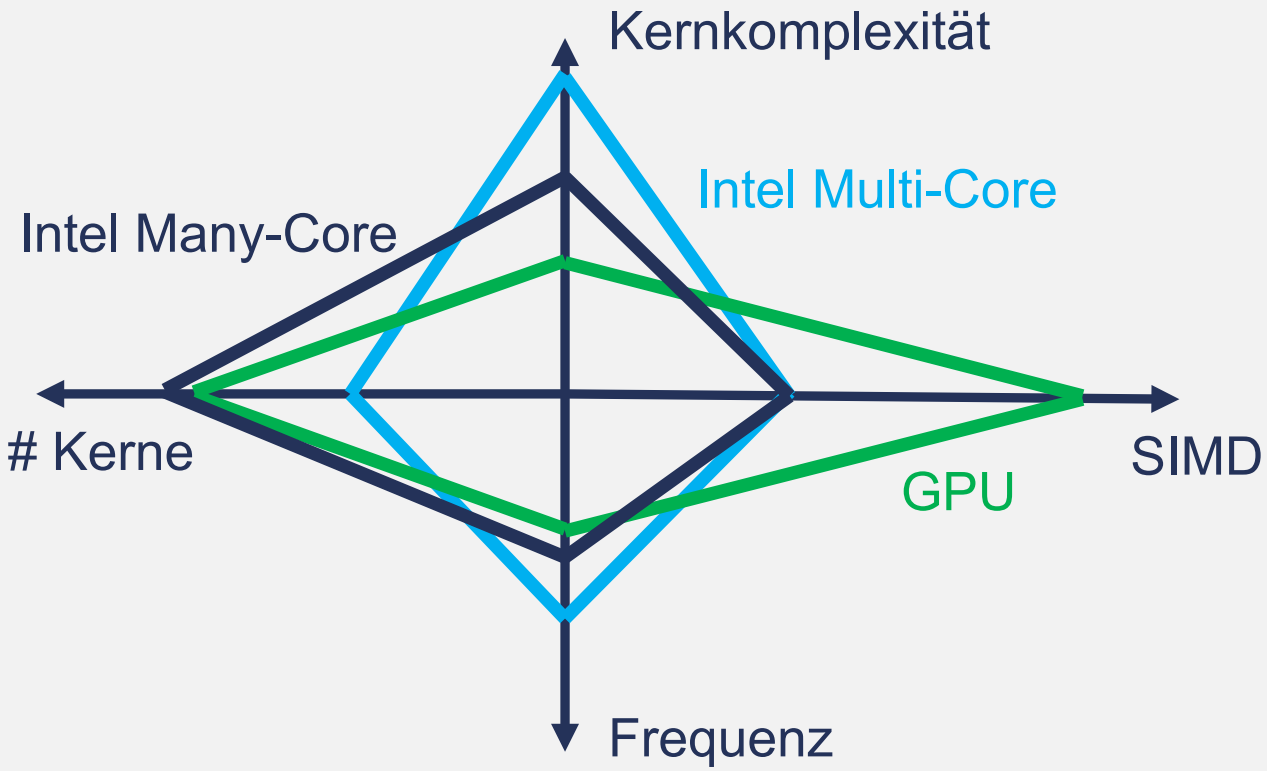


Entwicklung der Intel Xeon Chipleistung

Chipleistung **mit** SIMD/FMA



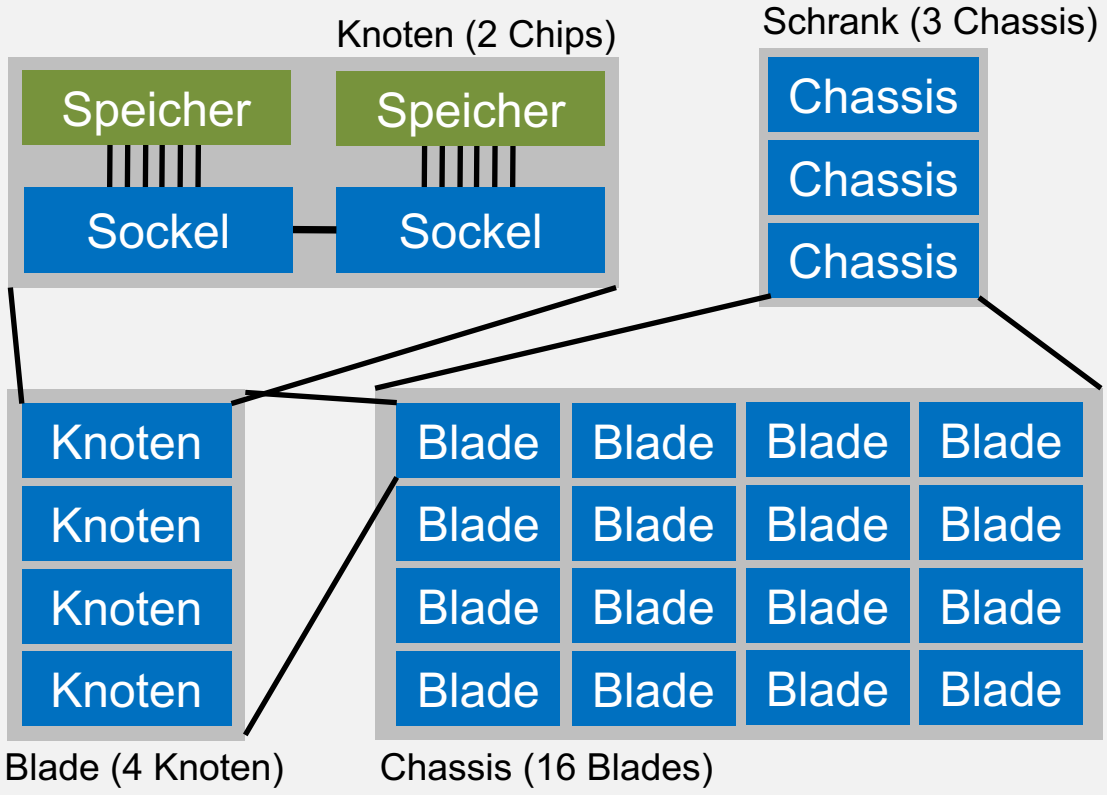
Architektur-Entscheidungen



Eingeschlossene Fläche entspricht:

- Energiebudget
- Transistorbudget

Beispiel-Topologie von Supercomputern



SuperMUC © LRZ

Ein System besteht aus **vielen** Schränken!

Gleitkomma-Rechenleistung Supercomputer

$$P_{System} = n_{Schränke} \cdot n_{Chassis} \cdot n_{Blades} \cdot n_{Knoten} \cdot P_{Knoten}$$

- $n_{Schränke}$: Anzahl Schränke
- $n_{Chassis}$: Chassis pro Schrank
- n_{Blades}: Blades pro Chassis
- n_{Knoten}: Knoten pro Blade
- P_{Knoten}: Leistung Knoten

Beispiel 3 PetaFlop System:

$$P_{System} = 5 \cdot 3 \cdot 16 \cdot 4 \cdot 3494 \times 10^9 \ \frac{\text{Operationen}}{\text{Sekunde}} = \mathbf{3,35 \times 10^{15}} \ \frac{\text{Operationen}}{\text{Sekunde}}$$

960 Knoten mit **49920** Rechenkernen

Top500 Liste: Linpack

Rank	Site	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	National Supercomputing Center in Wuxi	**Sunway TaihuLight** - Sunway MPP, Sunway SW26010 260C 1.45GHz,	10,649,600	93,014.6	125,435.9	15,371

HPC@RRZE

Meggie

728 Rechenknoten (14.560 Kerne)

Linpack: P_{max} = 470 TF/s

#346@TOP500 – Nov. 2016

5	DOE/SC/Oak Ridge National Laboratory United States	**Titan** - Cray XK7, Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x Cray Inc.	560,640	17,590.0	27,112.5	8,209

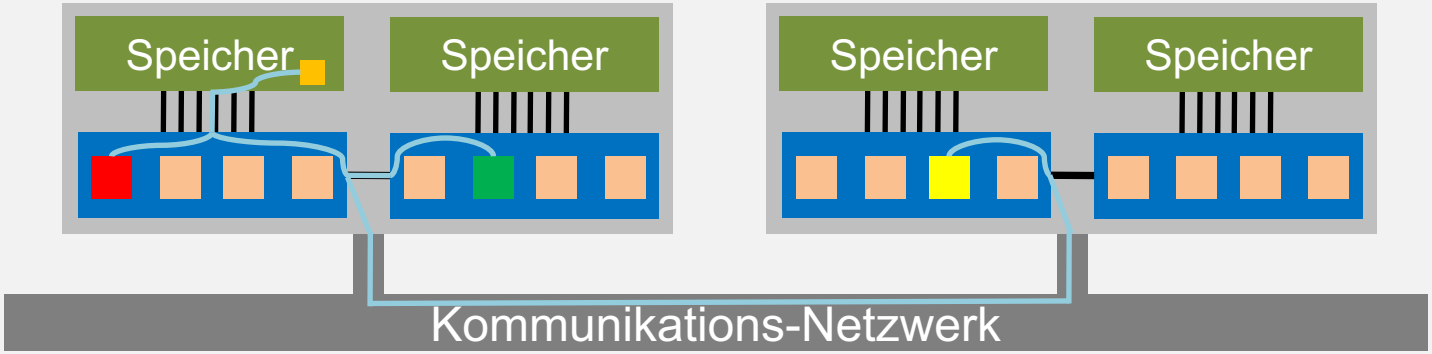
Rank	Site	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	National Supercomputing Center in Wuxi China	**Sunway TaihuLight** - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway NRCPC	10,649,600	93,014.6	125,435.9	15,371
2	National Super Computer Center in Guangzhou China	**Tianhe-2 (MilkyWay-2)** - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P NUDT	3,120,000	33,862.7	54,902.4	17,808
3	Swiss National Supercomputing Centre (CSCS) Switzerland	**Piz Daint** - Cray XC50, Xeon E5-2690v3 12C 2.6GHz, Aries interconnect , NVIDIA Tesla P100 Cray Inc.	361,760	19,590.0	25,326.3	2,272
4	Japan Agency for Marine-Earth Science and Technology Japan	**Gyoukou** - ZettaScaler-2.2 HPC system, Xeon D-1571 16C 1.3GHz, Infiniband EDR, PEZY-SC2 700Mhz ExaScaler	19,860,000	19,135.8	28,192.0	1,350
5	DOE/SC/Oak Ridge National Laboratory United States	**Titan** - Cray XK7, Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x Cray Inc.	560,640	17,590.0	27,112.5	8,209

Top500 Liste: Linpack Benchmark

93 Pflops/s

15 MWatt

Gemeinsamer und verteilter Speicher



Innerhalb eines Knotens:

- Austausch über Hauptspeicher

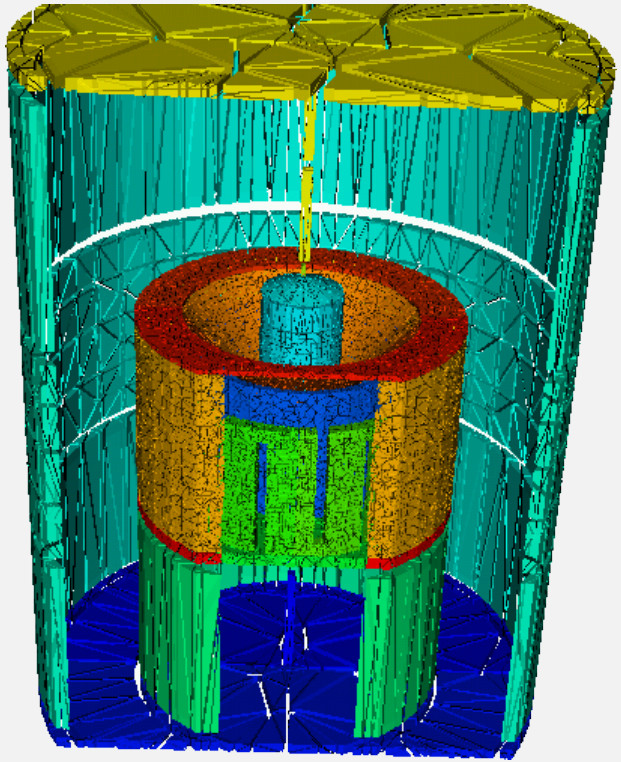
Gemeinsamer Speicher

Zwischen mehreren Knoten:

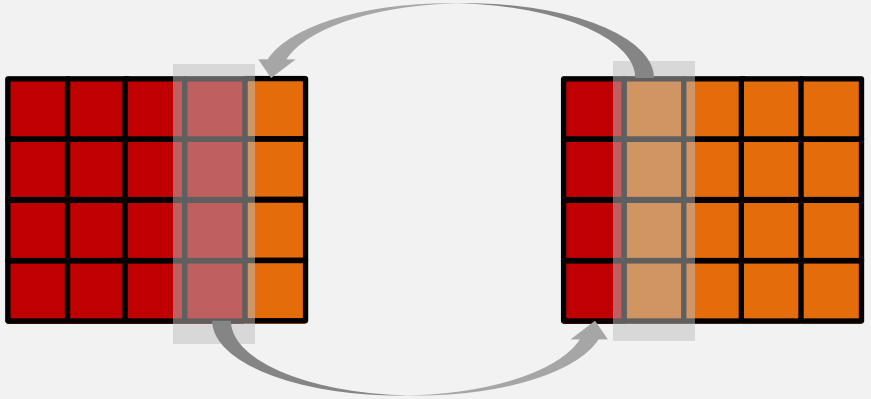
- Austausch über Netzwerk-nachrichten

Verteilter Speicher

Gebietszerlegung



Datenaustausch an Gebietsgrenzen



Rechnen

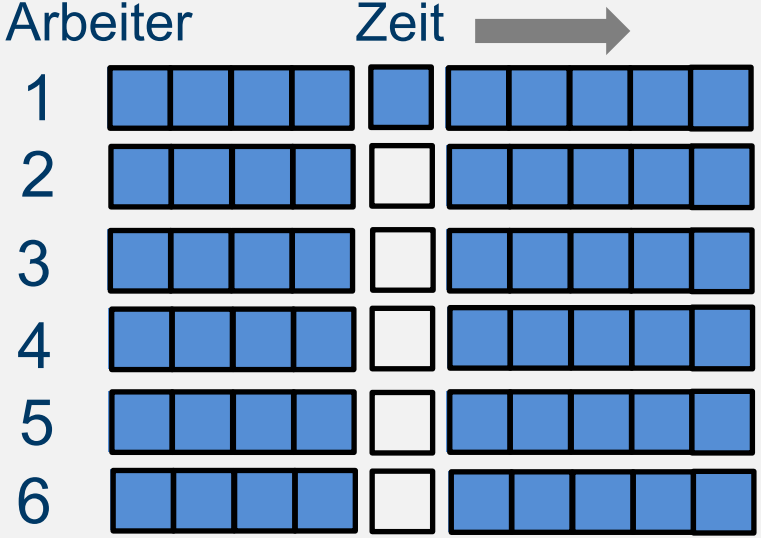
Datenaustausch

Rechnen

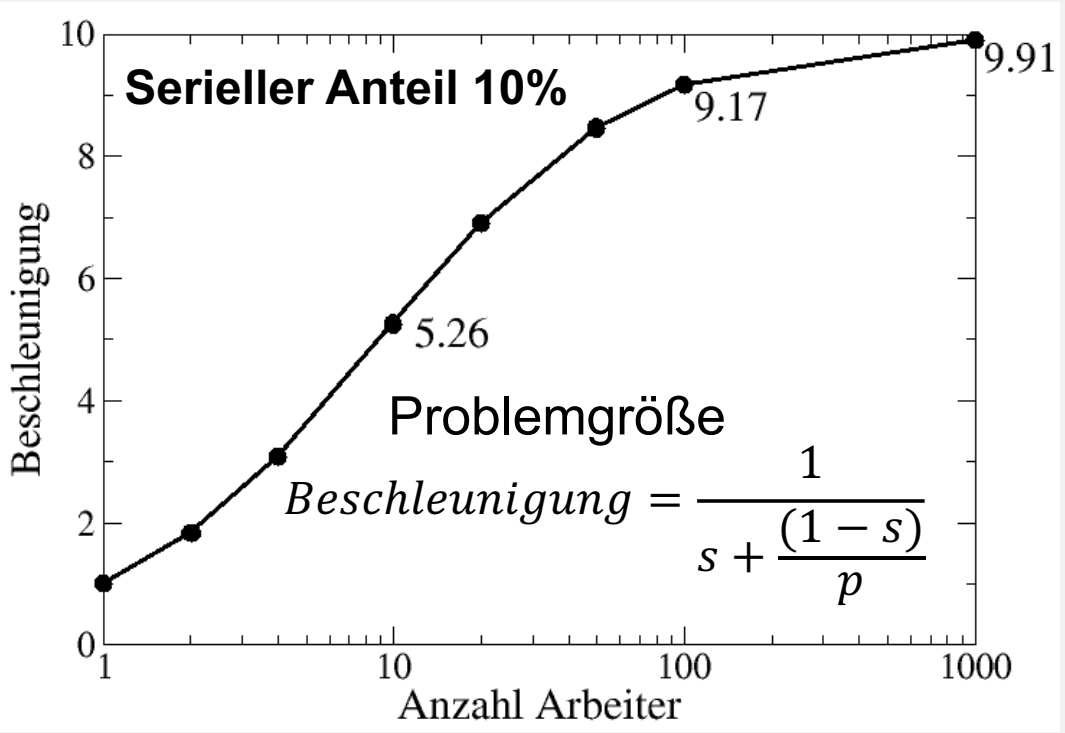
Datenaustausch

Starke Skalierung

Maximale parallele Beschleunigung bei **konstanter** Problemgröße



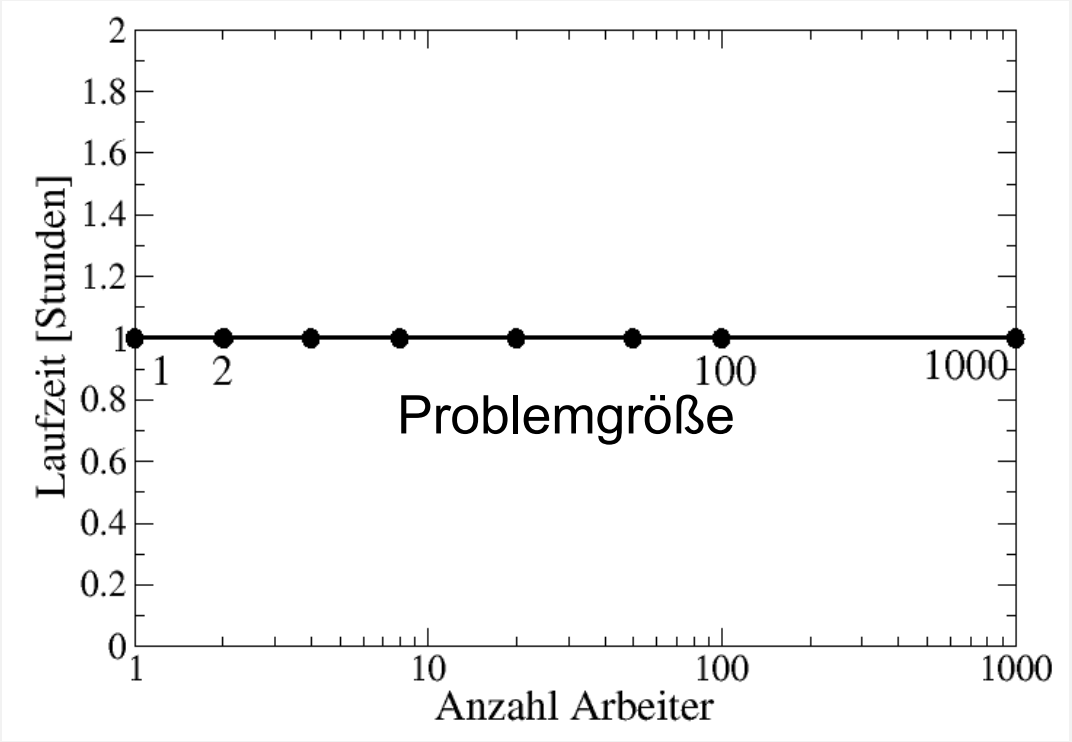
Amdahls Gesetz



Schwache Skalierung

Lösen eines größeren Problems in gleicher Zeit unter Verwendung einer entsprechenden Anzahl von Arbeitern.

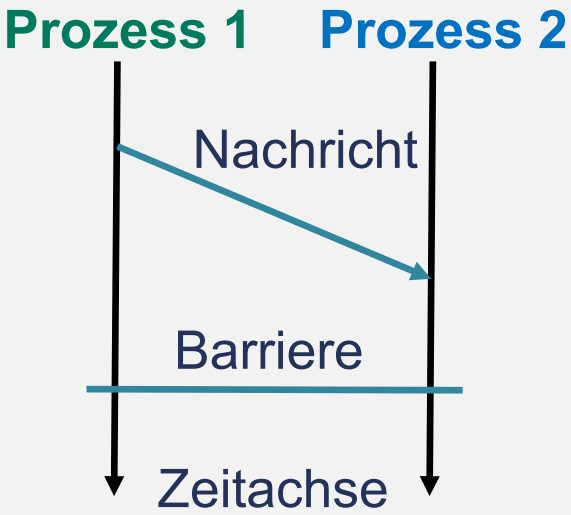
Gustavsons Gesetz



Wie programmiert man HPC Systeme?

- Message Passing Interface (MPI) Standard

- Programmierschnittstellen für:
  - Punkt-zu-Punkt Kommunikation
  - Kollektive Kommunikation
  - Parallele Dateioperationen



- Erste Spezifikation 1994
- Funktioniert für gemeinsamen und verteilten Speicher

MPI + X

X: Beliebiges anderes Programmiermodell auf Knotenebene

- **OpenMP** – Programmierschnittstelle für gemeinsamen Speicher. Erster Standard 1997

- **CUDA** – Programmierschnittstelle für GPGPUs von Nvidia

Weitere Lösungen:

OpenACC, OpenCL, SHMEM …

Alternativen zu MPI

- Partitioned global address space (**PGAS**) Programmiermodelle:
  - Unified Parallel C und Coarray Fortran
  - Chapel und X10
  - SHMEM

- Aus dem Big-Data Umfeld:
  - Apache Spark
  - Apache Hadoop
  - Google TensorFlow

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

Vielen Dank für Ihre Aufmerksamkeit!

https://hpc.fau.de

https://rrze.fau.de