

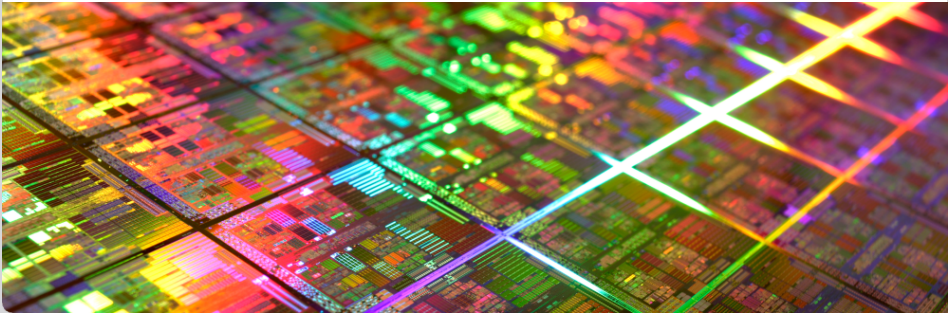
Zentralübung Rechnerstrukturen im SS 2016

Cache-Kohärenz

Michael Bromberger, Prof. Dr. Wolfgang Karl

Lehrstuhl für Rechnerarchitektur und Parallelverarbeitung

14. Juli 2016



Inhalt der 7. Übung

- Motivation für Caches
- Cacheorganisation
- Eigenschaften von Caches
- Cachekoheränzprotokolle
 - MESI
 - MOESI
- DSM-Systeme
- Organisatorisches zur Klausur
- Evaluation

Memory Bottleneck

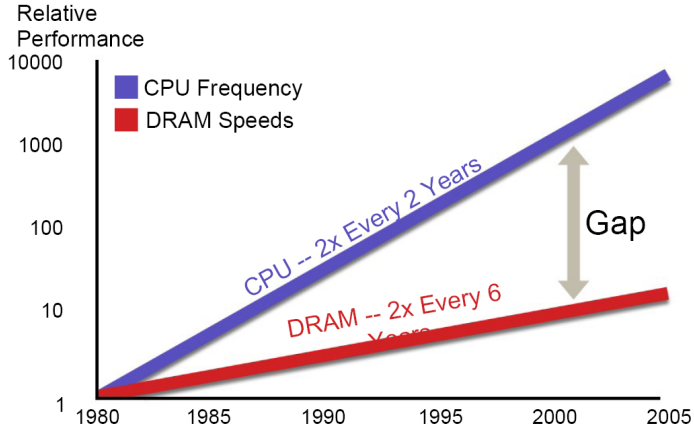


Abbildung: Quelle: <http://blogs.sun.com/toddjobson/resource/>

Cache - Motivation (forts.)

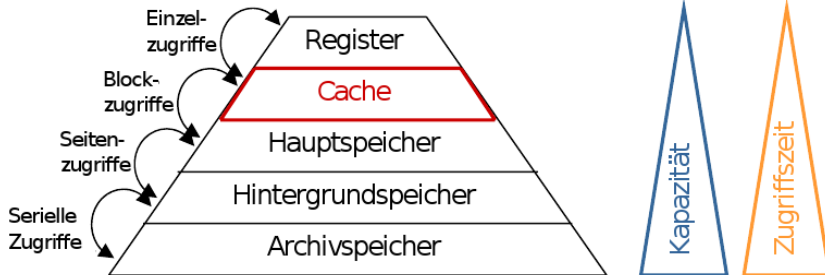


Abbildung: Speicherhierarchie

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

Organisation

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

Hierarchie

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

Größe

- Cacheline-Size
- # Cachelines bzw. Assoziativität * # Sätzen

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

- Bei Caches kann zwischen unterschiedlichen auftretenden Fehlzugriffen unterschieden werden

- Compulsory Miss
 - Miss, der durch ersten Zugriff auf ein Datum entsteht
 - Daten also noch nicht im Cache und daher Miss nicht vermeidbar

- Capacity Miss
 - Kapazität des Caches zu klein, weshalb bereits gecachtes Datum aus Cache entfernt werden musste

■ Conflict Miss

- Set zu klein und bereits gecachtes Datum wurde aus Set entfernt (Cache nicht zwangsläufig voll)
- Nur bei nicht-vollassoziativem Cache

■ Coherency Miss

- Nur bei Multiprozessor-Systemen mit Kohärenzprotokoll
- Unterscheidung zwischen False- und True-Sharing
- False-Sharing, falls nicht eigentliches Wort sondern anderes Wort in Cache-Block geändert wurde

Aufgabe 1.1: Cacheleistung

- Hit-Rate r_H
- Miss-Rate $r_M = 1 - r_H$
- Zugriffszeit bei Hit t_H
- Zugriffszeit Hauptspeicher t_{Mem} (falls Miss)

Mittlere Zugriffszeit

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

Mehrstufige Hierarchie: Konkretisierung des Miss-Zweigs

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

Aufgabe 1.1: Cacheleistung

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

- a) Geben Sie eine Formel zur Berechnung t_a der mittleren Zugriffszeit in einer zwei-stufigen Cache-Hierarchie, bei der Zugriffe auf die nächsten Ebenen parallel stattfinden, 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 1.1: Cacheleistung

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

- a) **Geben Sie eine Formel zur Berechnung t_a der mittleren Zugriffszeit in einer zwei stufigen Cache-Hierarchie, bei der Zugriffe auf die nächsten Ebenen parallel stattfinden, 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 1.1: Cacheleistung

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

- a) **Geben Sie eine Formel zur Berechnung t_a der mittleren Zugriffszeit in einer zwei stufigen Cache-Hierarchie, bei der Zugriffe auf die nächsten Ebenen parallel stattfinden, 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 1.1: Cacheleistung

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

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

$$\begin{aligned} t_a &= 70\% * 10 \text{ ns} + (1 - 70\%) * (40\% * 30 \text{ ns} + (1 - 40\%) * 100 \text{ ns}) \\ t_a &= 7 \text{ ns} + 0,3 * (12 \text{ ns} + 60 \text{ ns}) = 28,6 \text{ ns} \end{aligned}$$

Alternative B

$$\begin{aligned} t_a &= 75\% * 12 \text{ ns} + (1 - 75\%) * (35\% * 25 \text{ ns} + (1 - 35\%) * 100 \text{ ns}) \\ t_a &= 9 \text{ ns} + 0,25 * (8,75 \text{ ns} + 65 \text{ ns}) = 27,4375 \text{ ns} \end{aligned}$$

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

Aufgabe 1.1: Cacheleistung

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

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

$$t_a = 70\% * 10 \text{ ns} + (1 - 70\%) * (40\% * 30 \text{ ns} + (1 - 40\%) * 100 \text{ ns})$$
$$t_a = 7 \text{ ns} + 0,3 * (12 \text{ ns} + 60 \text{ ns}) = 28,6 \text{ ns}$$

Alternative B

$$t_a = 75\% * 12 \text{ ns} + (1 - 75\%) * (35\% * 25 \text{ ns} + (1 - 35\%) * 100 \text{ ns})$$
$$t_a = 9 \text{ ns} + 0,25 * (8,75 \text{ ns} + 65 \text{ ns}) = 27,4375 \text{ ns}$$

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

Aufgabe 1.1: Cacheleistung

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

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

$$t_a = 70\% * 10 \text{ ns} + (1 - 70\%) * (40\% * 30 \text{ ns} + (1 - 40\%) * 100 \text{ ns})$$
$$t_a = 7 \text{ ns} + 0,3 * (12 \text{ ns} + 60 \text{ ns}) = 28,6 \text{ ns}$$

Alternative B

$$t_a = 75\% * 12 \text{ ns} + (1 - 75\%) * (35\% * 25 \text{ ns} + (1 - 35\%) * 100 \text{ ns})$$
$$t_a = 9 \text{ ns} + 0,25 * (8,75 \text{ ns} + 65 \text{ ns}) = 27,4375 \text{ ns}$$

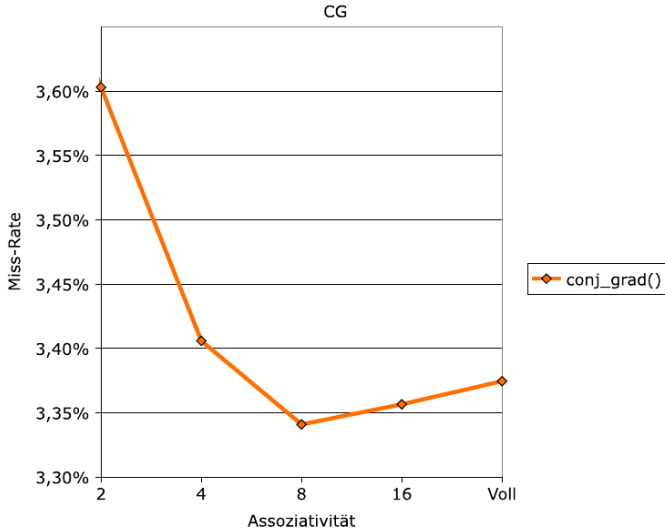
Die durchschnittliche Zugriffszeit in **Entwurfsalternative B** ist geringer, somit ist diese Entwurfsalternative zu wählen.

Aufgabe 1.2: Beweise

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 1.2: Beweise - Motivation



Aufgabe 1.2: Beweise

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:

Finden einer Folge von Cache-Belegungen bei der Aussage falsch

Aufgabe 1.2: 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:

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

Aufgabe 1.2: 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*

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

Aufgabe 1.2: 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*

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

Aufgabe 1.2: 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*

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

Aufgabe 1.2: 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*

Voraussetzung: Speicherzugriffe A, B, E, F werden auf den ersten Satz abgebildet

C und D auf den zweiten Satz des 2-fach assoziativen Cache

Aufgabe 1.2: 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: $ABCD(ABEFCD)^n$

Voraussetzung: Speicherzugriffe A, B, E, F werden auf den ersten Satz abgebildet

C und D auf den zweiten Satz des 2-fach assoziativen Cache

Aufgabe 1.2: Beweise (forts.)

Beweisen oder widerlegen Sie folgende Behauptungen:

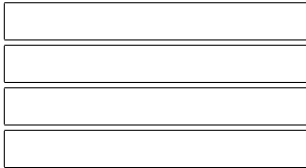
Hinweis: Gehen Sie in ihren Überlegungen davon aus, dass die Caches gemäß Least-Recently-Used (LRU)-Strategie verdrängen.

- 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.

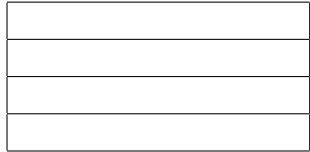
Aufgabe 1.2: Beweise (forts.)

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

Direct Mapped



Vollassoziativ



Folge:

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

Aufgabe 1.2: 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*

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

Aufgabe 1.2: Beweise (forts.)

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

Direct Mapped

A
B
C
D H

Vollassoziativ

A H
B
C
D

Folge: *ABCDH*

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

Aufgabe 1.2: Beweise (forts.)

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

Direct Mapped

A	A
B	B
C	C
D	H

Vollassoziativ

A	H
B	A
C	B
D	C

Folge: *ABCDHABC*

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

Aufgabe 1.2: Beweise (forts.)

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

Direct Mapped

A	A
B	B
C	C
D	H D H

Vollassoziativ

A	H	D
B	A	H
C	B	
D	C	

Folge: $(ABCDH)^n$

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

Aufgabe 1.2: 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	H	D	H

Vollassoziativ

A	H	D	C
B	A	H	
C	B	A	
D	C	B	

Folge: $(ABCDH)^n$

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

Aufgabe 1.3 - Verständnisfragen

- a) Welche Eigenschaft von Anwendungen werden von Caches ausgenutzt?

Antworten:

- a) Räumliche und Zeitliche Lokalität

Aufgabe 1.3 - Verständnisfragen

- a) Welche Eigenschaft von Anwendungen werden von Caches ausgenutzt?

Antworten:

- a) Räumliche und Zeitliche Lokalität

Aufgabe 1.3 - Verständnisfragen (forts.)

- 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 1.3 - Verständnisfragen (forts.)

- 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 1.3 c) - Motivation

- Cache-Speicher zeichnet sich durch seine schnellen Zugriffszeiten aus
- ⇒ Verwendet daher im Gegensatz zum Hauptspeicher static RAM Speicherbausteine
- SRAM verbraucht deutlich mehr Chipfläche als Speicherbausteine des Hauptspeichers
- ⇒ Deutlich geringere Datenkapazität bei gleichem Flächenverbrauch
- ⇒ Deutlich höhere Herstellungskosten pro Bit Speicher

Aufgabe 1.3 - Verständnisfragen (forts.)

- c) Warum wird der Cache-Speicher nicht deutlich größer (Größenbereich 100 MB - mehrere GB) angelegt?

Antworten:

- Benötigt deutlich größeren Die, wodurch der Yield bei der Herstellung sinkt
 - Herstellungskosten steigen
- Für Desktop-PCs auf Grund der erhöhten Herstellungskosten zu teuer
 - Kein ausreichendes Argument für Nicht-Anwendung
- SRAM Bausteine benötigen große Fläche für den angestrebten Größenbereich
 - Signale müssen Fläche durchlaufen, also deutlich längere Signalwege
 - ⇒ Signale brauchen länger und Zugriffszeit erhöht sich
 - ⇒ Vorteil von SRAM geht verloren

Aufgabe 1.3 - Verständnisfragen (forts.)

- c) Warum wird der Cache-Speicher nicht deutlich größer (Größenbereich 100 MB - mehrere GB) angelegt?

Antworten:

- Benötigt deutlich größeren Die, wodurch der Yield bei der Herstellung sinkt
 - Herstellungskosten steigen
- Für Desktop-PCs auf Grund der erhöhten Herstellungskosten zu teuer
 - Kein ausreichendes Argument für Nicht-Anwendung
- SRAM Bausteine benötigen große Fläche für den angestrebten Größenbereich
 - Signale müssen Fläche durchlaufen, also deutlich längere Signalwege
 - ⇒ Signale brauchen länger und Zugriffszeit erhöht sich
 - ⇒ Vorteil von SRAM geht verloren

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

Bus-Snooping-basiert

- Write-Invalidate Protokoll
 - arbeitet auf Cacheline-Ebene
 - benötigt nur ein Invalidate
 - Vertreter
 - MSI
 - MESI
 - MOESI
- Write-Update Protokoll
 - arbeitet auf Wort-Ebene
 - u. U. mehrere Update-Broadcasts nötig

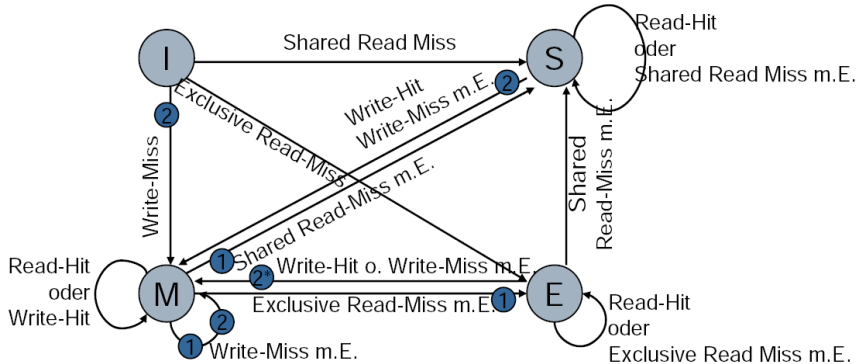
Verzeichnisbasiert

Wird in DSM-Systemen benötigt.

- 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

- Write-Invalidate Protokoll
- MESI: Zustandsautomat mit 4 Zuständen
 - M: exclusive 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)

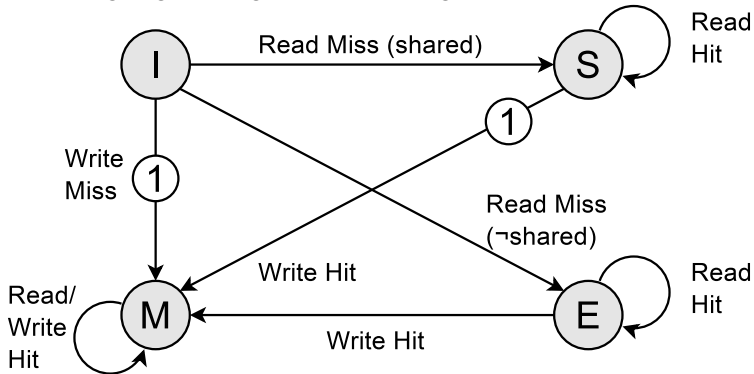


m.E.: mit Ersetzung

- 1 Cache-Zeile wird in den Hauptspeicher zurückkopiert (Line-Flush)
- 2 Cache-Zeilen in den anderen Caches mit gleicher Blockadresse werden invalidiert
- 2* Wie 2, gilt jedoch nur für Write-Miss mit Ersetzung

MESI-Zustandsautomat (2)

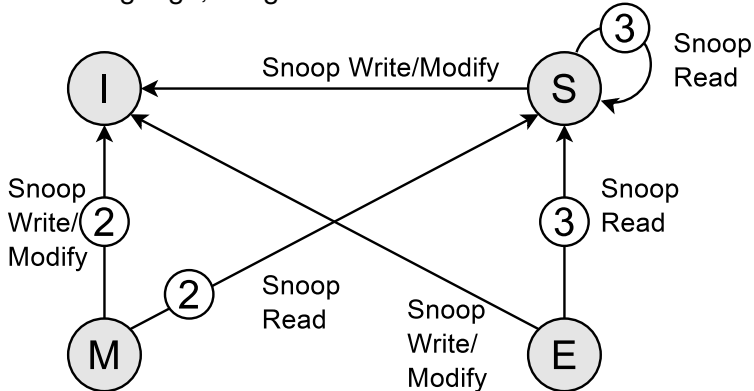
Zustandübergänge, ausgelöst durch eigene Aktionen:



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

MESI-Zustandsautomat (3)

Zustandsübergänge, ausgelöst durch andere Prozessoren:



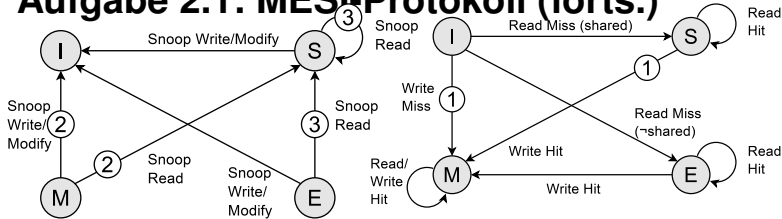
- 2 Retry-Signal wird aktiviert, Cache-Zeile in den Hauptspeicher kopiert, dann Signal deaktiviert
- 3 Shared-Signal wird aktiviert

Aufgabe 2.1: MESI-Protokoll

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, anderenfalls wird gemäß LRU-Strategie verdrängt. Als Cache-Kohärenzprotokoll komme das MESI-Protokoll zum Einsatz.

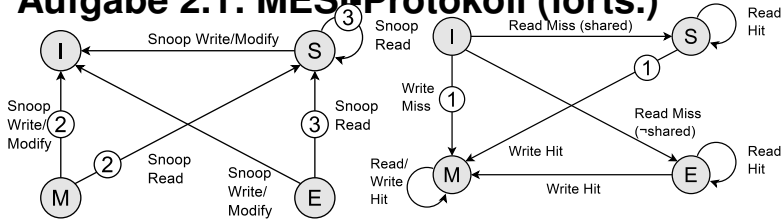
- Vervollständigen Sie die gegebene Tabelle: Geben Sie jeweils Inhalt der Cache-Zeile und MESI-Zustand an.

Aufgabe 2.1: MESI-Protokoll (forts.)



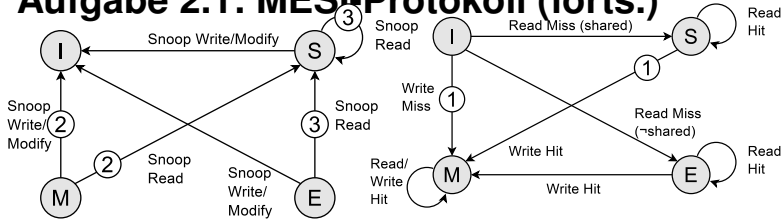
Proz.	Aktion	Prozessor 1		Prozessor 2		Prozessor 3	
		Line 1	Line 2	Line 1	Line 2	Line 1	Line 2
	init	-	-	-	-	-	-
1	rd 6						
2	rd 2						
1	rd 4						
3	rd 4						
2	rd 3						
3	wr 7						
1	wr 4						
2	rd 7						
3	wr 5						
1	rd 3						
3	wr 3						
2	wr 7						

Aufgabe 2.1: MESI-Protokoll (forts.)



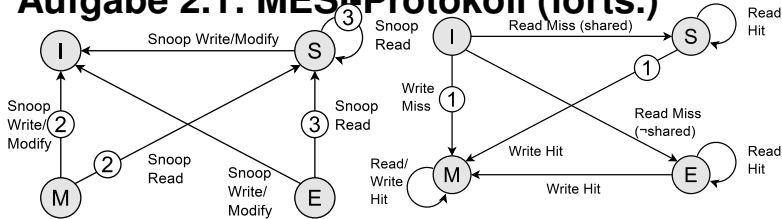
Proz.	Aktion	Prozessor 1		Prozessor 2		Prozessor 3	
		Line 1	Line 2	Line 1	Line 2	Line 1	Line 2
	init	-	-	-	-	-	-
1	rd 6	6/E					
2	rd 2			2/E			
1	rd 4		4/E				
3	rd 4						
2	rd 3						
3	wr 7						
1	wr 4						
2	rd 7						
3	wr 5						
1	rd 3						
3	wr 3						
2	wr 7						

Aufgabe 2.1: MESI-Protokoll (forts.)



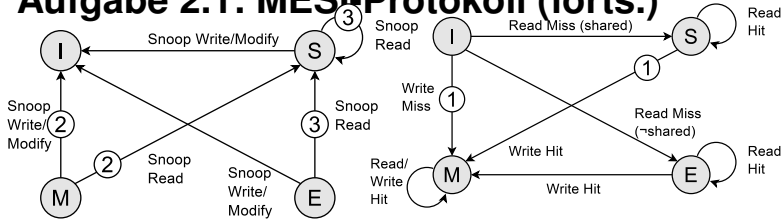
Proz.	Aktion	Prozessor 1		Prozessor 2		Prozessor 3	
		Line 1	Line 2	Line 1	Line 2	Line 1	Line 2
	init	-	-	-	-	-	-
1	rd 6	6/E					
2	rd 2			2/E			
1	rd 4		4/E				
3	rd 4		4/S			4/S	
2	rd 3						
3	wr 7						
1	wr 4						
2	rd 7						
3	wr 5						
1	rd 3						
3	wr 3						
2	wr 7						

Aufgabe 2.1: MESI-Protokoll (forts.)



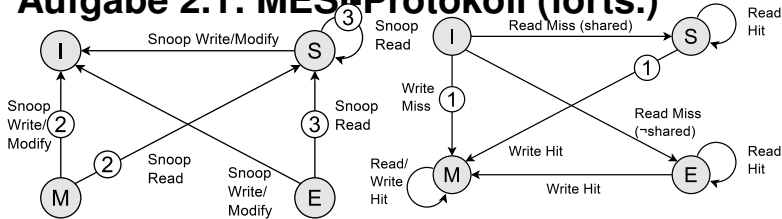
Proz.	Aktion	Prozessor 1		Prozessor 2		Prozessor 3	
		Line 1	Line 2	Line 1	Line 2	Line 1	Line 2
	init	-	-	-	-	-	-
1	rd 6	6/E					
2	rd 2			2/E			
1	rd 4		4/E				
3	rd 4		4/S			4/S	
2	rd 3				3/E		
3	wr 7						7/M
1	wr 4						
2	rd 7						
3	wr 5						
1	rd 3						
3	wr 3						
2	wr 7						

Aufgabe 2.1: MESI-Protokoll (forts.)



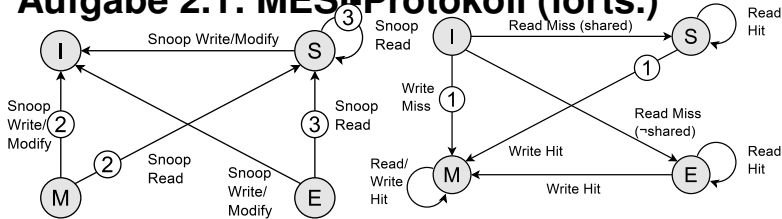
Proz.	Aktion	Prozessor 1		Prozessor 2		Prozessor 3	
		Line 1	Line 2	Line 1	Line 2	Line 1	Line 2
	init	-	-	-	-	-	-
1	rd 6	6/E					
2	rd 2			2/E			
1	rd 4		4/E				
3	rd 4		4/S			4/S	
2	rd 3				3/E		
3	wr 7						7/M
1	wr 4		4/M			4/I	
2	rd 7						
3	wr 5						
1	rd 3						
3	wr 3						
2	wr 7						

Aufgabe 2.1: MESI-Protokoll (forts.)



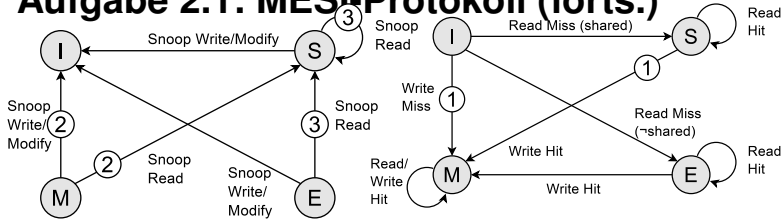
Proz.	Aktion	Prozessor 1		Prozessor 2		Prozessor 3	
		Line 1	Line 2	Line 1	Line 2	Line 1	Line 2
	init	-	-	-	-	-	-
1	rd 6	6/E					
2	rd 2			2/E			
1	rd 4		4/E				
3	rd 4		4/S			4/S	
2	rd 3				3/E		
3	wr 7						7/M
1	wr 4		4/M			4/I	
2	rd 7			7/S			7/S
3	wr 5						
1	rd 3						
3	wr 3						
2	wr 7						

Aufgabe 2.1: MESI-Protokoll (forts.)



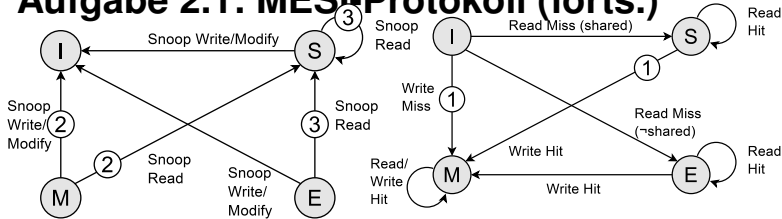
Proz.	Aktion	Prozessor 1		Prozessor 2		Prozessor 3	
		Line 1	Line 2	Line 1	Line 2	Line 1	Line 2
	init	-	-	-	-	-	-
1	rd 6	6/E					
2	rd 2			2/E			
1	rd 4		4/E				
3	rd 4		4/S			4/S	
2	rd 3				3/E		
3	wr 7						7/M
1	wr 4		4/M			4/I	
2	rd 7			7/S			7/S
3	wr 5					5/M	
1	rd 3						
3	wr 3						
2	wr 7						

Aufgabe 2.1: MESI-Protokoll (forts.)



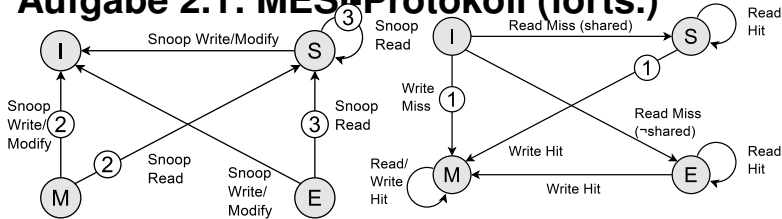
Proz.	Aktion	Prozessor 1		Prozessor 2		Prozessor 3	
		Line 1	Line 2	Line 1	Line 2	Line 1	Line 2
	init	-	-	-	-	-	-
1	rd 6	6/E					
2	rd 2			2/E			
1	rd 4		4/E				
3	rd 4		4/S			4/S	
2	rd 3				3/E		
3	wr 7						7/M
1	wr 4		4/M			4/I	
2	rd 7			7/S			7/S
3	wr 5					5/M	
1	rd 3	3/S			3/S		
3	wr 3						
2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

Aufgabe 2.1: 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 2.1: 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			

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

Vorteil

- Durch Cache-Cache-Transfers werden 2 Hauptspeicherzugriffe eingespart

Nachteile

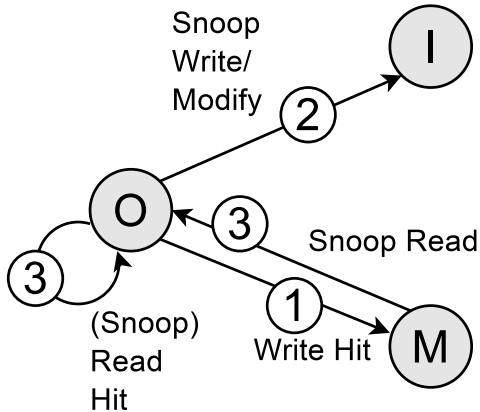
- Komplexere Logik notwendig
- 3 Zustands-Bits nötig

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

Zusätzliche Kanten für O-Zustand:



- 1 Invalidate
- 2 Retry
- 3 Shared

Aufgabe 2.2: MOESI-Protokoll

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 MOESI-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 2.2: 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 2.2: 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						
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 2.2: 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	wr 4						
2	rd 1						
3	rd 4						
3	rd 3						
1	wr 1						
3	rd 1						

Aufgabe 2.2: 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 2.2: 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 2.2: 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 2.2: 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 2.2: 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 2.2: 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 2.2: 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	

Aufgabe 2.2 - Teilaufgabe c

Wie viele 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 2.2 - Teilaufgabe c

Wie viele 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 **S** nach **O** kann es in einem Write-Invalidate-Protokoll nicht geben. Gemäß einem Write-Invalidate-Protokoll werden bei Veränderung eines geteilten Datums, die Daten in den entfernten Caches invalidiert und demzufolge dem Datum der 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-Update-Protokoll

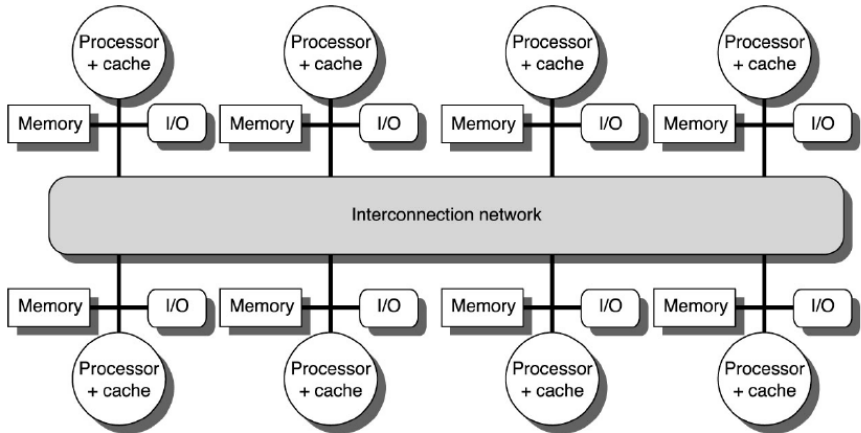
Aufgabe 2.3: Verständnisfragen

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

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

Distributed-Shared-Memory (DSM)-System

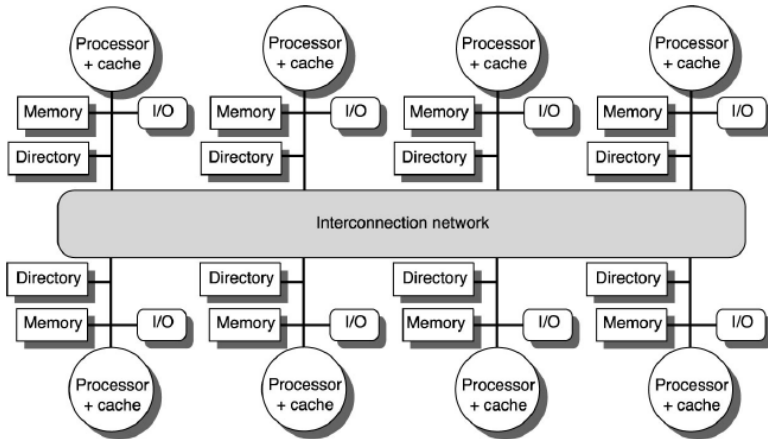


© 2003 Elsevier Science (USA). All rights reserved.

Einfachste Methode zur Wahrung der Kohärenz in DSM-Systemen:

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

Verzeichnisbasierte Cache-Kohärenzprotokolle



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

- 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

- 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.

Aufgabe 5

Ein Dreiprozessorsystem sei speichergekoppelt. Die Caches haben je eine Größe von zwei Cache-Zeilen, welche je genau ein Speicherwort aufnehmen können. Die Füllung des Caches erfolgt von der niedrigsten Cache-Zeile aufwärts, sofern noch freie Zeilen zur Verfügung stehen, andernfalls wird gemäß LRU-Strategie verdrängt. Als Cache-Kohärenzprotokoll komme das aus der Übung bekannte MOESI-Protokoll zum Einsatz. Der Cache sei initial leer. Aktionen, die durch das Cache-Kohärenzprotokoll ausgelöst werden und die eine Zustandsänderung einer Cache-Zeile bewirken, werden von der LRU-Strategie nicht als Zugriff gewertet.

- a) **Vervollständigen Sie die auf dem Lösungsblatt angegebene Tabelle: Geben Sie jeweils Inhalt der Cache-Zeile und MOESI-Zustand an.**

Prozessor	Aktion	Prozessor 1		Prozessor 2		Prozessor 3	
		Zeile 1	Zeile 2	Zeile 1	Zeile 2	Zeile 1	Zeile 2
	init	-	-	-	-	-	-
1	rd 3						
3	wr 3						
2	rd 3						
1	wr 4						
3	rd 4						
2	rd 1						
3	rd 2						
1	wr 2						
3	rd 1						
1	wr 4						

Prozessor	Aktion	Prozessor 1		Prozessor 2		Prozessor 3	
		Zeile 1	Zeile 2	Zeile 1	Zeile 2	Zeile 1	Zeile 2
	init	-	-	-	-	-	-
1	rd 3	3/E					
3	wr 3						
2	rd 3						
1	wr 4						
3	rd 4						
2	rd 1						
3	rd 2						
1	wr 2						
3	rd 1						
1	wr 4						

Prozessor	Aktion	Prozessor 1		Prozessor 2		Prozessor 3	
		Zeile 1	Zeile 2	Zeile 1	Zeile 2	Zeile 1	Zeile 2
	init	-	-	-	-	-	-
1	rd 3	3/E					
3	wr 3	3/I				3/M	
2	rd 3						
1	wr 4						
3	rd 4						
2	rd 1						
3	rd 2						
1	wr 2						
3	rd 1						
1	wr 4						

Prozessor	Aktion	Prozessor 1		Prozessor 2		Prozessor 3	
		Zeile 1	Zeile 2	Zeile 1	Zeile 2	Zeile 1	Zeile 2
	init	-	-	-	-	-	-
1	rd 3	3/E					
3	wr 3	3/I				3/M	
2	rd 3			3/S		3/O	
1	wr 4						
3	rd 4						
2	rd 1						
3	rd 2						
1	wr 2						
3	rd 1						
1	wr 4						

Prozessor	Aktion	Prozessor 1		Prozessor 2		Prozessor 3	
		Zeile 1	Zeile 2	Zeile 1	Zeile 2	Zeile 1	Zeile 2
	init	-	-	-	-	-	-
1	rd 3	3/E					
3	wr 3	3/I				3/M	
2	rd 3			3/S		3/O	
1	wr 4	4/M					
3	rd 4						
2	rd 1						
3	rd 2						
1	wr 2						
3	rd 1						
1	wr 4						

Prozessor	Aktion	Prozessor 1		Prozessor 2		Prozessor 3	
		Zeile 1	Zeile 2	Zeile 1	Zeile 2	Zeile 1	Zeile 2
	init	-	-	-	-	-	-
1	rd 3	3/E					
3	wr 3	3/I				3/M	
2	rd 3			3/S		3/O	
1	wr 4	4/M					
3	rd 4	4/O					4/S
2	rd 1						
3	rd 2						
1	wr 2						
3	rd 1						
1	wr 4						

Prozessor	Aktion	Prozessor 1		Prozessor 2		Prozessor 3	
		Zeile 1	Zeile 2	Zeile 1	Zeile 2	Zeile 1	Zeile 2
	init	-	-	-	-	-	-
1	rd 3	3/E					
3	wr 3	3/I				3/M	
2	rd 3			3/S		3/O	
1	wr 4	4/M					
3	rd 4	4/O					4/S
2	rd 1				1/E		
3	rd 2					2/E	
1	wr 2						
3	rd 1						
1	wr 4						

Prozessor	Aktion	Prozessor 1		Prozessor 2		Prozessor 3	
		Zeile 1	Zeile 2	Zeile 1	Zeile 2	Zeile 1	Zeile 2
	init	-	-	-	-	-	-
1	rd 3	3/E					
3	wr 3	3/I				3/M	
2	rd 3			3/S		3/O	
1	wr 4	4/M					
3	rd 4	4/O					4/S
2	rd 1				1/E		
3	rd 2					2/E	
1	wr 2		2/M			2/I	
3	rd 1						
1	wr 4						

Prozessor	Aktion	Prozessor 1		Prozessor 2		Prozessor 3	
		Zeile 1	Zeile 2	Zeile 1	Zeile 2	Zeile 1	Zeile 2
	init	-	-	-	-	-	-
1	rd 3	3/E					
3	wr 3	3/I				3/M	
2	rd 3			3/S		3/O	
1	wr 4	4/M					
3	rd 4	4/O					4/S
2	rd 1				1/E		
3	rd 2					2/E	
1	wr 2		2/M			2/I	
3	rd 1				1/S	1/S	
1	wr 4	4/M					4/I

Aufgabe 5

b) **Nennen Sie jeweils einen Vor- und Nachteil des MOESI-Protokolls gegenüber dem MESI-Protokoll.**

- Vorteil: Durch die Cache-Cache-Transfers werden 2 Hauptspeicherzugriffe gespart
- Nachteil:
 - Komplexere Logik notwendig
 - 3 Zustands-Bits nötig

Aufgabe 5

b) Nennen Sie jeweils einen Vor- und Nachteil des MOESI-Protokolls gegenüber dem MESI-Protokoll.

- Vorteil: Durch die Cache-Cache-Transfers werden 2 Hauptspeicherzugriffe gespart
- Nachteil:
 - Komplexere Logik notwendig
 - 3 Zustands-Bits nötig

Aufgabe 6

Für ein System stehen drei verschiedene Cache-Varianten zur Verfügung. Variante A ist eine zweistufige Cache-Hierarchie, wobei alle Ebenen gleichzeitig (parallel) angefragt werden. Variante B ist ebenfalls zweistufig, die verschiedenen Ebenen werden aber nacheinander (sequentiell) angefragt. Variante C besitzt nur eine Cache-Ebene, die Anfragen werden hier aber wieder gleichzeitig (parallel) durchgeführt. Das System darf eine durchschnittliche Antwortzeit von maximal 16 ns nicht überschreiten. Berechnen Sie die durchschnittlichen Antwortzeiten der drei Varianten und geben Sie an, welche der Varianten für das System infrage kommen.

Aufgabe 6

	Variante A	Variante B	Variante C
Zugriffszeit L1	8 <i>ns</i>	6 <i>ns</i>	6 <i>ns</i>
Hitrate L1	75 %	75 %	90 %
Zugriffszeit L2	20 <i>ns</i>	24 <i>ns</i>	-
Hitrate L2	80 %	80 %	-
Zugriffszeit Hauptspeicher	100 <i>ns</i>	100 <i>ns</i>	100 <i>ns</i>

- d) **Berechnen Sie die durchschnittlichen Antwortzeiten der drei Varianten und geben Sie an, welche der Varianten für das System infrage kommen.**

$t_{A-L1} = 8 \text{ ns}$, $t_{A-L2} = 20 \text{ ns}$, $t_{B-L1} = 6 \text{ ns}$, $t_{B-L2} = 24 \text{ ns}$, $t_{C-L1} = 6 \text{ ns}$, $t_{Mem} = 100 \text{ ns}$.

$r_{A-H1} = 75 \%$, $r_{A-H2} = 80 \%$, $r_{B-H1} = 75 \%$, $r_{B-H2} = 80 \%$, $r_{C-H1} = 90 \%$.

Aufgabe 6d) Lösung:

- Formel für parallele Zugriffe auf Hierarchieebene $\Rightarrow$ s. Folie 9
- Formel für sequentielle Zugriffe:

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

Alternative A:

$$\begin{aligned} & 0,75 * 8 \text{ ns} + 0,25 * (0,8 * 20 \text{ ns} + 0,2 * 100 \text{ ns}) \\ &= 6 \text{ ns} + 0,25 * (16 \text{ ns} + 20 \text{ ns}) \\ &= 6 \text{ ns} + 0,25 * 36 \text{ ns} \\ &= 6 \text{ ns} + 9 \text{ ns} = 15 \text{ ns} \end{aligned}$$

$t_{A-L1} = 8 \text{ ns}$, $t_{A-L2} = 20 \text{ ns}$, $t_{B-L1} = 6 \text{ ns}$, $t_{B-L2} = 24 \text{ ns}$, $t_{C-L1} = 6 \text{ ns}$, $t_{Mem} = 100 \text{ ns}$.

$r_{A-H1} = 75 \%$, $r_{A-H2} = 80 \%$, $r_{B-H1} = 75 \%$, $r_{B-H2} = 80 \%$, $r_{C-H1} = 90 \%$.

Aufgabe 6d) Lösung:

- Formel für parallele Zugriffe auf Hierarchieebene $\Rightarrow$ s. Folie 9
- Formel für sequentielle Zugriffe:

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

Alternative A:

$$\begin{aligned} & 0,75 * 8 \text{ ns} + 0,25 * (0,8 * 20 \text{ ns} + 0,2 * 100 \text{ ns}) \\ &= 6 \text{ ns} + 0,25 * (16 \text{ ns} + 20 \text{ ns}) \\ &= 6 \text{ ns} + 0,25 * 36 \text{ ns} \\ &= 6 \text{ ns} + 9 \text{ ns} = 15 \text{ ns} \end{aligned}$$

$$t_{A-L1} = 8 \text{ ns}, t_{A-L2} = 20 \text{ ns}, t_{B-L1} = 6 \text{ ns}, t_{B-L2} = 24 \text{ ns}, t_{C-L1} = 6 \text{ ns}, t_{Mem} = 100 \text{ ns}.$$

$$r_{A-H1} = 75 \%, r_{A-H2} = 80 \%, r_{B-H1} = 75 \%, r_{B-H2} = 80 \%, r_{C-H1} = 90 \%.$$

Aufgabe 6d) Lösung:

- Formel für parallele Zugriffe auf Hierarchieebene $\Rightarrow$ s. Folie 9
- Formel für sequentielle Zugriffe:

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

Alternative B:

$$\begin{aligned} & 0,75 * 6 \text{ ns} + 0,25 * (0,8 * 30 \text{ ns} + 0,2 * 130 \text{ ns}) \\ &= 4,5 \text{ ns} + 0,25 * (24 \text{ ns} + 26 \text{ ns}) \\ &= 4,5 \text{ ns} + 0,25 * 50 \text{ ns} \\ &= 4,5 \text{ ns} + 12,5 \text{ ns} = 17 \text{ ns} \end{aligned}$$

$t_{A-L1} = 8 \text{ ns}$, $t_{A-L2} = 20 \text{ ns}$, $t_{B-L1} = 6 \text{ ns}$, $t_{B-L2} = 24 \text{ ns}$, $t_{C-L1} = 6 \text{ ns}$, $t_{Mem} = 100 \text{ ns}$.

$r_{A-H1} = 75 \%$, $r_{A-H2} = 80 \%$, $r_{B-H1} = 75 \%$, $r_{B-H2} = 80 \%$, $r_{C-H1} = 90 \%$.

Aufgabe 6d) Lösung:

- Formel für parallele Zugriffe auf Hierarchieebene $\Rightarrow$ s. Folie 9
- Formel für sequentielle Zugriffe:

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

Alternative C:

$$0,9 * 6 \text{ ns} + 0,1 * 100 \text{ ns} = 5,4 \text{ ns} + 10 \text{ ns} = 15,4 \text{ ns}$$

Somit sind die Alternativen A und C für einen Einsatz im System geeignet.

Aufgabe 6

- e) **Was besagt die 90/10 Regel, die für die Verwendung von Caches eine wichtige Rolle spielt?**

Die 90/10 Regel besagt, dass bei 90 % der Speicherzugriffe innerhalb eines Programms nur etwa 10 % der Daten benötigt werden, die das Programm insgesamt während seiner Laufzeit benötigt. Somit ist also ein deutlich kleinerer Speicher ausreichend, um die meisten Anfragen beantworten zu können.

Aufgabe 6

- e) **Was besagt die 90/10 Regel, die für die Verwendung von Caches eine wichtige Rolle spielt?**

Die 90/10 Regel besagt, dass bei 90 % der Speicherzugriffe innerhalb eines Programms nur etwa 10 % der Daten benötigt werden, die das Programm insgesamt während seiner Laufzeit benötigt. Somit ist also ein deutlich kleinerer Speicher ausreichend, um die meisten Anfragen beantworten zu können.

Klausur am 24.08.2016, 11:00 Uhr

- Anmeldung ist online möglich via Studierendenportal!
 - Anmeldeschluss ist der 17.08.2016
 - **Danach besteht kein Anrecht mehr auf Klausurteilnahme!**

 - Abmeldung online bis 17.08.2016
 - **Hinweise auf der RS-Homepage beachten!**

 - Erasmus-Studenten,...
- ⇒ Bitte Prüfungszulassung persönlich im Büro oder nach den Übungen abgeben!

Vorlesung – 19.07.2016

- VL fällt aus

Übung #8 – 21.07.2016

- Vektorrechner
- Voraussichtlich
- Fragen/Antworten zur Klausur

Fragen?

Zentralübung Rechnerstrukturen im SS 2016

Cache-Kohärenz

Michael Bromberger, Prof. Dr. Wolfgang Karl

Lehrstuhl für Rechnerarchitektur und Parallelverarbeitung

14. Juli 2016

