

Vorlesung: Rechnerstrukturen, Teil 2 (Modul IP7)

J. Zhang

zhang@informatik.uni-hamburg.de

Universität Hamburg
Fachbereich Informatik
AB Technische Aspekte Multimodaler Systeme

Inhaltsverzeichnis

8. Die Speicherhierarchie 304

 DRAM als Cache 304

 virtuellen Speicher: Speicherverwaltung 306

 Virtuelle Speicher: Schutzmechanismen 309

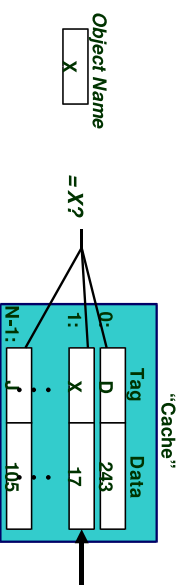
 Multi-Ebenen Seiten-Tabellen 329

 Zusammenfassung der virtuellen Speicher 331

 Das Speichersystem von Pentium und Linux 333

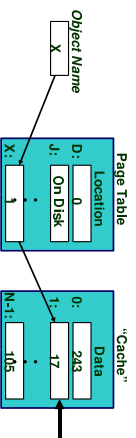
 Zusammenfassung Speichersystem 339

Auffinden eines Objekts in einem Cache



- Tag wird zusammen mit den Daten in der Cachezeile gespeichert
- Bildet vom Cacheblock auf Speicherblöcke ab
 - ◆ Spart einige Bits, da nur Tag gespeichert wird
- Kein Tag für nicht im Cache befindlichen Block
- Auslesen der Informationen mit spezieller Hardware
 - ◆ kann schnell mit mehreren Tags vergleichen

Auffinden eines Objekts in einem Cache (Forts.)



DRAM Cache

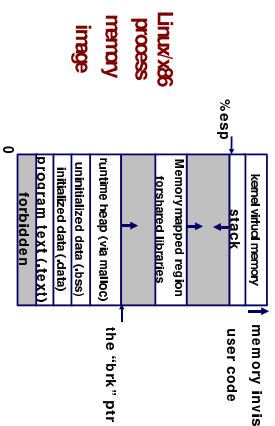
- Jede zugeteilte Seite des virtuellen Speichers hat einen Eintrag in der Seiten-Tabelle
- Abbilden von virtuellen Seiten auf physikalische
- Seiten-Tabelleneintrag existiert, selbst wenn die Seite nicht im Speicher liegt
 - ◆ Gibt Festplattenadresse an
 - ◆ Einzige Methode, um Seite zu finden
- OS liest Informationen aus

Motivation 2 für virt. Speicher: Speicherverwaltung

Mehrere Prozesse können im physikalischen Speicher liegen

Wie lösen wir die Adresskonflikte?

- Was passiert, wenn zwei Prozesse auf dieselbe Adresse zugreifen?



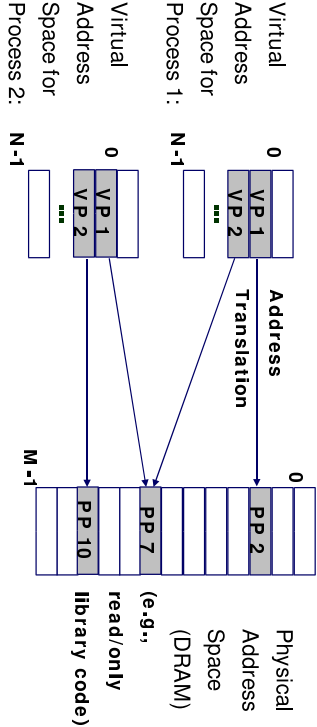
Seite 306

Lösung: Separate Virt. Adr. Räume

- Virtuelle und physikalische Adressräume sind in gleichgroße Blöcke unterteilt
 - ◆ Blöcke werden Seiten ("Pages") genannt (sowohl virtuelle als auch physikalische)
- Jeder Prozess hat seinen eigenen virtuellen Adressraum
 - ◆ Das Betriebssystem kontrolliert, wie virtuelle Seiten auf den physikalischen Speicher abgebildet werden

Seite 307

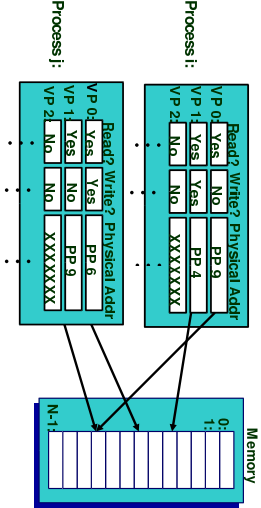
Lösung: Separate Virt. Adr. Räume (Forts.)



Motivation Nr. 3: Schutzmechanismen

Seiten-Tabelleneintrag enthält Informationen über Zugriffsrechte

- Hardware erzwingt den Schutz ("Trap" ("Exception") in OS bei Verstoß)



VM (“Virtual Memory”) Adressumsetzung

Virtueller Adressraum

- $V = \{0, 1, \dots, N-1\}$

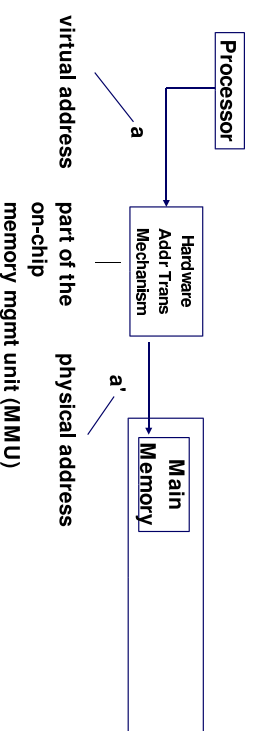
Physikalischer Adressraum

- $P = \{0, 1, \dots, M-1\}$
- $M < N$

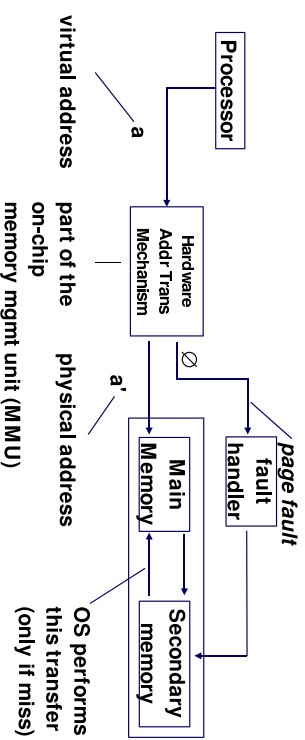
Adressumsetzung

- MAP: $V \rightarrow P \cup \{\emptyset\}$
- Für eine virtuelle Adresse a :
 - ◆ $\text{MAP}(a) = a'$, wenn Daten bei virtueller Adresse a und physikalischer Adresse a' in P sind
 - ◆ $\text{MAP}(a) = \emptyset$, wenn Daten bei virtueller Adresse a nicht im physikalischen Speicher sind, d. h. entweder ungültig oder nur auf Festplatte gespeichert sind.

VM Adressumsetzung: Hit



VM Adressumsetzung: Miss

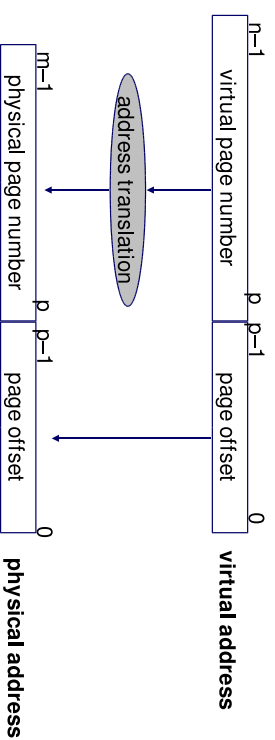


Seite 312

VM Adressumsetzung

Parameter

- $P = 2^p$ = Seitengröße (Bytes)
- $N = 2^n$ = Limit der virtuellen Adresse
- $M = 2^m$ = Limit der physikalischen Adresse

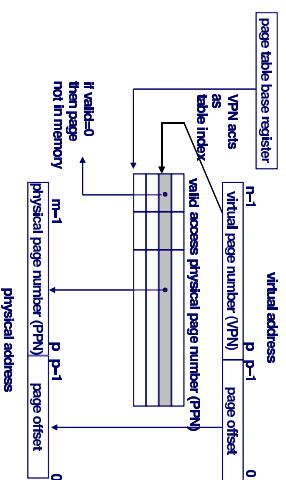


Seite 313

Seiten-Tabellenoperation

Umsetzung

- Eigene Menge von Seiten-Tabellen für jeden Prozess
- VPN ("Virtual Page Number") bildet den Index in der Seiten-Tabelle (zeigt auf einen Seiten-Tabelleneintrag)



Seiten-Tabellenoperation

Berechnung der physikalischen Adresse

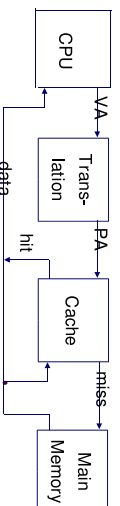
- Seiten-Tabelleneintrag liefert Informationen über die Seite.
 - ◆ wenn (valid Bit = 1), dann ist die Seite im Speicher, d.h. benutze physikalische Seitennummer (PPN "Physical Page Number"), um Adresse zu konstruieren
 - ◆ wenn (valid Bit = 0), dann ist die Seite auf der Festplatte, d.h. Seitenfehler

Seiten-Tabellenoperation

Schutzüberprüfung

- Zugriffsrechtefeld gibt Zugriffserlaubnis an
 - ◆ z.B. read-only, read-write, execute-only
 - ◆ typischerweise werden zahlreiche Schutzmodi unterstützt (z.B. Kernel gegen User)
- Schutzrechteverletzung: wenn Benutzer nicht die nötigen Rechte hat

Integration von VM und Cache



Die meisten Caches werden "physikalisch adressiert"

- Zugriff über physikalische Adressen
- Gestattet mehreren Prozessen, gleichzeitig Blöcke im Cache zu haben
- Gestattet mehreren Prozessen, die Seiten zu teilen
- Cache muss sich nicht mit Schutzproblemen befassen
 - ◆ Zugriffsrechte werden als Teil der Adressumsetzung überprüft

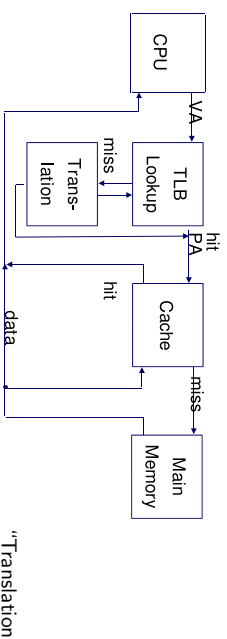
Die Adressumsetzung wird vor dem Cache "Lookup" durchgeführt

- Doch dies könnte selbst einen Speicherzugriff (auf den PTE) beinhalten
- Natürlich können Seiten-Tabelleneinträge auch "gecacht" werden

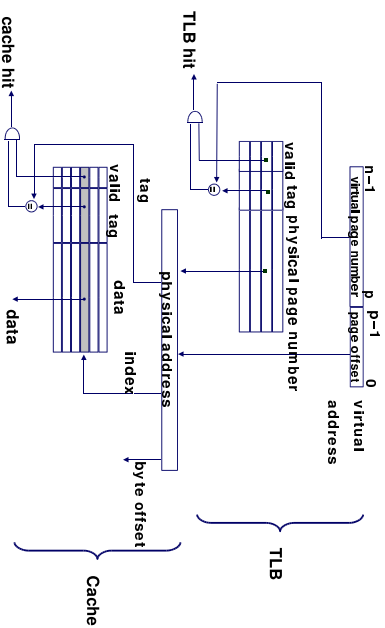
Beschleunigung der Umsetzung mit einem TLB

Lookaside Buffer" (TLB)

- Kleiner Hardware Cache in der MMU (Memory Management Unit)
- Bildet virtuelle Seitenzahlen auf physikalische ab
- Enthält die kompletten Seiten-Tabelleneinträge für wenige Seiten



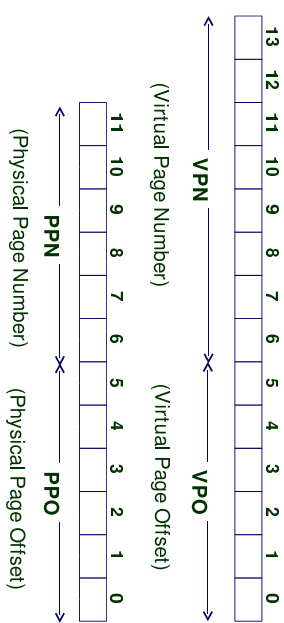
Adressumsetzung mit einem TLB



Beispiel für einfaches Speichersystem

Adressierung

- 14-Bit virtuelle Adresse
- 12-Bit physikalische Adresse
- Seitengröße = 64 Bytes



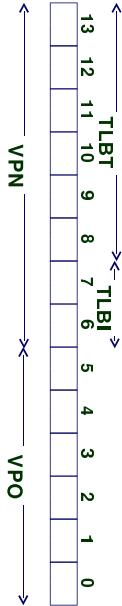
Seiten-Tabelle des einfachen Speichersystems

- Nur die ersten 16 Einträge werden gezeigt

VPN	PPN	Valid	VPN	PPN	Valid
00	28	1	08	13	1
01	-	0	09	17	1
02	33	1	0A	09	1
03	02	1	0B	-	0
04	-	0	0C	-	0
05	16	1	0D	2D	1
06	-	0	0E	11	1
07	-	0	0F	0D	1

Einfaches Speichersystem: TLB

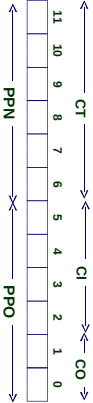
- 16 Einträge
- 4-fach assoziativ



Set	Tag	PPN	Valid	Tag	PPN	Valid	Tag	PPN	Valid	Tag	PPN	Valid
0	03	-	0	09	0D	0	04	-	0	0A	07	02
1	03	2D	1	02	-	0	04	-	0	0A	-	0
2	02	-	0	08	-	0	06	-	0	03	-	0
3	07	-	0	03	0D	1	0A	34	1	02	-	0

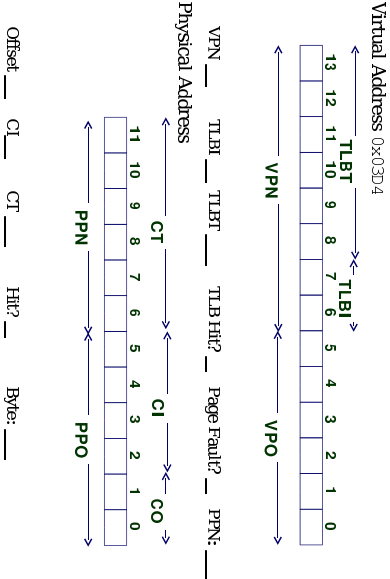
Einfaches Speichersystem: Cache

- 16 Zeilen
- 4-byte pro Zeile (line size)
- Direkt abgebildet

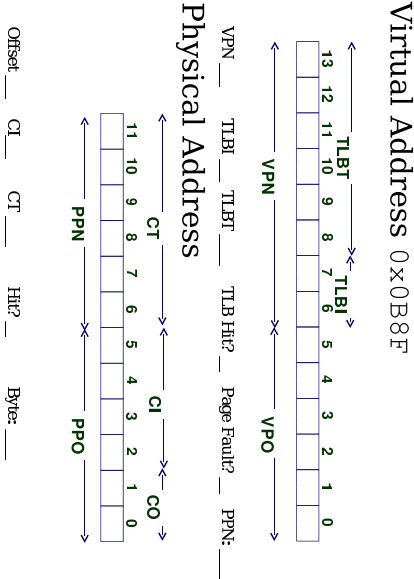


idx	Tag	Valid	B0	B1	B2	B3	idx	Tag	Valid	B0	B1	B2	B3
0	19	1	99	11	23	11	8	24	1	3A	00	51	89
1	15	0	-	-	-	-	9	2D	0	-	-	-	-
2	1B	1	00	02	04	08	A	2D	1	93	15	DA	3B
3	36	0	-	-	-	-	B	0B	0	-	-	-	-
4	32	1	43	6D	8F	09	C	12	0	-	-	-	-
5	0D	1	36	72	F0	1D	D	16	1	04	96	34	15
6	31	0	-	-	-	-	E	13	1	83	77	1B	D3
7	16	1	11	C2	DF	03	F	14	0	-	-	-	-

Adressumsetzungsbeispiel Nr. 1

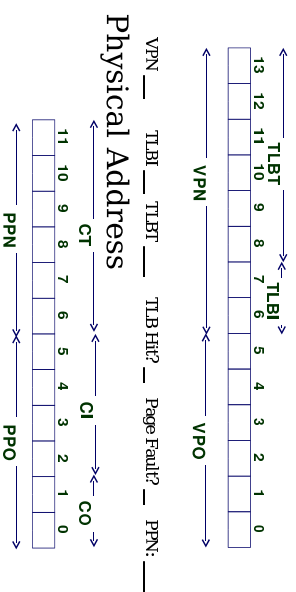


Adressumsetzungsbeispiel Nr. 2



Adressumsetzungsbeispiel Nr. 3

Virtual Address 0x0040

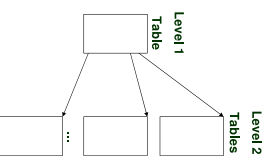


Offset __ Cl __ CT __ Hit? __ Byte: __

Multi-Ebenen Seiten-Tabellen

Gegeben:

- 4KB (2^{12}) Seitengröße
- 32-Bit Adressraum
- 4-Byte PTE ("Page Table Entry" = Seitentabelleneintrag)



Multi-Ebenen Seiten-Tabellen (Forts.)

Problem:

- Würde eine 4 MB Seiten-Tabelle benötigen!
 - ◆ 2^{20} Bytes

Übliche Lösung:

- Multi-Ebenen Seiten-Tabellen
- z.B. zweistufige Tabelle (Pentium P6)
 - ◆ Ebene-1 Tabelle: 1024 Einträge, wovon jeder auf eine Ebene-2 Seiten-Tabelle zeigt
 - ◆ Ebene-2 Tabelle: 1024 Einträge, wovon jeder auf eine Seite zeigt

Seite 330

Virtuelle Speicher: Hauptthemen

Aus Sicht des Programmiers:

- Großer "flacher" Adressraum
 - ◆ Kann große Blöcke benachbarter Adressen zuteilen
- Prozessor "besitzt" die gesamte Maschine
 - ◆ Hat privaten Adressraum
 - ◆ Bleibt unberührt vom Verhalten anderer Prozesse

Seite 331

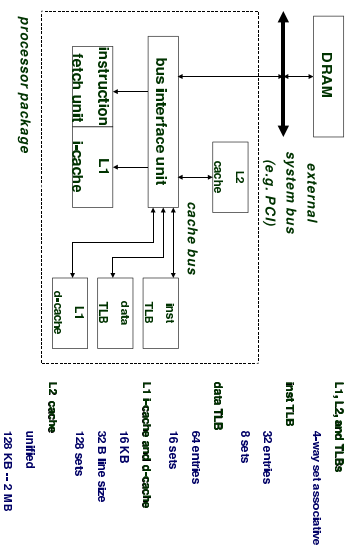
Virtuelle Speicher: Hauptthemen (Forts.)

Aus Sicht des Systems:

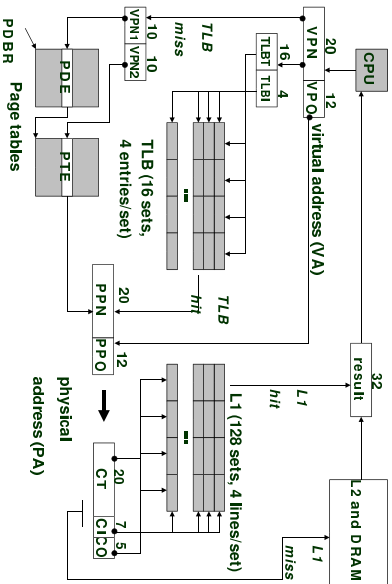
- Virtueller Adressraum des Benutzers wird durch Abbildung auf Seitenbereich erzeugt
 - ◆ Muss nicht fortlaufend sein
 - ◆ Wird dynamisch zugeteilt
 - ◆ Erzwingt Schutz bei Adressumsetzung
- OS kann viele Prozesse gleichzeitig verwalten
 - ◆ Ständiges Wechseln zwischen Prozessen
 - ◆ Vor allem wenn auf Ressourcen gewartet werden muss, z.B. Festplatten I/O um Seitenfehler zu beheben

Pentium Speichersystem (Pentium Pro+)

Pentium Memory System



Adressumsetzung beim Pentium



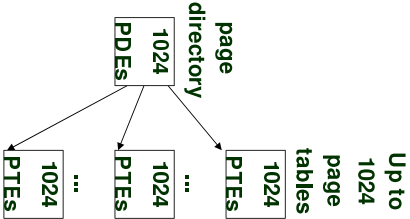
Zweistufige Seiten-Tabellenstruktur

Seiten-Verzeichnis (Page Directory)

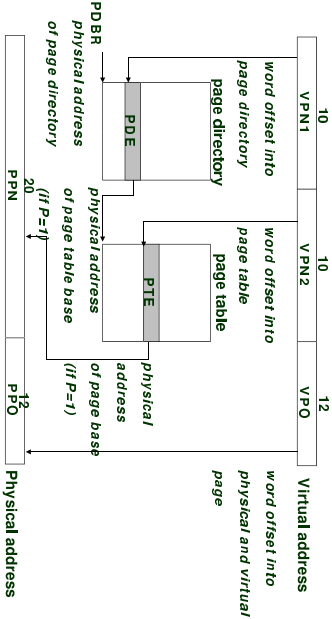
- 1024 4-Byte Einträge ("Page Directory Entries PDEs"), die auf Seitentabellen verweisen
- Ein Seiten-Verzeichnis pro Prozess
- Seiten-Verzeichnis muss im Speicher liegen, während der zugehörige Prozess läuft
- immer über PDBR (Page Directory Base Register) zugreifbar

Seiten-Tabellen:

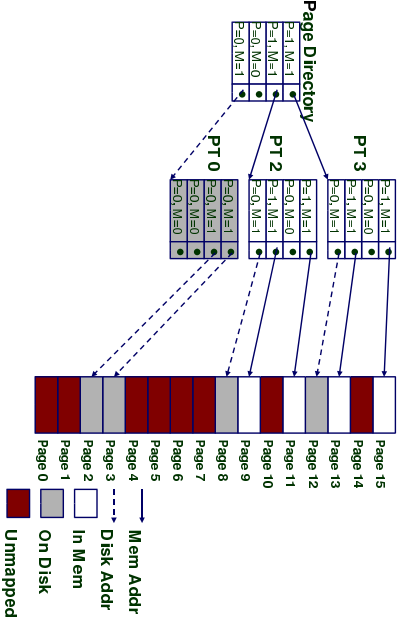
- 1024 4-byte Einträge ("Page Table Entries PTEs"), die auf Seiten verweisen
- Seiten-Tabellen können ein- und ausgelagert werden ("page in, page out")



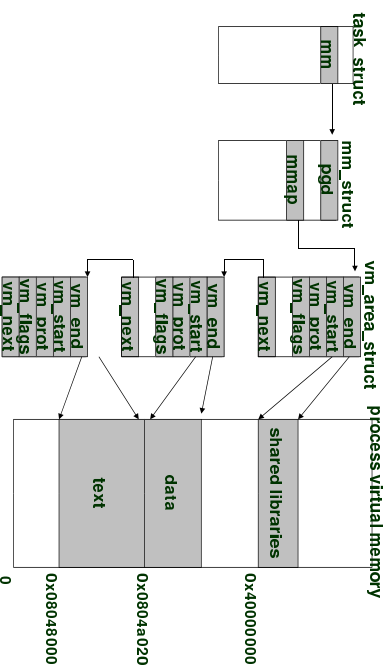
Umsetzung von virtuellen auf physikalische Adressen mit den Seiten-Tabellen des Pentiums



Darstellung des virtuellen Adressraums



Linux organisiert VM als Menge von Bereichen



Seite 338

Zusammenfassung Speichersystem

Cache Speicher

- Dient nur zur Beschleunigung
- Verhalten unsichtbar für Anwendungsprogrammierer und OS
- Komplet in Hardware implementiert

Virtueller Speicher

- Ermöglicht viele Funktionen des OS
 - ◆ Prozesse erzeugen (neue und abgeleitete ("forking"))
 - ◆ Taskwechsel
 - ◆ Schutzmechanismen
- Implementierung mit Hardware und Software
 - ◆ Software verwaltet die Tabellen und Zuteilungen
 - ◆ Hardwarezugriff auf die Tabellen
 - ◆ Hardware-Caching der Einträge (TLB)

Seite 339

Ergänzende Literatur

Zur Rechnerarchitektur (Teil 2, 8. Termin):

Literatur

- [1] Randal E. Bryant and David O'Hallaron. Computer systems. pages 690–721. Pearson Education, Inc., New Jersey, 2003.