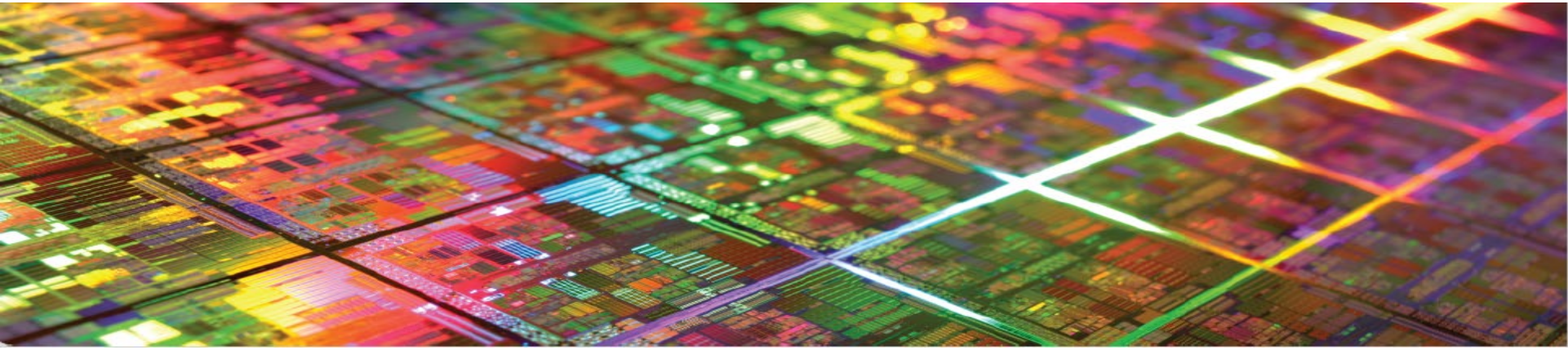
Karlsruher Institut für Technologie

Rechnerorganisation

Prof. Dr. Wolfgang Karl

Vorlesung im Wintersemester 2025/2026 – Foliensatz: RO25-FS13



Kapitel 8
Cache-Speicher

- Speicherhierarchie
- Systemaufbau mit Cache-Speicher
- Grundlegende Arbeitsweise
- Cache-Organisationsformen
- **Grundlegende Fragen beim Entwurf**
- Gültigkeitsproblem

Grundlegende Fragen beim Entwurf

- **Entwurfsparameter**
  - Kapazität
  - Blockgröße
  - Assoziativität
  - Ersetzungsstrategie
  - Aktualisierungsstrategie (write-back, write-through)
  - Cache-Hierarchie
    - Cache-Anbindung (look-through, look-aside, backside)
  - Ziel: möglichst gute Cache-Leistung

Bewertung der Cache-Leistung

- **Metriken**
  - Die **Trefferrate (hit rate)** bezeichnet die Trefferquote im Cache:

$$\text{Trefferrate} = \frac{\text{Anzahl Treffer}}{\text{Anzahl Zugriffe}}$$

  - Die **Fehlzugriffsrate (miss rate)** bezeichnet den Anteil der Cache-Zugriffe, bei denen der Block mit dem adressierten Objekt nicht im Cache vorhanden ist.

$$\text{Fehlzugriffsrate} = 1 - \text{Trefferrate}$$

Bewertung der Cache-Leistung

- **Metriken**
  - Die **mittlere Zugriffszeit t_{access} (average memory access time)** berechnet sich annähernd wie folgt:

$$t_{access} = \text{Trefferrate} * t_{hit} + (1 - \text{Trefferrate}) * t_{miss}$$

  - mit t_{hit}: **Zugriffszeit auf den Cache bei einem Treffer**
    t_{miss}: **Fehlzugriffsaufwand (miss penalty)**
    Zugriffszeit auf den Hauptspeicher, (genauer: Zeit für die Zugriffe auf alles, was nach dem Cache kommt; ggf. erst eine weitere Cache-Ebene)

Bewertung der Cache-Leistung

- **Metriken**
  - **CPU-Ausführungszeit (CPU-Zeit)**
    - Gleichung der Prozessorleitung (siehe Kapitel 4, Foliensatz RO25-FS05, Folie 85)

$$\text{CPU}_{\text{Zeit}} = \text{IC} * \text{CPI} * \text{Zyklendauer}$$

    - **Erweiterung der Formel für die Prozessorleistung**

$$\text{CPU}_{\text{Zeit}} = \left(\text{CPU}_{\text{Taktzyklen}} + \text{Speicherstillstandszyklen}\right) * \text{Zyklendauer}$$

      - **Speicherstillstandszyklen (memory stall cycles)**
        - Anzahl Taktzyklen, in denen die CPU auf das Speichersystem wartet
        - Ursache: Fehlzugriffe (vereinfachte Annahme)

$$\text{Speicherstillstandszyklen} = (\text{Anzahl Fehlzugriffe} * \text{Fehlzugriffsaufwand})$$

Fehlzugriffsaufwand: miss penalty

Bewertung der Cache-Leistung

- **Metriken**
  - **CPU-Ausführungszeit (CPU-Zeit)**
    - Gleichung der Prozessorleitung (siehe Kapitel 4, Foliensatz RO25-FS05, Folie 85)

$$\text{CPU}_{\text{time}} = \text{IC} * \text{CPI} * \text{Clock cycle time}$$

    - **Erweiterung der Formel für die Prozessorleistung**

$$\text{CPU}_{\text{time}} = \left(\text{CPU}_{\text{Clock cycles}} + \text{memory stall cycles}\right) * \text{Clock cycle time}$$

      - **Speicherstillstandszyklen (memory stall cycles)**

$$\text{Memory stall cycles} = (\text{Number misses} * \text{miss penalty})$$

Grundlegende Fragen beim Entwurf

- **Verbesserung der Cache-Leistung**
  - **Betrachtung der mittleren Zugriffszeit t_{access}**

$$t_{access} = \text{Trefferrate} * t_{hit} + (1 - \text{Trefferrate}) * t_{miss}$$

(1) **Reduzierung der Zugriffszeit auf den Cache bei einem Treffer (t_{hit})**
  - Kleine und einfache Caches

Grundlegende Fragen beim Entwurf

- **Verbesserung der Cache-Leistung**
  - **Betrachtung der mittleren Zugriffszeit t_{access}**

$$t_{access} = \text{Trefferrate} * t_{hit} + (1 - \text{Trefferrate}) * t_{miss}$$

(2) Reduzierung Fehlzugriffsrate $(1 - \text{Trefferrate})$

- Höhere Assoziativität
- Größere Blockgrößen
- Größere Kapazitäten

(3) Reduzierung der Kosten bei Fehlzugriffen (Fehlzugriffsaufwand, t_{miss})

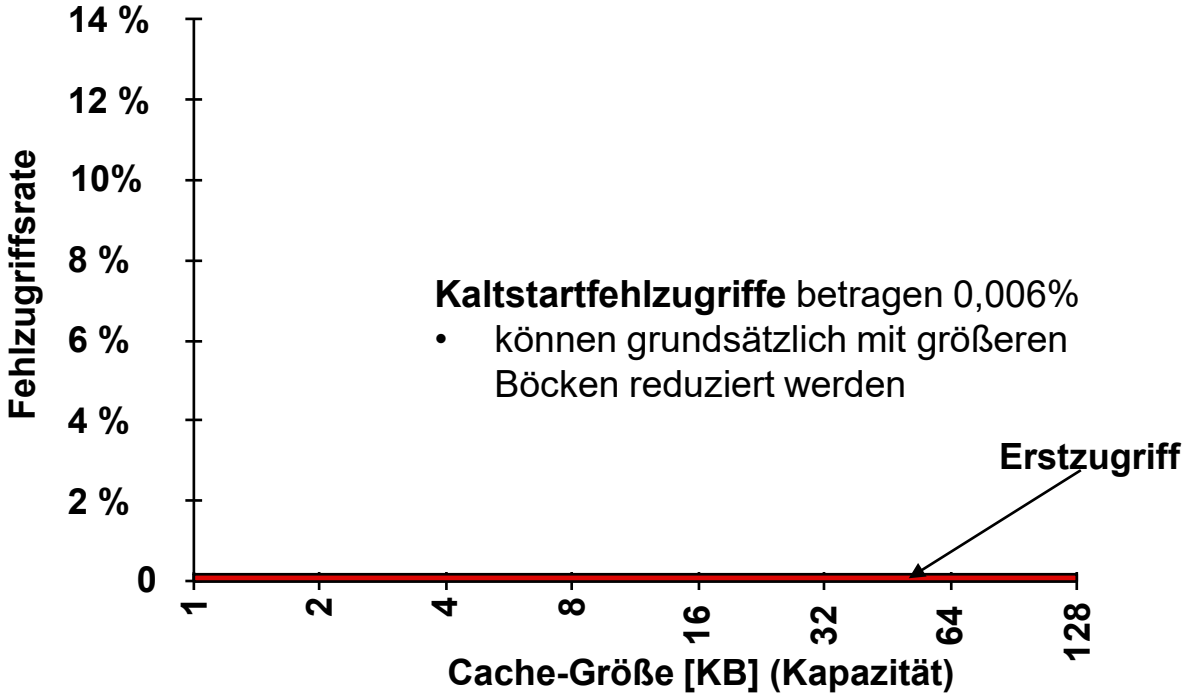
- Einführung einer Cache-Hierarchie

Grundlegende Fragen beim Entwurf

- **Ursachen für Fehlzugriffe**
  - **Erstzugriff-Fehlzugriff (Kaltstart, compulsory miss (obligatorisch), cold start miss):**
    - Beim ersten Zugriff auf einen Speicherblock kann dieser noch nicht im Cache sein und muss erstmals in den Cache geladen werden.
  - **Kapazitäts-Fehlzugriff (Speicherüberlastungs-Fehlzugriff, capacity miss):**
    - Falls der Cache-Speicher nicht alle Speicherblöcke, die während der Ausführung eines Programms benötigt werden, aufnehmen kann, müssen Blöcke aus Cache-Zeilen verdrängt und eventuell später wieder geladen werden.
  - **Konflikt-Fehlzugriff (Adresskonflikt-Fehlzugriff, Kollisions-Fehlzugriff, conflict miss):**
    - Treten in direkt abgebildeten oder satzassoziativen Cache-Speichern beschränkter Größe auf, wenn mehrere Blöcke um einen Satz konkurrieren;
    - Können in einem vollassoziativen Cache-Speicher derselben Größe eliminiert werden;

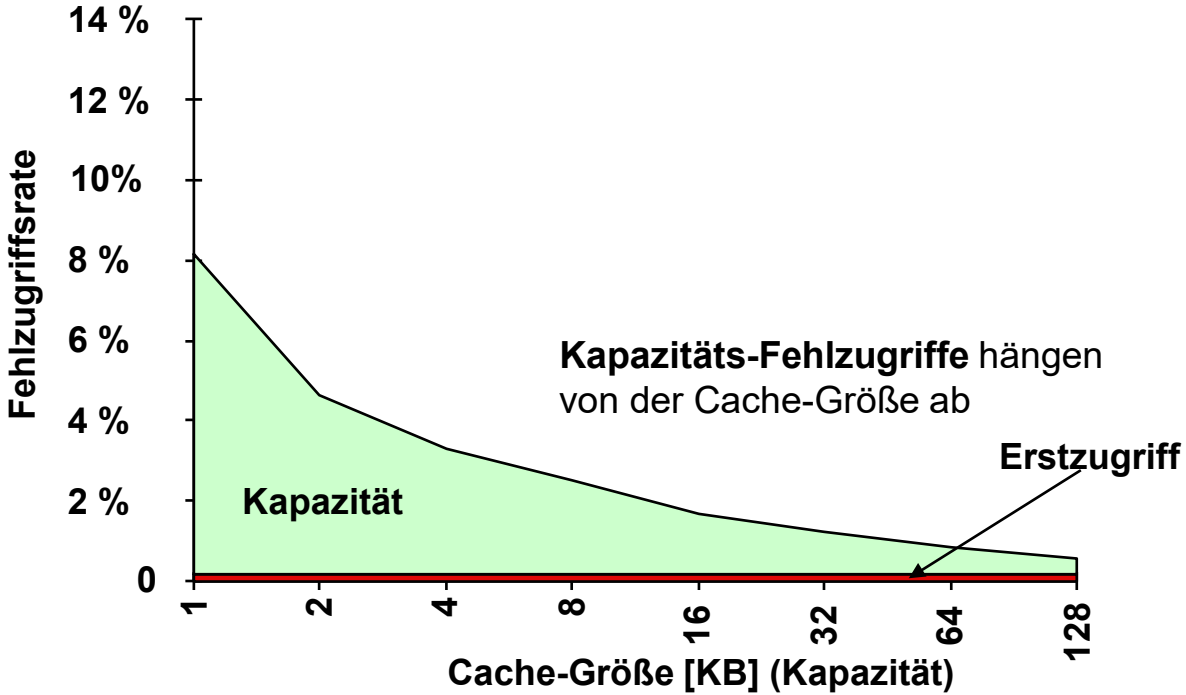
Grundlegende Fragen beim Entwurf

- **Ursachen für Fehlzugriffe**
  - **Erstzugriff-Fehlzugriff**



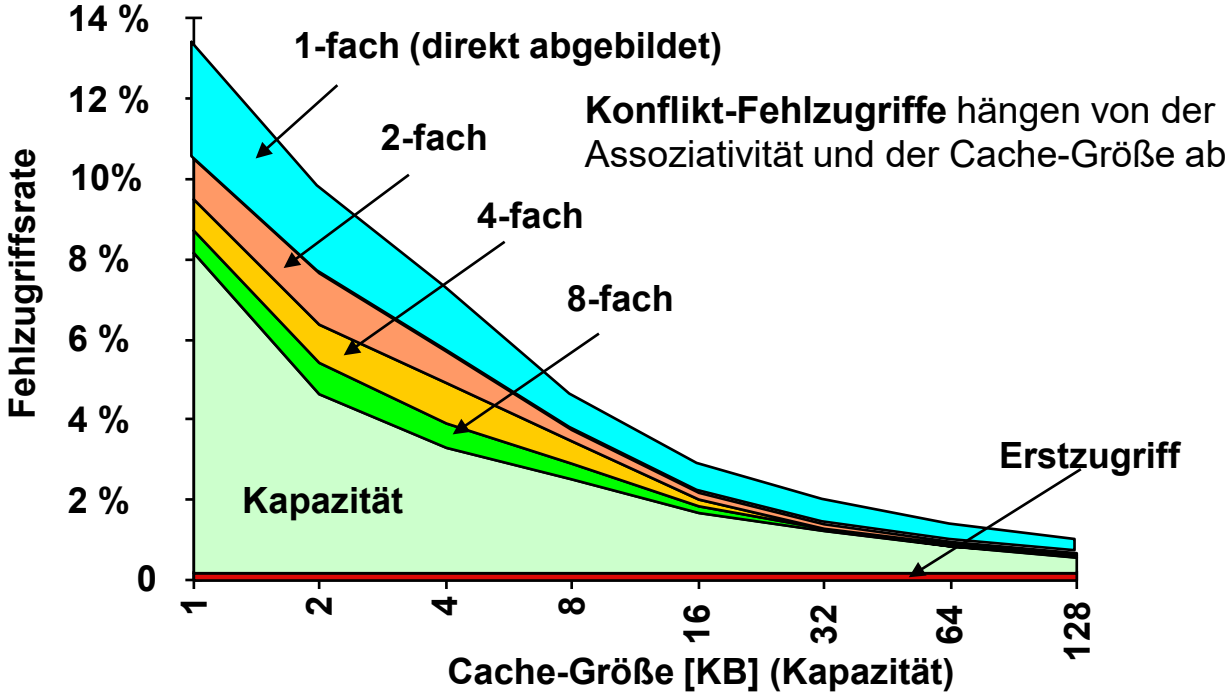
Grundlegende Fragen beim Entwurf

- **Ursachen für Fehlzugriffe**
  - **Kapazitäts-Fehlzugriff**



Grundlegende Fragen beim Entwurf

- **Ursachen für Fehlzugriffe**
  - **Konflikt-Fehlzugriff**

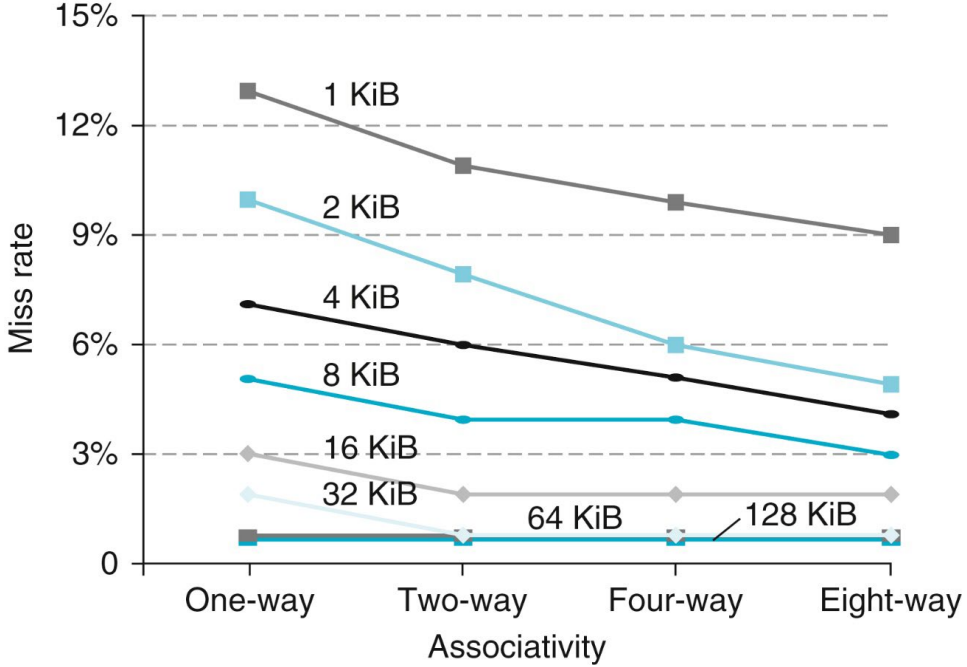


Grundlegende Fragen beim Entwurf

- **Ursachen für Fehlzugriffe**
  - Hinweis zu dem Diagramm auf Folie 13:
    - Daten wurden für die SPEC2000 Integer- und Gleitkomma-Benchmarks ermittelt;
    - Die Differenz in der Fehlzugriffsrate, die in dem direkt abgebildeten Cache im Vergleich zu einem vollassoziativen Cache derselben Größe entsteht, ist gegeben durch die Summer der vier Bereiche oberhalb der Speicherüberlastung (Kapazität).
      - Die Bereiche entsprechen den 8-fach, 4-fach, 2-fach und 1-fach Assoziativitäten.
      - Reduzierung der Assoziativität
        - **8-fach**: Fehlzugriffe im Vergleich von vollassoziativem (keine Konflikte) zu 8-fach satzassoziativem Cache
        - **4-fach**: Fehlzugriffe im Vergleich von 8-fach zu 4-fach satzassoziativem Cache
        - **2-fach**: Fehlzugriffe im Vergleich von 4-fach zu 2-fach satzassoziativen Cache
        - **1-fach**: Fehlzugriffe im Vergleich von 2-fach zu 1-fach satzassoziativen Cache (direkt abgebildeten Cache)

Grundlegende Fragen beim Entwurf

- **Entwurfsparameter: Assoziativität**
  - **Einfluss der Assoziativität auf die Fehlzugriffsrate**



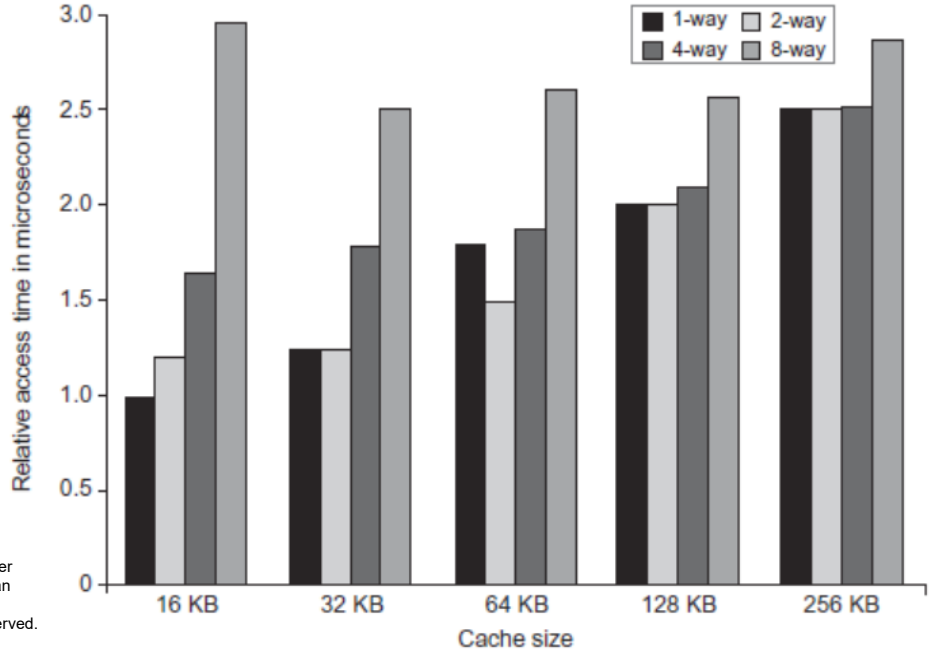
Quelle: J. Hennessy; D. Patterson: Computer Organization and Design RISC-V Edition. Copyright © 2021 Elsevier Inc. All rights reserved.

Grundlegende Fragen beim Entwurf

- **Entwurfsparameter: Assoziativität**
  - **Einfluss der Assoziativität auf die Fehlzugriffsrate**
    - Erhöhung der Assoziativität reduziert die Fehlzugriffsrate aufgrund von weniger Konflikten auf dieselbe Cache-Zeile;
      - Beispiel auf Folie 17 zeigt:
        - Die Fehlzugriffsrate sinkt für jeden der 8 Cachegrößen bei steigender Assoziativität.
        - Die Verbesserung ist signifikant zwischen direkt abgebildeten und 2-fach satz-assoziativer Cache-Organisation (20% – 30%)
        - Die Verbesserung ist weniger signifikant bei steigender Assoziativität.
        - Kleinere Caches profitieren mehr von höherer Assoziativität, da die grundlegende Fehlzugriffsrate von kleineren Caches höher ist.

Grundlegende Fragen beim Entwurf

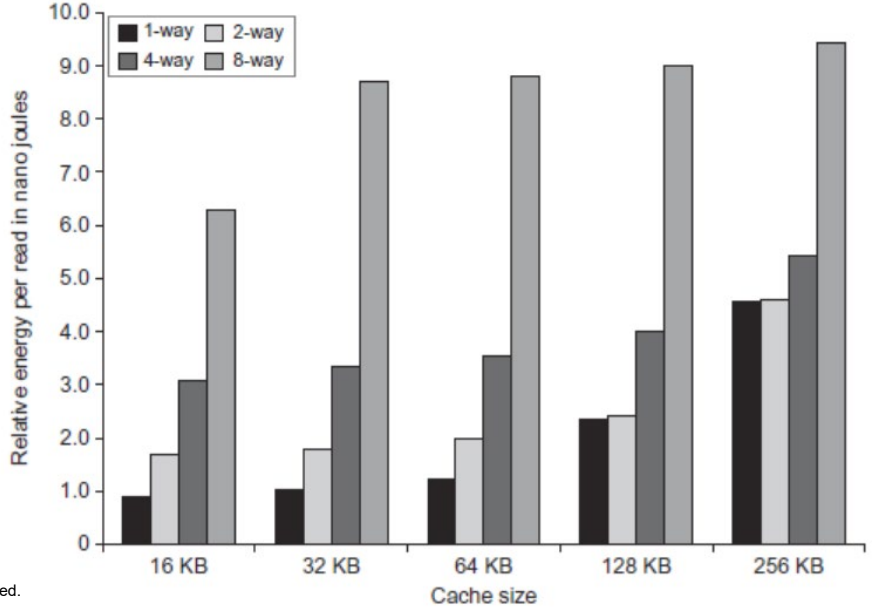
- **Entwurfsparameter: Assoziativität**
  - **Einfluss der Cache-Größe und Assoziativität auf die Zugriffszeit bei einem Treffer**



Steigende Assoziativität und größere Cache-Speicher sind jeweils mit langsameren Zugriffszeiten höherem Aufwand verbunden.

Grundlegende Fragen beim Entwurf

- **Entwurfsparameter: Assoziativität**
  - **Einfluss der Cache-Größe und Assoziativität auf den Energieverbrauch bei einem Lesezugriff**

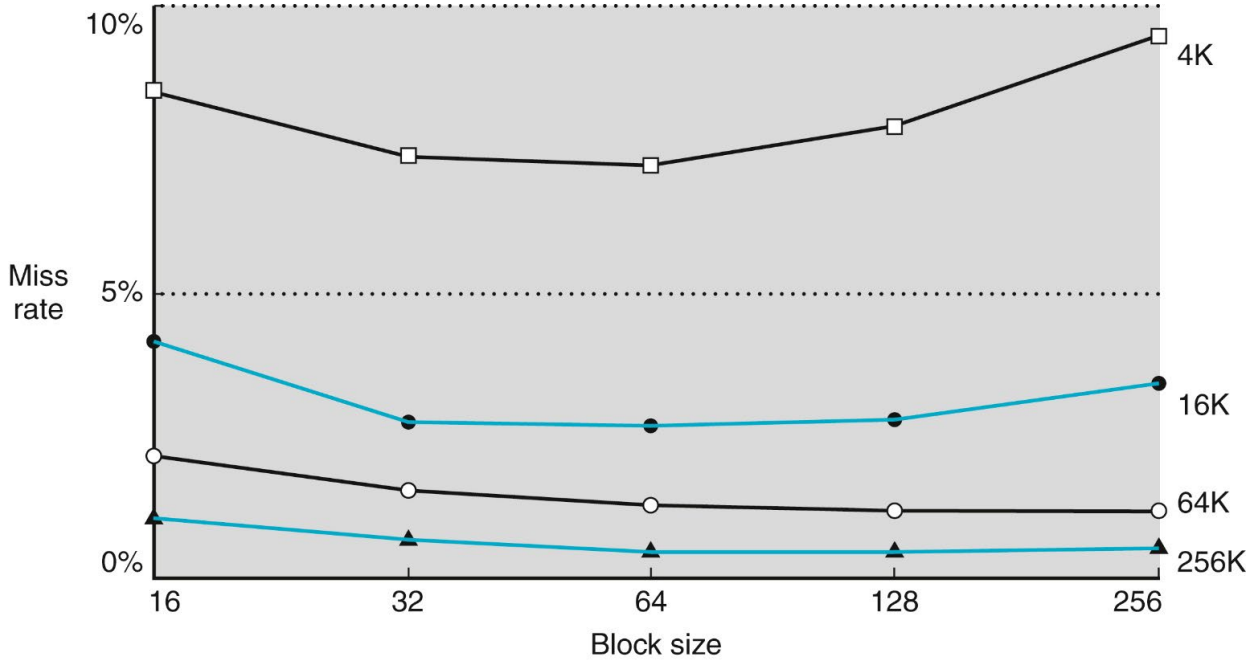


Der Energieverbrauch steigt mit einer höheren Assoziativität und größeren Cache-Speicher.

Quelle: Hennessy, J.; Patterson, D.: Computer Architecture A Quantitative Approach. Morgan Kaufman Publishers, 6th Edition, 2019,

Grundlegende Fragen beim Entwurf

- **Entwurfsparameter: Blockgröße**
  - **Einfluss der Blockgröße auf die Fehlzugriffsrate**



Quelle: J. Hennessy; D. Patterson: Computer Organization and Design RISC-V Edition. Copyright © 2021 Elsevier Inc. All rights reserved.

Grundlegende Fragen beim Entwurf

- **Entwurfsparameter: Blockgröße**
  - **Einfluss der Blockgröße auf die Fehlzugriffsrate**
    - Größere Blöcke nutzen die räumliche Lokalität aus, um die Fehlzugriffsraten zu senken
      - Fehlzugriffsrate sinkt bei steigender Blockgröße (im Beispiel bis zu einer Blockgröße von 64 Bytes;
      - Größere Blöcke reduzieren die Fehlzugriffe bei Erstzugriffen;

Grundlegende Fragen beim Entwurf

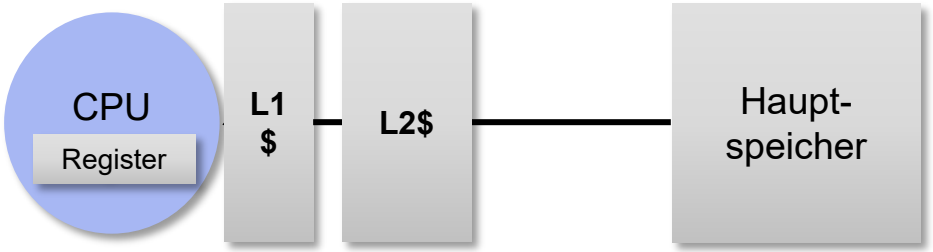
- **Entwurfsparameter: Blockgröße**
  - **Einfluss der Blockgröße auf die Fehlzugriffsrate**
    - Die Fehlzugriffsrate (unter der Annahme einer festen Cache-Größe) wieder, wenn die Blockgröße relativ groß in Bezug auf die Cache-Größe ist.
      - Weniger Blöcke können im Cache abgelegt werden.
      - Erhöhung der Kapazitäts- und Konflikt-Fehlzugriffe:
        - Ein Block kann aus dem Cache entfernt werden, bevor auf alle Wörter des Blockes zugegriffen worden ist.
        - Die räumliche Lokalität zwischen den Wörtern in einem Block ist großen Blöcken kleiner und der Vorteil ist geringer.
      - Die Kosten eines Fehlzugriffs steigen:
        - Mit der Blockgröße steigt die Übertragungszeit und damit der Fehlzugriffsaufwand.
        - Der Fehlzustandsaufwand umfasst die Zeit für das Laden des ersten Worts des Blocks und das Laden für den Rest des Blocks.

Grundlegende Fragen beim Entwurf

- **Entwurfsparameter: Kapazität des Cache-Speichers**
  - Größere Cache-Speicher reduzieren die Fehlzugriffsrate
  - **Aber:**
    - Die Zugriffszeit kann sich erhöhen!
      - Größerer Speicher = langsamerer Speicher (siehe Folie 17)
  - **Lösung:**
    - Einführung einer Cache-Hierarchie, wobei der L2$ eine größere Kapazität hat als der L1$.

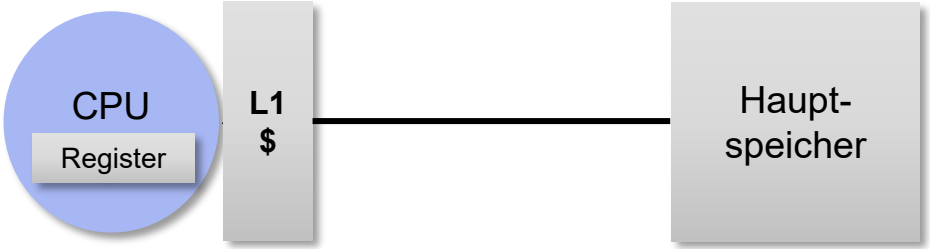
Grundlegende Fragen beim Entwurf

- **Entwurfsparameter: Kapazität des Cache-Speichers**
  - **Einführung einer Cache-Hierarchie**
    - **Wie wirkt sich ein L2$ auf die Rechenleistung aus?**



Grundlegende Fragen beim Entwurf

- **Entwurfsparameter: Kapazität des Cache-Speichers**
  - **Einführung einer Cache-Hierarchie**
    - **Wie wirkt sich ein L2$ auf die Rechenleistung aus?**



Base CPI: = 1,0 unter der Annahme, dass alle Zugriffe auf den L1$ Treffer sind;

Frequenz: 4 GHz bzw. Taktrate: 0,25 ns

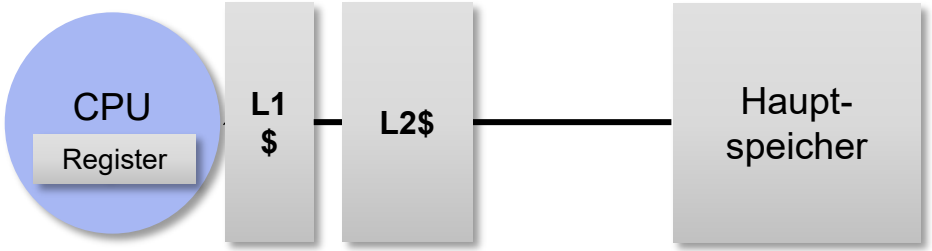
$\text{Fehlzugriffsrate}_{\text{L1\$}}$ pro Instruktion: 2%

Kosten Fehlzugriff (HS): 100 ns, bzw. 400 cc

- **Leistung:**
  - $\text{CPI} = \text{Base CPI} + \text{mittlere Zugriffszeit in Zyklen pro Instruktion}$
  - $\text{CPI} = \text{Base CPI} + \text{Fehlzugriffsrate}_{\text{L1\$}} * \text{Kosten Fehlzugriff}$
  - $CPI = 1{,}0 + 2\% * 400\ Zyklen = 9$

Grundlegende Fragen beim Entwurf

- **Entwurfsparameter: Kapazität des Cache-Speichers**
  - **Einführung einer Cache-Hierarchie**
    - **Wie wirkt sich ein L2$ auf die Rechenleistung aus?**



Annahmen:

Base CPI: = 1,0 unter der Annahme, dass alle Zugriffe auf den L1$ Treffer sind

Frequenz: 4 GHz bzw. Taktrate: 0,25 ns

$\text{Fehlzugriffsrate}_{L1\$}$ pro Instruktion: 2%

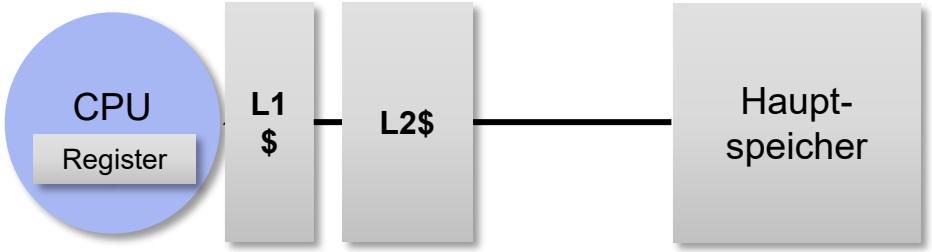
Kosten Fehlzugriff (HS): 100 ns, bzw. 400 cc

$\text{Fehlzugriffsrate}_{L2\$}$: 0,5%

Kosten Fehlzugriff (L2$): 5 ns, bzw. 20 cc

Grundlegende Fragen beim Entwurf

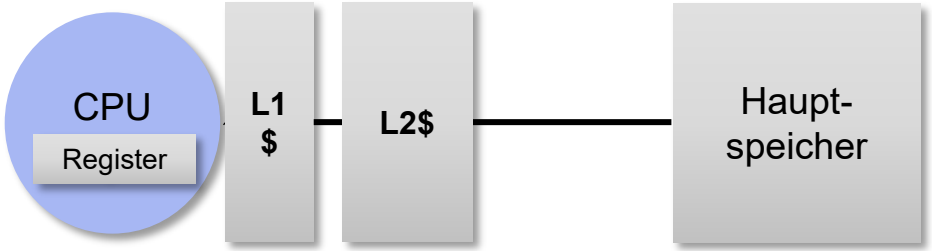
- **Entwurfsparameter: Kapazität des Cache-Speichers**
  - **Einführung einer Cache-Hierarchie**
    - **Wie wirkt sich ein L2$ auf die Rechenleistung aus?**



    - **Leistung:**
      - $\text{CPI} = \text{Base CPI} + \text{Fehlzugriffsrate}_{\text{L1\$}} * \text{Kosten Fehlzugriff (L2\$)} + \text{Fehlzugriffsrate}_{\text{L2\$}} * \text{Kosten Fehlzugriff (HS)}$
      - $\text{CPI} = 1{,}0 + 0{,}02 * 20 + 0{,}005 * 400 = 1 + 0{,}4 + 2 = 3{,}4$
      - **Prozessor mit L2$ ist um den Faktor $^{9}/_{3{,}4} = 2{,}6$ schneller.**

Grundlegende Fragen beim Entwurf

- **Entwurfsparameter: Kapazität des Cache-Speichers**
  - **Einführung einer Cache-Hierarchie**
    - **Wie wirkt sich ein L2$ auf die Rechenleistung aus?**

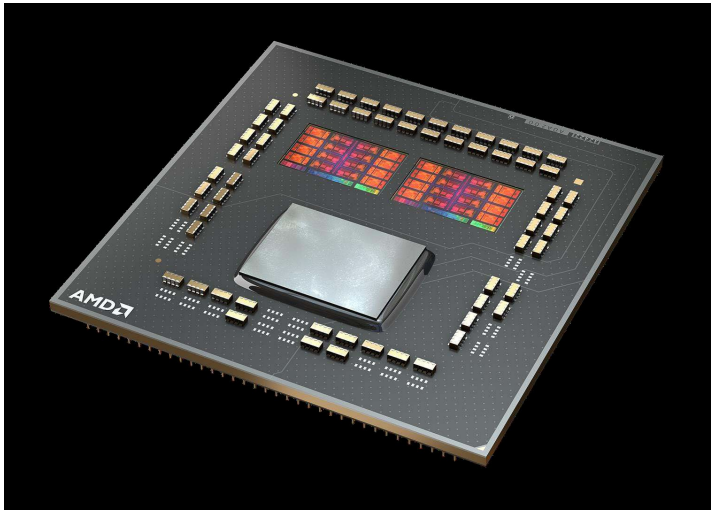


    - **Leistung:**
      - $\text{CPI} = \text{Base CPI} + \text{Fehlzugriffsrate}_{\text{L1\$}} * \text{Kosten Fehlzugriff (L2\$)} + \text{Fehlzugriffsrate}_{\text{L2\$}} * \text{Kosten Fehlzugriff (HS)}$
      - $\text{CPI} = 1{,}0 + 0{,}02 * 20 + 0{,}005 * 400 = 1 + 0{,}4 + 2 = 3{,}4$
      - **Prozessor mit L2$ ist um den Faktor $^{9}/_{3{,}4} = 2{,}6$ schneller.**

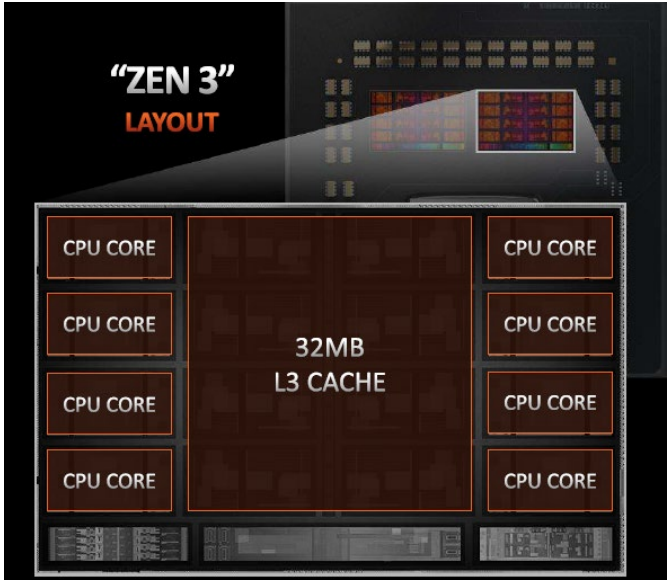
Grundlegende Fragen beim Entwurf

- **Ausblick: Fallstudie AMD ZEN3 (2021)**

Multi Chip Module



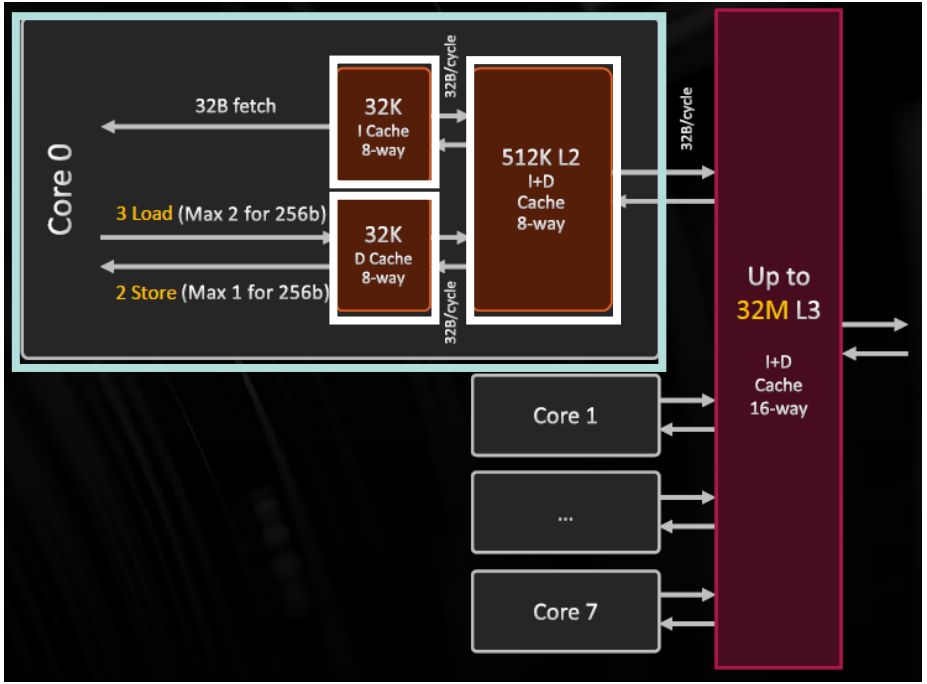
Prozessor Chip



Quelle: https://hc33.hotchips.org/assets/program/conference/day1/HC2021.C1.2%20AMD%20Mark%20Evers.pdf

Grundlegende Fragen beim Entwurf

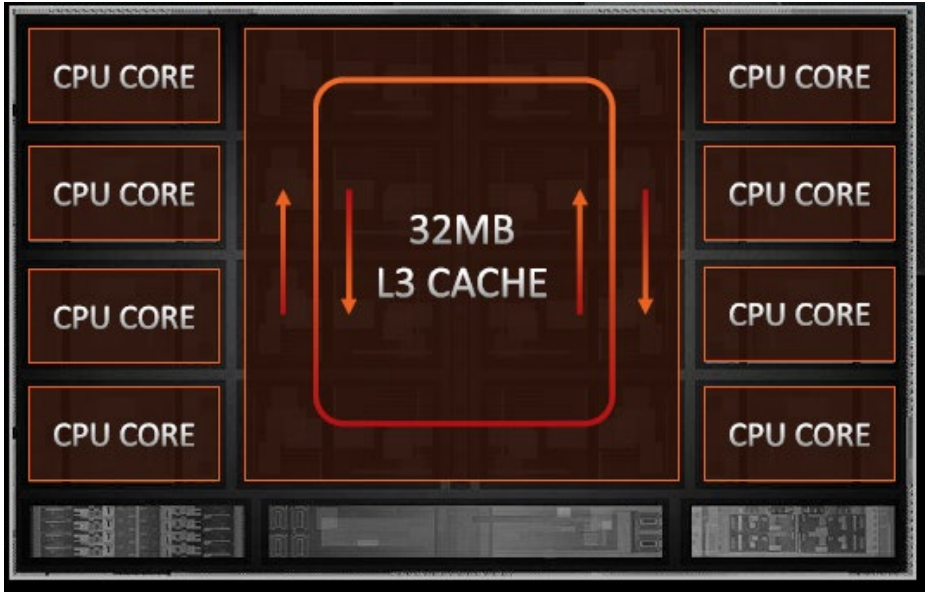
- **Ausblick: Fallstudie AMD ZEN3 (2021)**
  - **Cache-Hierarchie**
    - Pro Prozessorkern (Core)
      - Getrennte L1I$, L1D$
      - L2 Cache für Befehle und Daten



Quelle: https://hc33.hotchips.org/assets/program/conference/day1/HC2021.C1.2%20AMD%20Mark%20Evers.pdf

Grundlegende Fragen beim Entwurf

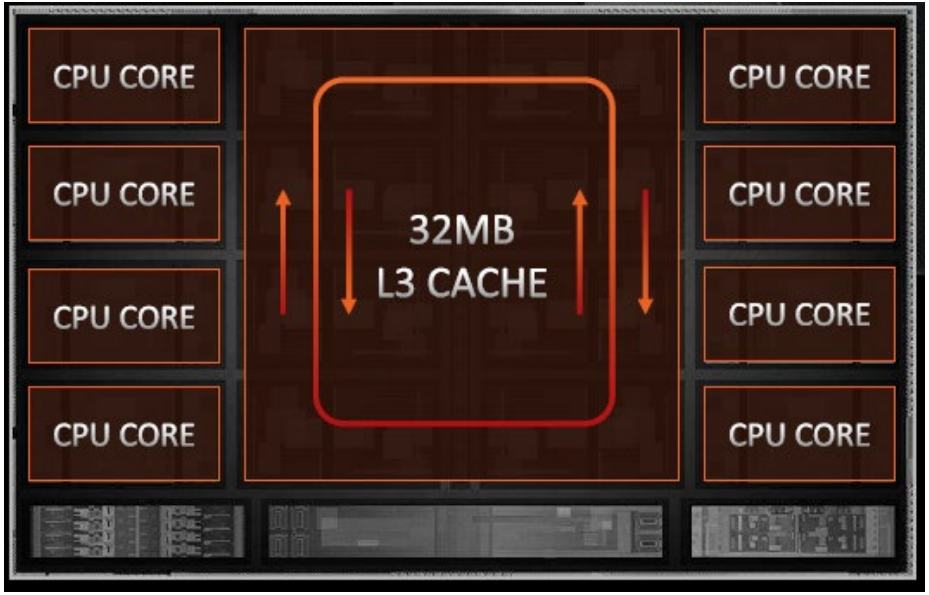
- **Ausblick: Fallstudie AMD ZEN3 (2021)**
  - **Cache-Hierarchie**
    - Pro Prozessorkern (Core)
      - Getrennte L1I$, L1D$
      - L2 Cache für Befehle und Daten
    - Gemeinsamer L3 Cache



Quelle: https://hc33.hotchips.org/assets/program/conference/day1/HC2021.C1.2%20AMD%20Mark%20Evers.pdf

Grundlegende Fragen beim Entwurf

- **Ausblick: Fallstudie AMD ZEN3 (2021)**
  - **Cache-Hierarchie**
    - Pro Prozessorkern (Core)
      - Getrennte L1I$, L1D$
      - L2 Cache für Befehle und Daten
    - Gemeinsamer L3 Cache



Quelle: https://hc33.hotchips.org/assets/program/conference/day1/HC2021.C1.2%20AMD%20Mark%20Evers.pdf

Grundlegende Fragen beim Entwurf

- **Herausforderungen beim Entwurf**
  - Betrachtung der drei Fehlzugriffsarten ist ein praktisches quantitatives Modell für den Entwurf
    - Weitere Fehlzugriffsart: Kohärenz-Fehlzugriffe bei Multiprozessoren
  - Beim Entwurf von realen Cache-Speicher spielen viele Entwurfsentscheidungen zusammen.
    - Änderung eines Cache-Parameters beeinflusst möglicherweise mehrere andere Komponenten der Fehlzugriffsraten
    - Mehrere alternative Implementierungstechniken (z. B. Aktualisierungsstrategien, Erstezungsstrategien)
    - Umfangreiche Simulationen

Grundlegende Fragen beim Entwurf

- **Herausforderungen im Betrieb**
  - **Software**
    - Schleifenoptimierungen
    - Optimierungsstrategien, die dazu beitragen, dass die Konfliktfehlzugriffsrate gesenkt wird
      - Beispiel: geschickte Ablage von Daten im Speicher