Karlsruher Institut für Technologie

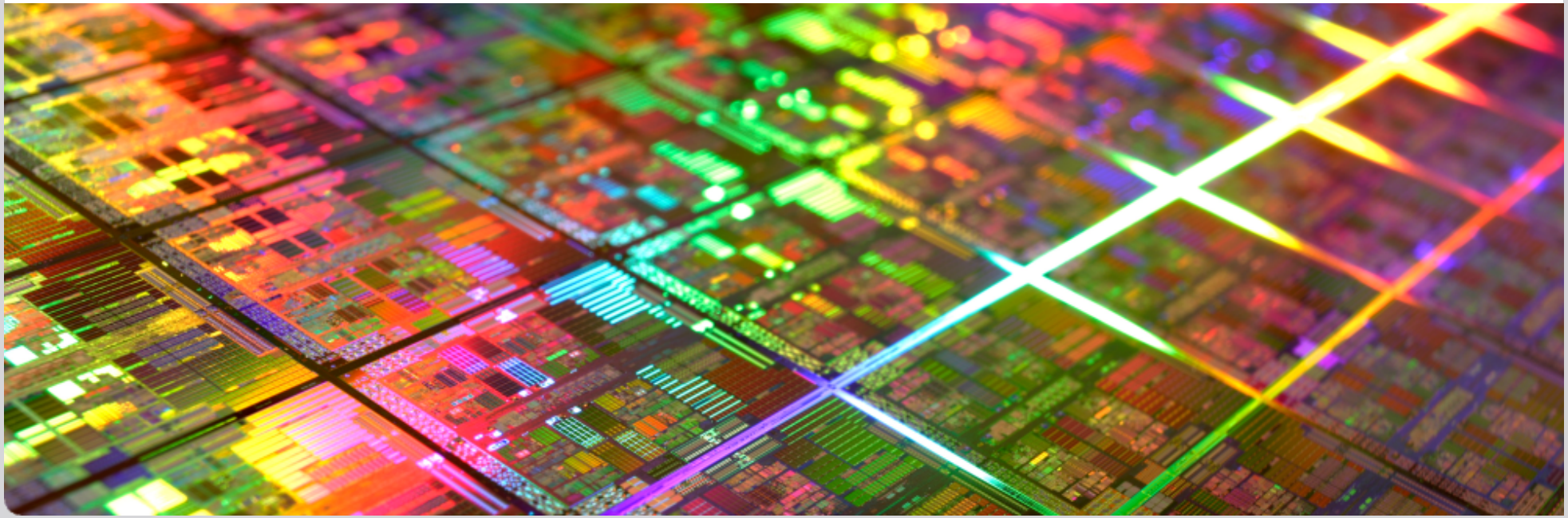
# Zentralübung Rechnerstrukturen im SS 2011

## Vektorrechner und Cache-Kohärenz

David Kramer, Prof. Dr. Wolfgang Karl

Lehrstuhl für Rechnerarchitektur und Parallelverarbeitung

12. Juli 2011



# Vektorrechner

Karlsruher Institut für Technologie

## Definition

- Unter einem Vektorprozessor (Vektorrechner) versteht man einen Rechner mit pipelineartig aufgebautem/n Rechenwerk/en zur Verarbeitung von Arrays von Gleitpunktzahlen.

## Merkmale

- Vektor = Array von Gleitpunktzahlen
- Vektoreinheit = Satz von Vektorpipelines in einem Rechenwerk
- Vektor- und Skalareinheit können parallel laufen
- Bei Vektoroperationen entfällt die sonst nötige Adressberechnungen
- Vektoroperationen können verkettet werden

# Vektorrechner

## Parallelverarbeitung in einem Vektorrechner

- Vektor-Pipeline-Parallelität
- Mehrere Vektor-Pipelines in einer Vektoreinheit
  - Verkettung
- Vervielfachung der Pipelines
- Mehrere Vektoreinheiten

# Vektorrechner

## Eigenschaften

- Vektor Stride
  - Stride-Capability notwendig, wenn Daten nicht konsekutiv im Speicher liegen (z.B. bei Standard Matrix-Matrix-Multiplikation).
- Vektor-Maskierungssteuerung / Vektor-Mask-Register
  - Jede Vektorinstruktion arbeitet nur auf den Vektorelementen, deren Einträge eine 1 haben.
  - Z.B. zur Auflösung von IF-Anweisungen

# Vektorrechner

## Aufgabe 1)

```
int i;
int a[n], b[n];
for (i = 0; i < n; i++)
{
    if (a[i] < b[i])
    {
        b[i] = i;
    }
}
```

a) Realisieren Sie diese Code-Fragment mittels Vektorbefehlen. Gehen Sie bei Ihren Überlegungen davon aus, dass ein Vektorregister je alle *n* Werte der Arrays *a* ober *b* aufnehmen kann.

# Aufgabe 1) (forts.)

## Lösung - Initialisierung

```
LV    V1, Ra          # int a[n] in V1 laden
LV    V2, Rb          # int b[n] in V2 laden
mov   R1, 1           # R1 mit 1 initialisieren
CVI   V3, R1          # Create Vektor Index
                      # 0, R1 * 1, R1 * 2, ...
                      # entspricht for (i=0;i<n,i++)
```

# Aufgabe 1) (forts.)

## Lösung - Berechnung

```
SLTV.D   V1, V2   # compare elements with 'Less Than'
                  # if true 1 in Vektor Mask Register
                  # else 0 in Vektor Mask Register
                  # if (a[i] < b[i])

SUBV.D   V2, V2   # alle Stellen mit 1 im VMR
                  # auf 0 setzen
ADDV.D   V2, V3   # und dann mit den Werten aus V3
                  # auffuellen
                  # B[i] = i;
```

# Aufgabe 1) (forts.)

## Lösung - Abschluss

```
CVM             # Clear Vektor Mask
                # Alle Eintraege im VMR auf 1 setzen

SV   Rb, V2     # V2 speichern
```

# Cache - Motivation

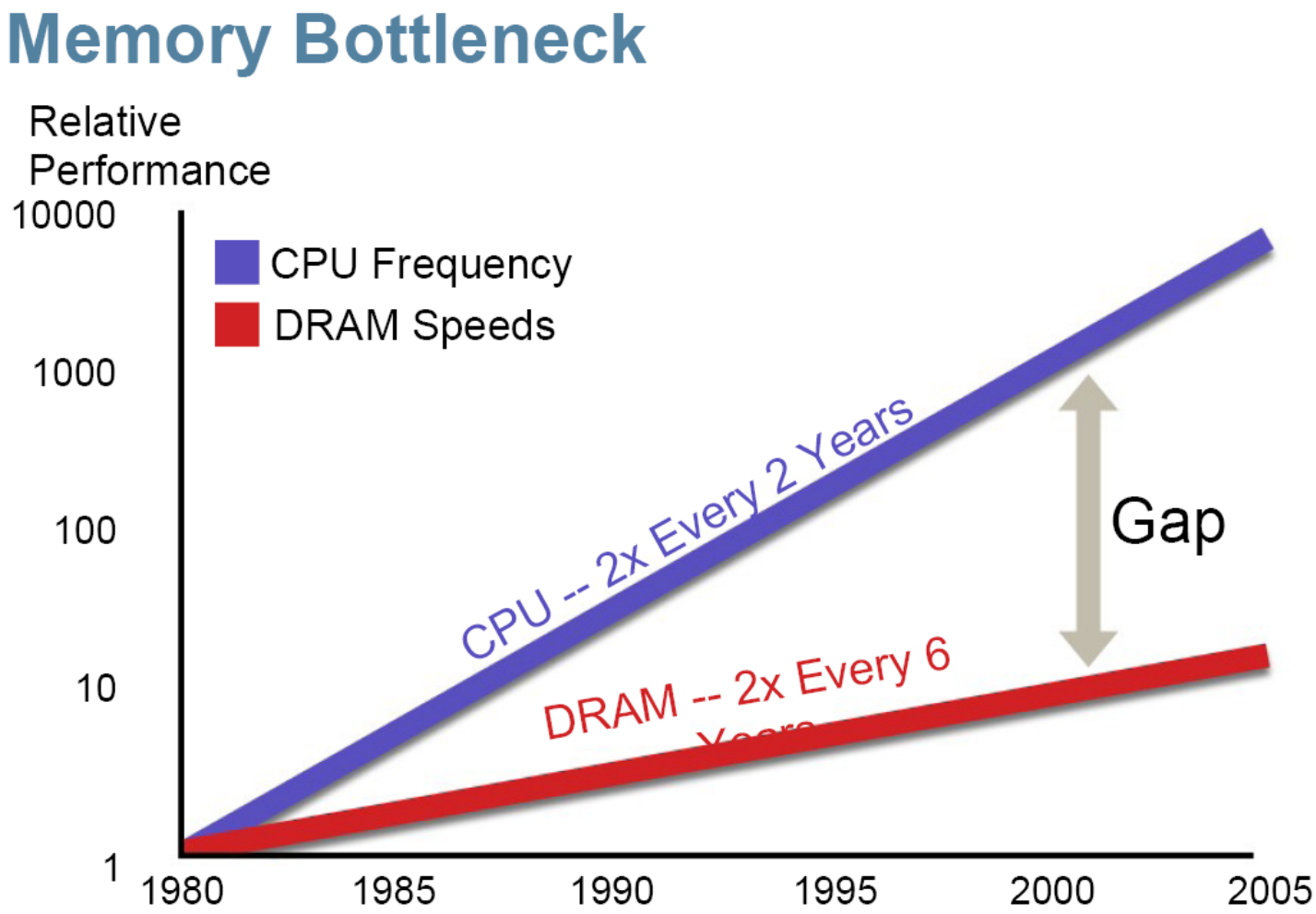



Abbildung: Quelle: http://blogs.sun.com/toddjobson/resource/

# Cache - Motivation (forts.)

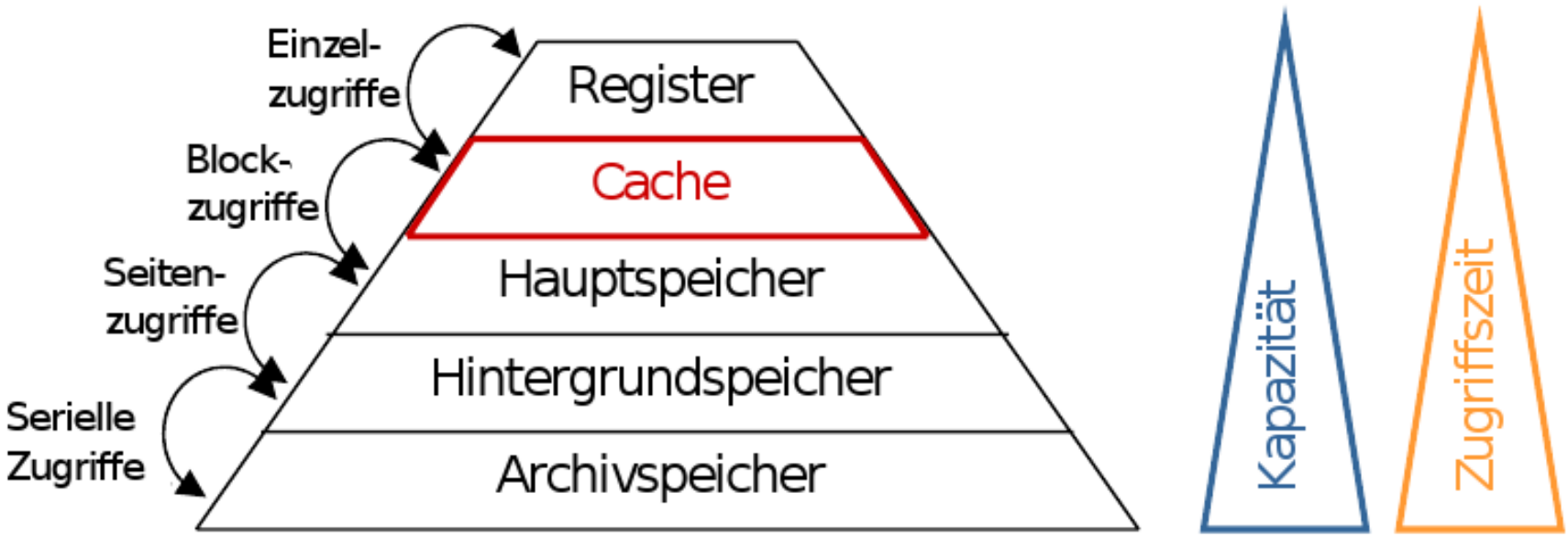



Abbildung: Speicherhierachie

- Ausnutzung der räumlichen und zeitlichen Lokalität von Anwendungen
- 90 / 10 - Regel

# Cache - Aufbau

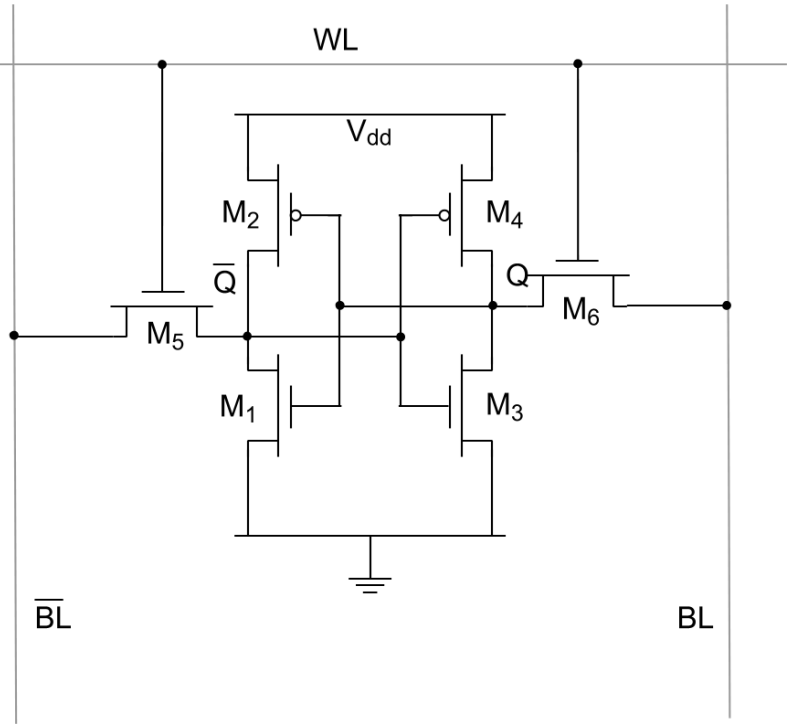



Abbildung: 6T-SRAM-Zelle

- Größe: über 100 $F^2$
- Sehr schnell

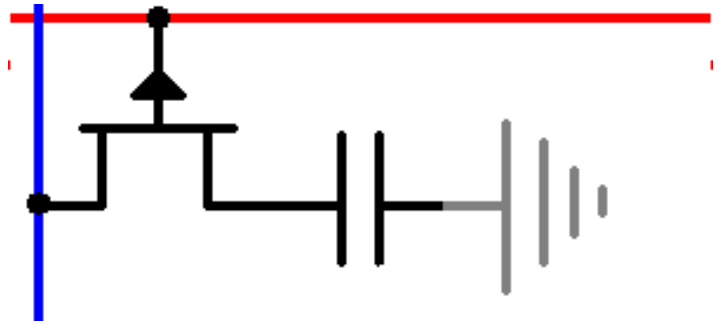


Abbildung: DRAM-Zelle

Quellen: Wikipedia

- Größe: ca. 6 – 10 $F^2$
- Destructive Read
- Refreshzyklen notwendig

# Cache - Parameter

Karlsruher Institut für Technologie

## Organisation

- Direkt abgebildet (direct mapped)
- Satzassoziativ (set associative)
- Vollassoziativ (fully associative)

## Hierachie

- Mehrere Level (L1, L2, L3)
- im Multiprozessorfall: private vs. shared
- inclusive vs. exclusive

## Größe

Charakterisiert durch:

- Cachelinesize
- # Cachelines bzw. Assoziativität * # Sätzen

# Cache - Parameter (forts.)

Karlsruher Institut für Technologie

## Ersetzungstrategie

- least recently used (LRU)
- least frequently used (LFU)
- Round Robin
- Random

## Schreibstrategie

- write-through vs. write-back
- write-allocation vs. no write-allocation

## Cachekohärenzprotokoll (im Multiprozessorfall)

- Busbasiert
- Verzeichnisbasiert

# Aufgabe 2.1: Cacheleistung

Karlsruher Institut für Technologie

## Cache-Performance

- Hit-Rate $r_H$
- Miss-Rate $r_M = 1 - r_H$
- Zugriffszeit bei Hit $t_H$
- Zugriffszeit bei Miss $t_M$

## Mittlere Zugriffszeit

$$t_a = \underbrace{r_H * t_H}_{Hit} + \underbrace{r_M * t_{Mem}}_{Miss}$$

## Mehrstufige Cache-Hierachie: Konkretisierung des Miss-Zeig

$$t_a = \underbrace{r_{H1} * t_{L1}}_{Hit\ L1} + \underbrace{r_{M1} * (\underbrace{r_{H2} * t_{L2}}_{Hit\ L2} + \underbrace{r_{M2} * t_{Mem}}_{Miss\ L2})}_{Miss\ L1}$$

# Aufgabe 2.1: Cacheleistung

Bei dem Entwurf eines Systems stehen zwei Entwurfsalternativen zur Auswahl. In beiden Entwurfsalternativen kommt eine zwei-stufige Cache-Hierachie zum Einsatz. Entwurfsalternative A hat ein kleinen L1-Cache mit einer Zugriffszeit von $t_{A-L1} = 10\,ns$, sowie einen L2-Cache mit einer Zugriffszeit von $t_{A-L2} = 30\,ns$. In Entwurfsalternative B ein größerer L1-Cache mit einer Zugriffszeit von $t_{B-L1} = 12\,ns$ zum Einsatz, sowie ein L2-Cache mit einer Zugriffszeit von $t_{B-L2} = 25\,ns$. Die Zugriffszeit des Hauptspeichers sei in beiden Entwurfsalternativen gleich und betrage $t_{Mem} = 100\,ns$.

a) Geben Sie eine Formel zur Berechung $t_a$ der mittleren Zugriffszeit in einer zwei stufigen Cache-Hierachie an.

Antwort:

$$t_a = \underbrace{r_{H1} * t_{L1}}_{\text{Hit L1}} + \underbrace{r_{M1} * (\underbrace{r_{H2} * t_{L2}}_{\text{Hit L2}} + \underbrace{r_{M2} * t_{Mem}}_{\text{Miss L2}})}_{\text{Miss L1}}$$

# Aufgabe 2.1: Cacheleistung

Bei dem Entwurf eines Systems stehen zwei Entwurfsalternativen zur Auswahl. In beiden Entwurfsalternativen kommt eine zwei-stufige Cache-Hierachie zum Einsatz. Entwurfsalternative A hat ein kleinen L1-Cache mit einer Zugriffszeit von $t_{A-L1} = 10\,ns$, sowie einen L2-Cache mit einer Zugriffszeit von $t_{A-L2} = 30\,ns$. In Entwurfsalternative B ein größerer L1-Cache mit einer Zugriffszeit von $t_{B-L1} = 12\,ns$ zum Einsatz, sowie ein L2-Cache mit einer Zugriffszeit von $t_{B-L2} = 25\,ns$. Die Zugriffszeit des Hauptspeichers sei in beiden Entwurfsalternativen gleich und betrage $t_{Mem} = 100\,ns$.

a) Geben Sie eine Formel zur Berechung $t_a$ der mittleren Zugriffszeit in einer zwei stufigen Cache-Hierachie an.

Antwort: $t_a = \underbrace{r_{H1} * t_{L1}}_{Hit\ L1} + \underbrace{r_{M1} * (\underbrace{r_{H2} * t_{L2}}_{Hit\ L2} + \underbrace{r_{M2} * t_{Mem}}_{Miss\ L2})}_{Miss\ L1}$

# Aufgabe 2.1: Cacheleistung

Bei dem Entwurf eines Systems stehen zwei Entwurfsalternativen zur Auswahl. In beiden Entwurfsalternativen kommt eine zwei-stufige Cache-Hierachie zum Einsatz. Entwurfsalternative A hat ein kleinen L1-Cache mit einer Zugriffszeit von $t_{A-L1} = 10\,ns$, sowie einen L2-Cache mit einer Zugriffszeit von $t_{A-L2} = 30\,ns$. In Entwurfsalternative B ein größerer L1-Cache mit einer Zugriffszeit von $t_{B-L1} = 12\,ns$ zum Einsatz, sowie ein L2-Cache mit einer Zugriffszeit von $t_{B-L2} = 25\,ns$. Die Zugriffszeit des Hauptspeichers sei in beiden Entwurfsalternativen gleich und betrage $t_{Mem} = 100\,ns$.

a) Geben Sie eine Formel zur Berechung $t_a$ der mittleren Zugriffszeit in einer zwei stufigen Cache-Hierachie an.

Antwort:

$$t_a = \underbrace{r_{H1} * t_{L1}}_{Hit\ L1} + \underbrace{r_{M1} * (\underbrace{r_{H2} * t_{L2}}_{Hit\ L2} + \underbrace{r_{M2} * t_{Mem}}_{Miss\ L2})}_{Miss\ L1}$$

# Aufgabe 2.1: Cacheleistung

Karlsruher Institut für Technologie

$t_{A-L1} = 10\,ns$, $t_{A-L2} = 30\,ns$, $t_{B-L1} = 12\,ns$, $t_{B-L2} = 25\,ns$, $t_{Mem} = 100\,ns$.

Bei der Evaluation wurden folgende Hit-Raten gemessen:

- Alternative A: $r_{A-L1} = 70\,\%$, sowie $r_{A-L2} = 40\,\%$
- Alternative B: $r_{B-L1} = 75\,\%$, sowie $r_{B-L2} = 35\,\%$

## Alternative A

$$t_a = 10ns * 70\% + (1 - 70\%) * (30ns * 40\% + 100ns * (1 - 40\%))$$
$$t_a = 7ns + 0.3 * (12ns + 60ns) = 28.6ns$$

## Alternative B

$$t_a = 12ns * 75\% + (1 - 75\%) * (25ns * 35\% + 100ns * (1 - 35\%))$$
$$t_a = 9ns + 0.25 * (8.75ns + 65ns) = 27,4375ns$$

Die durchschnittliche Zugriffszeit in **Entwurfsalternative B** ist geringer, somit ist diese Entwurfsalternative zu wählen.

# Aufgabe 2.1: Cacheleistung

Karlsruher Institut für Technologie

$t_{A-L1} = 10\,ns$, $t_{A-L2} = 30\,ns$, $t_{B-L1} = 12\,ns$, $t_{B-L2} = 25\,ns$, $t_{Mem} = 100\,ns$.

Bei der Evaluation wurden folgende Hit-Raten gemessen:

- Alternative A: $r_{A-L1} = 70\,\%$, sowie $r_{A-L2} = 40\,\%$
- Alternative B: $r_{B-L1} = 75\,\%$, sowie $r_{B-L2} = 35\,\%$

## Alternative A

$$t_a = 10ns * 70\% + (1 - 70\%) * (30ns * 40\% + 100ns * (1 - 40\%))$$
$$t_a = 7ns + 0.3 * (12ns + 60ns) = 28.6ns$$

## Alternative B

$$t_a = 12ns * 75\% + (1 - 75\%) * (25ns * 35\% + 100ns * (1 - 35\%))$$
$$t_a = 9ns + 0.25 * (8.75ns + 65ns) = 27,4375ns$$

Die durchschnittliche Zugriffszeit in **Entwurfsalternative B** ist geringer, somit ist diese Entwurfsalternative zu wählen.

# Aufgabe 2.1: Cacheleistung

Karlsruher Institut für Technologie

$t_{A-L1} = 10\,ns$, $t_{A-L2} = 30\,ns$, $t_{B-L1} = 12\,ns$, $t_{B-L2} = 25\,ns$, $t_{Mem} = 100\,ns$.

Bei der Evaluation wurden folgende Hit-Raten gemessen:

- Alternative A: $r_{A-L1} = 70\,\%$, sowie $r_{A-L2} = 40\,\%$
- Alternative B: $r_{B-L1} = 75\,\%$, sowie $r_{B-L2} = 35\,\%$

## Alternative A

$$t_a = 10ns * 70\% + (1 - 70\%) * (30ns * 40\% + 100ns * (1 - 40\%))$$
$$t_a = 7ns + 0.3 * (12ns + 60ns) = 28.6ns$$

## Alternative B

$$t_a = 12ns * 75\% + (1 - 75\%) * (25ns * 35\% + 100ns * (1 - 35\%))$$
$$t_a = 9ns + 0.25 * (8.75ns + 65ns) = 27,4375ns$$

Die durchschnittliche Zugriffszeit in **Entwurfsalternative B** ist geringer, somit ist diese Entwurfsalternative zu wählen.

# Aufgabe 2.2: Cacheleistung

Bei dem Entwurf eines Systems stehen zwei Entwurfsalternativen zur Auswahl. In beiden Entwurfsalternativen kommt eine zwei-stufige Cache-Hierachie zum Einsatz.
Entwurfsalternative A hat einen größeren L1-Cache mit einer Zugriffszeit von $t_{A-L1} = 2,5\,ns$, Entwurfsalternative B besitzt einen kleineren L1-Cache mit einer Zugriffszeit von $t_{B-L1} = 2\,ns$. Beide Entwurfsalternativen besitzen einen L2-Cache mit $t_{L2} = 10\,ns$. Die Zugriffszeit des Hauptspeichers sei in beiden Entwurfsalternativen gleich und betrage $t_{Mem} = 100\,ns$.

# Aufgabe 2.2: Cacheleistung

Karlsruher Institut für Technologie

Um die Leistung der Cache-Hierarchie zu steigern, wird bei Entwurfsalternative B alle Hierarchieebenen parallel angefragt.

Bei Entwurfsalternative A findet ein Zugriff auf die nächste Hierarchieebene erst statt, wenn die vorherige Ebene das angeforderte Datum nicht gespeichert hat.

a) Bei der Evaluation beider Entwurfsalternativen wurden folgende Hit-Raten gemessen:
   - Alternative A: $r_{A-L1} = 80\,\%$, sowie $r_{A-L2} = 90\,\%$
   - Alternative B: $r_{B-L1} = 70\,\%$, sowie $r_{B-L2} = 90\,\%$

   Für welche Entwurfsalternative würden Sie sich aus Gründen der Leistung entscheiden? Begründen Sie ihre Antwort.

# Aufgabe 2.2: Cacheleistung - Lösung

Karlsruher Institut für Technologie

$t_{A-L1} = 2.5\,ns$, $t_{B-L1} = 2\,ns$, $t_{L2} = 10\,ns$, $t_{Mem} = 100\,ns$.
$r_{A-L1} = 80\,\%$, $r_{A-L2} = 90\,\%$, $r_{B-L1} = 70\,\%$, $r_{B-L2} = 90\,\%$

## Alternative A - Sequentielle Anfrage

$t_A = 0.8 * 2.5\,ns + 0.2 * (0.9 * 12.5\,ns + 0.1 * 112.5\,ns)$
$t_A = 2\,ns + 0.2 * (11.25\,ns + 11.25\,ns) = 2\,ns + 0.2 * 22.5\,ns$
$t_A = 2\,ns + 4.5\,ns = 6.5\,ns$

## Alternative B - Parallele Anfrage

$t_B = 0.7 * 2\,ns + 0.3 * (0.9 * 10\,ns + 0.1 * 100\,ns)$
$t_B = 1.4\,ns + 0.3 * (9\,ns + 10\,ns) = 1.4\,ns + 0.3 * 19\,ns$
$t_B = 1.4\,ns + 5.7\,ns = 7.1\,ns$

$t_A < t_B$, Entwurfsalternative A ist schneller und daher zu wählen.

# Aufgabe 2.2: Cacheleistung - Lösung

Karlsruher Institut für Technologie

$t_{A-L1} = 2.5\,ns$, $t_{B-L1} = 2\,ns$, $t_{L2} = 10\,ns$, $t_{Mem} = 100\,ns$.
$r_{A-L1} = 80\,\%$, $r_{A-L2} = 90\,\%$, $r_{B-L1} = 70\,\%$, $r_{B-L2} = 90\,\%$

## Alternative A - Sequentielle Anfrage

$t_A = 0.8 * 2.5\,ns + 0.2 * (0.9 * 12.5\,ns + 0.1 * 112.5\,ns)$
$t_A = 2\,ns + 0.2 * (11.25\,ns + 11.25\,ns) = 2\,ns + 0.2 * 22.5\,ns$
$t_A = 2\,ns + 4.5\,ns = 6.5\,ns$

## Alternative B - Parallele Anfrage

$t_B = 0.7 * 2\,ns + 0.3 * (0.9 * 10\,ns + 0.1 * 100\,ns)$
$t_B = 1.4\,ns + 0.3 * (9\,ns + 10\,ns) = 1.4\,ns + 0.3 * 19\,ns$
$t_B = 1.4\,ns + 5.7\,ns = 7.1\,ns$

$t_A < t_B$, Entwurfsalternative A ist schneller und daher zu wählen.

# Aufgabe 2.2: Cacheleistung - Lösung

Karlsruher Institut für Technologie

$t_{A-L1} = 2.5\,ns$, $t_{B-L1} = 2\,ns$, $t_{L2} = 10\,ns$, $t_{Mem} = 100\,ns$.
$r_{A-L1} = 80\,\%$, $r_{A-L2} = 90\,\%$, $r_{B-L1} = 70\,\%$, $r_{B-L2} = 90\,\%$

## Alternative A - Sequentielle Anfrage

$t_A = 0.8 * 2.5\,ns + 0.2 * (0.9 * 12.5\,ns + 0.1 * 112.5\,ns)$
$t_A = 2\,ns + 0.2 * (11.25\,ns + 11.25\,ns) = 2\,ns + 0.2 * 22.5\,ns$
$t_A = 2\,ns + 4.5\,ns = 6.5\,ns$

## Alternative B - Parallele Anfrage

$t_B = 0.7 * 2\,ns + 0.3 * (0.9 * 10\,ns + 0.1 * 100\,ns)$
$t_B = 1.4\,ns + 0.3 * (9\,ns + 10\,ns) = 1.4\,ns + 0.3 * 19\,ns$
$t_B = 1.4\,ns + 5.7\,ns = 7.1\,ns$

$t_A < t_B$, Entwurfsalternative A ist schneller und daher zu wählen.

# Aufgabe 2.2: Cacheleistung - Lösung

Karlsruher Institut für Technologie

$t_{A-L1} = 2.5\,ns$, $t_{B-L1} = 2\,ns$, $t_{L2} = 10\,ns$, $t_{Mem} = 100\,ns$.
$r_{A-L1} = 80\,\%$, $r_{A-L2} = 90\,\%$, $r_{B-L1} = 70\,\%$, $r_{B-L2} = 90\,\%$

## Alternative A - Sequentielle Anfrage

$t_A = 0.8 * 2.5\,ns + 0.2 * (0.9 * 12.5\,ns + 0.1 * 112.5\,ns)$
$t_A = 2\,ns + 0.2 * (11.25\,ns + 11.25\,ns) = 2\,ns + 0.2 * 22.5\,ns$
$t_A = 2\,ns + 4.5\,ns = 6.5\,ns$

## Alternative B - Parallele Anfrage

$t_B = 0.7 * 2\,ns + 0.3 * (0.9 * 10\,ns + 0.1 * 100\,ns)$
$t_B = 1.4\,ns + 0.3 * (9\,ns + 10\,ns) = 1.4\,ns + 0.3 * 19\,ns$
$t_B = 1.4\,ns + 5.7\,ns = 7.1\,ns$

$t_A < t_B$, Entwurfsalternative A ist schneller und daher zu wählen.

# Aufgabe 2.3: Beweise

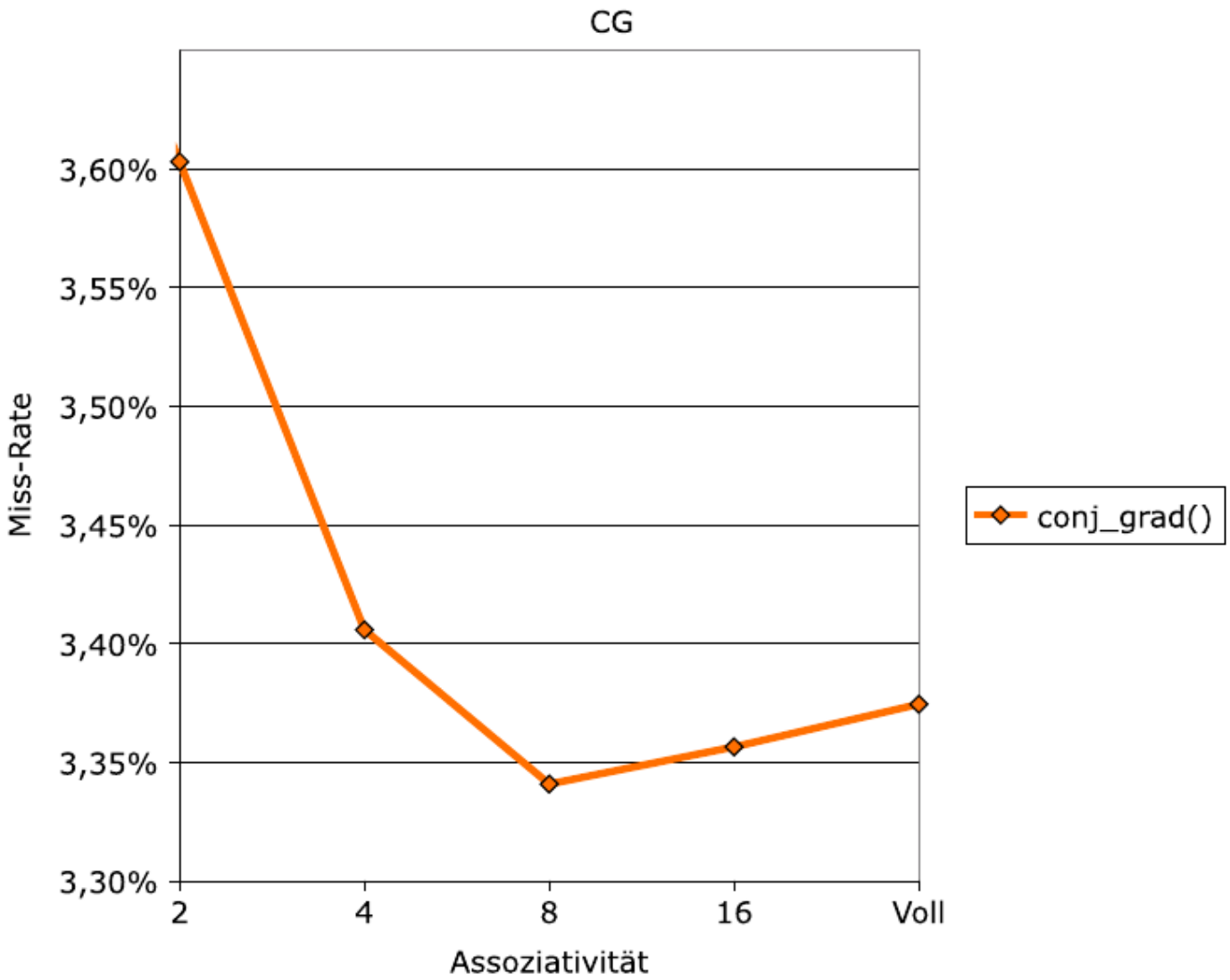
Karlsruher Institut für Technologie

Beweisen oder widerlegen Sie folgende Behauptungen:
Hinweis: Gehen Sie in ihren Überlegungen davon aus, dass die Caches gemäß Least-Recently-Used (LRU)-Strategie verdrängen.

a) Eine Erhöhung der Assoziativität eines Caches zieht immer eine Verringerung der Miss-Rate nach sich.

b) Vollassoziative Caches haben satzassoziativen Caches gegenüber immer eine niedrigere Miss-Rate.

# Aufgabe 2.3: Beweise - Motivation



# Aufgabe 2.3: Beweise

Karlsruher Institut für Technologie

Beweisen oder widerlegen Sie folgende Behauptungen:
**Hinweis:** Gehen Sie in ihren Überlegungen davon aus, dass die Caches gemäß Least-Recently-Used (LRU)-Strategie verdrängen.

a) Eine Erhöhung der Assoziativität eines Caches zieht immer eine Verringerung der Miss-Rate nach sich.

b) Vollassoziative Caches haben satzassoziativen Caches gegenüber immer eine niedrigere Miss-Rate.

**Widerlegung:**

Analog zu **Belady's Anomaly** [1]!

# Aufgabe 2.3: Beweise (forts.)

a) Eine Erhöhung der Assoziativität eines Caches zieht immer eine Verringerung der Miss-Rate nach sich.

2-fach assoziativ

4-fach assoziativ

Folge:

**Definition:** Speicherzugriffe A, B, E, F werden auf den ersten Satz abgebildet
C und D auf den zweiten Satz des 2-fach assoziativen Cache

# Aufgabe 2.3: Beweise (forts.)

a) Eine Erhöhung der Assoziativität eines Caches zieht immer eine Verringerung der Miss-Rate nach sich.

2-fach assoziativ

| | |
|---|---|
| A | **A** |
| B | **B** |

| |
|---|
| C |
| D |

4-fach assoziativ

| | |
|---|---|
| A | **A** |
| B | **B** |
| C | |
| D | |

Folge: *ABCDAB*

**Definition:** Speicherzugriffe A, B, E, F werden auf den ersten Satz abgebildet
C und D auf den zweiten Satz des 2-fach assoziativen Cache

# Aufgabe 2.3: Beweise (forts.)

a) Eine Erhöhung der Assoziativität eines Caches zieht immer eine Verringerung der Miss-Rate nach sich.

2-fach assoziativ

| | | |
|---|---|---|
| A | **A** | E |
| B | **B** | F |

| |
|---|
| C |
| D |

4-fach assoziativ

| | |
|---|---|
| A | **A** |
| B | **B** |
| C | E |
| D | F |

Folge: *ABCDABEF*

**Definition:** Speicherzugriffe A, B, E, F werden auf den ersten Satz abgebildet
C und D auf den zweiten Satz des 2-fach assoziativen Cache

# Aufgabe 2.3: Beweise (forts.)

a) Eine Erhöhung der Assoziativität eines Caches zieht immer eine Verringerung der Miss-Rate nach sich.

2-fach assoziativ

| | | |
|---|---|---|
| A | **A** | E |
| B | **B** | F |

| | |
|---|---|
| C | **C** |
| D | **D** |

4-fach assoziativ

| | | |
|---|---|---|
| A | **A** | C |
| B | **B** | D |
| C | E | |
| D | F | |

Folge: *ABCDABEFCD*

**Definition:** Speicherzugriffe A, B, E, F werden auf den ersten Satz abgebildet
C und D auf den zweiten Satz des 2-fach assoziativen Cache

# Aufgabe 2.3: Beweise (forts.)

a) Eine Erhöhung der Assoziativität eines Caches zieht immer eine Verringerung der Miss-Rate nach sich.

2-fach assoziativ

| | | | | |
|---|---|---|---|---|
| A | **A** | E | A | E |
| B | **B** | F | B | F |

| | |
|---|---|
| C | **C** |
| D | **D** |

4-fach assoziativ

| | | | |
|---|---|---|---|
| A | **A** | C | E |
| B | **B** | D | F |
| C | E | A | |
| D | F | B | |

Folge: *ABCDABEFCDABEF*

**Definition:** Speicherzugriffe A, B, E, F werden auf den ersten Satz abgebildet
C und D auf den zweiten Satz des 2-fach assoziativen Cache

# Aufgabe 2.3: Beweise (forts.)

a) Eine Erhöhung der Assoziativität eines Caches zieht immer eine Verringerung der Miss-Rate nach sich.

2-fach assoziativ

| | | | | |
|---|---|---|---|---|
| A | **A** | E | A | E |
| B | **B** | F | B | F |

| | | |
|---|---|---|
| C | **C** | **C** |
| D | **D** | **D** |

4-fach assoziativ

| | | | |
|---|---|---|---|
| A | **A** | C | E |
| B | **B** | D | F |
| C | E | A | C |
| D | F | B | D |

Folge: $ABCD(ABEFCD)^n$

**Definition:** Speicherzugriffe A, B, E, F werden auf den ersten Satz abgebildet
C und D auf den zweiten Satz des 2-fach assoziativen Cache

# Aufgabe 2.3: Beweise (forts.)

Karlsruher Institut für Technologie

Beweisen oder widerlegen Sie folgende Behauptungen:
**Hinweis:** Gehen Sie in ihren Überlegungen davon aus, dass die Caches gemäß Least-Recently-Used (LRU)-Strategie verdrängen.

a) Eine Erhöhung der Assoziativität eines Caches zieht immer eine Verringerung der Miss-Rate nach sich.

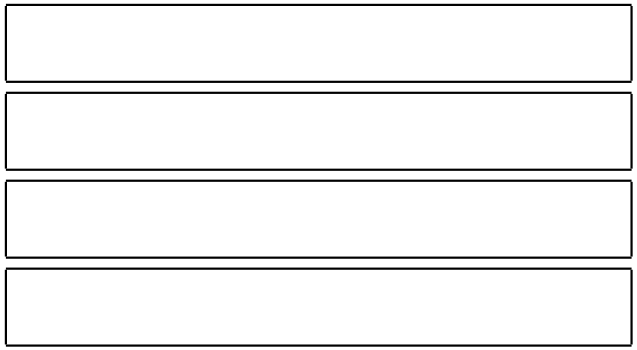
b) Vollassoziative Caches haben satzassoziativen Caches gegenüber immer eine niedrigere Miss-Rate.

# Aufgabe 2.3: Beweise (forts.)

Karlsruher Institut für Technologie

b) Vollassoziative Caches haben satzassoziativen Caches gegenüber immer eine niedrigere Missrate.

Direct Mapped



Vollassoziativ

Folge:

**Definition:** A wird auf die erste, B auf die zweite, C auf die dritte und D und E auf die vierte Cachezeile des DM-Cache abgebildet.

# Aufgabe 2.3: Beweise (forts.)

b) Vollassoziative Caches haben satzassoziativen Caches gegenüber immer eine niedrigere Missrate.

Direct Mapped

| |
|---|
| A |
| B |
| C |
| D |

Vollassoziativ

| |
|---|
| A |
| B |
| C |
| D |

Folge: *ABCD*

**Definition:** A wird auf die erste, B auf die zweite, C auf die dritte und D und E auf die vierte Cachezeile des DM-Cache abgebildet.

# Aufgabe 2.3: Beweise (forts.)

b) Vollassoziative Caches haben satzassoziativen Caches gegenüber immer eine niedrigere Missrate.

Direct Mapped

| |
|---|
| A |
| B |
| C |
| D E |

Vollassoziativ

| | |
|---|---|
| A | E |
| B | |
| C | |
| D | |

Folge: *ABCDE*

**Definition:** A wird auf die erste, B auf die zweite, C auf die dritte und D und E auf die vierte Cachezeile des DM-Cache abgebildet.

# Aufgabe 2.3: Beweise (forts.)

b) Vollassoziative Caches haben satzassoziativen Caches gegenüber immer eine niedrigere Missrate.

Direct Mapped

| | |
|---|---|
| A | **A** |
| B | **B** |
| C | **C** |
| D | E |

Vollassoziativ

| | |
|---|---|
| A | E |
| B | A |
| C | B |
| D | C |

Folge: *ABCDEABC*

**Definition:** A wird auf die erste, B auf die zweite, C auf die dritte und D und E auf die vierte Cachezeile des DM-Cache abgebildet.

# Aufgabe 2.3: Beweise (forts.)

b) Vollassoziative Caches haben satzassoziativen Caches gegenüber immer eine niedrigere Missrate.

Direct Mapped

| | |
|---|---|
| A | **A** |
| B | **B** |
| C | **C** |
| D E D E | |

Vollassoziativ

| | | |
|---|---|---|
| A | E | D |
| B | A | E |
| C | B | |
| D | C | |

Folge: $(ABCDE)^n$

**Definition:** A wird auf die erste, B auf die zweite, C auf die dritte und D und E auf die vierte Cachezeile des DM-Cache abgebildet.

# Aufgabe 2.3: Beweise (forts.)

b) Vollassoziative Caches haben satzassoziativen Caches gegenüber immer eine niedrigere Missrate.

Direct Mapped

| | | | |
|---|---|---|---|
| A | **A** | **A** | |
| B | **B** | **B** | |
| C | **C** | **C** | |
| D | E | D | E |

Vollassoziativ

| | | | |
|---|---|---|---|
| A | E | D | C |
| B | A | E | |
| C | B | A | |
| D | C | B | |

Folge: $(ABCDE)^n$

**Definition:** A wird auf die erste, B auf die zweite, C auf die dritte und D und E auf die vierte Cachezeile des DM-Cache abgebildet.

# Aufgabe 2.4 - Verständnisfragen

a) Welche Eigenschaft von Anwendungen werden von Caches ausgenutzt?

Antworten:

a) Räumliche und Zeitliche Lokalität

# Aufgabe 2.4 - Verständnisfragen

a) Welche Eigenschaft von Anwendungen werden von Caches ausgenutzt?

Antworten:

a) Räumliche und Zeitliche Lokalität

# Aufgabe 2.4 - Verständnisfragen (forts.)

Karlsruher Institut für Technologie

b) Welche Arten von Cache-Misses können unterschieden werden?

Antworten:

b) Conflict-, Capacity-, Compulsory-Misses

**Im Mehrprozessorfall:**

- Coherency-Miss
- Unterscheidung in: False-Sharing-Miss bzw. True-Sharing-Miss

# Aufgabe 2.4 - Verständnisfragen (forts.)

Karlsruher Institut für Technologie

b) Welche Arten von Cache-Misses können unterschieden werden?

Antworten:

b) Conflict-, Capacity-, Compulsory-Misses

**Im Mehrprozessorfall:**

- Coherency-Miss
- Unterscheidung in: False-Sharing-Miss bzw. True-Sharing-Miss

# Aufgabe 2.4 - Verständnisfragen (forts.)

c) Warum ist der Aufbau des Hauptspeichers aus SRAM-Zellen nicht sinnvoll?

Antworten:

c) SRAM-Zellen benötigen viel Chipfläche, deswegen ist der Aufbau des Hauptspeichers aus SRAM-Zellen entweder zu teuer oder man könnte nur geringe Kapazitäten anbieten.

# Aufgabe 2.4 - Verständnisfragen (forts.)

c) Warum ist der Aufbau des Hauptspeichers aus SRAM-Zellen nicht sinnvoll?

Antworten:

c) SRAM-Zellen benötigen viel Chipfläche, deswegen ist der Aufbau des Hauptspeichers aus SRAM-Zellen entweder zu teuer oder man könnte nur geringe Kapazitäten anbieten.

# Caches in Multiprozessorsystemen

Karlsruher Institut für Technologie

## Problem bei Verwendung von Caches in MP-Systemen

- Wahrung der Konsistenz
- CPUs operieren auf lokalen Kopien
- Daten in den Caches können sich von den Daten im Hauptspeicher unterscheiden

# Kohärenzprotokolle

## Bus-Snooping-basiert

- Write-Back Invalidate Protokoll
  - MSI [2]
  - MESI [2]
  - MOESI [3]
- Write-back Update Protokoll
  - Dragon [2]

## Verzeichnisbasiert

Wird in DSM-Systemen benötigt. (siehe [2], Kapitel 8)

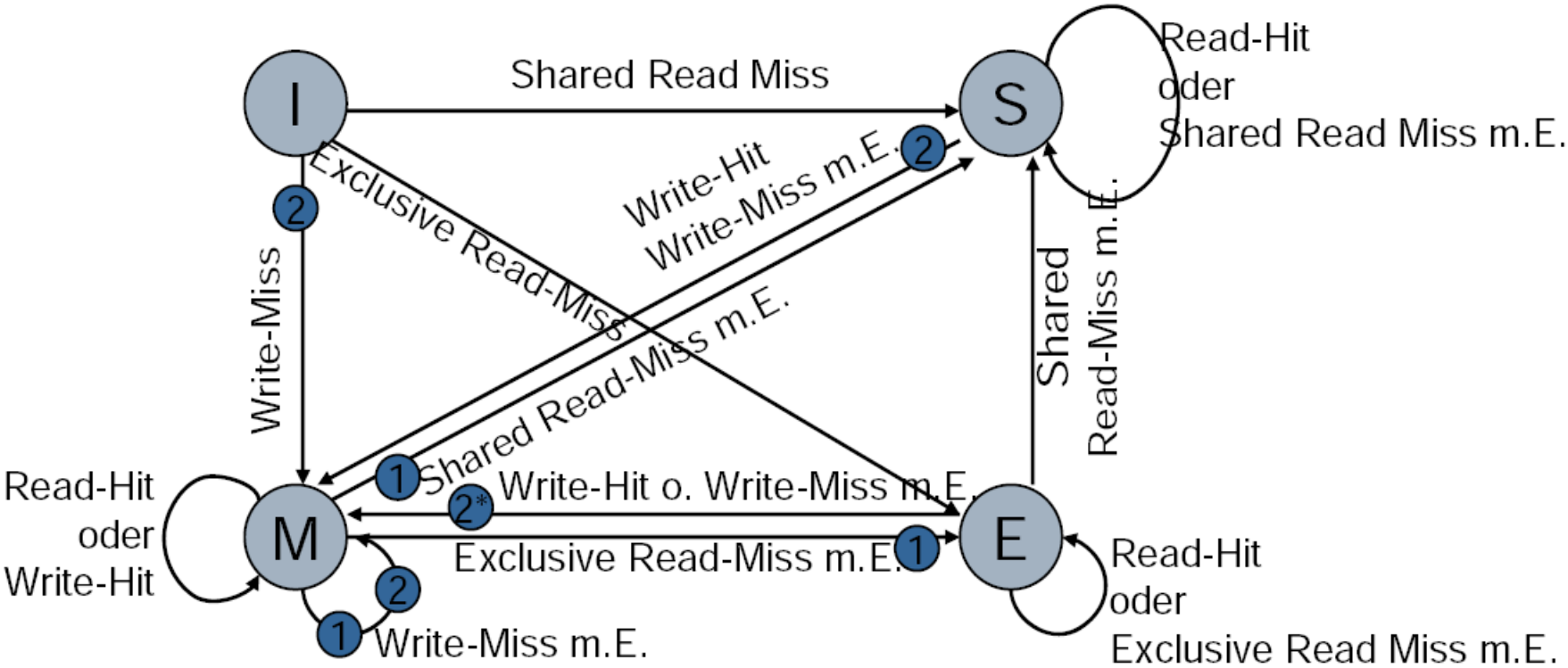
# Bus-Snooping

- Beobachtung des Speicherbusses hinsichtlich Speicherzugriffe anderer Bus-Master
- Bus-Snooping erfordert einen **globalen Speicherbus**
- Jeder Cache muss um sog. **Snoop-Logik** und **Steuersignale** zu anderen Caches erweitert werden.
  - Invalidate-Signal
  - Shared-Signal
  - Retry-Signal

# Das MESI-Protokoll

- Write-Back Invalidation Protokoll
- MESI: Zustandsautomat mit 4 Zuständen
  - M: Modified
  - E: Exclusive unmodified
  - S: Shared unmodified
  - I: Invalid
- Jede Cachezeile befindet sich in einem dieser Zustände
- d.h. jede Cachezeile wird um **2 Zustandsbits** erweitert
- Zustandsübergänge werden durch lokale und entfernte Speicherzugriffe ausgelöst

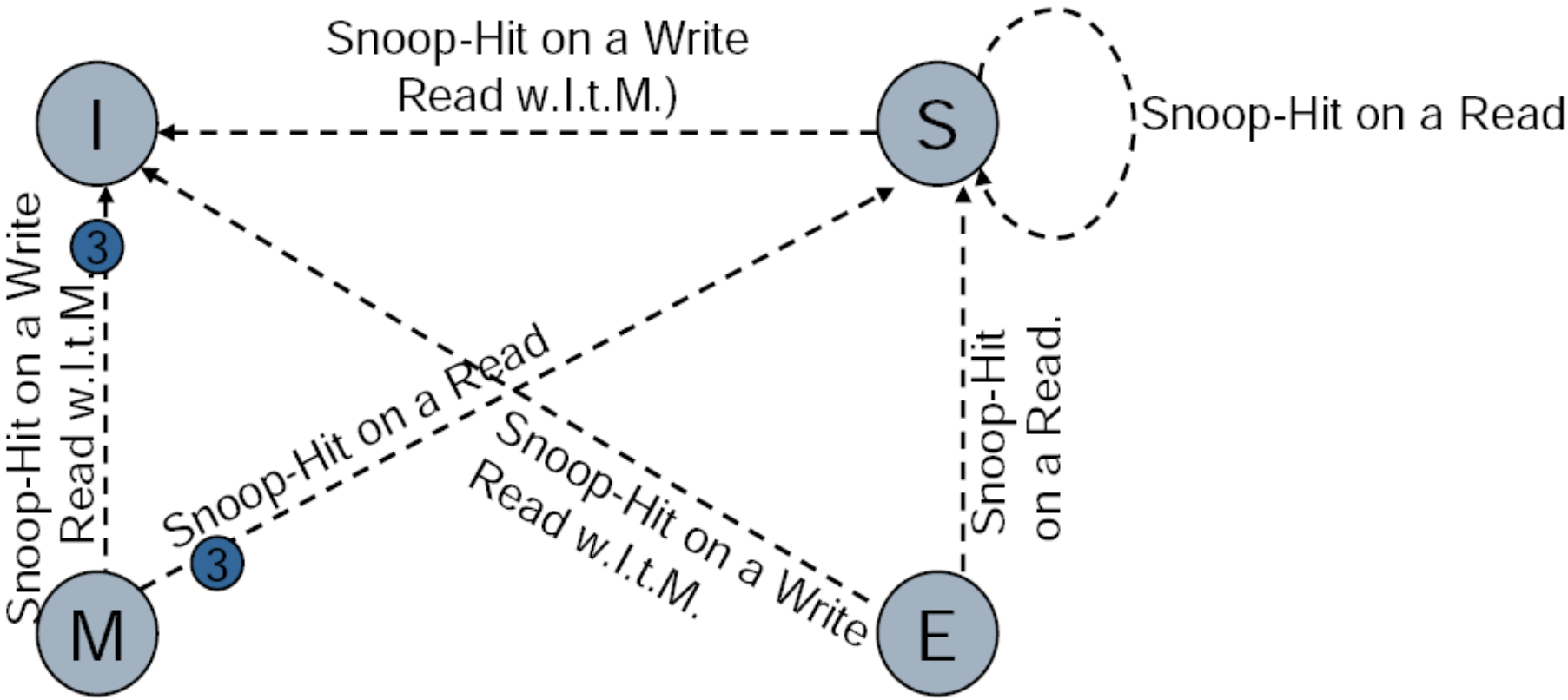
# MESI-Zustandsautomat (1)




1 Cache-Zeile wird in den Hauptspeicher zurückkopiert

2 Cache-Zeilen in den anderen Caches mit gleicher Blockadresse werden invalidiert

# MESI-Zustandsautomat (2) (aus [4])




3 Retry-Signal wird aktiviert und danach wird die Cache-Zeile in den Hauptspeicher kopiert

# Aufgabe 3.1: MESI Cachekohärenz-Protokoll

Karlsruher Institut für Technologie

Ein Dreiprozessorsystem sei speichergekoppelt. Die Caches haben je eine Größe von zwei Cachezeilen, welche je genau ein Speicherwort aufnehmen können. Die Füllung des Caches erfolgt von der niedrigsten Cachezeile aufwärts, sofern noch freie Zeilen zur Verfügung stehen, andernfalls wird gemäß LRU-Strategie verdrängt. Als Cache-Kohärenzprotokoll komme das MESI-Protokoll zum Einsatz.

- Vervollständigen Sie die auf gegebene Tabelle: Geben Sie jeweils Inhalt der Cache-Zeile und MESI-Zustand an.

# Aufgabe 3.1: MESI-Protokoll (forts.)

| Proz. | Aktion | Prozessor 1 | | Prozessor 2 | | Prozessor 3 | |
|---|---|---|---|---|---|---|---|
| | | Line 1 | Line 2 | Line 1 | Line 2 | Line 1 | Line 2 |
| | init | - | - | - | - | - | - |
| 1 | rd 6 | | | | | | |
| 2 | rd 2 | | | | | | |
| 1 | rd 4 | | | | | | |
| 3 | rd 4 | | | | | | |
| 2 | rd 3 | | | | | | |
| 3 | wr 7 | | | | | | |
| 1 | wr 4 | | | | | | |
| 2 | rd 7 | | | | | | |
| 3 | wr 5 | | | | | | |
| 1 | rd 3 | | | | | | |
| 3 | wr 3 | | | | | | |
| 2 | wr 7 | | | | | | |

# Aufgabe 3.1: MESI-Protokoll (forts.)

| Proz. | Aktion | Prozessor 1 | | Prozessor 2 | | Prozessor 3 | |
|---|---|---|---|---|---|---|---|
| | | Line 1 | Line 2 | Line 1 | Line 2 | Line 1 | Line 2 |
| | init | - | - | - | - | - | - |
| 1 | rd 6 | 6/E | | | | | |
| 2 | rd 2 | | | 2/E | | | |
| 1 | rd 4 | | 4/E | | | | |
| 3 | rd 4 | | | | | | |
| 2 | rd 3 | | | | | | |
| 3 | wr 7 | | | | | | |
| 1 | wr 4 | | | | | | |
| 2 | rd 7 | | | | | | |
| 3 | wr 5 | | | | | | |
| 1 | rd 3 | | | | | | |
| 3 | wr 3 | | | | | | |
| 2 | wr 7 | | | | | | |

# Aufgabe 3.1: MESI-Protokoll (forts.)

| Proz. | Aktion | Prozessor 1 | | Prozessor 2 | | Prozessor 3 | |
|---|---|---|---|---|---|---|---|
| | | Line 1 | Line 2 | Line 1 | Line 2 | Line 1 | Line 2 |
| | init | - | - | - | - | - | - |
| 1 | rd 6 | 6/E | | | | | |
| 2 | rd 2 | | | 2/E | | | |
| 1 | rd 4 | | 4/E | | | | |
| 3 | rd 4 | | 4/S | | | 4/S | |
| 2 | rd 3 | | | | | | |
| 3 | wr 7 | | | | | | |
| 1 | wr 4 | | | | | | |
| 2 | rd 7 | | | | | | |
| 3 | wr 5 | | | | | | |
| 1 | rd 3 | | | | | | |
| 3 | wr 3 | | | | | | |
| 2 | wr 7 | | | | | | |

# Aufgabe 3.1: MESI-Protokoll (forts.)

| Proz. | Aktion | Prozessor 1 | | Prozessor 2 | | Prozessor 3 | |
|---|---|---|---|---|---|---|---|
| | | Line 1 | Line 2 | Line 1 | Line 2 | Line 1 | Line 2 |
| | init | - | - | - | - | - | - |
| 1 | rd 6 | 6/E | | | | | |
| 2 | rd 2 | | | 2/E | | | |
| 1 | rd 4 | | 4/E | | | | |
| 3 | rd 4 | | 4/S | | | 4/S | |
| 2 | rd 3 | | | | 3/E | | |
| 3 | wr 7 | | | | | | 7/M |
| 1 | wr 4 | | | | | | |
| 2 | rd 7 | | | | | | |
| 3 | wr 5 | | | | | | |
| 1 | rd 3 | | | | | | |
| 3 | wr 3 | | | | | | |
| 2 | wr 7 | | | | | | |

# Aufgabe 3.1: MESI-Protokoll (forts.)

| Proz. | Aktion | Prozessor 1 | | Prozessor 2 | | Prozessor 3 | |
|---|---|---|---|---|---|---|---|
| | | Line 1 | Line 2 | Line 1 | Line 2 | Line 1 | Line 2 |
| | init | - | - | - | - | - | - |
| 1 | rd 6 | 6/E | | | | | |
| 2 | rd 2 | | | 2/E | | | |
| 1 | rd 4 | | 4/E | | | | |
| 3 | rd 4 | | 4/S | | | 4/S | |
| 2 | rd 3 | | | | 3/E | | |
| 3 | wr 7 | | | | | | 7/M |
| 1 | wr 4 | | 4/M | | | 4/I | |
| 2 | rd 7 | | | | | | |
| 3 | wr 5 | | | | | | |
| 1 | rd 3 | | | | | | |
| 3 | wr 3 | | | | | | |
| 2 | wr 7 | | | | | | |

# Aufgabe 3.1: MESI-Protokoll (forts.)

| Proz. | Aktion | Prozessor 1 | | Prozessor 2 | | Prozessor 3 | |
|---|---|---|---|---|---|---|---|
| | | Line 1 | Line 2 | Line 1 | Line 2 | Line 1 | Line 2 |
| | init | - | - | - | - | - | - |
| 1 | rd 6 | 6/E | | | | | |
| 2 | rd 2 | | | 2/E | | | |
| 1 | rd 4 | | 4/E | | | | |
| 3 | rd 4 | | 4/S | | | 4/S | |
| 2 | rd 3 | | | | 3/E | | |
| 3 | wr 7 | | | | | | 7/M |
| 1 | wr 4 | | 4/M | | | 4/I | |
| 2 | rd 7 | | | 7/S | | | 7/S |
| 3 | wr 5 | | | | | | |
| 1 | rd 3 | | | | | | |
| 3 | wr 3 | | | | | | |
| 2 | wr 7 | | | | | | |

# Aufgabe 3.1: MESI-Protokoll (forts.)

| Proz. | Aktion | Prozessor 1 | | Prozessor 2 | | Prozessor 3 | |
|---|---|---|---|---|---|---|---|
| | | Line 1 | Line 2 | Line 1 | Line 2 | Line 1 | Line 2 |
| | init | - | - | - | - | - | - |
| 1 | rd 6 | 6/E | | | | | |
| 2 | rd 2 | | | 2/E | | | |
| 1 | rd 4 | | 4/E | | | | |
| 3 | rd 4 | | 4/S | | | 4/S | |
| 2 | rd 3 | | | | 3/E | | |
| 3 | wr 7 | | | | | | 7/M |
| 1 | wr 4 | | 4/M | | | 4/I | |
| 2 | rd 7 | | | 7/S | | | 7/S |
| 3 | wr 5 | | | | | 5/M | |
| 1 | rd 3 | | | | | | |
| 3 | wr 3 | | | | | | |
| 2 | wr 7 | | | | | | |

# Aufgabe 3.1: MESI-Protokoll (forts.)

| Proz. | Aktion | Prozessor 1 | | Prozessor 2 | | Prozessor 3 | |
|---|---|---|---|---|---|---|---|
| | | Line 1 | Line 2 | Line 1 | Line 2 | Line 1 | Line 2 |
| | init | - | - | - | - | - | - |
| 1 | rd 6 | 6/E | | | | | |
| 2 | rd 2 | | | 2/E | | | |
| 1 | rd 4 | | 4/E | | | | |
| 3 | rd 4 | | 4/S | | | 4/S | |
| 2 | rd 3 | | | | 3/E | | |
| 3 | wr 7 | | | | | | 7/M |
| 1 | wr 4 | | 4/M | | | 4/I | |
| 2 | rd 7 | | | 7/S | | | 7/S |
| 3 | wr 5 | | | | | 5/M | |
| 1 | rd 3 | 3/S | | | 3/S | | |
| 3 | wr 3 | 3/I | | | 3/I | | 3/M |
| 2 | wr 7 | | | | | | |

# Aufgabe 3.1: MESI-Protokoll (forts.)

| Proz. | Aktion | Prozessor 1 | | Prozessor 2 | | Prozessor 3 | |
|---|---|---|---|---|---|---|---|
| | | Line 1 | Line 2 | Line 1 | Line 2 | Line 1 | Line 2 |
| | init | - | - | - | - | - | - |
| 1 | rd 6 | 6/E | | | | | |
| 2 | rd 2 | | | 2/E | | | |
| 1 | rd 4 | | 4/E | | | | |
| 3 | rd 4 | | 4/S | | | 4/S | |
| 2 | rd 3 | | | | 3/E | | |
| 3 | wr 7 | | | | | | 7/M |
| 1 | wr 4 | | 4/M | | | 4/I | |
| 2 | rd 7 | | | 7/S | | | 7/S |
| 3 | wr 5 | | | | | 5/M | |
| 1 | rd 3 | 3/S | | | 3/S | | |
| 3 | wr 3 | 3/I | | | 3/I | | 3/M |
| 2 | wr 7 | | | 7/M | | | |

# MOESI-Protokoll

- Erweiterung des MESI-Protokolls um einen weiteren Zustand
- Neuer Zustand O: Owned (shared modified)

## Vorteil

- Durch Cache-Cache-Transfers werden 2 Hauptspeicherzugriffe eingespart

## Nachteile

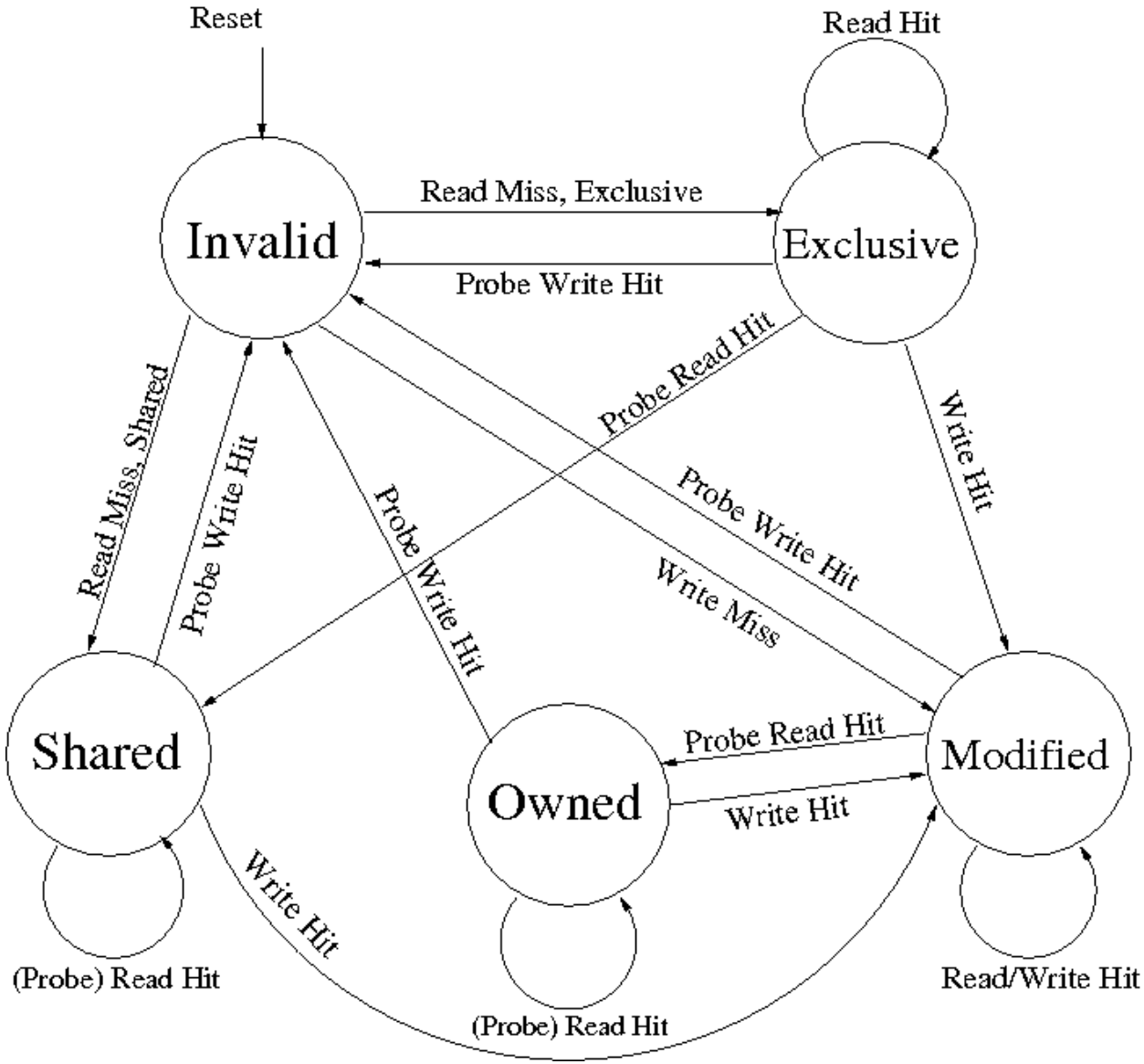
- Komplexere Logik notwenig
- 3 Zustands-Bits nötig

# MOESI Zustandsautomat

Wird ein in einem Cache bereits modifiziertes Datum von einem weiteren Cache gelesen, so wechselt der Cache mit dem modifizierten Datum von Zustand **M** in den Zustand **O**. Der lesende Cache übernimmt das Datum aus dem Cache des ersten Prozessors und lagert das Datum, gemäß MESI-Protokoll, als Shared **S** markiert ein. Werden die Daten in Caches mit dem Zustand **S** modifiziert, so wird das Datum in den anderen Caches invalidiert (Zustand **I**) und in den Zustand **M** gewechselt.

a) Erweitern Sie den aus der Vorlesung bekannte MESI-Zustandsautomat um die oben beschriebene Erweiterungen zum MOESI-Zustandsautomat.

# MOESI Zustandsautomat (nach [3])



# Aufgabe 3.2: MOESI-Protokoll

Karlsruher Institut für Technologie

Ein Dreiprozessorsystem sei speichergekoppelt. Die Caches haben je eine Größe von zwei Cachezeilen, welche je genau ein Speicherwort aufnehmen können. Die Füllung des Caches erfolgt von der niedrigsten Cachezeile aufwärts, sofern noch freie Zeilen zur Verfügung stehen, andernfalls wird gemäß LRU-Strategie verdrängt. Als Cache-Kohärenzprotokoll komme das MESI-Protokoll zum Einsatz. Der Cache sei initial leer.

b) Vervollständigen sie die gegebene Tabelle: Geben Sie jeweils Inhalt der Cache-Zeile und MOESI-Zustand an.

# Aufgabe 3.2: MOESI-Protokoll

| Proz. | Aktion | Prozessor 1 | | Prozessor 2 | | Prozessor 3 | |
|---|---|---|---|---|---|---|---|
| | | Line 1 | Line 2 | Line 1 | Line 2 | Line 1 | Line 2 |
| | init | - | - | - | - | - | - |
| 1 | rd 4 | | | | | | |
| 3 | rd 4 | | | | | | |
| 2 | rd 3 | | | | | | |
| 2 | wr 5 | | | | | | |
| 1 | rd 2 | | | | | | |
| 2 | wr 4 | | | | | | |
| 2 | rd 1 | | | | | | |
| 3 | rd 4 | | | | | | |
| 3 | rd 3 | | | | | | |
| 1 | wr 1 | | | | | | |
| 3 | rd 1 | | | | | | |

# Aufgabe 3.2: MOESI-Protokoll

Karlsruher Institut für Technologie

| Proz. | Aktion | Prozessor 1 | | Prozessor 2 | | Prozessor 3 | |
|---|---|---|---|---|---|---|---|
| | | Line 1 | Line 2 | Line 1 | Line 2 | Line 1 | Line 2 |
| | init | - | - | - | - | - | - |
| 1 | rd 4 | 4/E | | | | | |
| 3 | rd 4 | 4/S | | | | 4/S | |
| 2 | rd 3 | | | | | | |
| 2 | wr 5 | | | | | | |
| 1 | rd 2 | | | | | | |
| 2 | wr 4 | | | | | | |
| 2 | rd 1 | | | | | | |
| 3 | rd 4 | | | | | | |
| 3 | rd 3 | | | | | | |
| 1 | wr 1 | | | | | | |
| 3 | rd 1 | | | | | | |

# Aufgabe 3.2: MOESI-Protokoll

Karlsruher Institut für Technologie

| Proz. | Aktion | Prozessor 1 | | Prozessor 2 | | Prozessor 3 | |
|---|---|---|---|---|---|---|---|
| | | Line 1 | Line 2 | Line 1 | Line 2 | Line 1 | Line 2 |
| | init | - | - | - | - | - | - |
| 1 | rd 4 | 4/E | | | | | |
| 3 | rd 4 | 4/S | | | | 4/S | |
| 2 | rd 3 | | | 3/E | | | |
| 2 | wr 5 | | | | 5/M | | |
| 1 | rd 2 | | | | | | |
| 2 | wr 4 | | | | | | |
| 2 | rd 1 | | | | | | |
| 3 | rd 4 | | | | | | |
| 3 | rd 3 | | | | | | |
| 1 | wr 1 | | | | | | |
| 3 | rd 1 | | | | | | |

# Aufgabe 3.2: MOESI-Protokoll

| Proz. | Aktion | Prozessor 1 | | Prozessor 2 | | Prozessor 3 | |
|---|---|---|---|---|---|---|---|
| | | Line 1 | Line 2 | Line 1 | Line 2 | Line 1 | Line 2 |
| | init | - | - | - | - | - | - |
| 1 | rd 4 | 4/E | | | | | |
| 3 | rd 4 | 4/S | | | | 4/S | |
| 2 | rd 3 | | | 3/E | | | |
| 2 | wr 5 | | | | 5/M | | |
| 1 | rd 2 | | 2/E | | | | |
| 2 | wr 4 | | | | | | |
| 2 | rd 1 | | | | | | |
| 3 | rd 4 | | | | | | |
| 3 | rd 3 | | | | | | |
| 1 | wr 1 | | | | | | |
| 3 | rd 1 | | | | | | |

# Aufgabe 3.2: MOESI-Protokoll

| Proz. | Aktion | Prozessor 1 | | Prozessor 2 | | Prozessor 3 | |
|---|---|---|---|---|---|---|---|
| | | Line 1 | Line 2 | Line 1 | Line 2 | Line 1 | Line 2 |
| | init | - | - | - | - | - | - |
| 1 | rd 4 | 4/E | | | | | |
| 3 | rd 4 | 4/S | | | | 4/S | |
| 2 | rd 3 | | | 3/E | | | |
| 2 | wr 5 | | | | 5/M | | |
| 1 | rd 2 | | 2/E | | | | |
| 2 | wr 4 | 4/I | | 4/M | | 4/I | |
| 2 | rd 1 | | | | | | |
| 3 | rd 4 | | | | | | |
| 3 | rd 3 | | | | | | |
| 1 | wr 1 | | | | | | |
| 3 | rd 1 | | | | | | |

# Aufgabe 3.2: MOESI-Protokoll

| Proz. | Aktion | Prozessor 1 | | Prozessor 2 | | Prozessor 3 | |
|---|---|---|---|---|---|---|---|
| | | Line 1 | Line 2 | Line 1 | Line 2 | Line 1 | Line 2 |
| | init | - | - | - | - | - | - |
| 1 | rd 4 | 4/E | | | | | |
| 3 | rd 4 | 4/S | | | | 4/S | |
| 2 | rd 3 | | | 3/E | | | |
| 2 | wr 5 | | | | 5/M | | |
| 1 | rd 2 | | 2/E | | | | |
| 2 | wr 4 | 4/I | | 4/M | | 4/I | |
| 2 | rd 1 | | | | 1/E | | |
| 3 | rd 4 | | | | | | |
| 3 | rd 3 | | | | | | |
| 1 | wr 1 | | | | | | |
| 3 | rd 1 | | | | | | |

# Aufgabe 3.2: MOESI-Protokoll

| Proz. | Aktion | Prozessor 1 | | Prozessor 2 | | Prozessor 3 | |
|---|---|---|---|---|---|---|---|
| | | Line 1 | Line 2 | Line 1 | Line 2 | Line 1 | Line 2 |
| | init | - | - | - | - | - | - |
| 1 | rd 4 | 4/E | | | | | |
| 3 | rd 4 | 4/S | | | | 4/S | |
| 2 | rd 3 | | | 3/E | | | |
| 2 | wr 5 | | | | 5/M | | |
| 1 | rd 2 | | 2/E | | | | |
| 2 | wr 4 | 4/I | | 4/M | | 4/I | |
| 2 | rd 1 | | | | 1/E | | |
| 3 | rd 4 | | | 4/O | | 4/S | |
| 3 | rd 3 | | | | | | |
| 1 | wr 1 | | | | | | |
| 3 | rd 1 | | | | | | |

# Aufgabe 3.2: MOESI-Protokoll

| Proz. | Aktion | Prozessor 1 | | Prozessor 2 | | Prozessor 3 | |
|---|---|---|---|---|---|---|---|
| | | Line 1 | Line 2 | Line 1 | Line 2 | Line 1 | Line 2 |
| | init | - | - | - | - | - | - |
| 1 | rd 4 | 4/E | | | | | |
| 3 | rd 4 | 4/S | | | | 4/S | |
| 2 | rd 3 | | | 3/E | | | |
| 2 | wr 5 | | | | 5/M | | |
| 1 | rd 2 | | 2/E | | | | |
| 2 | wr 4 | 4/I | | 4/M | | 4/I | |
| 2 | rd 1 | | | | 1/E | | |
| 3 | rd 4 | | | 4/O | | 4/S | |
| 3 | rd 3 | | | | | | 3/E |
| 1 | wr 1 | | | | | | |
| 3 | rd 1 | | | | | | |

# Aufgabe 3.2: MOESI-Protokoll

| Proz. | Aktion | Prozessor 1 | | Prozessor 2 | | Prozessor 3 | |
|---|---|---|---|---|---|---|---|
| | | Line 1 | Line 2 | Line 1 | Line 2 | Line 1 | Line 2 |
| | init | - | - | - | - | - | - |
| 1 | rd 4 | 4/E | | | | | |
| 3 | rd 4 | 4/S | | | | 4/S | |
| 2 | rd 3 | | | 3/E | | | |
| 2 | wr 5 | | | | 5/M | | |
| 1 | rd 2 | | 2/E | | | | |
| 2 | wr 4 | 4/I | | 4/M | | 4/I | |
| 2 | rd 1 | | | | 1/E | | |
| 3 | rd 4 | | | 4/O | | 4/S | |
| 3 | rd 3 | | | | | | 3/E |
| 1 | wr 1 | 1/M | | | 1/I | | |
| 3 | rd 1 | | | | | | |

# Aufgabe 3.2: MOESI-Protokoll

| Proz. | Aktion | Prozessor 1 | | Prozessor 2 | | Prozessor 3 | |
|---|---|---|---|---|---|---|---|
| | | Line 1 | Line 2 | Line 1 | Line 2 | Line 1 | Line 2 |
| | init | - | - | - | - | - | - |
| 1 | rd 4 | 4/E | | | | | |
| 3 | rd 4 | 4/S | | | | 4/S | |
| 2 | rd 3 | | | 3/E | | | |
| 2 | wr 5 | | | | 5/M | | |
| 1 | rd 2 | | 2/E | | | | |
| 2 | wr 4 | 4/I | | 4/M | | 4/I | |
| 2 | rd 1 | | | | 1/E | | |
| 3 | rd 4 | | | 4/O | | 4/S | |
| 3 | rd 3 | | | | | | 3/E |
| 1 | wr 1 | 1/M | | | 1/I | | |
| 3 | rd 1 | 1/O | | | | 1/S | |

# MOESI-Zustandsautomat

Karlsruher Institut für Technologie

## Aufgabe 3.2 - Teilaufgabe c

Wieviele Hauptspeicherzugriffe werden durch diese Speicherzugriffsfolge verursacht? Führt hier die Verwendung des MOESI gegenüber des MESI-Protokolls zu einer Leistungssteigerung und wenn ja, warum?

## Antwort

MOESI: Lesend: 6 Zugriffe – Schreibend: 1 Zugriff
MESI: Lesend: 8 Zugriffe – Schreibend: 3 Zugriffe

Es würden zwei Lesezugriffe und zwei Schreibzugriffe eingespart → Leistungssteigerung vorhanden

# MOESI-Zustandsautomat

Karlsruher Institut für Technologie

## Aufgabe 3.2 - Teilaufgabe c

Wieviele Hauptspeicherzugriffe werden durch diese Speicherzugriffsfolge verursacht? Führt hier die Verwendung des MOESI gegenüber des MESI-Protokolls zu einer Leistungssteigerung und wenn ja, warum?

## Antwort

MOESI: Lesend: 6 Zugriffe – Schreibend: 1 Zugriff
MESI: Lesend: 8 Zugriffe – Schreibend: 3 Zugriffe

Es würden zwei Lesezugriffe und zwei Schreibzugriffe eingespart → Leistungssteigerung vorhanden

# Aufgabe 2.3: Verständnisfragen

a) Warum existiert im MOESI-Protokoll kein Zustandsübergang vom Zustand **S** nach Zustand **O**?

**Antwort:** Ein Zustandsübergang von **O** nach **S** kann es in einem Write-Back Invalidate-Protokoll nicht geben. Gemäß einem Write-Back Invalidate-Protokoll werden bei Veränderung eines geteilten Datums, die Datums in den entfernten Caches invalidiert und demzufolge dem Datum den Zustand **M** zugewiesen.

Der Wechsel von Zustand **S** in den Zustand **O**, müsste eine Aktualisierung des Datum in den entfernten Caches nach sich ziehen. Diese Vorgehensweise entspricht einem Write-Back Update-Protokoll

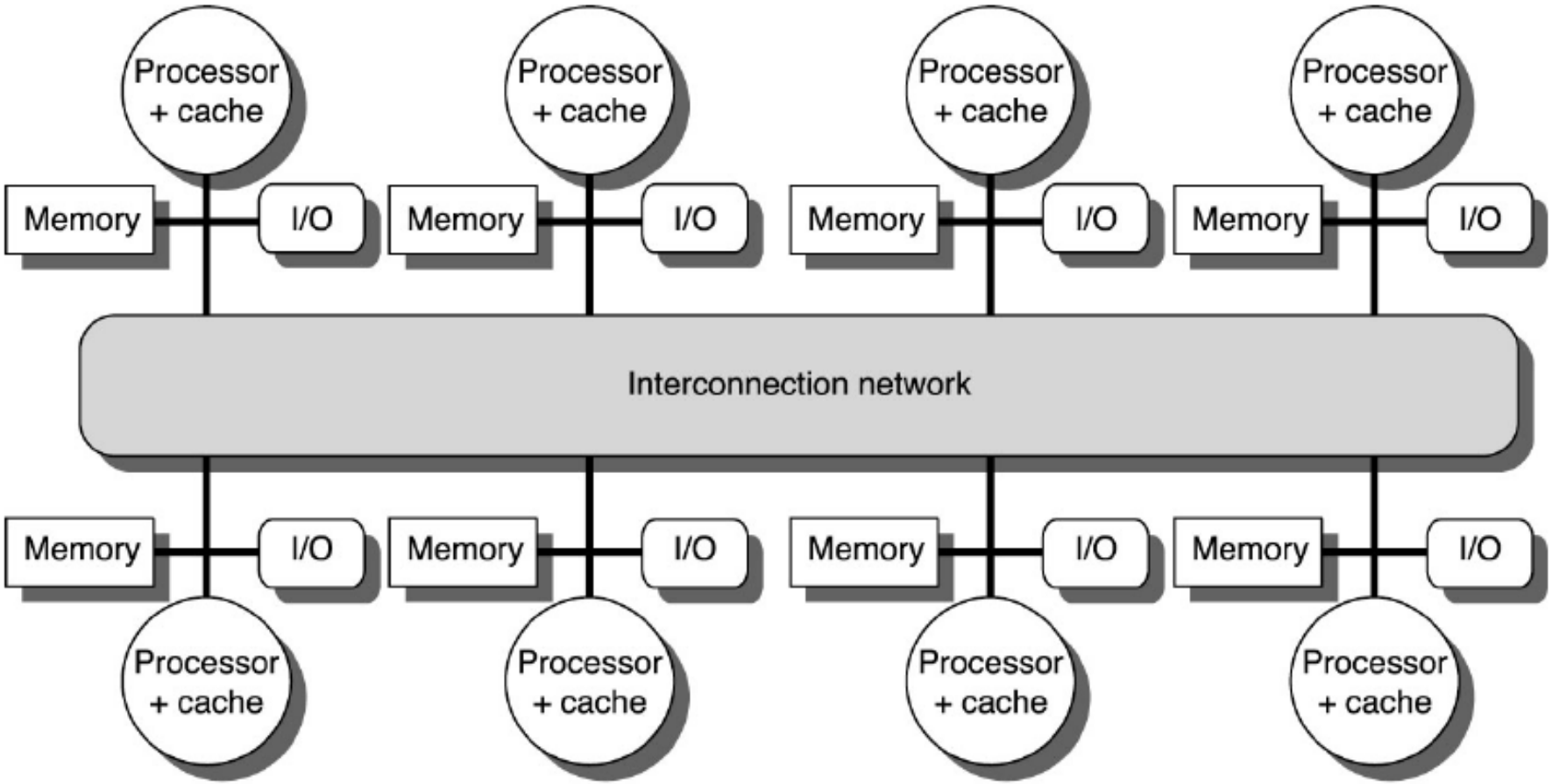
# Aufgabe 2.3: Verständnisfragen

Karlsruher Institut für Technologie

a) Warum existiert im MOESI-Protokoll kein Zustandsübergang vom Zustand **S** nach Zustand **O**?
**Antwort:** Ein Zustandsübergang von **O** nach **S** kann es in einem Write-Back Invalidate-Protokoll nicht geben. Gemäß einem Write-Back Invalidate-Protokoll werden bei Veränderung eines geteilten Datums, die Datums in den entfernten Caches invalidiert und demzufolge dem Datum den Zustand **M** zugewiesen.

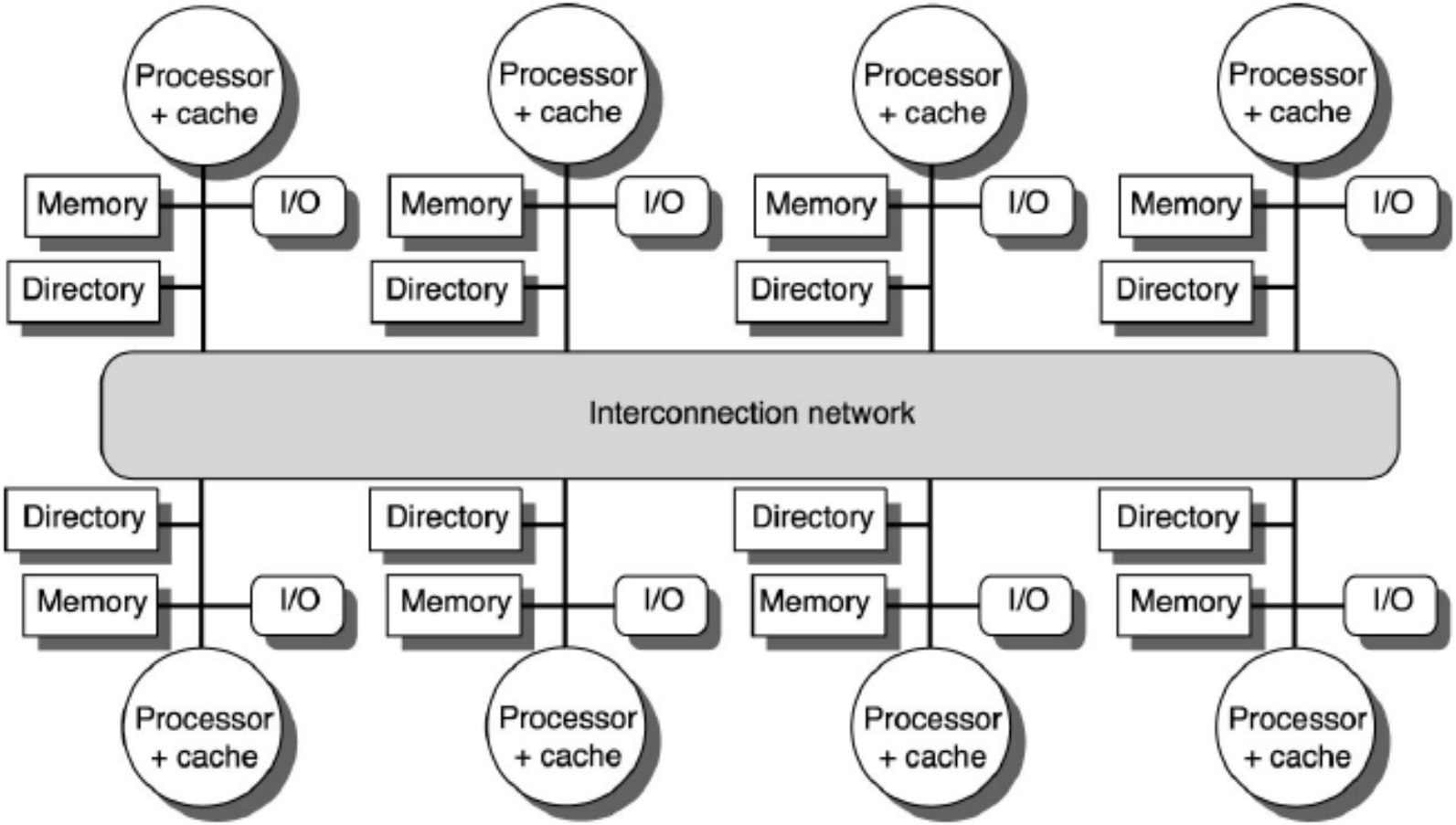
Der Wechsel von Zustand **S** in den Zustand **O**, müsste eine Aktualisierung des Datum in den entfernten Caches nach sich ziehen. Diese Vorgehensweise entspricht einem Write-Back Update-Protokoll

# Distributed-Shared-Memory (DSM)-System




© 2003 Elsevier Science (USA). All rights reserved.

# Verzeichnisbasierte Cache-Kohärenzprotokolle




© 2003 Elsevier Science (USA). All rights reserved.

# Verzeichnisbasierte Cache-Kohärenzprotokolle

Karlsruher Institut für Technologie

## Einfachste Methode zur Wahrung der Kohärenz in DSM-Systemen

Nur Speicherzugriffe auf den lokalen Speicher werden in den Cache geladen!

- DSM-Systeme haben kein gemeinsamer Speicherbus, den eine Snooping-Logik überwachen könnte
- Herstellen der Cache-Kohärenz über Verzeichnistabellen
- Implementierung in Hard- oder Software möglich
- Die Tabelle protokolliert für jeden Blockrahmen, ob dieser in den lokalen oder einem entfernten Cache-Speicher als Cache-Block übertragen worden ist.
- Zustände werden analog zu den MESI-Zuständen definiert

# Aufgabe 2.3: Verständnisfragen

b) Warum läßt sich MESI nicht in Distributed Shared Memory (DSM) Systemen einsetzen?

Antwort:

b) In DSM-Systemen existiert kein gemeinsamer Speicherbus, den eine Snooping-Logik überwachen könnte.

# Aufgabe 2.3: Verständnisfragen

Karlsruher Institut für Technologie

b) Warum läßt sich MESI nicht in Distributed Shared Memory (DSM) Systemen einsetzen?

Antwort:

b) In DSM-Systemen existiert kein gemeinsamer Speicherbus, den eine Snooping-Logik überwachen könnte.

# Aufgabe 2.3: Verständnisfragen

c) Welche Protokolle stellen in DSM-Systemen die Cache-Kohärenz sicher?

Antwort:

c) In DSM-Systemen kommen verzeichnisbasierte Cache-Kohärenzprotokolle zum Einsatz.

# Aufgabe 2.3: Verständnisfragen

c) Welche Protokolle stellen in DSM-Systemen die Cache-Kohärenz sicher?

Antwort:

c) In DSM-Systemen kommen verzeichnisbasierte Cache-Kohärenzprotokolle zum Einsatz.

# Literatur


1. Belady et al.: An anomaly in space-time characteristics of certain programs running in a paging machine, Communications of the ACM, Volume 12, Issue 6, Pages: 349 - 353, 1969
2. David E. Culler, Jaswinder Pal Singh: Parallel Computer Architecture - A Hardware/Software Approach Morgan Kaufmann, 1999, ISBN 1-55860-343-3
3. AMD64 Architecture Programmer's Manual Vol 2 'System Programming'
   `http://www.amd.com/us-en/assets/content_type/white_papers_and_tech_docs/24593.pdf`

# Organisatorisches - Klausur

Karlsruher Institut für Technologie

## Klausur

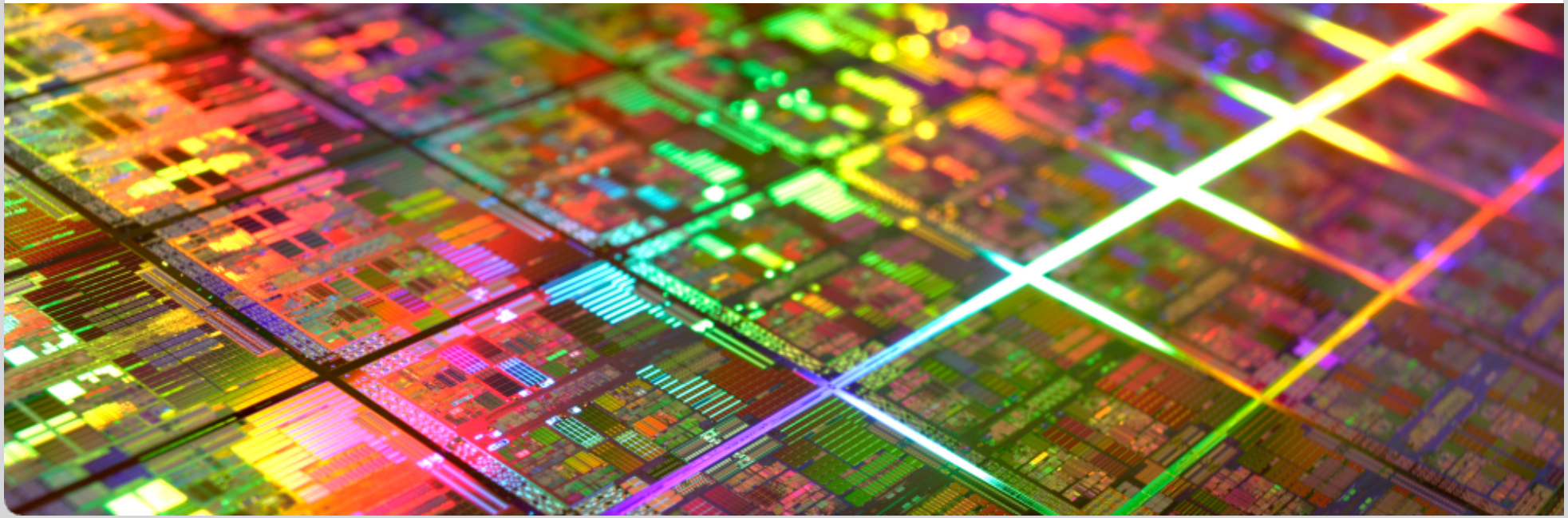
- Anmeldung zur Klausur via Studierendenportal
- Anmeldeschluss ist der 07.08.2011
- Danach besteht kein Anrecht mehr auf Klausurteilnahme!
- Abmeldeschluss ist der 09.08.2011
- Klausurtermin: 11.8.2011 um 14 Uhr
- Hörsäale: Gerthsen-HS, Benz-HS, Daimler-HS
  - Bekanntgabe Hörsaalverteilung: 10.8.2011
- **Es sind keine Hilfsmittel zugelassen!**

Karlsruher Institut für Technologie

# Zentralübung Rechnerstrukturen im SS 2011

## Vektorrechner und Cache-Kohärenz

David Kramer, Prof. Dr. Wolfgang Karl

Lehrstuhl für Rechnerarchitektur und Parallelverarbeitung

12. Juli 2011