GGI II Zusammenfassung

Marius Geis

17. April 2011

1 Historischer Überblick

2 Aufgbau und Funktion eines Digitalrechners

2.1 Aufbau eines Arbeitsplatzrechners

2.2 Kennwerte eines Digitalrechners

3 Informationsdarstellung und Codierung

3.1 Codierung

- **Information:** Bedeutung eines Sachverhalts
- **Nachricht:** Darstellung einer Information
- **Alphabet** A**:** endliche Menge von Zeichen
- A^*: Menge aller Wörter über A
- A^m: Menge aller Wörter der Länge m über A
- **Wort** $a \in A^*$**:** Folge verketteter Zeichen
- **Kardinalität:** Anzahl verschiedener Wort $A^m = n^m$ mit n-Zeichen Alphabet
- **Codierung:** Meistens: Abbildung eines Zeichen durch ein Wort

  $$\kappa : A \to B^*$$

- **Code:**
  - Eine Vorschrift für die eindeutige Zuordnung (Codierung) der Zeichen eines Zeichenvorrats zu denjenigen eines anderen Zeichenvorrats (Bildmenge).
  - Zeichenvorrat der Bildmenge

3.2 Informationsgehalt einer Nachricht

- **Informationsgehalt:**

  $$I(a_j) = -\log p_j$$

- Einheit: **bit**
- **Entropie:**

  $$H = -\sum_{j=1}^{n} p_j \cdot \log p_j$$

- **Maximale Entropie:**

  $$H_0 = \log n$$

- **Redundanz:**

$$R = H_0 - H$$

- **Relative Redundanz:**

$$r = \frac{R}{H_0}$$

3.3 Wichtige Codes

3.3.1 Optimalcodes (Codeworte unterschiedlicher Länge)

- **Fano-Bedingung:** Kein Wort des Codes darf Anfang eines anderen Codewortes sein.

3.3.2 Block-/Wortcodes (Codeworte fester Länge)

- **ASCII:** American Standard Code for Information Interchange
- **Tetradencodes:** binäre Darstellung von Dezimalzahlen
  - **BCD-Code:** Die ersten 10 sind Zahlen, die letzen 6 Pseudotetraden
  - **Aiken-Code:** Dier ersten und letzen 5 sind Zahlen, dazwischen Pseudotetraden
  - **Exzess-3 Code:** Die ersten und letzen 3 sind Pseudotetraden, dazwischen Zahlen
- **Gray-Code:** Aufeinanderfolgende Zahlen unterscheiden sich nur in einer Bitstelle
- m**-aus-**n**-Codes:** m Einsen in einem n-Zeichen Wort
- **Biquinär-Code:** 2-er Teil und 5-er Teil

3.4 Erkennung un dKorrektur von Übertragungsfehlern

3.4.1 Zur Rolle der Hamming-Distanz

- **Hamming-Distanz** d**:**

$$d = \text{ Anzahl verschiedener Bitstellen}$$

- **Hamming-Distanz** d_m**:** kleinste Auftretende Hamming-Distanz d
- Erkennbare Fehler:

$$e = (d_m - 1)$$

- Korrigierbare Fehler:

$$k = \left\lfloor \frac{e}{2} \right\rfloor$$

- Wahrscheinlichkeit für das Aufterten eines k-Fachen Fehler in einem m-stelligen Wort

$$P(k) = \binom{m}{k} p^k (1-p)^{m-k}$$

3.4.2 Quer-und Längsparitätsprüfung

- **gerade Parität:** Einsen zu einer geraden Anzahl ergänzen
- **ungerade Parität:** Einsen zu einer ungeraden Anzahl ergänzen

4 Zahlendarstellung

4.1 Polyadische Zahlendarstellung

- **Polyadische Zahlendarstellung** (mit **Basis** B, **Ziffern** z_i, n Stellen):

$$Z = \sum_{i=0}^{n-1} z_i B^i$$

- **Ziffernschreibweise:**

$$Z = (z_{n-1} z_{n-2} \dots z_2 z_1 z_0)_B$$

- **Hornerschema:**

$$Z = ((z_{n-1}B + z_{n-2})B + \ldots + z_1)B + z_0$$

- rationale Zahlen:

$$\begin{aligned} Z &= \sum_{i=-k}^{n-1} z_i B^i \\ &= (z_{n-1} z_{n-2} \dots z_2 z_1 z_0, z_{-1} z_{-2} \dots z_{-k})_B \\ &= ((z_{n-1}B + z_{n-2})B + \ldots + z_1)B + z_0 + (z_{-1} + \ldots + (z_{-(k-1)} + z_{-k}B^{-1})B^{-1})B^{-1} \end{aligned}$$

4.2 Umwandlung in Zahlensysteme mit andere Basis

4.2.1 Unwandlung in Dezimalzahlen

- Summenformel

$$Z = \sum_{i=-k}^{n-1} z_i B^i$$

- Hornerschema

$$Z = ((z_{n-1}B + z_{n-2})B + \ldots + z_1)B + z_0 + (z_{-1} + \ldots + (z_{-(k-1)} + z_{-k}B^{-1})B^{-1})B^{-1}$$

4.2.2 Umwandlung von Dezimalzahlen

- **Umwandlung des ganzzahligen Anteils**
  - 1. Methode: Division durch 2^n und anschließende Subtraktion von 2^n. Erste Ziffer ist MSB.
  - 2. Methode: Horner-Schema. Erste Ziffer ist LSB.
- **Umwandlung des gebrochenen Anteils**
  - nicht immer exakt möglich
  - Horner-Schema
  - Fortgesetze Multiplikation des gebrochenen Anteils mit der Zielbasis und Abspaltung der jeweils vordersten Ziffer
  - Erste Ziffer ist MSB

4.2.3 Dual-,Oktal- und Hexadezimalzahlen

- 3 bzw 4 Dualzifferm können zu einer Oktal- bzw. Hexadezimalzahl zusammengefasst werden

4.3 Zahlendarstellung im Digitalrechner

4.3.1 Darstellung ganzer Zahlen

- Darstellung negativer Zahlen
  - Darstellung mit Hilfe des Vorzeichens
    - Zwei Nullen
    - Rechenwerk muss addieren sowie subtrahieren können
  - Darstellung im **Einerkomplement**
    - Definition:

$$K_1(Z) = (2^n - 1) - Z = (11\ldots1)_2 - Z$$

    - Bitweises Komplementieren
    - Zwei Nullen
  - Darstellung im **Zweierkomplement**
    - Definition:

$$K_2(Z) = 2^n - Z = K_1(Z) + 1$$

    - Eindeutig
    - Zweierkomplement des Zweierkomplements ergibt Normaldarstellung

$$K_2(K_2(Z)) = 2^n - (2^n - Z) = Z$$

- Ganzzahlenarithmetik
  - Addition
  - Subtraktion
  - Subtraktion durch Addition des Zweierkomplements
    - Beweis

$$Z_1 - Z_2 = Z_1 - Z_2 + 2^n = Z^1 + (2^n - Z_2) = Z_1 + K_2(Z_2)$$

    - Positives Ergebnis: Übertrag 1
    - Negatives Ergebnis: Übertrag fehlt und das höchstwertige Bit ist gleich 1
  - Multiplikation durch fortgesetze Addition und Verschiebung
  - Division: fortgesetzte Subtraktion und Verschiebung

4.3.2 Festkomma-Darstellung

- n-stellige Dualzahl, wobei das Komma an beliebiger, fester Stelle steht
- Zahlenbereich sehr beschränkt

4.3.3 Gleitkommadarstellung

- Form (M Mantisse, E Exponent, B Basis):

$$Z = M \cdot B^E$$

- Normalisierung ermöglicht eindeutige Darstellung
- Charakteristik ermöglicher bitweisen Vergleich und beschränkt Rechnung mit Exponenten auf positive Zahlen

$$C = E + \left(\frac{1}{2}B^c - 1\right) = E + \left(B^{c-1} - 1\right)$$

- IEEE-Gleitkommaformat 754

$$Z = (-1)^{VZ} \cdot M \cdot 2^{C-(2^7)-1}$$

- $C = E + (2^7 - 1)$
- $VZ = 0$ entspricht +, $VZ = 1$ entspricht -
- Normalisierung: Mantisse hat die Form $(1, ...)_2$, MSB ist hidden bit
- Null: nur Nullen
- $C = 255$ ist NaN oder $\pm\infty$

- Gleitkommaarithmetik
  - Addition und Subtraktion
    1. Angleichen der Exponenten durch Verschiebung des betragsmäßig kleineren Summanden
    2. Addition bzw. Subtraktion der Mantissen
    3. Normalisierung des Ergebnisses
  - Multiplikation und Division
    1. Multiplikation bzw. Division der Mantissen
    2. Addition bzw. Subtraktion der Exponenten
    3. Vorzeichen setzen
    4. Normalisierung des Ergebnisses

- Probleme der Gleitkommadarstellung
  - Wegen Ungenauigkeiten nie $Z = 0$ abfragen
  - Assoziativgesetz gilt nicht mehr uneingeschränkt

5 Schaltunglogik

5.1 Zwecke und Ziele

- Theoretische Grundlage für den Schaltungsenfwurf

5.2 Boolesche Algebra

- Operationen Φ und Ψ, Menge B
- Quadrupel $(B, \Phi, \Psi, ^-)$ heißt **Boolesche Algebra**
- **Huntingtonschen Axiome**

1. Kommutativgesetz

$$a\Phi b = b\Phi a$$
$$a\Psi b = b\Psi a$$

2. Distributivgesetz

$$a\Phi(b\Psi c) = (a\Phi b)\Psi(a\Phi c)$$
$$a\Psi(b\Phi c) = (a\Psi b)\Phi(a\Psi c)$$

3. Neutrale Elemente: $n \in B$ Nullelement, $e \in B$ Einselement

$$a\Phi e = a$$
$$a\Psi n = a$$

4. Komplement

$$a\Psi\overline{a} = e$$
$$a\Phi\overline{a} = n$$

- Abgeleitete Sätze:

- Assoziativgesetz

$$(a\Phi b)\Phi c = a\Phi(b\Phi c)$$
$$(a\Psi b)\Psi c = a\Psi(b\Psi c)$$

- Idempotenzgesetz

$$a\Phi a = a$$
$$a\Psi a = a$$

- Absorptionsgesetz

$$a\Psi(a\Phi b) = a$$
$$a\Phi(a\Psi b) = a$$

- DeMorgansche Gesetze

$$\overline{(a\Psi b)} = \overline{a}\Phi\overline{b}$$
$$\overline{(a\Phi b)} = \overline{a}\Psi\overline{a}$$

5.3 Beispiele Boolescher Algebren

5.3.1 Mengenalgebra

- Ψ Vereinigung $\cup$, Φ Durchschnitt $\cap$, n leere Menge $\varnothing$, e Allmenge M
- Gilt für Potenzmengen von M
- Venndiagramme

5.3.2 Schaltalgebra

- $B = \{0, 1\}$
- Ψ - Oder-Verknüpfung bzw. Disjunktion $(+, \vee)$
- Φ - Und-Verknüpfung bzw. Konjunktion $(\cdot, \wedge)$
- $e = 1$ (wahr)
- $n = 0$ (falsch)
- Komplementbildung - Negation $(\overline{a}, \neg a)$
- Huntingtonsche Axiome
  - Kommutativgesetz

$$a \cdot b = b \cdot a$$
$$a + b = b + a$$

  - Distributivgesetz

$$a \cdot (b + c) = (a \cdot b) + (a \cdot c)$$
$$a + (b \cdot c) = (a + b) \cdot (a + c)$$

  - Neutrale Elemente

$$a \cdot 1 = a$$
$$a + 0 = a$$

  - Komplement

$$a + \overline{a} = 1$$
$$a \cdot \overline{a} = 0$$

- Aus den Axiomem abgeleitete Sätze
  - Assoziativgesetz

$$(a \cdot b) \cdot c = a \cdot (b \cdot c)$$
$$(a + b) + c = a + (b + c)$$

  - Idempotenzgesetz

$$a \cdot a = a$$
$$a + a = a$$

  - Absorptionsgesetz

$$a + (a \cdot b) = a$$
$$a \cdot (a + b) = a$$

  - DeMorgansche Gesetze

$$\overline{(a + b)} = \overline{a} \cdot \overline{b}$$
$$\overline{(a \cdot b)} = \overline{a} + \overline{a}$$

  - Shannon-Gesetze

$$\overline{a + b + c + \ldots + d} = \overline{a} \cdot \overline{b} \cdot \overline{c} \cdot \ldots \cdot \overline{d}$$
$$\overline{a \cdot b \cdot c \cdot \ldots \cdot d} = \overline{a} + \overline{b} + \overline{c} + \ldots + \overline{d}$$

  - Reduktionsgesetze

$$a \cdot (\overline{a} + b) = a \cdot b$$
$$a + \overline{a} \cdot b = a + b$$
$$(a + b) \cdot (\overline{a} + c) = a \cdot c + \overline{a} \cdot b$$

  - Doppelte Negation

$$\overline{\overline{a}} = a$$

- Vereinbahrung: Punktrechnung vor Strichrechnung

$$a + (b \cdot c) = a + b \cdot c = a + bc$$

5.4 Boolesche Funktionen

- **Schaltfunktion**

$$f : B^n \to B^n$$

- n-stellige **Boolesche Funktion**

$$f : B^n \to B$$

5.4.1 Darstellung von Booleschen Funktionen

- Jede Boolesche Funktion ist unter alleiniger Verwender von NOR oder NAND darstellbar
- Darstellungmöglichkeiten
  - **Funktionstabelle**
  - **Schalterdiagramm**
  - **Schaltnetz**
  - Venndiagramme

- Programme
- Formel
  - **Disjunktive Form (DF):** Disjunktion von Konjunktionen
  - **Disjunktive Normalform (DNF):** Disjunktion von Mintermen
  - **Minterm:** Konjunktion, die alle Eingangsvariablen enthält
  - DNF als Einsen in Funktionstabelle ablesbar
  - **Konjunktive Form (KF):** Konjunktion von Disjunktionen
  - **Konjunktive Normalform (KNF):** Konjunktion von Maxtermen
  - **Maxterm:** Disjunktion, die alle Eingangsvariablen enthält
  - KNF als Nullen in Funktionstabelle ablesbar
- **Karnaugh-Veitch-Diagramm**

5.4.2 Minimierung Boolescher Funktionen

5.4.3 Behandlung unvollständig definierter Schaltfunktionen

- **don't care**
- **Quine-McCluskey-Verfahren**
  - Gruppiere Minterme mit i nicht-negierten Variablen zu einer Klasse K_i zusammen.
  - Fasse Minterme die sich nur in einer Variable unterscheiden zusammen.
  - Verfahre analog mit bereits zusammengefassten Termen.
  - Stelle Überdeckungsmatrix auf
  - Stelle Restmatrix auf

6 Logische Schaltungen

6.1 Technische Realisierung Logischer Funktionen

6.2 Standard-Schaltnetze

- **Schaltnetz:** Gatterschaltung, bei der die Ausgangssignale nur von den Eingangssignalen abhängen (keine Rükführungen).

6.2.1 Addierer

- **Halbaddierer**

$$\begin{aligned} s &= \overline{x}_1 x_0 + x_1 \overline{x}_0 = x_1 \oplus x_0 \\ &= (x_0 + x_1)\overline{\ddot{u}} \\ \ddot{u} &= x_1 x_0 \end{aligned}$$

- **Volladdierer**

$$\begin{aligned} s &= (x_1 \oplus x_0) \oplus \ddot{u}' \\ \ddot{u} &= (x_1 \oplus x_0)\ddot{u}' + x_1 x_0 \end{aligned}$$

  - Summe: 1 oder 3 Eingangsvariablen = 1
  - Übertrag: 2 oder 3 Eingangsvariablen = 1
  - Kann aus 2 Halbaddierern gebaut werden:
    - Erster HA addiert $x_0 + x_1$, liefert $s', \ddot{u}'$
    - Zweiter HA addiert $s', \ddot{u}'''$, liefert $\underline{s}, \ddot{u}''$
    - Abgehender Übertrag $\underline{\ddot{u}} = \ddot{u}' + \ddot{u}''$

6.2.2 Schaltketten

- **Schaltkette:** Mehrstufiger Schaltnetz, in dem mehrere Teilschaltnetze gleicher Struktur kettenartig hintereinandergeschaltet sind
- **Vergleicher:** (Übertrag kommt von LSB)

$$\begin{aligned} \ddot{u}_n &= x > y \\ \ddot{u}_{i+1} &= (x_i \overline{y}_i) + \ddot{u}_i(\overline{\overline{x}_i y_i}) \\ &= x_i \overline{y}_i + \ddot{u}_i(x_i + \overline{y}_i) \end{aligned}$$

- Addierer
- Laufzeit (mit t_l Laufzeit, n_k Stufenzahl, n_s Stufigkeit, t_s Gatterlaufzeit)

$$t_l = (n_k \cdot n_s \cdot t_s)$$

- Laufzeitverkürzung möglich durch
  - Zusammenfassen von Kettengliedern
  - Umformung von Kettenglidern in 2-stufige Form

6.2.3 Codeumsetzer

- Exzess-3 → Dezimal
- Dual-Code → Gray-Code

$$\begin{aligned} D_n &= G_n \\ D_i &= D_{i+1} \oplus G_i \\ G_n &= D_n \\ G_i &= D_i \oplus D_{i+1} \end{aligned}$$

6.2.4 Datenwähler

- **Demultiplexer (DEMUX)**: Schaltet ein Eingangssignal auf eine von 2^n Ausgansleitungen
- **Multiplexer (MUX)**: Schaltet eines von 2^n Eingangssignal auf einen Ausgang
- **Kreuzschienenverteiler (crossbar switch):** beliebige Verbindung von Eingags- mit Ausgangsleitungen
- **Permutationsnetzwerk:** Zusammenschaltung von Kreuzschienenvertilern

6.3 Speicherglieder

6.3.1 RS-Flipflop (Latch)

- Speichert Daten ohne Zeitbegrenzung
- R - Reset, S - Set
- Zwei NOR-Gatter, „über Kreuz“ rückgekoppelt
- Bistabile Kippschaltung aus rückgekoppelten Gattern können zwei stabile Zustände annehmen.
-

S	R	Q_{n+1}	
0	0	Q_n	Speichern
0	1	0	Rücksetzen
1	0	1	Setzen
1	1	-	unzulässig

6.3.2 RS-Flipflop mit Zustandssteuerung

- **asynchron:** aliegende Ansteuerung wird sofort wirksam
- **synchron:** Zeitpunkt, zu dem die Eingangsvariablen wirksam werden, wird durch Grundtakt festgelegt
- Zustandänderung nur bei $C = 1$ möglich
C	S	R	Q_{n+1}	
0	0	0	Q_n	Speichern
0	0	1	Q_n	Speichern
0	1	0	Q_n	Speichern
0	1	1	Q_n	Speichern
1	0	0	Q_n	Speichern
1	0	1	0	Rücksetzen
1	1	0	1	Setzen
1	1	1	-	unzulässig

6.3.3 D-Flipflop mit Zustandssteuerung

- Verhindert $R = S = 1$ indem beide Eingänge über einen Inverter verknüpft sind
C	D	Q_{n+1}	
0	0	Q_n	Speichern
0	1	Q_n	Speichern
1	0	0	Rücksetzen
1	1	1	Setzen

6.3.4 RS-Flipflop mit Zwei-Zustandsteuerung

- Master Flipflop übernimmt bei $C = 1$
- Slave Flipflop übernimmt bei $C = 0$
- $\Rightarrow$ rückflangengetriggert
- Verhindert in Registern und Zählern, dass Eingangssignale durch alle Glieder durchgeschaltet werden
- Analog bei D-Flipflops möglich

6.3.5 J-K-Flipflop

- J - Set
- K - Reset
- Verhindert $R = S = 1$ indem:
  - Set-Funktion wird nur dann ausgelöst wenn $\overline{Q} = 1$
  - Reset-Funktion wird nur dann ausgelöst wenn $Q = 1$
- $J = K = 1$ wechselt Zustand
- **J-K-Flipflop mit Rückflankensteuerung**
  - Verhindert oszillieren wenn $C = J = K = 1$
  - Q / $\overline{Q}$ Rückführung nur ans Master-Flipflop
  - Zustandstabelle:

J	K	Q_{n+1}	
0	0	Q_n	Speichern
0	1	0	Rücksetzen
1	0	1	Setzen
1	1	$\overline{Q}_n$	Zustand wechseln

- Synthesetabelle:

Q_n	$\rightarrow$	Q_{n+1}	J	K
0	$\rightarrow$	0	0	$*$
0	$\rightarrow$	1	1	$*$
1	$\rightarrow$	0	$*$	1
1	$\rightarrow$	1	$*$	0

6.3.6 Zähler

- **synchron:** Alle Flipflops schalten gleichzeitig
- **asynchron:** Flipflops schalten in einer zeitlichen Abfolge
- **Asynchronezähler**
  - $J = K = 1$
  - Eingang: C vom ersten Flipflop
  - $C_{n+1} = Q_n$
  - Kann beliebig durch Anfügen zusätzlicher Flipflops erweitert werden
- **Synchronzähler**
  - Explizite Ansteuerung der Flipflops über J und K Eingänge

6.3.7 Schieberegister

- FIFO
- Abschaltung des Taktes bewirkt Speicherung der Information
- Hintereinanderschaltung von D-Flipflops
- $D_{n+1} = Q_n$
- Gemeinsamer Takt C
- Erweiterungen
  - Parallele Ausgabe
  - Verriegelung der Parallelausgabe, da Eingabe während der Parallelen Ausgabe die Infomration verfälschen würde
  - Umschaltbare Schieberichtung

6.4 Programmierbare Logik

- **ROM:** Read Only Memory (Festwertspeicher)
  - **ROM:** Bei der Herstellung Programmiert
  - **PROM:** Programmable ROM: durch den Anwender programmierbar
  - **REPROM:** Reprogrammable ROM: noch Löschung erneut programmierbar
    - **EPROM:** Erasable PROM: mit UV-Licht löschbar
    - **EEPROM:** Electronically Erasable PROM
- **RAM:** Random Access Memory

6.4.1 PROM

- Beispiel: 16 festverdrahtete Konjunktionen (Minterme), 4 Programmierbare 16-stellige Disjunktionen

6.4.2 PAL-Bausteine (Programmable Array Logic)

- 16 programmierbare Konjunktionen, 4 festverdrahtete 4-stellige Disjunkionen

6.4.3 (Field) Programmable Logic Array (FPLA)

- UND-Matrix, sowie ODER-Matrix programmierbar
- Rückführung des Ausgangs möglich

6.4.4 Makrozellen (Programmierbare IC-Bausteine)

- Erweiterung der AND/OR-Arrays um Register und E/A-Blöcke

7 Automaten

7.1 Einführung

- **Automatentheorie:** mathematische Beschreibung von Systemen

7.2 Das Quintupel des Automaten

- **Quintupel**: Vollständige Beschreibung eines Automaten

$$A = (X, Y, Z, f, g)$$

- **Eingangsmenge X**
- **Ausgangsmenge Y**
- **Zustandsmenge Z**
- **Übergangsfunktion f**

$$f : X^{n+1} \times Z^n \to Z^{n+1}$$

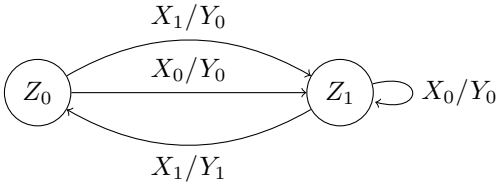
- **Ausgangsfunktion g**

$$g : X^{n+1} \times Z^n \to Y^{n+1}$$

7.3 Darstellungsweisen von Automaten

7.3.1 Automatengraphen

- Besteht aus Knoten und gerichteten Kanten
- Knoten: Innerer Zustand
- Kante: Zustandsübergang, Beschriftung: X^{n+1}/Y^{n+1}
- z.B.



7.3.2 Automatentabellen

- **Übergangstabelle:** Gibt f an, z.B.

n	$n+1$	
	X_0	X_1
Z_0	Z_1	Z_1
Z_1	Z_1	Z_0

- **Ausgangstabelle:** Gibt g an, z.B.

n	$n+1$	
	X_0	X_1
Z_0	Y_0	Y_1
Z_1	Y_0	Y_0

- **Automatentabelle:** Gibt f und g an, z.B.

n	$n+1$	
	X_0	X_1
Z_0	Z_1/Y_0	Z_1/Y_1
Z_1	Z_1/Y_0	Z_0/Y_0

7.3.3 Automatenband

7.4 Automatentypen

- **Mealy-Automat**
  - keine Einschränkungen; allgemeinster Automat
  - Übergangsautomat
- **Moore-Automat**
  - Zustandsautomat
  - $g : Z^n \to Y^n$
  - Ausgangsfunktion g hängt nur von dem Zustand Z und nicht von Eingabe X ab
  - Ausgabe kann im Graphen im Knoten angegeben werden
  - Automatentafel vereinfacht sich, z.B.

n	$n+1$		n
	X_0	X_1	
Z_{00}	Z_{10}	Z_{11}	Y_0
Z_{01}	Z_{10}	Z_{11}	Y_1
Z_{10}	Z_{10}	Z_{11}	Y_0
Z_{11}	Z_{00}	Z_{01}	Y_0

- **Zuordner**
  - kombinatorische Schaltung
  - $Z = \{Z_0\}$
  - $X^n \to Y^n$
- **Autonomer Automat**
  - $X = \{X_0\}$
  - Verhalten nicht von außen beeinflussbar
- **Halb-Automat**
  - $Y = \{Y_0\}$
  - keine praktische Bedeutung
- **Akzeptor**
  - Nicht als Einschränkung des Quintupels definierbar
  - Akzeptierte und Zurückgewiesene Eingabefolgen

7.5 Umwandlung zwischen Moore- und Mealy-Automat

- Jeder Moore-Automat kann in einen äquivalente Mealy-Automaten umgewandelt werden und umgekehrt
- Von außen ist nicht unterscheidbar welcher Typ ein Automat ist
- Moore → Mealy
  - Zustandmenge unverändert
  - Übergangsfunktion unverändert
  - Ausgabe Y^{n+1}_{Mealy} ist die Ausgabe des Folgezustands im Moore-Automaten
- Mealy → Moore
  - Jedem Kreuzprodukt aus neuem Eingangselement und altem Zustand des Mealy-Automaten entspricht einem Zustand des äquivalenten Moore-Automats

    $$X^{n+1} \times Z_{\text{Mealy}} = Z^{n+1}_{\text{Moore}}$$

7.6 Äquivalenz und Zustandsreduktion

7.6.1 Äquivalente Zustände

- Zwei Zustände Z_i und Z_j eines Automaten sind äquivalent, wenn der Automat auf eine beliebige Eingangsfolge stets mit derselben Ausgangsfolge reagiert, gleichgültig ob im Zustand Z_i oder Z_j begonnen wird.
- Zwei Zustände Z_i und Z_j eines Automaten sind k-äquivalent, wenn der Automat auf eine beliebige Eingangsfolge der Länge k stest mit derselben Ausgangsfolge reagiert, gleichgültig ob im Zustand Z_i oder Z_j begonnen wird.
- Ein Automat heißt zustandsminimal oder reduziert, wenn er keine äquivalenten Zuständer besitzt.
- Eine k-Äquivalenzklasse enthält alle k-äquivalenten Zustände eines Automaten.
- Eine k-Äquivalenzklasse enthält alle Zustände, die $(k-1)$-äquivalent sind und deren Folgezuständer für eine beliebige Eingabe $X_i \in X$ in denselben Äquivalenzklassen liegen.
- Moore-Automat fängt bei 0-äquivalent an, Mealy bei 1-äquivalent
- K^i_j, j-tes Element der i-Äquivalenzklasse

7.7 Technische Realisierung von Automaten

1. Aufstellen der Automatentabelle
2. Minmierung, falls redundante Zustände vorhanden
3. Auswahl der Bauelemente für die Schaltung
4. Festlegung der Codierungstabellen
5. Bestimmung der Schaltfunktion

7.7.1 Realisierung des Modellautomaten als Mealy-Automat

- PROM und D-Flipflops: Automatenrealisierung für Faule

8 Aufbau und Funktion einer Zentraleinheit

8.1 Der Von-Neumann Rechner

- Komponenten
  - Zentraleinheit (CPU: Central Processing Unit)
    - Steuerwerk (CU: Control Unit)
    - Rechenwerk (ALU: Arithmetic and Logic Unit)
    - Speicher
      - Porgramm-oder Befehlszähler (BZ, PC)
      - Befehlsregister
      - Speicheradressregister
      - Speicherdatenregister
      - Arbeitsregister
  - Hauptspeicher
  - Adressbus
  - Datenbus
  - Steuerleitungen
  - E/A-Geräte
- Instruction Fetch
  - Befehlszähler addressiert über SAR den Hauptspeicher
  - Lädt Befehl über SCR in Befehlsregister
- Steuerwerk
  - Ausführung von arithmetischen oder logischen Operationen durch die ALU
  - Beretistellen von Operanden für die ALU
  - Laden/Speichern von Arbeitsregistern
  - Erhöhen des BZ
  - laden von Sprungbefehlen
- Von-Neumann
  - Geteilter Daten- und Programmspeicher

8.2 Rechenwerk

- Komponenten
  - Wortaddierer
  - Schieberegister
  - parametrische Eingangsschaltung
    - liefert alle 4 einstelligen Booleschen Funktionen einer Eingangsgröße
- Bei logischen Operationen wird Übertrag unterbrochen
- Multiplikation durch fortgesetze Addition mithilfe Schieberegister
- Statusregister
  - C - Carry
  - N - Negative
  - Z - Zero
  - V - Overflow

8.3 Steuerwerk

- Aufgaben
  - Befehl holen
  - Vorbereitung der Adressierung des nächsten Befehls
  - Befehl interpretieren (Decodieren)
  - Operanden holen
  - Operation ausführen
  - Wegschaltung der Busse
- 1-Adress-Maschine: Akku + Operand → Akku
- 2-Adress-Maschine: Operand1 + Operand2 → Operand1
- 3-Adress-Maschine: Operand1 + Operand2 → Zielregister

8.4 Mikroprogrammierung

- Bei jedem Schritt wird ein Mikroprogrammowrt aus dem Mikroprogrammspeicher gelesen
- Mikrobefehle sind z.B.
  - Instruction fetch
  - Operand fetch
  - Execute

8.5 Kern der Zentraleinheit (CPU)

- ALU
- Steuerwerk
- Register
- Memory Mapped I/O

8.6 Abweichungen vom von Neumann-Konzept

8.6.1 Harvard-Architektur

- Getrennter Daten- und Programm-Speicher

8.6.2 Pipelining

- Zeitgewinn:

$$T = n \cdot k$$
$$T_p = n + k - 1$$
$$\frac{T_p}{T} \xrightarrow{n \to \infty} k$$

8.6.3 Sprungvorhersage (Branch Prediction)

- Branch Target Buffer
- Branch History Table
- Warscheinlicherer Code wird spekulativ bereits ausgeführt

8.6.4 Parallelrechner-Architekturen

- **SIMD:** Single Instruction, Multiple Data
  - Ein Steuerwerk
  - Mehrere ALUs
  - Eine Operation kann auf mehrere Operanden gleichzeitig ausgeführt werden
- **MIMD:** Multiple Instruction, Multiple Data
  - Mehrere ALUs
  - Ein Steuerwerk je ALU
  - Ermöglicht, dass jede ALU andere Befehle ausführt

8.6.5 Rechenwerk mit Vektoreinheit

8.7 Gleitkomma-Koprozessoren

- Komponenten
  - Gleitkomma-Arithmetikeinheit
  - Gleitkommaregister
  - Steuer- und Statusregister
- Bei modernen Prozessoren im Prozessorkern integriert

8.8 Superskalarität

- Gleichzeitige Ausführung zweier oder mehrerer Instruktionen durch zwei oder mehrere Ausführungseinheiten
- Instruction Reorder nötig
- bzw. Out of Order Execution

8.9 Register Renaming

- Löst Problem bei Out of Order Execution, dass mehrere Instruktion auf ein Regster schreiben wollen
- Register Alias Table

8.10 CISC-versus RISC-Maschinen

- **CISC:** Complex Instruction Set Computer
- **RISC:** Reduced Instruction Set Computer
- Kriterien (5 von 8 für RISC)
  - weniger als 50 Maschenenbefehle
  - weniger als 4 Adressierungsarten
  - weniger als 4 Befehlsformate
  - Speicherzugriff nut über LOAD/STORE Befehl
  - mehr als 32 Prozessorregister
  - fest verdrahtete Maschinenbefehle (keine Mikroprogrammierung)
  - Befehlsausführung in (meist) einem Taktzyklus
  - Verfügbarkeit optimierender Compiler für höhere Progrommiersprachen

8.11 VLIW-Prozessoren

- **VLIW:** Very Long Instruction Word
- $\approx$ 1024-bit Wortlänge
- Mehrere Operanden
- Festlegung der Instuktionsworte ist Aufgabe des Assemblers
- Beispiel: TI C6xx, Intel Itanium

8.12 Multithreading-Architekturen

- **Thread:** Unabhängiger Kontrollfluss innerhalb eines Programms
- Threads arbeiten im gegensatz zu Prozessen in einem gemeinsamen Adressraum
- Unterschied zur Superskalarität: Befehle stammen aus unterschiedlichen Threads, weniger Datenabhängigkeiten
- Befehlszähler und Register müssen mehrfach vorhanden sein

9 Maschinensprache und Assembler

- **Maschinensprache:** Befehlswörter als Nullen und Einsen
- **Assemblersprache:** Symbolische Maschinensprache, erlabut Symbole für Speicheradressen und Daten

9.1 Zugrunde gelegte Hardware

- Atmel AVRmega8
  - RISC
  - 8-bit
  - Harvard-Architektur
  - 2-Adress ALU

9.2 Register

9.2.1 Der Statusregister

- C - Carry
- Z - Zero
- N - Negative
- V - Twos Complement Flag
- S - Sign-Bit
- H - Half Carry
- T - Bit Copy Storage
- I - Global Interupt Enable
- Definition

$$
\begin{aligned}
C &= Rd7 \cdot Rr7 + Rd7 \cdot \overline{R7} + Rr7 \cdot \overline{R7} \\
Z &= \overline{R7} \cdot \overline{R6} \cdot \overline{R5} \cdot \overline{R4} \cdot \overline{R3} \cdot \overline{R2} \cdot \overline{R1} \cdot \overline{R0} \\
N &= R7 \\
V &= Rd7 \cdot Rr7 \cdot \overline{R7} + \overline{Rd7} \cdot \overline{Rr7} \cdot R7 \\
S &= N \oplus V
\end{aligned}
$$

9.2.2 General Purpose Working Register

- R0 bis R31
- 16-bit Register
  - R26 - X-Register (Low)
  - R27 - X-Register (High)
  - R28 - Y-Register (Low)
  - R29 - Y-Register (High)
  - R30 - Z-Register (Low)
  - R31 - Z-Register (High)

9.2.3 Stack-Pointer

- LIFO
- POP - Lesen
- PUSH - Schreiben
- Initialisierung

```
.include "m8def.inc"

LDI r16, LOW(RAMEND)
OUT SPL, r16
LDI r16, HIGH(RAMEND)
OUT SPH, r16
```

9.3 Assembler

- Symbolische Maschinensprache
- Maschinenspezifisch
- Maschinennah
- Menschenlesbar
- keine Hochsprache

9.3.1 Assemblerbefehle

- Kein Befehl für Programmende

9.3.2 Ablauf eines Assemblerprogramms

9.4 Untergliederung der Assemblerbefehle

- Datentransferbefehle
- Arithmetische Operationen
- Logische Operationen
- Vergleichsbefehle
- Bit-Befehle
- Status-Befehle
- Verzweigungsbefehle
- Befehle zur Programmablaufsteurung
- Befehle zur Unterbrechungsverarbeitung

9.4.1 Datentransferbefehle

- `MOV Rd, Rr`
- `LDI Rd, K`
- `LDS Rd, k`
- `LD Rd, Z`
- `LDD Rd, Z+q`
- `STS k, Rr`
- `ST Z, Rr`
- `STD Z+q, Rr`
- `IN Rd, P`
- `OUT P, Rr`
- `PUSH Rr`
- `POP Rd`

9.4.2 Arithmetische Operationen

- `ADD Rd, Rr`
- `SUB Rd, Rr`
- `MUL Rd, Rr`
- `LSL Rd`
- `LSR Rd`
- `INC Rd`
- `Dec Rd`

9.4.3 Logische Operationen

- `AND Rd, Rr`
- `OR Rd, Rr`
- `EOR Rd, Rr`
- `COM, Rd`

9.4.4 Vergleichsbefehle

- `CP Rd, Rr`
- `CPI Rd, K`

9.4.5 Bit-Befehle

- `SBR Rd, K` (eigentlich `ORI`)
- `CBR Rd, K` (eigentlich `ANDI`)
- `BSET s`
- `BCLR s`

9.4.6 Verzweigungsbefehle

- `RCALL k`
- `RET`
- `RJMP k`
- `BREQ k`
- `BRLO k`
- `IJMP`
- `ICALL`

9.4.7 Programmierung von Schleifen

9.4.8 Rekursion

- `RCALL`
- Rekursionsanfang
- Rekursionsschritt

10 Organisation der Ein-/Ausgabe

10.1 Ein-/Ausgabe-Hardware

10.1.1 Grafikkarten

- Video-RAM
- Frame Buffer Controller (FBC)
- RAMDAC
  - Color Lookup Table (CLUT)
  - DAC
- Beschleunigung der Grafikausgabe
  - 2D/GUI Beschleuniger
  - 3D Beschleuniger
    - Z-Buffer
- Digitale Ansteuerung von Bildschirmen
  - TMDS-Protokoll (Transition-Minimized Differential Signaling) wird von allen Standards verwendet

10.1.2 Drucker

- **Nadeldrucker**
  - Ermöglicht Durchschläge
  - Niedrige Druckkosten
- **Tintenstrahldrucker**
  - Tinte wird durch Düsen auf Papier geschleudert
  - Überdruck kann mitt Piezo-Verfahren oder Bubble-Jet (Heizelementen) generiert werden
- **Laserdrucker**
  - Bildtrommel: ein mit fotoelektrischem Material beschichteter Metallzylinder
  - Trommel wird elektrostatisch aufgeladen

  - Laser schreibt negatives Bild auf die geladene Trommel
  - An den verbleibenden Stellen beibt Toner haften
  - Postscript sprache von Adobe rasterisiert Grafik im Drucker
  - Hohe Wartungskosten
- **Farbdrucker**
  - Thermo-Sublimationstechnik
  - Geschmolzene Wachsfarbe
  - Hoche Qualität, teuer

10.1.3 Maus und Trackball

- Opto-Mechanisch
  - Kugel
  - Zwei Achsen, je eine Rolle, eine Lochscheibe, zwei Lichtschranken
  - Aus Phasenlage der zwei Lichtschranke wird Bewegungsrichtung erkannt
- Optisch
  - CMOS-Sensor
  - DSP mit Motion Estimation

10.2 Busse und Schnittstellen

10.2.1 Einführung

- Schnittstelle (Interface): Baugruppe zum physikalischen Zugriff auf ein Bussystem
- Verbindungsnetz
  - Physikalische Möglichkeit zur Übertragung der Daten
  - Protokoll
- logischer Bus
  - Gemeinsames Verbindungsnetz für alle Komponenten
  - eindeutige Adressierung der Kommunikationpartner
  - Verfahre zur Vermeidung von Konflikten

10.2.2 Grundlagen der Busarten

- **Überblick der Busarten**
  - Parallel
    - Getrennte Leitungen für Adressbus, Datenbus, Steuerbus
  - Seriell
    - synchron oder asynchron
  - Busarten
    - Register-Bus
    - System-Bus (CPU-Bus)
    - Speicherbus: Schneller Parallelbus zum Hauptspeicher
    - Perpheriebus
    - Ein-/Ausgabebus
  - Bustopologien
    - Stern: am häufigsten, zentraler Switch (Vermittlungsknoten)
    - Ring: unidirektional von jedem Teilnehmer an den Nachbarn
    - Strang: von Sternpunkten

        * Baum: verbunden über Hubs

- Koppeleinheiten
  - Repeater: elektrische Verstärkung, Synchronisation, Daten werden nicht gepuffert
  - Hub: Siehe Baumtopologie
  - Bridge: komplex, verbindet gleiche oder verschiedene Busse. Puffert daten, wandelt in andere Protokolle um
  - Switch: kann mehrere Verbindungen zwischen Komponenten gleichzeitig verwalten
- Adressierung der Busteilnehmer
  - Bei seriellen Bussen enthalten die über den Bus verschickten Datenpakete auch die Empfängeradresse
  - Bei Baum-Topologien kann entweder nur mit dem Wurzelknoten kommuniziert werden (z.B. USB) oder zwischen allen Teilnehmern (z.B. Firewire)
- Busarbitrierung
  - Auflösung von Konflikten bei gleichzeitigem Zugriff mehrere Busmaster auf einen Bussystem
  - Bus Request (BRQ) an Arbiter, Bus Grant (BGR) kommt zurück, Komponente muss auf BUSY Singal warten
    * unabhängige Anforderung: Zentraler Arbiter, jede Komponente besetzt eine BRQ und eine BGR Leitung
    * dezentraler Arbiter: Jede Komponente besitzt einen Arbiter, untereinander mit BRQ Leitungen verbunden. Alle Komponenten die nicht die höchste Priorität besitzen, ziehen ihre Anforderungen zurück
    * Daisy-Chain: Jede Komponente requestet auf einem BRQ, BGR wird von einem zum nächsten gereicht. Priorität entspricht enfernung zum Bus
  - Bei Baum-Topologien: Polling-Verfahren

10.2.3 Beispiele für Peripherie-Busse

- PC-AT-Bus (ISA, Industrie-Standard-Architektur)
  - 16 Datenleitungen
  - 24 Adressleitungen
  - 2 Takte pro Wort
  - 8 MHz
- PCI (Peripheral Component Interconnect)
  - Wichtigste Koppeleinheit: Host-Brücke, zentraler Bus-Arbiter
  - Adressierung der Busteilnehmer: Positiver Decodiermodus
  - Busarbitrierung: „faire“ Zuteilung
  - PCI-PCI Transfer möglich
  - 32 oder 64 Leitungen, 32 oder 64 breite Adress-und Datenwörter im Multiplexverfahren
  - AGP
- PCI-Express
  - Seriell über Lanes im gegensatz zu parallel
  - Zwei Unidirektionale Leitungen (voll duplex) pro Lane
  - Bis zu 2 Gbit/s pro Lane
  - Bis zu 16 Lanes parallel
  - Für Betriebssystem identisch zu PCI
- PCMCIA
  - Personal Computer Memory Card International Association
  - PC-Steckkarte

10.2.4 Beispiele für E/A-Busse

- SCSI-Bus (Small Computer System Interface)
  - Synchron sowie asynchron
  - bis zu 16 Teilnehmer
  - Kommandobasiert
  - Topologie: Strang, Ankopplung an Peripheriebus über eine Brücke
  - Adressierung: Adressierungsphase
  - Arbitrierung: nach Bus ID 7, . . . , 0, 15,. . . , 8
- USB-Bus (Universal Serial Bus)
  - Bis zu 127 Geräte
  - Hotplugging
  - Stromversorgung
  - Topologie: Baum
    - Host - Wurzel
    - Hub - inneren Knoten
    - Function - Blätter
  - Adressierung: Durch Enumeration Process vom Host erteilt
  - Busarbitrierung: Polling-Verfahren
  - Pakettypen
    - Start-of-Frame-Paket
    - Token-Paket
    - Datenpaket
    - Handshake-Paket
  - Übertragungsarten
    - Isochrone Übertragung: Garantiert konstante Datenrate, keine Fehlererkennung, 8,2 Mbit/s
    - Bulk-Übertragung: Große Datenmenge, Fehelrerkennung, 9,7 Mbit/s
    - Interrupt: Maus, Keyboard
    - Steuerdaten
    - Low Speed
- FireWire (IEEE-1394-Bus)
  - Weitgehend ähnlich zu USB
  - Höhere Datenrate
  - Hot-Plugging
  - Topologie: Baum, kein Rechner notwendig, alle Geräte mit FireWire Anschluss als Knoten erlaubt
  - maximal 63 Knoten pro Bus, bis zu 1023 Busse über Brücken gekoppelt
  - Maximal 16 Kabelabschnitte zwischen 2 Kommunizierenden Knoten
  - Pakete Entsprechen denen des USB
  - Adressierung: 10 Bits zur Adressierung des Teilbusses, 6 zur Selektion der Komponente, 48 zur Adressierung von 256 TByte pro Knoten
  - Busarbitrierung: durch den Root-Knoten
  - Kommunikation: Entwerder asynchron (mit Fehlererkernnung) oder isochron (ohne Fehlererkennung)

10.2.5 Schnittstellen für Punkt-zu-Punkt Verbindungen

- **Schnittstelle:** Übergangsstelel zwischen den Geräten eines Systems, an der Daten und Signale ausgetauscht werden
- Seriell oder parallel
- **V.25 (RS232) Schnittstelle**
  - Asynchron
  - Seriell
  - Ein Startbit mit L-Pegel
  - Ein oder Zwei Stopbits mit H-Pegel
  - Optional ein Paritätsbit
  - Es reichen 3 Pins: RD (Data transmit), RD (Data receive), GND (ground)
  - Baudrate wählbar
- **Centronics-Schnittstelle**
  - Drucker
  - 8 parallele Datenbits
  - Unidirektional

10.2.6 Drahtlose Kommunikation

- **IrDa**
  - Infrared Data Association
- **Bluetooth**
  - Piconet (Ein Master, bis zu 7 Slaves)
  - Scatternet: Gekoppelte Piconets
  - 2,4 GHz
  - entweder 1 mW, 2,5 mW, oder 100 mW

10.3 Ein-/Ausgabetechnik

10.3.1 Ein-/Ausgabe über den CPU-Bus

- **Direkte Ein/Ausgabe**
  - Programmgesteuert oder
  - Unterbrechungsgesteuert (Interrupts)
- **Gepufferte Ein/Ausgabe**
  - Höhere Übertragungsgeschwindigkeiten
  - Wird in getrenntem Microcontroller gepuffert
  - Eingaben bleiben solange bereitgehalten, bis es von der Eingaberoutine abgeholt wird
  - Ausgaben werden im Puffer geschrieben, und (bei Freigabe des Übertragungskanals) übernommen und übertragen

10.3.2 Ein-/Ausgabe über Direct Memory Access (DMA)

- DMA-Controller erhält Startadresse und Länge eines zu übertragenden Blocks entwerder program- oder interruptgesteuert durch die CPU
- Üblicherweise hat DMA Priorität und PC kann nur dann schrieben, wenn Bus freigegeben ist

10.4 Ein-/Ausgabe von Analogdaten

10.4.1 Analogeingabe

- Sägezahspannung
  - Generator erzeugt sägezahnartigen Spannungsanstieg
  - Ziet bis zum Erreichen des Pegels der Messspannung ist proportional zur Messspannung
- Stufenumsetzung/Wägeverfahren
  - Eine Art Binäre Suche der Messspannung

10.4.2 Analogausgabe

- Stromquelle mit binär abgestuften Widerständen

11 Speichertechnik

11.1 Speichermerkmale

- **Zugriffszeit:** Dauer zum Lesen bzw. Schreiben einer Speicherzelle
- **Zugriffsart:** Methode, mit der auf dem Speicher zugegriffen wird
  - RAM
  - ROM
  - Speicher mit sequentiellem Zugriff
  - Speicher mit zyklischem oder halbdirektem Zugriff
- **Zykluszeit:** Zeitdauer zwischen den Beginn eines Speichervorgangs bis zum Beginn des nächsten Speichervorgangs
- **Speicherkapazität:** Anzahl der verfügbaren Speicherzellen
- **Statische Speicher:** Behält Inhalt, solange die Versorgungsspannung anliegt.
- **Dynamische Speicher:** müssen regelmäßig mit ihrem Inhalt nachgeladen werden.
- **Speicherprinzip:** Funktionsweise und Aufbau des Speichers

11.2 Halbleiterspeicher

11.2.1 Halbleiterbauelemente

- Dioden
- Transistoren
  - Bipolar
  - Feldeffekt: Metall, Oxid, und Halbleiter
  - Zwei n-dotierte Inseln (Source, Drain) im p-dotierten Kristall
  - Gate ist nichtleitend
  - Gate-Source Spannung erzeugt leitenden Tunnel zwischen Source und Drain

11.2.2 Integrierte Schaltungen

- Schaltungen aus Halbleiterbauelementen auf ein einzelnes Siliziumplättchen
- verschiedene Schaltkreisfamilien
- Bipolare Familien
  - RTL - Resistor-Transistor-Logic
  - DTL - Diode-Transistor-Logic
  - TTL - Transistor-Transistor-Logic
  - ECL - Emitter-Coupled-Logic
  - I^2L - Integrated-Injection Logic
- MOS-Familien
  - PMOS - P-Channel-MOS
  - NMOS - N-Channel-MOS
  - CMOS - Complementary-MOS

11.2.3 Schreib-Lese-Speicher (RAM)

- Matrixförmig angeordnete Speicherzellen
- Schreib-Lese-Umschaltung
- Chip-Select-Anschluss
- **Statische RAMs:** Flipflops
- **Dynamische RAMs:** Kondensatorbasierend, müssen refreshed werden

11.2.4 Assoziativspeicher

- Adressierung nicht per Zelle, sondern gleichzeitig auf alle Speicherzellen
- Es wird nach allen Speicherzellen die bestimmte Vorgaben erfüllen gesucht
- Aufwändige Logik

11.2.5 Festwertspeicher (ROM)

- Nicht von Programm beschreibbar
- Erhalten Inhalt nach abschalten der Betriebsspannung
- Arten
  - **Maskenprogrammierbare ROMs (MROMs)**
    - Siliziumchip mit photoempfindlicher Schicht wird belichtet
    - Nur bei hohen Stückzahlen
  - **Programmable ROM (PROM)**
    - Einmal vom Anwender beschreibbar
    - Als programmiertes Bauelement wird eine zerstörbare Verbindungsleitung verwendet
  - **UV-löschbarer Festwertspeicher (EPROM)**
  - **Elektronisch löschbarer Festwertspeicher (EEPROM)**

11.3 Magnetische Massenspeicher

- Data Catridge: Magnetbandkasette in einem Kunstoff-Metallgehäuse
- 8mm Data Tape
- 4mm Data Tape (DAT)
- VHS-Videobänder
- Disketten
  - 300 U/min
  - Kunstofffolie mit Eisenoxyd beschichtet
- Festplatten

11.4 Optische Massenspeicher

11.5 Magneto-optische Massenspeicher

11.6 Speicherorganisation

11.6.1 Cache-Speicher

11.6.2 Virtueller Speicher

12 Speicheraufbau am konkreten Beispiel

12.1 Pentium-Familie

12.1.1 Übersicht

12.1.2 Beispiel: Pentium 4

12.2 PowerPC-Familie

12.2.1 Übersicht

12.2.2 Beispiel: PowerPC 7400

12.3 Leistungsbewertung von Rechnersystemen

12.3.1 Benchmarkverfahren

12.3.2 Die SPEC CPU-Suite

12.4 Entwicklungsperspektiven bei Speicherkapazität und Rechengeschwindigkeit