Speicher (4)

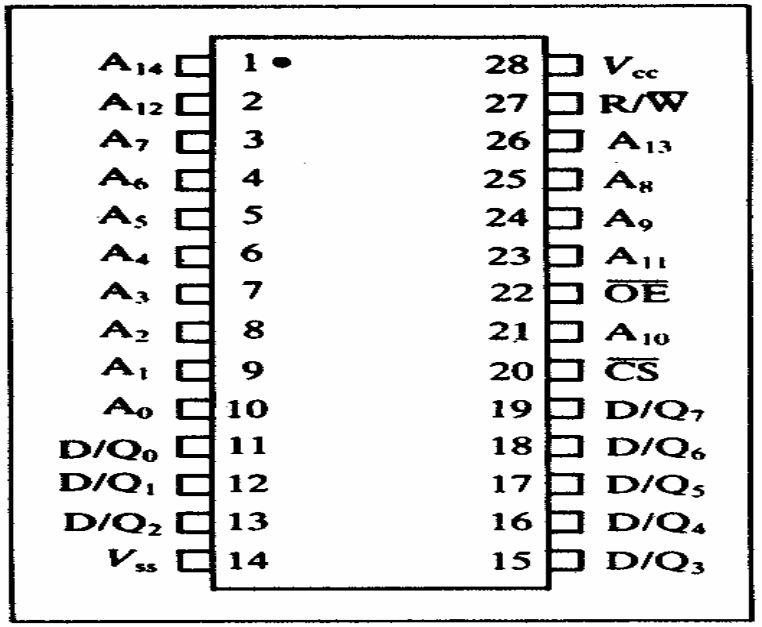
Statischer Speicher:

Eigenschaften:

- einfacherer Entwurf
- größere Zuverlässigkeit
- höhere Kosten (4 bis 6 Transistoren pro Flip-Flop)

---> Einsatz als kleiner oder mittelgroßer Speicher

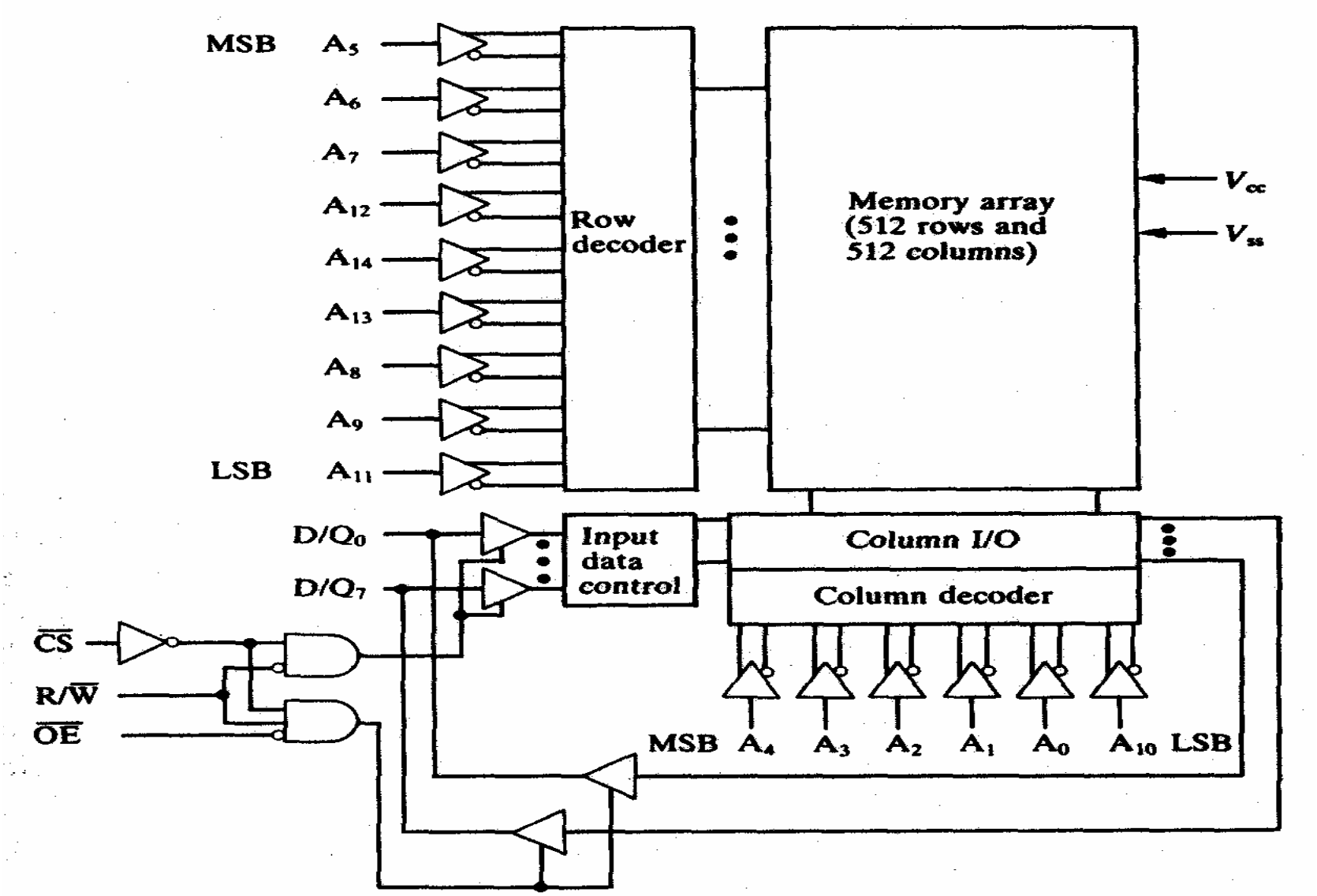
Beispiel: Der 62256 32K x 8 statische RAM



PIN NAMES	
A_0–A_{14}	Address
R/$\overline{W}$	Write Enable
$\overline{OE}$	Chip Enable
$\overline{CS}$	Output Enable
D/Q_0–D/Q_7	Data Input/Output
V_{cc}	+5 V Power Supply
V_{ss}	Ground

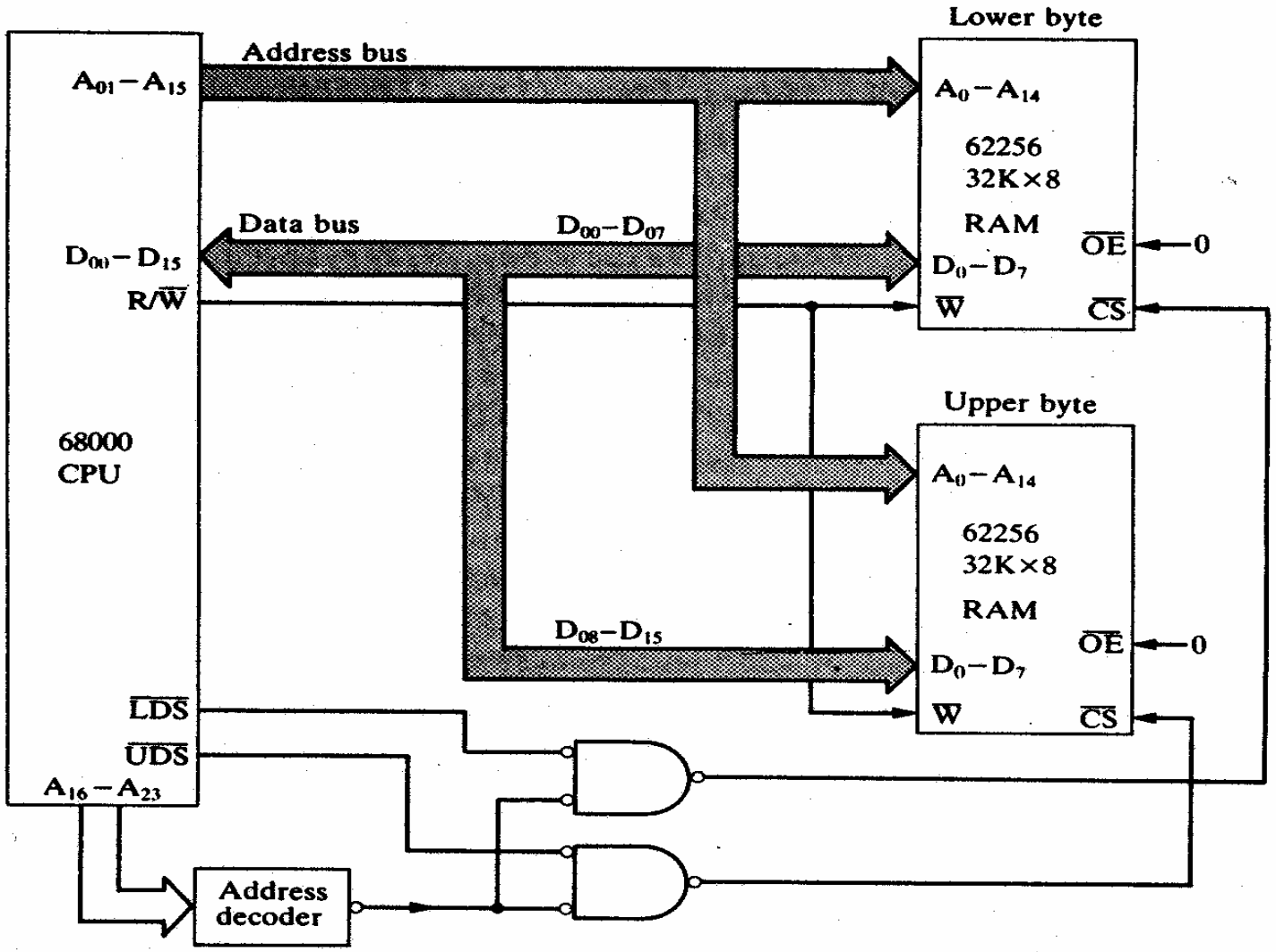
Speicher (5)

Interne Anordnung:



Speicher (6)

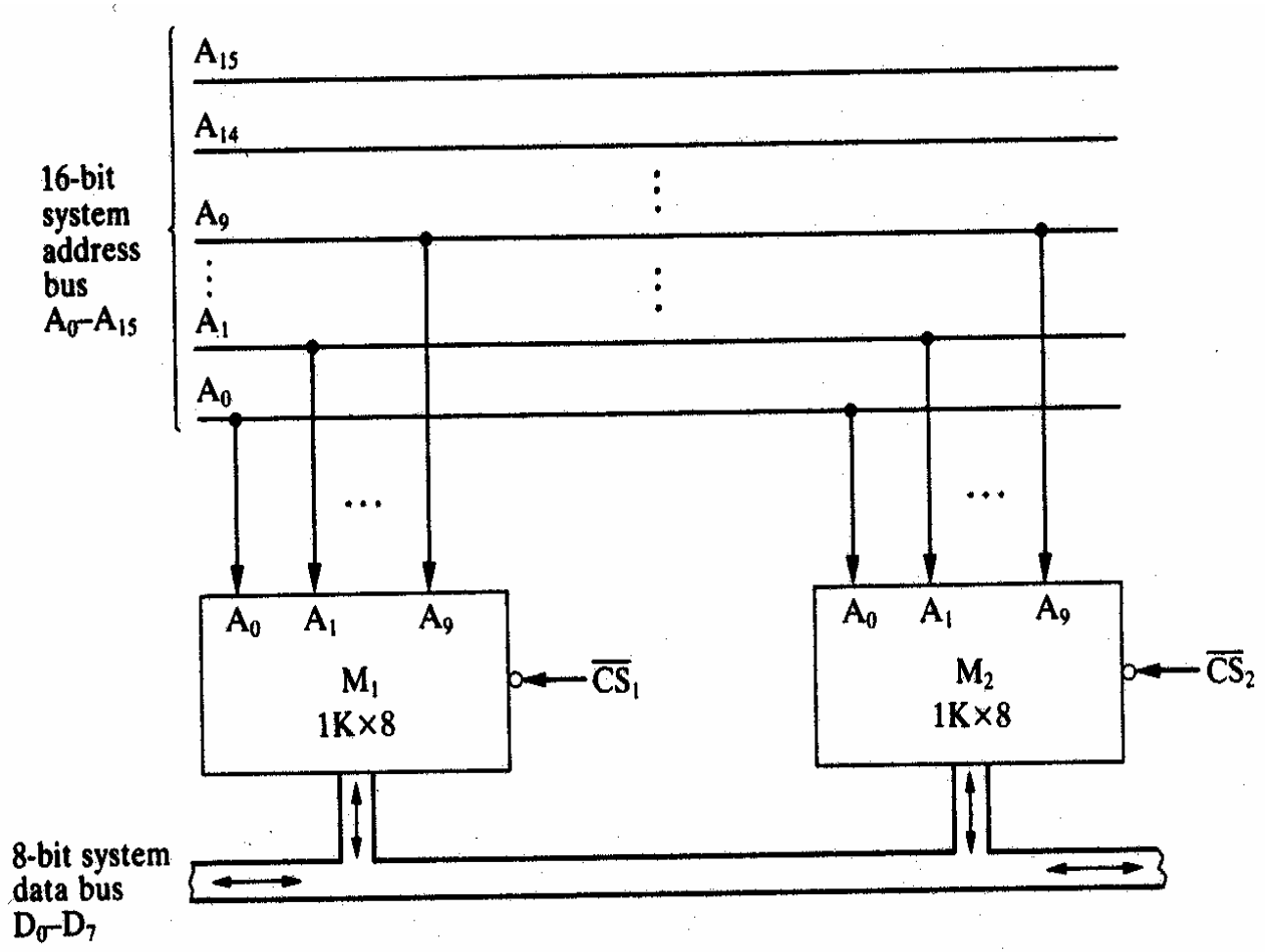
Verbindung des 62256 statischer RAM mit einer 68000 CPU:



Note: $\overline{LDS}$ selects the lower byte ($D_{00}-D_{07}$)
$\overline{UDS}$ selects the upper byte ($D_{08}-D_{15}$)

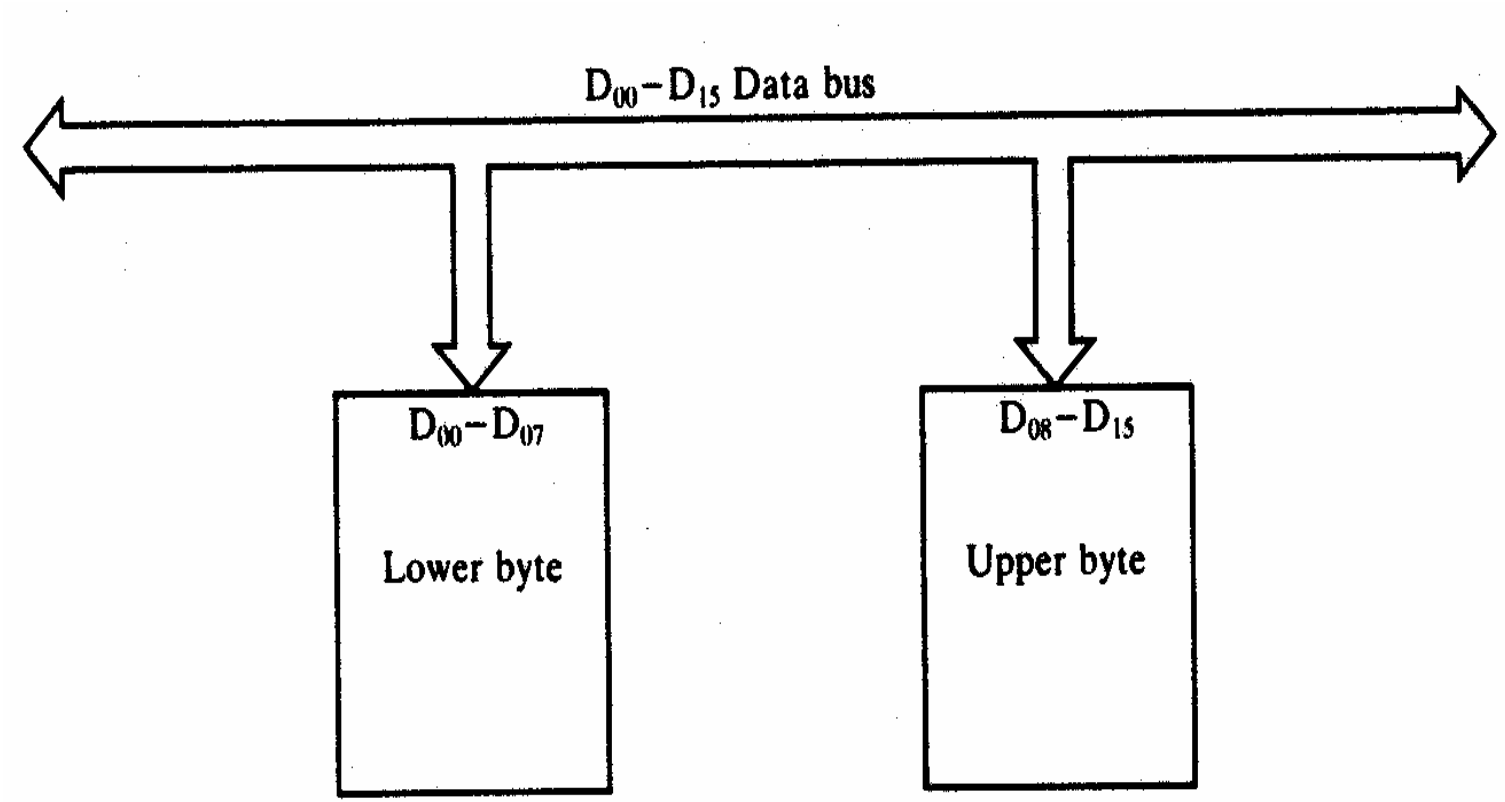
Speicher (9)

Verbindung zweier 1K x 8 - Speicher mit einem 16 Bit - Adressbus und 8 Bit - Datenbus:



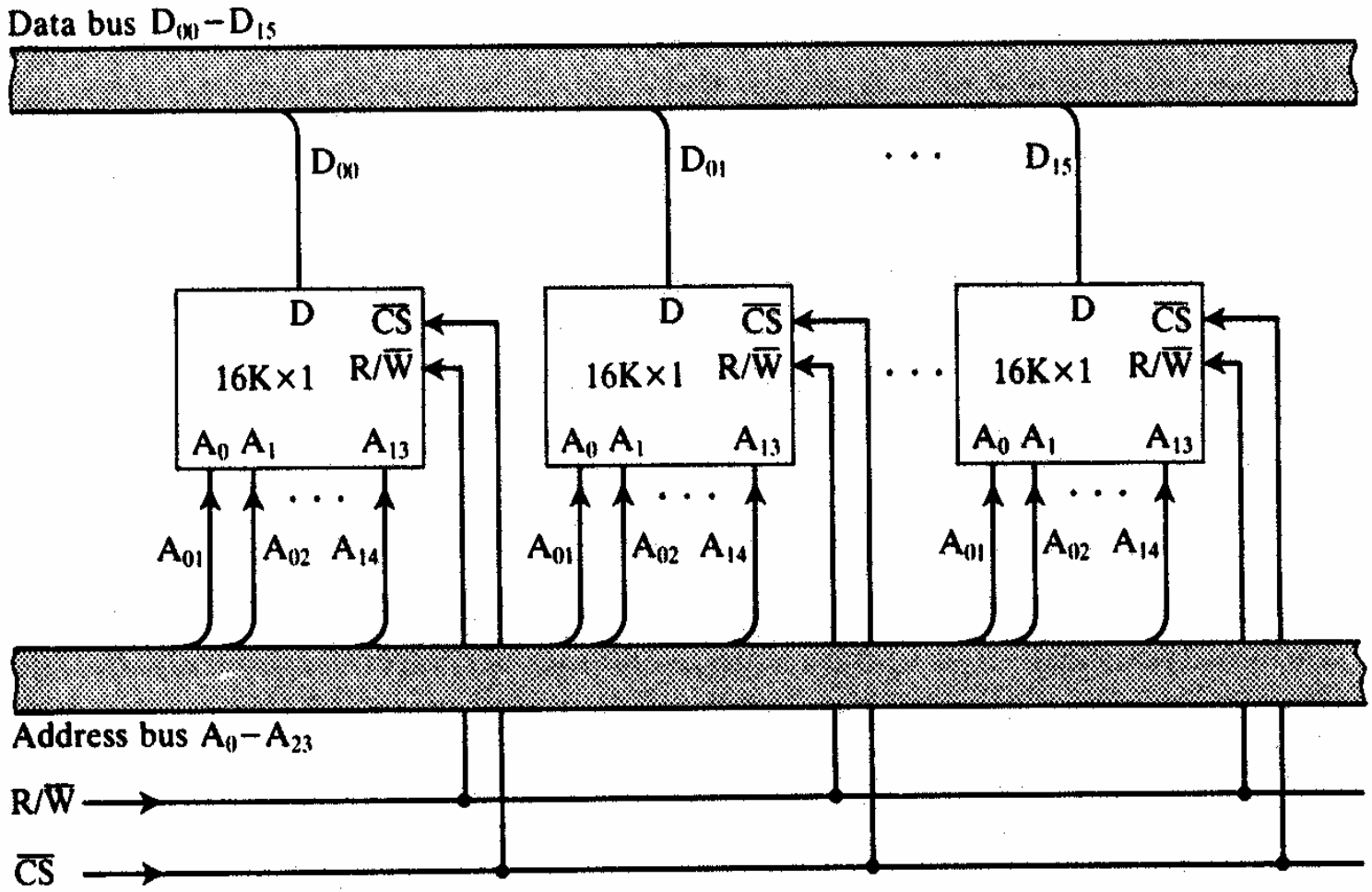
Speicher (12)

Beispiel eines Byte-organisierten Speichers:



Speicher (13)

Beispiel eines Bit-organisierten Speichers:



Speicher (14)

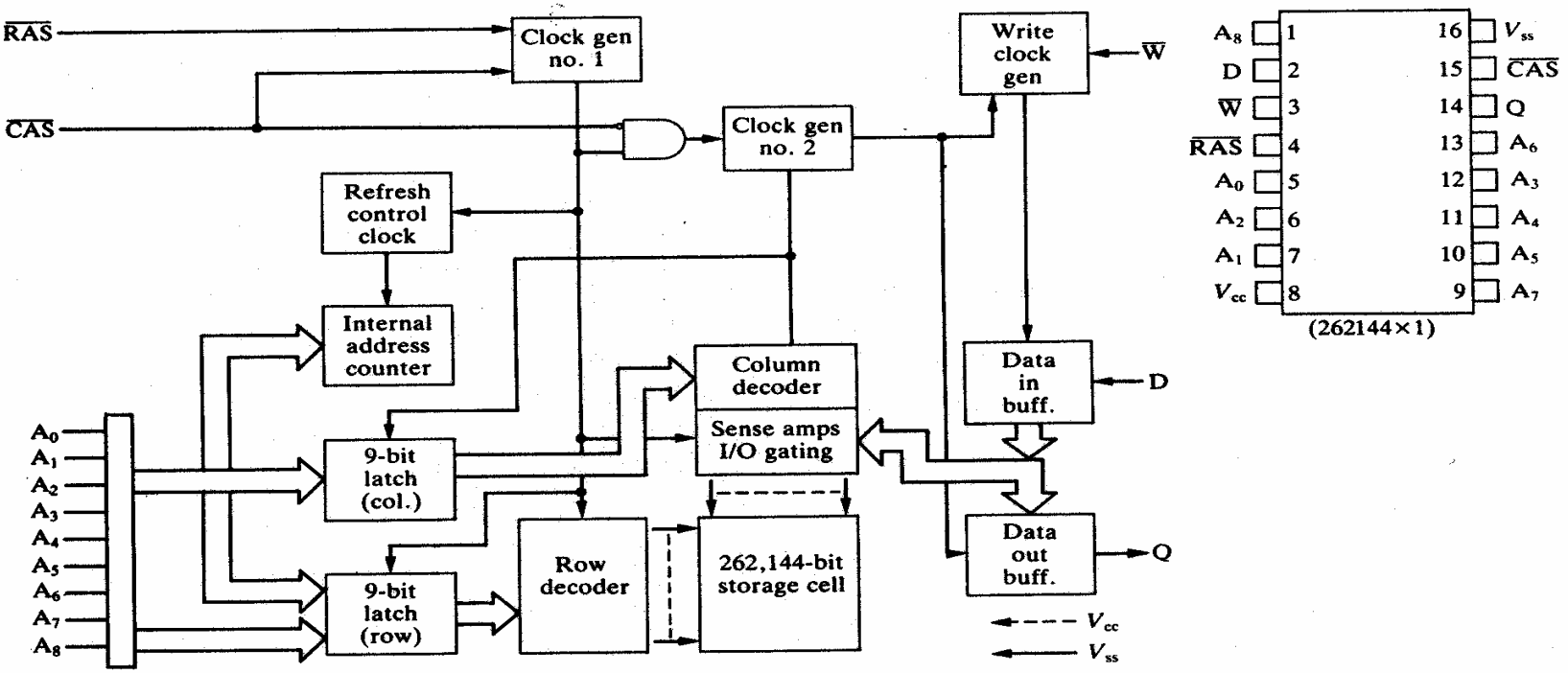
Dynamischer Speicher:

Eigenschaften:

- Höhere Dichte (1 Transistor pro Bit)
- geringere Kosten
- komplexerer Entwurf ---> geringere Zuverlässigkeit

---> Einsatz als Massen (Haupt)speicher

Interne Anordnung:



Speicher (15)

Schwächen des dynamischen Speichers:

• zusätzliche refresh - Logik

• zieht mehr Strom ---> sorgfältiges layout notwendig

• anfällig für Verunreinigung durch Alphapartikel (Heliumkerne)

- sorgfältige Qualitätskontrolle
- Anwendung fehlerkorrigierender Codes

dynamischer RAM**:** Hauptspeicher in

- Workstations
- Mainframes
- PC´s

statischer RAM:

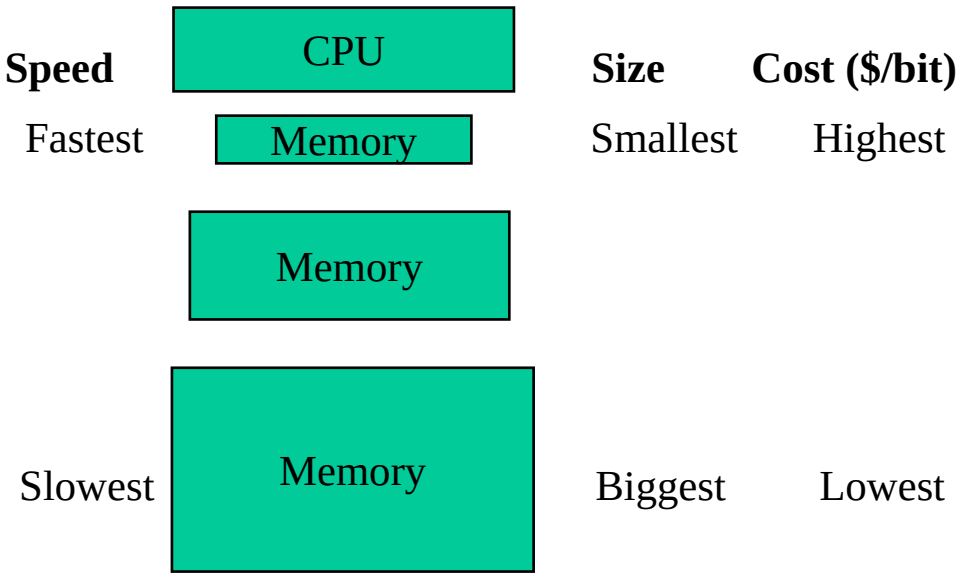
- single-board Mikrocontroller mit Speicherkapazität < 64k bytes eingesetzt in eingebetteten Systemen (z.B. Autos) bzw. als Cache
- hochzuverlässige Mikroprozessorsysteme

Speicherarchitektur (1)

Die 3 wichtigsten Speichertechnologien:

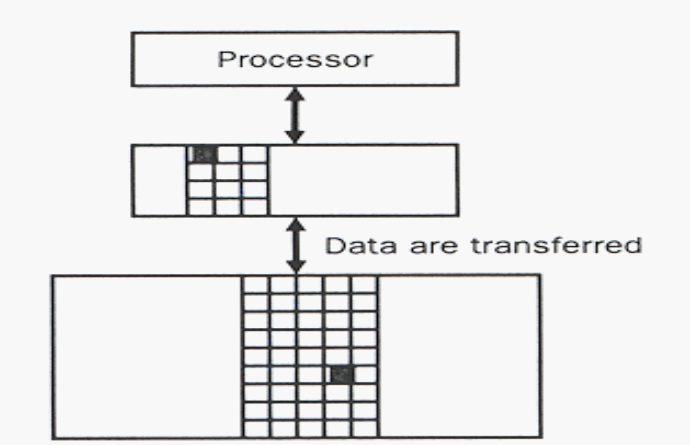
Memory technology	Typical access time	$ per MByte in 1997
SRAM	5–25 ns	$100–$250
DRAM	60–120 ns	$5–$10
Magnetic disk	10–20 million ns	$0.10–$0.20

Einsatz dieser Speichertechnologien in einer Hierarchie:

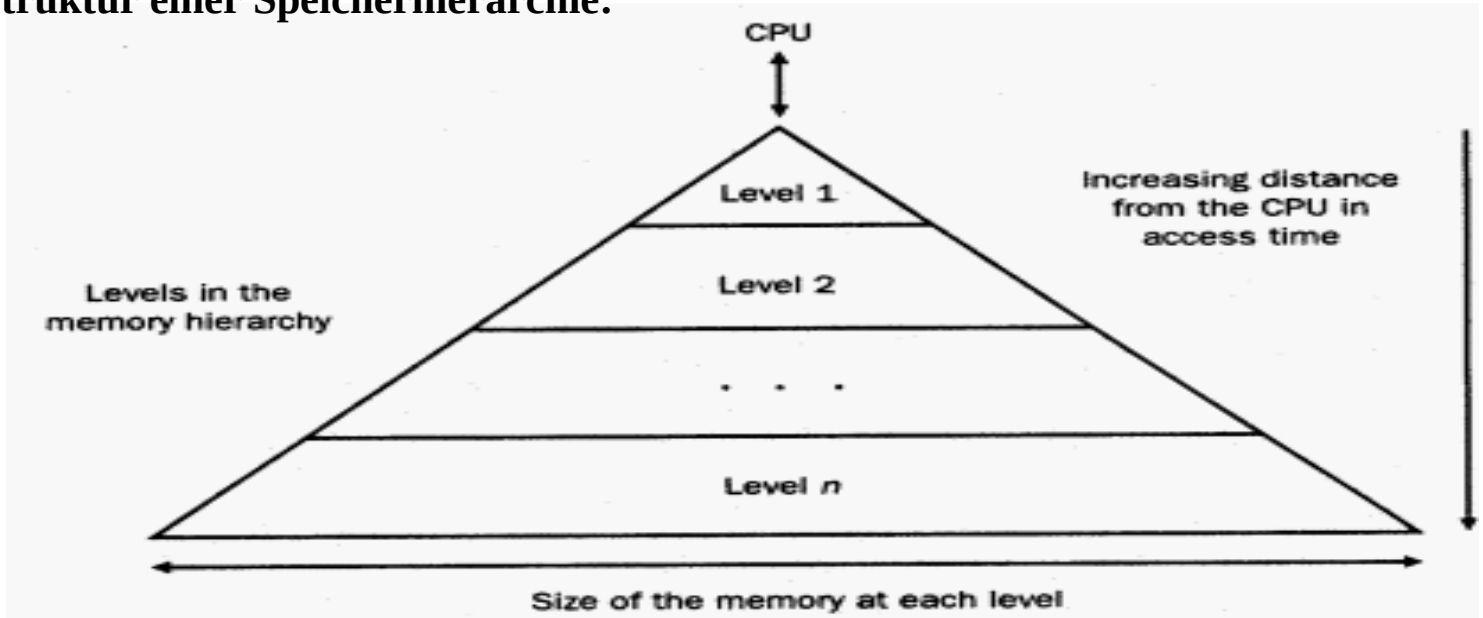


Speicherarchitektur (2)

Prinzip des Blocktransfers:



Struktur einer Speicherhierarchie:



Speicherarchitektur (3)

Ziel:

Schaffung der Illusion der Verfügbarkeit von unbegrenztem, schnellen Speicher

Feststellung:

Programme zeitigen in der Regel *Lokalität,* unterscheidbar in

•*zeitliche Lokalität*
(Tendenz zur Wiederverwendung von jüngst zugegriffenen Daten) und

•*räumliche Lokalität*
(Tendenz zur Referenzierung von benachbarten Daten)

Schlussfolgerung:

Aufbau von Speicherhierarchien, denn:

Speicherhierarchien nutzen die obigen Eigenschaften durch

•Speichern von jüngst zugegriffenen Daten „nahe“ beim Prozessor

•Datentransfer zwischen Hierarchieebenen auf der Basis von Datenblöcken

•Implementierung von kleiner und schneller Speichertechnologie auf der höchsten Hierarchieebene

Resultat:

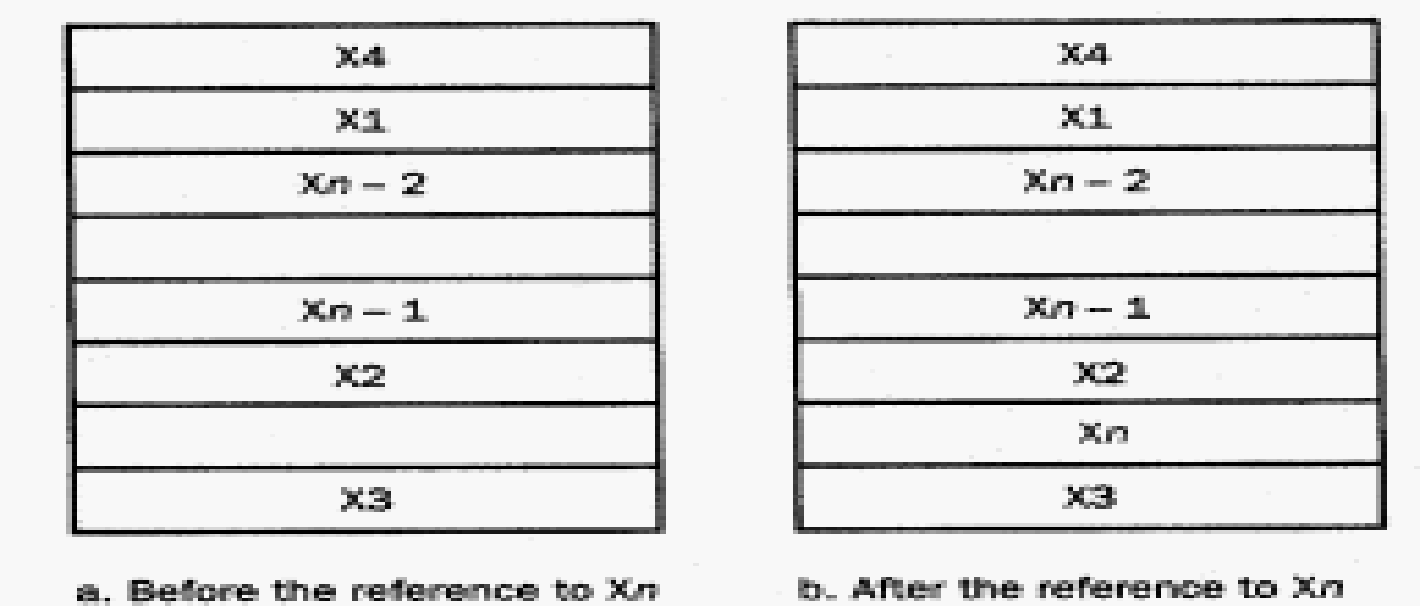
Ist die Trefferquote hoch genug --->
Speicherhierarchie hat eine effektive Zugriffszeit nahe der der höchsten (schnellsten) Ebene und eine Größe nahe der der untersten (umfangreichsten) Ebene

Speicherarchitektur (4)

C a c h e

Zwischenspeicher (d.h. zwischen CPU und Hauptspeicher gelegen) für Befehle und/oder Daten, auf den sich transparent, d.h. ohne eine eigenständige, nach außen bekannte Adresse zu haben, wesentlich schneller zugreifen lässt als auf den Hauptspeicher.

Sehr einfacher Cache vor und nach einer Referenz:



- Wie weiß man, ob das referenzierte Datum im Cache ist?
- Wenn ja, wie findet man es?

Speicherarchitektur (5)

Prinzip der direkten Zuordnung (direct-mapped)

Abbildungsvorschrift:

(Blockaddresse) modulo (Cachegröße:= Anzahl der Blöcke im Cache)

Wenn Cachegröße 2er-Potenz

----> modulo = $\log_2$ (Cachegröße) bits der Adresse

---> Man benutze einfach von der Blockaddresse die $\log_2$ niedrigsten bits.