

Digitaltechnik

Kapitel 11, Speicher

Prof. Dr.-Ing. M. Winzker

*Nutzung nur für Studierende der Hochschule Bonn-Rhein-Sieg gestattet.
(Stand: 21.03.2022)*

11.1 Übersicht

Speicherbausteine gliedern sich in folgende Kategorien

- **Flüchtige Speicher** → **Speicherung erfordert Versorgungsspannung**
 - SRAM – „Static Random Access Memory“
 - DRAM – „Dynamic Random Access Memory“
- **Nichtflüchtige Speicher** → **Speicherung auch ohne Versorgungsspannung**

Bezeichnung: NVRAM, Non-Volatile RAM

 - Elektrostatische Speicherung
 - EEPROM – Electrically Erasable Programmable Read-Only Memory
 - Innovative Speichertechniken

Manche Begriffe sind „historisch“ und die „Alternativen“ nicht direkt vergleichbar

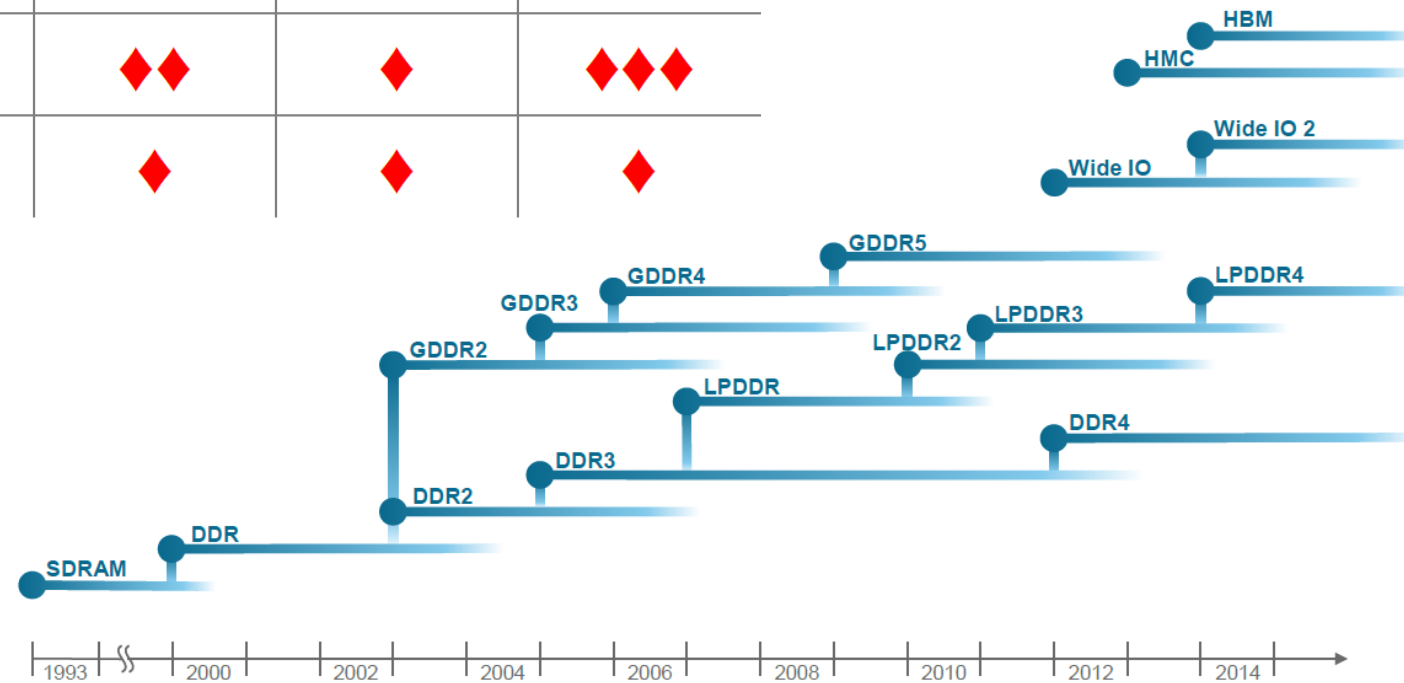
- Random Access Memory – Es gibt auf FPGA/ASIC FIFO-Speicher, bei denen Daten in gleicher Reihenfolge geschrieben und gelesen werden (First In First Out)
- Read-Only Memory – In FPGAs/ASICs wird der Inhalt von ROMs bei der Konfiguration bzw. Fertigung festgelegt

Anforderungen an Speicher

- Steigende Anforderungen erfordern neue Speichertechnologien und immer neue Interfaces

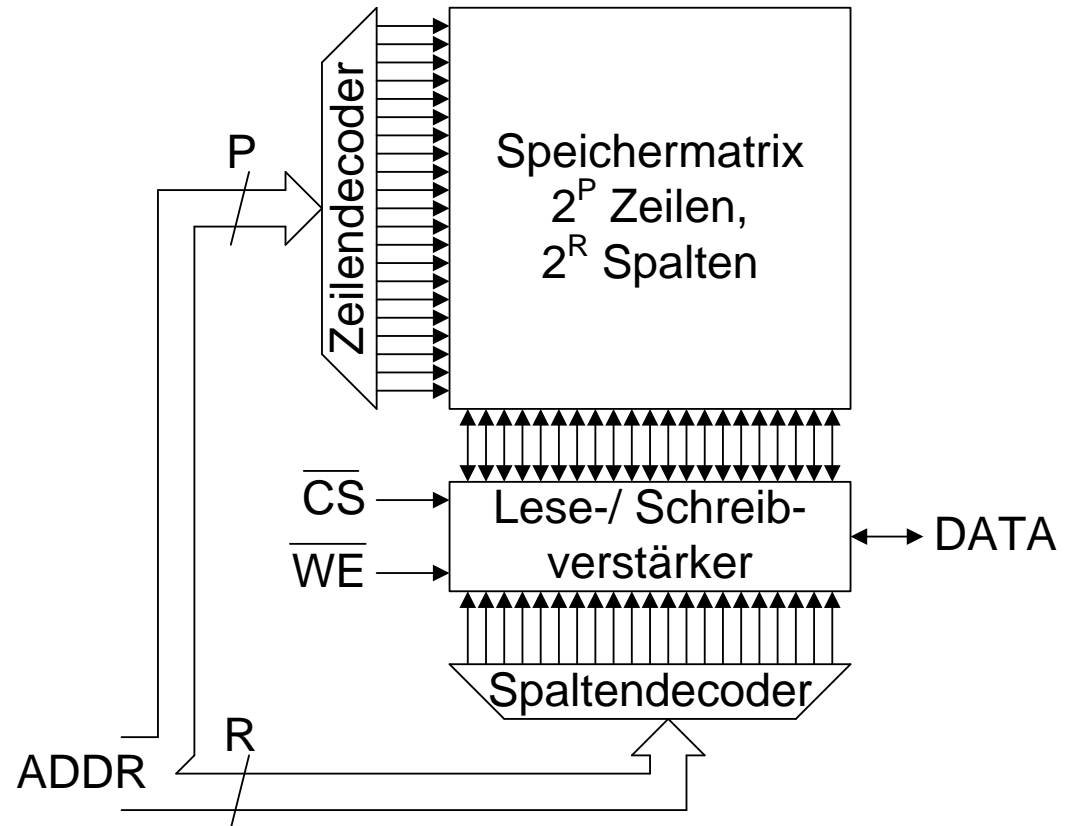
	Bandwidth	Power	Cost	Capacity
Mobile	♦♦	♦♦♦♦	♦♦	♦♦
Consumer	♦	♦♦	♦♦♦♦	♦
PC/Server	♦♦♦♦	♦♦	♦	♦♦♦♦
Networking	♦♦♦♦	♦	♦	♦

Quelle: M. Lund, „Memory: A System Optimization Challenge,“
Cadence MemCon 2013.



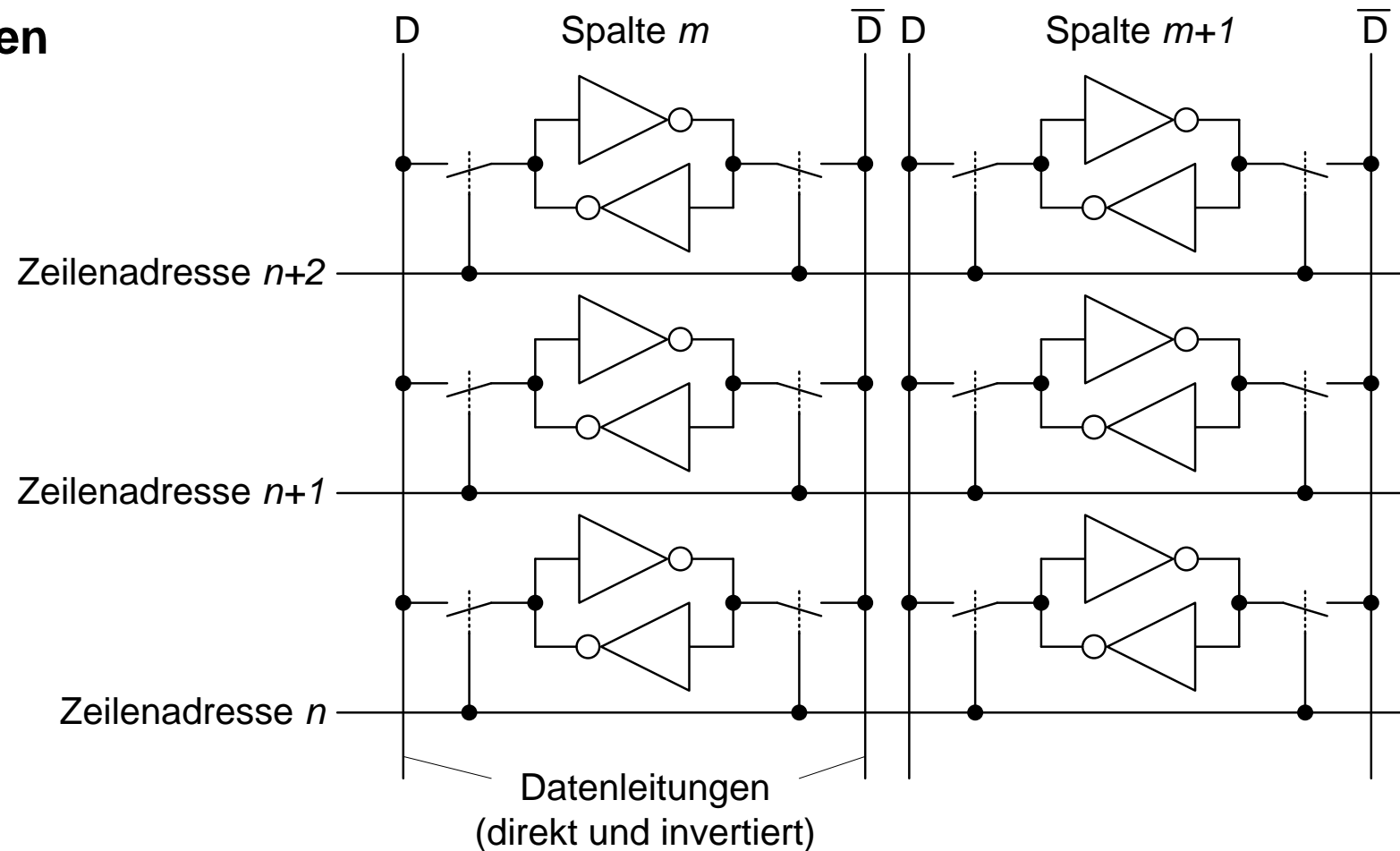
11.2 Speichertechnologien

- Daten werden generell in einer Matrix aus Speicherzellen gespeichert
- Die Adresse wird in Zeilen- und Spaltenadresse aufgeteilt
 - P Adressleitungen für 2^P Zeilen
 - R Adressleitungen für 2^R Spalten
 - P und R sind etwa gleich groß
- Diese Adressaufteilung erleichtert den internen Aufbau
 - Zwei Decoder mit etwa halber Anzahl an Adressen einfacher als ein großer Decoder
 - Die Speichermatrix ist etwa quadratisch und ergibt ein günstiges IC-Layout
- Zeile und Spalte können gleichzeitig oder nacheinander anliegen
 - Je nach Speichertyp



11.2.1 SRAM

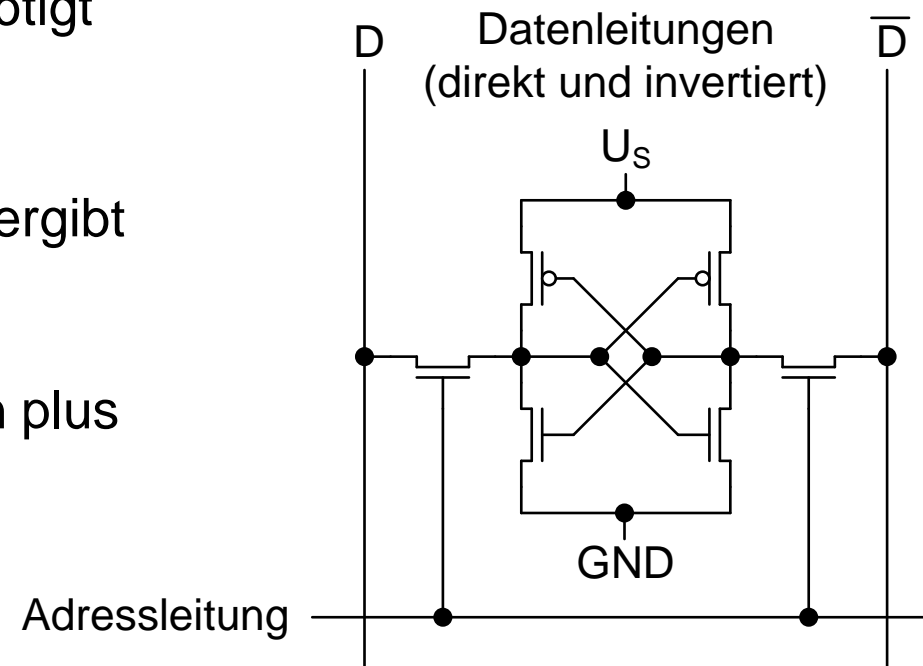
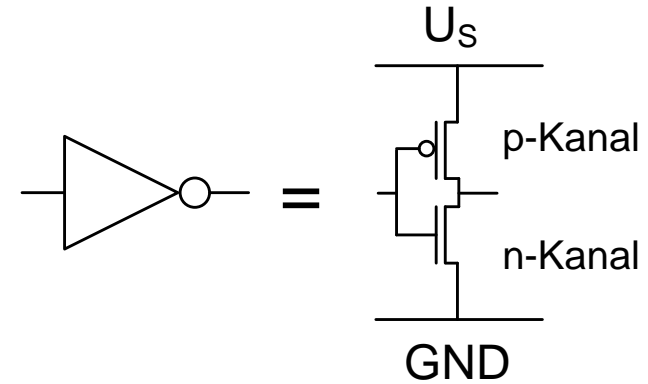
- SRAM Speicherzellen bestehen (prinzipiell) aus zwei rückgekoppelten Invertern
- Der rückgekoppelte Wert kann durch dieselben Datenleitungen:
 - **gelesen** oder
 - **überschrieben**werden



Aufbau der Speicherzellen (II)

Als Technologie für SRAM-Bausteine wird CMOS verwendet

- Ein Inverter besteht aus je zwei Transistoren
- Die Schalter können durch je einen Transistor realisiert werden
- Je Speicherzelle werden **6 Transistoren** benötigt
 - Diese Schaltungsstruktur wird als „**6-Transistor-Zelle**“ bezeichnet
- Die Anzahl an Transistoren eines SRAM-ICs ergibt sich als Produkt aus Speicherkapazität und Transistoren pro Speicherzelle
 - Ein 8 Mbit-SRAM hat 48 Mio. Transistoren plus Ansteuerung (z.B. Adressdecoder)

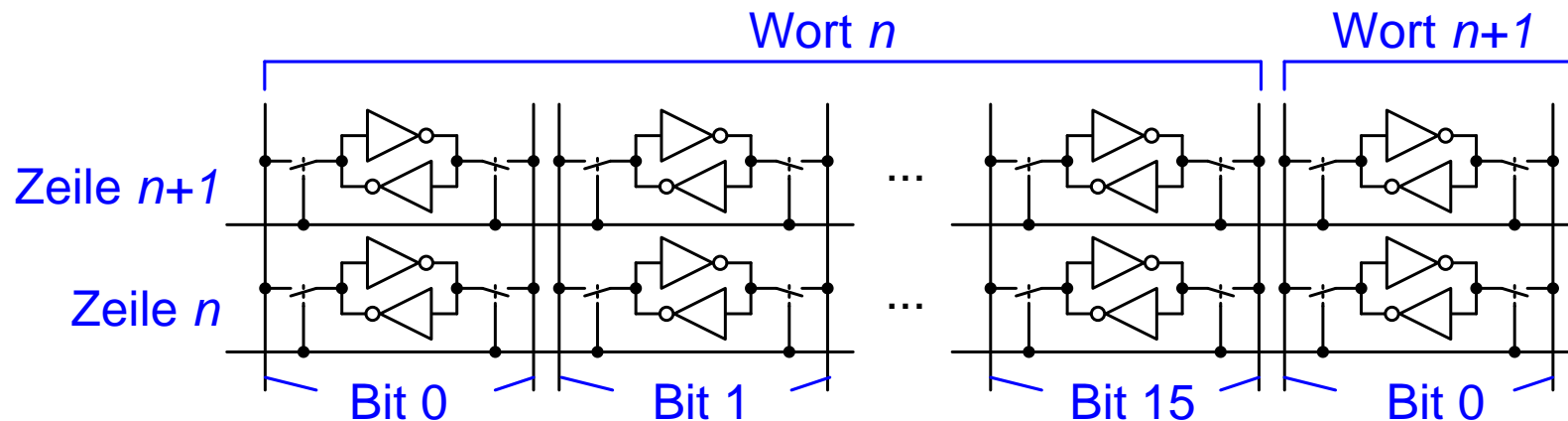


Adressierung und Row/Column-Struktur

- Adressierung und Struktur berechnet sich wieder mit Zweierpotenzen

Beispiel: Speicher mit 2^{20} Datenworten zu 16 bit

- 1.048.576 Datenworte (rund 1 Million) mit 16 Millionen Speicherzellen
- Speichermatrix aus 4096 Zeilen und 4096 Spalten
- 16 Zellen einer Zeile bilden ein Datenwort und haben die gleiche Adresse
- Darum müssen 4096 Zeilen und $4096 / 16 = 256$ Spalten angesteuert werden
 - ➔ 4096 Zeilen = 12 Bit Adresse
 - ➔ 256 Spalten = 8 Bit Adresse
 - ➔ Insgesamt 20 Bit Adresse, entsprechend 2^{20} Datenworten



Exkurs: Auslegung einer SRAM-Zelle

Drei Ansteuerungsarten

1) Datenspeicherung

- $WL = 0$; $BL = \text{beliebig}$

2) Schreiben in SRAM-Zelle

- $WL = 1$; $BL = \text{neue Daten}$

3) Lesen aus SRAM-Zelle

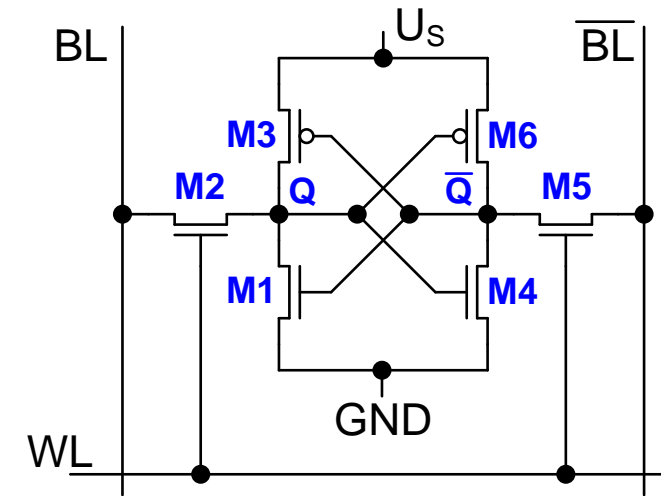
- $WL = 1$; $BL = \text{hochohmig}$

Problem

- Die Rückkopplungen und die Datenleitungen arbeiten in zwei Fällen gegeneinander
 - Beim Schreiben muss BL stärker als die Rückkopplung durch $M1, M3$ sein
 - Beim Lesen hingegen darf eine Ladung auf BL die Rückkopplung $M1, M3$ nicht überschreiben
- Gleiches gilt für \overline{BL} und $M4, M6$

BL = Bit-Line (Datenleitungen)

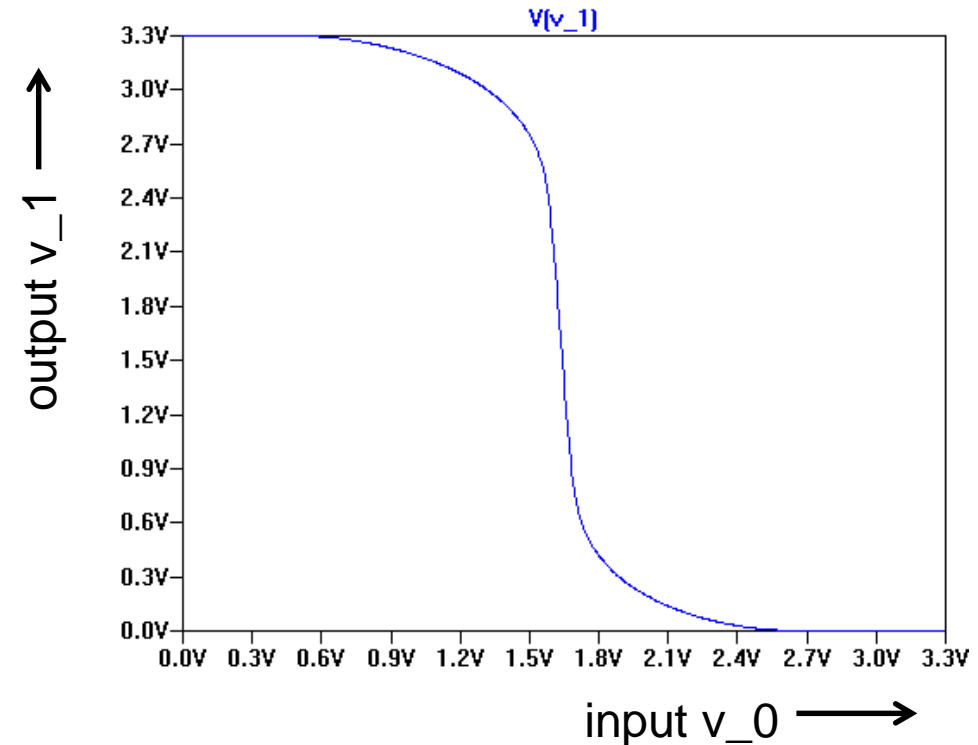
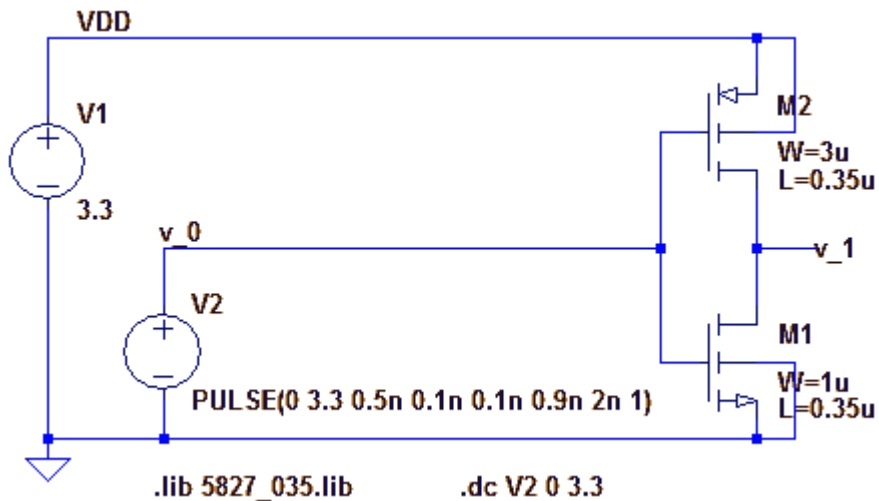
WL = Write-Line (Adressleitung)



Verhalten der SRAM-Zelle

(Screenshots aus LTspice
Farben angepasst)

- Inverter-Kennlinie kann durch LTSpice bestimmt werden
- DC Sweep der Eingangsspannungsquelle V2 von 0V bis 3.3V
 - Kommando: `.dc V2 0 3.3`
 - Pulse-Parameter wird dann ignoriert
- Deutliche (negative) Steigung sorgt für Unempfindlichkeit gegen Störungen

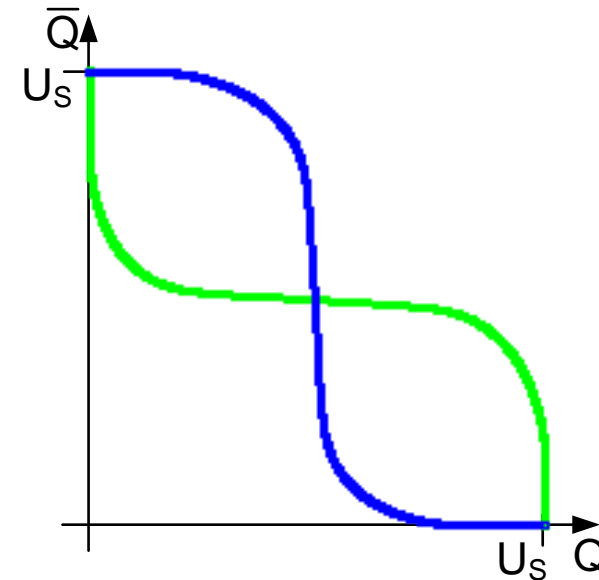
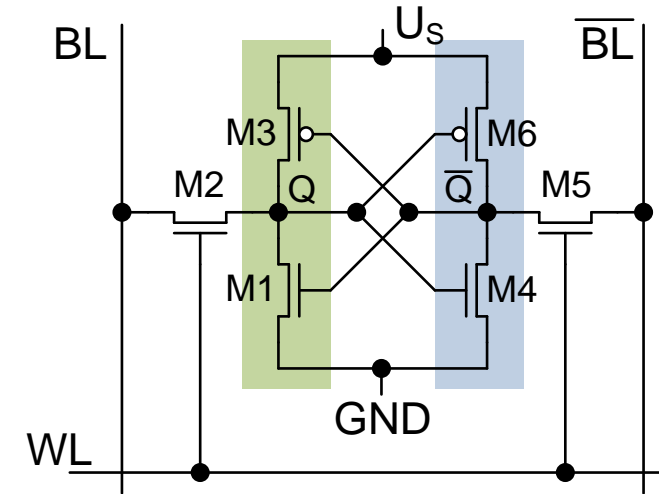


Verhalten der SRAM-Zelle (II)

- Rückgekoppelte Inverter des SRAM stabilisieren sich gegenseitig
 - Inverter Q nach \bar{Q} in **blau**
 - Inverter \bar{Q} nach Q in **grün**
- Rückkopplung hat zwei stabile Zustände
 - Zwei Kombinationen mit U_S und 0V
 - Dies sind die beiden Zustände für Speicherwerte 0 und 1
- Kennlinienkreuzung bei $U_S/2$ ist instabil

Darstellungsweise wird häufig benutzt

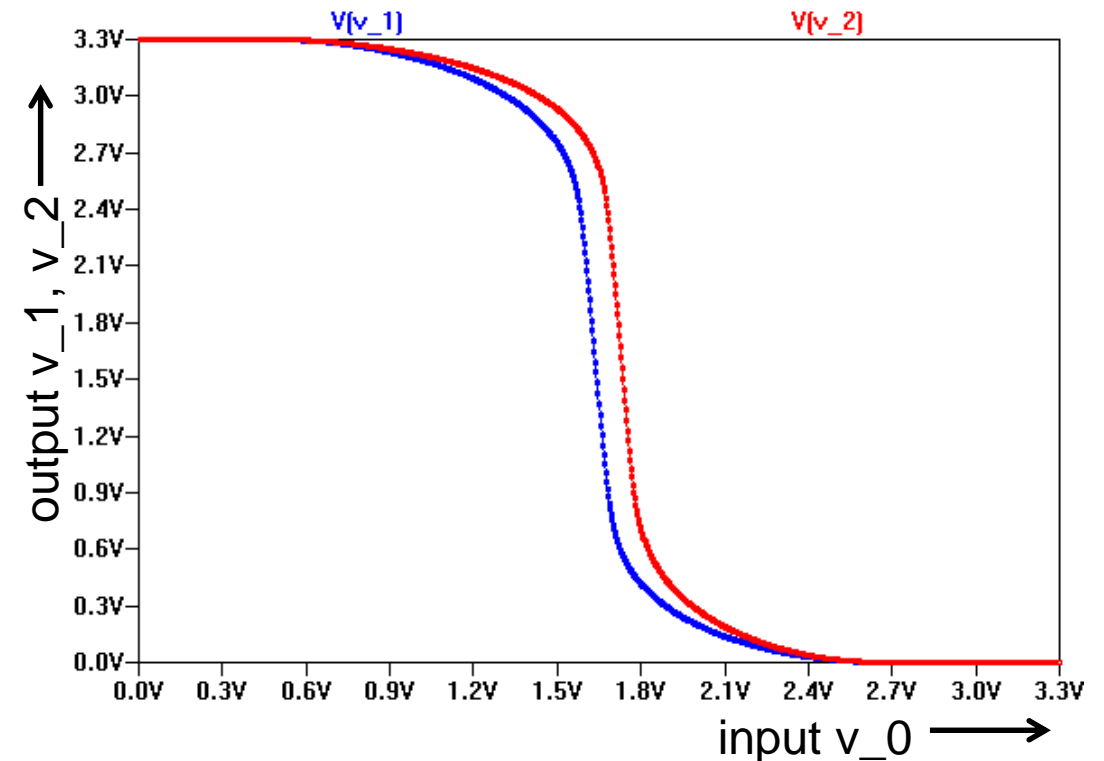
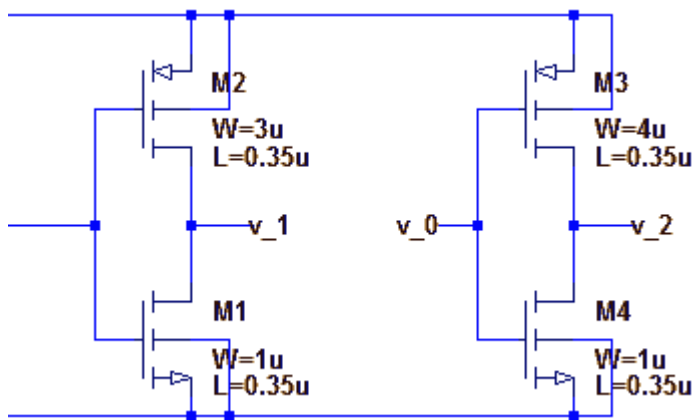
- Bezeichnung: **Butterfly-Curve**



Variationen der SRAM-Zelle

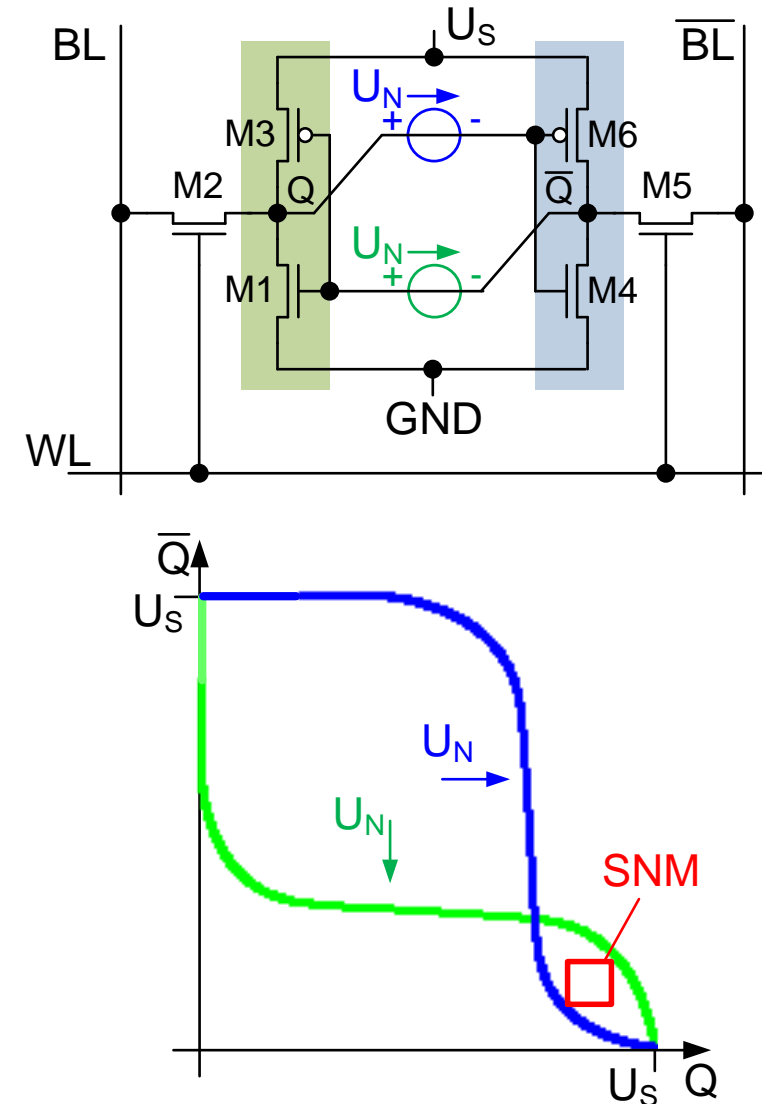
(Screenshots aus LTspice
Farben angepasst)

- Die ideale Kennlinie wird durch Variationen von Prozess (Dotierungen und Abmessungen), Spannungen und Temperatur verändert
- Einfluss kann mit LTSpice ausprobiert werden
 - ➔ P-Kanal-Transistor wird auf $W=4\mu$ verändert
 - Entspricht Variation der Abmessungen und/oder der Beweglichkeit der Löcher
 - Vergleiche Folie 13
- Kurve verschiebt sich von **blau** nach **rot**



Variationen der SRAM-Zelle (II)

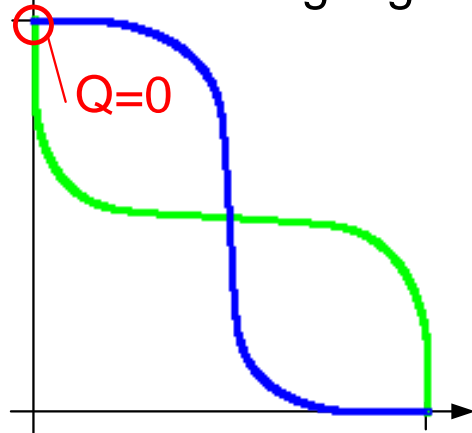
- Die Stabilität einer SRAM-Zelle wird durch den „Static Noise Margin“ (SNM) beschrieben
- Variationen werden durch Verschiebung der Kennlinie modelliert
 - In beide Rückkopplungen wird eine **Spannungsquelle U_N** eingefügt
 - Zur Einordnung: Bei Low-Power-Anwendungen kann U_S weniger als 1 V sein
- Das verbleibende SNM-Fenster gibt die verbleibende Störsicherheit an
 - Kritisch sind Variationen, die nur einen Transistortyp (p, n) betreffen



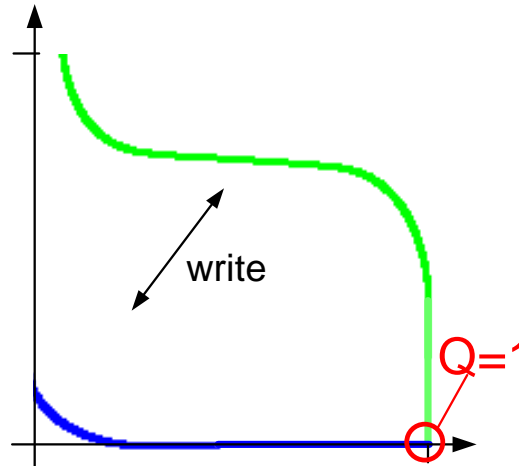
Schreiben in eine SRAM-Zelle

- Zum Schreiben eines Wertes wird an die Bitleitungen BL und \overline{BL} der neue Wert angelegt und WL aktiviert
 - Die Bitleitung versucht dann die Rückkopplung zu überwinden
 - In der Butterfly-Curve entspricht dies einem Verschieben der beiden Kurven
- Nur wenn die Schreibleitung die Kurven ausreichend verschiebt, erfolgt der Schreibvorgang

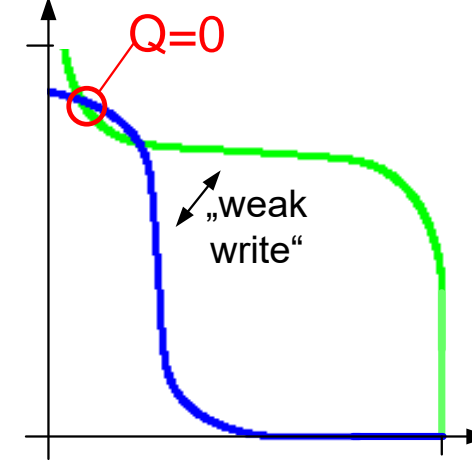
Datenspeicherung vor Schreibvorgang



Schreiben von Q=1



Fehler beim Schreiben



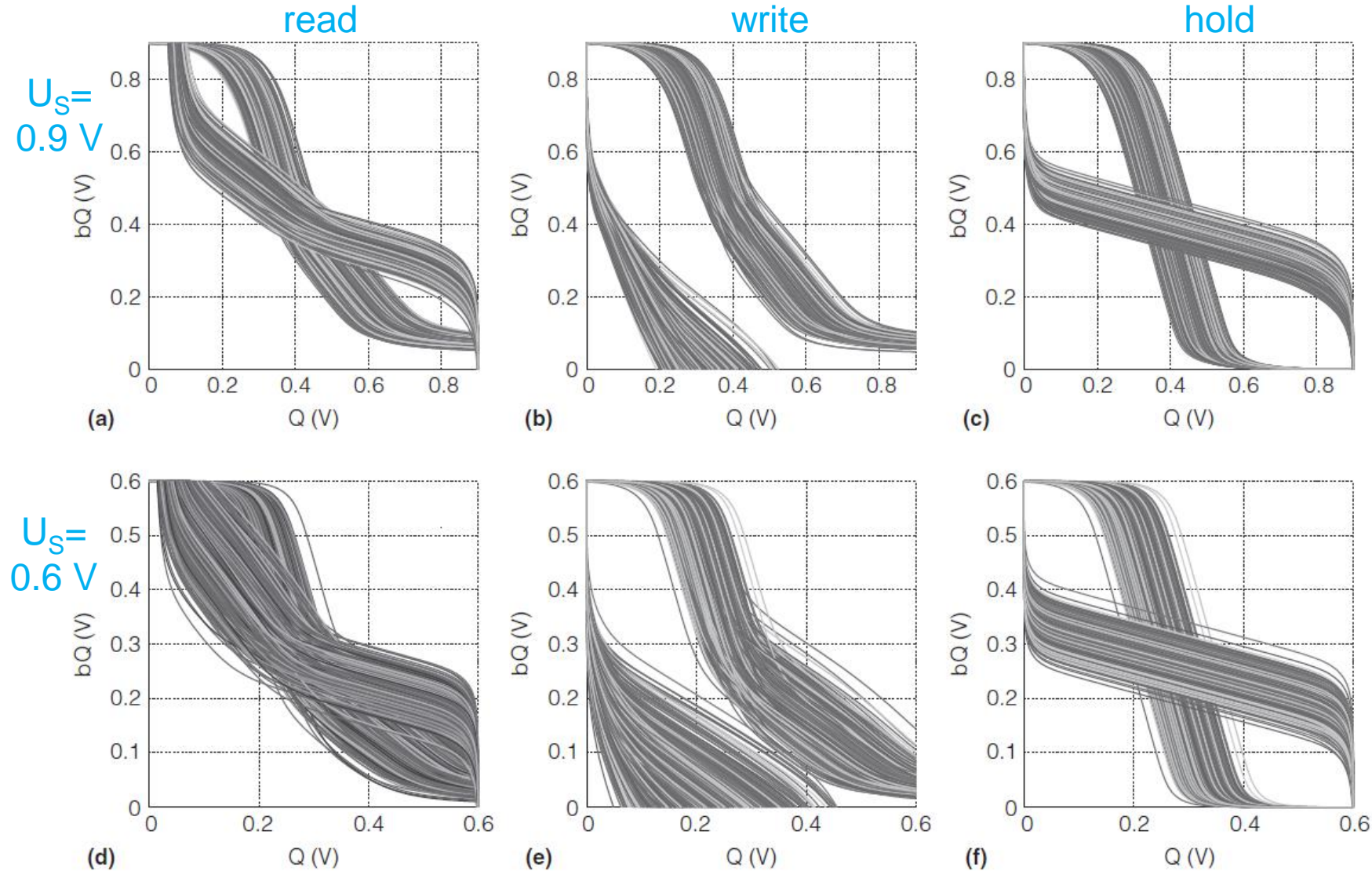
Schreiben in eine SRAM-Zelle (II)

- Die drei SRAM-Operationen Read, Write, Hold müssen für alle Variationen im Betrieb funktionieren
- Diese Variationen werden als PVT bezeichnet
 - Process (Dotierungen und Abmessungen)
 - Voltage
 - Temperature
- Realistische Angaben für einen 32nm Prozess finden sich in:
M. Qazi, M. E. Sinangil, A. P. Chandrakasan, „Challenges and Directions for Low-Voltage SRAM,“ IEEE Design and Test of Computers, pp. 32-41, Jan/Feb 2011.
- Diagramme auf Folgefolie
 - Dargestellt sind die drei SRAM-Operationen für die Spannungen von 0,9V und 0,6 V

Aufgabe

- Interpretieren Sie die Ergebnisse

Schreiben in eine SRAM-Zelle (III)

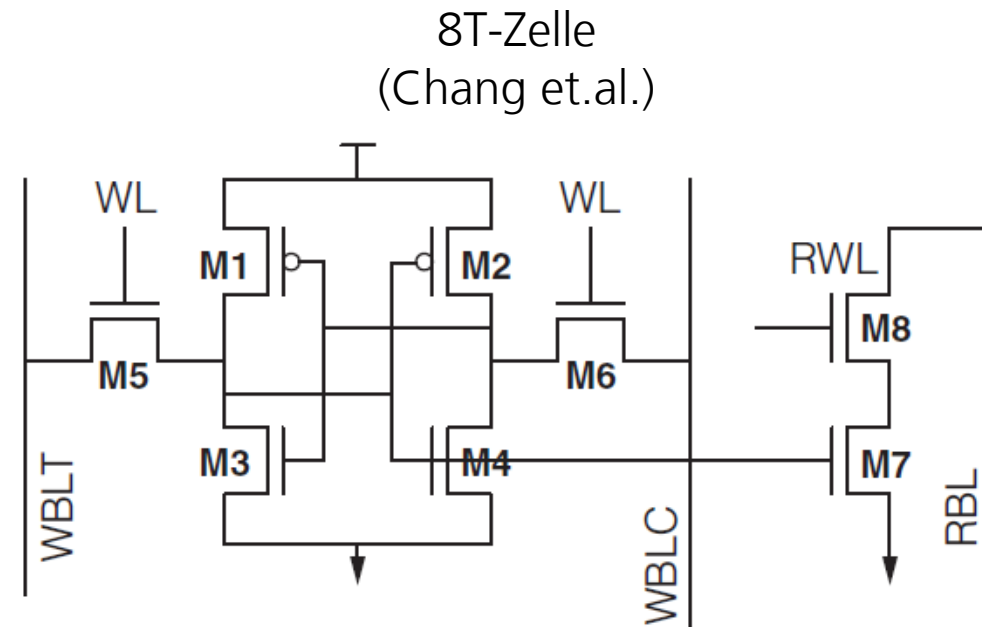
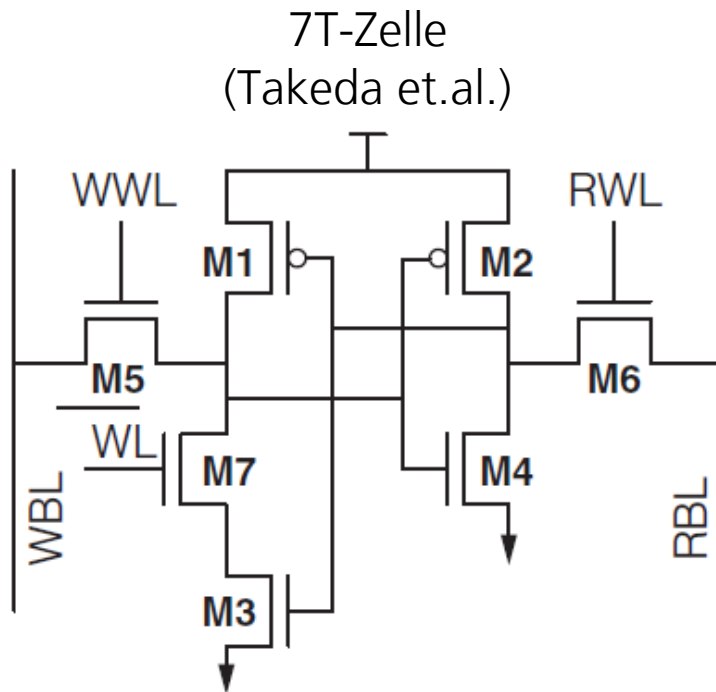


Quelle: M. Qazi, M. E. Sinangil, A. P. Chandrakasan, „Challenges and Directions for Low-Voltage SRAM,“ IEEE Design and Test of Computers, pp. 32-41, Jan/Feb 2011.

Alternative SRAM-Zellen

Aufgabe

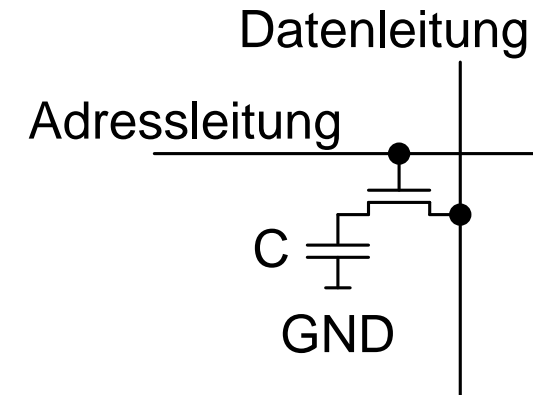
- In Qazi, et.al. werden mehrere alternative SRAM-Zellen vorgestellt, die allerdings mehr als 6 Transistoren benötigen.
- Analysieren Sie die beiden dargestellten Schaltungen. Wie ist die Funktion?
 - Tipp: Achten Sie auf die Abkürzungen an den Leitungen.



Quelle: M. Qazi, M. E. Sinangil, A. P. Chandrakasan, „Challenges and Directions for Low-Voltage SRAM,” IEEE Design and Test of Computers, pp. 32-41, Jan/Feb 2011.

11.2.2 DRAM

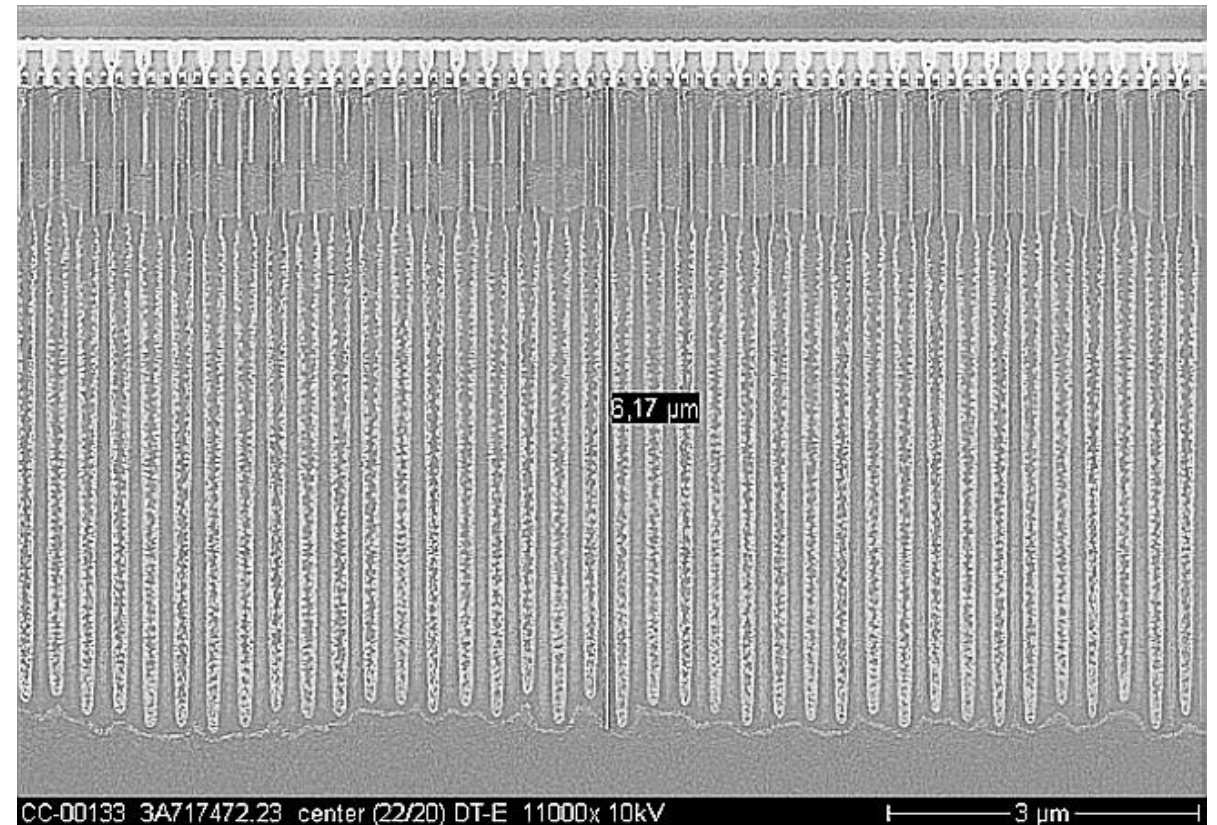
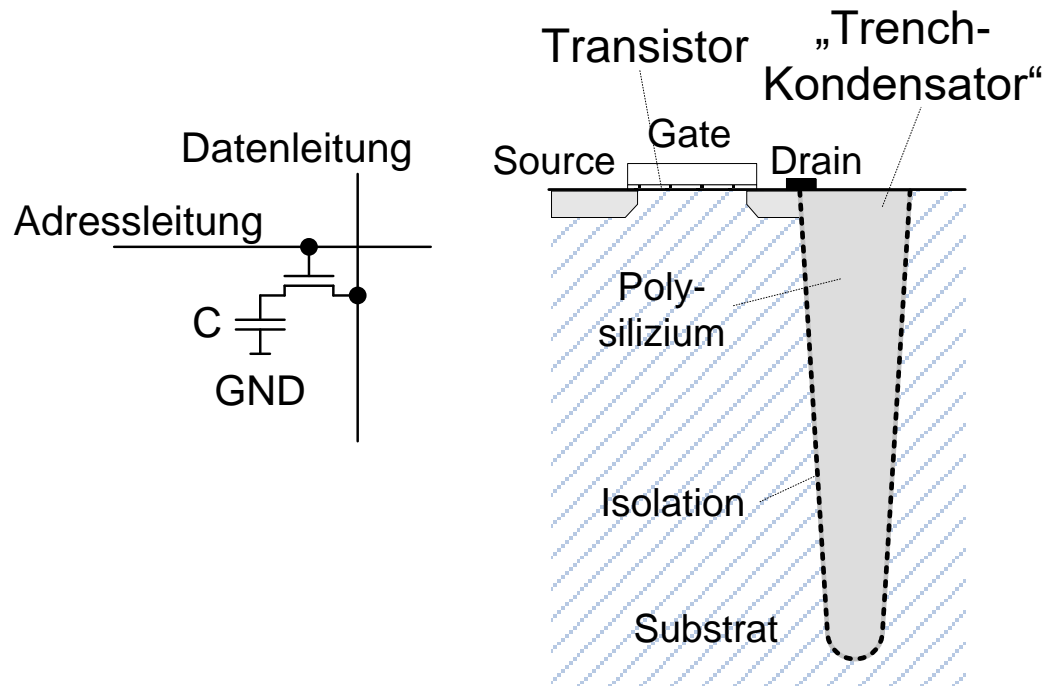
- DRAMs („**Dynamic Random Access Memory**“) bieten wesentlich höhere Speicherdichten als SRAMs
 - SRAM: etwa bis 144 Mbit Speicherkapazität
 - DRAM: etwa bis 16 Gbit Speicherkapazität (Stand 2018)
- Im Unterschied zum SRAM verliert das DRAM seinen Speicherinhalt nach einer gewissen Zeit
- Die Datenspeicherung in einem DRAM erfolgt als Ladung auf einem Kondensator
- Ein Transistor dient als Schalter zu einer Wortleitung
- Die Adressleitung öffnet den Transistor, so dass die Ladung gespeichert oder abgefragt werden kann
- Diese „**1-Transistor-Zelle**“ ist wesentlich kleiner als die „6-Transistor-Zelle“ des SRAMs
- Außerdem wird keine Versorgungsspannung und nur eine Datenleitung benötigt



Speichereigenschaften der DRAM-Zelle

Wichtig für die Informationsspeicherung ist ein Kondensator mit hoher Kapazität

- Eine Möglichkeit sind „**Trench-Kondensatoren**“, die tief in das Silizium-Substrat geätzt werden
 - „Trench“ = Graben



(Foto: Qimonda/Nanya)

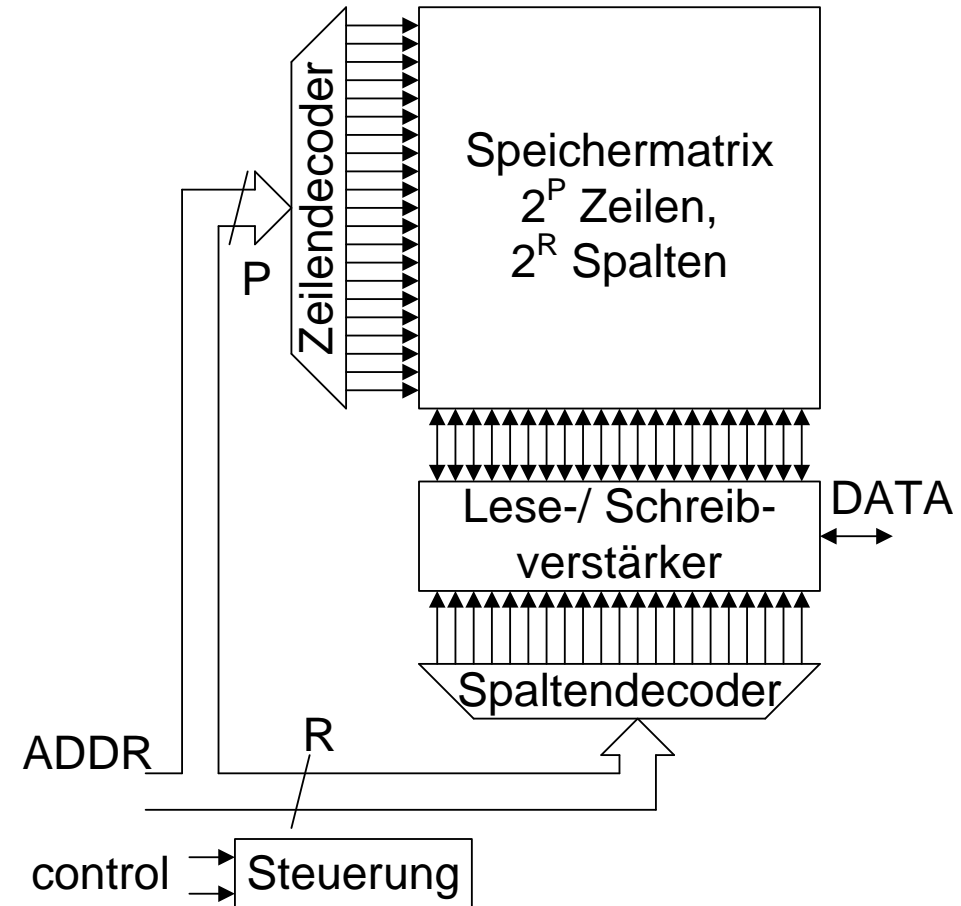
Speichereigenschaften der DRAM-Zelle (II)

Die Informationsspeicherung auf einem Kondensator führt zu folgenden **Problemen**:

- Die Ladung entlädt sich langsam (und das DRAM verliert seine Daten)
 - ➔ Der Speicherinhalt muss periodisch erneuert werden. Dies bezeichnet man als „**Refresh**“.
 - Die garantierte Speicherzeit, für alle möglichen Randbedingungen, liegt in der Größenordnung von 100 ms
 - Speicherinhalte können sich mit Glück (oder Pech?) jedoch auch mehrere Sekunden halten
- Das Lesen der Daten ist schwierig
 - Auf einer langen Datenleitung muss die Ladung eines einzelnen winzigen Kondensators erkannt werden
 - ➔ Die Leseverstärker müssen sehr empfindlich sein
- Beim Lesen wird die Information (Ladung) zerstört
 - ➔ Die Daten müssen nach dem Lesen erneut geschrieben werden

Ansteuerung eines DRAMs

- Aufgrund der Eigenschaften der DRAM-Zelle ist die Ansteuerung eines DRAMs komplexer als beim SRAM
- Folgender Ablauf ist für einen Lese- oder Schreibzugriff erforderlich
 - Auswahl der Zeile („**row**“)
 - Komplette Zeile wird in Lese-/ Schreibverstärker gespeichert
 - Durch Auswahl der Spalte („**column**“) werden Daten gelesen oder beschrieben
 - Komplette Zeile wird in Speichermatrix geschrieben
- Lesen und Schreiben der Speichermatrix benötigt relativ lange Zeit
- Möglichst mehrere Daten aus einer Zeile lesen, denn nur der erste Zugriff ist langsam



11.2.3 ROM

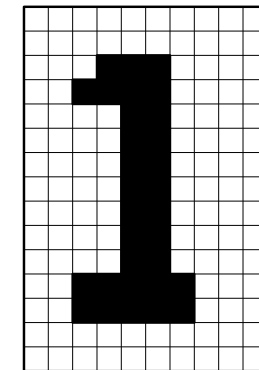
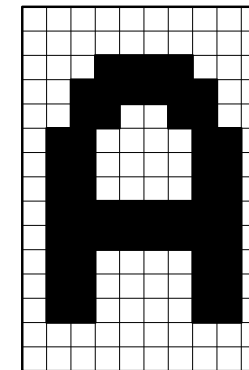
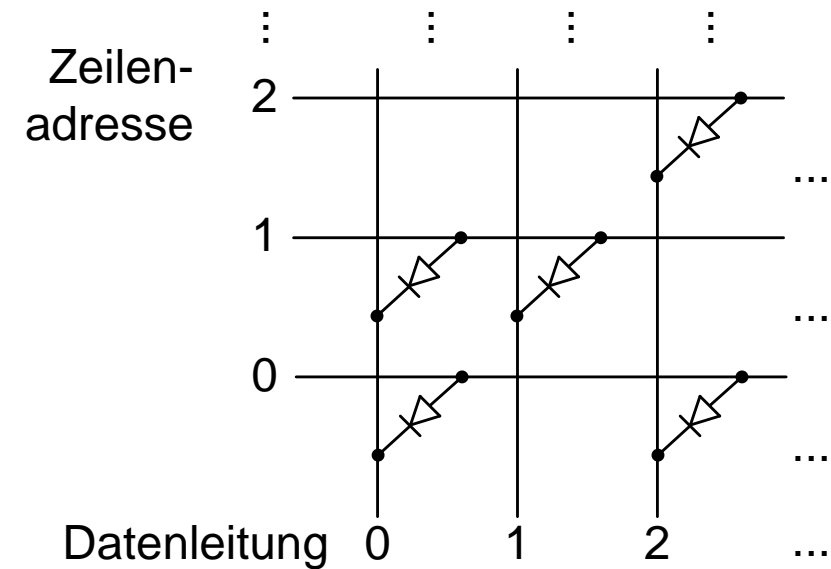
- Ein **ROM** („**Read-Only-Memory**“) kann speichert Daten permanent
- Die Grundstruktur (Speichermatrix, Zeilen-, Spaltendecoder) ist ähnlich zu SRAM, DRAM
- An den Speicherstellen wird durch Verbindungen eine ‚0‘ oder ‚1‘ gespeichert

Lesen einer Information

- Adressleitung auf ‚1‘ legen
- Leseverstärker überprüft, ob auf der Datenleitung ein Strom fließt
- Die unbenutzten Adressleitungen liegen auf ‚0‘ und sind durch die Dioden abgetrennt

Beispiele für Verwendung

- Boot-Code
- Fester Zeichensatz für Buchstaben
 - Grafik aus Digitaltechnik 1

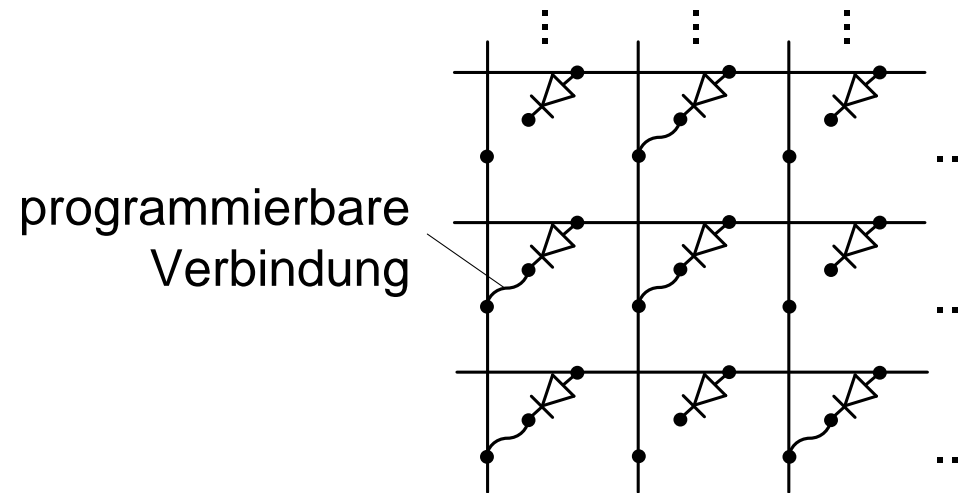


11.2.4 Einmalprogrammierbare Speicher

- Ein **PROM** („**P**rogrammable-**R**ead-**O**nly-**M**emory“) kann einmal beschrieben werden
 - Die Speicherelemente sind wie eine Sicherung aufgebaut
 - Sie können durch Programmierung aufgetrennt werden
 - Alternativ werden sie durch Programmierung leitend

Verwendung

- „Früher“
 - ROM
- „Heute“
 - Einmalprogrammierbare FPGAs
 - Sichere Konfiguration von ASICs



Z.B.: Xian Li et.al., „Reliable Antifuse One-Time-Programmable Scheme With Charge Pump for Postpackage Repair of DRAM,“ IEEE Trans. on Very Large Scale Integration (VLSI) Systems, 2015.

11.2.5 Flash EEPROM

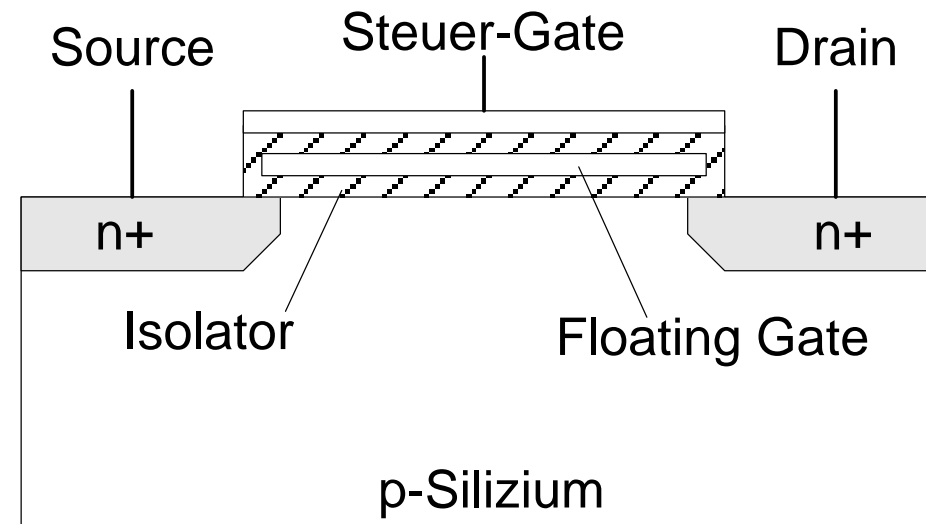
- Grundelement eines EPROMs ist ein CMOS-Transistor mit zusätzlichem isolierten Gate (engl.: „**Floating Gate**“)
- Auf dem Floating Gate kann Ladung sehr lange (z.B. 40 Jahre) gespeichert werden
 - Bei einer negativen Ladung auf dem Floating Gate ändert sich die Schwellspannung
→ Auch durch Anlegen einer Gate-Spannung öffnet der Transistor nicht

Programmierung „**Früher**“:

- Speicherung von Ladung durch hohe Spannungen (z.B. 12V)
- Löschen von Ladungen durch UV-Licht

Programmierung „**Heute**“:

- Speicherung und Löschen von Ladungen durch Tunneleffekt und/oder „Hot Electron Injection“
- Spannung wird, wenn erforderlich, intern erzeugt



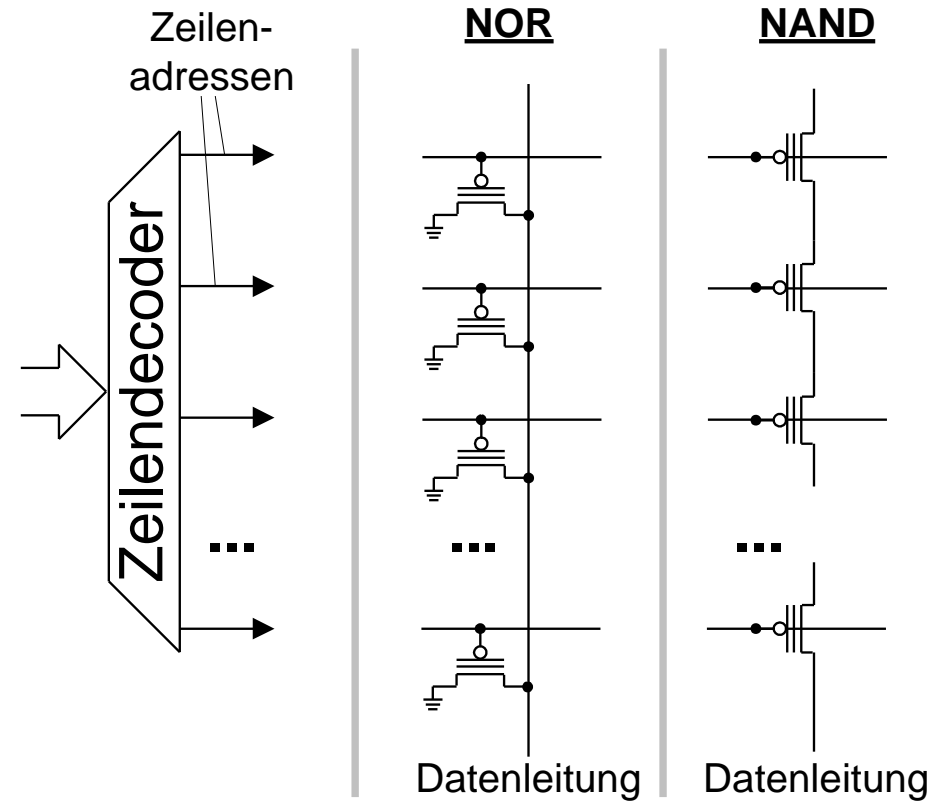
NOR- und NAND-Struktur

NOR-Struktur

- Speichertransistoren schalten die Datenleitung parallel nach Masse.
- Nicht aktive Transistoren stellen keine Verbindung nach Masse dar
- **Vorteil:** Geringer Widerstand auf der Datenleitung, darum gute Lesbarkeit der Daten
- **Nachteil:** Flächenbedarf, da jeder Transistor einen Kontakt zu Masse benötigt

NAND-Struktur

- Speichertransistoren sind in Reihe angeordnet
- Nicht aktive Transistoren sind leitend geschaltet
- Transistor, der gelesen wird, schaltet die Reihenschaltung leitend oder nicht leitend
- **Vorteil:** Geringe Fläche, da Speichertransistoren direkt aneinander geschaltet
- **Nachteil:** Auslesen schwieriger, da nicht aktive Transistoren auch im leitenden Zustand noch einen gewissen Widerstand haben



11.2.6 Innovative Speicher

- Der Markt für nicht-flüchtige Halbleiterspeicher (NVRAM, Non-Volatile RAM) wächst kontinuierlich
- Es werden weitere Speichertechniken entwickelt, die höhere Speicherkapazitäten, geringere Kosten oder einfachere Ansteuerung ermöglichen
 - Einige Techniken sind bereits im praktischen Einsatz
 - Marktanteile sind noch recht klein
 - Kommerzielle Entwicklung schwer absehbar

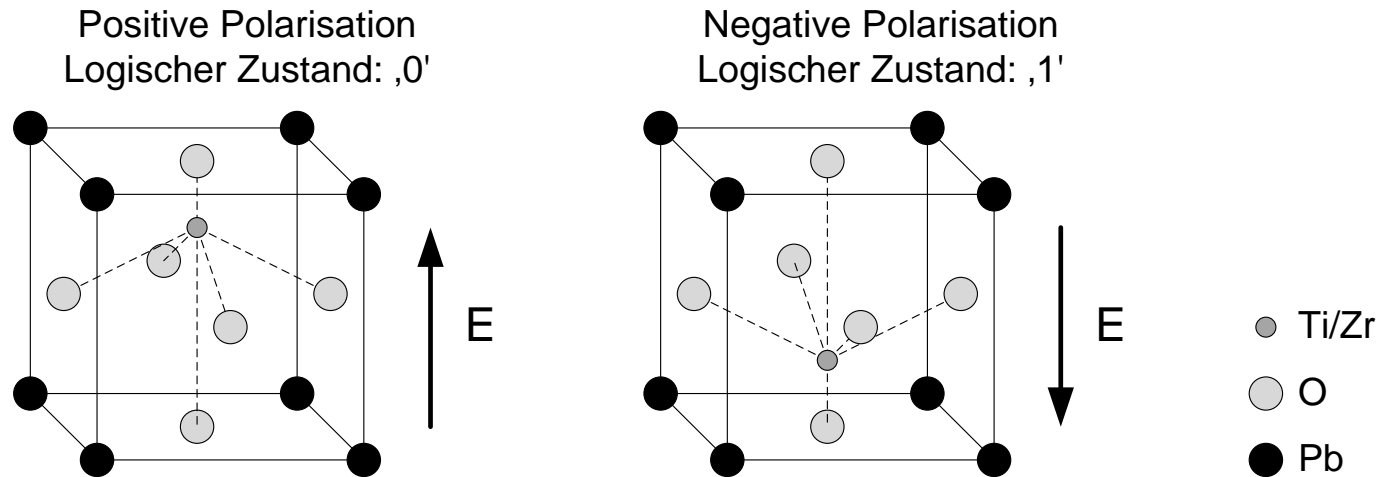
Für die Speicherung wird ein Material gesucht, welches

- zwei verschiedene Zustände hat, die sich in ihren elektrischen Eigenschaften unterscheiden
- einen einfachen Wechsel zwischen diesen Zuständen ermöglicht
- beide Zustände stabil über Jahre hinweg behält
- sehr oft zwischen diesen Zuständen wechseln kann, also mindestens hunderttausend, möglichst eine Milliarde Mal
- platzsparend und kostengünstig zu einem CMOS-Prozess ergänzt werden kann

FRAM

FRAM = Ferroelectric RAM

- Grundstruktur (Matrix) wie in anderen Speichertechnologien
- Grundzelle verwendet Kondensator mit ferroelektrischem Dielektrikum
 - Zwei stabile Zustände mit unterschiedlichem elektrischen Feld, dadurch unterschiedliche Spannung
 - Material, z. B. Blei-Zirkonat-Titanat (PZT) ($\text{Pb}(\text{ZrTi})\text{O}_3$)

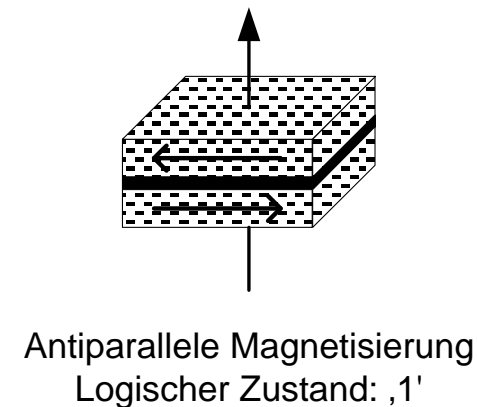
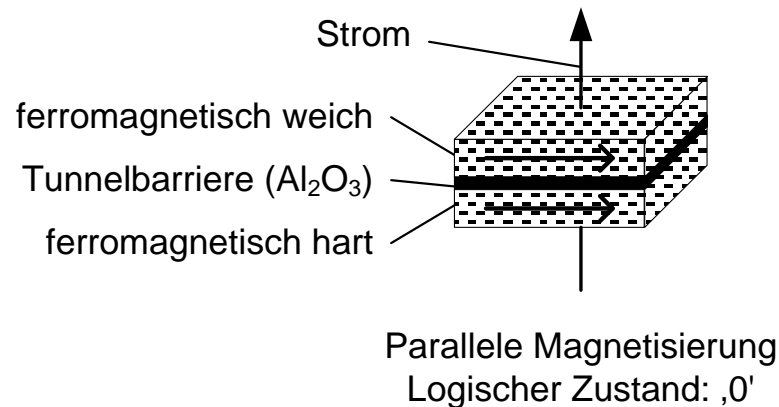


- Literatur: A. Sheikholeslami, P. G. Gulak, "A survey of circuit innovations in Ferroelectric random-access memories", Proceedings of the IEEE, 2000.

MRAM

MRAM = Magnetoresistive RAM

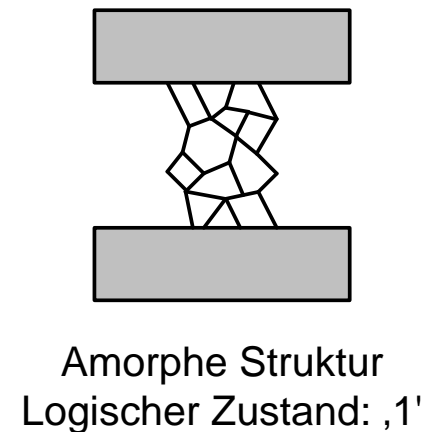
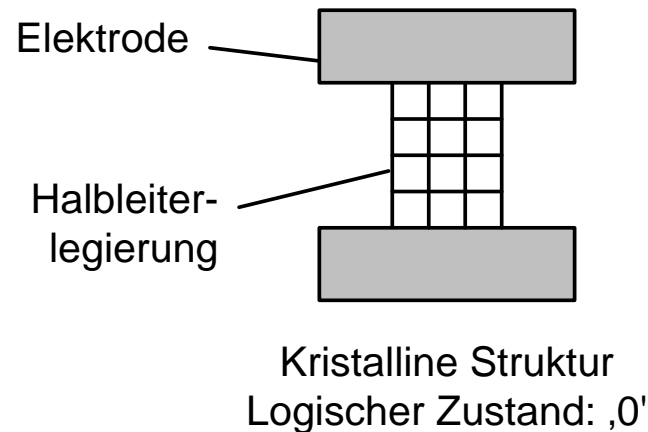
- Speicherung von Informationen in ferromagnetischer Schicht
- Schicht getrennt durch dünnes Dielektrikum aus Aluminiumdioxid (Al_2O_3) gegenüber einer weiteren magnetischen Schicht
 - Obere Magnetschicht ist magnetisch weich und kann in ihrer Orientierung gedreht werden.
 - Untere Magnetschicht ist magnetisch hart und hat eine feste Orientierung.
 - Der Strom durch das Dielektrikum ist durch einen Tunneleffekt abhängig davon, ob die magnetische Orientierung parallel oder antiparallel ist



PCRAM

PCRAM = Phase-Change-RAM

- Ein Phasenwechspeicher nutzt ein Material, welches kristalline oder amorphe Struktur einnehmen kann
- Je nach Struktur ist der elektrische Widerstand unterschiedlich und zeigt so eine ,0' oder ,1' an
- Der Wechsel zwischen den Strukturen erfolgt über Aufheizen durch elektrischen Strom
- Je nach Geschwindigkeit der Abkühlung wird das Material kristallin oder amorph



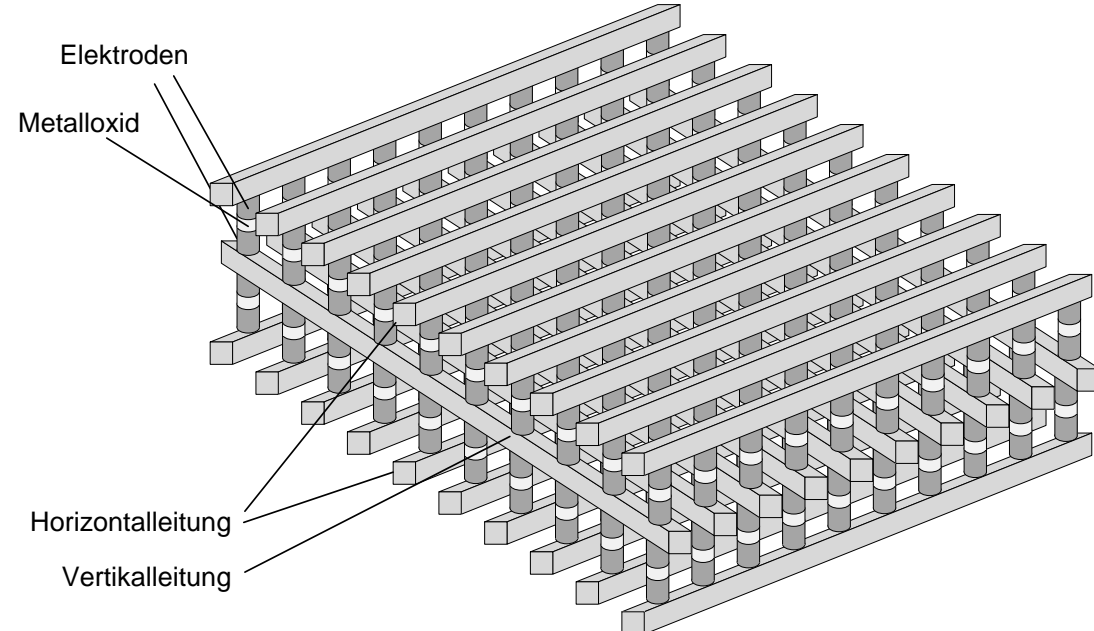
RRAM

RRAM, auch ReRAM = Resistive RAM

- Verändert wie PCRAM den Widerstand eines Speichermaterials
- Durch einen Strom kann der Widerstand eines Metalloxids zwischen hohem und niedrigem Widerstand wechseln
 - Dafür ist allerdings keine Erwärmung und Abkühlung des Materials nötig
 - Speichervorgang kann prinzipiell einfacher erfolgen

Ansteuerung

- Ansteuerung prinzipiell in Matrixstruktur
- Bei RRAM besonders kompakte Anordnung möglich
- Horizontale und vertikale Leitungen sprechen Speicherzelle direkt an
- Mehrere Lagen werden gestapelt
- Leitungen gemeinsam für zwei Ebenen an Speicherzellen nutzbar



Übungsaufgaben

Aufgabe 11-1

Auf einem FPGA wird das Quadrat und die Quadratwurzel von einer Dualzahl benötigt. Es sind nur positive Zahlen möglich; bei der Wurzel wird gerundet.

Die Funktion soll als ROM mit Block-RAMs implementiert werden. Gehen Sie von dem im Praktikum verwendeten Cyclone IV FPGA aus:

- Das EP4CE22E22C7 enthält 66 Block-RAMs, konfigurierbar z.B. zu
8192 Worte mit 1 bit Wortbreite, 4096 Worte mit 2 bit Wortbreite,
2048 Worte mit 4 bit Wortbreite, 1024 Worte mit 8 bit Wortbreite,
512 Worte mit 16 bit Wortbreite

Wie viele Block-RAMs sind erforderlich?

Welcher Prozentsatz des FPGA-RAMs wird verwendet?

- a) Quadrat n^2 einer Zahl n im Wertebereich von $n = [0:255]$
- b) Quadratwurzel \sqrt{m} einer Zahl m im Wertebereich von $m = [0:2047]$
- c) Quadratwurzel \sqrt{m} einer Zahl m im Wertebereich von $m = [0:8191]$

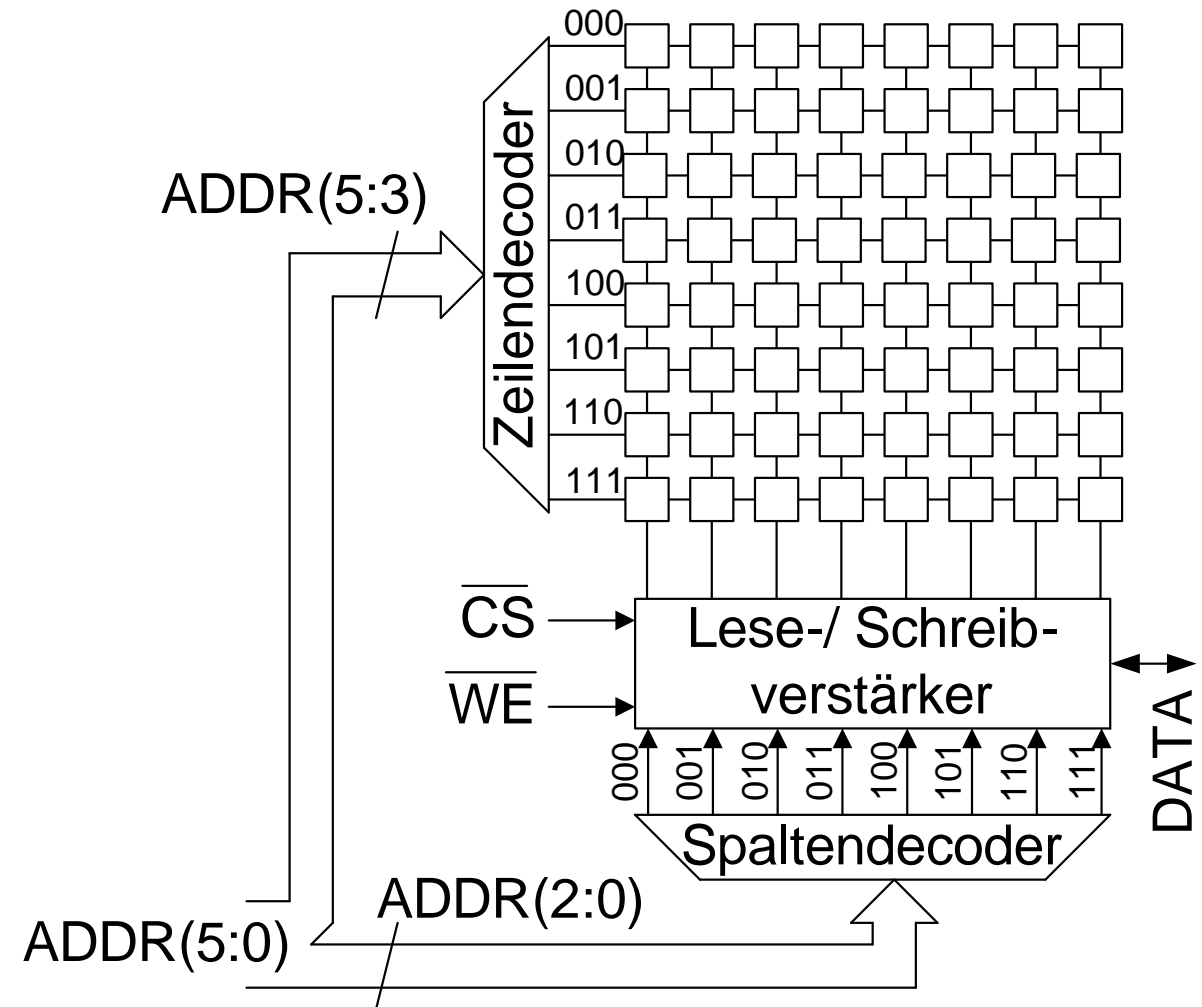
Übungsaufgaben

Aufgabe 11-2

Das Bild zeigt einen vereinfachten Speicherbaustein mit 6 Adressleitungen und 1 Bit Wortbreite. Damit soll ein Primzahl-Detektor realisiert werden.

Programmieren Sie die Speicherelemente so mit ,0' oder ,1', dass der Speicher eine ,1' ausgibt, wenn eine Primzahl am Eingang anliegt.

Die Eingangszahl lautet Z, der Ausgang P.



11.3 Eingebetteter Speicher

- Eingebetteter Speicher, engl. Embedded Memory, sind Speicherblöcke, die sich gemeinsam mit einer größeren Schaltung auf einem Chip befinden

SRAM

- In fast jedem größeren digitalen Chip befinden sich SRAM-Speicherblöcke
- Mit normaler CMOS-Fertigungstechnik hergestellt, daher keinen zusätzlicher Fertigungsaufwand
 - Z.B Cache einer CPU, Speicherung von Netzwerkdaten, ...

DRAM

- Deutlich höhere Speicherkapazität als SRAM, jedoch speziellen CMOS-Prozess
- Embedded-DRAM wird eingesetzt, wenn große Datenmengen gespeichert werden
 - Aber „sehr große Datenmengen“ übersteigen die Speicherkapazität
 - Bei kleineren bis mittleren Datenmengen wird Embedded-SRAM verwendet

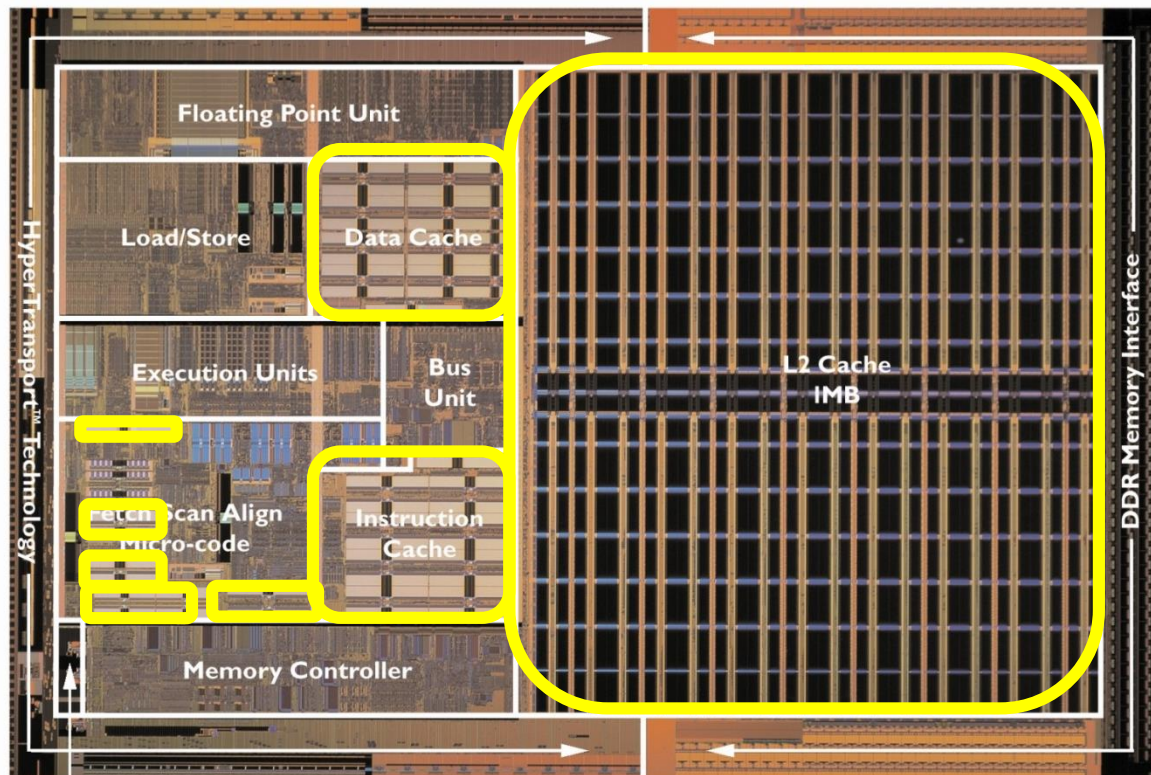
Beispiel für Embedded-DRAM: Grafik-Prozessor SM768 von Silicon Motions

- 256 Mbyte Embedded-DRAM
- Ansteuerung über USB 3.0 ohne Grafikkarte
- Kostengünstiger Aufbau eines intelligenten Monitors

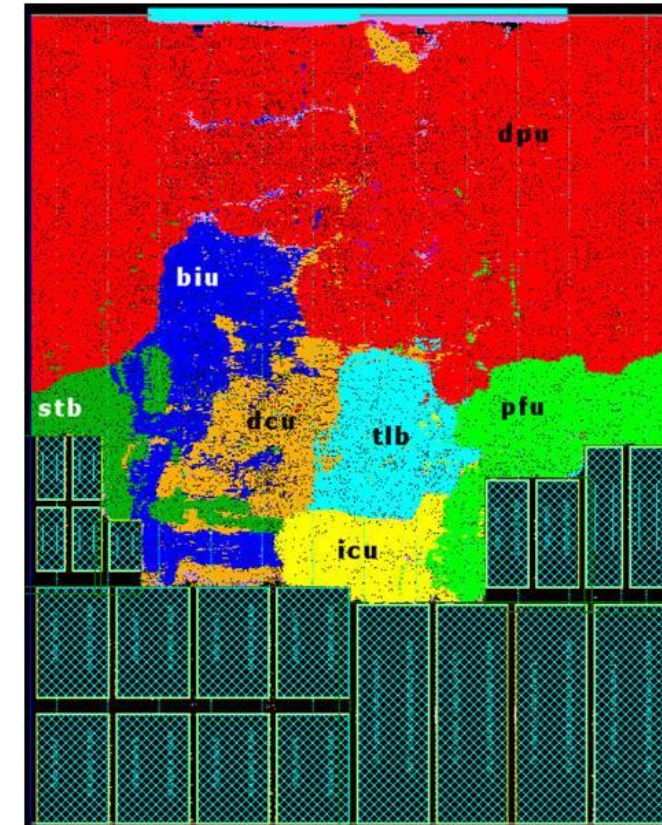
Beispiele für eingebettetes SRAM

- Links: AMD Athlon 64 FX (2003), SRAMs gelb markiert
- Rechts: Cortex-A7 CPU für Smartphones (2011)
 - Rechteckige Blöcke sind SRAMs (eventuell 1 oder 2 Spezialfunktionen, z.B. Takt)

(Foto: AMD)

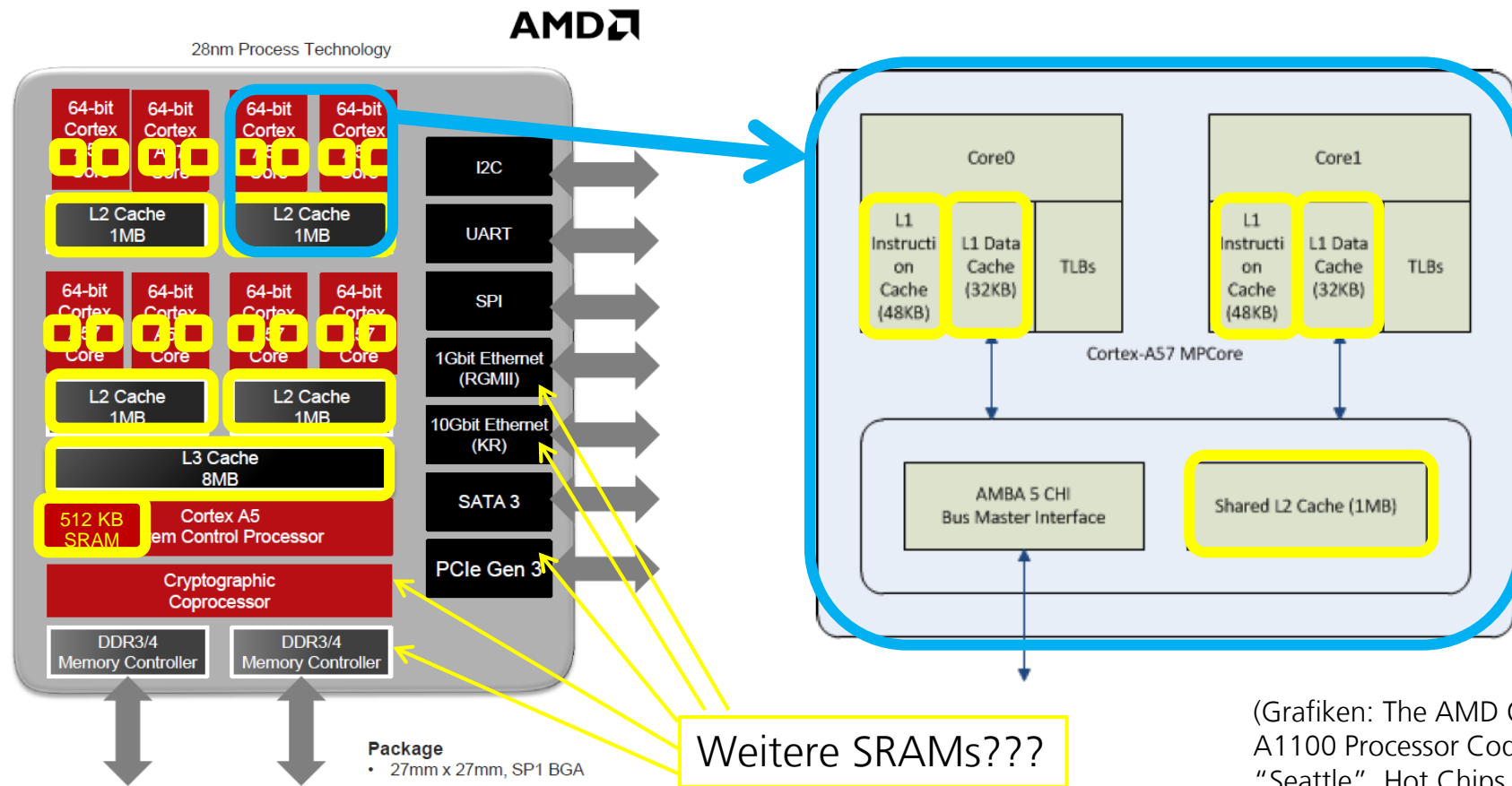


(Grafik: ARM)



Beispiele für eingebettetes SRAM (II)

- AMD Opteron A1100 Acht-Kern CPU (SRAMs wieder gelb markiert)
 - Quelle: Hot Chips Conference 2014 <http://www.hotchips.org/archives/2010s/hc26/>
 - Weitere Infos: <http://www.heise.de/ix/meldung/AMD-zeigt-erste-Opterons-mit-ARM-Technik-2099511.html>



(Grafiken: The AMD Opteron™ A1100 Processor Codenamed "Seattle", Hot Chips 2014)

Eingebetteter Speicher - ROM

ROM

- Wird wie SRAM mit normaler CMOS-Fertigungstechnik hergestellt

Beispiel: Tabelle für arithmetische Operationen

- Von einer 10 bit Dualzahl wird die Wurzel benötigt.
- Ausgabewert soll auf ganze Zahlen gerundet werden
 - Eingangswerte 0 bis 1023
 - Wurzel hiervon ist 0 bis 31,98, gerundet 0 bis 32
- Der Eingangswert wird als Adresse an das ROM angelegt
- Am Ausgang wird das Ergebnis der Wurzelberechnung angezeigt

<div>ADDR(9:0) — <div>ROM 1024x6</div> — DOUT(5:0)</div>				
ADDR (in hex)	Zahlen- wert	Wurzel	Wurzel gerundet	DOUT (in hex)
000	0	0	0	00
001	1	1	1	01
002	2	1,41	1	01
003	3	1,73	2	02
...
123	291	17,06	17	11
...
3FF	1023	31,98	32	20

Eingebetteter Speicher - NVRAM

NVRAM

- Erfordert speziellen CMOS-Prozess, wie bei DRAM
 - Anders als beim DRAM gibt es jedoch keine Alternative, wenn ein Chip Daten auch ohne Versorgungsspannung speichern soll
- Häufig eingesetzt bei Mikrocontrollern
 - Programmspeicher ist als NVRAM implementiert
- Anwendungsbeispiel ATmega328-Controller der Firma Atmel
 - Eingesetzt auf Arduino Uno
 - Programmspeicher: 32 Kbyte
 - Datenspeicher: 1 KByte, kann vom Programm gelesen und beschrieben werden
- Alternativ kann das System auch auf zwei Chips aufgeteilt werden.
 - Ein Chip in Standard-CMOS enthält den Mikrocontroller
 - Ein Speicher-Chip enthält den Programmspeicher

11.4 Diskrete Speicherbausteine

- Digitale Systeme bestehen häufig aus mehreren Chips
 - Signalverarbeitungschips, gefertigt in einem Standard-CMOS-Prozess
 - Ein oder mehreren Speicher-Chips, gefertigt in speziellen CMOS-Varianten

Vorteile

- Hohe Kapazität externer Speicherbausteine
- Höhere Flexibilität des Systems
 - Im PC können DRAM-Riegel nach Bedarf eingesetzt werden
 - Smartphones werden mit verschiedener Speicherkapazität verkauft
- Externe Speicherbausteine sind gut verfügbar
 - Embedded-DRAM ist nur für Großkunden möglich
- Neue Speichertechnologien zunächst für diskreten Speicherbausteine angeboten
- Kosten für Standard-CMOS-Chip plus Speicher-Chip oft geringer

Nachteile

- Platzbedarf, insbesondere für mobile Geräte ungünstig
- Speicherzugriff auf externe Bauelemente hat eine geringere Bandbreite
- Produktlebensdauer der Speicherbausteine beachten

11.4.1 QDR-II-SRAM

- Dual-Port-SRAMs, mit zwei Anschlüssen: Schreib- und Lese-Interface
- Anschlüsse übertragen Daten bei steigender und fallender Taktflanke (Double-Data-Rate), darum Bezeichnung Quad-Data-Rate (QDR)
- Einsatzgebiet: Anwendungen mit sehr hoher Datenrate, bei denen Lese- und Schreiboperationen etwa gleich häufig vorkommen.
 - Beispiel: Zwischenspeicherung von Datenpaketen in Netzwerkanwendungen
- Speichergrößen im Bereich von 18 bis 144 Mbit
- Datenwortbreiten von 9, 18 und 36 Bit
 - Vielfache von 9 bit, nicht 8 bit, da in der Telekommunikation häufig zusätzliche Bits zur Fehlererkennung verwendet werden
- Typischer Baustein ist der CY7C1514KV18 von Cypress
 - Speicherkapazität von 72 Mbit
 - 36 Bit Datenwortbreite
 - Taktgeschwindigkeit darf 350 MHz betragen
 - Vergleichbare Bausteine unter anderem von IDT und Renesas

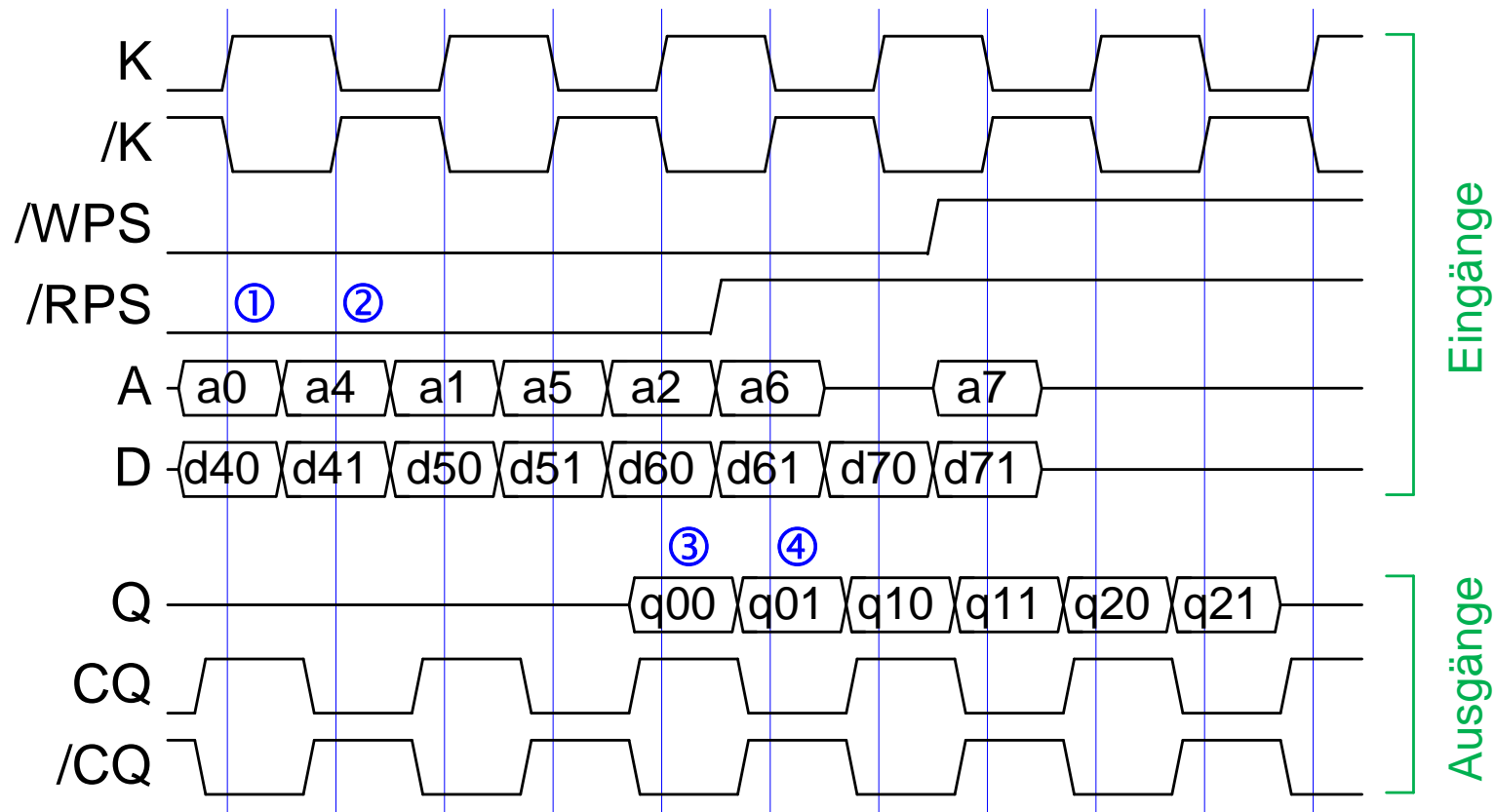
QDR-II-SRAM (II)

Anschlüsse

- A, 20 bit, Adresse, gemeinsam für Schreib- und Lese-Interface
- D, 36 bit, Dateneingang
- Q, 36 bit, Datenausgang
- /WPS, Write-Port-Select aktiviert einen Schreibzugriff
- /RPS, Read-Port-Select aktiviert einen Lesezugriff
- K und /K, Takt für Schreib-Interface in positiver und negativer Polarität
- C und /C, Takt für Lese-Interface in positiver und negativer Polarität
- CQ und /CQ, Ausgabe des Takts C für Anpassung an Laufzeiten
- VREF, Referenzspannung für Datenleitungen
- weitere Pins für Steuerfunktionen, Stromversorgung und Fertigungstest
- Insgesamt 165 Pins, Schrägstrich (/) kennzeichnet Low-aktive Signale
- Hohe Anzahl an Taktanschlüssen.
 - Takte sind nicht unabhängig voneinander, sondern prinzipiell der gleiche Takt mit unterschiedlicher Verzögerung

QDR-II-SRAM (III)

- Drei **Lesezugriffe** auf die Adressen a0 bis a2 als **Burst von zwei Datenworten**
 - Adresse a0 bei steigender Flanke von K ①
 - Datenwort mit der Bezeichnung q00 wird bei ③ ausgegeben, bei ④ folgt q01
- Vier **Schreibzugriffe** auf die Adressen a4 bis a7, ebenfalls als **Burst zweier Worte**
 - Adresse bei fallender Flanke von K ②, Daten bei ① und ②



Übungsaufgaben

Aufgabe 11-3

Ein SRAM enthält 2^{23} Datenworten zu 8 bit.

- a) Wie groß ist die Speicherkapazität in Bit? Geben Sie den Wert mit einem sinnvollen Vorsatz an (kBit, Mbit, Gbit).
- b) Wie viele Transistoren enthält das SRAM etwa? Berechnen Sie die Anzahl der Transistoren der Speichermatrix und addieren Sie 20% für die Ansteuerung.
- c) Wie viele Zeilen und Spalten hat die Speichermatrix? (Annahme: Quadratische Anordnung)
- d) Wie viele Adressleitungen sind für Zeilen, Spalten, insgesamt erforderlich?

Aufgabe 11-4

Das QDR-II-SRAM CY7C1525KV18 hat eine Speicherstruktur von „8 M x 9“. Eine Adresse spricht immer zwei Datenworte als Burst an.

- a) Wie groß ist die Speicherkapazität in Bit?
- b) Wie viele Transistoren enthält das SRAM etwa? (20% Overhead für Ansteuerung)
- c) Wie viele Zeilen und Spalten hat die Speichermatrix? (Annahme: Etwa quadratische Anordnung, Daten eines Burst werden gleichzeitig aus Speichermatrix gelesen)
- d) Wie viele Adressleitungen sind für Zeilen, Spalten, insgesamt erforderlich?

11.4.2 DDR3-SDRAM

- Moderne Familie von DRAM-Speichern mit einer Kapazität bis zu 4 Gbit
 - Rund 30mal so viel wie ein QDR-II-SRAM
- **Beispiel:** MT41J512M8 von Micron, 4 GBit DDR3-DRAM mit 8 bit Wortbreite
 - Vergleichbare Bausteine von Samsung und Hynix angeboten
 - Verwendung Baustein beispielsweise in DRAM-Modulen für PCs

Anschlüsse

- A, 16 bit, Adresse
- BA, 3 bit, Bankadresse, wählt einen von acht internen Speicherbänken aus
- DQ, 8 bit, Datenbus, Bidirektional als Dateneingang und Datenausgang
- DQS und /DQS, Referenzsignal für das Ausgangstiming
- /RAS, /CAS, /WE, Steuersignale für Lese- und Schreiboperationen
- CK und /CK, Takt in positiver und negativer Polarität (bis 1 GHz)
- VREF_DQ, VREF_CA, Referenzspannungen für Daten- und Steuerleitungen
- weitere Pins für Steuerfunktionen, Stromversorgung und Fertigungstest
- Insgesamt 78 Anschlüsse, also weniger als die Hälfte des QDR-II-SRAMs

DDR3-SDRAM (II)

- Beim Speicherzugriff muss der innere Aufbau beachtet werden

Beispiel: Lesevorgang

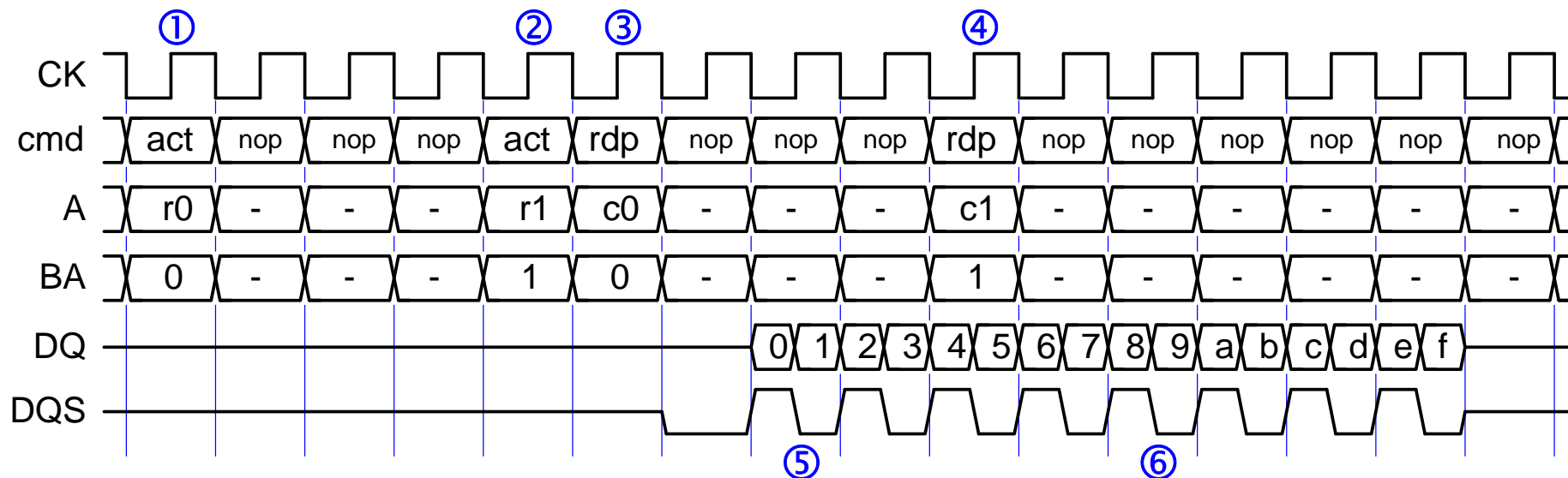
- Eine komplette Zeile wird in den Schreib/Lese-Verstärker geladen.
 - Ladung in den Speicherzellen wird gelöscht und muss zurückgespeichert werden
 - Lesen und Zurückschreiben benötigt mehrere Takte, während dessen das DRAM blockiert ist
 - Darum sind in einem DRAM-Chip acht unabhängige Speicherbänke
 - Während eine Bank durch Zurückschreiben von Daten belegt ist, kann auf eine andere Bank zugegriffen werden.
- Lesezugriff auf den Speicher erfolgt in drei Schritten.
 - Activate: Zeile in den Leseverstärker laden
 - Read: Aus der Zeile werden Datenworte gelesen; mehrere Leseoperationen für die aktivierte Zeile sind möglich und jede Leseoperation liest einen Burst von vier oder acht Worten
 - Precharge: Der Zugriff auf die Zeile wird beendet und die Daten wieder in die Speicherzellen zurückgeschrieben

DDR3-SDRAM (III)

Zeitablauf für zwei Leseoperationen (Burst 8 Worte) auf zwei verschiedene Bänke

➤ Steuersignale /RAS, /CAS, /WE zusammengefasst zum Kommandowort ,cmd‘

1. Aktivierung der Zeile r0 (r wie row) in der Bank 0 mit dem Kommando ,act‘
2. Aktivierung der Zeile r1 in der Bank 1.
3. Lesezugriff auf Spalte c0 (c wie column) in der Bank 0; nach Ausführen der Leseoperation Precharge, darum Kommando ,rdp‘ (Read with Precharge)
4. Lesezugriff auf Spalte c1 in der Bank 1, ebenfalls mit Precharge.
5. Nach Latenzzeit werden Daten des Zugriffs ③ ausgegeben, 8 Daten: 0 bis 7
6. Daten des Lesezugriffs ④ werden ausgegeben, 8 Daten: 8 bis f



DDR3-SDRAM (IV)

- Die Taktgeschwindigkeit und die Wartezeiten werden als Kennziffern des DRAMs angegeben
- **Beispiel:** DDR3-1866 CL13 13-13-32
 - DDR3-1866: DDR3-SDRAM mit 1866 Millionen Transfers je Sekunde, also 933 MHz Taktgeschwindigkeit, da zwei Transfers pro Takt.
 - CL13 (CAS Latency): Anzahl der Takte zwischen Read und Ausgabe der Daten
 - Die folgenden drei Zahlen bezeichnen weitere Zeiten
 - 13 Takte zwischen Activate einer Zeile und erstem Read-Zugriff
 - 13 Takte für den Precharge
 - 32 Takte insgesamt zwischen zwei Activate-Befehlen auf dieselbe Bank
- Der Zugriff auf ein DRAM erfordert also das Beachten der internen Speicherorganisation
 - Hohe Datenrate, wenn mehrere Daten aus der gleichen Zeile gelesen und Zugriffe ansonsten auf verschiedene Speicherbänke verteilt werden
 - Diese Zugriffsmuster werden z.B. von CPUs in einem PC berücksichtigt; der Cache liest größere Datenblöcke aus dem DRAM

Übungsaufgaben

Aufgabe 11-5

Berechnen Sie die Datenrate beim Lesen aus dem DRAM DDR3-1866 CL13 13-13-32 für folgende Fälle:

- a) Optimale Datenstruktur: Es werden stets Blöcke zu 8 Byte gelesen, die Zugriffe erfolgen nacheinander auf verschiedene Bänke.
- b) Unregelmäßiger Blockzugriff: Es werden Blöcke zu 8 Byte gelesen, die Zugriffe erfolgen auf zufällige Adressen.
- c) Unregelmäßiger Bytezugriff: Es werden einzelne Byte gelesen, die Zugriffe erfolgen auf zufällige Adressen.
- d) Bytezugriff mit Abhängigkeit: Es werden einzelne Byte gelesen, die Zugriffe erfolgen abhängig vom Inhalt des letzten Zugriffs.

Es muss also das Ergebnis eines Zugriffs abgewartet werden, danach wird durch eine Berechnung die nächste Adresse ermittelt. Gehen Sie von 5 Takten für die Berechnung der neuen Adresse aus.

- e) Wie groß ist die Datenrate des QDR-II-SRAMs CY7C1525KV18?

11.4.3 EEPROM

- Große Vielfalt an EEPROMs
 - Kleine Speicher (kByte) für Geräteeinstellungen: Netzwerk, WLAN-Passwort
 - Mittlere Speicher (Mbyte) für Messdaten, Programmcode für Prozessoren
 - Große Speicher (Gbyte) als Massenspeicher, z.B. Smartphone oder SSD
- Der Speicherzugriff kann seriell oder parallel erfolgen
 - Seriell ist ausreichend, wenn nur wenige Daten benötigt werden
 - Parallel ist schneller und für größere Datenmengen sinnvoll

Beispiel: TH58NVG3S0HTA00 von Toshiba

- Flash-EEPROM mit größerer Speicherkapazität von 8 GBit, also 1 Gbyte
- NAND-Struktur mit 4096 Blöcken
 - Jeder Block hat 64 Seiten mit jeweils 4352 Bytes
 - Je Seite 4096 Bytes Nutzdaten plus 256 Bytes für Speicherverwaltung und Fehlerkorrektur (wegen NAND-Struktur)
 - Flash-Löschvorgang immer auf einen Block von 64 Seiten, also 256 KByte

EEPROM (II)

- Der Baustein ist darauf ausgelegt, mit einem fehlerkorrigierenden Controller zusammenzuarbeiten

Anschlüsse

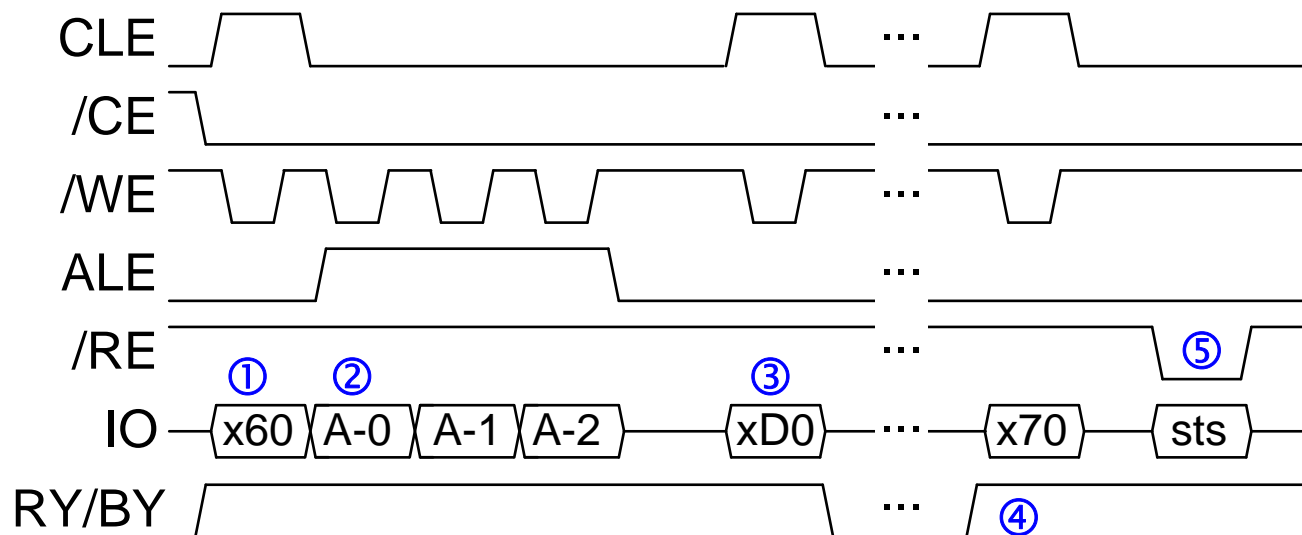
- IO, 8 bit, I/O-Port für Adresse und Daten
- CLE, Command Latch Enable, Übernahmesignal für Befehle
- ALE, Address Latch Enable, Übernahmesignal für Adresse
- /CE, Chip Enable
- /WE, Write Enable
- /RE, Read Enable
- RY/BY, Ready/Busy, zeigt an, ob der Baustein noch einen Befehl ausführt
- /WP, Write Protect, für einen Schreibschutz
- Pins für Stromversorgung

- Gehäuse mit 48 Anschlüssen, davon einige nicht verwendet
- Das Speicherinterface arbeitet **ohne Takt**

EEPROM (III)

Zeitablauf beim Löschen eines Blocks

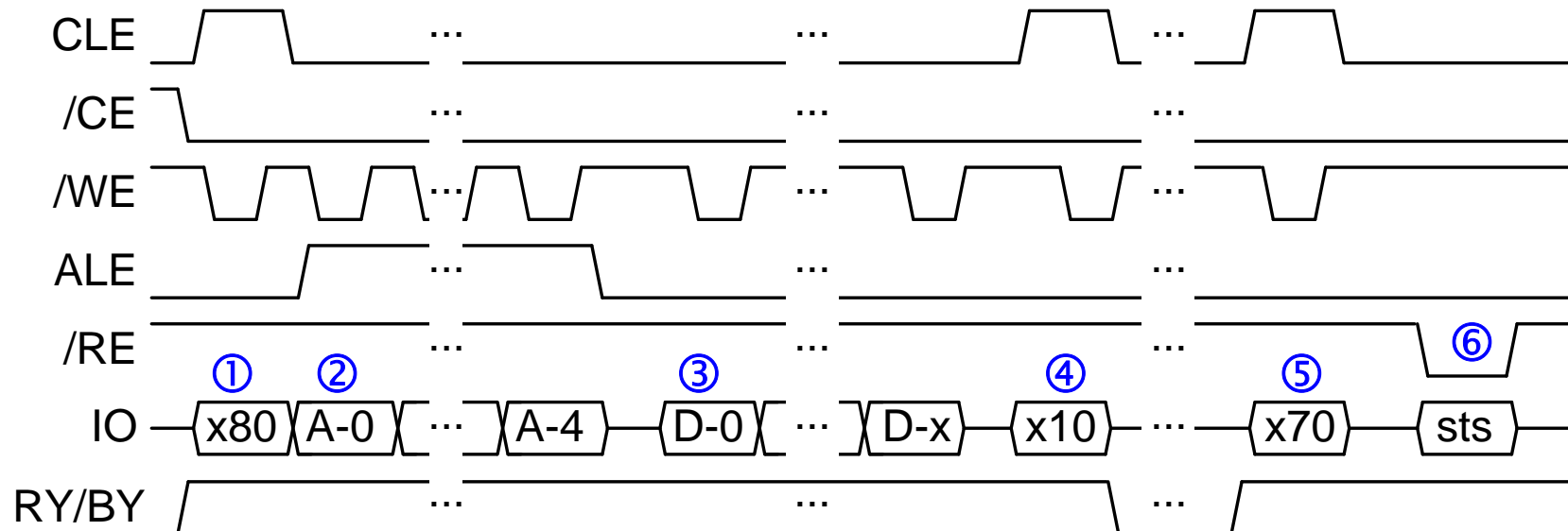
1. Löschvorgang wird durch ein spezielles Kommando (x60) gestartet; CLE zeigt an, dass es sich bei dieser Information um ein Kommando handelt
2. Die Adresse des zu löschenden Blocks wird in drei Teile A-0, A1, A-2 auf der Datenleitung übertragen
3. Das Kommando xD0 löst den Löschvorgang aus; RY/BY zeigt etwa 2,5 bis 5 ms lang an, dass der Baustein beschäftigt ist
4. Kommando x70 überprüft, ob das Löschen erfolgreich war
5. Baustein antwortet mit Statuswort (sts); Leseoperation wird durch /RE angesteuert



EEPROM (IV)

Zeitablauf beim Schreiben von Daten

1. Kommando x80 gibt an, dass ein Schreibvorgang ausgeführt werden soll
2. Die Adresse von Block und Seite wird in fünf Teilen von A-0 bis A-4 übertragen
3. Bis zu 4352 Byte werden mit der steigende Flanke von /WE übertragen
4. Kommando x10 startet den eigentlichen Schreibvorgang von internem Zwischenspeicher in die Speichermatrix
 - Das EEPROM überprüft den Schreibvorgang und schreibt eventuell mehrfach
 - Die Programmierung einer Seite dauert 300 bis 700 μ s (bis RY/BY = ,1‘)
5. Kommando x70 überprüft das Schreiben, Baustein antwortet mit „sts“ (6.)



Übungsaufgaben

Aufgabe 11-6

Berechnen Sie die Datenrate beim Schreiben in das TH58NVG3S0HTA00 von Toshiba.

Gehen Sie von folgenden Annahmen aus:

- Blöcke von 4K Byte werden geschrieben und geben mit Fehlerkorrektur 4352 Byte
- Taktfrequenz des Controllers 40 MHz
 - $WE=,0'$ und $,1'$ liegen für einen Takt an
 - Daten an IO liegen dann zwei Takte an
- Berechnung der Fehlerkorrektur erfolgt parallel zum Schreiben und verursacht keine Zeitverzögerung (optimistische Annahme)

11.4.4 NVRAM mit innovativer Speichertechnologie

- MB85RS64V von Fujitsu
- Ferroelektrisches RAM (FRAM) mit 8192 Worten zu 8 Bit
 - Jede Zelle kann einzeln beschrieben werden
- Baustein ist sehr kompakt mit serielltem Interface und einem Gehäuse mit nur 8 Pins
- Besonderer Vorteil sind 10^{12} mögliche Zugriffe pro Zelle spezifiziert
 - EEPROMs üblicherweise 10^5 bis 10^6 Schreibvorgänge spezifiziert

Anschlüsse

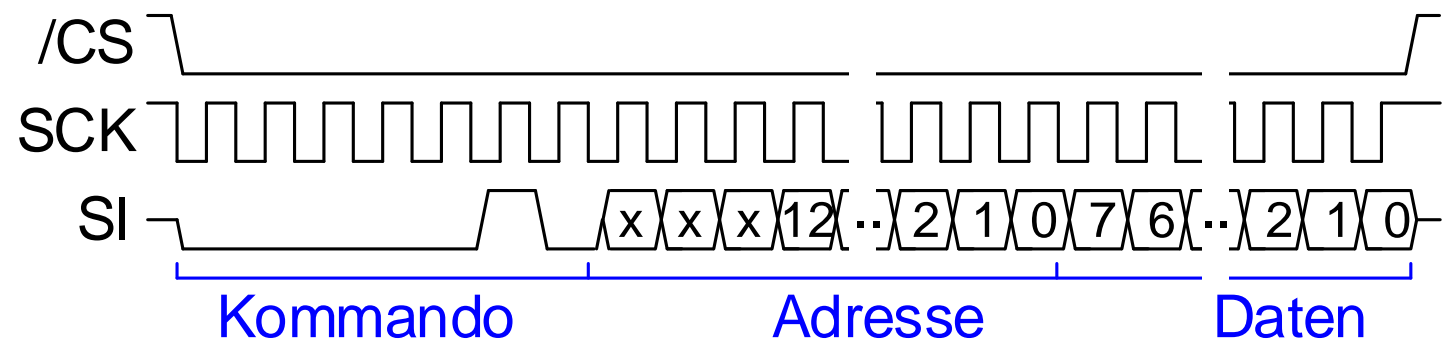
- SCK, Serial Clock, Takt für den seriellen Zugriff (bis 20 MHz)
- SI, Serial Data Input, serieller Dateneingang
- SO, Serial, Data Output, serieller Datenausgang
- /CS, Chip Select, zum Aktivieren des Bausteins
- /WP, Write Protect, Schreibschutz
- /HOLD, pausiert einen Zugriff
- Versorgungsspannung und Masse

NVRAM mit innovativer Speichertechnologie (II)

- Kommunikation über das Serial Peripheral Interface (SPI)
- Getrennte Leitungen für Dateneingang und -ausgang, also keine bidirektionale Datenleitung

Zeitablauf eines Schreibvorgangs

- /CS aktiviert den Baustein
- 32 Bits werden durch die steigende Flanke von SCK übertragen
 - 8 Bit: Schreibkommando x02
 - 16 Bit: Adresse (für 8 KByte nur 13 Bit nötig, darum drei Adressbits unbelegt)
 - 8 Bit: Daten
 - Weitere Daten für Folgeadressen möglich (nicht dargestellt)
- /CS beendet die Übertragung



Übungsaufgaben

Aufgabe 11-7

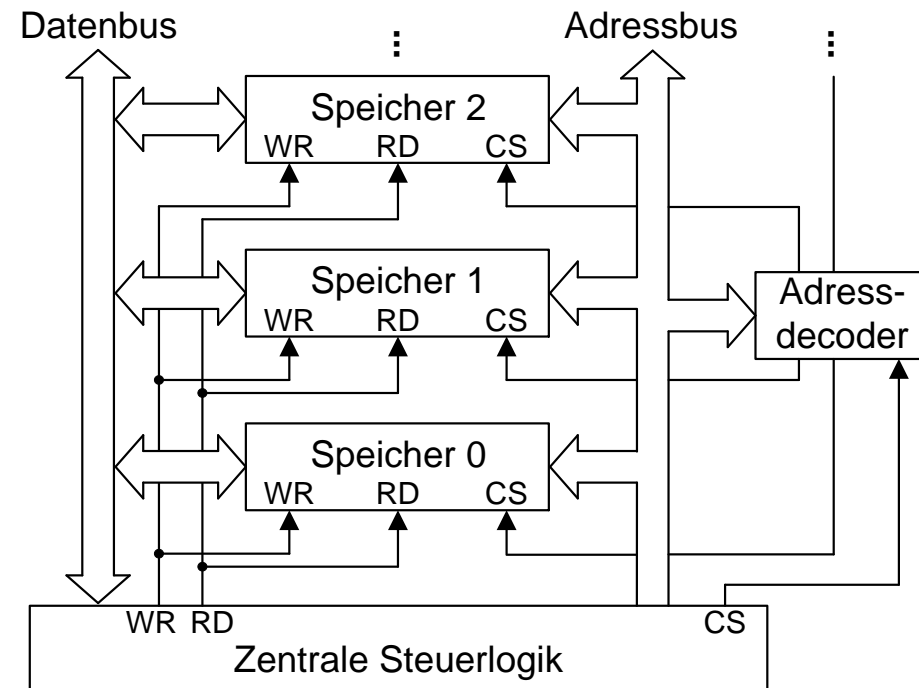
Berechnen Sie die Datenrate beim Schreiben in das MB85RS64V von Fujitsu.

Gehen Sie von folgenden Annahmen aus:

- Schreiben einzelner Bytes; die Option Daten in Folgeadressen zu schreiben wird nicht genutzt
- Taktfrequenz 20 MHz für SCK
- 2 Takte Abstand zwischen Schreiboperationen

11.5 Speichersysteme

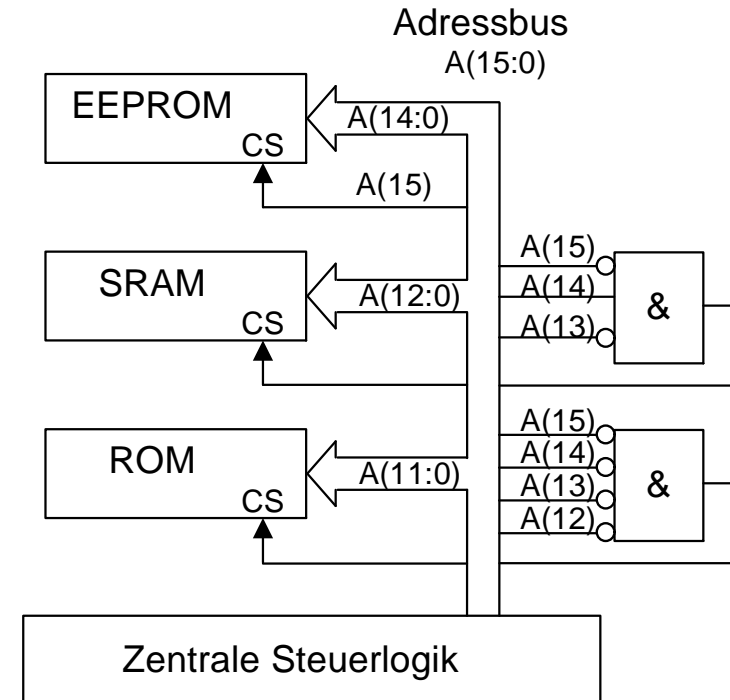
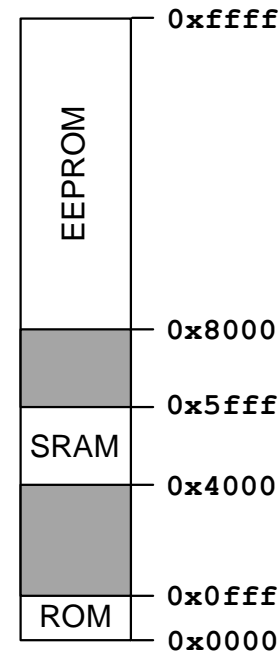
- Oft werden Speicher gemeinsam von einer zentralen Steuerlogik angesprochen
- Die Unterscheidung der Speicher erfolgt durch die Speicheradresse
 - Gesamter Adressraum mit Adressbereichen für die unterschiedlichen Speicher
 - Je nach Adressraum und Speichergröße können Adressbereiche unbelegt sein
- Prinzipieller Aufbau eines Speichersystems im Bild dargestellt
 - Zentrale Steuerlogik gibt Adresse sowie Steuersignale an das Speichersystem
 - CS, Chip Select: Zugriff
 - WR, Write: Schreibzugriff
 - RD, Read: Lesezugriff
 - Ein Adressdecoder ermittelt das adressierte Speichermodul
 - Datenbus gibt Daten an das Speichermodul oder es werden Daten empfangen



Beispiel: Speichersystem eines Mikrocontrollers

Fiktiver Mikrocontroller

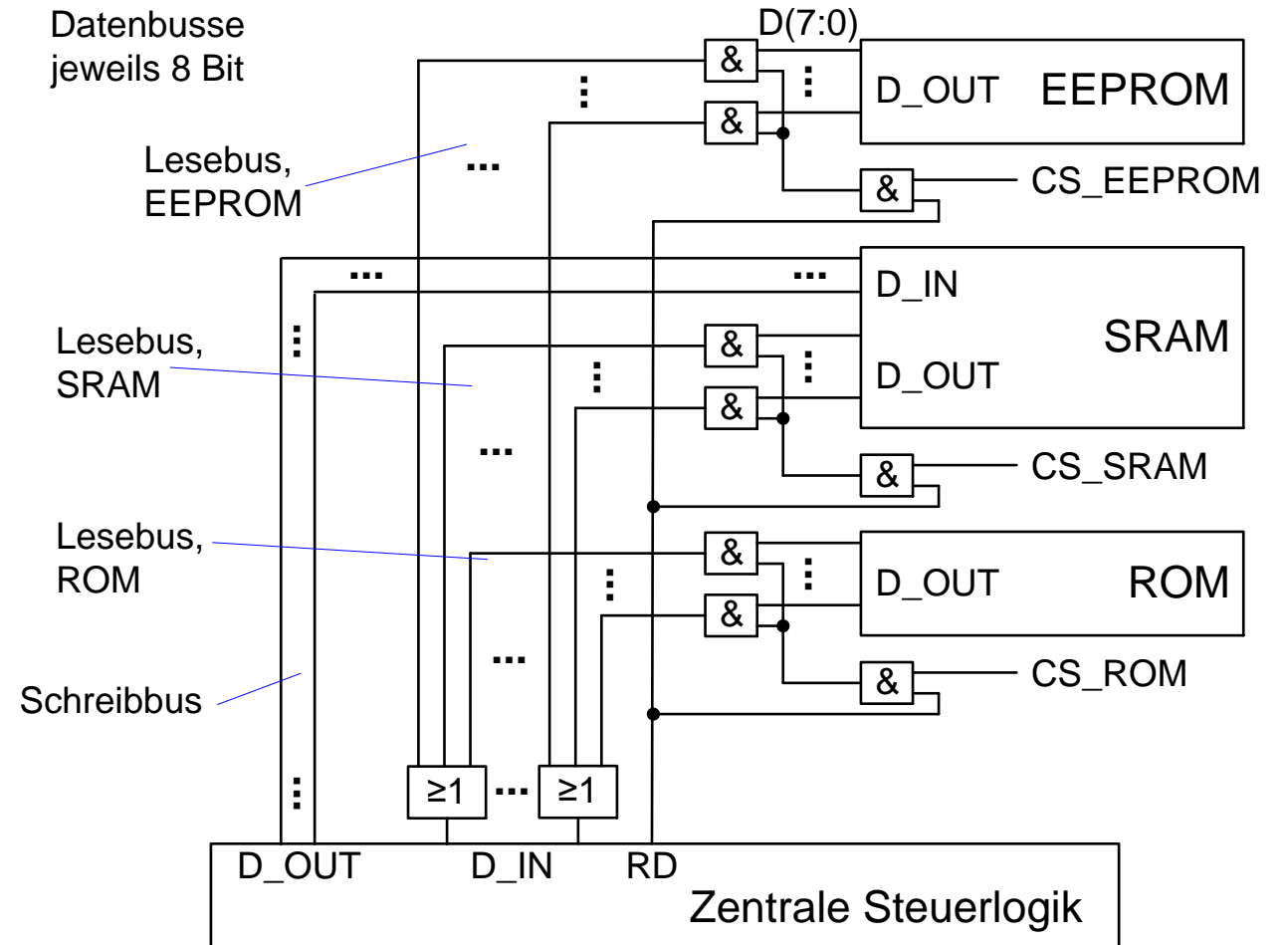
- Adresse hat Wortbreite von 16 Bit und kann 64 kByte adressieren
- Speicherzugriffe erfolgen jeweils auf ein Byte
- Drei Speicher:
 - ROM von 4 kByte für den Bootcode
 - SRAM von 8 kByte als Datenspeicher
 - EEPROM von 32 kByte für Programmcode
- Adressbelegung:
 - 0x8000 – 0xffff: EEPROM
 - 0x6000 – 0x7fff: unbelegt
 - 0x4000 – 0x5fff: SRAM
 - 0x1000 – 0x3fff: unbelegt
 - 0x0000 – 0x0fff: ROM



Multiplexing des Datenbusses – Innerhalb eines ICs

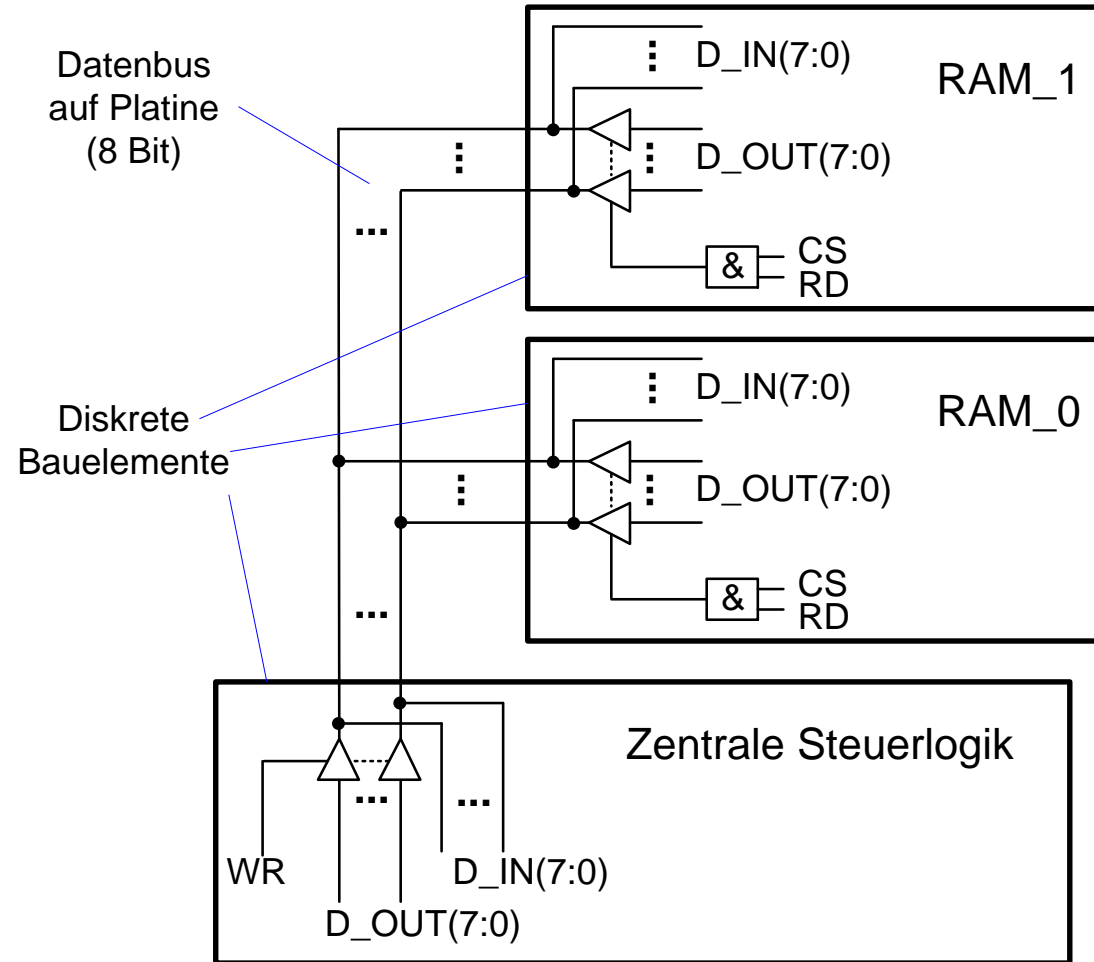
- Getrennte Datenleitungen für Schreib- und Leseoperationen (kein Tri-State)
- Schreiben nur ins SRAM (EEPROM wird gesondert programmiert)
- Multiplexing des Lesebusses zwischen den drei Speichern

Grafik wieder für
fiktiven
Mikrocontroller



Multiplexing des Datenbusses – Diskrete Speicherbausteine

- Gemeinsame Datenleitungen für Schreib- und Leseoperationen
- Ansteuerung mit Tri-State
- Speicher und Steuerlogik müssen für den Betrieb an einem Tri-State-Bus vorgesehen sein
- Tri-State-Treiber der Steuerlogik wird durch das Write-Signal angesteuert
- Tri-State-Treiber der Speicher wird durch UND-Verknüpfung aus Read und jeweiligem Chip Select angesteuert



Übungsaufgaben

Aufgabe 11-8

Ein fiktiver Mikrocontroller hat einen Adressbus mit 12 Bit Wortbreite. Speicherzugriffe erfolgen jeweils auf ein Byte.

a) Wie groß ist der Adressraum?

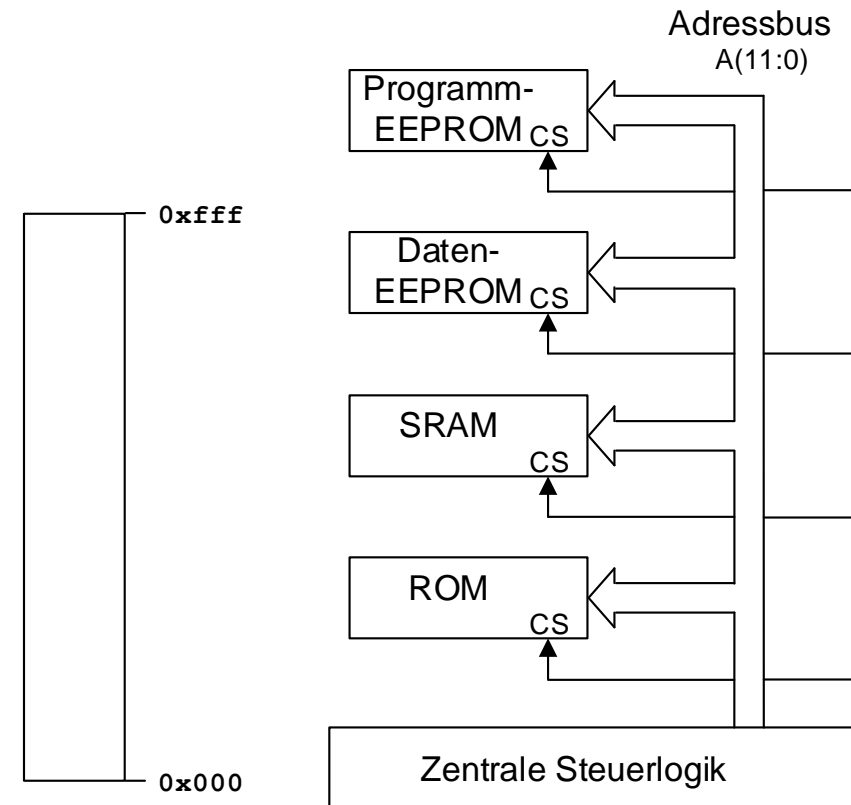
Das System enthält vier Speicher:

- ROM von 512 Byte für Bootcode
- SRAM von 1 kByte als Datenspeicher
- EEPROM von 128 Byte für Daten (von CPU beschreibbar)
- EEPROM von 2 kByte für Programmcode (Über externe Schnittstelle beschreibbar)

b) Schlagen Sie eine Aufteilung des Adressraums vor. Der Bootcode soll bei 0x000 liegen.

c) Welche Logikfunktion hat die Decodierung für CS der Speicher?

Angabe bitte als Funktion, z.B.: $CS_SRAM = (A(11) \& \overline{A(10)}) \vee A(9)$

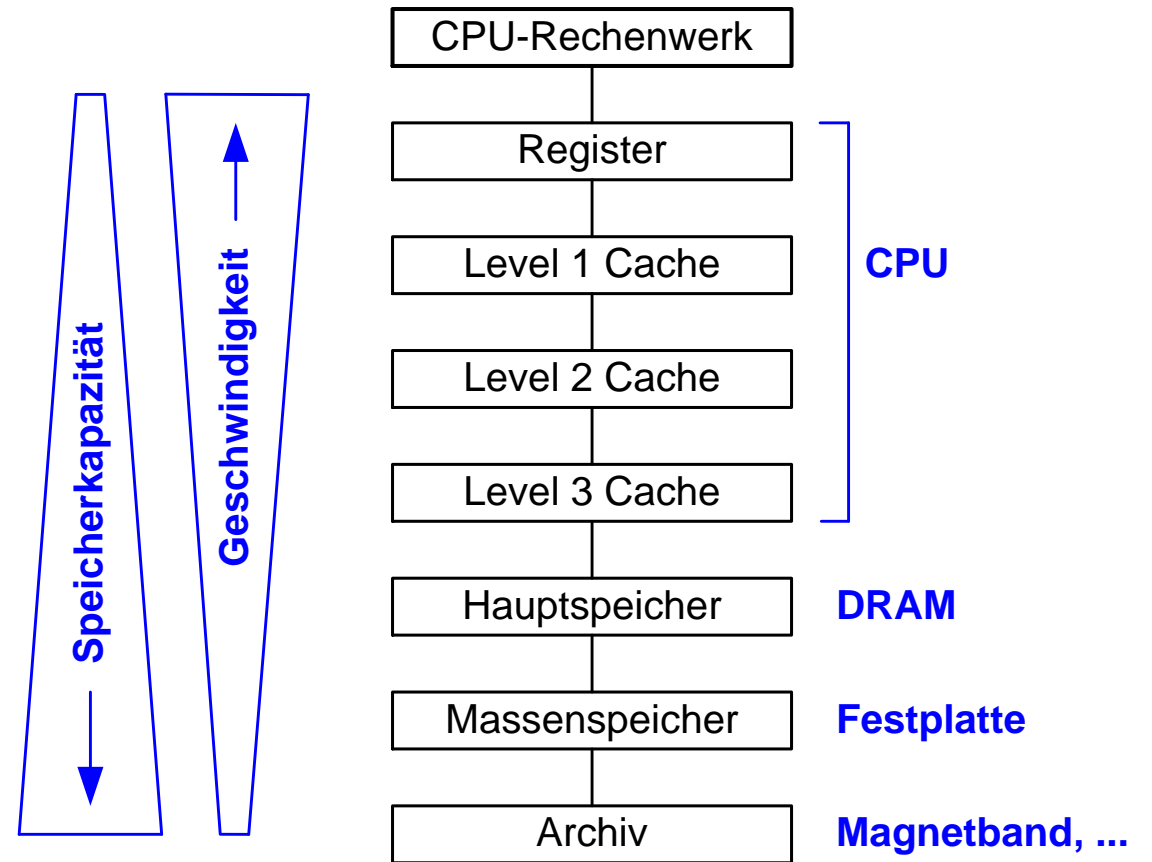


11.6 Speicherhierarchie

- Für Computersysteme wäre ein sehr großer, sehr schneller Speicher ideal
- In der Realität geht aber nur
 - Klein und sehr schnell
 - Mittelgroß und mittelschnell
 - Groß und nicht so schnell
 - Ganz groß und langsam

Frage: Was sind die Gründe dafür?

- Computersysteme nutzen daher eine Speicherhierarchie



Beispiele für Speicherhierarchie

	Server		„Personal Mobile Device“	
Ebene	Größe	Zugriffszeit	Größe	Zugriffszeit
Register	1 KB	0,3 ns	0,5 kB	0,5 ns
Level 1 Cache	64 kB	1 ns	64 kB	2 ns
Level 2 Cache	256 kB	3-10 ns	256 kB	10-20 ns
Level 3 Cache	2-4 MB	10-20 ns	-	-
Hauptspeicher	4-16 GB	50-100 ns	256-512 MB	50-100 ns
Massenspeicher	4-16 TB ¹	5-10 ms	4-8 GB ²	25-50 us
Archiv	100 TB	Minuten	-	-

Anmerkungen:

1) Festplatte 2) Flash-Speicher

Siehe auch: AMD Opteron A1100, Folie 10 sowie
 Supercomputing - Robots Move Data Inside
 NCAR's AMSTAR Digital Storage Library

<https://www.youtube.com/watch?v=d-eWDuEo-3Q>

Quellen:

J. Hennessy, D. Patterson, "Computer Architecture - A Quantitative Approach," 5th edition, Morgan Kaufmann, 2012.

http://business.chip.de/artikel/Datensicherung-Die-besten-Storage-Alternativen-5_39942032.html

Verwendung der Speicherhierarchien

Das modernste Konzept für CPU-Architekturen ist die **RISC-Architektur** (Reduced Instruction Set Computer)

- Register werden direkt von der CPU angesprochen
 - $R5 = R4 + R3$
- Zugriff auf den Hauptspeicher erfolgt durch Kopier-Befehle Register \Leftrightarrow Speicher
 - Load-Store-Architektur
 - Load R4, 0x12345678
- Cache ist Kopie eines Teil des Hauptspeichers
- Massenspeicher und Archiv werden aus dem Programm angesprochen
 - Datei öffnen

Andere Prozessoren haben die **CISC-Architektur** (Complex Instruction Set Computer)

- Mehr Befehle, die allerdings teilweise mehrere Takte benötigen
 - Z.B. Arithmetik auch mit Hauptspeicher
 - ADD A, (HL) Im Register HL steht eine Speicheradresse. Addiere den Wert zum Register A. (Befehl des Z80)

Funktion des Cache

- Ein Cache ist das Abbild eines Teil des Hauptspeichers
- Der Datenzugriff der CPU geht zunächst an den Level 1 Cache
 - Sind die Daten dort nicht vorhanden, werden sie im Level 2 Cache gesucht
 - Falls dort nicht vorhanden, wird auf Level 3 Cache und dann auf den Hauptspeicher zugegriffen
- Ein Cache speichert Daten immer in Blöcken
 - Benachbarte Daten werden oft nacheinander verwendet („**Lokalität**“)
 - Die Verwaltung des Caches ist dadurch einfacher
 - Auch das Laden von Daten kann blockweise einfacher sein
 - Insbesondere bei DRAMs (und Massenspeichern)
- Wesentlich für die Performance der Caches ist
 - Größe und Aufteilung in Blöcke
 - Strategie zum Nutzen und Freigeben von Blöcken
- Es gibt auch Festplatten-Caches, für die ähnliches gilt

Exkurs: Datenunsicherheit durch Cache

Der Cache ist Angriffspunkt für die Prozessor-Schwachstellen Spectre und Meltdown

- Die Speicherstelle X enthält ein geheimes Byte und kann nicht gelesen werden
- Es gibt ein erlaubtes Array A mit 256 Einträgen

Funktionsprinzip (vereinfacht):

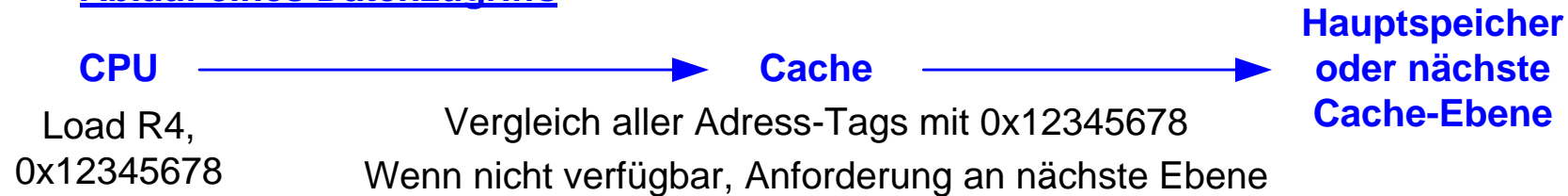
- Der Cache wird gelöscht
- Es erfolgt ein Zugriff auf A(X), also geheime Information als Array-Index
- Der Prozessor lädt A(X) in den Cache
 - Spekulativer Zugriff, um Rechengeschwindigkeit zu erhöhen
- Der Prozessor erkennt, Zugriff auf X ist verboten und bricht die Operation ab
- Zugriff auf A(0):
 - Wenn Zugriff schnell geht, ist A(0) im Cache und der geheime Wert ist 0
 - Ansonsten neue Ausführung und Probieren ob Wert gleich 1, 2, 3, ...

Zum Weiterlesen: „How the spectre and meltdown hacks really worked,”
N. Abu-Ghazaleh, D. Ponomarev, D. Evtushkin, IEEE Spectrum, March 2019.

Aufbau eines Cache

- Jeder Block eines Cache besteht aus drei Teilen
 - Daten
 - Adressbereich, der gerade gespeichert ist; Bezeichnung: Tag
 - Status, z.B. ob die Daten gültig sind (Valid Bit)

Ablauf eines Datenzugriffs



Tag	Daten	Flag
0x01012300	0x0000, 0x0001, ...	1
0xaf123400	0xffff, 0xfffe, ...	1
0x87654200	0x0123, 0x0124, ...	1
...

- Datenblöcke beginnen immer bei festen Addresspositionen im Abstand der Blockgröße
- Dadurch müssen nicht alle Bits einer Adresse durchsucht werden

Aufgabe: Cache

Aufgabe 11-9

Ein CPU-System hat einen Adressraum von 32 bit. Je Adresse wird ein Byte gespeichert. Es ist ein Cache mit 256 Blöcken zu je 256 Byte vorhanden.

- a) Wie viele Adressen kann die CPU ansprechen?
- b) Wie groß ist der Speicherplatz des Cache?
- c) Welcher Anteil des möglichen Hauptspeichers kann im Cache gespeichert werden?

Die Cacheblöcke beginnen jeweils bei festen Addresspositionen im Abstand der Blockgröße. Dadurch muss nicht die gesamte Adresse verglichen werden.

- d) Wie viele Bits der Adresse müssen verglichen werden?

Adressvergleich beim Cache

- Daten des Hauptspeichers können in jedem Cache-Block gespeichert werden
 - Bezeichnung: **Vollassoziativer Cache**
- **Problem:** Bei einem Cache mit 256 Blöcken muss CPU-Adresse mit 256 Tags verglichen werden.
 - ➔ Hoher Aufwand

Erste Lösung

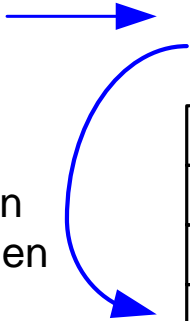
- Feste Zuordnung von Adressen zu Cache-Block
 - Bezeichnung: **Nicht-assoziativer Cache** („direct-mapped“)
- **Problem:** Bei ungünstigem Zugriffsmuster wird stets der gleiche Cache-Block benötigt, obwohl der Cache ansonsten leer ist

Nicht-assoziativer Cache

Load R4,
0x12345678

Bit 7:0 - Offset: Daten stehen an Position 0x78 des Blocks
Bit 15:8 - Index: Daten können nur in Block 0x56 stehen

Überprüfen
einer einzigen
Adresse



Tag	Daten	Flag
0x0101	0x0000, 0x0001, ...	1
...	...	1
0x8765	0x0123, 0x0124, ...	1
...

Adressvergleich beim Cache (II)

Mittelweg als verbesserte Lösung

- Zuordnung von Adressen zu mehreren, wenigen Cache-Blöcken
 - Bezeichnung: **Set-assoziativer Cache**
- Beispiel: 4 Cache-Blöcke pro Teiladresse
 - Vergleich von nur 4 Adressen
 - Bezeichnung: 4-way-set-associative, 4-fach satzassoziativ

Die Cache-Adresse teilt sich damit in **drei Teile**, und zwar in der Reihenfolge **von LSB nach MSB**:

- **Offset**: Position der Speicherstelle im Cacheblock
- **Index**: Auswahl von Cache-Zeile oder Block (nicht bei vollassoziativem Cache)
- **Tag**: Vordere Adressposition, die verglichen werden muss

Beispiel: Adressvergleich beim Set-assoziativem Cache

- 4-fach satzassoziativer Cache
 - Blockgröße 64 Byte
 - 256 Blöcke
- also:
- 64 Bereiche mit je 4 Blöcken
-
- Adressbereiche:
 - Offset: 6 bit
 - Index: 6 bit
 - Jeweils für 64 Adressen

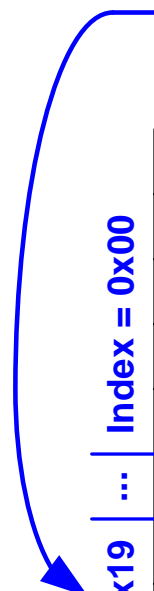
Load R4, 4-way-set-associative Cache
0x12345678



Bit 5:0 - Offset 0b11 1000 = 0x38

Bit 11:6 - Index: 0b0110 01 = 0x19

Bit 31:12 - Tag: 0x12345



Tag	Daten	Flag
0x01010	0x0000, 0x0001, ...	1
0x23456	0x0011, 0x0012, ...	1
0x00000	0x0000, 0x0000, ...	0
0x00000	0x0000, 0x0000, ...	0
...
0x87654	0x0123, 0x0124, ...	1
0xfecdb	0x0133, 0x0134, ...	1
0x12345	0x4567, 0x4568, ...	1
0xb9753	0xabcd, 0xabce, ...	1
...

Adressvergleich beim Cache (II)

Aufgabe 11-10

Ein CPU-System (Adressraum 32 bit) hat einen Cache von 256 Blöcken zu je 256 Byte. Der Cache ist 4-fach satzassoziativ.

- a) Wie viele Teiladressen gibt es jeweils für Offset und Index?
- b) Wie ist die Aufteilung der 32-bit-Adresse in Offset, Index, Tag?

Aufgabe 11-11

Ein Cache hat die gleiche Größe wie in Aufgabe 11-10. Er ist 2-fach satzassoziativ. Beantworten Sie wieder Teilaufgaben a) und b).

Aufgabe 11-12

Ein Cache hat die gleiche Größe wie in Aufgabe 11-10. Er ist vollassoziativ. Beantworten Sie wieder Teilaufgaben a) und b).

Aufgabe 11-13

Ein Cache hat die gleiche Größe wie in Aufgabe 11-10. Er ist nicht-assoziativ („direct-mapped“). Beantworten Sie wieder Teilaufgaben a) und b).

Miss beim Cache-Zugriff

- Daten die nicht im Cache gefunden werden (**Cache-Miss**), werden aus der nächsten Ebene geladen
 - Das Laden erfolgt zunächst in den Cache
 - Von dort kann die CPU zugreifen

Drei Gründe für einen Cache-Miss („Three-C-Model“)

- Compulsory (unvermeidlich):
 - Der erste Zugriff auf einen Speicherbereich kann nicht im Cache sein
- Capacity (Kapazität):
 - Der Block war schon im Cache, wurde aber wegen begrenzter Kapazität entfernt
- Conflict (Konflikt):
 - Der Block war schon im Cache, eigentlich wäre noch Kapazität frei, aber unter dem Index sind alle Blöcke belegt

Frage: Bei welcher Cache-Organisation kann kein Conflict-Cache-Miss auftreten?

Ersetzungsstrategie beim Cache

- Wenn der Cache belegt ist, muss ein anderer Block ersetzt werden
 - **Problem:** Welcher Block soll ersetzt werden?

Drei Strategien

- Random: Zufälliger Block
 - Einfach zu implementieren
- Least Recently Used (LRU): Block der am längsten nicht benutzt wurde
 - Nutzt Lokalität der Daten aus, also Annahme, das häufig benachbarte Daten benötigt werden
 - Aufwändig nachzuverfolgen
- First In, First Out (FIFO): Ältester Block wird überschrieben
 - Versucht LRU mit weniger Aufwand anzunähern

Ersetzungsstrategie beim Cache (II)

- Eine vereinfachte LRU-Variante ist **Pseudo-LRU**:
 - Alle Blöcke haben ein Zugriffs-Bit
 - Beim Zugriff wird das Bit auf ,1' gesetzt
 - Wenn alle Blöcke eines Sets auf ,1' sind, werden alle auf ,0' zurückgesetzt, ausgenommen die zuletzt gesetzte ,1'
 - Beim Löschen eines Blocks wird ein Block mit Zugriffs-Bit ,0' zufällig ausgewählt

Aufgabe 11-14

Simulieren Sie das Pseudo-LRU-Verfahren. Für eine Teiladresse gibt es 4 Cache-Blöcke. Auf die **markierte Zelle** wird zugegriffen. Tragen Sie in die Kästen 0 oder 1 ein.

		Zeit →															
Block 0	1	<div></div>											<div></div>				<div></div>
Block 1	0		<div></div>				<div></div>				<div></div>						
Block 2	0				<div></div>		<div></div>		<div></div>						<div></div>		
Block 3	0			<div></div>		<div></div>			<div></div>						<div></div>		

Vergleich der Cache-Strategien

- Die Performance ist abhängig von den Datenzugriffen der Software
- In Hennessy, Patterson wird ein Vergleich für einen Satz an Benchmark-Programmen angegeben (aus SPEC2000)
 - Messgröße ist „Misses per 1000 Instructions“
 - Blockgröße jeweils 64 Byte

	2-fach assoziativ			4-fach assoziativ			8-fach assoziativ		
Cache-Größe	LRU	Random	FIFO	LRU	Random	FIFO	LRU	Random	FIFO
16 KB	114,1	117,3	115,5	111,7	115,1	113,3	109,0	111,8	110,4
64 KB	103,4	104,3	103,9	102,4	102,3	103,1	99,7	100,5	100,3
256 KB	92,2	92,1	92,5	92,1	92,1	92,5	92,1	92,1	92,5

Aufgabe 11-15

Interpretieren Sie die Werte. Betrachten Sie dazu auch die Zugriffszeiten von Folien 22.

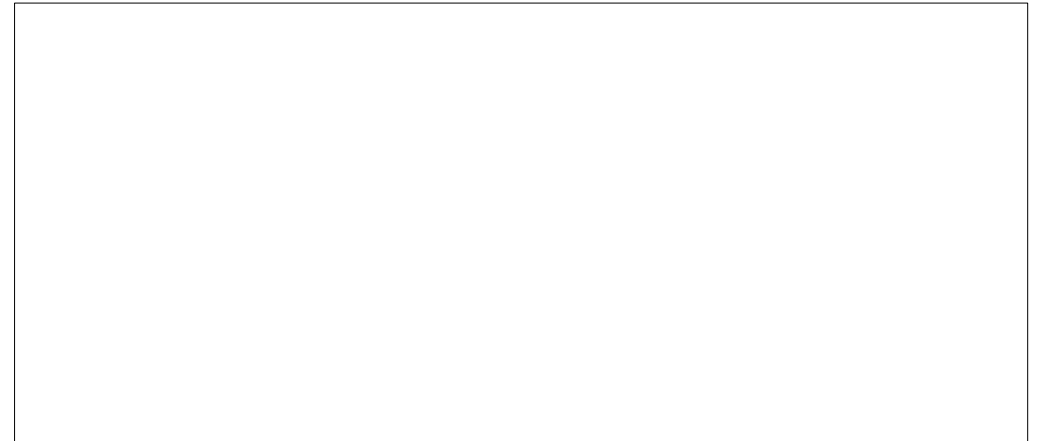
Quelle: J. Hennessy, D. Patterson, "Computer Architecture - A Quantitative Approach," 5th edition, Morgan Kaufmann, 2012.

Schreiben in den Cache

- Die meisten Speicheroperationen sind Leseoperationen
 - 7% Schreibzugriffe für gemischten Cache (Instruktionen und Daten)
 - 28% Schreibzugriffe falls reiner Datencache
- Zwei Strategien für Schreibzugriffe
 - Write-Through: Schreibzugriffe erfolgen in den Cache und den Hauptspeicher
 - ➔ Vorteil: Block kann bei Bedarf einfach gelöscht werden
 - Write-Back: Schreibzugriffe erfolgen nur in den Cache und erst beim Löschen in den Hauptspeicher
 - ➔ Vorteil: Eventuell weniger Schreibvorgänge
- Bei Write-Back wird ein veränderter Block durch ein „Dirty Flag“ gekennzeichnet
 - Steuereinheit weiß, dass der Block zurückgeschrieben werden muss
- Beide Strategien werden verwendet:
 - Write-Through ist einfacher und für Multiprozessoren sinnvoll
 - Write-Back ist effizienter bezüglich Datenrate zum Speicher und Verlustleistung

Tag-Schreibweise für Übungsaufgaben

- Für den Tag werden die hinteren Bits von Index und Offset weggelassen
 - Beispiel Folie 5-29
 - Adresse 0x12345678
 - Je 6 Bit für Index und Offset
 - 12 Bit weglassen ergibt 0x12345
 - Aber wenn Index und Offset kein vielfaches von 4 bit sind, muss man umrechnen
 - Adresse 0x12345678
 - Je 5 Bit für Index und Offset
 - 10 Bit weglassen ergibt ...?
 - Für Übungsaufgaben **vereinfachte Schreibweise** möglich
 - Letzte 10 bit zu Null setzen, 0x weglassen
 - Adresse 12345400
- **Achtung:** Sonderbehandlung für drittletztes Byte
- Zwei Bit auf Null setzen, entspricht Abrunden auf vielfaches von 4, also: **6 → 4**



Aufgabe: Cache

Aufgabe 11-16

Ein CPU-System hat einen Adressraum von 16 bit. Der Zugriff erfolgt immer byte-weise. Es ist ein Cache mit 8 Blöcken zu je 16 Byte vorhanden.

Der Cache ist als **vollassoziativer Cache** organisiert. Nutzen Sie, falls nötig, als Ersetzungsstrategie FIFO (First In, First Out).

- Wie teilen sich die 16 bit der Adresse in Offset, Index, Tag?
- Ein Datenzugriff erfolgt auf die rechts angegebenen Adressen. Wie lauten Tag und (falls vorhanden) Index.
- Geben Sie an, ob die Daten sich im Cache befinden (Cache-Hit) oder nicht (Cache-Miss). Tragen Sie jeweils H (Hit) oder M (Miss) ein.

Zu Beginn ist der Cache leer. Die vereinfachte Tag-Schreibweise kann verwendet werden.

Datenzugriffe

Adresse (in hex)	Tag / Index (in hex)	Hit / Miss
0000		
0001		
0011		
0023		
0034		
0045		
0086		
ffff		
fffe		
0007		
001a		
a04b		
a08c		
a108		
a143		
a100		

↓ Zeit

Hilfsblatt zu Cache-Aufgabe

Zeit →

Index (falls nötig)	Block	Tag	Neuer Tag	Neuer Tag
	0			
	1			
	2			
	3			
	4			
	5			
	6			
	7			

- Kurzanleitung:
- Teilen Sie, falls nötig, die Cache-Blöcke für den Index auf
 - Schreiben Sie den Tag bei einem Zugriff in das Feld
 - Nutzen Sie beim Überschreiben eines Blockes das nächste Feld

Aufgabe: Cache

Aufgabe 11-17

Ein CPU-System hat einen Adressraum von 16 bit. Der Zugriff erfolgt immer byte-weise. Es ist ein Cache mit 8 Blöcken zu je 16 Byte vorhanden.

Der Cache ist als **nicht-assoziativer Cache** organisiert. Nutzen Sie, falls nötig, als Ersetzungsstrategie FIFO (First In, First Out).

- Wie teilen sich die 16 bit der Adresse in Offset, Index, Tag?
- Ein Datenzugriff erfolgt auf die rechts angegebenen Adressen. Wie lauten Tag und (falls vorhanden) Index.
- Geben Sie an, ob die Daten sich im Cache befinden (Cache-Hit) oder nicht (Cache-Miss). Tragen Sie jeweils H (Hit) oder M (Miss) ein.

Zu Beginn ist der Cache leer. Die vereinfachte Tag-Schreibweise kann verwendet werden.

Datenzugriffe

Adresse (in hex)	Tag / Index (in hex)	Hit / Miss
0000		
0001		
0011		
0023		
0034		
0045		
0086		
ffff		
fffe		
0007		
001a		
a04b		
a08c		
a108		
a143		
a100		

↓ Zeit

Hilfsblatt zu Cache-Aufgabe

Zeit →

Index (falls nötig)	Block	Tag	Neuer Tag	Neuer Tag
	0			
	1			
	2			
	3			
	4			
	5			
	6			
	7			

- Kurzanleitung:
- Teilen Sie, falls nötig, die Cache-Blöcke für den Index auf
 - Schreiben Sie den Tag bei einem Zugriff in das Feld
 - Nutzen Sie beim Überschreiben eines Blockes das nächste Feld

Aufgabe: Cache

Aufgabe 11-18

Ein CPU-System hat einen Adressraum von 16 bit. Der Zugriff erfolgt immer byte-weise. Es ist ein Cache mit 8 Blöcken zu je 16 Byte vorhanden.

Der Cache ist als **2-fach satzassoziativer Cache** organisiert. Nutzen Sie, falls nötig, als Ersetzungsstrategie FIFO (First In, First Out).

- Wie teilen sich die 16 bit der Adresse in Offset, Index, Tag?
- Ein Datenzugriff erfolgt auf die rechts angegebenen Adressen. Wie lauten Tag und (falls vorhanden) Index.
- Geben Sie an, ob die Daten sich im Cache befinden (Cache-Hit) oder nicht (Cache-Miss). Tragen Sie jeweils H (Hit) oder M (Miss) ein.

Zu Beginn ist der Cache leer. Die vereinfachte Tag-Schreibweise kann verwendet werden.

Datenzugriffe

Adresse (in hex)	Tag / Index (in hex)	Hit / Miss
0000		
0001		
0011		
0023		
0034		
0045		
0086		
ffff		
fffe		
0007		
001a		
a04b		
a08c		
a108		
a143		
a100		

↓ Zeit

Hilfsblatt zu Cache-Aufgabe

Zeit →

Index (falls nötig)	Block	Tag	Neuer Tag	Neuer Tag
	0			
	1			
	2			
	3			
	4			
	5			
	6			
	7			

- Kurzanleitung:
- Teilen Sie, falls nötig, die Cache-Blöcke für den Index auf
 - Schreiben Sie den Tag bei einem Zugriff in das Feld
 - Nutzen Sie beim Überschreiben eines Blockes das nächste Feld

5.4 Cache-Optimierung und Verlustleistung

- Grundsätzlich ist ein Cache positiv für Performance und Verlustleistung
 - Zugriffe sind lokal mit höherer Geschwindigkeit und weniger Schaltaktivität

Verschiedene Design-Entscheidungen können getroffen werden

➔ Für alle Entscheidungen kann die Bilanz insgesamt positiv oder negativ sein

- Größere Blöcke, um Miss-Rate zu verringern
 - Ausnutzen von Lokalität
 - Verzögerung bei Miss kann steigen, da Nachladen größer Blöcke eventuell langsamer
 - Geringere Anzahl an Tags reduzieren Verlustleistung etwas
 - Weniger Vergleiche nötig
- Größerer Cache, um Miss-Rate zu verringern
 - Längere Zugriffszeiten
 - Höhere statische und dynamische Verlustleistung

Cache-Optimierung und Verlustleistung (II)

- Höhere Assoziativität, um Miss-Rate zu verringern
 - Zugriffszeit und Verlustleistung steigen
- Mehrere Cache-Ebenen, um Zeitverlust bei Miss zu verringern
 - Gute Möglichkeit, das System zu balancieren
 - Kleine schnelle Puffer nahe an CPU
 - Größere, langsamere Puffer nahe am Hauptspeicher
 - Design wird komplexer
- Bevorzugung von Lese-Operationen gegenüber Schreib-Operationen, um Zeitverlust bei Miss zu verringern
 - Problem, falls Daten aus Schreibpuffer sofort wieder gelesen werden
 - Kann durch Adressvergleich abgefangen werden
 - Geringer Einfluss auf Verlustleistung

Umfangreiche Informationen und Beispiele in J. Hennessy, D. Patterson, "Computer Architecture - A Quantitative Approach," 5th edition, Morgan Kaufmann, 2012.