Large and Fast: Exploiting Memory Hierarchy

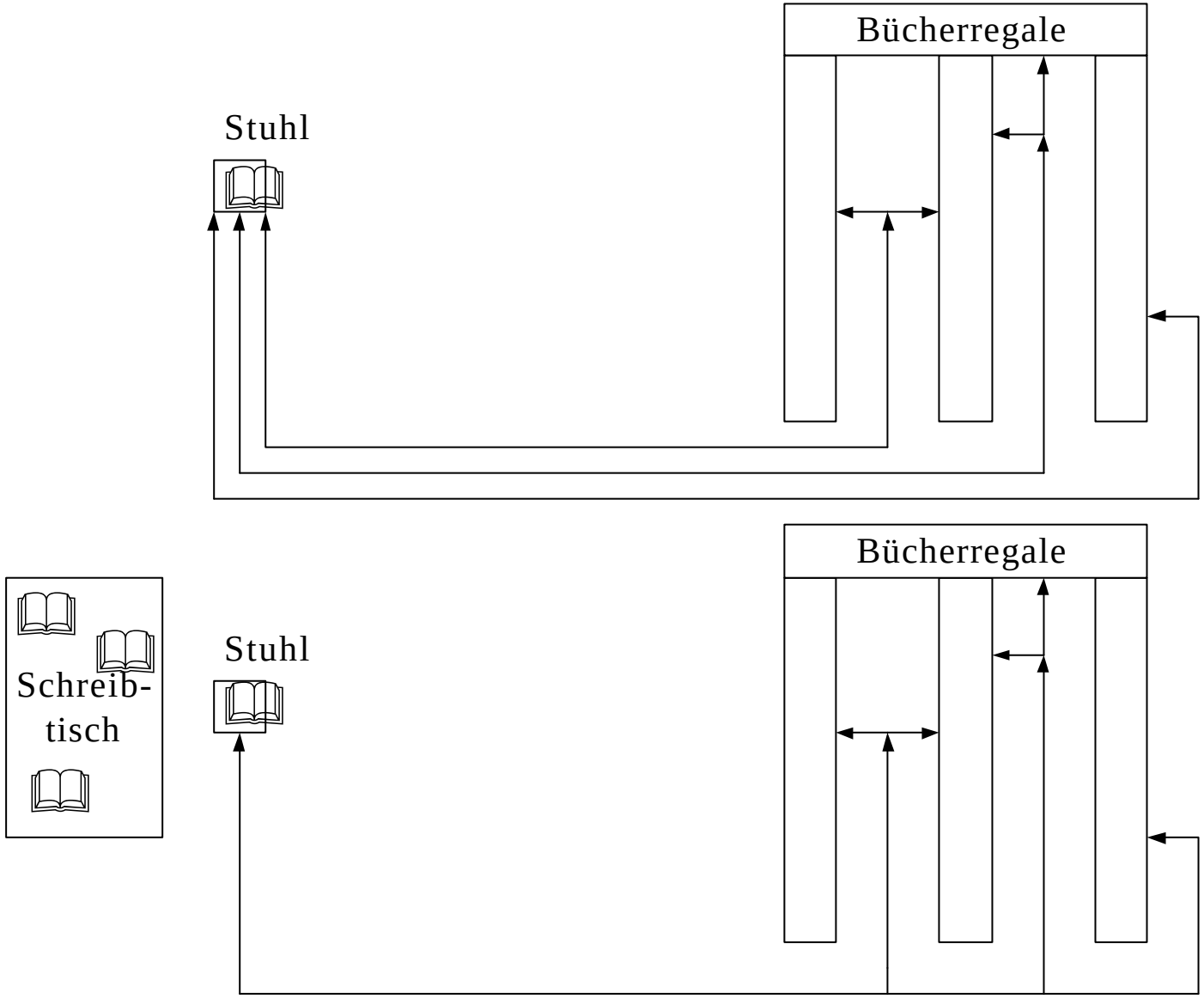
Cache und Speicherhierarchie

Inhalt :

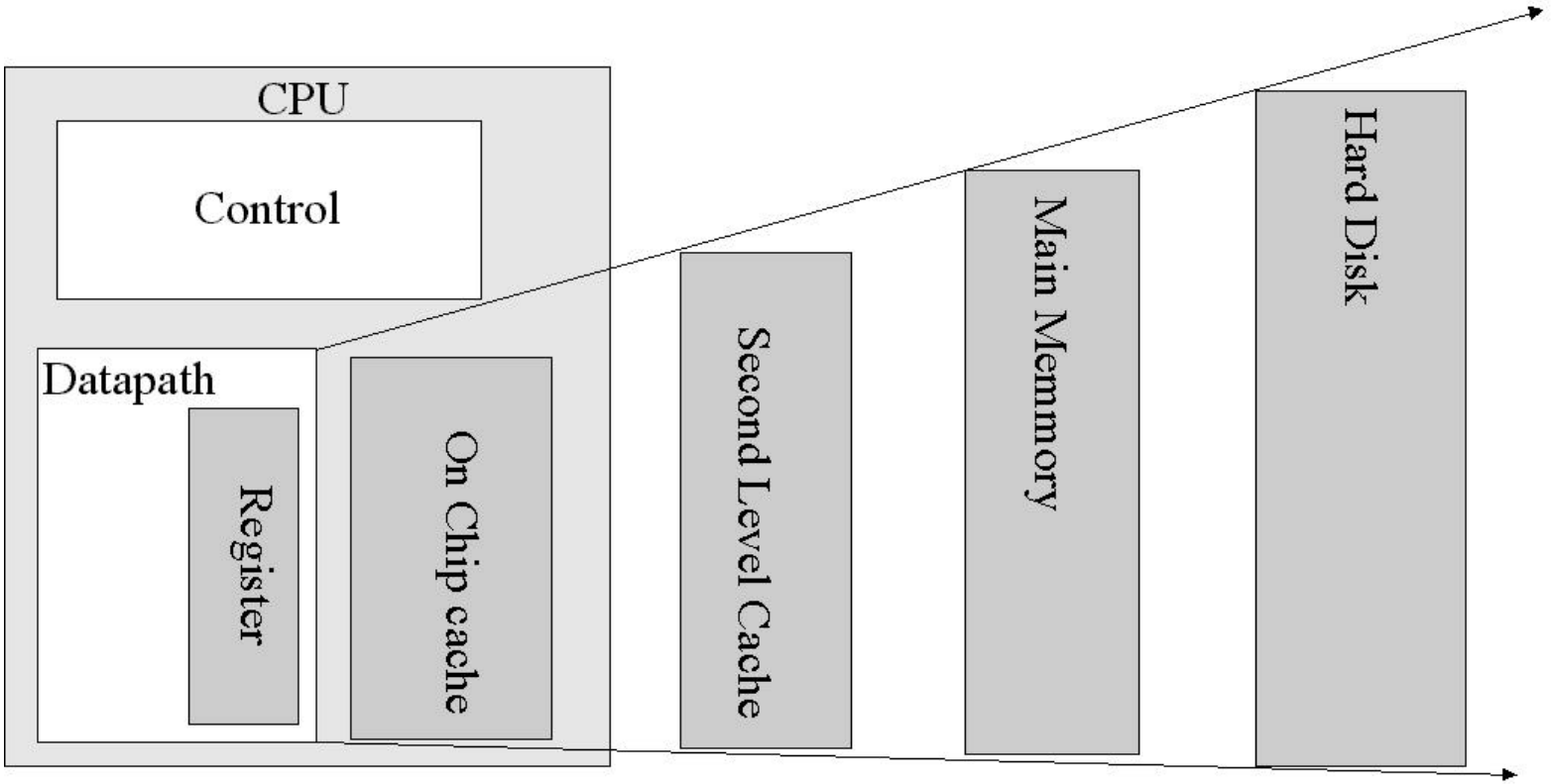
- Einleitung
- Grundlagen der Speicherhierarchie
- Cache: Aufbau und Verwaltung
- Virtueller Speicher
- Messen und Verbessern der Cache Performance
- Umsetzung am PC (Beispiele)
- Historisches

Idealer Weise würde man eine unbestimmt große Speicherkapazität anlegen, so daß jedes einzelne Wort sofort verfügbar ist...
Wir sind durch die Erkennung der Konstruktionsmöglichkeiten gezwungen, eine Speicherhierarchie zu finden, in der jede Stufe eine größere Kapazität als ihr Vorgänger hat, hieran gekoppelt ist allerdings eine längere Zugriffszeit.

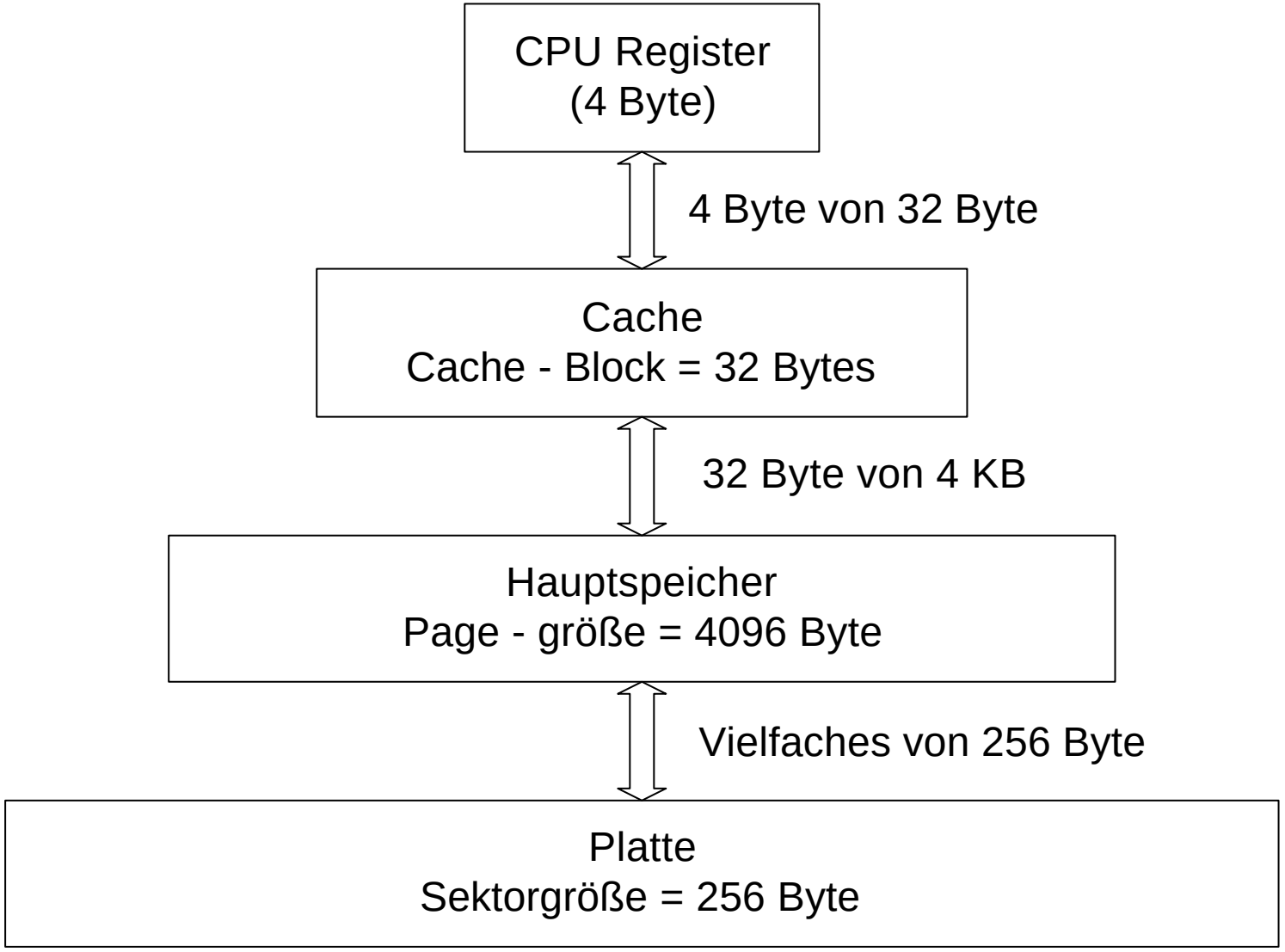
A.W. Burks, H. H. Goldstine, and J. von Neumann
Preliminary Discussion of the Logical Design of an Electronic Computing Instrument, 1946

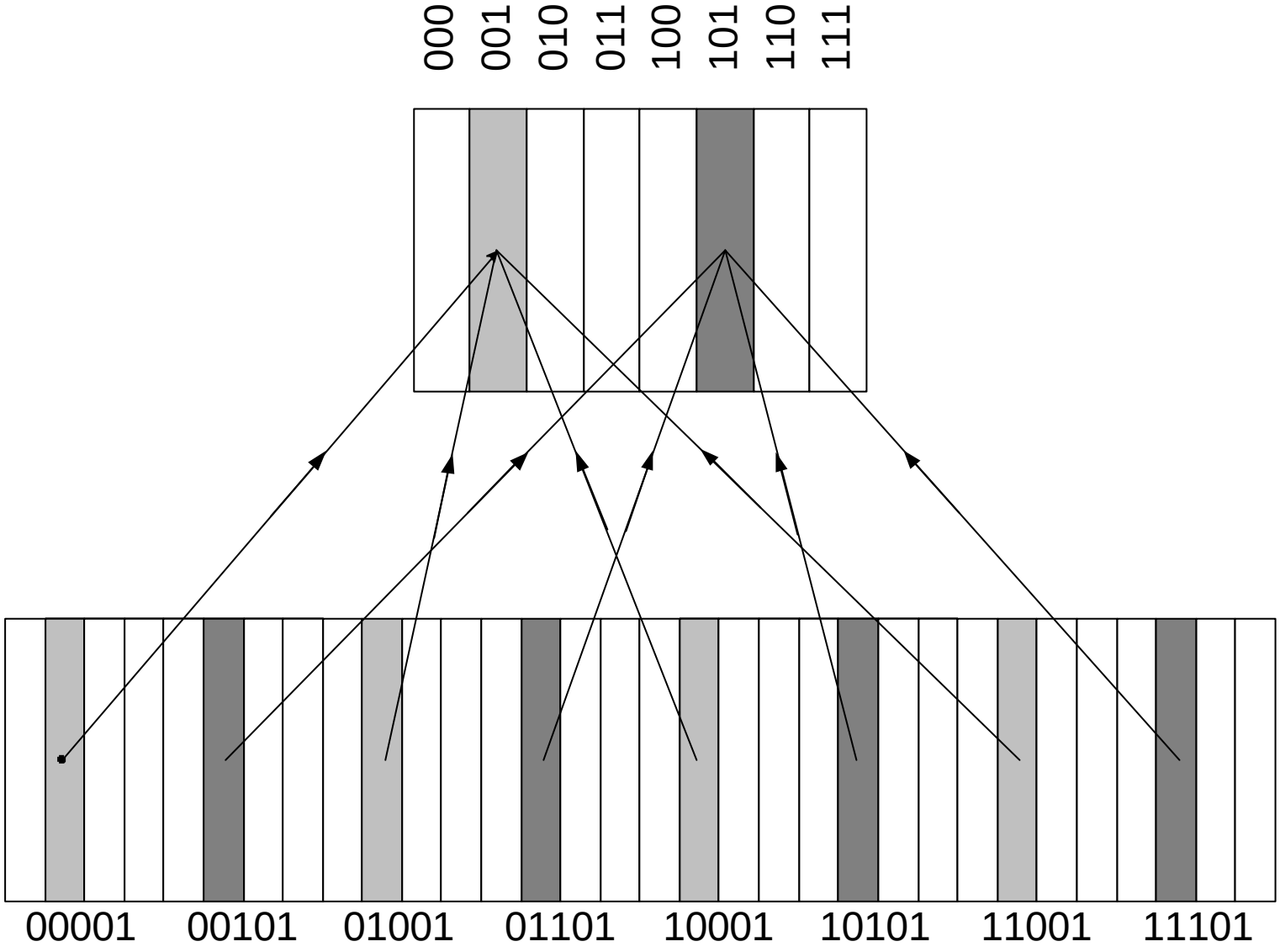

Bücherregale
Stuhl
Bücherregale
Schreib-
tisch
Stuhl

Speichertyp	Technologie	Größe	Zugriffszeit
Register		1 KB	1 ns
Cache	Statisches Halbleiter - RAM	512 KB	10 ns
Hauptspeicher	Dynamisches Halbleiter - RAM	128 MB	50 ns
Magnetische Platte	Festplatte	40 GB	10 ms
Optische Platte	CD-ROM	650 MB	300 ms
Archivband	Magnetband	100 MB - 5 GB	5 sec - 30 Min



Speed	1ns	1-3 ns	10 ns	10 ms
Size	100	Kilo	Mega	Giga
Cost	Highest			Lowest


CPU Register
(4 Byte)
4 Byte von 32 Byte
Cache
Cache - Block = 32 Bytes
32 Byte von 4 KB
Hauptspeicher
Page - größe = 4096 Byte
Vielfaches von 256 Byte
Platte
Sektorgröße = 256 Byte


000
001
010
011
100
101
110
111
00001
00101
01001
01101
10001
10101
11001
11101

Decimal address of reference	Binary address of reference	Hit or miss in Cache	Assigned cache block (where found or placed)
22	10110two	miss	(10110two mod 8)=110two
26	11010two	miss	(11010two mod 8)=010two
22	10110two	Hit	(10110two mod 8)=110two
26	11010two	Hit	(11010two mod 8)=010two
16	10000two	miss	(10000two mod 8)=000two
3	00011two	miss	(00011two mod 8)=011two
16	10000two	hit	(10000two mod 8)=000two
18	10010two	miss	(10010two mod 8)=010two

INDEX	V	Tag	Data
000			
001			
010			
011			
100			
101			
110			
111			

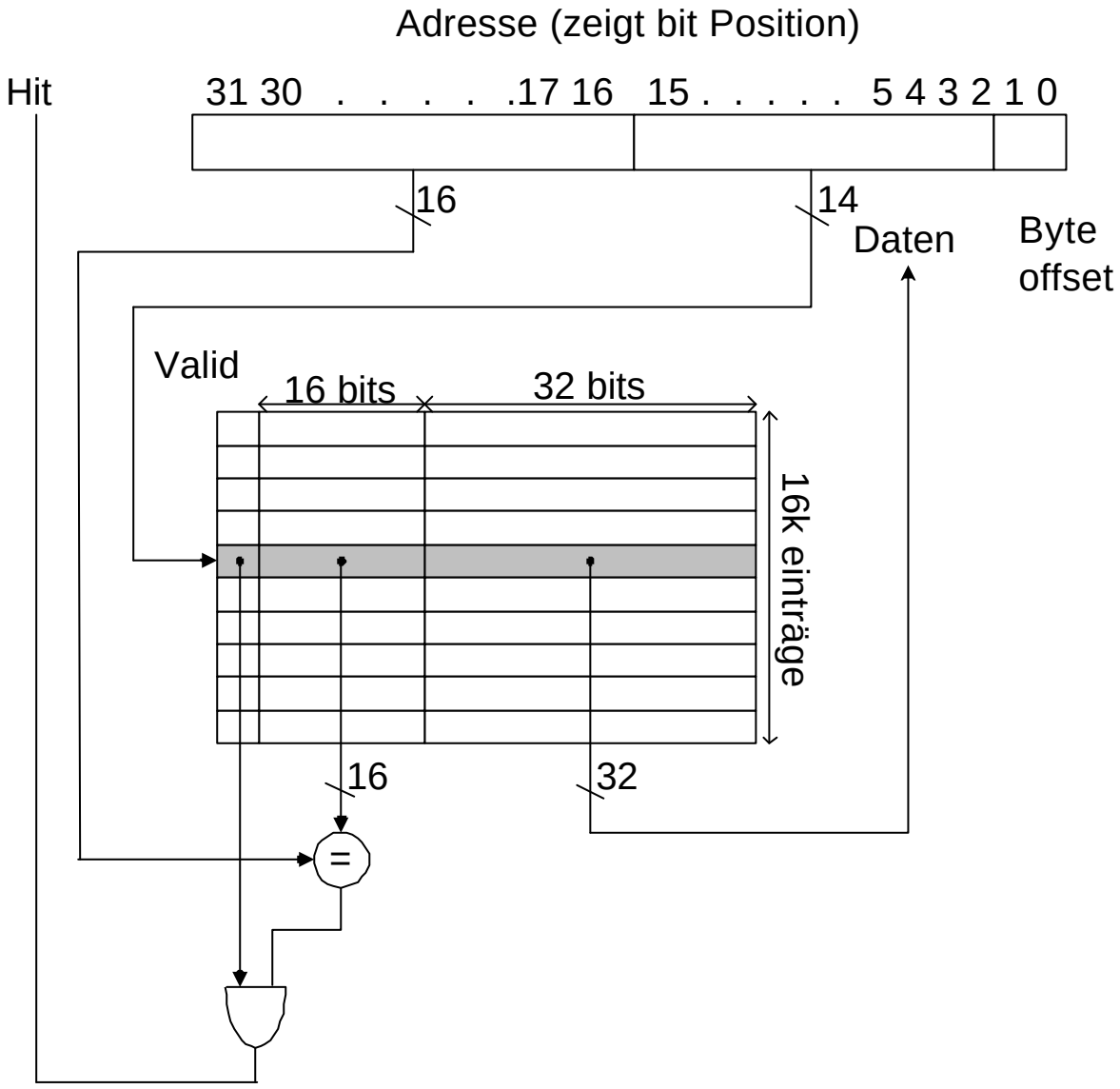
V:= Valid

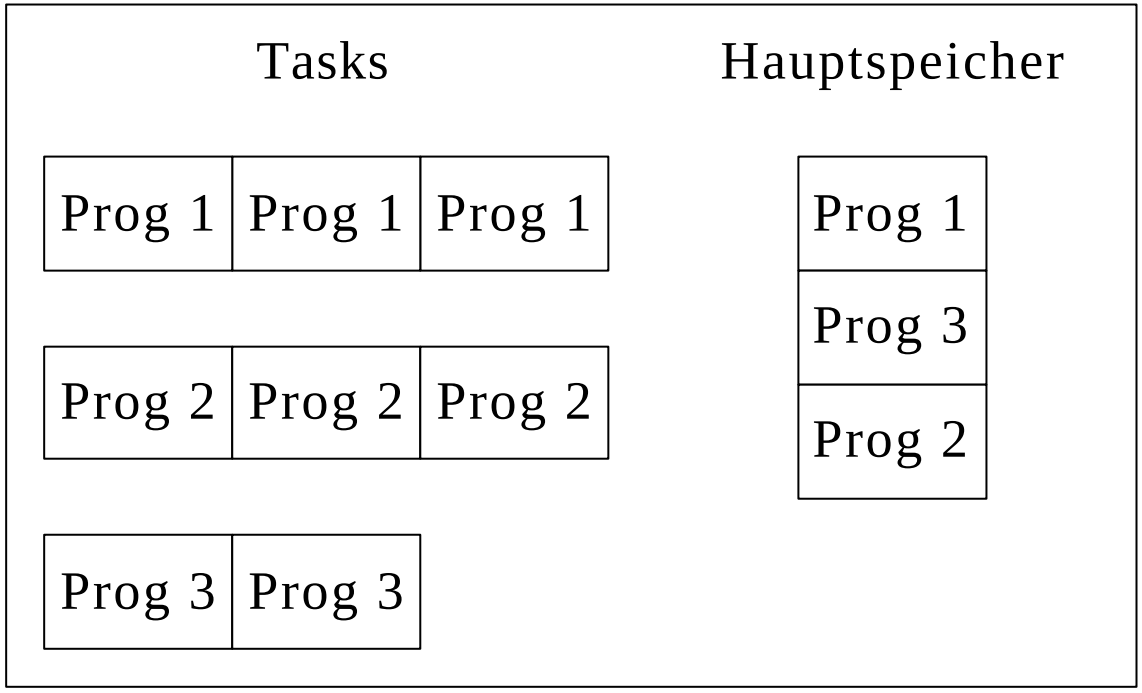
Prinzipielle Funktionsweise:
Lesezugriff
1. Lese Datum aus dem Arbeitsspeicher unter Adresse *address*
2. CPU überprüft, ob eine Kopie der Hauptspeicherzelle *address* im Cache abgelegt ist.

- Falls ja (**cache hit**)
  - so entnimmt die CPU das Datum aus dem Cache.
    Die Überprüfung und das eigentliche Lesen aus dem Cache erfolgt in einem Zyklus, ohne ein Wartezyklus einfügen zu müssen.
- Falls nein (**cache miss**),
  - so greift die CPU auf den Arbeitsspeicher zu
  - lädt den umgebenden Block des Datums in den Cache und
  - lädt das Datum von dort in die CPU.

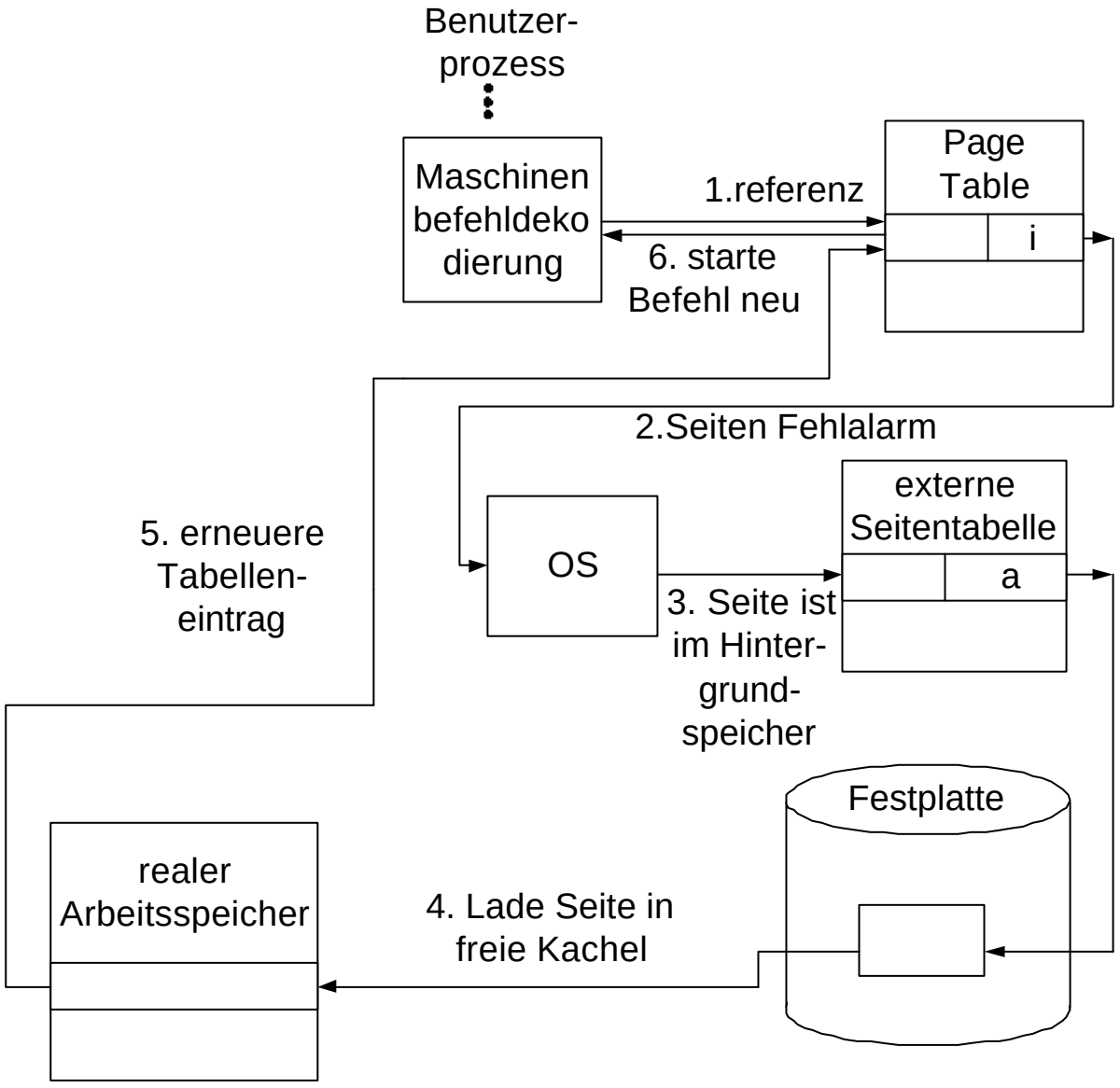
CPU überprüft, ob eine Kopie der Hauptspeicherzelle *address* im Cache abgelegt ist.

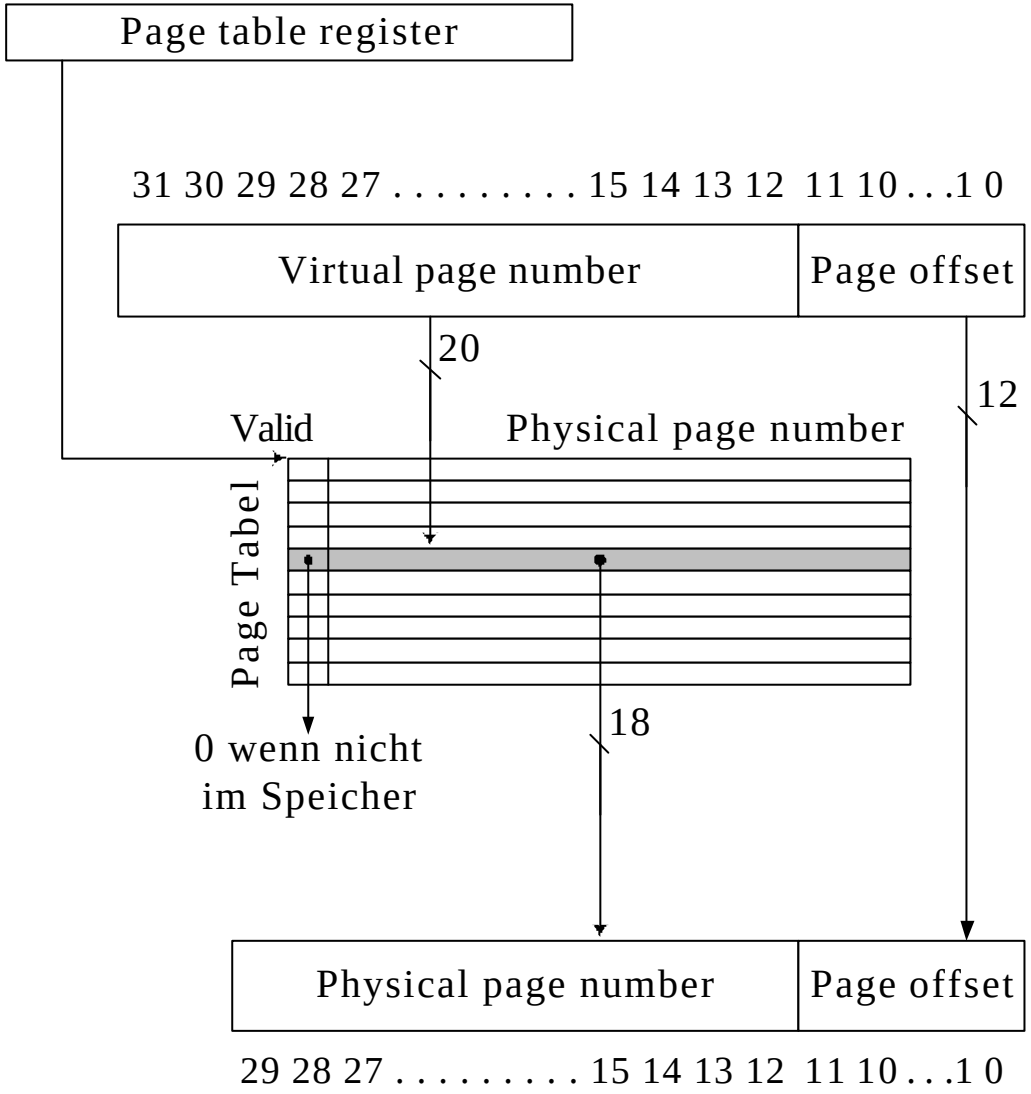
- Falls nein (**cache miss**)
  - CPU schreibt das Datum in die Hauptspeicherzelle mit Adresse *address.* Der Inhalt des Cache wird nicht verändert.
- Falls ja (**cache hit**):
  - die Kopie der Hauptspeicherzelle im Cache wird aktualisiert
  - **write-through Verfahren**: Hauptspeicherzelle wird sofort aktualisiert
  - **write-back Verfahren**: Hauptspeicherzelle wird erst später aktualisiert, nämlich wenn die Kopie aus dem Cache verdrängt wird.


Adresse (zeigt bit Position)
Hit
31 30 . . . . .17 16 15 . . . . . 5 4 3 2 1 0
16
14
Daten
Byte
offset
Valid
16 bits
32 bits
16k einträge
16
32
=


Tasks
Hauptspeicher
Prog 1
Prog 1
Prog 1
Prog 2
Prog 2
Prog 2
Prog 3
Prog 3
Prog 1
Prog 3
Prog 2

Da der virtuelle Speicher größer als der dem Prozess verfügbare Hauptspeicher sein kann, kann es vorkommen, dass eine Seite nicht im Hauptspeicher zu finden ist, d.h. sie ist auf die Festplatte ausgelagert. In diesem Fall muß die Seite von der Festplatte geholt, und in eine freie Kachel kopiert werden.




Page table register
31 30 29 28 27 . . . . . 15 14 13 12 11 10 . . .1 0
Virtual page number
Page offset
20
12
Valid
Physical page number
Page Tabel
18
0 wenn nicht
im Speicher
Physical page number
Page offset
29 28 27 . . . . . 15 14 13 12 11 10 . . .1 0

Messen und Verbessern der Cache Performance

Ansätze:

1.) Reduzierung der Fehlerrate, durch die reduzierung der Wahrscheinlichkeit, dass 2 verschiedene Memory Blöcke auf die selbe Stelle im Cache zugreifen.

2.) Reduzierung der Fehlerwartezeit, durch hinzufügen einer weiteren Ebene in Hierarchie.(multilevel Caching)

Aufteilung der CPU Zeit :

CPU time = (CPU execution clock cycles + Memory-stall clock cycles) x Clock cycle time

Memory-stall clock cycles = read-stall cycles + write-stall cycles

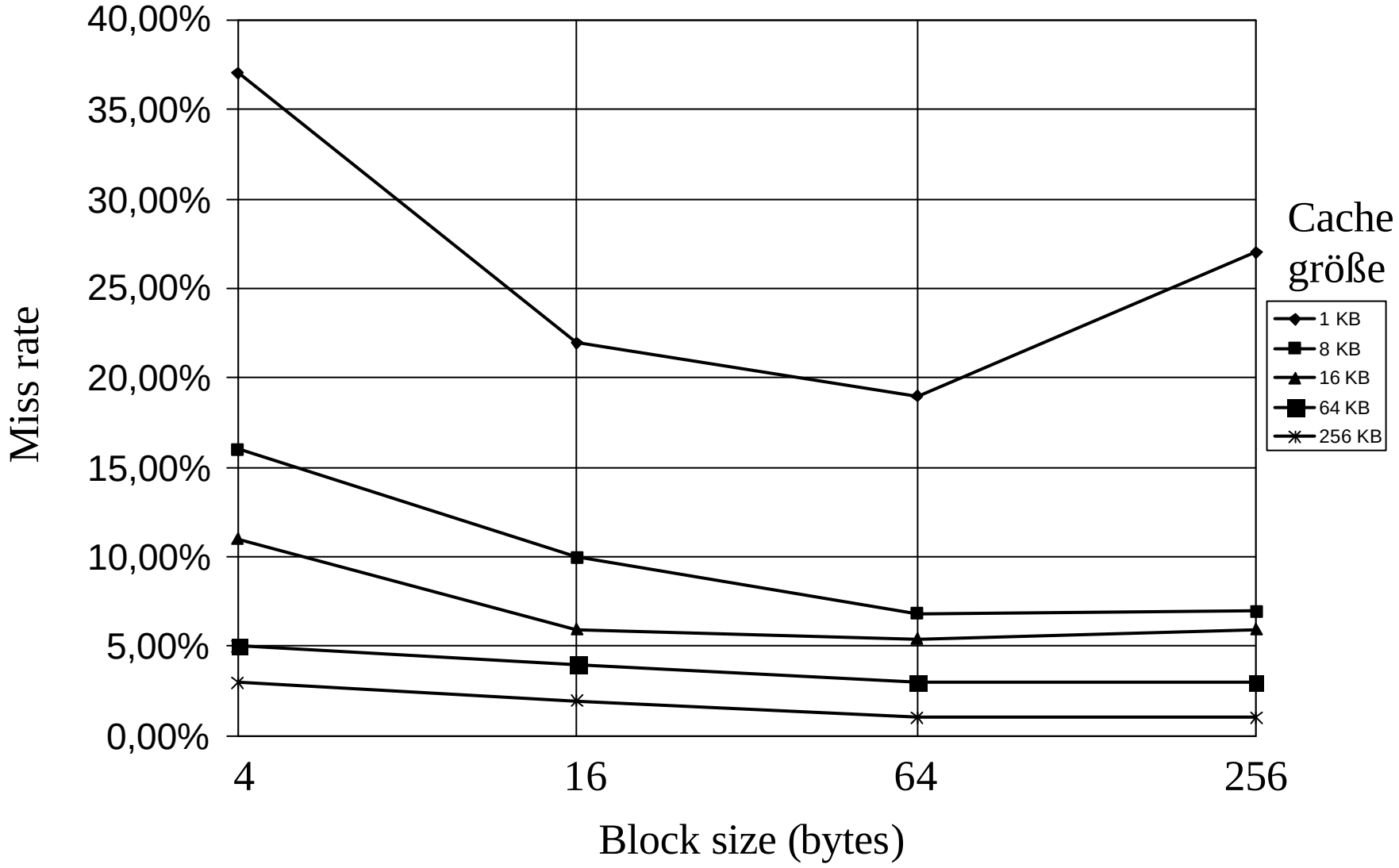
Read-stall cycles = ($\frac{READS}{PROGRAM}$ x Read miss rate x Raed miss penalty)

Write-stall cycles = ($\frac{WRITES}{PROGRAM}$ x Write miss rate x Write miss penalty) + write buffer stalls

Unter der Anahme das die Schreibpuffer Unterbrechung unwesentlich ist, können wir das Lesen und Schreiben durch das Verwenden einer einzelnen Miss-rate und Miss penalty vereiniegen

Memory-stall clock cycles = $\frac{MEMORY\ \ ACCESSES}{PROGRAM}$ x Miss rate x Miss penalty

Wir können auch schreiben:


40,00%
35,00%
30,00%
25,00%
20,00%
15,00%
10,00%
5,00%
0,00%
Miss rate
4
16
64
256
Block size (bytes)
Cache größe
1 KB
8 KB
16 KB
64 KB
256 KB

Einfach assoziativer Cache
(direct mapped)

Block	Tag	Data
0		
1		
2		
3		
4		
5		
6		
7		

Zweifach assoziativer Cache

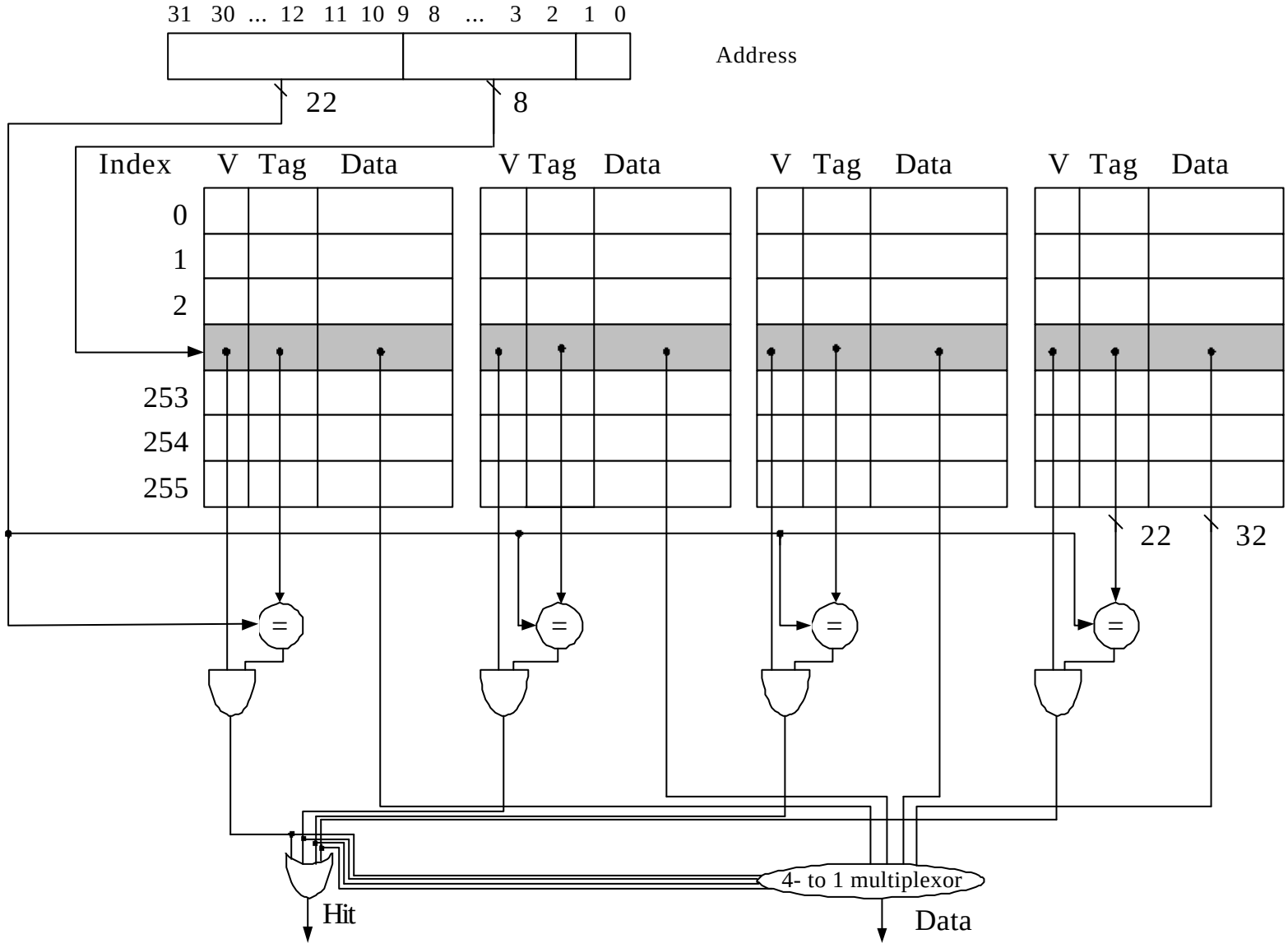
Set	Tag	Data	Tag	Data
0				
1				
2				
3				

Vierfach assoziativer Cach

Set	Tag	Data	Tag	Data	Tag	Data	Tag	Data
0								
1								

Achtfach assoziativer Cach (full associative)

Tag	Data	Tag	Data	Tag	Data	Tag	Data	Tag	Data	Tag	Data	Tag	Data	Tag	Data


31 30 ... 12 11 10 9 8 ... 3 2 1 0
Address
22
8
Index
V Tag Data
V Tag Data
V Tag Data
V Tag Data
0
1
2
253
254
255
22
32
=
=
=
=
4- to 1 multiplexor
Hit
Data

Merkmale	Power PC 604	Intel Pentium Pro
Jahr :	1997	1997
Nutzung : (Firmen)	Apple, IBM	Dell, DEC, Gateway, Hp, Intergraph, Micron
Modelle	Mac(7600,8500,9500) IBM(RS/6000)	High end Pc desktops,Server
Taktfrequenz :	100 - 250 MHz	150 - 250 MHz
2nd Level Cache :	256-KB später 512KB	256-KB oder 512-KB on die
Virtuelle Adresse :	52 Bit	32 Bit
Physikalische Adr.:	32 Bit	32 Bit
Page Size :	4KB, ausw. & 256MB	4KB, 4MB
1st Level Cache:	16 KB each	8 KB each
Block size :	32 byte	32 byte
Write police :	Write back or write through	Write back
Replacement :	LRU	LRU Annäherung

Technik	Größe	Zeit	Beschreibung
Vakuum-Röhren	20 Register	> 1950	
Quecksilber delay lines	0,5 Kbit	> 1950	Elektrische Signale konvertiert in Vibrationen werden durch eine Quecksilberröhre geleitet. Am anderen Ende wird das Signal ausgelesen und zurück in die Röhre geleitet.
Eisenkern	32 Kbit	1950	Nutzte einen Eisenkern der magnetisiert werden konnte. (wie bei Harddisk)
Dram	1Kbit	1960	Integrierte Schaltkreise (derzeitig in Gebrauch)