Einführung in die Informatik 3

für den Studiengang Ingenieur-Informatik
im Fachbereich Feinwerktechnik
der Fachhochschule Frankfurt am Main

Prof. Dr. Erik Jacobson
Fachbereich MND

Sommersemester 1999

Übersicht

Belegnummer: 06 75 31

Inhalte

1. Hardware (wie ein Computer funktioniert)

- Schaltungstechnik: Schaltungslogik, Schaltelemente
- Beschreibungsmittel (Graphen): Petri-Netze, Zustandsdiagramme
- Datendarstellungen
- Rechnerarchitekturen:
  - Die Zentraleinheit
  - Die CPU, Maschinenbefehle und Mikroprogrammierung
  - Der Bus
  - Der Arbeitsspeicher
  - Ein- und Ausgabe, Schnittstellen
  - Periphere Geräte: Terminal, Drucker, Plotter, Massenspeicher

2. Softwareware (Die Basismaschine)

- Betriebsarten, Betriebssystemfunktionen
- Prozesse, Prozeßverwaltung
- Speicherverwaltung
- Dateiverwaltung
- Interprozeßkommunikation
- Benutzeroberflächen
- Schichtenmodell
- Anwendungsbeispiel: UNIX, MVS, VMS, BS2000, RSX, RT-11, o.ä.
- Spezialthemen, z.B.: TP-Systeme, TP-Monitore

3. Prozeßdatenverarbeitung, PDV (Anwendungsgebiet)

- Prozeßelemente, Prozeßglieder (Sensoren, Motoren, ADCs, DACs)
- Prozeßbetrieb, Realzeitbetrieb
- Prozeßsoftware

Literatur **(eine Auswahl)**

- Coy, W.: Aufbau und Arbeitsweise von Rechenanlagen. Vieweg 1988
- Jacobson, E.: Einführung in die Prozeßdatenverarbeitung. Hanser 1996
- Sokolowski/ Huge: Digitale Schaltungs- und Rechnertechnik.

Labor **(nach Vereinbarung)**

- Bussignale und -protokolle
- COM-Schnittstelle (RS 232C, V.24)
- Interrupt Handler
- Laufzeitmessungen

Leistungsnachweise

- Klausur am Semesterende (s. WWW) (Minimalanforderung = 50 %)
- Laborberichte (Anrechnung bis 20 %)
- Nachklausur

Kapitel 0

Einleitung

0.1 Historische Grundlagen

0.1.1 Hilfsmittel zur Datenverarbeitung

0.1.1.1 Primitive Hilfsmittel

- 6000 v Chr.	2 Hände x 5 Finger = biquinär = 10 Finger = dezimal Finger = lat.: digitus → digital Zählen = lat.: computare → computer Zählstäbchen = lat.: calculi → Kalkül
- 1000 v Chr.	Abakus (Griechenland, China, Rußland, Rom)

0.1.1.2 Mechanische Hilfsmittel

1623	Schickard, W. Tübingen: 6-stellige Rechenuhr (Add, Sub)
1641	Pascal, Blaise Frankreich: 6-stellige Addiermaschine
1650	Partridge, England: Rechenschieber
1671	Leibniz, G.W. Hannover: 4-Spezies-Rechner mit Staffelwalzen (Kurbelmaschine)
1900	elektrischer Antrieb für Kurbelrechenmaschinen

0.1.1.3 Logische Hilfsmittel

- 1000	Zahldarstellung mittels Buchstaben (Alphabet = α, β)
- 0	Mathematische Grundlagen: Pythagoras, Aristoteles, ..
$\pm$ 0	Römisches Zahlensystem (I, V, X, L, C, D, M)
500	Hindu-Arabisches Zahlensystem: (dezimale) Stellenschreibweise
1574	Adam Riese: Popularisierung der Rechenkünste
1614	Napier: Logarithmentafeln
1703	Leibniz: Dualsystem
1847	Boole: Schaltalgebra
1833	Babbage & Ada (Augusta Comtess of Lovelace geb. Byron): Difference Engine, Analytic Engine (Mill, Store, Control)
1940	Programmiersprachen

0.1.2 DV-Systeme

1890	Hollerith:	Lochkartenmaschine zur Volkszählung in Chikago
1934	Konrad Zuse, Berlin:	Z1: erster programmgesteuerter Rechner Z3: CPU mit 600 Relais, 22 Dualstellen, Speicher mit 64 Worten (2000 Relais), Steuerung mit Programm auf Lochstreifen
1944	Aiken @ Harvard, USA	Relais-Rechner MARK 1
1946	Ekert & Mauchly @ MIT, USA	Röhren-Rechner ENIAC (18 000 Röhren)
1946	John v. Neumann @ MIT	Rechner-Architektur: Steuerwerk + Rechenwerk ein Speicher für Daten und Programme, Ein- und Ausgabeeinheiten (-werke)
1950	Universalrechner:	Hardware und Betriebssysteme (in ROM)
1959	Siemens 2002	2000 Worte à 14 BCD-Stellen (56 Bit)

0.1.3 Rechner-Generationen

	Zeit	Schaltelement	-zeit	Operationszeit	Add/Sek
0	1935 - 1945	Relais	1 ms	100 ms	30
1	1945 - 1955	Röhren	10 μs	1 ms	1000
	1951	Kernspeicher	1 μs		
2	1955 - 1965	Transistoren	1 μs	100 μs	10^4
3	1965 - 1971	Integrierte Schaltung (SSI)	10 ns	1 μs	10^4
4	1971 - 1985	LSI	1 ns	100 ns	10^6
5	1985 - 1995	VLSI, XLSI	1 ns	10 ns	10^7

0.1.4 Physikalische Grenzen

Größe	aktuell	Grenze
Breite einer Leiterbahn	$0.4\mu m$ (400 nm)	100 Atome $= 100 \cdot 10^{-10} m = 10nm$
Taktfrequenz	200 – 400 MHz	20 GHz ($\lambda = c^*/f = 1cm$) ($c^* = c/\varepsilon \approx 200\,000$ km/s)

0.2 Begriffe

DIN 44 300, 66 xxx, ISO 2382 Teile 1 bis 36

Informatik (engl.: Computer Science, CS): Wissenschaft von der Informationsverarbeitung

Anwendung: Informationstechnik (Information Technology, IT)

0.2.1 Grundbegriffe

Information (objektartig): Dargestelltes Wissen über beliebige Objekte: Fakten, Ereignisse, Dinge, Vorgänge, Ideen, Konzepte, ...

Daten: Formalisierte Darstellung von Information, die geeignet ist zur Weitergabe, Interpretation oder Verarbeitung.

Signale: Veränderung einer physikalischen Größe zur Darstellung von Daten. Zum Beispiel Strom, Spannung, Helligkeit, Farbe.

0.2.2 Basisreferenzmodell

Schichten-, Schalen-, Strukturmodell

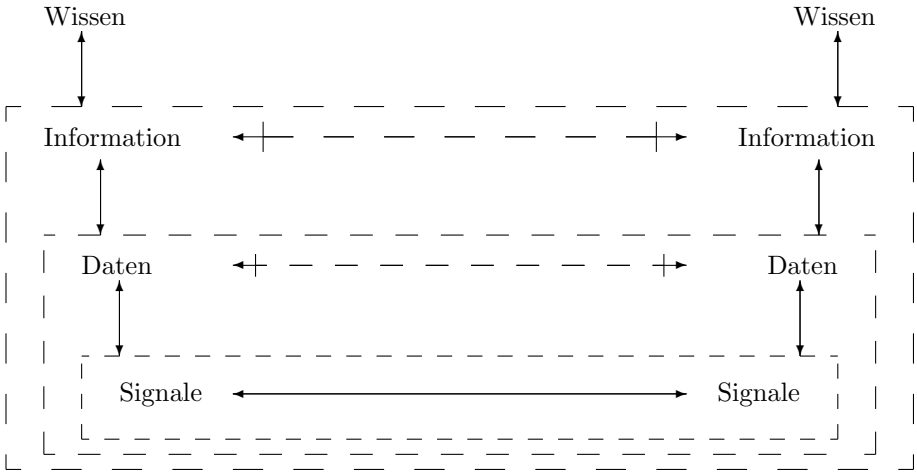
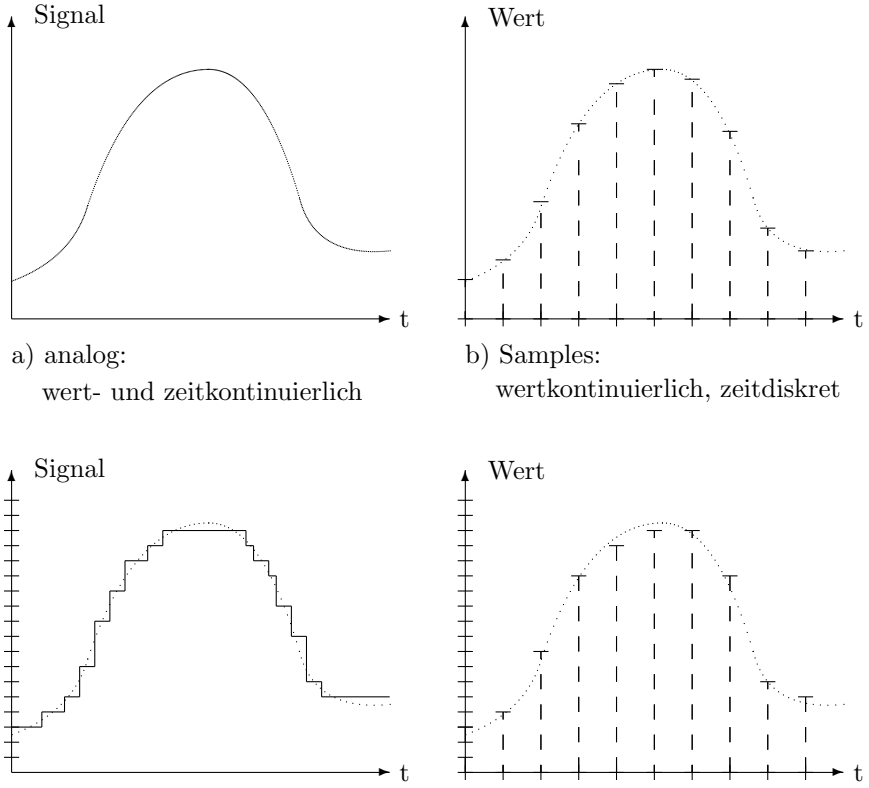


Bild b1p01

↓ : Darstellung, Repräsentation, Codierung
↑ : Erwerb, Interpretation, Decodierung
⟷ : Übertragung, Verarbeitung, Speicherung

0.2.3 Signale

Signalwert (Signalparameter): veränderlicher Wert, abhängig von Ort oder Zeit (y-Koordinate). z.B. el. Spannung in Volt
Signalverlauf in der Zeit (t) oder als Funktion des Ortes (x-Koordinate)
Signalqualität:
- kontinuierlich (auf reele Zahlen abbildbar)
- diskret (auf natürliche (ganze) Zahle abbildbar)



a) analog:
wert- und zeitkontinuierlich

b) Samples:
wertkontinuierlich, zeitdiskret

c) quantisiert:
wertdiskret, zeitkontinuierlich

d) digital:
wert- und zeitdiskret

Bild b1p02

0.2.4 Daten

analog: kontinuierlich.

diskret:diskontinuierlich, unterscheidbare Werte.

digital: durch Daten aus Zeichen dargestellt.

numerisch: Daten aus Numeralen (Zeichenfolgen für Zahlen).
Beispiel: 11, elf, XI, B_{16}

alphanumerisch: Daten aus Buchstaben und Ziffern.

Zeichen: Element einer Menge von Symbolen.

Ziffer: Zeichen zur Darstellung einer (nichtnegativen) Zahl.

Symbol: Graphische Darstellung mit vorgegebener Bedeutung (kontextabhängig).

Bit: Zeichen zur Darstellung von Ziffern im Dualsystem.

Wort: Syntaktische Einheit von Zeichen.

Byte: Teilwort bestimmter Länge (meist 8 Bit).

Oktett: Folge von 8 Zeichen (Bits).

String: Folge von Zeichen unbestimmter Länge.

Satz: Folge von Worten (mit Struktur).

Datei: Folge von Sätzen (gleicher Struktur).

Text: Darstellung von Konstrukten einer Sprache durch Daten. Gewöhnlich in Form von Zeichen, Symbolen, Worten, Phrasen, Absätzen, Sätzen, Tabellen oder anderen Zusammenstellungen von Zeichen.

Datenobjekt: Daten mit vorgegebenen Datentyp.

Datentyp: Aufbau und Anordnung von Daten und die auf ihnen erlaubten Operationen.

Adresse: Wort zur Kennzeichnung (Lokalisierung) eines Datenobjekts.

Adreßraum: Menge von (zugelassenen) Adressen

Programm: Folge von Anweisungen und Vereinbarungen zur Lösung einer Aufgabe (eines Problems).

Anweisung: Arbeitsvorschrift, abgefaßt in einer (Programmier-)Sprache

Vereinbarung: Festlegung von Sprachelementen für Anweisungen.

Operation: Ein Methode, um nach vorgegebenen Regeln aus gegebenen Objekten (Operanden) ein neues Objekt (Resultat) zu erzeugen.

Instruktion: elementare Anweisung.

System: Eine Menge von Objekten und ihren Beziehungen, die von ihrer Umgebung abgegrenzt erscheint.

Struktur: Die Beziehungen zwischen den Elementen eines Systems.

Kapitel 1

Grundlagen der Digitalelektronik

1.1 Boole'sche Algebra

1.1.1 Definition

Einfachste Algebra mit
- 2 Elementen: {0, 1} {L, H} {O, L}
- 3 Grund-Operationen:
  - unäre Operation:
    Komplement, Negation, NICHT, NOT,

$x \rightarrow \bar{x}$	
0	1
1	0

  - binäre Operationen:
    Disjunktion, ODER, OR,
    f(x,y) = Max(x,y) = x + y

$x \vee y \rightarrow z$		
0	0	0
0	1	1
1	0	1
1	1	1

    .Konjunktion, UND, AND,
    f(x,y) = Min(x, y) = $x \cdot y$

$x \wedge y \rightarrow z$		
0	0	0
0	1	0
1	0	0
1	1	1

- Zusammengesetzte binäre Operationen (2 Beispiele):

Äquivalenz,
$f(x,y) = x \cdot y + \overline{x} \cdot \overline{y}$

$x \equiv y \rightarrow z$		
0	0	1
0	1	0
1	0	0
1	1	1

.Antivalenz, Exklusives Oder, XOR,
$f(x,y) = x \cdot \overline{y} + \overline{x} \cdot y$

$x \oplus y \rightarrow z$		
0	0	0
0	1	1
1	0	1
1	1	0

Gesetze der Boole'schen Algebra:

Assoziativgesetz

$$(x + y) + z = x + (y + z) = x + y + z$$
$$(x \cdot y) \cdot z = x \cdot (y \cdot z) = x \cdot y \cdot z$$
$$(x \oplus y) \oplus z = x \oplus (y \oplus z) = x \oplus y \oplus z$$

Kommutativgesetz

$$x + y = y + x$$
$$x \cdot y = y \cdot x$$
$$x \oplus y = y \oplus x$$

Distributivgesetz

$$x \cdot (y + z) = (x \cdot y) + (x \cdot z)$$
$$x + (y \cdot z) = (x + y) \cdot (x + z)$$

DeMorgansche Regeln

$$\overline{(x + y)} = \overline{x} \cdot \overline{y}$$
$$\overline{(x \cdot y)} = \overline{x} + \overline{y}$$

Absorptionsregeln

$$x + \overline{x} = 1 \qquad x \cdot \overline{x} = 0 \qquad x \oplus \overline{x} = 1$$
$$x + x = x \qquad x \cdot x = x \qquad x \oplus x = 0$$
$$x + x \cdot y = x \qquad x \cdot (x + y) = x$$

Neutrale Elemente

0 bezüglich OR (+): $x + 0 = x$ $\qquad x \cdot 0 = 0$ $\qquad x \oplus 0 = x$

1 bezüglich AND ($\cdot$): $x \cdot 1 = x$ $\qquad x + 1 = 1$ $\qquad x \oplus 1 = \overline{x}$

1.1.2 Boole'sche Funktionen

Abbildung von m logischen Werten auf n logische Werte

$B^m \rightarrow B^n$ mit $B = \{0, 1\}$

dargestellt durch Funktionen: $\left\{ \begin{array}{c} x \\ y \end{array} \right\} = \left\{ \begin{array}{c} f_1(a, b, c) \\ f_2(a, b, c) \end{array} \right\}$

oder durch Wertetabellen,

zum Beispiel:

a	b	c	x	y
0	0	0	0	1
0	0	1	1	1
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	1	1
1	1	0	0	1
1	1	1	0	0

1.1.3 Normalformen

Erstellung von Boole'schen Funktionen aus Wertetafeln oder KV-Diagrammen.
Vollständige Normalformen enthalten in allen Funktionstermen alle Variablen (oder deren Inverse). Sie können meist unter Anwendung der Gesetze der Boole'schen Algebra oder mit Hilfe von KV-Diagrammen vereinfacht werden.

Beispiel:

a	b	c	x
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

1.1.3.1 Disjunktive Normalform

DNF (Min-Terme)
- Tabellenzeilen mit Funktionswert '1' liefern einen Funktionsterm (Min-Term)
- Funktionsterme werden aus den Eingangsvariablen mit UND verknüpft
- alle Funktionsterme werden mit ODER verknüpft
- DNF = $\Sigma\Pi x_i$ (Summe von Produkten)

Beispiel:
$x = \bar{a} \cdot \bar{b} \cdot c + \bar{a} \cdot b \cdot \bar{c} + a \cdot \bar{b} \cdot \bar{c} + a \cdot \bar{b} \cdot c$ (vollständige DNF)
$x = \bar{a} \cdot \bar{b} \cdot c + \bar{a} \cdot b \cdot \bar{c} + a \cdot \bar{b}$ (reduzierte DNF) (Absorption von $(c + \bar{c}) = 1$)

1.1.3.2 Konjunktive Normalform

KNF (Max-Terme)
- Tabellenzeilen mit Funktionswert '0' liefern einen Funktionsterm (Max-Term)
- Funktionsterme werden aus den invertierten Eingangsvariablen mit ODER verknüpft
- alle Funktionsterme werden mit UND verknüpft
- KNF = $\Pi\Sigma\overline{x_i}$ (Produkt über Summen)

Beispiel:
$x = (a + b + c) \cdot (a + \bar{b} + \bar{c}) \cdot (\bar{a} + \bar{b} + c) \cdot (\bar{a} + \bar{b} + \bar{c})$

1.2 Elektrotechnische Grundlagen

1.2.1 Zweipole

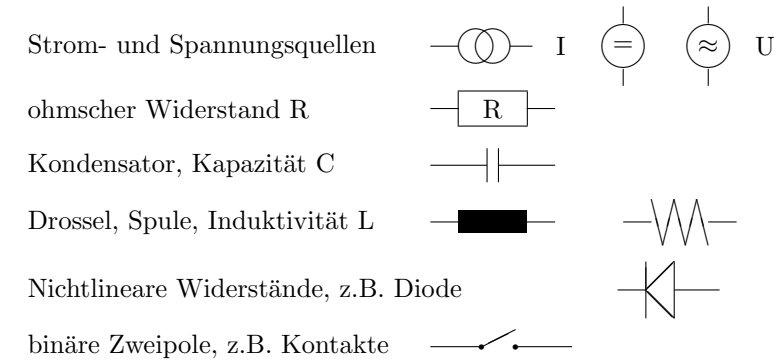


Bild b1p03a

Ohm'sches Gesetz: $U = I \cdot R$

Komplexe Wechselstromlehre
$\tilde{U} = \hat{U} \cdot e^{i\omega t}$ und $\tilde{I} = \hat{I} \cdot e^{i\omega t}$
$\tilde{U} = \tilde{I} \cdot \tilde{R}$ mit $\tilde{R} \in \{R, R_C, R_L\}$

Multipole
Verknüpfungen von Zweipolen zu Multipolen mit. . Maschen (M) mit Knoten (K). (Jeder Knoten kann als Pol betrachtet werden)

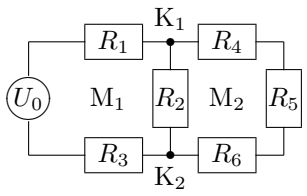


Bild b1p03

Kirchhoffsche Regeln:

Maschenregel: $\sum U_i = 0$

Knotenregel: $\sum I_j = 0$

Vierpole . .Beispiel:

Eingang: U_1, I_1
Ausgang: U_2, I_2

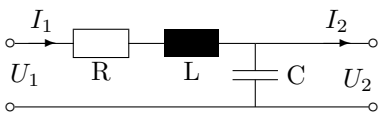
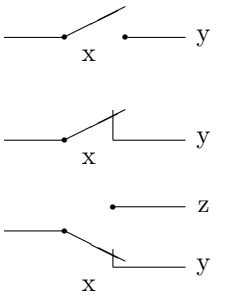


Bild b1p03b

Übertragungsfunktion:
$\tilde{H} = \tilde{U}_2/\tilde{U}_1$

$U_2/U_1 = 1/(1 - \omega^2 LC + i\omega RC)$

1.2.2 Schalter



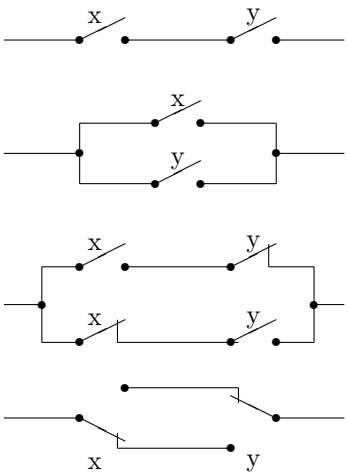
Schließer: Arbeitskontakt (Einschalter) $y = x$

Öffner: Ruhekontakt (Ausschalter) $y = \overline{x}$

Wechsler:
Arbeits- und Ruhekontakt (Umschalter) $y = \overline{x}, z = x$

Bild b1p04

1.2.3 Verknüpfungen



$z = x \cdot y$ UND (AND)

$z = x + y$ ODER (OR)

$z = x \oplus y$ Exklusives ODER (XOR)

Bild b1p05

1.2.4 Technische Realisierungen

Passive Relaisschaltungen:

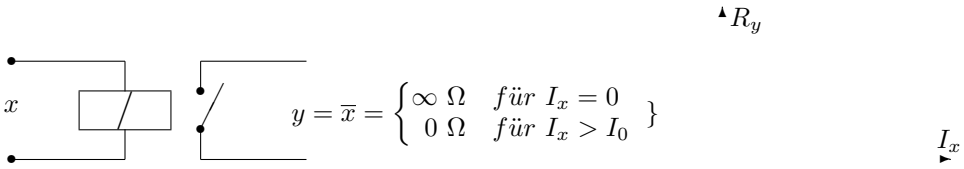
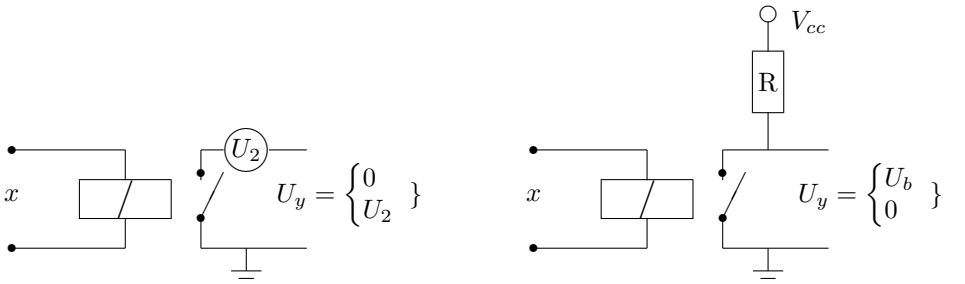


Bild b1p06

Die Eingangsgrößen (Ströme x) entsprechen nicht den Ausgangsgrößen (Durchlaßwiderständen R)

Aktive Schaltungen benötigen eine Hilfsspannung (V_{cc}) und haben die gleichen Eingangs- und Ausgangsgrößen



aktive Relaisschaltung — aktive Relaisschaltung (invertierend)

Bild b1p06a

1.2.5 Transistoren

Transistoren als Vierpole
z.B. Transistor in Emitterschaltung:
$I_1 = Y_{11} \cdot U_1 + Y_{12} \cdot U_2$ . .Eingangswiderstand $R_{in} = 1/Y_{11}$ (groß) . .$Y_{21} \approx 0$
$I_2 = Y_{21} \cdot U_1 + Y_{22} \cdot U_2$ Steilheit $S = Y_{21}$ (Verstärkung) $Y_{22} \approx 0$
Beim FET (Feldeffekttransistor) ist : $I_1 \approx 0$ und $Y_{22} \approx 0$

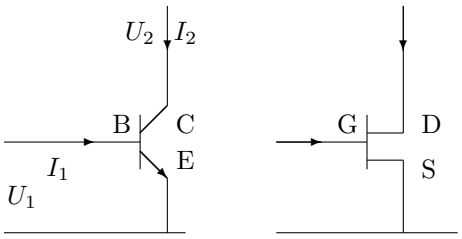


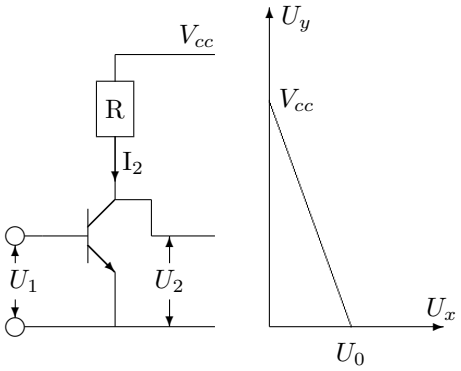
Bild b1p07

. .

NPN-Transistor:
C = Collektor
E = Emitter
B = Basis

FET (verallgemeinert):
D = Drain
S = Source
G = Gate

1.2.6 Verstärker



Schaltung Kennlinie

Bild b1p08

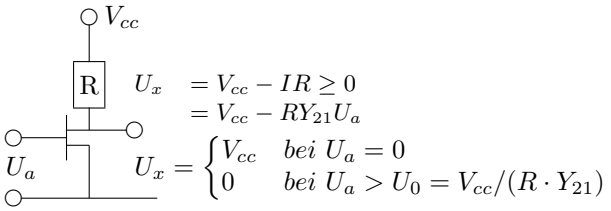
$U_{out} = V_{cc} - I_2 R_2 \quad \text{mit } I_2 = Y_{21} U_1 + 0$
$U_1 \quad = V_{cc} - I_1 R_1 \quad \text{mit } I_1 = Y_{21} U_{in} + 0$
$U_{out} = V_{cc}(1 - R_1 Y_{21}) + R_1 R_2 Y_{21}^2 U_{in} \quad \text{(lineare Verstärkung)}$

1.3 Logische Schaltungen

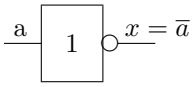
1.3.1 Elementare logische Schaltungen

Aus FETs

1.3.1.1 Inverter (NOT)



Schaltung



Schaltzeichen

Bild b1p09

. .

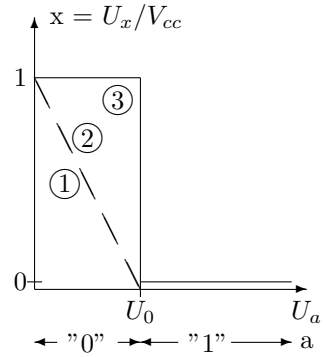


Bild b1p10

Ausgangswert (x) in Abhängigkeit vom Eingangswert (a)
1: bei lineare Verstärkung
2: bei Berücksichtigung von Nichtlinearitäten und Schwellspannungen
3: idealisierter Verlauf für binäre Ein- und Ausgangsvariablen (a, x)

1.3.1.2 Zweistellige Gatter

NAND-Gatter. .

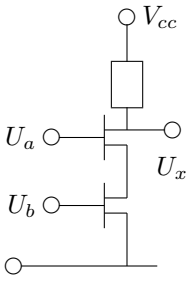


Bild b1p11

$$U_x = \begin{cases} V_{cc} \; falls \; I = 0 \\ 0 \quad falls \text{ beide FETs leiten, } U_a \wedge U_b \geq U_0 \end{cases}$$

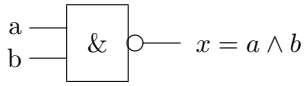


Bild b1p11a

NOR-Gatter . .

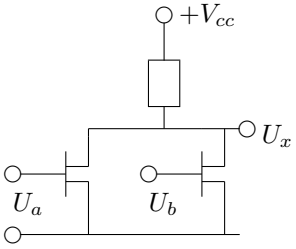


Bild b1p12

$$U_x = \begin{cases} V_{cc} \; falls \; I_a \; und \; I_b = 0 \\ 0 \quad falls \text{ ein FET leitet, } U_a \vee U_b \geq U_0 \end{cases}$$

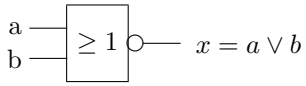
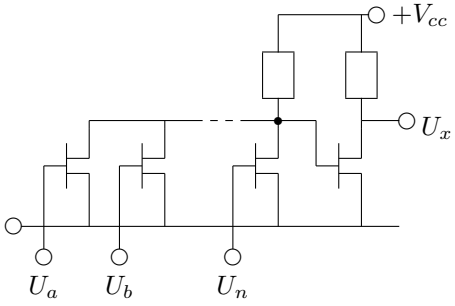


Bild b1p12a

1.3.1.3 Mehrstellige Gatter

– Multi-AND (Vielfach-UND) . .– Multi-OR (Vielfach-ODER)



Schaltzeichen

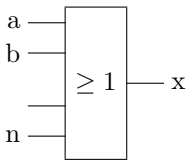
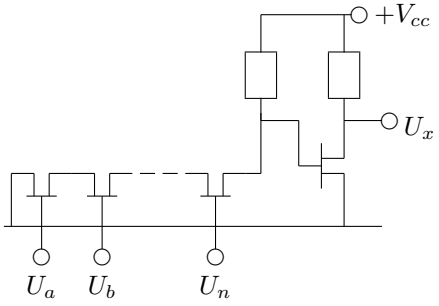


Bild b1p13



Schaltzeichen

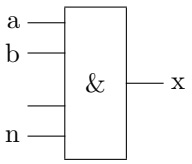


Bild b1p14

1.3.1.4 Schaltnetze für Boole'sche Funktionen

Boole'sche Funktionen können in Normalformen dargestellt werden, welche sich direkt in Schaltnetze umsetzen lassen: jeder Funktionsterm läßt sich durch ein Mehrfach-Gatter darstellen, und ebenso deren Verküpfungen.

– DNF = $\Sigma\Pi x_i$ (ODER-Verküpfung über UND-Gatter)

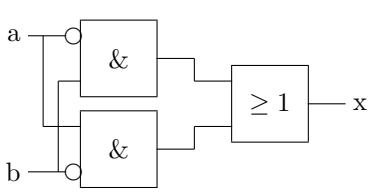
– KNF = $\Pi\Sigma\overline{x_i}$ (UND-Verknüpfung über ODER-Gatter)

Bei Anwendung der De-Morganschen Regeln können aus Min-Termen oder Max-Termen Schaltungen mit gleichen Typen von invertierenden mehrstelligen Gattern verwendet werden.

Min-Terme der DNF = $\Sigma\Pi x_i = \overline{\Pi\overline{\Pi x_i}}$ (nur NAND-Gatter)

Max-Terme der KNF = $\Pi\Sigma\overline{x_i} = \overline{\Sigma\overline{\Sigma\overline{x_i}}}$ (nur NOR-Gatter)

Beispiel 1 Das XOR $x = a \oplus b = a \cdot \overline{b} + \overline{a} \cdot b$



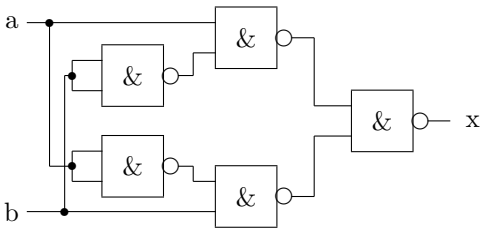


Bild b1p15

Beispiel 2

$x = \bar{a} \cdot \bar{b} \cdot c + \bar{a} \cdot b \cdot \bar{c} + a \cdot \bar{b} \cdot \bar{c} + a \cdot \bar{b} \cdot c$

$x = \overline{\bar{a} \cdot \bar{b} \cdot c} + \overline{\bar{a} \cdot b \cdot \bar{c}} + \overline{a \cdot \bar{b}}$

$x = \overline{\overline{\bar{a} \cdot \bar{b} \cdot c} \cdot \overline{\bar{a} \cdot b \cdot \bar{c}} \cdot \overline{a \cdot \bar{b}}}$

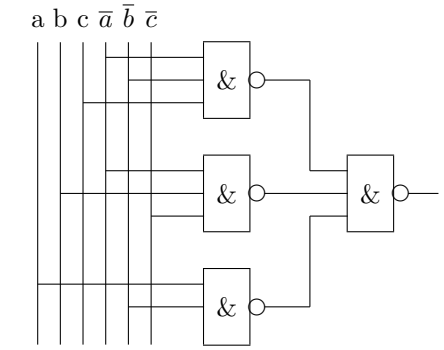


Bild b1p16

1.3.2 Rechnerbausteine

1.3.2.1 Addierer

Halbaddierer a + b = Ergebnis e und Übertrag c

a	b	e	c
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$e = a \oplus b$$
$$c = a \cdot b$$

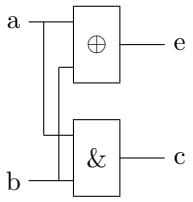


Bild b1p17

Volladdierer $a_i + b_i + c_i$ = Ergebnis e_i und Übertrag c_{i+1}

a_i	b_i	c_i	e_i	c_{i+1}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$e_i = (a_i \oplus b_i) \oplus c_i$$
$$c_{i+1} = a_i \cdot b_i + c_i \cdot (a_i \oplus b_i)$$

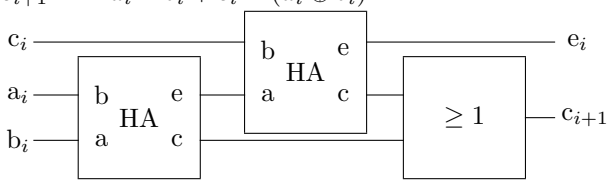
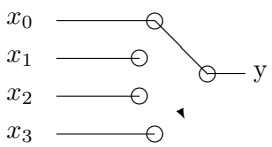


Bild b1p18

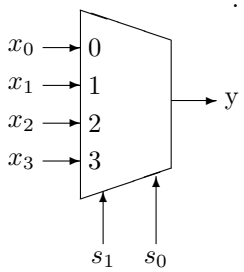
Subtrahierer x = a + (-b) mit -b = $\overline{b} + 1$ (2-er Komplement)

1.3.2.2 Multiplexer

mehrstufige Umschalter



mehrstufiger Umschalter



s_1	s_0	y
0	0	x_0
0	1	x_1
1	0	x_2
1	1	x_3

Bild b1p20

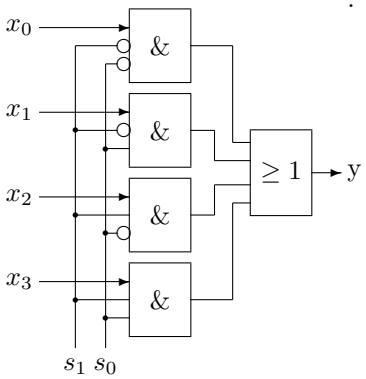
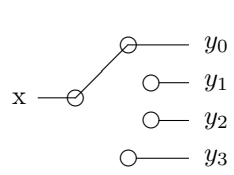


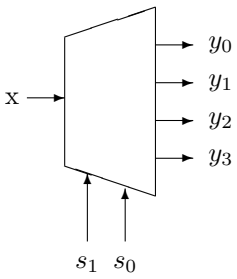
Bild b1p21

Boole'sche Funktion y = f(s) durch statische Belegung der Eingänge eines Multiplexers mit festen Werten x_i

Demultiplexer



mehrstufiger Umschalter



s_1	s_0	y_0	y_1	y_2	y_3
0	0	x	0	0	0
0	1	0	x	0	0
1	0	0	0	x	0
1	1	0	0	0	x

Bild b1p20a

1.3.2.3 PROM

Programmable Read-only Memory
Funktionswerte { d_i } = f(a) werden über Adreßwerte { a_i } aus einem Speicher gelesen, der wahlfrei (RAM) adressierbar ist. Die Zuordnung der Funktionswerte zu den Adressen erfolgt durch "Brennen" der zugehörigen Verbindungen.

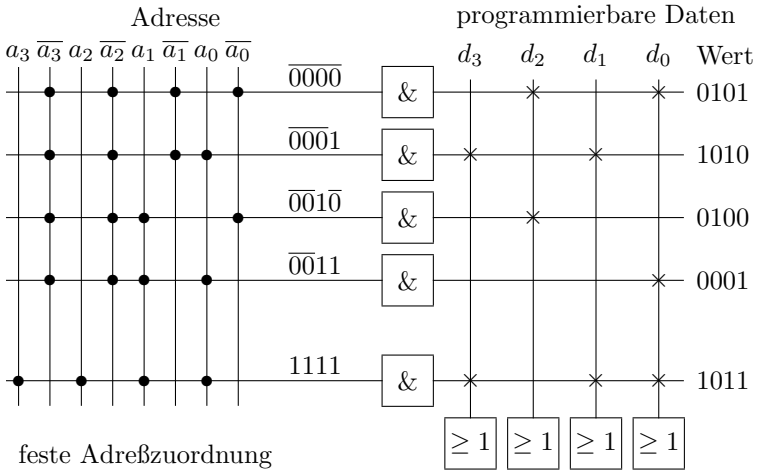


Bild b1p22

1.3.2.4 PAL

Programmable Array Logic
Zum PROM inverse Festlegung von Funktionswerten zu (programmierbaren) Adressen.

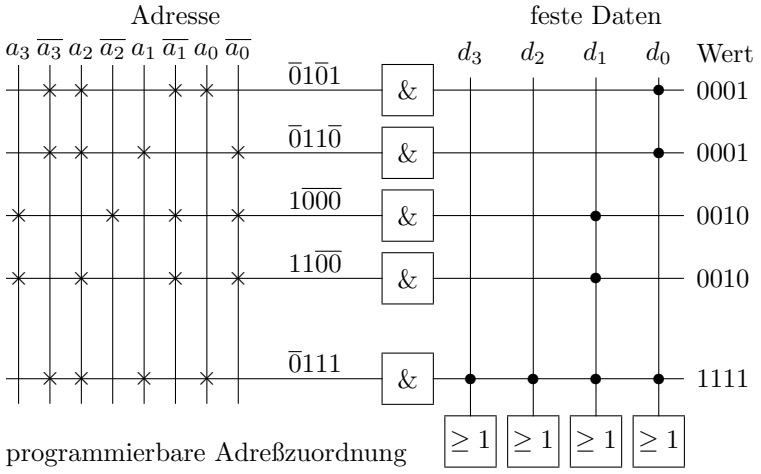


Bild b1p23

1.4 Sequentielle Schaltungen

1.4.1 Zeitverhalten

Gatterlaufzeiten
Schaltverzögerung (delay time t_d)

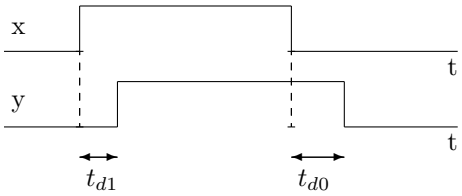


Bild b1p24

Flankensteilheit Anstiegs- und Abfallzeit (rise time t_r, fall time t_f)

Bild b1p25

Ansprechschwellen (threshold)

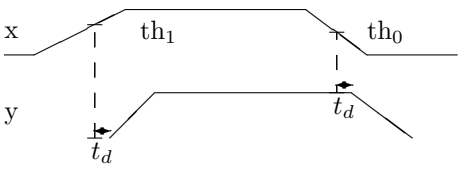
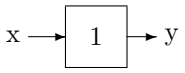
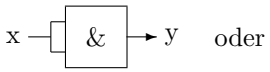


Bild b1p26

. . Identität



z.B.



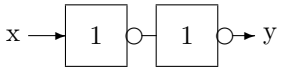


Bild b1p24a

in der Regel ist $t_{d0} \approx t_{d0} \approx t_r \approx t_r$ und hängt von der verwendeten Halbleitertechnologie ab.

Technologie	t_d
TTL	5 ns
ECL	1 ns
CMOS	15 ns

. .

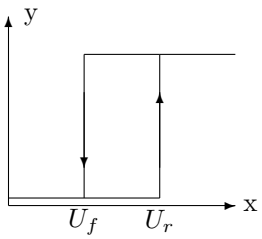


Bild b1p26a Kennlinie (Schmitt-Trigger: $U_f < U_r$)

1.4.2 Bistabile Schaltungen

Elementare Schaltwerke, Automaten

a) Rückgekoppeltes NAND instabile Kippstufe, Multivibrator

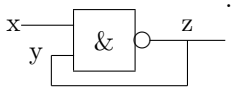


Bild b1p27

x	y	z = $\overline{xy}$
0	0	1
0	1	1
1	0	1 = $\overline{y}$
1	1	0 = $\overline{y}$

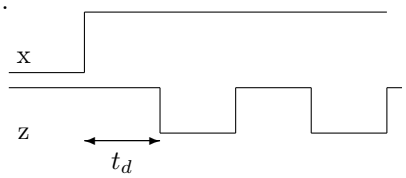


Bild b1p27a

x	y(t)	y(t+t_d)	y(t+2t_d)	
0	0	1	1	y = 1 = const
1	1	0	1	$f_y = 1/2t_d \approx 100MHZ$

b) bistabile Kippstufe

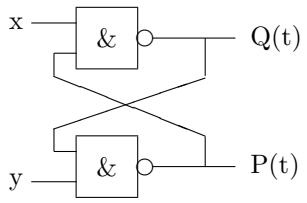
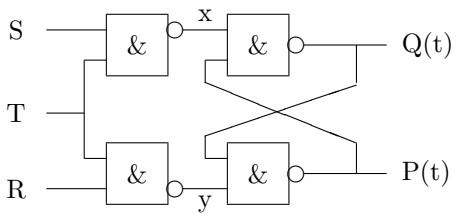
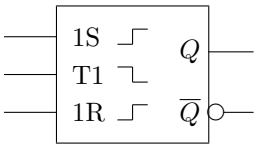


Bild b1p28

x	y	Q(t)	P(t)	Q	P(t+t_d)	Q	P(t+2t_d)	
0	0	1	1	1	1	1	1 = const	$\overline{clear}$
0	1	1	P_t	1	0 = $\overline{Q_t}$	1	0 = const	(set)
1	0	Q_t	1	0	1	0	1 = const	(reset)
1	1	1	1	0	0	1	1 ≠ const	(instabil)
1	1	1	0	1	0	1	0 = const	(bi)stabil
1	1	0	1	0	1	0	1 = const	(bi)stabil
1	1	0	0	1	1	0	0 ≠ const	instabil

c) RS – Flipflop



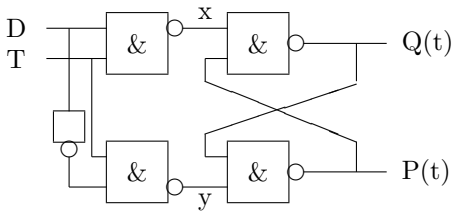


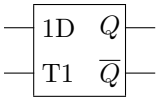
Schaltsymbol

Bild b1p29

R	S	T	x	y	Q	P
0	0	0	1	1	Q_0	$P_0 = \overline{Q_0}$ = const
1	0	0	1	1	Q_0	$P_0 = \overline{Q_0}$
0	1	0	1	1	Q_0	$P_0 = \overline{Q_0}$
1	1	0	1	1	Q_0	$P_0 = \overline{Q_0}$
0	0	1	1	1	Q_0	$P_0 = \overline{Q_0}$ = const
1	0	1	1	0	0	1 (reset)
0	1	1	0	1	1	0 (set)
1	1	1	0	0	1	1 instabil

d) D-Flipflop Speicherelement für 1 Bit
Ausschließen des instabilen Zustands durch D = S = $\bar{R}$





Schaltsymbol

Bild b1p30

D	T	x	y	Q
0	0	1	1	Q_0
1	0	1	1	Q_0
0	1	0	1	0 = D (set)
1	1	1	0	1 = D (reset)

e) JK-Flipflop

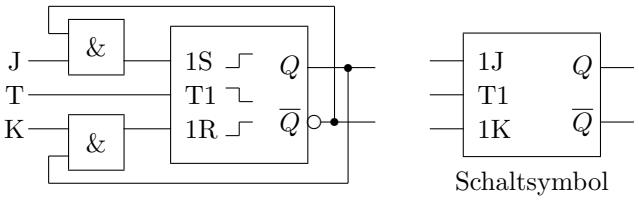


Bild b1p31

J	K	Q_0	S	R	Q(T) für T = 1	
0	0	0	0	0	Q_0	Store
0	0	1	0	0	Q_0	Store
0	1	0	0	0	$0=Q_0$	Reset
0	1	1	0	1	0	Reset
1	0	0	1	0	1	Set
1	0	1	0	0	$1=Q_0$	Set
1	1	0	1	0	$1=\overline{Q_0}$	Kippen
1	1	1	0	1	$0=\overline{Q_0}$	Kippen

f) Master-Slave-Flipflop

Serienschaltung von einem JK– und einem RS–Flipflop. Ein- und Ausgang werden entkoppelt. Das neue Signal erscheint erst nach Beendigung des Taktimpulses (verzögert), an dessen Endflanke (flankengesteuertes Flipflop).

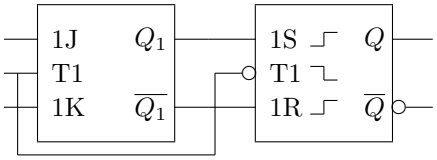


Bild b1p32

J	K	S(T)	Q(T)	Q($\overline{T}$)
0	0	Q_0	Q_0	Q_0
0	1	0	Q_0	0
1	0	1	Q_0	1
1	1	$\overline{Q_0}$	Q_0	$\overline{Q_0}$

g) T-Flipflop

Anwendung eines Master-Slave-Flipflops zum Zählen

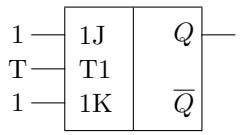


Bild b1p32a

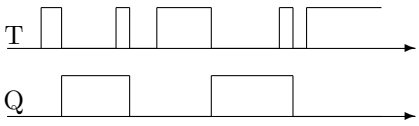


Bild b1p32b

1.4.3 Register

a) Speicherregister, (latch) . . .
Parallelschaltung von n D-Flipflops mit gemeinsamen Ladevorgang (Load)

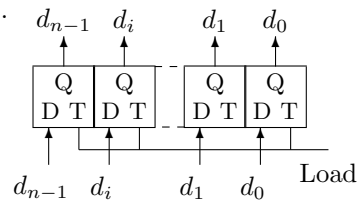


Bild b1p33

b) Zähler (counter) Sequentielle Addition
Serienschaltung von T-Flipflops. Rücksetzen (reset) durch J = K = 0 beim nächsten Taktimpuls (synchrones Reset)
Die Steuerung für Vorwärts/Rückwärtszählen (F/R) erfolgt durch Nutzung des Q- bzw des $\overline{Q}$-Ausgangs (Die Umschaltung kann durch ein XOR simuliert werden)

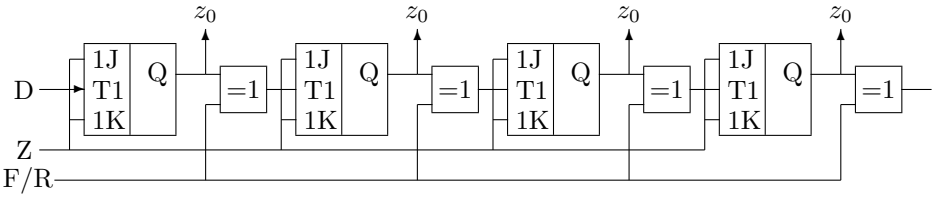


Bild b1p34

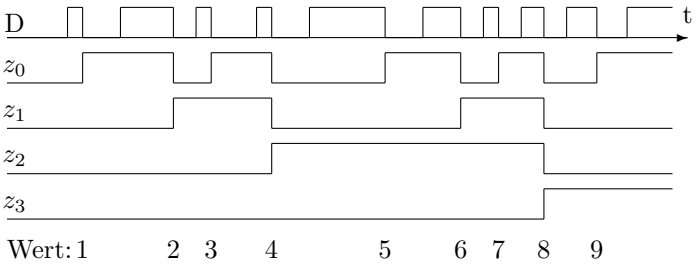


Bild b1p34a

c) Teiler . .

Serienschaltung von Master-Slave-Flipflops und Rücksetzen nach Erreichen bestimmter Werte (binär, dezimal, beliebig), die durch eine logische Verküpfung der Zählerstandswerte bestimmt wird (R = f(z)). Dieser Übertrag (Carry) kann als Eingangstakt für eine nächste Stufe (Stelle) dienen.

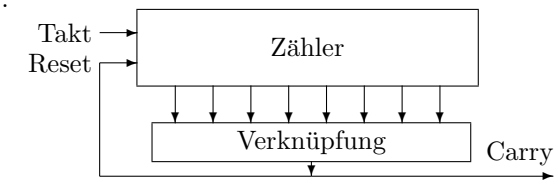


Bild b1p35

Z.B. z = (1001) = 9 ergibt einen Dezimalzähler (0 ... 9).

d) Schieberregister

Serienschaltung von JK-Flipflops mit seriellem Eingang und parallelem Ausgang (Seriell-Parallel-Wandler)

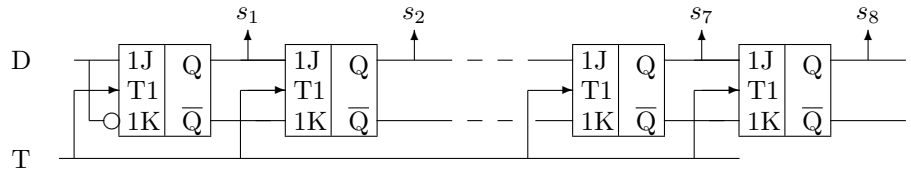


Bild b1p36

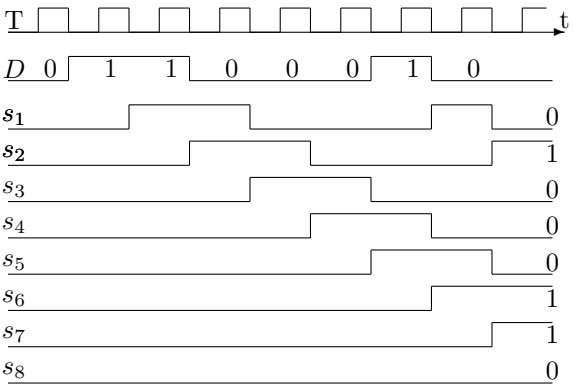


Bild b1p36a

Nach 8 Takten stehen die 8 seriell eingegangenen Bits ”0 1 1 0 0 0 1 0” an den Ausgängen $s_8..s_1$ zur Verfügung

e) Parallel-Seriell-Wandler
Schieberegister mit parallem Laden (Load) und seriellem Ausgang

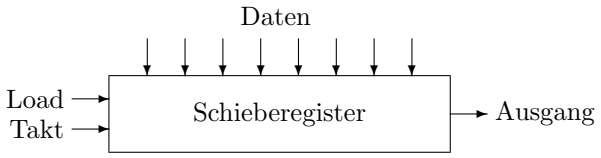


Bild b1p37

f) FIFO (First In First Out)
Registersatz, der von oben geladen und nach unten entleert wird.
Die oben mit einem Ladesignal eingegebenen Daten werden in das unterste noch freie Register gebracht. Wenn das FIFO voll ist, wird ein entsprechendes Signal gegeben.
Mit dem Signal "Hole" werden Daten aus dem untersten Register abgeholt. Die übrigen Daten rücken im FIFO nach. Wenn das FIFO keine Daten mehr enthält, wird das Signal "Leer" abgegeben.
Das FIFO ist ein Speicher mit sequentiellem Zugriff (SAM, sequential access memory).

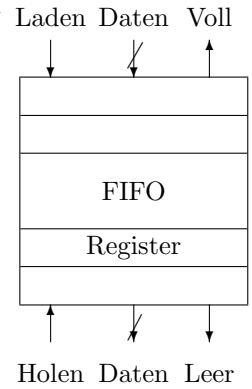


Bild b1p38

Kapitel 2

Systeme, Automaten und Modelle

Kleine Einführung in die Automatentheorie

2.0 Grundbegriffe

a) System (DIN 44 300 T1, DIN 66 201)
Eine Gesamtheit von Objekten, die sich aufgrund ihrer gegenseitigen Beziehungen
– von ihrer Umwelt abhebt,
– davon abgegrenzt erscheint und
– daher ein als Einheit anzusehendes und gegliedertes Ganzes bildet
Die Objekte können sein:
– reale oder abstrakte Gegenstände
– Methoden oder Vorgänge
Aus Systemen können durch
– Unterteilen oder
– Zusammenfügen neue Systeme gebildet werden.
Das bedeutet: Systeme können Objekte in (d.h. Bestandteile von) Systemen sein.

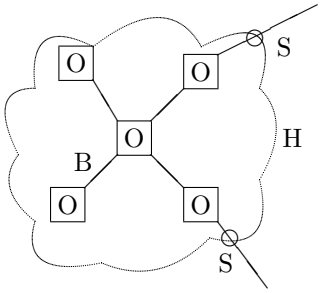


Bild n2p01

..O: Objekte des Systems (objects)
B: Beziehungen zwischen den Objekten (relations, links)
H: Hüllfläche (surface)
S: Schnittstellen zur Umwelt (interfaces)

Beispiele:
1. Haus: Räume (O), Anordnung, Zugänge (B), Dach, Mauern (H), Türen, Versorgung, Entsorgung, Kommunikation (S).
2. Computer: Hardware, Software (O), Signale (B), Gehäuse (H), Netzleitung, Bildschirm, Tastatur (S).

Als Abgrenzung eines Systems gegenüber seiner Umwelt läßt sich eine Hüllfläche denken, die von allen Beziehungen zur Umwelt durchdrungen wird. Die Orte der Durchdringung lassen sich als Schnittstellen auffassen; die dort geltenden Bedingungen beschreiben das Systemverhalten nach außen.

Das Wort "System" sollte stets durch einen Präfix näher bestimmt werden, z.B. Rechnersystem, Versorgungssystem.

Im **ITV** (ISO/IEC 2382-01.01.04A) und im **IEV** (351-01-01) ist "system":
A set of interrelated elements considered in a defined context as a whole and separated from their environment.
NOTE 1: Such elements may be both material objects and modes of thinking as well as the results thereof (e.g. forms of organisation, mathematical methods, and programming languages).
NOTE 2: The system is considered to be separated from the environment and other external systems by an imaginary surface, which can cut the links between them and the considered system.

System (in der deutschen Übersetzung):
System: Gesamtheit untereinander zusammenhängender Elemente, die als Ganzes und von ihrer Umgebung abgegrenzt betrachtet werden, um ein bestimmtes Ziel zu erreichen.
ANMERKUNG 1: Solche Elemente können Gegenstände oder auch Denkweisen und deren Ergebnisse (z. B. Organisationsformen, mathematische Verfahren und Programmiersprachen) sein.
ANMERKUNG 2: Das System wird als von der Umgebung und anderen äußeren Systemen durch eine imaginäre Hüllfläche abgegrenzt gedacht, welche die Verbindungen zwischen diesen Systemen und dem betrachteten System durchschneiden kann.

b) Struktur
Die Menge der Beziehungen zwischen den Teilen eine Ganzen, zwischen den Objekten eines Systems (abstrakt).
Einfachere Definition in ISO/IEC 2382-01.01.3B und IEV 351-01-02 "Struktur":
Gesamtheit der Beziehungen zwischen den Elementen eines Systems.
(engl. structure: The relations among the elements of a system.)
Beispiele:
1. Hierarchie, Baumstruktur
2. Folge, Sequenz
3. Maschennetz
4. Fraktale
5. Abhngigkeit, Versorgung (Erzeuger - Verbraucher)

c) Automat (DIN 19 223)
Ein Automat ist ein künstliches System, das selbstätig ein Programm befolgt. Auf Grund des Programms trifft das System, Entscheidungen, die auf der Verknüpfung von Eingaben mit dem jeweiligen Zustand des Systems beruhen, und Ausgaben zur Folge haben.

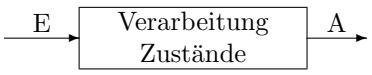


Bild n2p02

EVA-Prinzip:
E: Eingabe
V: Verabeitung
A: Ausgabe

Beispiele:
– inhärente Automaten, z.B. Sicherungsautomat (kein endlicher Automat !)
– festverdrahtete Automaten, z.B. Kaffee-, Zigarettenautomat
– programmierbare Automaten, z.B. SPS (speicherprogrammierbare Steuerung), Computer.
Endliche Automaten haben nur endlich viele Zustände.

d) Zustand (eines Systems):
Die Gesamtheit der aktuellen Werte der veränderlichen Objekte oder Beziehungen eines Systems (Zustandsvektor).
Beispiel: Zigarettenautomat
Schacht: leer/gefüllt (Anzahl der Packungen {0 ... Max})
Geldeinwurf: nichts, zu wenig, ausreichend, zu viel.
Geldvorrat: 0, ...
Bereitschaft: bereit, defekt

2.1 Automaten

Endliche Automaten, Finite State Machines
Sequentielle Maschinen M = { E, Z, A, δ, λ, Z_0 } (eine Menge von Mengen)
E: eine (endliche) Menge von Eingabewerten = Eingabealphabet, -vektor
Z: eine (endliche) Zustandsmenge = Zustandsalphabet, -vektor
A: eine (endliche) Menge von Ausgabewerten = Ausgabealphabet, -vektor
Z_0: Anfangszustand
δ: Zustandsübergangsfunktion, $Z' = Z(T + \delta t) = \delta(Z(t), E(t))$
λ: Ausgabefunktion $A = \lambda(Z, E)$

Schematischer Aufbau:

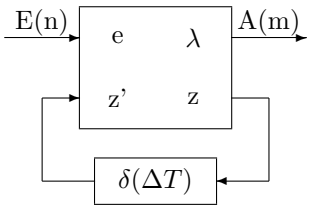


Bild n2p03

Eingabevektor E mit n Komponenten
Ausgabevektor A mit m Komponenten

Rückkopplung mit Verzögerung (ΔT)
– getaktet, synchron: ΔT = 1/ f (Taktfrequenz)
– ungetaktet, asynchron: $\Delta T = \Sigma t_i$ (Laufzeiten)

2.1.1 Moore-Automat

$Z' = \delta(Z, E)$ (immer) . .
$A = \lambda(Z)$
die Ausgabe A hängt nur vom jeweiligen Zustand (Z) ab,
die Ausgabe ist ein stationäres Signal.

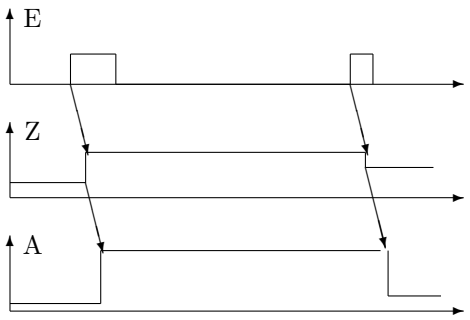


Bild n2p04

Beispiel: Belegung eines Parkhauses
E = Ereignis, ein- oder ausfahrendes Auto
Z = Anzahl der geparkten Autos { 0 .. Max}, δ: Z' = Z $\pm$ 1
A = Anzeige der Anzahl der freien Plätze. { Max .. 0 }. λ: A = Max - Z
Sobald sich die Zahl Z ändert, wird auch A geändert.

2.1.2 Mealy-Automat

$Z' = \delta(Z, E)$ . .
$A = \lambda(Z, E)$
die Ausgabe A hängt von der Eingabe und vom aktuellen Zustand ab.
Die Ausgabe ist ein (transientes) Ereignis, ein Impuls.

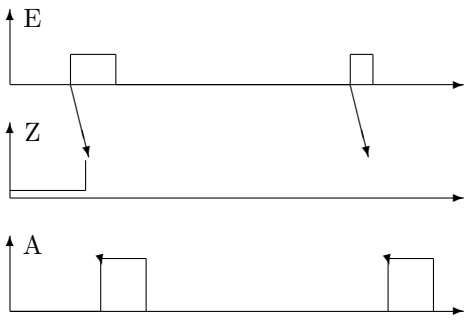


Bild n2p05

Beispiel: Zigarettenautomat
E = Eingabe von Geld,
Z = gespeicherte Zigarettenschachteln Z und eingegebenes Geld G
A = Ausgabe einer Zigarettenschachtel und/oder Geldrückgabe. Nach der Eingabe von ausreichend Geld (λ: 5 DM/Schachtel) erfolgt die Ausgabe als Ereignis, und das System geht in einen neuen Zustand über (δ: Z' = Z - 1, G' = G + x).

2.2 Zustandsdiagramme und -tafeln

Beschreibung von Automaten

2.2.1 Moore-Automat

Zustandstafel

	Eingaben			
Zustand	E_1	E_2	E_n	Ausgabe λ
Z_0	Z_a	Z_b	Z_c	A_1
Z_i		δ: Folgezustände		A_i
Z_m	Z_x	Z_y	Z_z	A_n

Zustandsdiagramm
Knoten (Kreise): Zustände und Ausgaben
Kanten (Pfeile): Eingaben, Ereignisse

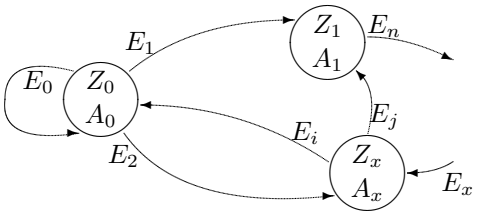


Bild n2p06

Strukturierung
Verfeinerung durch Aufgliedern und Vergröberung durch Zusammenfassen

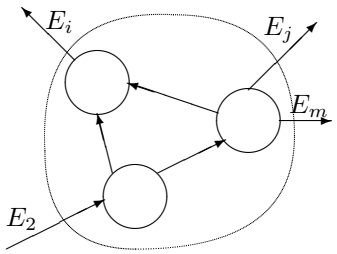


Bild n2p07

Beispiel 1 Das JK-Flipflop

Z = {0, 1} = Q
E={J, K, T}
λ: A=Z
δ: Z'=δ(Z,E) gemäß Wertetabelle

J	K	Q_0	Q(T) für T = 1	
0	0	0	0	} Q_0
0	0	1	1	
0	1	0	0	} Reset
0	1	1	0	
1	0	0	1	} Set
1	0	1	1	
1	1	0	1	} $\overline{Q_0}$
1	1	1	0	

Zustandstafel

	E = (J, K)				
Zustand	(0,0)	(0,1)	(1,0)	(1,1)	Ausgabe
Z_0	Z_0	Z_0	Z_1	Z_1	Q=0
Z_1	Z_1	Z_0	Z_1	Z_0	Q=1

Zustandsdiagramm

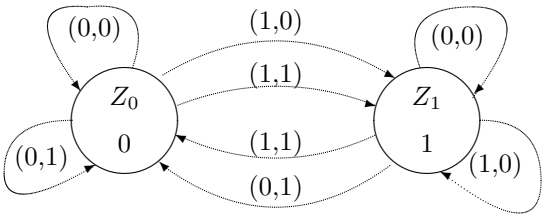


Bild n2p09

2.2.2 Mealy-Automat

Zustandstafel

	Eingaben			
Zustand	E_1	E_2	E_3	E_n
Z_0	Z_a/A_{01}	Z_b/A_{02}	Z_c/A_{03}	Z_d/A_{0n}
Z_i	Z_j/A_{i1}	Z_k/A_{i2}	Z_l/A_{i3}	Z_p/A_{in}
	Folgezustände/Ausgaben			
Z_m	Z_r/A_{m1}	Z_s/A_{m2}	Z_u/A_{m3}	Z_v/A_{mn}

Zustandsdiagramm

Knoten (Kreise): Zustände Z
Kanten (Pfeile): Eingaben und Ausgaben (E/A),
auch die leere Ausgabe (-) ist möglich.

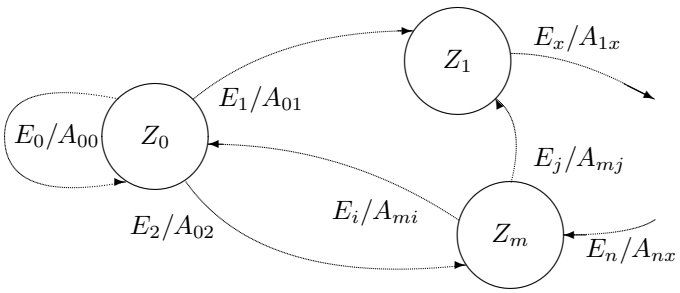


Bild n2p08

Strukturierung
Verfeinerung durch Aufgliedern und Vergröberung durch Zusammenfassen

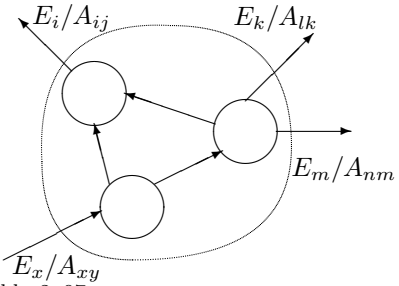


Bild n2p07a

Beispiel 2 Der Kaffee-Automat

Zustände: bereit (Z_0), bereit mit n DPf Guthaben (Z_{nn})
Eingaben: Münzen á 10, 50 Dpf = E(10), E(50) (nicht berücksichtigt ist eine Auswahl)
Ausgabe: nichts (-) oder Kaffee (+ Rückgeld)
λ: Kaffeeausgabe falls mindestens 70 DPf eingeworfen wurden
δ: Z'=δ(Z,E(x)) gemäß Wertetabelle

Zustandsdiagramm

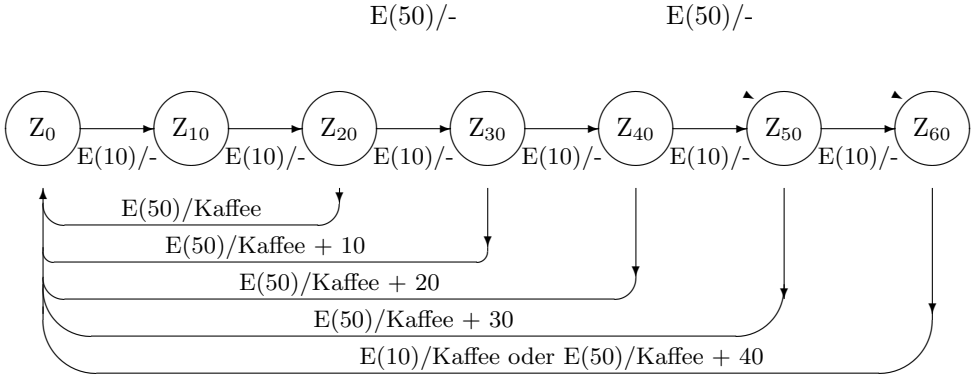


Bild n2p10

Zustandstafel

Zustand	E(10)	E(50)
Z_0	Z_{10}/-	Z_{50}/-
Z_{10}	Z_{20}/-	Z_{60}/-
Z_{20}	Z_{30}/-	Z_0/Kaffee
Z_{30}	Z_{40}/-	Z_0/Kaffee + 10
Z_{40}	Z_{50}/-	Z_0/Kaffee + 20
Z_{50}	Z_{60}/-	Z_0/Kaffee + 30
Z_{60}	Z_0/Kaffee	Z_0/Kaffee + 40

2.3 Netze

Netze aus Instanzen und Kanälen, NIK

Beschreibung statischer und stationärer Strukturen und Systeme (aus DIN 66 200)

Grundelemente

Knoten: 1. Instanzen = Verarbeitungseinheiten mit Entscheidungsbefugnis (Rechtecke)
2. Kanäle = Objekte, die ver- oder bearbeitet werden, z.B. Daten (Kreise)
Kanten: Beziehungen mit Flußrichtung, Verarbeitungsfluß (Pfeile):

Grundstruktur

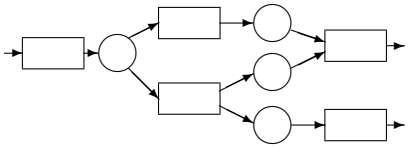


Bild n2p11

Verknüpfungsregel:
Nur unterschiedliche Knoten dürfen durch Pfeile verknüpft werden (EVA-Prinzip).

Verfeinerung / Zusammenfassung
Nur Knoten gleichen Typs dürfen an den Randstellen einer neuen Einheit sitzen. Sie legen damit den Typ des neuen Knotens fest.

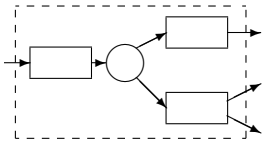


Bild n2p12

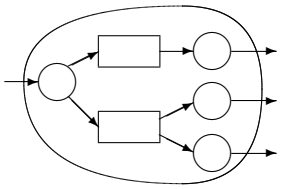


Bild n2p12a

Beispiel 1: Programmerstellung

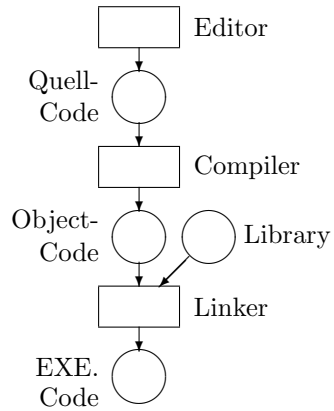


Bild n2p13

Beispiel 2: Sequentieller Addierer

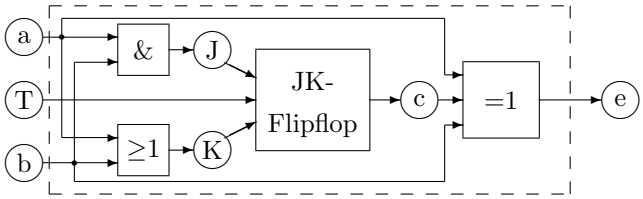


Bild n2p13a

Man überlege sich anhand eines Impulsdiagramms die Wirkungsweise dieses Addierers.

2.4 Petri-Netze

2.4.1 Statische Petri-Netze

Variante der Netze aus Instanzen und Kanäle (NIK)

Anwendung: Beschreibung von Beziehungen in (abgeschlossenen) Systemen, d.h. zur Strukturbeschreibung von Systemen.

Strukturelemente:

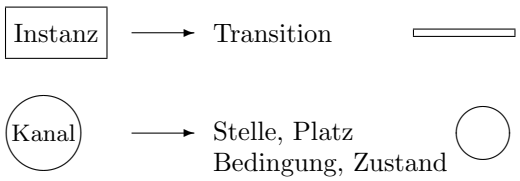


Bild n2p14

Grundstruktur:

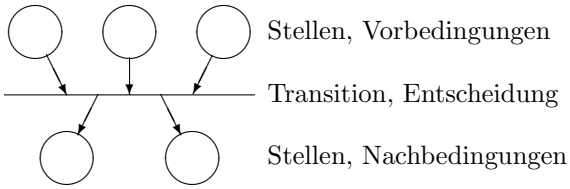


Bild n2p14a

Strukturvarianten:

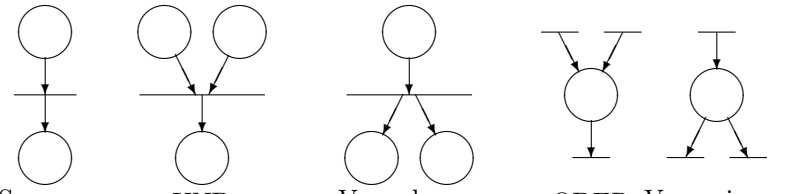


Bild n2p15

Strukturierung (Verfeinerung / Zusammenfassung):

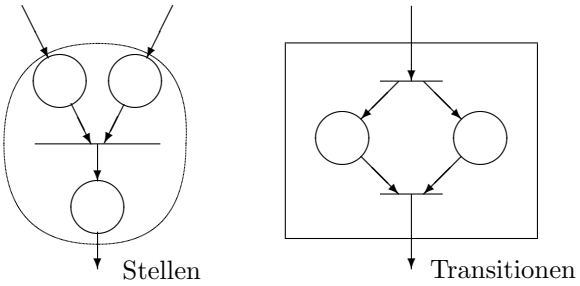


Bild n2p16

2.4.2 markierte Petri-Netze

Anwendung: Beschreibung der Dynamik, d.h. der zeitlichen Kopplungen in Systemen.

a) Beschreibung von Zuständen

Zustände von Stellen werden durch Marken (gefüllte Kreise) beschrieben.

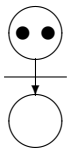


Bild n2p17

b) Beschreibung von Zustandsänderungen

(transiente Vorgänge, deren Zeitdauer unbeobachtbar kurz ist)

Regeln:

- Alle Vorbedingungen einer Transition müssen erfüllt sein, d.h. alle Eingangsstellen müssen jeweils mindestens eine Marke enthalten (Anfangszustand).
- Ereignis (Verarbeitung): Die Transition schaltet und
  - verbraucht je eine Marke aus jeder Eingangsstelle,
  - erzeugt je eine Marke in jeder Ausgangsstelle (Nachbedingung)

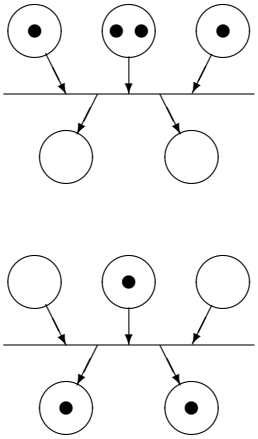


Bild n2p18

c) Beschreibung von Systemverhalten

Lebendigkeit (Liveness): jede Transition wird mindestens einmal aktiviert.

Eindeutigkeit (Determiniertheit): Jeder Gesamtzustand (Verteilung der Marken auf die Stellen) hat einen eindeutig definierten Folgezustand.

Gegenbeispiele:

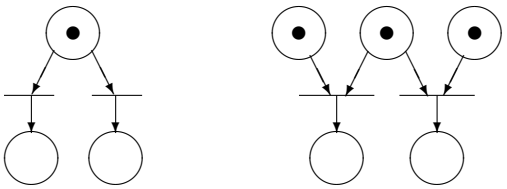


Bild n2p19

Die Marke (in der mittleren Stelle) kann nur für eine der beiden Transitionen genutzt (verbraucht) werden. Eine Vorschrift, für welche, liegt hier nicht vor. Ein Lösung kann in unterschiedlicher Bewichtung der einzelnen Kanten (Pfeile) erfolgen (bewichtete Petrie-Netze), durch die unterschiedliche Prioritäten vergeben werden.

Verklemmung (deadlock): tritt ein, falls im Lauf der Zeit auf, wenn die Anzahl der Marken so abnimmt, daß keine Transition mehr schalten kann.

Beispiel:

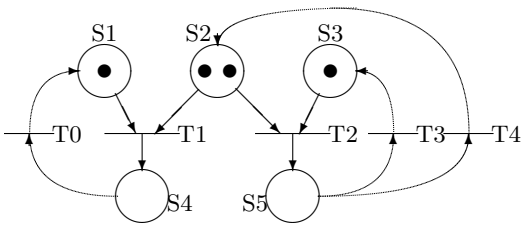


Bild n2p20

Nach dem ersten Durchlauf (Zünden der Transitionen T1 und T2) sind nur noch 2 Marken übrig. Die linke wandert aus S4 wieder in ihre Startposition in S1, für die rechte (aus S5) kann entweder nach S2 oder nach S3 wandern. Im ersten Fall kann die Transition T1 noch einmal zünden, die T2 nie wieder. Im anderen Fall wird S3 besetzt und keine der Transitionen kann zünden, da S2 leer bleibt.

Überlauf (overflow): wenn sich im Lauf der Zeit die Zahl der Marken in einer Stelle stark vermehrt, kann die Kapazität dieser Stelle überschritten werden (wie der Überlauf in einem Datenspeicher - "Festplatte voll!!") und das System geht in eine Verklemmung, die sich aber unter Umständen nach einiger Zeit wieder auflösen kann.

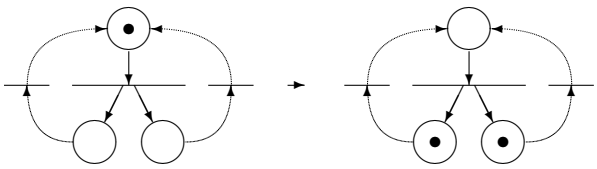


Bild n2p21

2.4.3 Formale Beschreibung von Petri-Netzen

Statische Beschreibung
Alle Transitionen können durch die eingehenden und die ausgehenden Stellen in Form einer Tabelle beschrieben werden. Z.B. (p20)

Transition	Eingangsstellen	Ausgangsstellen
T0	S4	S1
T1	S1 S2	S4
T2	S2 S3	S5
T3	S5	S3
T4	S5	S2

Eine äquivalente Beschreibung der Stellen, mit anliefernden und abnehmenden Transitionen ist ebenso möglich.

Dynamische Beschreibung
Die dynamische Beschreibung soll hier für getaktete Systeme gezeigt werden, wo zu jedem Zeitintervall ein bestimmter Gesamtzustand, d.h. eine bestimmte Verteilung der Marken auf die Stellen gegeben ist. Die dann schaltenden Transitionen sind im rechten Teil durch 'x' markiert. Bei Mehrdeutigkeiten müssen ab dem entsprechenden Zustand 2 oder mehr Folgetabellen geführt werden.

Als Beispiel wird die Verklemmung des vorhergehenden Abschnitts (Bild 20) beschrieben.

Zustand	Stellen					Transitionen					
Z	S1	S2	S3	S4	S5	T0	T1	T2	T3	T4	
Z0	1	2	1	0	0	-	x	x	-	-	
Z1	0	0	0	1	1	x	-	-	x	-	1. Alternative
Z2	1	0	1	0	0	-	-	-	-	-	deadlock
Z1a	0	0	0	1	1	x	-	-	-	x	2. Alternative
Z2a	1	1	0	0	0	-	x	-	-	-	
Z3a	0	0	0	0	0	-	-	-	-	-	deadlock

2.4.4 Anwendungen von Petri-Netzen

Semaphore Nachrichtenstelle zum gegenseitigen Aus-. . schluß (mutual exclusion) bei der Belegung von Betriebsmitteln, z.B. Drucker in einem Netzwerk (Sperrsynchronisation).
Konkurrierende Prozesse P1 und P2 können quasi gleichzeitig eine Druckanforderung erzeugen (S1 und S2). Da aber nur ein Drucker vorhanden (nur eine Marke in S0) ist, kann nur ein Auftrag bearbeitet werden, nur eine Transition (T1) wird aktiv. Nach Durchlaufen des Druckprozesses (S3) fährt der Prozeß P1 mit der Transition T3 fort, geht in einen neuen Abschnitt S5 und gibt den Drucker frei, indem eine neue Marke in S0 erzeugt wird, mit der Prozeß P2 weitermachen kann.

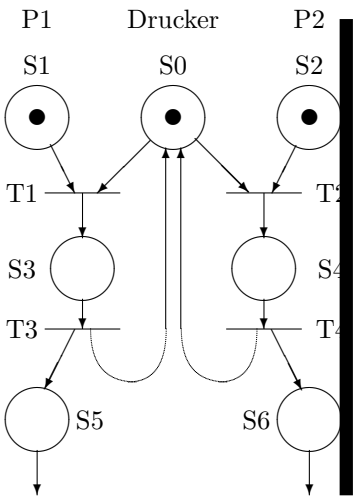


Bild n2p22

Z	S0	S1	S2	S3	S4	S5	S6	T1	T2	T3	T4	
Z0	1	1	1	0	0	0	0	x	-	-	-	1. Prozess druckt
Z1	0	0	1	1	0	0	0	-	-	x	-	Drucken beendet
Z2	1	0	1	0	0	0	0	-	x	-	-	2. Prozess druckt
Z3	0	0	0	0	1	0	0	-	-	-	x	Drucken beendet
Z4	1	0	0	0	0	0	1	-	-	-	-	Drucker bereit

FIFO

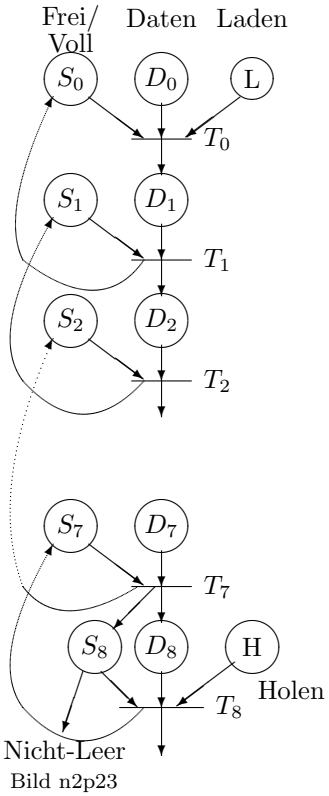


Bild n2p23

. .Die Erstellung der statischen und der dynamischen Tabellen soll dem Leser überlassen bleiben.

2.4.5 Varianten von Petri-Netzen

a) gewichtetet Petri-Netze
Bei dieser Variante werden den einzelnen Zuführungen von Transitionen zu Stellen unterschiedliche Gewichte zugeordnet, damit können Konkurrenzsituationen besser aufgelöst werden. (vgl. w.o.)

b) farbige Petri-Netze
Bei dieser Variante können den Marken unterschiedliche Farben zugeordnet werden, derart daß für die Aktivierung einer Transition stets Marken gleicher Farbe notwendig sind.

c) Petri-Netze mit Nachbedingungen
Bei dieser Variante müssen für die von einer bestimmtem Transition erzeugten Marken bestimmte Bedingungen erfüllt sein, so z.B. daß die Ausgangsstellen leer sind.

Kapitel 3

Rechnerarchitekturen

3.0 Der Rechner als System

Grobstrukturen:
- Hardware: Zentraleinheit und Peripherie
- Software: Programme und Daten

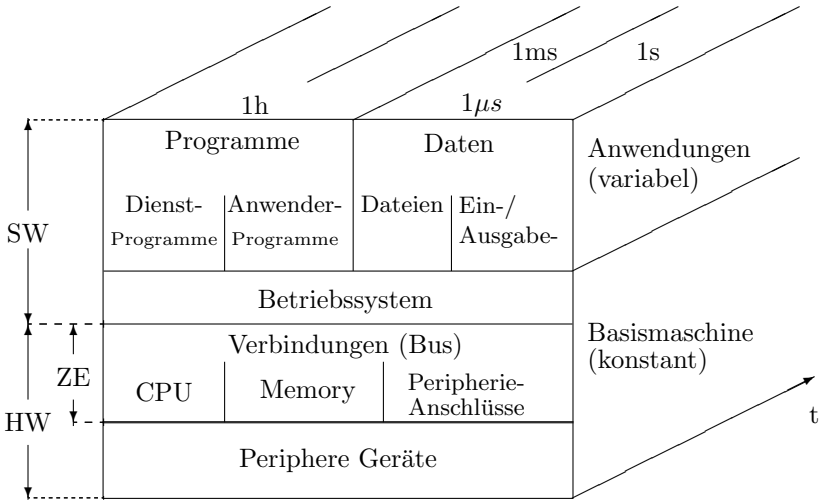


Bild n3p00

Das Bindeglied zwischen Hard- und Software ist die CPU (Central Processing Unit, der Zentralprozessor). Deren Leitwerk liest die Instruktionen aus dem Arbeitsspeicher, interpretiert sie und führt sie aus. Dabei steuert sie das interne Rechenwerk, den CPU-internen Datenfluß und den gesamten Datenverkehrüber über das Verbindungsnetzwerk (meist einen Bus).

Betriebssysteme und Betriebsarten:

1.	Single User z.B. MS-DOS ohne Datenschutz	Multi User Unix mit Datenschutz (Benutzerkennung, Passwort)
2.	Single Tasking ein einziges (komplexes) Programm in Betrieb nur Spezialrechner	Multi Tasking Mehrprogrammbetrieb mehrere Programme in Konkurrenz um 1 oder n CPU (concurrency) Zuteilungsstrategien (sheduling): – time-sharing – resource-sharing (MS-DOS), Unix
3.	Single Processing Ein Prozessor (1 CPU) z.B. Prozeßrechner in Geräten	Multiprocessing Mehrere Prozessoren ($n \leq 2^{16} = 65\,536$) Massive Parallel Computing z.B. Hypercube, Suprenum

Der Rechner als endlicher Automat:

- Eingabealphabet: Binärwerte
- Ausgabealphabet: Binärwerte
- Zustandsalphabet: $Z = \{Z_0 ... Z_z\}$

Bei einem Binärrechner ist die Menge $z = 2^x$, wobei x die Zahl aller Binärvariablen (Bits) ist.

Abschätzung: x = c + m + p

c: Bits innerhalb der CPU ca. 20 Register á 16 bit:

m: Kapazität des Arbeitsspeichers, typ. 1 MB = 1 000 000 * 8 bit

p: Kapazität der Peripherie, einschließlich Bildschirm und Tastatur.

Einheit	Menge	Zugriffszeit
CPU	c = 320	1 - 20 ns
Arbeitsspeicher	$m = 8 * 10^6$	1 μs
Peripherie:		
Tastatur	4 * 102 Tasten	1 s
Bildschirm	1 Mio Pixel	10 ms (100 Hz)
Festplatte	1 GB = $8 * 10^9$ Bit	10 ms
	$p > 8 * 10^9$	

Damit wird x im wesentlichen durch p (die Speicherkapazität der Peripherie) bestimmt; im folgenden wird mit $x = p = 8 \cdot 10^9$ gerechnet.

Abschätzung:

$z = 2^x = 2^{8 \cdot 10^9} = 2^{10 \cdot 8 \cdot 10^8} = (2^{10})^{8 \cdot 10^8} \approx (10^3)^{8 \cdot 10^8} = 10^{3 \cdot 8 \cdot 10^8} = 10^{24 \cdot 10^8} = 10^{2.4 \cdot 10^9}$

Das ist eine Zahl mit 2,4 Milliarden Nullen. Es ist also eine sehr große Zahl, die aber endlich ist !

Man überlege sich die Zeit für das Ausdrucken einer so großen Zahl.

3.1 Rechnerkonfiguration

Rechner := Zentraleinheit + Peripherie

Zentraleinheit: Rechnerkern

Peripherie:
- Datenendgeräte (Terminals): Datensichtgerät (Monitor + Tastatur), Drucker, Plotter,
- Massenspeicher (Hintergrundspeicher): Magnetplatte (auch Floppydisk), Magnetband (auch Cassetten),
- Kommunikationsperipherie: Modem, Ethernetanschluß o.ä.
- Prozeßperipherie, insbesondere A/D- und D/A-Wandler

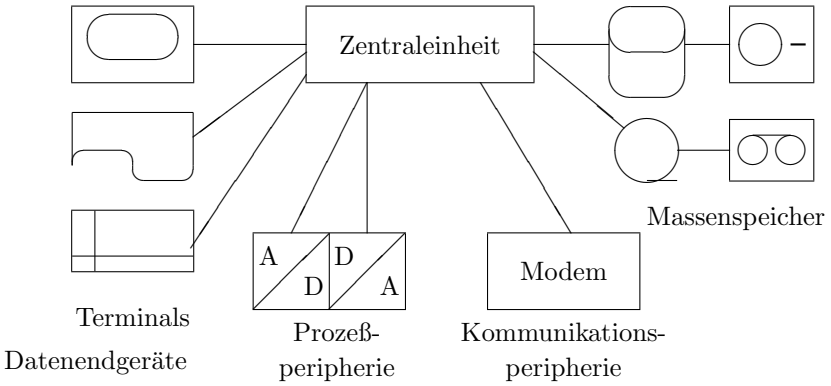


Bild n3p01

Begriffsbildungen:

Betrachtungs-einheiten	Baueinheiten / Beispiel	Funktionseinheiten / Beispiel
	Baueinheit: Haus	Funktionseinheit: Wohnhaus
	Baugruppe: Raum	Funktionsgruppen: Zugänge
	Bauelement: Schraube	Funktionselement: Befestigung

Computer:
Der sog. Rechnerkern umfaßt als Baueinheit oft mehrere Funktionseinheiten: Die Zentraleinheit, Massenspeicher, Stromversorgung.
Eine Funktionseinheit, wie z.B. der Arbeitsspeicher, kann auf mehrere Baueinheiten (Speicherwerke) verteilt sein.

3.2 Die Zentraleinheit

Definition: Alle Funktionseinheiten, die vom Zentralprozessor (CPU) direkt erreicht (adressiert) werden können.

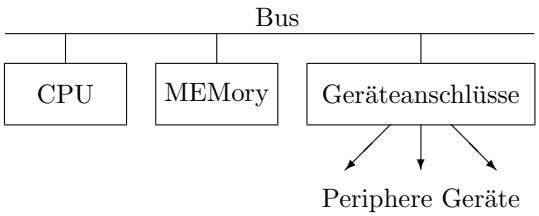


Bild n3p02

- Die CPU ist der eigentliche (einzige) aktive Kern des Rechners
- Der Arbeitsspeicher (MEMory) enthält Daten und Instruktionen
- Die Geräteanschlüsse sorgen für die Kommunikation mit den peripheren Geräten (oft enthalten sie eigene Prozessoren)
- Der Bus verbindet alle Funktionseinheiten (es können auch andere Topologien eingesetzt werde, z.B. Ring oder Stern)

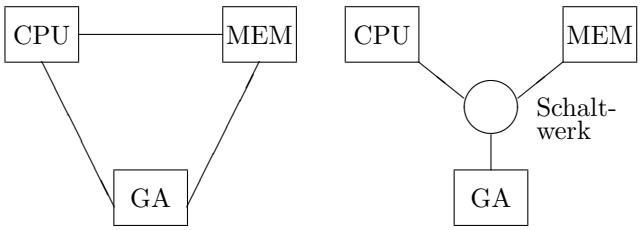


Bild n3p02a

3.2.1 Die CPU

- Leitwerk: interpretiert Instruktionen anhand eines Mikroprogramms, das in einem ROM gespeichert ist.
- Rechenwerk: führt Operationen aus, meist nur auf ganze Zahlen (Integer)
  Der Kern ist die ALU (Arithmetic and Logic Unit)
- Coprozessoren für Gleitpunktoperationen (Floating Point Processor), Textoperationen (Byte Processor)
- Register enthalten die wichtigsten Daten
- Der Zugang zum Bus ist meist gepuffert, d.h. hier werden Zwischenspeicher für Adressen und Daten eingesetzt, die einen größeren Umfang annehmen können und dann als Cache bezeichnet werden.

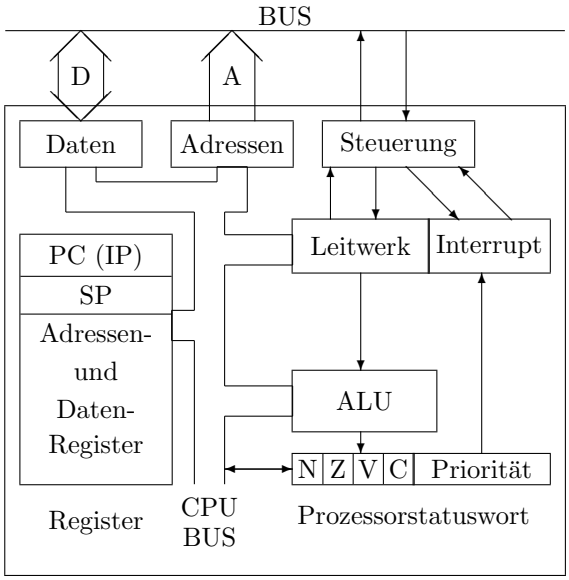


Bild n3p03

..

Die wichtigsten Register der CPU

PC (IP): Der Program Counter (Instruction Pointer) zeigt auf die nächste zu bearbeitende Instruktion

SP: Der Stack Pointer (Kellerzeiger) zeigt auf die zuletzt auf dem Stack benutzte Adresse (LIFO, Last In First Out)

Die Flags im Prozessorstatuswort

N (negativ): Resultat war negativ
Z (zero): Resultat war Null
V (oVerflow): Arithmetischer Überlauf
C (carry): Übertrag

- Das Unterbrechungswerk (Interrupt Arbitrator, IA) bearbeitet externe Unterbrechungsanforderungen, indem es geeignet Hilfsprogramme (Interrupt Service Routinen) aufruft, die vom Betriebssystem zur Verfügungs gestellt werden müssen.
- Der Zugang zum Bus ist meist gepuffert, d.h. hier werden Zwischenspeicher für Adressen und Daten eingesetzt, die einen größeren Umfang annehmen können und dann als Cache bezeichnet werden.

Das Mikroprogramm
Jede Instruktion, die aus dem Arbeitsspeicher gelesen wird, wird im Leitwerk anhand eines Mikroprogramms interpretiert. Das Mikroprogramm ist festverdrahtet, bzw. in einem ROM gespeichert. (Darstellung als Jackson-Struktogramm)

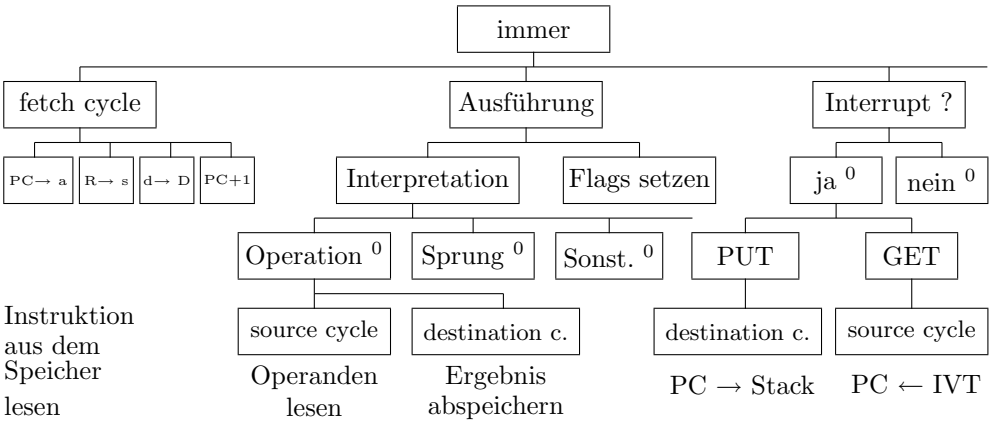


Bild n3p04

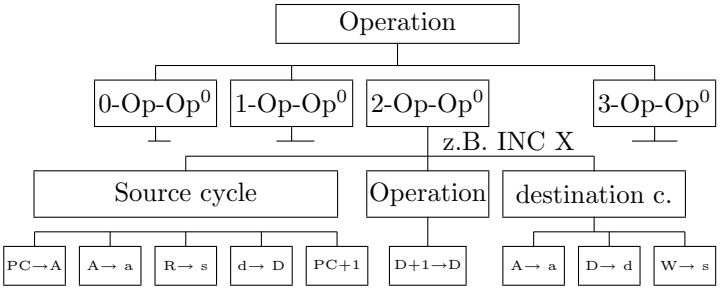


Bild n3p04b

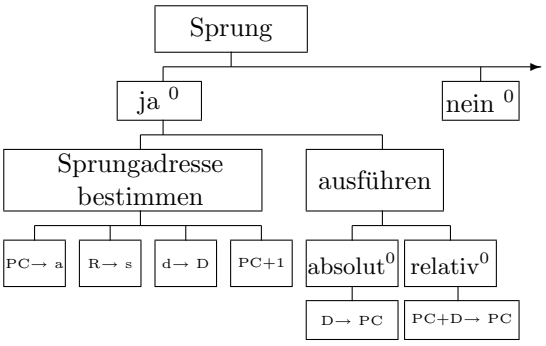


Bild n3p04a

3.2.2 Der Bus

Bustypen:
- parallele Busse, Daten- Adreß- und Steuerbus
- serielle Busse (in der Zentraleinheit selten)
- synchrone (mit Taktsignal) und asynchrone Busse
- Adressen und Daten im Raummultiplex (eigene Leitungen)
- Adressen und Daten im Zeitmultiplex (nacheinander auf den gleichen Leitungen)

a) Das Busprotokoll für einen synchronen Bus (im Raummultiplex)
Adressen und Daten werden als Worte parallel übertragen.

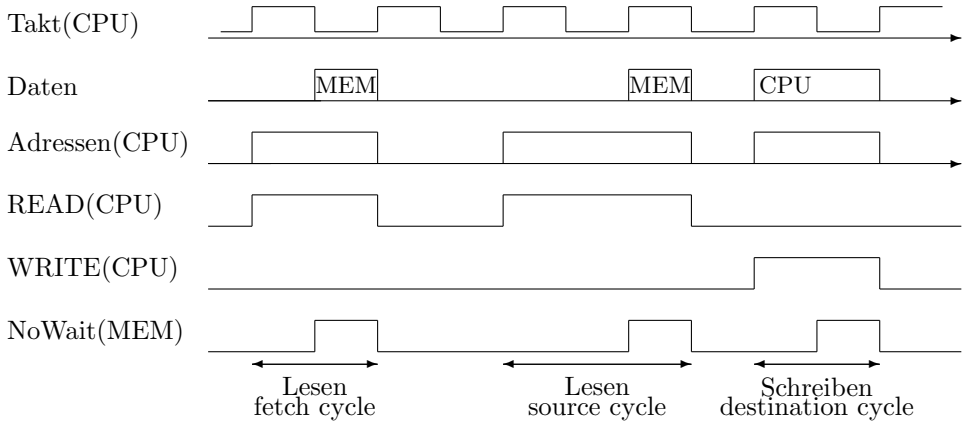


Bild n3p05a

b) Das Busprotokoll für einen asynchronen Multiplex-Bus (im Zeitmultiplex)
Adressen und Daten werden als Worte auf denselben Leitungen nacheinander übertragen. Die Steuersignale sorgen im Handshaking-Verfahren für eine gesicherte Datenübertragung.

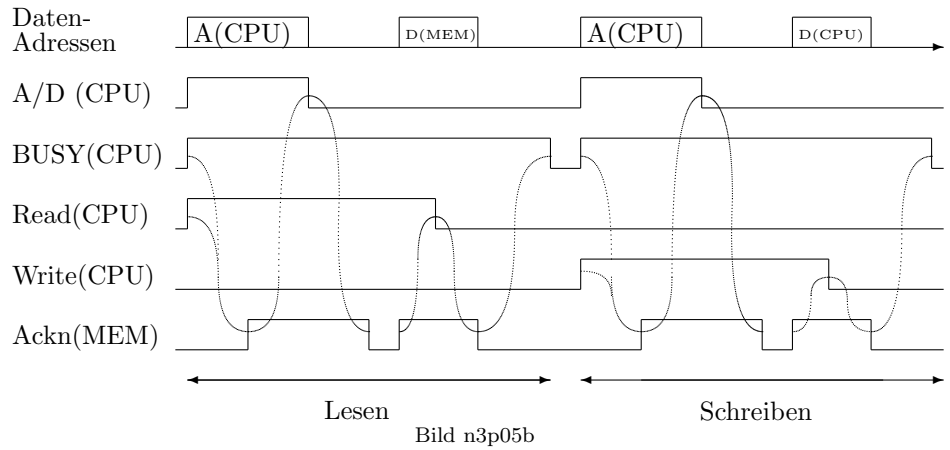


Bild n3p05b

c) Interrupts
Die Bussignale bei einer Interrupt-Anforderung (READ- bzw. WRITE-Signale sind hier weggelassen)

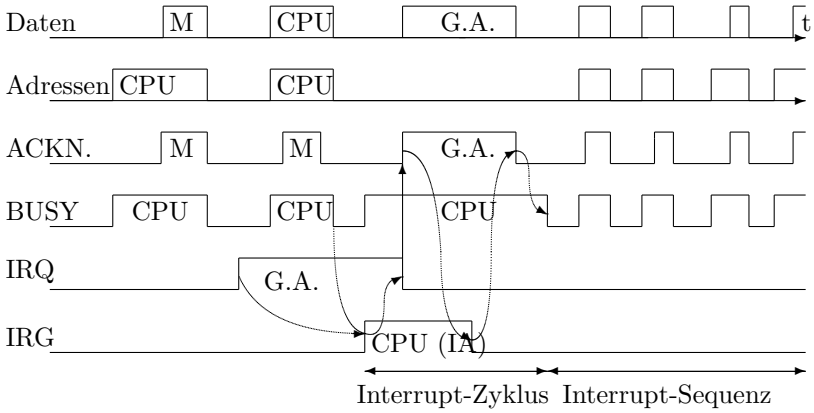


Bild n3p06

Ein Interrupt Request (IRQ) wird erst nach Abarbeitung einer Instruktion von der CPU zugelassen (IRG, Interrupt Grant). Dann schickt der Geräteanschluß auf den Datenleitungen einen Wert, der als Interrupt Vektor von der CPU (bzw dem Umterbrechungswerk) interpretiert wird und dazu dient, aus der Interrupt Vektor Tabelle (IVT) eine Adresse für eine Interrupt Service Routine in den Programm Counter (PC) zu lesen (und auch eine neues Prozessor Status Wort (PSW). Der alte PC (und auch der Prozessor Stutus, PSW) werden für einen Rücksprung auf dem Stack gesichert. Diese 4 Speicherzugriffe stellen die Interrupt Sequenz dar.

d) einige Bussysteme

Bussystem	Übertragungs-Protokoll	Daten-	Adreß-	Steuer-	Gesamt-	Interrupt-Ebenen	Buszyklus (nsec)	Standards
		Leitungen						
S-100 Bus	asynchron	8+8	16(24)	44	100	2	≈ 500	IEEE 696
STD-Bus	synchron (+WAIT)	8	16	22	56	2	≈ 500	div. Hersteller
MULTIBUS	asynchron	8/16	20(24)	23	86+60	8	≈ 100	IEEE 796
VME-BUS	asynchron	16(64)	24(32)	38	96+96	7	≥ 50	IEC 821
Q - BUS	asynchron multiplex	16	18(22)	22	72	4	≈ 500	DEC
EISA-Bus	synchron	32	32			16+8	120	div. Hersteller
MCA-Bus	synchron	32	32			16+8	100	IBM

e) Kombinierte Bussysteme

– Die Bussystem der VAX-11 (DEC)

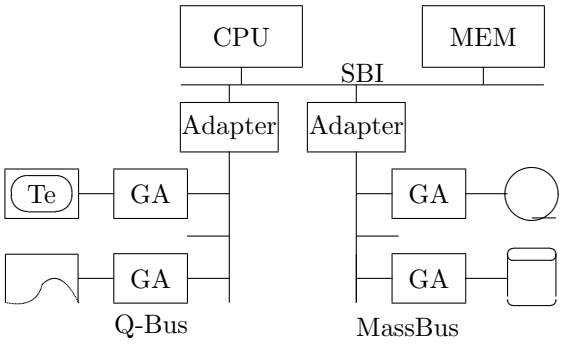


Bild n3p06a (interne Busse der ZE)

– Die Bussysteme eine Prozeaßrechners (Motorola)

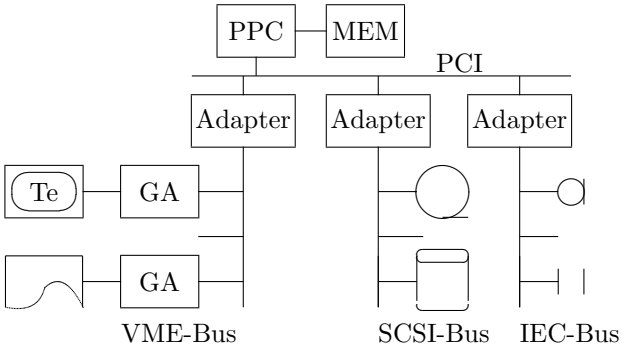


Bild n3p06b (externe Busse für periphere Geräte)

3.2.3 Der Arbeitsspeicher (Main Memory)

a) Grundstruktur

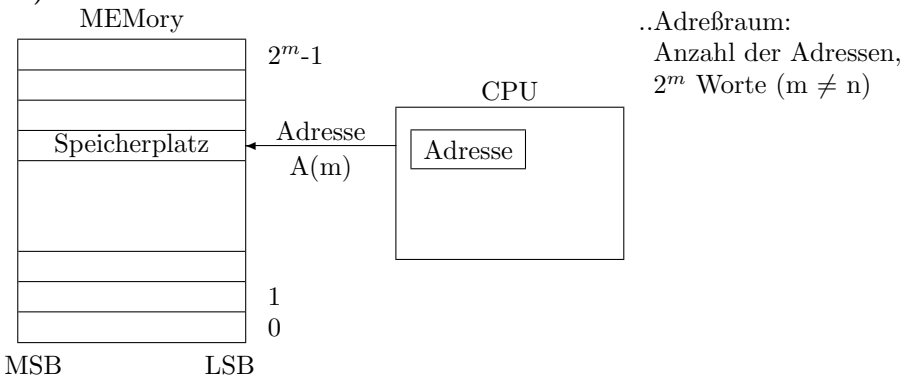


Bild n3p07

b) Bytestruktur

Ein Byte ist ein Teilwort, meist 8 bit breit und wird meist zur Speicherung von Zeichen genutzt.

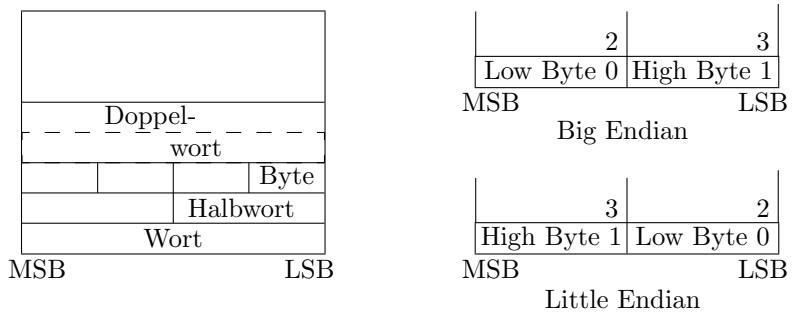


Bild n3p07a

c) Begriffe

- RAM = Random Access Memory (Method): die Adresse kann eine beliebige zufällige Zahl sein (wahlfreier Speicherzugriff)
- SAM = Sequential Access Memory (Method): es gibt eine Startadresse, die folgenden Werte müssen dann sequentiell gelesen werden, z.B. FIFO, Stack, Lochstreife, Magnetband
- ROM = Read Only Memory (Nur-Lese-Speicher): z.B. Lochstreifen
- RWM = "RAM" = Read-Write Memory: Schreib-Lese-Speicher

d) Kenngrößen

- Speichergröße, in Bit, Byte oder Worten (1 K = 2^{10} = 1024, 1 M = 1 Mebi = 2^{20})
- Zugriffszeit (typ: t_a = 20 ... 1000 ns)
- Zykluszeit, bei wiederholtem Zugriff ($t_c \leq t_a$)
- Aufbau und Struktur

e) Adreßräume

Flacher Adreßraum: Die CPU kann logische Adressen (mit n bit Wortbreite) erstellen, welche in ihrer Breite (m bit) ausreichen, um alle Speicherplätze des realen Adreßraums direkt zu adressieren (d.h. $n \geq m$).

Hierarchischer Adreßraum: Die CPU liefert Adressen, welche in einem Speicherverwaltungswerk (Memory Management Unit, MMU) zu realen Adressen umgewandelt werden. Hier ist in der Regel $n < m$. Durch Hinzufügen von weiteren Adreßbits aus Registern der MMU (Memory Management Register, MMR) wird die reale Adresse erstellt.

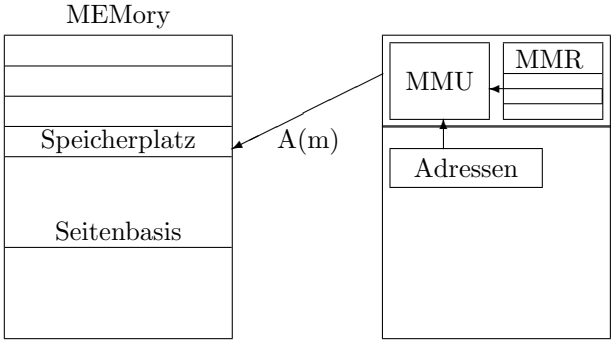


Bild n3p09

Die Bildung der realen Adressen kann nach unterschiedlichsten Regeln erfolgen:
a) durch einfaches Anfügen weiterer Bits aus dem MMR (nur ein MMR möglich)
b) durch Addieren des MMR zur logischen Adresse.
c) durch Auswahl eines MMR durch die obersten Bits der logischen Adresse und anschließender Verknüpfung wie in a) oder b).
In jedem Fall bestimmt das MMR die unterste verfügbare reale Speicheradresse, die Seitenbasis. Die Seitengröße wird durch die von der CPU gelieferte Adresse (oder deren Rest im Falle c)) bestimmt, also höchsten 2^n Speicherplätze.

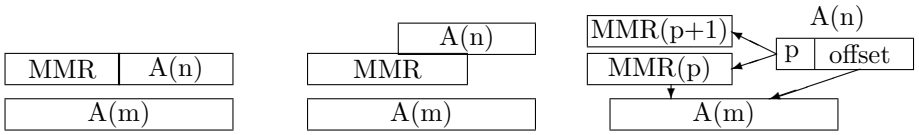


Bild n3p09a

3.2.4 Speicherwerke (Memory Units)

a) Grundstruktur

Ein Speicherwerk (memory unit) enthält eine Speichermatrix, welche die Daten und Instruktionen in Speicherelementen enthält. Diese werden über die (decodierte) Adresse aktiviert. Der Adreßdecoder kann auch integrierter Bestandteil eines Speicherchips sein. Die Basisadresse wird nur benötigt, wenn mehrere Speicherwerke (Speicherkarten) vorhanden sind. Die Auswahl der Karte erfolgt meist über die höchstwertigen Adreßbits. Über die Steuersignale (READ/WRITE) wird die Richtung der Datenübertragung bestimmt (R/W).

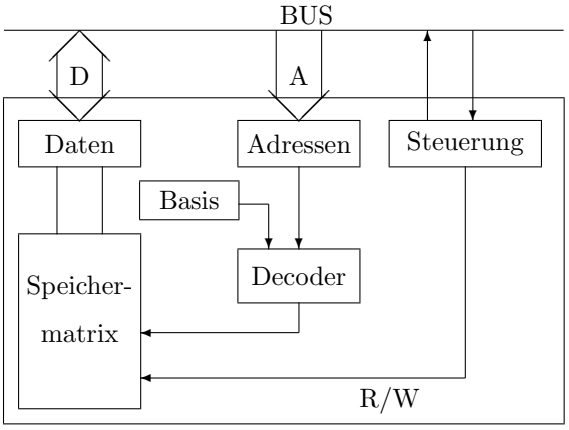


Bild n3p08

b) Adreßdecoder

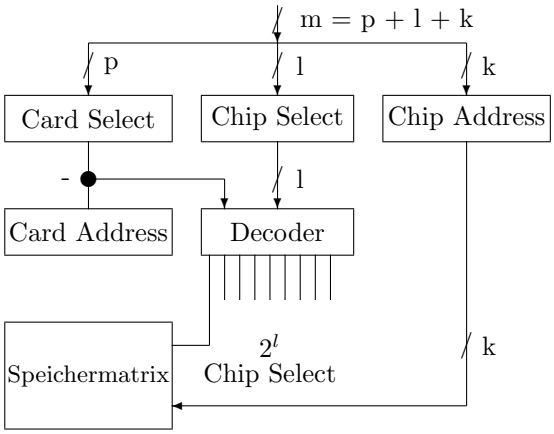


Bild n3p08a

c) Speicherbausteine (Chips)

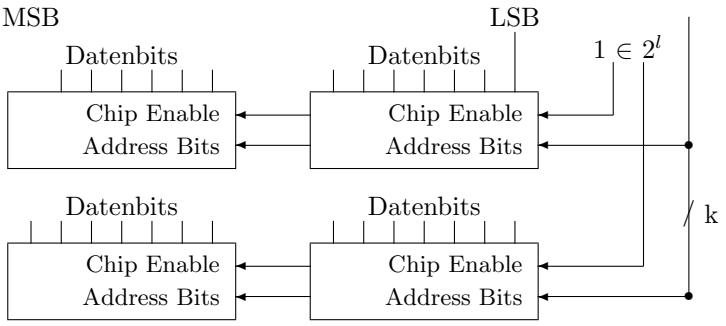


Bild n3p08b

d) Chipstruktur

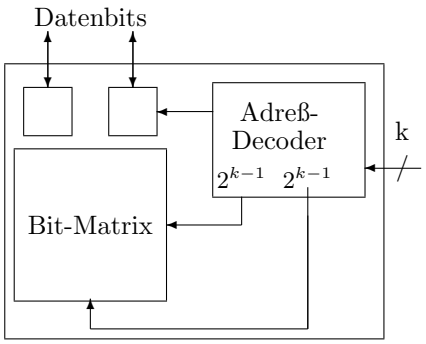


Bild n3p08c

. .Die Bits im Chip sind in der Regel in einer quadratischen Matrix angeordnet; z.B. 1 M(ebi) Bits in 1K Reihen und 1K Spalten. Der Zugang kann unterschiedlich sein; z.B. kann bei eine 1M Bit Chip meist nur auf ein Bit (von 2^{20}) mit 20 Adreßleitungen zugegriffen werden, oder auf 2 Bit, bei 19 Adreßbits.

3.2.5 Speicherelemente (Memory Elements)

a) Grundstruktur

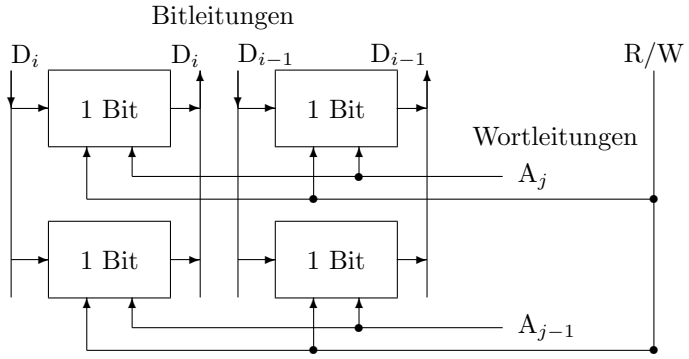


Bild n3p10

b) Ladungsspeicher (dynamisches RAM, DRAM)

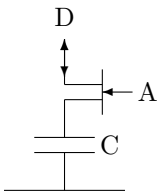


Bild n3p10a

. .Vorteile:
- sehr klein, hohe Speicherdichte: ca 1 Mbit / Chip
- billig: ca 10 DM/Mbit
- schnell: $t_a \approx 10 - 200$ ns
- sparsam: ca 1 W / Chip

Nachteile
- flüchtig (volatil) bei Stromausfall
- destruktives Lesen: $t_c \approx 2\ t_a$
- selbstentladend: $\tau = 1/RC \approx 1$ ms (refresh erforderlich)
- störanfällig: Entladung (Soft-Error) durch ionisierende Strahlung (Röntgen, Höhenstrahlung, o.ä.)

Anwendung: Arbeitsspeicher in sicherheitsunkritischen Anwendungen (PC)

c) Flipflop-Speicher (statisches RAM, SRAM)

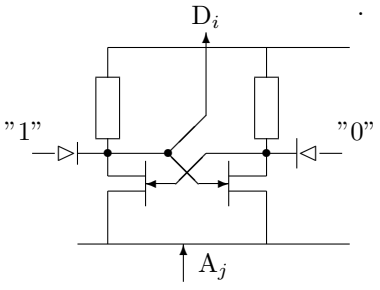


Bild n3p10b

. .Vorteile:
- sehr schnell: $t_a \approx 2 - 20$ ns
- nicht destruktiv beim Lesen: $t_c = t_a$
- nicht selbstentladend
- (klein) ca 8 Kbit / Chip

Nachteile
- flüchtig (volatil)
- konsumptiv (viel Strom verbrauchend, ein Transistor leitet immer) ca 3 W / Chip

Anwendung: Register und Puffer

d) Ringkernspeicher (Ferritkerne)

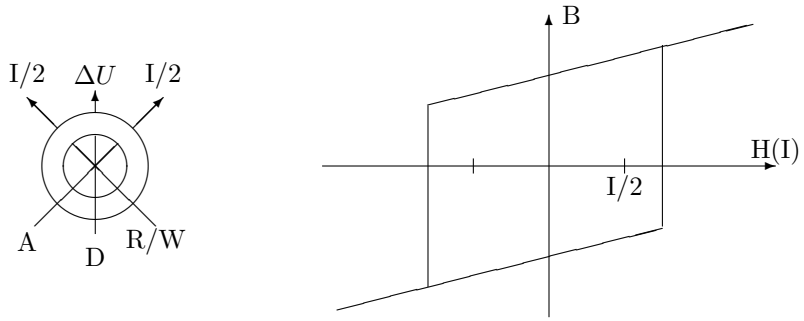


Bild n3p10c

Nachteile:
- groß: ca 100 bit / 1 mm^3
- langsam: t$_a$ ≈ 500 ns
- destruktives Lesen
- komplexe Ansteuerung
- teuer (ca 500 DM/ Kbyte)

. .Vorteile:
- nicht flüchtig (nonvolatil)
- störsicher gegen praktisch alle Umwelteinflüsse (Strahlung, Felder, Temperatur, Schock)

Anwendung: in sicherheitsrelevanten Anwendungen, Raumfahrt, Medizintechnik

e) Festwertspeicher (ROM)

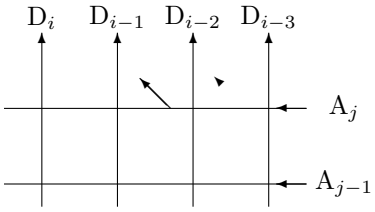


Bild n3p10d

. .Vorteile:
- billig: ab 10 DM/Mbit
- schnell: t$_a$ ≈ 10 – 200 ns
- sehr sparsam: ca 0.01 W / Chip
- nicht flüchtig (nonvolatil)
- störsicher gegen die meisten Umwelteinflüsse (Strahlung, Felder, Schock)

Nachteile
- einmaliges oder aufwendiges Schreiben

Anwendung: Betriebssystem-Komponenten, sicherheitsrelevante Programme

Varianten
- Diodenmatrix
- PROM, Programmable ROM: die Dioden werden vom Anwender (für "0"en) durchgebrannt (WORM = Write Once, Read Multiple)
- EPROM, Erasable PROM, wiederlöschbares PROM: Statt der Dioden werden Transitoren eingesetzt, die durch UV-Licht oder Röntgenstrahlung wieder "geheilt' werden können.
- EEPROM, Electrically Erasable PROM: die Löschung erfolgt elektrisch.

3.3 Geräteanschlüsse

Aufgaben:
- Steuerung von Geräten und Zustandsabfrage
- Datenübertragung von und zu Geräten
- Adressierung von Geräten
- Interfaces: byteweise Übertragung zu und und von Datenendgeräten
- Controller: blockweise Übertragung von und zu Massenspeichern

3.3.1 Grundstruktur

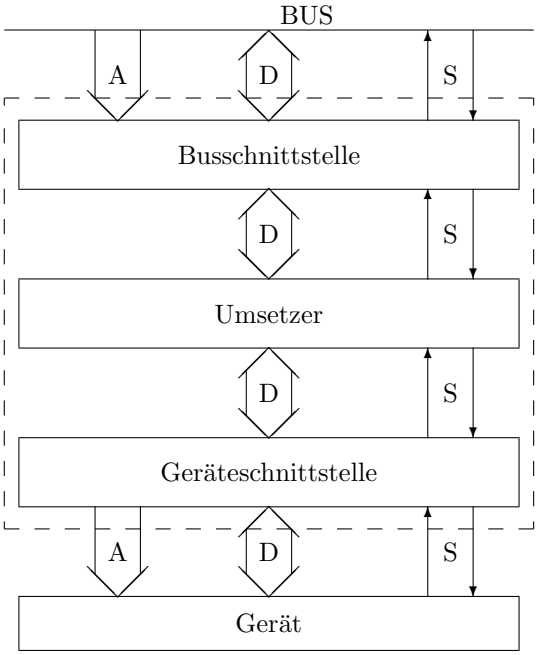


Bild n3p11

- Busschnittstelle (rechnerspezifisch): die Kommunikation erfolgt über Register, die wie Speicherworte angesprochen werden können.
- Umsetzer für Daten und Steuersignale. Adressen für Geräte müssen aus Daten von der CPU gebildet werden.
- Geräteschnittstelle (gerätespezifisch): byte- oder blockorientiert.

3.3.2 Busschnittstelle

a) Grundstruktur
Für alle Geräteanschlüsse einheitlich.

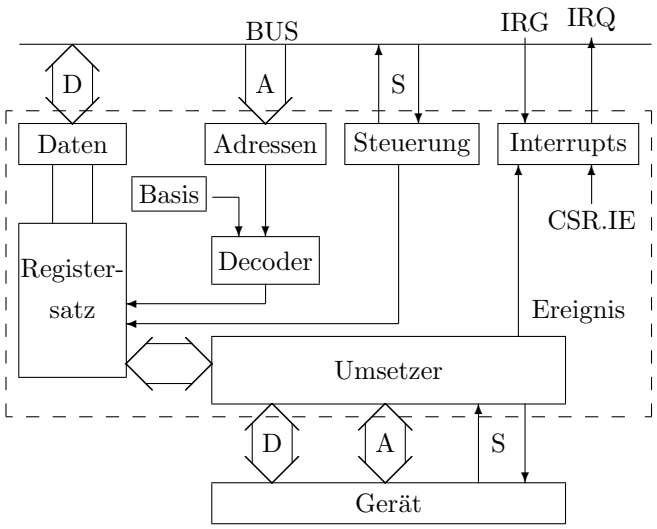


Bild n3p11a

b) Zugriff der CPU auf Register in einem speziellen Adreßraum (IO-page)
Basisadresse für jedes Gerät konfigurierbar,
z.B. COM-1 Port = 3F8h
Registeradressen mit festem Offset
DBR: Data Buffer Register, Datenpuffer
CSR: Control and Status Register, Abfrage und Steuerung von Gerät und Umsetzer.
z.B. beim COM-1 Port (UART 8250/16450):

- 00h | MSB LSB |
  Empfangs-Datenregister (Receive Data Buffer Register,R-DBR) read only
  Sende-Datenregister (Transmit Data Buffer Register,T-DBR) write only
- 01h Interrupt-Aktivierungsregister (Interrupt Enable)

0	0	0	0	SINP	ERBK	TBE	RxRd

  Interrupt-Aktivierung wenn Bitwert = 1
  - SINP = Serial INPut (Zustandsänderung einer Empfangsleitung)
  - ERBK = Error or Break: Parity-, Overflow- oder Framing-Fehler oder BREAK
  - TBE = Transmitter Buffer Empty (Sendepuffer leer)
  - RxRd = Received Data Ready (ein Zeichen im Empfangspuffer)

3.3.3 Übertragung, Umsetzung, Interrupts und DMA

- Geräteschnittstelle: byteorientiert oder blockorientiert
- Busschnittstelle: CPU-Kontrolle oder DMA-Betrieb

3.3.3.1 Übertragung unter CPU-Kontrolle, Polling

- Einzelzeichenübertragung:
  MEM ↔ CPU ↔ DBR ↔ Gerät (Datenübertragung)
  MEM ↔ CPU ↔ CSR ↔ Gerät (Steuerung)
  unter Programmkontrolle z.B.

```
TSTB CSR        ; Abfrage ob READY
BPL  .-1        ; Rücksprung falls nicht
MOV  DBR, MEM   ; Daten holen und speichern
```

- Blockpufferung: Die zu übertragenden Zeichen werden in einem FIFO zwischengespeichert und erst bei Bedarf ohne CPU Kontrolle übertragen.
  (Das FIFO kann auch im Gerät angesiedelt sein.)
  Datenübertragung: MEM ↔ CPU ↔ DBR ↔ FIFO ↔ Gerät

3.3.3.2 Interrupts

Mehrere Ebenen

a) Interrupt-Erzeugung

- Quelle für (fast) jede Interrupt-Anforderung ist ein Geräte-Anschluß.
  Seine Interrupt-Logik hat folgende Aufgaben:
  - äußere Ereignisse und Meldungen erfassen und eventuell speichern,
  - Interrupt-Anforderung (Interrupt Request, IRQ) an die CPU senden und auf deren Bestätigung (Interrupt Grant, IRG) warten,
  - weitere Daten über den geforderten Interrupt senden, (Priorität des Interrupts HWP, Zeiger (IV) auf einen Eintrag in der IVT (vektorisierte Interrupts).)

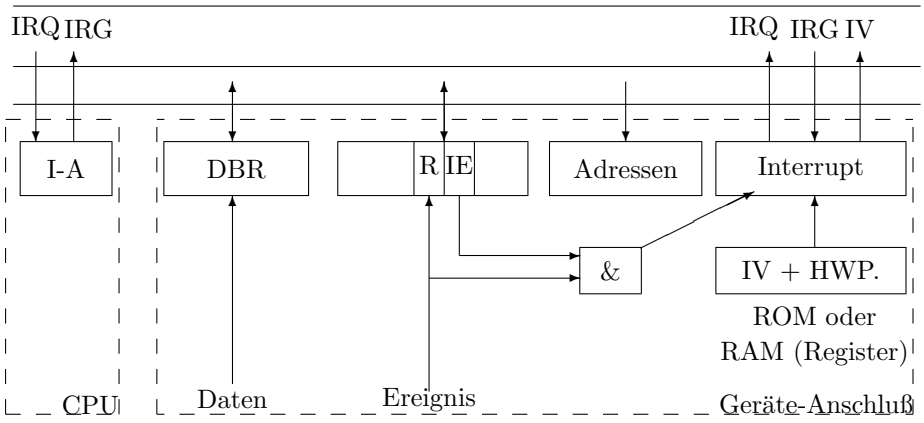


Bild b3p15

- Filterung der Interrupt-Anforderungen in der CPU im Interrupt-Verarbeitungswerk (Interrupt Arbitrator, I-A)

a) Maskierung, Verriegelung
Maskenregister wird unter Programmkontrolle geladen, dessen Bits einzelnen Geräte-Anschlüssen zugeordnet sind und deren Interruptanforderungen weitergeben oder blockieren.

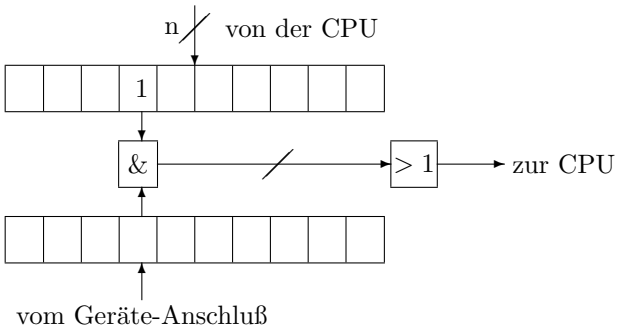


Bild b3p16

b) Interrupt-Gewichtung
Die von den Geräte-Anschlüssen ankommenden IRQs werden über einen Decoder in einen Zahlenwert umgewandelt, die Hardware-Priorität (HWP), der mit einem arithmetischen Vergleicher mit dem in einem Register stehenden Zahlenwert, der Software-Priorität (SWP), verglichen wird. An die CPU wird die Interrupt-Anforderung nur dann weitergegeben, wenn $HWP > SWP$ ist.

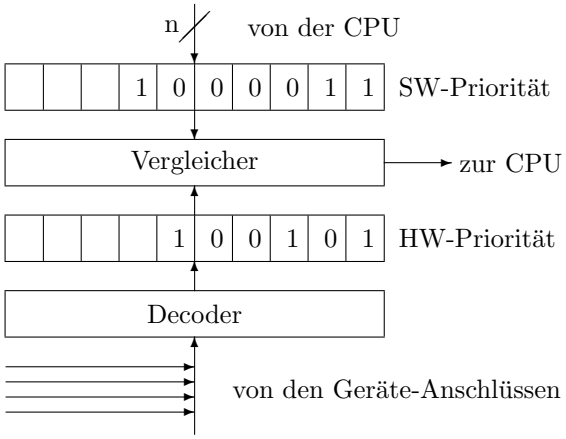


Bild b3p17

c) Darstellung der Prioritätensteuerung durch P-t-Diagramme
Hier wird in Abhängigkeit von der Zeit t die Software-Priorität P (der CPU) als Kurvenzug aufgetragen, der nur (wenige) diskrete Werte annehmen kann. Als Ereignispfeile (E_i) werden die auftretenden Interrupts mit ihren Hardware-Prioritäten eingetragen. Falls ein solcher Pfeil eine höhere Priorität anzeigt, wird ein Interrupt ausgelöst, der in einem parallel verlaufenden A-t-Diagramm dargestellt werden kann. Falls die Hardware-Priorität niedriger ist und der Interrupt nicht sofort durchgeführt wird, kann die dabei auftretende Wartezeit (Reaktionszeit) dargestellt und bestimmt werden.

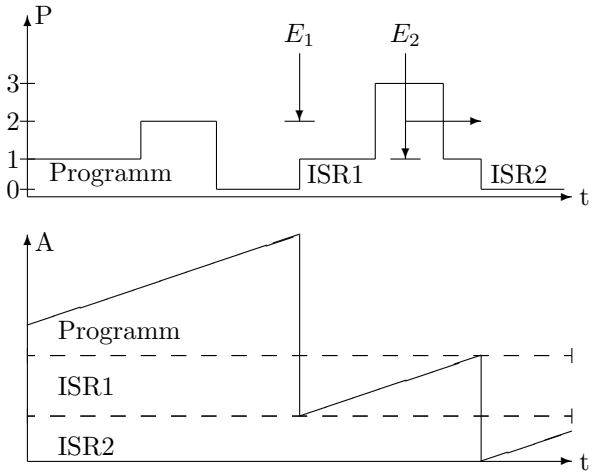


Bild b3p18

- Bearbeitung des Interrupts durch das Mikroprogramm der CPU

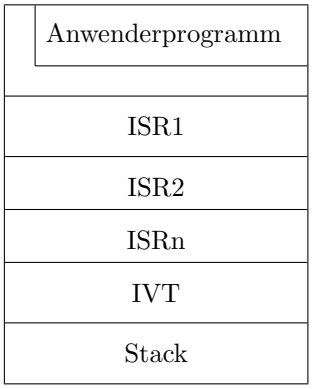


Bild b3p09

- Ein Anwenderprogramm P läuft (in einer Schleife).
- Ein Ereignis löst eine Interruptanforderung aus.
- Die CPU bestätigt die Anforderung am Ende einer Instruktion.
- Der aktuelle PC (und eventuell auch andere Werte, wie das Prozessor-Status-Wort PSW) wird auf dem Stack gesichert.
- Aus der Interrupt Vektor Tabelle IVT wird ein neuer PC, die Einsprungadresse der ISR geholt.
- Nach Abarbeitung der ISR wird beim Rücksprung der alte PC (und die übrigen Werte) vom Stack zurückgeholt und das unterbrochene Programm fortgesetzt.

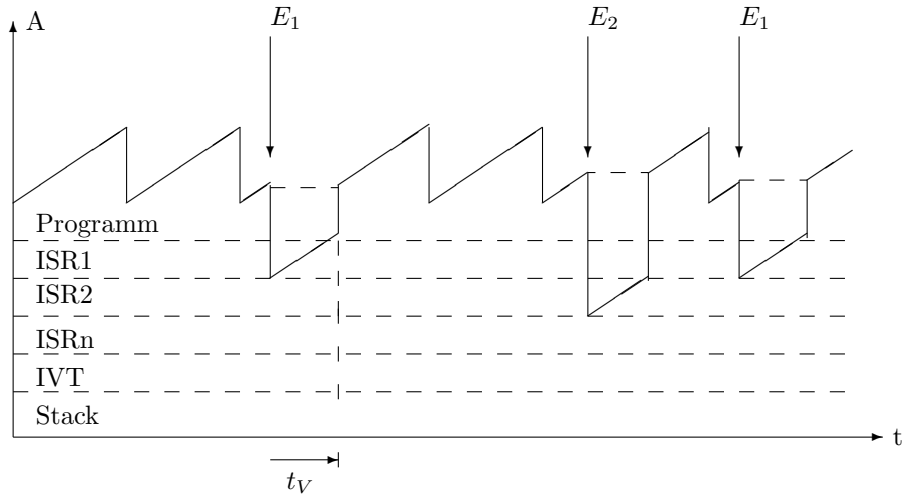


Bild b3p10

Vergleich der Betriebsarten:

	Polling	Interrupt
Vorteile	einfache Hard- und Software	CPU-Auslastung $< 100\%$
Nachteile	CPU-Auslastung $= 100\%$	komplexe Hard- und Software

3.3.3.3 DMA-Betrieb

(Speicherdirektzugriff) Direct Memory Access

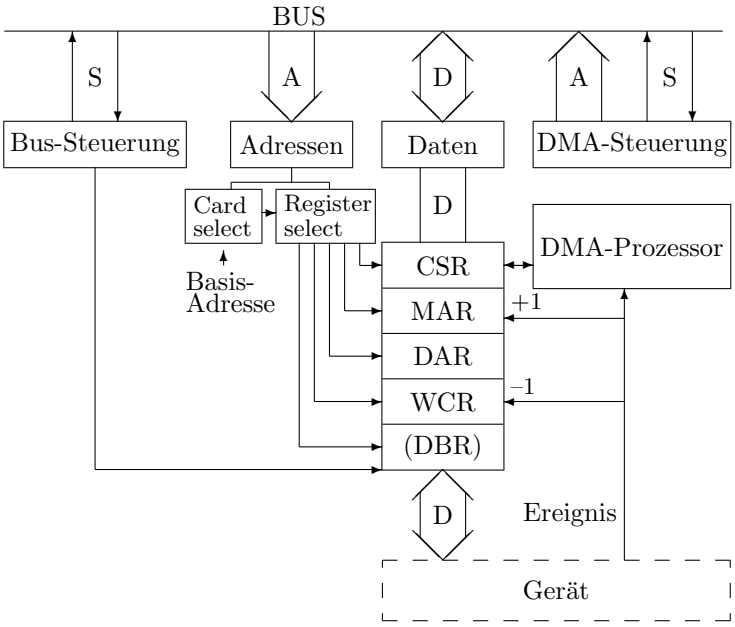


Bild n3p12

- MAR: Memory Address Register
  Adresse im Arbeitsspeicher, von/zu der das nächste Wort übertragen wird.
- DAR: Device Address Register
  Adresse des Gerätes bzw. Gerätebereichs (Plattenfläche, Sektor, Spur)
- WCR: Word Count Register: Zähler für die zu übertragenden Daten.

DMA-Vorbereitung unter Programm-Kontrolle:

```
MOV X, MAR      ; MAR laden
MOV Y, DAR      ; DAR laden
MOV Z, WCR      ; WCR laden
MOV F, CSR      ; Funktion laden
INC  CSR        ; Go-Bit setzen
```

Übertragung ohne CPU-Mitwirkung (internes Programm des DMA-Controllers in ROM):

- DMA-Request, sobald Daten übetragen werden (Ereignis)
- DMA-Grant von der CPU nach dem nächsten Buszugriff $\overline{BUSY}$
- Buszugriff durch Controller "BUSY" (DMA-Zyklus)
- Controller bedient A- und S-Leitungen
- Daten kommen vom Memory (bei READ)
- Controller gibt den Bus wieder frei

- MAR + 1
- WCR - 1

DMA-Zugriffsmethoden:
- Cycle-Stealing: CPU und Controller teilen sich den Bus 1:1
- Burst-Mode: Controller macht n Zugriffe nacheinander (1:n)
- Prozessor-Halt: Controller macht beliebig viele Zugriffe.

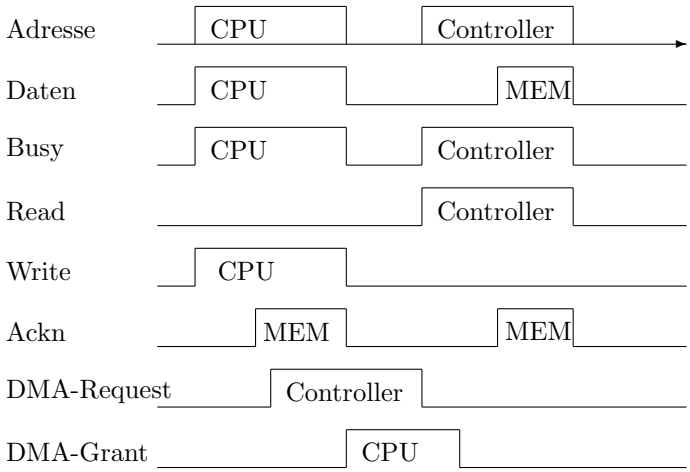


Bild n3p12a

DMA-Abschluß unter Programm-Kontrolle
- Wenn WCR = 0, wird das READY-Bit im CSR setzen
- Falls ein Fehler auftritt, wird das ERROR-Bit im CSR gesetzt
- Falls Interrupt Enable gesetzt, erfolgt Aufruf einer ISR
  Andernfalls muß die CPU immer wieder mal nachfragen (pollen)

3.3.4 Geräteschnittstellen

zur Ankopplung von peripheren Geräten, für jeden Gerätetyp unterschiedlich:

a) Parallel-Schnittstelle (für Drucker von Centronics)
Die Daten werden aus dem DBR (über Leitungsverstärker) parallel ausgegeben. Steuersignale werden aus dem CSR gebildet.

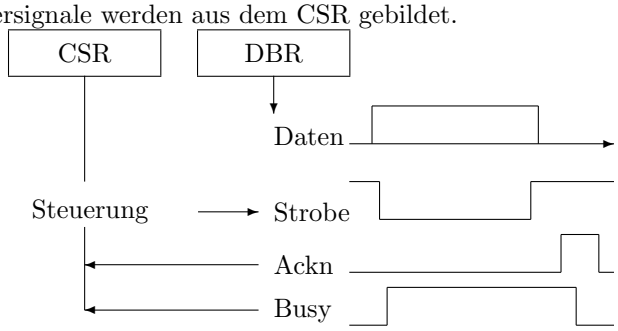


Bild n3p13

b) Terminalschnittstelle seriell, asynchron (V.24, RS 232-C)
Der Zugriff auf diese Schnittstelle erfolgt gewöhnlich über je ein CSR und ein DBR in Sende- und in Empfangsrichtung.

Zur Umsetzung wird ein Parallel-Seriell-Wandler eingesetzt, der die Datenbits eines Zeichens in eine Folge von (gleich langen) Signalen umsetzt.

Die V.24-Schnittstelle ist also sowohl für synchrone als auch für asynchrone Datenübertragung vorgesehen. Also solche, bei der die Zeichen in beliebigen Abständen oder unterbrechungslos (mit Pausenzeichen) übertragen werden.

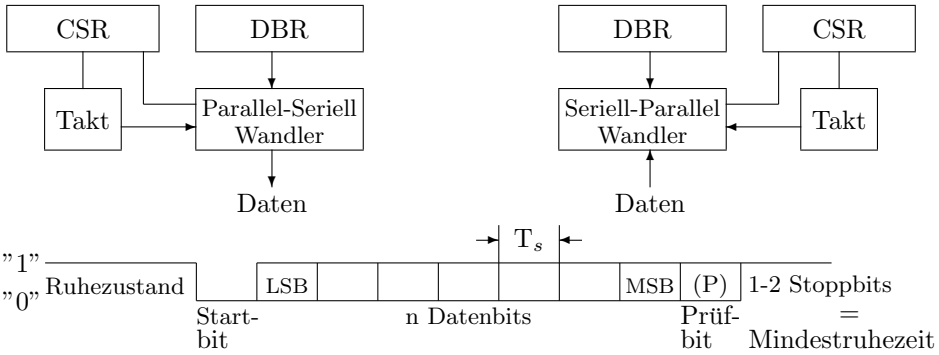


Bild n3p14

V.4 (DIN 66 022) legt die Zeichendarstellung auf den Datenleitungen für die asynchrone Übertragung, das **Start-Stop-Verfahren** fest:
- Der **Ruhezustand** der Leitung ist '1'.
- Jedes Bit wird in einem **Taktschritt** fester Länge T_s übertragen.
- Der Beginn eines Zeichens wird durch ein **Startbit** mit dem Zustand '0' markiert.

- Die **Datenbits** werden mit dem niederwertigsten Bit (**LSB** = Least Significant Bit) zuerst und dem höchstwertigen Bit (**MSB** = Most Significant Bit) zuletzt übertragen. In der Regel werden 7 Datenbits des ASCII-Zeichensatzes übertragen.
- Den Datenbits kann ein **Prüfbit** gerader oder ungerader **Parität** folgen (even/ odd **parity**). Das Prüfbit wird so gesetzt, daß hiermit die Summe aller '1'-en gerade bzw. ungerade, d.h. die gesamte Quersumme modulo 2 eine 0 oder eine 1 ergibt.
- Das Ende eines Zeichens wird signalisiert durch 1, 1.5 oder 2 **Stoppbits**, bei denen die Leitung im Ruhezustand '1' bleibt, bevor das nächste Zeichen übertragen wird. Die Stoppbits beinhalten keine Informationsübertragung, und die Angabe ihrer Anzahl bedeutet nur die **Mindestruhezeit** in Einheiten von Taktschritten T_s.
- Das nächste Zeichen kann unmittelbar darauf folgen oder zu einem beliebigen späteren Zeitpunkt.

Freie Parameter, die am Sender und am Empfänger übereinstimmend eingestellt sein müssen:
- **Baudrate** oder **Schrittgeschwindigkeit** $f_s = 1/T_s$, für die folgende Werte gewählt werden können: 75, 110, 150, 200, 300, 600, 1200, 2400, 4800, 9600, und 19200 Baud. Wegen der binären Codierung ist hier die Schrittgeschwindigkeit (in Baud) mit der **Bitübertragungsrate** (in **bps = bits per second**) identisch.
- Zahl der Datenbits (n = 5,6,7,8). Für die Übertragung von beliebigen, uncodierten Dualwerten in Form von Bytes ist n = 8 zu wählen.
- Anwendung eines Prüfbits (p = 0 oder 1). Falls ein Prüfbit verwendet werden soll (p=1), muß noch die **Parität** auf **'gerade'** oder **'ungerade'** eingestellt werden. In manchen Fällen werden auch Prüfbits mit festen Werten verwendet:
'0' = **space parity** oder '1' = **mark parity**, die vom Sender erzeugt, vom Empfänger nicht ausgewertet, aber mitgezählt werden (vgl. 2.6.6).
- Anzahl der **Stoppbits** (z = 1, 1.5, 2).

Die **Übertragungszeit** für ein Zeichen ergibt sich dann zu

$$T_z = (1 + n + p + z) * T_s$$

und die **Zeichenübertragungsrate** zu

$$f_z = 1/T_z \qquad \text{(in } \mathbf{cps} \text{ = characters per second)}$$

Typische Einstellungen sind:
- Für die amerikanischen Fernschreiber (**Teletype**):
n = 7 (ASCII-Zeichen), p = 1, odd parity, z = 2, f_s = 110 Baud;
das ergibt eine Zeichen(druck)geschwindigkeit $f_z = 110/(1+7+1+2) = 10$ cps
- Für Binärdaten bei hoher Geschwindigkeit:
n = 8 (Byte), p = 0, z = 1, f_s = 9600 Baud;
und einer resultierenden Übertragungsrate $f_z = 9600/(1+8+1) = 960$ Byte/sec.

Die V.24 Schnittstelle ist ursprünglich für die Kopplung von 2 gleichwertigen Datenstationen über eine Telefonleitung konzipiert worden (a). Dazu werden auf beiden Seiten Modems eingesetzt, welche die Aufgabe haben, die digitalen Signale von Terminal und Rechner in Wechselspannungen umzusetzen, wie sie auch bei der Sprachübertragung auf Telefonleitungen übermittelt werden. Wenn das Terminal nahe beim Rechner steht (b), können die Modems entfallen; dann muß aber ein Nullmodemkabel

eingesetzt werden, welches eine Sende- auf die zugehörige Empfangsleitung umsetzt, und umgekehrt (c).

Dazu sind an jedem Stecker jeweils folgende Verbindungen herzustellen:
- RTS, CTS (Stifte 4 und 5) und
- DSR, DTR, DCD (Stifte 6 und 20 und 8).

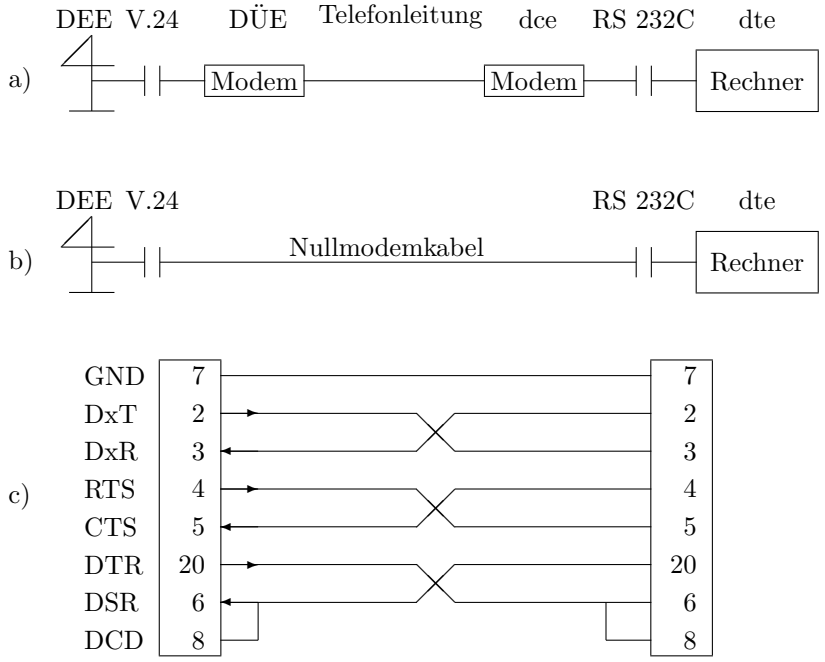


Bild n3p14a

V.24 (DIN 66 020, 66 021) legt die physische und die logische **Schnittstelle** fest, d.h. die Stiftbelegung eines genormten, 25-poligen Anschlußsteckers (nach ISO 2110) und die Bedeutung der auf den Leitungen übertragenen Signale. Sie beziehen sich immer auf die Verbindung zwischen der **Datenendeinrichtung** (**DEE**, engl.: **data terminal equipment**, **dte**), welche sowohl ein Terminal als auch ein Rechner sein kann, und dem **Modem** (Modulator/Demodulator), der auch als **Datenübertragungseinrichtung** (**DÜE**, engl.: **data communication equipment** oder **data circuit terminating equipment**, **dce**) bezeichnet wird.

V.28 (DIN 66 259; T1) legt die elektrischen Werte für die Darstellung der Daten und der Steuersignale fest:
- Die Datenwerte werden durch **Signalpegel** dargestellt, die über einen **Taktschritt**, eine **Bitdauer** (engl.: **bit time**) konstant bleiben. Dieses Verfahren wird als **NRZ (Non Return to Zero)** bezeichnet. Für die Pegel wird ein **Doppelstromverfahren** benutzt, also Pegel entgegengesetzter Polarität, die bestimmte Grenzwerte einhalten müssen (s. Tabelle 2.8).
- Die Steuersignale können beliebige Längen haben, die Pegel für ihre EIN- bzw. AUS-Zustände sind in Tabelle 2.8 angegeben.

Pegel der V.24-Schnittstelle (V.28)

Schnittstellenleitungen nach V.24

Kurzzeichen			Stift	Beschreibung		Richtung
CCITT V.24	EIA RS232	DIN 66020	ISO 2110	deutsch	englisch	Terminal Modem
101	P-GND	E 1	1	Schutzerde	protectiv ground	○——○
102	S-GND	E 2	7	Signalerde	signal ground	○——○
103	TD	D 1	2	Sendedaten	Transmitted Data	○——►
104	RD	D 2	3	Empfangsdaten	Received Data	◄——○
105	RTS	S 2	4	Sendeteil einschalten	Request to Send	○——►
106	CTS	M 2	5	Sendebereitschaft	Clear to Send	◄——○
107	DSR	M 1	6	Betriebsbereitschaft	Data Set Ready	◄——○
108.1		S 1.1	20	Übertragung. anschalten		○——►
108.2	DTR	S 1.2	20	Terminal betriebsbereit	Data Terminal Ready	○——►
109	DCD	M 5	8	Empfangssignalpegel	Carrier Detected	◄——○
110	SQ	M 6	21	Empfangsgüte	Signal Quality	◄——○
125	RI	M 3	22	Ankommender Ruf	Ring Indicator	◄——○
111	DRS	S 4	23	Übertragungsgeschw.	Rate selctor (dte)	○——►
112		M 4	23	dto	Rate selctor (dce)	◄——○
126	STF	S 5	11	Sendefrequenz(200 Baud)	Select frequency	○——►
113	TC	T 1	24	Sendeschrittakt	Transmit Clock (dte)	○——►
114	TC	T 2	15	Sendeschrittakt	Transmit Clock (dce)	◄——○
115	RC	T 3	17	Empfangschrittakt	Receive Clock	◄——○
118	TD.2	HD 1	14	Hilfskanal Sendedaten	Second. Trans. Data	○——►
119	RD.2	HD 2	16	Hilfskanal Empfangsdaten	Second. Receiv. Data	◄——○

Pegel	Daten	Signale
-15 V bis -3 V	'1' (mark)	AUS
+15 V bis +3 V	'0' (space)	EIN
-3 V bis +3 V	(no carrier)	undefiniert

Varianten für die (elektrische) Darstellung der Bits gegenüber der V.28 Norm sind:
– die historische **20-mA-Stromschleife** (current **loop** vgl. DIN 66 258; T1), ein **Einfachstromverfahren** mit folgenden Werten:

'0' = 0 mA
'1' = 20 mA

– **V.10** bzw. **X.26** (DIN 66 259; T2), eine unsymmetrische **Doppelstromschnittstelle**, die auch für Koaxialleiter gedacht ist, und **Übertragungsraten** bis 100 kBaud ermöglichen soll. Sie arbeitet mit **Signalpegeln**, die mit integrierten Schaltungen leicht realisiert werden können:

'0' = +0.3V bis +5.0 V
'1' = –0.3V bis –5.0 V

– **V.11** bzw. **X.27** (DIN 66 259; T3), eine symmetrische **Doppelstromschnittstelle** bis 10 MBaud mit denselben Signalpegeln wie V.10, jedoch für verdrillte Doppeladern.
Neue **Schnittstellennormen**, welche die **RS-232-C** Schnittstelle ablösen, sind:
– **RS-449** bis 10 MBaud und elektrischen Werten von V.11 (RS-422A) oder V.10

(RS-423A). Neue Signalleitungen und ein 37-poliger Stecker.
– In den neuen Datennetzen (ISDN) sollen **X.21** und **X.24** die V.24-Norm ablösen.

Rechnerbausteine:
UART, Universal Asynchronous receiver and Transmitter oder
PUSART, Programmable Universal Synchroneous and Asynchroneous Receiver and Transmitter

Beispiel: UART 8250 (COM-1 Port)

- 00h

MSB	LSB

  Empfangs-Datenregister (Receive Data Buffer Register,R-DBR) read only
  Sende-Datenregister (Transmit Data Buffer Register,T-DBR) write only
- 01h Interrupt-Aktivierungsregister (Interrupt Enable)

0	0	0	0	SINP	ERBK	TBE	RxRd

  Interrupt-Aktivierung wenn Bitwert = 1
  - SINP = Serial INPut (Zustandsänderung einer Empfangsleitung)
  - ERBK = ERror or BReak: Parity-, Overflow-, Framing-Fehler oder BREAK
  - TBE = Transmitter Buffer Empty (Sendepuffer leer)
  - RxRd = Received Data Ready (ein Zeichen im Empfangspuffer)
- 02h Interrupt-Identifizierungsregister (Interrupt Enable)

0	0	0	0	0	ID 1	ID 2	PND

  Interrupt-Quelle; Löschen durch
  - ID1 =
  - ID2 =
  - PND =
- 03h Datenformat (Leitungssteuerung)

DLAB	BRK	PAR2	PAR1	PAR0	STOP	DAB 1	DAB0

  - DLAB
  - BRK
  - PAR2 PAR1 PAR0: Parität
    000 = keine 001 = ungerade 011 = gerade 101 = mark 111 = space
  - STOP
  - DAB1 DAB0: Anzahl der Datenbits: 00 = 5 Datenbits, 01 = 6 Datenbits, 10 = 7 Datenbits, 11 = 8 Datenbits
- 04h Modemsteuerung (RS 232 Ausgabe)

0	0	0	LOOP	OUT2	OUT1	RTS	DTR

  Modemsteuersignale aktiv = ”1”
  - LOOP (Prüfschleife)
  - OUT2
  - OUT1
  - RTS (Request to Send)
  - DTR (Data Terminal Ready)
- 05h Serialisierungsstatus (Leitungsstatus)

0	TXE	TBE	BRK	FRME	PARE	OVFL	RxRd

  - TXE (Transmitter Empty)

- TBE (Transmitter Buffer Empty)
- BRK (Break Signal)
- FRME (Framing Error) Rahmenfehler
- PARE (Parity Error) Prüfsummenfehler
- OVFL (Overflow) Überlauf (im Empfangspuffer)
- RxRd = Received Data Ready (ein Zeichen im Empfangspuffer)

- 06h Modemstatus (RS 232 Eingabe)

DCD	RI	DSR	CTS	DDCD	DRI	DDSR	DCTS

  - DCD (Data Carrier Detected)
  - RI (Ring Indicator)
  - DSR (Data Set Ready)
  - CTS (Clear To Send)
  - DDCD (Delta Data Carrier Detected) "1"= Veränderung seit letztem Lesen
  - DRI (Delta Ring Indicator)
  - DDSR (Delta Data Set Ready)
  - DCTS (Delta Clear To Send)

- 07h Scratch-Pad (Zwischenspeicher) | MSB LSB |

c) Massenspeicheranschlüsse

SCSI (Small Computer System Interface), DMA-Controller

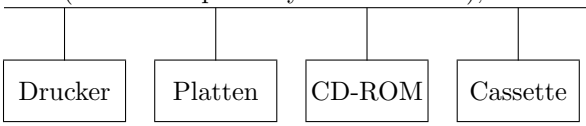


Bild n3p14b

- Datenleitungen: 8, 16, 32
- Adreßleitungen: 3
- Steuerleitungen: 8
- Taktrate: 1 bis 25 MHz

IEC-Bus (IEEE 488) für diverse Geräte, auch Prozeßperipherie

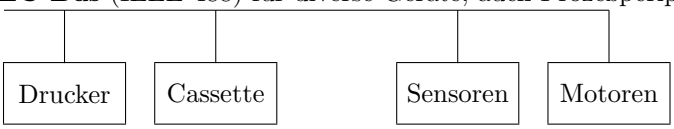


Bild n3p14c

- Adreß/Datenleitungen: 8
- Steuerleitungen: 8
- Taktrate: 1 MHz

3.4 Standardperipherie

- Ein/Ausgabegeräte für den Benutzer:
  - Terminal, Datenstation, Datensichtgerät (Bildschirm und Tastatur)
  - Drucker
  - Plotter
  - Maus
  - Scanner
- Massenspeicher
  - Magnetplatten, Floppydisk, CD-ROM
  - Magnetband, Cassetten
- Kommunikation
  - Modem
  - Netzkarten
- Prozeßperipherie
  - Sensoren
  - Motoren
  - AD-Wandler
  - DA-Wandler

3.4.1 Terminal

Das Terminal besteht aus 2 Einheiten, dem Bildschirm und der Tastatur. Für jede wird ein eigener Satz von Registern (DBR, CSR) zum Senden (transmit, T) und Empfangen (receive, R) benötigt.

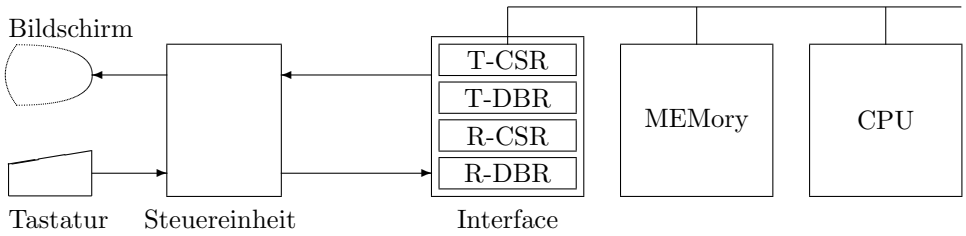


Bild n3p15

a) Der Bildschirm (Monitor)

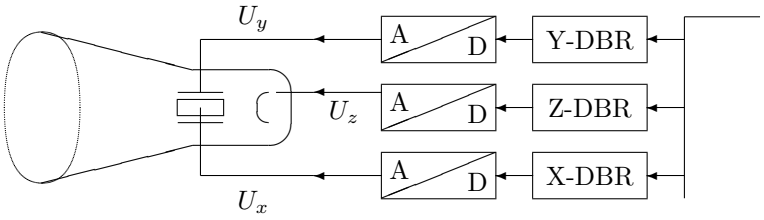


Bild n3p16

Als Bildschirm wird üblicherweise eine Kathodenstrahlröhre (CRT, Cathode Ray Tube) verwendet, wie beim Fernseher. Die Ablenkung in X- und in Y-Richtung erfolgt durch Analogspannungen (U_x und U_y), die aus Digitalwerten durch A/D-Wandler erzeugt werden. Die Helligkeit wird durch eine weitere Spannung (U_z) gesteuert. Bei Farbbildröhren werden 3 solcher Spannungen benötigt für die Grundfarben Rot, Grüen, Blau, RGB). In vielen CRT-Geräten wird die X- und Y-Ablenkung mit Schaltungen der Fernsehtechnik realisiert, die anstelle von Analogsignalen nur noch Synchronisationsimpulse (horizontal und vertikal) benötigen.

- Vektorbildschirm (selten und teuer): Jeder Bildpunkt P wird durch geeignete Ablenkspannungen U_x(P) und U_y(P) im RAM-Verfahren erstellt.
- Rasterbildschirm: periodische Ablenkung durch U_x(t) und U_y(t), Helligkeit unfd Farbe werden zum "richtigen" Zeitpunkt durch die Steuerspannung(en) U_z(RGB) erzeugt (SAM).

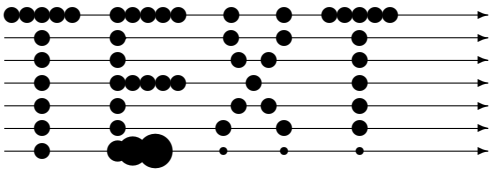


Bild n3p16a

Kennwerte von (Raster-)Bildschirmen

- horizontale Auflösung: Pixel/Zeile ≈ 1000
- vertikale Auflösung: Zeilen/Bild ≈ 1000
- zeitliche Auflösung: Bilder/Sekunde ≈ 100 (Bildfrequenz)

Zeilenfrequenz: ≈ 100 kHz Übertragungsrate: $\approx 100 \cdot 10^6$ Hz $= 100$ MHz

Ablenkgeschwindigkeit (in x-Richtung): $v = \frac{\Delta s}{\Delta t} = \frac{0.5m}{100 \cdot 10^3 s^{-1}} = 50 km/s$

Röntgenstrahlung durch Abbremsen der Elektronen beim Aufprall auf den Bildschirm ($U_b = 35$ kV), Elektronenaufprallgeschwindigkeit $v_e = 111$ km/s (ca. 1/3 Lichtgeschwindigkeit).

Nachleuchten der Bildschirmpunkte:

- lange Nachleuchtdauer → Schmieren
- kurze Nachleuchtdauer → Flimmern

b) Die Tastatur Das Drücken einer Taste adressiert ein Wort in einem ROM. Sondertasten (Alt, Shift, Ctrl/Strg) oder deren wählen verschiedene Zeichensätze (ROMs) aus

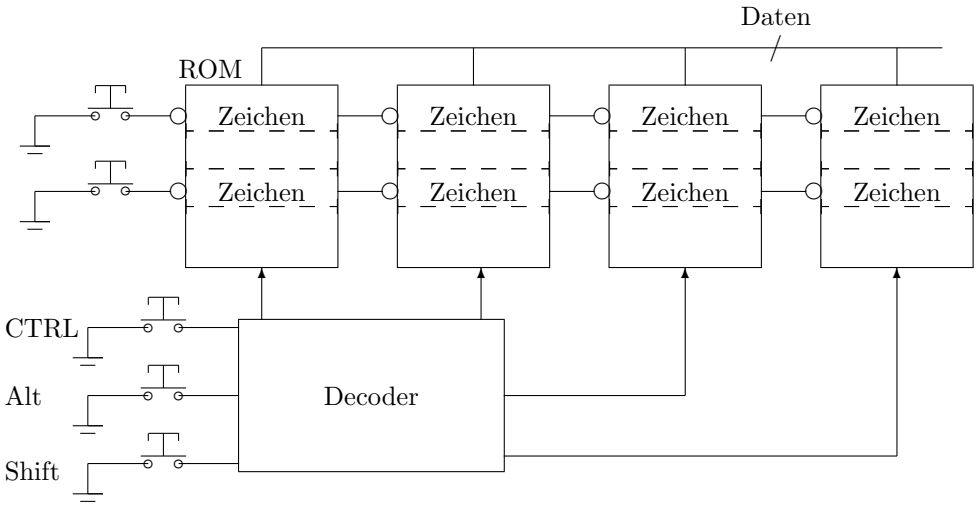


Bild n3p17

3.4.2 Drucker

- Typendrucker (Fernschreiber, Trommel-, Banddrucker)
- Matrixdrucker Pixel → Zeichen → Zeile → Blatt (-Drucker)

Schwärzung, Farbe

- Farbband (anschlagende (impact) Drucker)
- Tinte (nichtanschlagende (non-impact) Drucker)
- Toner (Laserdrucker)
- Wärme (Thermodrucker)

Druckertyp	Druckeinheit	Druckfolge	Druckgeschwindigkeit
Trommeldrucker Ketten-, Banddrucker	Typen	zeichen- parallel	200 - 2400 Zeile/min 3 - 5000 Zeichen/sec
Typenhebel-, -rad-, Typenkorb-, Kugelkopfdrucker		zeichen- seriell	10 - 30 Zeichen/sec
Nadeldrucker Tintenstrahl- Thermodrucker	Punkte	spalten- seriell	30 - 300 Zeichen/sec
Elektrostatische, Laserdrucker		zeilen- parallel	10 - 60 Seiten/min 1 - 200 Seiten/min

3.4.3 Plotter

- Transport von Papier oder Stift in X- und Y-Richtung
- Heben und Senken des Stifts in Z-Richtung
- Auswahl von farben (4. Dimension)

Die Basismaschine ist ein Inkrementalplotter, bei dem ein Zeichenstift (Z) abgehoben und gesenkt werden kann (Z-Richtung). Die Bewegung in X- und Y-Richtung erfolgt Schrittmotoren, die 2-phasig angesteuert werden müssen, und so ein Vorwärts- und Rückwärtsbewegung ermöglichen. Zur Ansteuerung müssen Impulse in einer der beiden Drehrichtungen erzeugt werden:

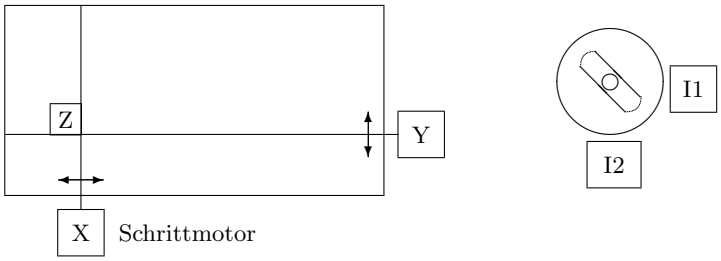


Bild n3p27

Die Impuls erzeugen über Elektromagnete Kraftwir-. kungen auf den (magnetischen) Rotor. In der Praxis werden hier meist 7 Paare verwendet, und die Ansteuerung wird von spezielle Chips übernommen.

. I2	I1	Bewegung
0	0	Ruhe
0	1	+1
1	0	–1
1	1	Ruhe

Die Benutzerschnittstelle wird durch ein DBR realisiert, in welches geeignet Bitkombinationen zu schreiben sind, die über eine Parallelschnittstelle an den Plotter übertragen werden. Zum Erzeugen der Impulse kann ein zusätzliches Startbit (Go) notwendig sein.

Z	+Y	-Y	+X	-X	Go

Bild n3p28

Mit der dargestellten Anordnung können von jedem Punkt aus 8 verschiedene Linienelemente benutzt werden. Aus diesen setzt sich dann jede Linie und Kurve zusammen.

Moderne Plotter enthalten in der Regel die notwendige Logik in Form von Firmware, die als Interpreter mehr oder minder abstrakte Plottbefehle ausführen kann (z.B. HPGL). Die Übertragung zum Plotter erfolgt hier meist über serielle Schnittstellen als ASCII-Text.

3.4.4 Maus

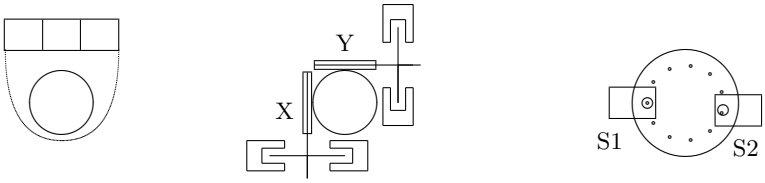


Bild n3p24

Die Bewgung der Maus auf einem Untergrund wird durch die Rollkugel auf 2 orthogonale Achsen übertragen, an denen Lochscheiben mit je 2 Lichtschranken sitzen. Die Lichtschranken müssen um 1/4 Lochabstand versetzt sein, um eine Rückwärts- von einer Vorwärtsbewegung zu unterscheiden (orthogonale Impulsfolgen). Die Taster ergeben Z-Signale (setzen Flags im CSR).

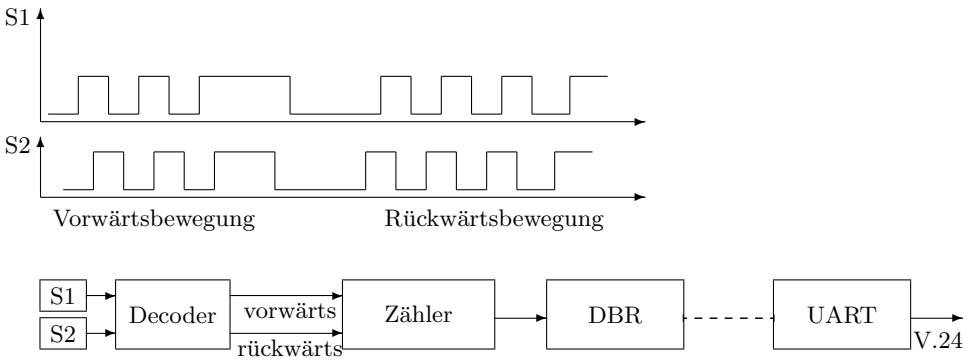


Bild n3p25

3.4.5 Scanner

Optische Abtastung einer Fläche (Vorlage) durch einen oder viele. .
Lichtstrahlen und Erfassung der reflektierten Intensität durch Sensoren (Photozellen, -dioden, -transistoren)
- einen Lichtstrahl mit X-Y-Ablenkung (vgl. Plotter)
- eine Zeile von Sensoren oder Leuchtpunkten
- Auswahl von Farben durch Lichtfilter

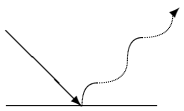


Bild n3p26

3.5 Massenspeicher

3.5.1 Magnetplatten

- Floppydisk, CD-ROM
- Festplatte (Plattenstapel)
- (CD-ROM)

Magnetplatten Ein Magnetplatte besteht in der Regel aus mehreren Plattenflächen mit je 2 Seiten. Jeder Seite ist ein Schreib/Lese-Kopf zugeordnet.

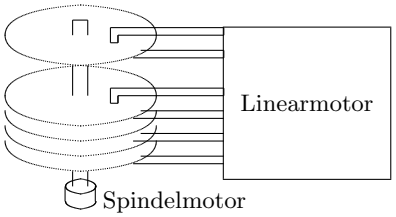


Bild n3p18

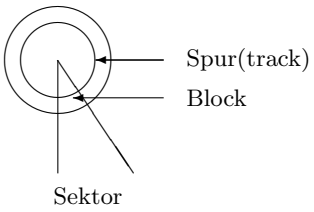


Bild n3p19

Einteilung in Spuren, Sektoren und Blöcke
Ein Datenblock liegt im Schnitt einer Spur mit einem Sektor. Übereinander liegende Spuren, die bei fester Stellung des Linearmotors erreicht werden, bilden einen Zylinder. Jeder Block kann wahlfrei adressiert werden (RAM).

Zugriff:

- auf Sektoren durch die Rotation ($\phi = \omega \cdot t$ mit $\omega = 2\pi n$)
- auf Spuren durch Radialbewegung r(t)
- CD-ROM: Spiralbewegung !

Platten-Kapazität C = B * p * t * s

- s = Anzahl der Sektoren = 10 ... 100)
- t = Anzahl der Spuren (tracks) = 100 ... 1000
- p = Anzahl der Plattenflächen = 1 ... 20
- B = Blockkapazität = 4096 Bit (= 512 Byte = 1/2 KB)

Typisch sind heute: 512 Byte · 20 Seiten · 800 Spuren · 50 Sektoren · = 400 Mbyte

Ein Datenblock wird aus einer (sequentiellen) Folge von Bits gebildet (SAM). Zusätzlich zu den Nutzdaten enthält ein Block weitere Felder: eine Präambel zur Synchronisation, ein Adreßfeld (Seite, Spur, Sektor), eine Prüfsumme und sonstige Daten am Ende (Trailer). Zwischen 2 Blocks liegt immer ein Zwischenraum (Gap), der nicht beschrieben wird.

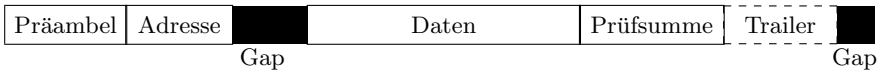


Bild n3p20

Beispiel: IBM-Diskette 3.5" (HDD) 1.44 MB . .
Hohe Schreibdichte (High Density)
Doppelseitige Belegung (Double Sided),
2 Köpfe (Heads)
80 Spuren (Tracks) mit je 18 Sektoren (Datenblocks)

60	$4E	Track-Header
18		Datenblocks →
ca. 1400	$4E	Track-Filler

12	$D0	Block-Header
3	$F5	..
1	$FE	Adreßmarke
		Track #
		Seite #
		Sektor #
	$02	Längenkennung
	$F7	
22	$4E	
3	$F5	
1	$FB	
512		Daten
1	$F7	Block-Ende
40	$4E	Block-Trailer

Magnetische Aufzeichnung

Beim Schreiben wird durch den Schreibstrom (I) ein Magnetfeld erzeugt, welches auf der Magnetschicht eine Magnetisierung (Φ) hinterläßt, deren Richtung die binäre Information repräsentiert.
Beim Lesen wird an den Stellen, wo ein Flußwechsel in der Magnetschicht erreicht wird, eine Induktionsspannung erzeugt ($U_{ind} = d\Phi/dt$).

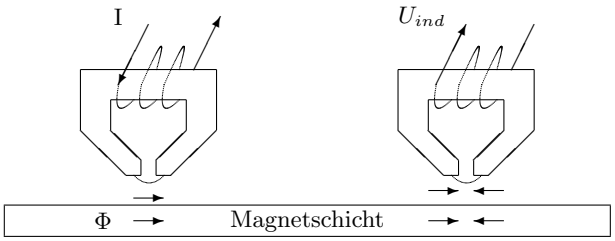


Bild n3p21

Für die Darstellung von Daten können unterschiedliche Verfahren eingesetzt werden (vgl. DIN 66010):

- Richtungsschrift, NRZ (non return to Zero): Flußwechsel bei Datenwechsel:
- Richtungstaktschrift (phase encoding): positiver Flußwechsel bei "1", negativer bei "0" (u.U. werden zusätzliche Flußwechsel eingefügt).
- Wechselschrift (NRZ 1): Flußwechsel bei "1", kein Flußwechsel bei "0".
- Wechseltaktschrift: Flußwechsel für jedes Bit, zusätzlicher Flußwechsel bei "1".

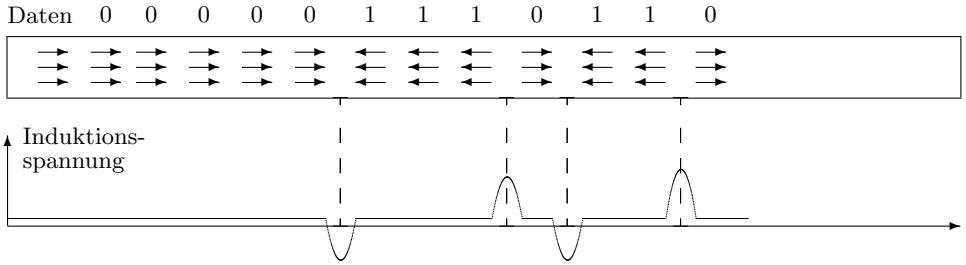


Bild n3p22

Die Darstellung in Richtungsschrift erweist als recht ungünstig, da beim Lesen langer Folgen von gleichartigen Bits kein Lesesignal erzeugt wird.

Aufzeichnungsdichten werden in Bits/mm, bits/inch (bpi) oder Bits/rad gemessen. In Deutschland werden Flußwechsel pro cm bzw. pro Zoll (1- 2.54 cm) oder pro rad ($1rad = 55.6°$, bzw. 1 Umdrehung = $2\pi rad$) angegeben. Die Werte liegen zwischen 800 und 10000 bpi bzw. 5 bis 65 Kbit/rad.
Die (radiale) Dichte der Spuren wird in Spuren/cm bzw tracks/inch (tpi) gemessen.

Bei konstanter Aufzeichnungsrate (bits/s) erhält man eine konstante Aufzeichnungsdichte in bit/rad. Wegen der nach außen zunehmenden Radien auf einer Platte wird die lineare Aufzeichnungsdichte (bpi) nach Außen ab nehmen. d.h. die einzelnen Bits sind weniger dicht gepackt und damit auch weniger störanfällig.

Die Übertragungsrate von der Platte in den Arbeittspeicher wird im wesentlichen von der Spurkapazität und der Drehzahl bestimmt. Typische Drehzahlen liegen bei 3600 min^{-1}, d.h. 60 /sec. Bei einer Spurkapazität von 100 Sektoren a 4096 bit erhält man ca 24 Mbit/s oder 3 MByte/s.

3.5.2 Magnetbänder

Die Aufzeichnung auf Magnetbänder erfolgt im Prinzip genau so wie bei Magnetplatten, allerdings werden auf einem Band oft mehrere Schreibspuren parallel aufgezeichnet; meist sind es 8 Daten- und eine Prüfspur (z.B. DIN 66015).

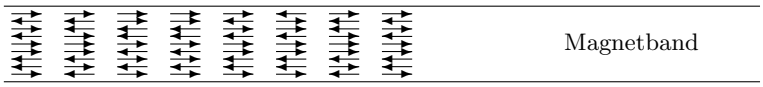


Bild n3p23

3.5.3 CD-ROM

Merkmale:
- Spiralförmige Anordnung (wie bei Schallplatte)
- Datenblöcke konstanter (linearer) Länge (variable Rotationsgeschwindigkeit)
- Sequentieller Zugriff (“intelligente“ Controller können überspringen)
- bitserielle (binäre) Aufzeichnung (Reflexion hell/dunkel)

.

Kapitel 4

Betriebssysteme

4.0 Der Rechner als System

Grobstrukturen:
- Hardware: Zentraleinheit und Peripherie
- Software: Programme und Daten

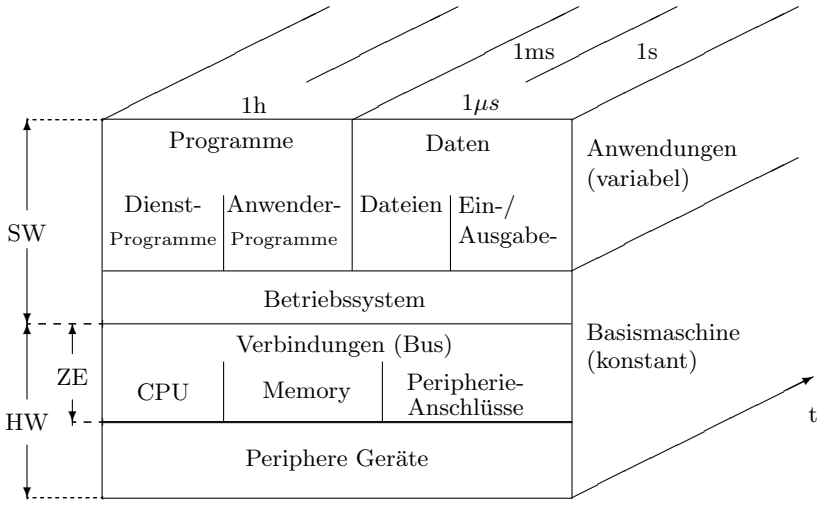


Bild n3p00

- **Dienstprogramme**
  - zur Verwaltung des Rechners (Dateiverwaltung, Systemsteuerung).
  - zur Programmentwicklung (Editor, Compiler, Binder, etc),
- **Anwenderprogramme** zur Produktion:
  Programmentwicklung, Textverarbeitung, Betriebsdatenverarbeitung, Prozeßdatenverarbeitung (CAM = Computer Aided Manufactoring), Betriebslenkung (CIM = Computer Integrated Manufactoring), Entwicklung (CAD = Computer Aided Design), etc.
- **Dateien** auf den Massenspeichern (Magnetband oder -platte) umfassen sowohl Programme, als auch Daten, die von den Produktionsprozessen stammen oder für sie bestimmt sind. Oft sind diese in **Datenbanken** organisiert.

- **Ein- und Ausgaben** vom/zum **Anwender** oder Prozeß über EA-Geräte (Eingaben von der Tastatur, der Maus oder von externen Quellen, Ausgaben auf Bildschirm, Drucker oder Plotter).

Verwaltungsaufgaben:

Verwaltung von Programmen und Daten . .Systemsteuerung: z.B. Booten
Programme laden, starten, koordinieren (→ BS) - LOAD program
Daten erstellen (durch Anwenderprogramme) - INSTALL “
Daten verwalten (durch Dienstprogramme, Betriebssystemfunktionen, Kommandos): - BUFFERS reservieren
- STACKS “

- DIR (ls): Dateien auflisten
- COPY (cp). Dateien kopieren
- RENAME (mv), Dateien umbenennen
- DELETE (rm), Dateien löschen
- TYPE (cat), Dateien anzeigen
- PRINT (lpr), Dateien ausdrucken
- FORMAT (fmt), Formatieren oder Initialisieren von Datenträgern.

Programmentwicklung . .

- **Editoren** zur Erstellung von Quellprogrammen;
- **Compiler** und **Assembler** zum Übersetzen der Quellprogramme in Maschinensprache.
- **Linker** (**Binder**) zum Anbinden von Bibliotheksroutinen, die nicht im Betriebssystem eingebettet sind, sondern zur Unterstützung einer bestimmten Programmiersprache als Laufzeitroutinen benötigt werden;
- **Loader (Lader)** zum Laden der gebundenen Programme in den Arbeitsspeicher und ihrer Eingliederung in das System; meist ist der Lader in das Betriebssystem vollständig integriert und kein Dienstprogramm mehr.

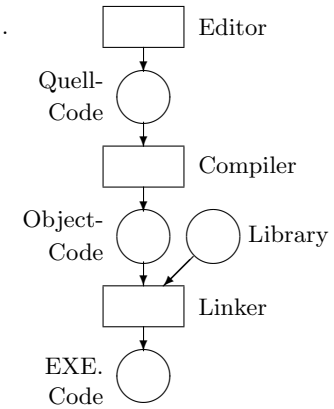


Bild n2p13

- **Testprogramme** zur Kontrolle des Funktionsablaufs und

- **Debugger** zum Auffinden von Programmfehlern (engl.: **bugs**)

- **Dateiverwaltungsprogramme**, die bis zu **Datenbanksystemen** anwachsen können (Workflow-Systeme).

Die Aufteilung der Aufgaben und Funktionen auf Dienstprogramme oder Systemroutinen ist nicht exakt festlegbar und wird in jedem **Betriebssystem** anders gehandhabt.
Zwei Grundfunktionen (Module) müssen im BS verankert sein:

- zur Kommunikation mit dem Anwender (Benutzeroberfläche)

- zum Laden und Nachladen (overlays) des BS von der Systemplatte (engl.: disk, **DOS** = disk operating system) oder von anderen Ressourcen (z.B. von einer CD-ROM oder von einem Server übers Netz).

4.1 Die Benutzeroberfläche

Zur Bedienung des Rechners müssen **Kommandos** an das Betriebssystems gegeben werden. Dazu dient eine Funktion des Betriebssystems, deren Aufgabe es ist, Benutzereingaben für das Betriebssystem anzunehmen, sie zu interpretieren und in einen entsprechenden Teil des Betriebssystems zu verzweigen.
Man unterscheidet 2 Formen

- den **textorientierten Kommandointerpreter**, **Monitor** oder die **Schale** (engl.: **shell**) z.B. unter MS-DOS oder UNIX.
- die **graphische Benutzeroberfläche** (GUI Graphical User Interface) von der aus auf die Dienst- und Anwenderprogramme zugegriffen wird. In der Regel werden hier im Multitaskingbetrieb, der viele Prozesse nebeneinander (engl.: concurrent) ablaufen läßt, dem Benutzer mehrere Fenster (engl.: Windows) für Kommandointerpreter oder für die Handhabung von bildhaften Symbolen (Icons) zur Verfügung gestellt.

In allen Fällen werden mit den Benutzereingaben Dienstprogramme oder -funktionen aktiviert, die die gewünschten Operationen durchführen.
Beispiel: Das Löschen einer Datei erfolgt durch

- das Kommando `DELETE datei` unter DOS oder `rm datei` unter UNIX
- das "Ziehen" eines Dateisymbols in einen "Papierkorb"mit der Maus unter GEM.

4.2 Betriebsarten

- stand-alone Betrieb
Im einfachsten Fall besteht die **Software** eines Rechners aus einem einzigen, festinstallierten Programm, das direkt auf die Hardware zugreift. Man findet sie in Kompaktsystemen (engl.: embedded systems), in Speicherprogrammierten Steuerungen (SPS) oder als Firmware in einem „intelligenten“ (μ-Prozessor gesteuerten) Gerät. Manchmal basiert sie auf einem speicherresidenten Betriebssystem, das eine reduzierte (und ROM-residente) Variante eines konventionellen BS ist (z.B. Windows CE).
- Stapelbetrieb (engl.: **batch mode**):
der Benutzer stellt alle Anforderungen in Form von Kommandos und alle Eingabedaten zu einem Auftrag zusammen (engl.: **batch job**). Erst nach dessen Beendigung stehen die Ergebnisse zur Verfügung. Dabei hat man (bewußt) keine Möglichkeit vorgesehen, in den Bearbeitungsablauf steuernd einzugreifen. Lediglich einen Abbruch (engl.: **abort**) kann man erzwingen; dann sind in der Regel aber alle Zwischenergebnisse verloren.
Beispiele: Lohnabrechnungen, Steuerbescheide .
- Dialogbetrieb (engl.: **interactive mode**):
der Benutzer macht Eingaben, wenn das Programm läuft und auf Eingaben wartet. Unter Umständen kann er sie sogar kurz vorher eingeben (engl.: **type ahead**), wenn ein Puffer dafür zur Verfügung steht. Die Ausgaben erscheinen, sobald sie berechnet sind, sodaß der Benutzer seine folgenden Eingaben danach ausrichten kann. Beim Multitasking können hierbei Verzögerungen eintreten, wenn mehrere Benutzer auf

dieselben Betriebsmittel zugreifen, z.B. auf den Drucker.
- **Realzeitbetrieb** (engl.: **real time processing**):
zeichnet sich dadurch aus, daß bestimmte Antwortzeiten eingehalten werden müssen, d.h. daß auf eine Eingabe innerhalb einer bestimmten Zeit eine Ausgabe erfolgt.

Betriebssysteme und Betriebsarten:

1. Single User	Multi User (Mehrbenutzerbetrieb)
z.B. MS-DOS	Unix
ohne Datenschutz	mit Datenschutz: (Benutzerkennung, Passwort)
2. Single Tasking	Multi Tasking (Mehrprogrammbetrieb)
ein einziges (komplexes)	mehrere Programme in Konkurrenz
Programm in Betrieb	um 1 oder n CPU (concurrency)
	Zuteilungsstrategien (scheduling):
nur Spezialrechner z.B. für	– time-sharing
"Intelligente" Geräte	– resource-sharing
Wetterkarte-Super-Computer	Windows, Unix, MVS(IBM), VMS(DEC)
3. Single Processing	Multiprocessing
Ein Prozessor (1 CPU / ZE)	Mehrere Prozessoren ($n \leq 2^{16} = 65\,536$)
	Massive Parallel Computing
z.B. Prozeßrechner in Geräten	z.B. Hypercube, Suprenum

Mehrbenutzerbetrieb (engl.: multi user) erlaubt die Nutzung eines Rechnersystems durch mehrere Benutzer, die gegeneinander abgeschirmt werden müssen, damit keiner einen anderen fahrlässig oder absichtlich stören kann. Diese Problematik wird als "Zugangsschutz" und als "Speicherschutz" im Bundesdatenschutzgesetz (BDGS) behandelt.

Multitasking oder **Multiprogramming** ist der nächste Schritt zu einer effektiveren Nutzung des Rechners. Hier werden die Programme nicht mehr nacheinander abgearbeitet, sondern abwechselnd, so daß alle Benutzer den Eindruck erhalten, gleichzeitig bedient zu werden, oder ein einzelner Benutzer nutzt mehrere Programme gleichzeitig. Tatsächlich wird jedes Programm immer nur für eine kurze Zeit bearbeitet und dann zum nächsten weitergeschaltet. Dazu ist ein ausgefeiltes Taskmanagement notwendig (s.w.u.).

Aufgaben:
Bereitstellung von Funktionen durch Module und Schnittstellen

- Module für den Zugriff auf die Hardware (Gerätetreiber)

- Module für komplexe Rechenoperationen (sin x oder ASCI–HEX) (SW-Treiber)

- Module zur Systemsteuerung (Interrupt-, Task-, Zeit-Management)

- Abstrahierung von der konkreten Hardware (Hardware-Software-Interface)

4.3 Aufgabe und Struktur eines Betriebssystems

4.3.1 Struktur

Hierarchie von abstrakten Maschinen:

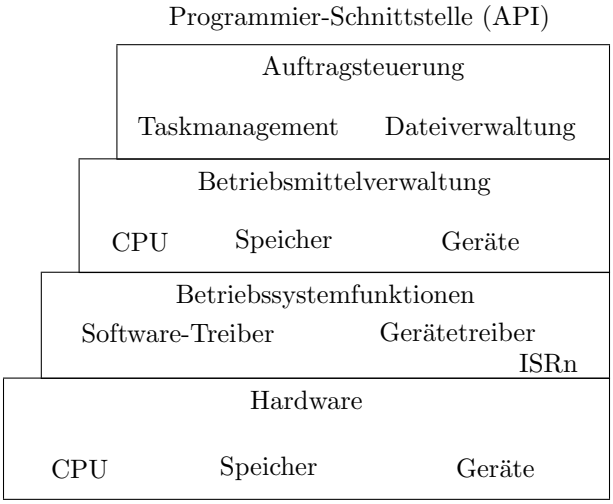


Bild r3p03

Der Betriebssystemkern

Der Betriebssystemkern (kernel, nucleus) umfaßt

- Taskmanagement (Prozeßverwaltung) mit Tabellen und Algorithmen zu deren Verwaltung, insbesondere für die Taskumschaltung bei Multitasking
- Dateiverwaltung mit Routinen für Dateizugriff und Tabellen für Dateistrukturen.
- Betriebsmittelverwaltung mit Tabellen für die Speicherbelegung, Gerätebelegung, laufende Tasks, Interrupt-Vektoren und Algorithmen zu deren Verwaltung.
- Betriebssystemfunktionen, z.B. Datenkonvertierung, Gerätetreiber, Interrupt Service Routinen (ISR)

Der Betriebssystemkern ist der Teil der Betriebssystem-Software, der der Hardware am nächsten ist und ihr speziell angepaßt sein muß.

Der Betriebssystemkern wird oft in einem ROM gespeichert, um größtmögliche Sicherheit gegen ein Überschreiben dieser Software zu haben. Variablen, Tabellen und ladbare Treiber müssen dann natürlich in einem "RAM" abgespeichert werden.

4.3.2 Taskmanagement

Beim **Multitasking** stehen in der Regel die Programme schon ablaufbereit im Speicher. Dann wird zu jedem Programm ein Speicherbereich zur Ablage von Zwischenergebnissen und Zustandsparametern benötigt, die zum Zeitpunkt der Weiterschaltung vorliegen und 'gerettet' werden müssen; dazu dient zum einen der **Stack** zum anderen der Task Control Block (**TCB**), die den Task-Kontext enthalten. Das Betriebssystem bildet daraus eine Tabelle in Form einer verketteten Liste.

Der Mechanismus zur **Weiterschaltung** von einer Task zur anderen stellt eine besondere Aufgabe dar. Im einfachsten Fall wird die Weiterschaltung zu festen Zeiten vorgenommen (engl.: **time sharing**) und die Programme in fester Reihenfolge abgearbeitet (engl.: **round robbin**).

Eine verbesserte Ausführung bearbeitet die Programme nach ihren Anforderungen und Prioritäten, so daß alle Betriebsmittel möglichst gut verteilt und ausgenutzt werden (engl.: **resource sharing**). Dabei erfolgt die Weiterschaltung auf Grund von Ereignissen (events). Eine wichtige Aufgabe des Betriebssystems besteht nun darin, die **Kommunikation** zwischen den einzelnen Programmen (engl.: **inter task communcation**) zu ermöglichen. Dabei entsteht eine ganze Reihe von Problemen, zu deren Lösung geeignete Strategien erforderlich sind. Da die Zahl der möglichen Beziehungen zwischen mehreren Tasks mit deren Anzahl dramatisch (etwa mit n!) anwächst, kann ein solches System bald unüberschaubar werden.

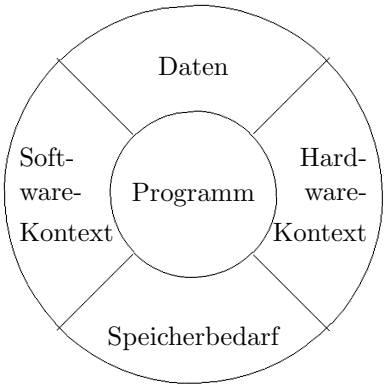


Bild r3p04

. .Eine **Task** besteht aus dem **Programm** und dem **Kontext**, der im **Taskheader** eingetragen ist.
Beim Laden einer Task werden die benötigten und im Taskheader angegebenen Betriebsmittel zugeteilt und im Task Control Block (TCB) abgelegt.

- Der Hardware-Kontext bestimmt die benötigten Geräte (exclusive oder sharable)
- Der Software-Kontext bestimmt die benötigten Systemfunktionen, Prioritäten, Zugriffsrechte, Umgebungsvariablen, Prozeßnummer (pid).
- Die Datenstrukturen werden im Speicher angelegt (Stack, Register)
- Daraus ergibt sich der gesamte Speicherbedarf

Die Aktivierung einer Tasks bei der Weiterschaltung durch das BS erfolgt als Unterprogrammaufruf, bei dem ein neuer Instruction Pointer (Program Counter) in der CPU gesetzt wird.

Taskzustände und -übergänge

E_1: Laden und Zuteilen von Betriebsmitteln

E_2: Zuteilung der CPU

E_3: Anfordern von Betriebsmitteln

E_4: Zuteilen von Betriebsmitteln

E_5: Unterbrechung

E_6: Programmende

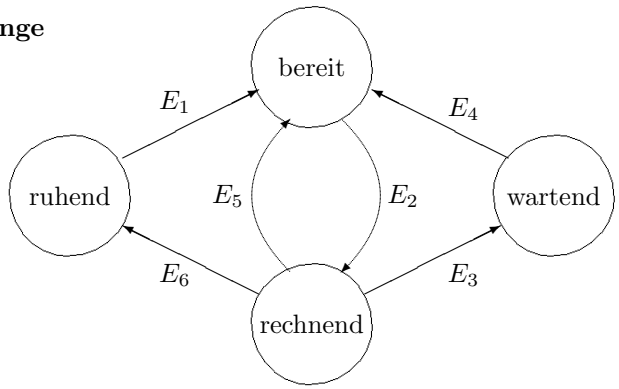


Bild r3p05

Task Scheduling, Algorithmen zur CPU-Zuteilung . .
- preemptiv
- Run-to-Completion
- Round-Robin
- Prioritäts-Scheduling
- Klassen-Scheduling

- zeitgesteuert
- ereignisgesteuert

Prozeßliste: LINUX-Kommando "ps" . .**Prozeßbaum:** Kindprozesse

pid	ppid	status	name
1234	1	Running	ps
1235	1	Hibernating	xyz
1236	1234	Stopped	rst
1237	1236	Waiting	uvw

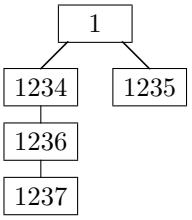


Bild r3p05a

4.3.3 Dateiverwaltung

a) Dateien auf Massenspeichern (files)
- auf blockorientierten Massenspeichern (oder im Arbeitsspeicher)
- Verzeichnisstruktur baumförmig (vernetzt bei UNIX durch links)
- Dateinamen und Pfade
- Dateiattribute ugo(a) * rwx (dlc)

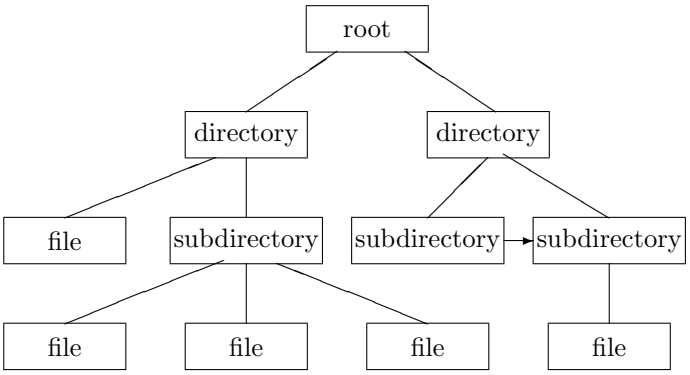


Bild r3p06

b) Ein-/Ausgabedaten
- Gerät = Datei ohne Namen (special file), Pseudo-Datei
- zeichenorientiert
- Standardein- und -ausgabe (Terminal = Tastatur und Bildschirm)

c) Interne Dateien
- files, arrays
- Puffer, Cache für Ein-/ausgabe
- Kommunikation: Pipe, Shared Memory, Semaphore, Signale

d) Beispiel: Das Diskettenformat unter MS-DOS

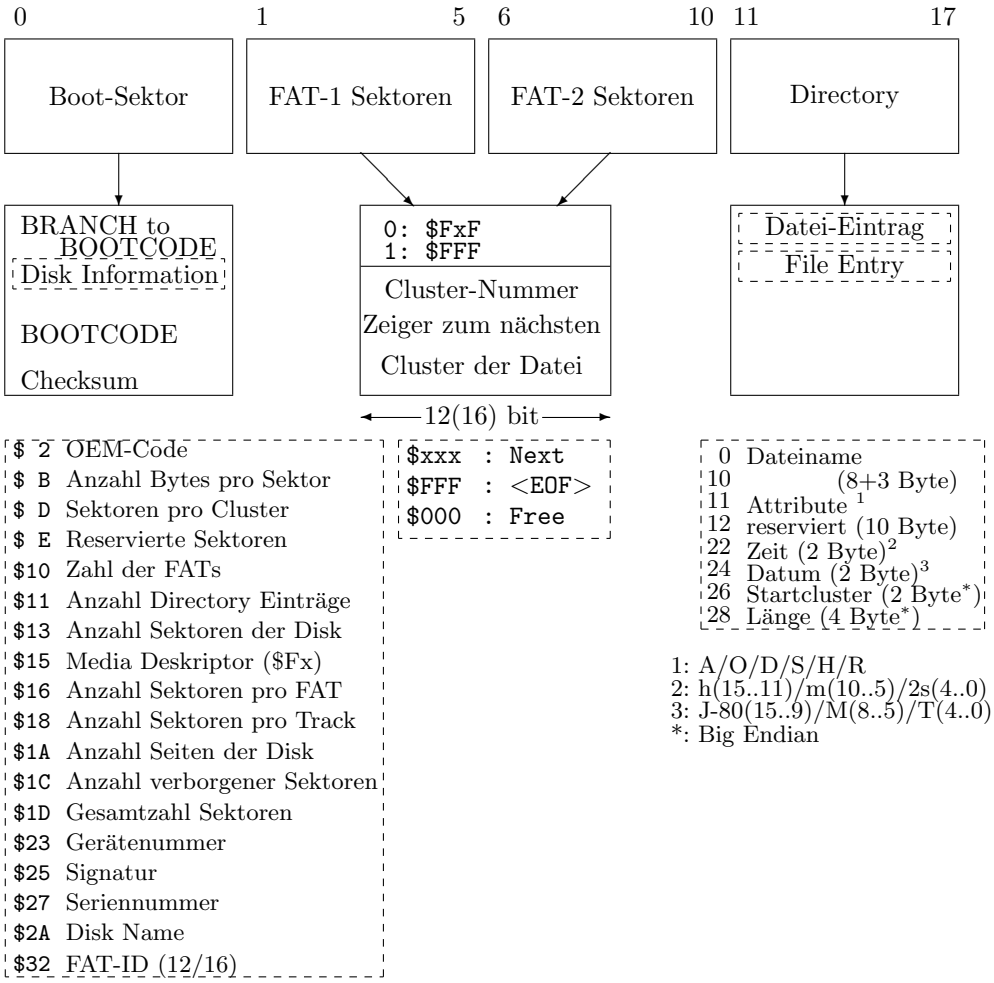


Bild r3p06a

4.3.4 Betriebsmittelverwaltung

- Prozessor (CPU) (auch mehrere bei Multiprocessing)
  Zuteilung durch Task-Scheduler
- Arbeitsspeicher (Mmemory), Puffer (Cache)
  - Segmentation (Speicherschutz durch MMU)
  - Paging (Auslagern von Seiten auf Massenspeicher)
  - Swapping (Auslagern von Tasks auf Massenspeicher)
- Geräte (devices)
  Geräte-Treiber (driver)

4.3.5 Gerätetreiber

Treiber (Gerätetreiber, device driver) sind Software-Module innerhalb des Betriebssystems, die zu Datenübertragung und Kommunikation zwischen Zentraleinheit und peripheren Geräten dienen. Für jedes Gerät bzw. jede Geräteart muß ein Treiber vorhanden sein.

Ein Treiber enthält die Interrupt-Service-Routine (ISR) und Module zur Vorbereitung, Inbetriebnahme und zur Abschaltung des Geräts.

In Realzeitsystemen müssen oft spezielle Treiber für spezielle Geräte (z.B. ADC) installiert werden. Das muß vom Betriebssystem unterstützt werden.

Treiber sollen resident gehalten werden können, d.h. sie sollen dann nicht, um z.B. Speicherplatz für andere Programme zu schaffen, auf Massenspeicher ausgelagert werden. Das kann zu unvertretbar langen Reaktionszeiten führen.

Jeder vorhandene Treiber belegt Betriebsmittel (Speicherplatz, CPU-Zeit). Daher ist es vorteilhaft, wenn Treiber temporär (z.B. beim Booten) ge- oder entladen werden können, um maximale Systemleistung zu erhalten.

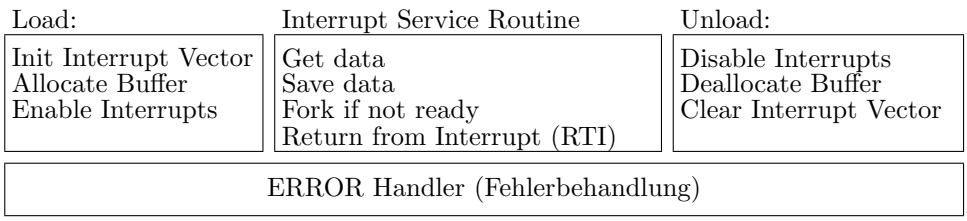


Bild r3p07

4.3.6 Anforderungen

Systemeigenschaften

- Interrupt-Verarbeitung
- Prioritätensteuerung: priorisierte Interrupts, priorisierte Tasks
- residente oder ladbare Teiber
- Resource sharing
- Multitasking
- Multiprozessorfähigkeit
- ROM-fähig
- skalierbare Prozessorleistung
- Skalierbarkeit der Hardware und der Peripherie

Systemfunktionen

- Erfassung von gleichzeitig auftretenden externen Prozeßereignissen
- IPC, Inter Process Communikation: Semaphore (switches, signals, flags), shared memory, pipes, mailboxes
- Dateisystem mit schnellem Zugriff, gesichertem Zugriff und Pufferung (Caching)
- Zeitschrankenüberwachung
- Schutzmechanismen: Speicherschutz, Zugriffsschutz
- Fenstertechnik (Benutzeroberflächen)

Systemmanagement
- Konfigurierbarkeit (Installation von Tasks und Treibern)
- Power-fail Vorkehrungen (bei Netzspannungsausfall)
- Stapelverarbeitung, Batchbetrieb
- Dienstprogramme für Dateiverwaltung und Kommunikation
- Spooling
- Error-Logger
- Backup
- Dokumentation

Unterstützung für die Programmentwicklung:
- definierte/normierte Anwenderschnittstellen (User Interface)
  API (Application Program Interface)
- Entwicklungsumgebung
- Datenbanksystem als Entwicklungstool
- Testumgebung: interaktiver Debugger

4.3.7 Beispiele für Betriebssysteme

Betriebssystem	Prozessor	Hersteller	Anmerkungen
MS-DOS	Intel	MicroSoft	für PCs
Windows	Intel 80x86	MicroSoft	dito
OS/2	Intel 80x86	IBM	für Workstations
UNIX	diverse	AT&T; u.a.	dito
VAX-VMS	VAX	DEC	für "Minicomputer"
OS-9	viele		zur Realzeitverarbeitung
LynxOS	diverse		dito
RTX	M 68 000		dito
RSX-11	PDP-11	DIGITAL (DEC)	dito, historisch
RT-11	PDP-11	DIGITAL (DEC)	dito, historisch

4.4 Systembetrieb

4.4.1 Systemstart (Booten)

Das Einschalten des Rechners erfolgt in mehreren Arbeitsschritten:
- Erstellung eines definierten Anfangszustands (Z_0), insbesondere der Instruction Pointer (IP oder PC, Programm Conter) muß auf einen bestimmten Wert gesetzt werden, der auf einen "Urlader" (Bootstrap) in einem ROM (im BIOS) verweist. Auch der Stack Pointer (SP) muß einen bestimmten Wert erhalten und alle Flags der CPU (im PSW).
  Durch ein geeignetes Signal (INIT) werden alle Geräteanschlüsse über den Bus vom Bootvorgang informiert und haben sich selber in einen definierten Anfangszustand zu bringen.
- Der Urlader führt eine Reihe von Systemtests (auf den Arbeitsspeicher und die Geräteanschlußregister) durch, bevor er das Betriebssystem von der Systemplatte in den Arbeitsspeicher lädt. Dabei kann er Konfigurationsparameter aus einer vom Administrator verwalteten Datei (z.B. config.sys) lesen und verwenden.
- Nach dem Laden des Betriebsystems führt dieses wiederum weitere Tests an Hard- und Software aus, bevor es weitere Systemkomponenten und Dienst- und Anwenderprogramme lädt, wie sie in einer oder mehreren weiteren Konfigurationsdateien (z.B. autoexec.bat) festgelegt sind.
- In den meisten Fällen kann der Bootvorgang durch Benutzereingriff unterbrochen und in interaktive Module verzweigt werden, um neue Systemparameter einzustellen oder Alternativen auszuwählen.

4.4.2 Systempflege

Systemkonfigurierung, Systemmanagement
- Installation von Treibern,
- Entfernen von Treibern,
- Einrichten von Hintergrund-Tasks (Daemons)
- Einstellen von Systemparametern wie Prioritäten von Tasks, Zeitschranken

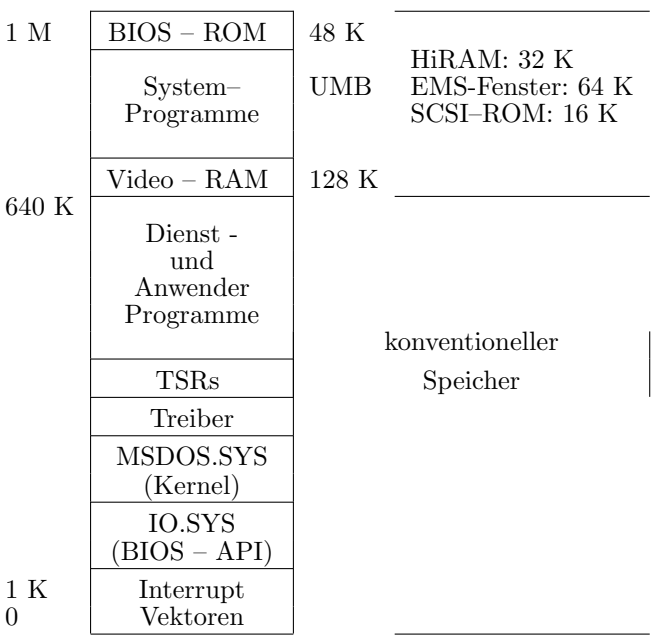
4.5 Aufbau eines Betriebssystems

Konstruktive, statische Aspekte
Typische Bestandteile sind:
- Tabellen, z.B. Interrupt Vektor Tabelle (IVT)
- Datenpuffer, z.B. Video-RAM
- BS-Routinen, wiedereintrittsfähig (reentrant), verschiebbar (position independent)

Die Programme und Daten werden unterschiedlich gespeichert:
- im ROM residente Basisfunktionen, z.B. BIOS (Basic Input/Output System)
- im RAM residente Tabellen, z.B. die Interrupt Vektor Tabelle (IVT), und Hintergrundprogramme, z.B. Daemon-Programme oder TSR-Programme (Terminate and Stay Resident)
- im RAM und auf der Systemplatte austauschbare Routinen (Swapp oder Overlay)

Im Adreßraum liegt das BS meist am unteren oder am oberen Ende. Manche BS (z.B. MS-DOS) machen beides.
Die Geräteregister liegen in einem eigenen Adreßbereich, der Input/Output Page, der entweder ein (gemappter) Bereich im obersten Abschnitt des Adreßraums ist, oder ein eigener Adreßraum, der durch ein zusätzliches Adreßbit zu erreichen ist. (Beim Intel 80x86 wird dieses nur durch bestimmte Instruktionen wie IN und OUT aktiviert.)

Beispiel:
Belegung des Speichers unter MS-DOS (V 6.20)



Kapitel 5

Prozeßdatenverarbeitung

Prozeßketten
- Meßketten zur Erfassung von Meßwerten
- Steuerketten zur Erzeugung von Stellwerten

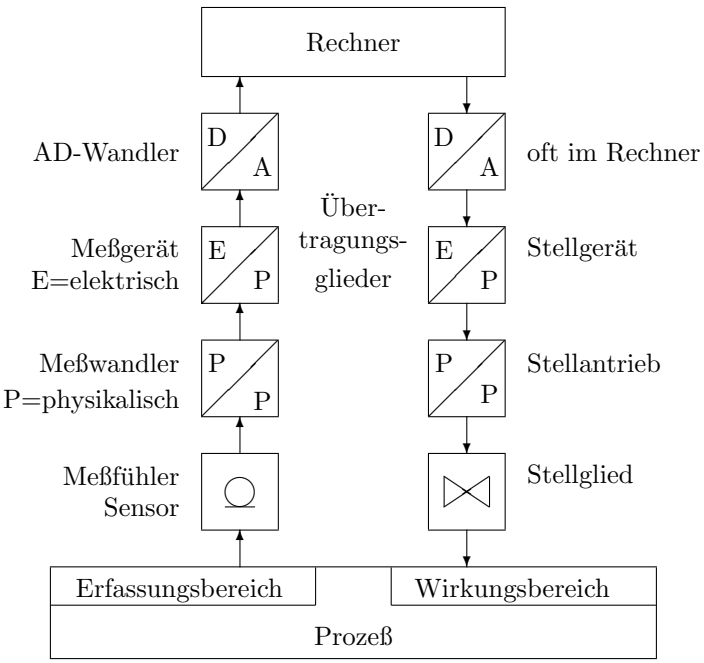


Bild n3p50

Die Prozeßperipherie bildet zwischen Rechner und Prozeß jeweils eine Kette zur Erfassung von Meßgrößen als Eingabedaten und zur Erzeugung von Stellgrößen aus den Ausgabedaten des Rechners.
Die wichtigsten Glieder in diesen Ketten machen Umsetzungen zwischen digitalen und analogen Werten (D/A) sowie zwischen elektrischen und nichtelektrischen physikalischen Größen (E/P). Auch Umwandlungen zwischen verschiedenen physikalischen Größen kommen vor (P/P).

5.1 Prozeßperipherie

Digitale Ein- und Ausgabe
Analogein- und -ausgabe, ADC, DAC

5.1.1 Digitale Ausgabe

Digitale Ausgabe (digital output, DO) erfolgt in der Regel bitparallel direkt aus einem Datenregister (DBR).
Die Nachbearbeitung erfolgt in einem Übertragungsglied (D/D)

- zur Verstärkung der elektrischen Spannung oder Leistung
- zur Differenzierung (Impulsbildung)
- zum Integrieren (Zähler).

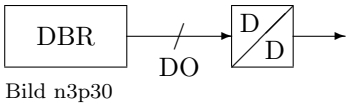


Bild n3p30

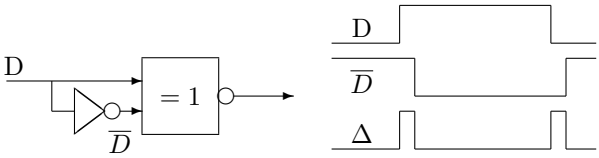


Bild n3p31

Differenzieren mit Hilfe eines Inverters (mit geeigneter Durchlaufverzögerung) und einem XOR-Glied.

5.1.2 Digitale Eingabe

Digitale Eingabe (digital input, DI) erfolgt ebenfalls meist bitparallel in ein Datenregister (DBR) und benötigt vorangehende Signalverarbeitung (D/D).

- zur Impulsformung (Begrenzer, Hoch- oder Tiefpaß)
- zur Differenzierung (Impulsbildung)
- zum Integrieren (Zähler).

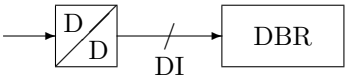


Bild n3p32

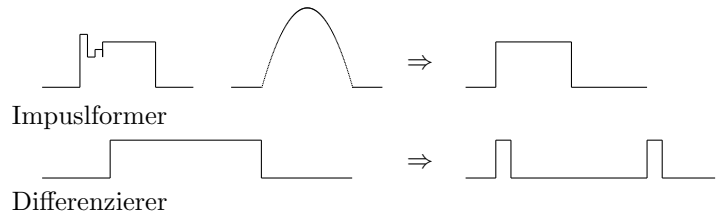


Bild n3p33

5.1.3 Analogausgabe

Bei der Analogausgabe (analog output, AO) werden die Daten (D) aus einem Datenregister (DBR) mittels eines Digital-Analog-Wandlers (Digital to Analog Converter, DAC) in elektrische Spannungswerte (U) umgesetzt.

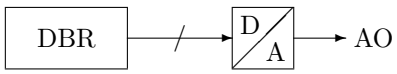


Bild n3p34

$$U_x = U_{ref} \cdot D/2^n$$

a) Linear-Umsetzer (Potentiometric Ladder)

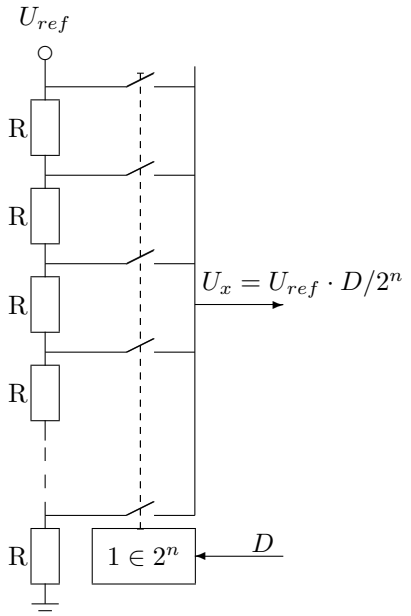


Bild n3p35

Beim Linearumsetzer wird von einer Spannungsteilerkette ein Spannungswert durch eine geeignete Anordnung von Schaltern abgegriffen. Es bildet also ein diskretisiertes Potentiometer.
Hier ist eine vereinfachte Anordnung gezeigt, für die eine geeignete Decodierung der n Datenbits für einen von 2^n Schaltern erforderlich ist ($1 \in 2^n$).
In der Praxis wird meist ein binärer Baum von Schaltern verwendet.

Eigenschaften
- schnell ($t_W < 1\mu s$)
- grob ($n = 10$ Auflösung ca. 1:1000)
- aufwendig (je 2^n Widerstände und Schalter)

b) binär gewichteter Stromaddierer (Stromsummenwandler)
Beim Stromaddierer werden mit den Bits b_i des Dualwerts Stromflüsse I_i geschaltet und addiert, so daß daraus eine dem Digitalwert entsprechende Analogspannung U_x entsteht.

$$U_x = R \cdot \sum_{i=1}^{n} I_i \cdot b_n \qquad \text{mit} \qquad I_i = U_{Ref}/R \cdot 2^{n-i-1}$$

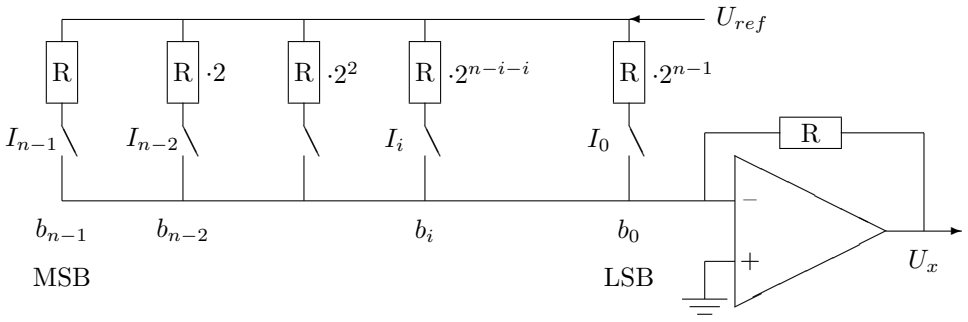
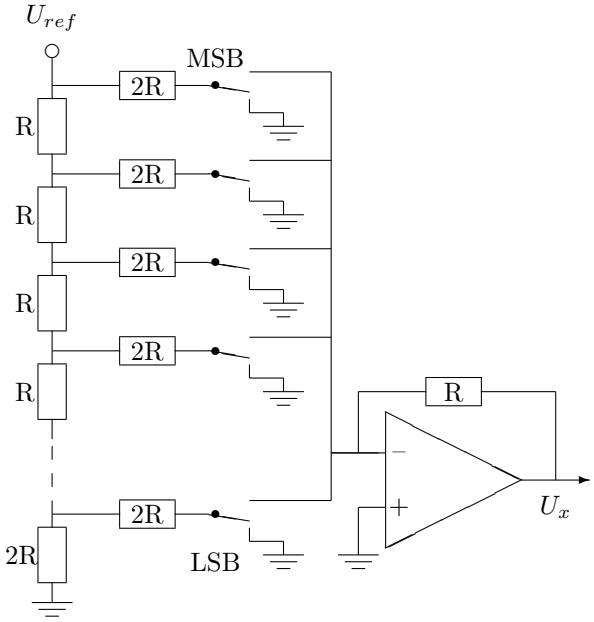


Bild n3p36

Eigenschaften
- geringer Aufwand (n Widerstände, n Schalter, 1 Verstärker)
- mittelschnell ($t_W > 1\mu s$)
- fein ($n \geq 16$ Auflösung ca. 1:1000)
- Monotoniefehler (Sprung beim Umschalten zum höchstwertigen Widerstand)

c) R-2R-Netzwerk



. .Mittels eines geeigneten Schalt- und Widerstandsnetzwerks werden Ströme erzeugt und geschaltet, die dem Dualwert D entsprechen. Der Vorteil dieser Schaltung liegt darin, daß alle Widerstände in derselben Größenordnung (R, 2R) liegen und fertigungstechnisch leichter auf den Sollwert abgeglichen werden können; auch die Langzeitstabilität verbessert sich hierdurch deutlich.

Bild n3p37

Eigenschaften
- geringer Aufwand (2n Widerstände, 2n Schalter, 1 Verstärker)
- mittelschnell ($t_W > 1\mu s$)
- fein ($n \geq 16$ Auflösung ca. 1:100 000)
- keine Monotoniefehler, geringe Linearitätsfehler

d) Bipolare Ausgangsspannungen

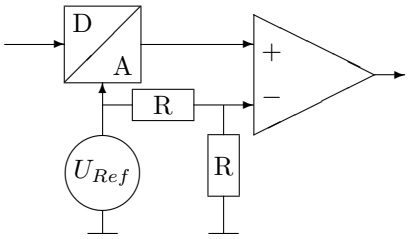


Bild n3p38

. .Um Ausgangsspannungen beliebiger Polarität (±) zu erzeugen, wird eine Vorspannung (Bias) mit dem halben Wert der Referenzspannung (U_{Ref}) auf den verwendeten Operationsverstärker gegeben. Dann müssen die Digitalwerte im 2-er Offset dargestellt werden (nicht im 2-er Komplement):

D	U_x
00000	$-U_{Ref}/2$
10000	0
11111	$+U_{Ref}/2$

e) Qualitätskriterien

- Auflösung: Die Anzahl der Datenbits n ergibt die Zahl der Stufen und die kleinste Schrittweite ($U_{Ref}/2^n$).
- Linearität: Die Stufen können durch unterschiedliche Widerstandswerte innerhalb des Umsetzers unterschiedliche Höhen besitzen.
- Monotonie: Bei groben Abweichungen der Widerstandswerte vom Sollwert kann die Monotonie gestört werden, d.h. Ausgangsspannungen, die zu einem höheren Digitalwert gehören, sind kleiner als der Vorgängerwert.
- Geschwindigkeit: wird meist durch den verwendeten Operationsverstärker bestimmt. Typisch sind hier $5V/\mu s$

Die hier vorgestellten Verfahren:

- Linearumsetzer: sehr schnell ($3ns$), sehr linear, monoton, geringe Auflösung ($n \leq 8$), teuer
- Stromsummenwandler: wenig linear, wenig monoton, langsam ($1\mu s$), hohe Auflösung ($n \leq 18$), billig
- R-2R-Wandler: linear, mäßig monoton, hohe Auflösung ($n \leq 18$), preiswert

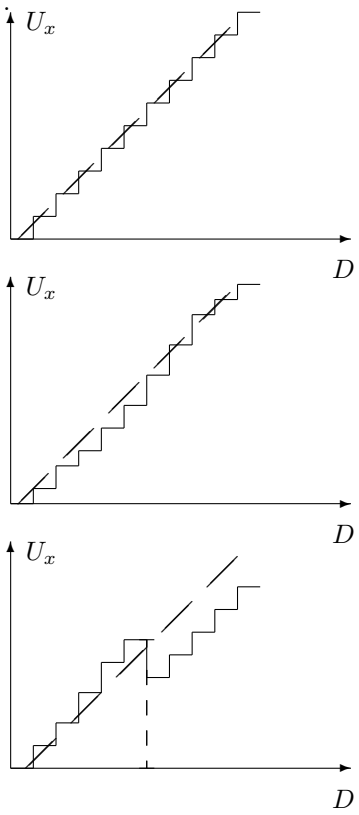


Bild n3p39

5.1.4 Analogeingabe

Bei der Analogeingabe (analog input, AI) werden die Daten durch Vergleich mit einer Referenzspannung U_{Ref} in einem Analog-Digital-Wandler (Anlog to Digital Converter, ADC) gewonnen und im Datenregister DBR abgelegt. Im Gegensatz zur Anlogausgabe müssen in den meisten Fällen Zeitbedingungen beachtet werden, denn die Erfassung und Umwandlung eines Meßwerts erfordert hier eine beträchliche Zeit (die Konvertierungszeit t_c).

Zur Steuerung werden zusätzliche Informatio-. . nen benutzt z.B. ein BUSY-Bit im CSR, welches anzeigt, daß eine Wandlung gerade erfolgt, bzw. ein READY-Bit, welches eine erfolgreiche Wandlung anzeigt. Manche Wandler benötigen ein Startsignal, das vom Programm durch das Setzen eines Bits (GO) im CSR erzeugt werden muß.
Hierin unterscheiden sich die Verfahren stark.

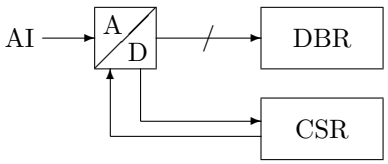


Bild n3p40

$$D = \frac{U_x}{U_{Ref}}(2^n - 1)$$

Die Qualitätskriterien sind vergleichbar mit denen der Digital-Analog-Wandler.

a) Parallelwandler (Flash-Converter)

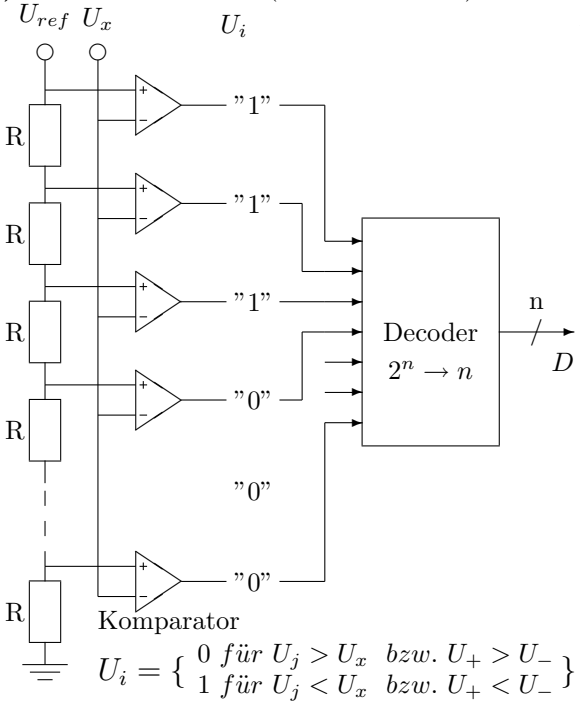


Bild n3p41

. .Im Parallelwandler wird die Eingangsspannung U_x mit einer Reihe von Spannungen verglichen, die durch potentiometrische Teilung aus der Referenzspannung U_{Ref} gewonnen werden. Die hierbei entstehende Reihe von Binärwerten stellt aber noch keine Dualzahl dar, sondern muß durch einen geeigneten Decoder erst umgewandelt werden. Der Vorteil dieses Wandlers ist seine hohe Geschwindigkeit ($t_c \leq 10ns$), der Nachteil seine geringe Auflösung ($n \approx 8$).
Eigenschaften
- sehr schnell ($t_W \leq 10ns$)
- sehr grob
  ($n = 8$ Auflösung ca. 1:250)
- sehr aufwendig
  (je 2^n Widerstände und Komparatoren)

b) Integrationswandler

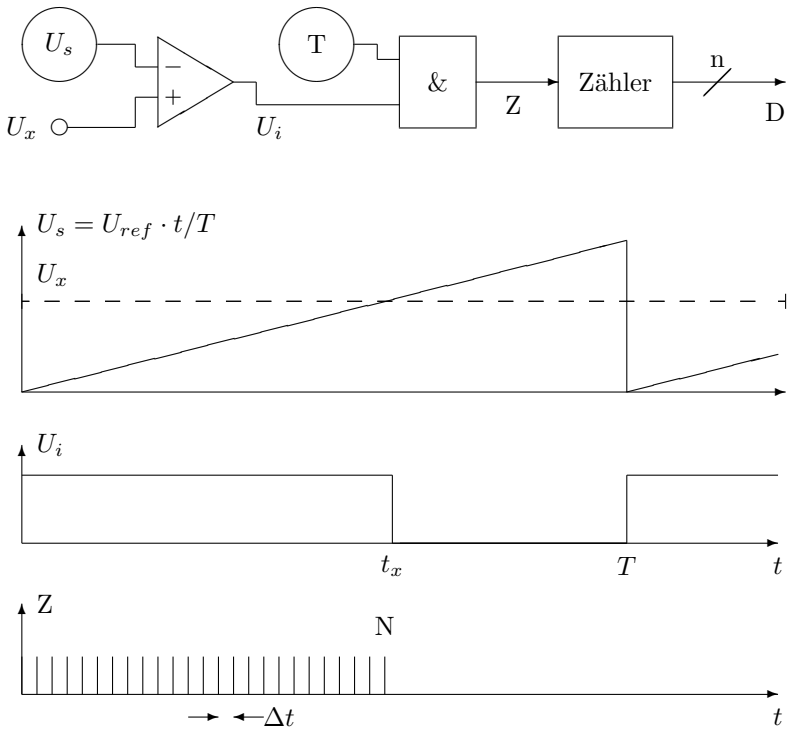


Bild n3p42

Beim Integrationswandler wird die Eingangsspannung U_x mittels eines einzigen Komparators mit einer zeitlich linear ansteigenden (Sägezahn)Spannung U_s verglichen. Bis zum Erreichen des Istwerts U_x wird eine Impulsfolge Z gezählt, deren Wert N den Digitalwert ergibt:

$$\boxed{N = t_x/\Delta t = \frac{T}{\Delta t} \cdot U_x/U_{Ref}}$$

Zur Erzeugung der Sägezahnspannung kann vorteilhaft ein DAC eingesetzt werden, der über einen Zähler eine monoton ansteigende Folge von Werten erhält. Die Auflösung diese Wandlers kann praktisch beliebig groß gemacht werden, dazu muß nur die Impulsfolgefrequenz $f_p = 1/\Delta t$ oder die Meßzeit T großgemacht werden. Letzteres verringert aber die Geschwindigkeit ($t_c = T$). Typische Wandlungszeiten liegen bei 10 ms bei einer Auflösung von 20 bit.
Eigenschaften
- sehr langsam ($t_W \geq 1s$)
- sehr fein ($n \geq 16$ Auflösung ca. 1:100 000)
- aufwendig (verschiedenste Komponenten)
- Linearität abhängig von Sägezahn)

c) Stufenumsetzer

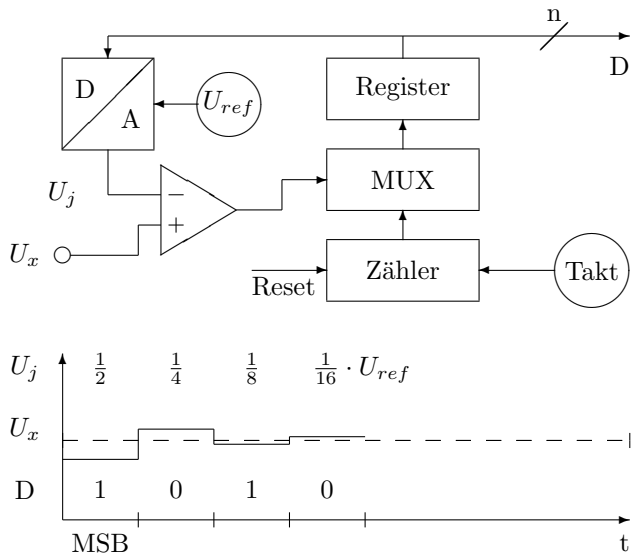


Bild n3p43

Der Stufenumsetzer (Wägecodierer) arbeitet nach dem Prinzip der sukzessiven Approximation. Ähnlich wie beim Integrationswandler wird hier die Eingangsspannung U_x mit einer Spannung U_j verglichen, die jedoch nicht linear durchgefahren, sondern in einem Rückkopplungsverfahren in Binärstufen variiert wird. Der Endwert wird so schon nach n Schritten erreicht. Dieses Verfahren ist recht schnell, pro Schritt werden typ. 1 μs benötigt, so daß eine Wandlung mit einer Auflösung von typ. 16 bit insgesamt $t_c = 16\ \mu s$ benötigt, was einer Konvertierungsfrequenz von $62.5kHz$ entspricht.
Eigenschaften
- schnell ($t_W \approx n \cdot 1\mu s$)
- sehr fein ($n \geq 16$ Auflösung ca. 1:100 000)
- aufwendig (verschiedenste Komponenten)
- Linearität und Monotonie abhängig vom DAC)

5.1.5 Zeitgeber

a) absolute Zeit: Die Systemuhr

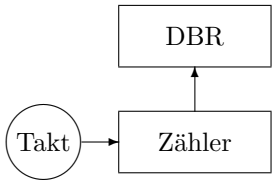


Bild n3p45

. .Die absolute Zeit erhält man von einer Digitaluhr, die im wesentlichen nur aus einem Taktgeber und einem Zähler besteht.
Funkuhr: Von der PTB (Physikalisch Technische Bundesanstalt in Braunschweig) wird ein Signal hoher Frequenzstabilität über den Sender DCF 77 ausgestrahlt, dem die aktuelle Zeit digital aufmoduliert ist.

b) relative Zeit: Timer

Zeiterfassung (Stoppuhr):

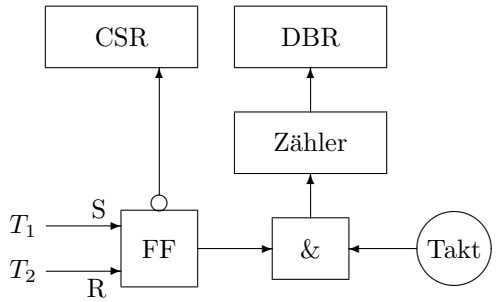


Bild n3p46

..Bildung der Zeitdifferenz ΔT aus einem Start-Ereignis T_1 und einem Stopp-Ereignis T_2.

$$\Delta T = T_2 - T_1$$

Zeitgeber (Wecker):

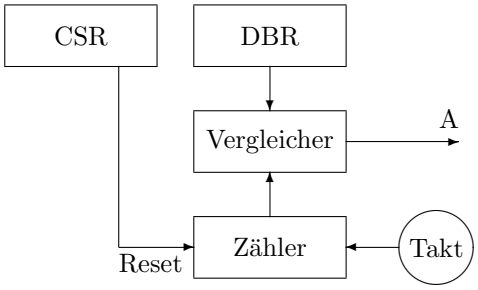


Bild n3p47

..Zählung einer Impulsfolge bekannter Frequenz f bzw. einer Zeitbasis $T = 1/f$

$$\Delta T = N \cdot T$$

während der das Ausgangssignal $A = 1$ ist.

5.2 Prozeßglieder

Prozeßketten
- Meßketten zur Erfassung von Meßwerten
- Steuerketten zur Erzeugung von Stellwerten

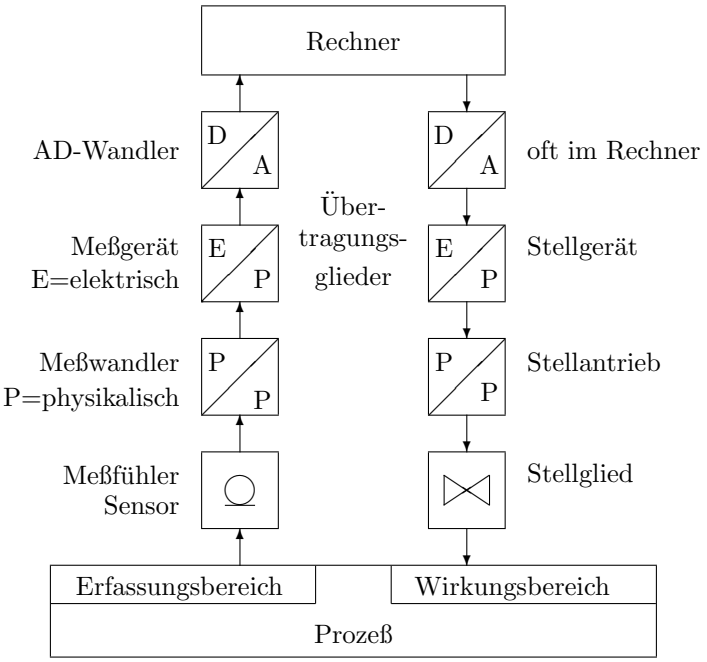


Bild n3p50

Die Prozeßperipherie bildet zwischen Rechner und Prozeß jeweils eine Kette zur Erfassung von Meßgrößen als Eingabedaten und zur Erzeugung von Stellgrößen aus den Ausgabedaten des Rechners.
Die wichtigsten Glieder in diesen Ketten bewirken Umsetzungen zwischen digitalen und analogen Werten (D/A) sowie zwischen elektrischen und nichtelektrischen physikalischen Größen (E/P). Auch Umwandlungen zwischen verschiedenen physikalischen Größen kommen vor (P/P).

5.2.1 Meßglieder, -wandler, und -geräte

5.2.1.1 Übersicht

Meßgröße	Meßglied/-prinzip	Meßbereich (-genauigkeit)	Meßergebnis
Ort (x)	Schleifdraht	1cm bis 10m (±0.1 mm)	el. Widerstand
	Codelineal	dito (± 1 Bit)	Digitalwert (opt., el.)
	Lichtschranken	0.1 mm (±0,1 mm)	Binärwert
	Strichgitter	1cm bis 1m (±0,01 mm)	binäre Impulsfolge
	RADAR	1m bis >100 km (± 1 cm)	Laufzeit
Winkel(φ)	Potentiometer	0 bis 360° $<$ (±1°)	el. Widerstand
	Codescheibe	dito (± 1 Bit)	Digitalwert (opt., el.)
Weg (s = Δx)	Dehnungsmeßstreifen	0.1 bis 100 nm	el. Widerstand
	kapazitive und	100 nm bis 1 mm	Kapazität, Frequenz
	induktive Weggeber	100 nm bis 1 m	Induktivität, Frequenz
	Interferometer	100 nm bis 10 m	Helligkeit, Impulsfolge
Dicke (d = s)	Interferometrie:		
(z.B. Draht)	Optische Beugung	500 nm bis 0.5 mm	Helligkeitsmuster
Zeitintervall	Uhr, Zähler	10^{-9} bis 10^{+9}s ($\pm 10^{-12}$)	Impulsfolge
Frequenz (f)	Zähler	1 Hz bis 1 GHz	binäre Impulsfolge
	Teiler	$>$ 1 GHz	Frequenz
	Überlagerung	bis 10^{15}Hz	Frequenz
Geschwindigkeit	$\bar{v} = \Delta s/\Delta t$		
	Dopplereffekt	$<$ c	Frequenzverschiebung
	Tachometer		Drehzahl
Drehzahl	Tachogenerator	0.1 bis 100000 /min	el. Spannung
	Rasterscheiben		Impulsfolge
Beschleunigung	$\bar{a} = \Delta v/\Delta t$		
	Trägheitssensor	0,1 bis 1000 m/s^2	Trägheitskraft $F = m * a$
Masse (m)	Waage	10^{-6}g bis 1000 to	Kraft
Kraft (F)	elast. Verformung	10^{-6}N bis 10^{6}N	Weg, Winkel
	Hydraulik		Druck
Druck	Manometer	10^{-5} Pa bis 10^{10} Pa	Weg, Winkel
(p= F/A)	Piezoeffekt		el. Spannung
Drehmoment M	Hebel (r)		Kraft $F = M/r$
Durchfluß Q	Meßuhr	10^{-6} bis 10^{6} m^3/s	Drehzahl
	Schwebekörper		Weg
	Staudruckmesser		Druckdifferenz
Dichte	ρ = m / V		
	Auftriebskörper		Kraft, Weg
Feuchte	Hygroskop.Material	0 - 100 %	el. Widerstand
	Dehnung	dito	Weg
	Verdunstung	dito	Temperaturdifferenz
Temperatur	Ausdehung	-270 bis 1000 °C	Weg
	Thermoelement	-273 bis 2000 °C	el. Spannung
	Widerstand (z.B. NTC)	-273 bis 2000 °C	el. Widerstand
Helligkeit	Photoelement	10^{-8} bis 10^{6} W/m^2	el. Spannung
(Intensität)	Photowiderstand, -zelle	10^{-6} bis 10^{6} W/m^2	el. Widerstand
	Photodiode,-transistor	dito	el. Widerstand
	Photonenzähler	$< 10^{-7} W/m^2$	Impulsfolge
Farbe	Spektrometer		Ort, Helligkeit
Stoffzusammen-	Chromatograph		Ort, Farbe,
setzung	Spektrograph		Geschwindigkeit

5.2.1.2 Ausgewählte Verfahren

a) Ortsmessungen

– **Schleifdraht** (Potentiometer) . .
Die abgegriffene Spannung U_x ist genau proportional zum Abstand x:

$$U_x = U_{Ref} \cdot x/l$$

Diese Spannung kann an einen Analogeingang (AI) gelegt werden.

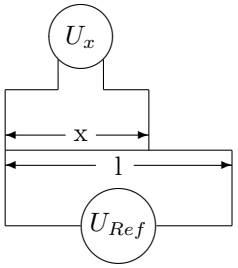


Bild n3p51

– **Rasterlineal** (Strichgitter) ..
Umwandlung (Diskretisierung) der analogen Größe in eine digitale schon auf der physikalischen Ebene.
Bei der Abtastung wird eine (serielle) Bitfolge erzeugt, die von einem Zähler gezählt und auf einen Digitaleingang (DI) gegeben werden kann.

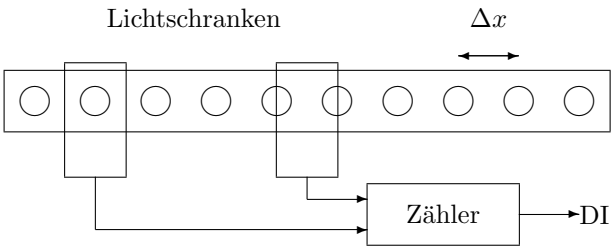


Bild n3p52

Zur Erkennung von Vorwärts- und Rückwärtsbewegungen müssen 2 Sensoren im Abstand $d = (2n + 1) \cdot \Delta x/2$ („auf Lücke“) verwendet werden.

– **Codelineal**

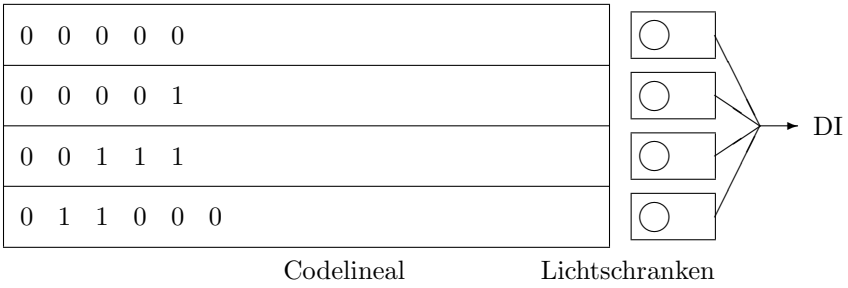


Bild n3p53

Diskretisierung und Digitalisierung des Orts.
Zur Darstellung von ”0” und ”1” können schwarze und weiße Felder bei Reflektionsabtastung verwendet werden, und Löcher für die ”1”-en bei Transmissionsabtastung.
Bei der Abtastung werden n bit breite Digitalwerte erzeugt.
Zur Verringerung der Störanfälligkeit wird oft der Gray-Code verwendet, bei dem sich von Feld zu Feld immer nur 1 Bit ändert. Zur Umwandlung in eine Dualzahl ist ein Decoder notwendig.

b) Geschwindigkeitsmessungen
- Durchschnittsgeschwindigkeit $\bar{v} = \Delta x/\Delta t$
- Momentangeschwindigkeit:
  Tachometer, Tachogenerator: $U_{ind} = C \cdot v$ (Induktionsgesetz)
- Doppler-Radar

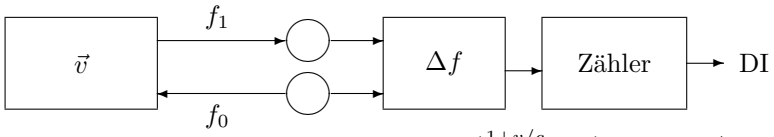


Bild n3p54 Frequenzverschiebung: $\Delta f = f_0(\frac{1+v/c}{1-v/c} - 1) \approx f_0 \cdot 2v/c$

($c = 3 \cdot 10^8 m/s$ ist die Lichtgeschwindigkeit, die stets sehr viel größer ist als v)

c) Beschleunigungsmessungen
- Durchschnittsbeschleunigung: $\bar{a} = \Delta v/\Delta t$
- Trägheitssensor: $\vec{F} = m \cdot \vec{a}$

d) Kraftmessungen
- Federwaage: $\vec{F} = D \cdot \vec{y}$
- Piezo-Effekt: $U = k \cdot A$ $p = k \cdot F$

e) Temperaturmessungen
- Thermische Ausdehnung: $\Delta l = \alpha \cdot \Delta T$ (bis ca. $300°C$)
- Thermospannung: $\Delta U = k \cdot \Delta T$ (bis ca. $1000°C$)
- Strahlungsintensität: $I \sim T^4$ (bis ca. $8000°C$)

5.2.1.3 Fehler

Die einzelnen Verfahren sind jeweils mit Fehlern behaftet, die sich in der Mekette fortpflanzen. Dies muß bei der Auswertung berücksichtigt werden.

5.2.2 Stellglieder, -wandler, und -geräte

5.2.2.1 Übersicht

Stellglied	Stellgröße	Eigenschaft	Steuergröße
Elektromotor	Ort Geschwindigkeit Beschleunigung Drehmoment Kraft Winkel Drehzahl	stetig	el. Leistung
Schrittmotor	dito	diskret	Impulszahl
Elektromagnet	Ort, Weg	diskret	el. Leistung
Stellventil	Durchfluß	stetig	Elektromotor
Drossel	dito		
Sperrventil	Durchfluß	diskret	Elektromagnet
Schieber	Transportfluß	stetig	Elektromotor
Weiche	Transportrichtung	diskret	Elektromotor
Pumpe	Durchfluß	stetig	Elektromotor
Verdichter	dito		
Heizung	Temperatur	stetig	el. Leistung
Elektroden	Stoffeigenschaften	stetig	el. Strom / Spannung
Relais,Schütz	el. Strom	diskret	el. Leistung
Transistor,	el. Strom	stetig	el. Spannung
Thyristor	el. Strom	diskret	el. Spannung
el. Widerstand	el. Strom	stetig	Widerstandswert
Potentiometer	el. Widerstand	stetig	Winkel, Weg
Stelltransformator	el. Spannung	stetig	Winkel, Weg
Lampe	Helligkeit	stetig	el. Leistung

5.2.2.2 Ausgewählte Verfahren

a) Motoren (für x, v, a)
- analog: $v = r \cdot n$ mit r = Radius, n = Drehzahl $\sim P$ = el. Leistung
- digital: (Schrittmotor): $v \sim f_p$ mit f_p= Impulsfolgefrequenz. Ähnlich wie beim Strichgitter müssen stets 2 (orthogonale) Impulsfolgen erzeugt werden, um eine Drehrichtung (links/rechts) festzulegen.

b) Ventile (für Durchfluß und Richtung)
- Sperrventile (diskret, auf/zu)
- Drosselventile (kontinuierlich)
- Weiche (Richtung)

c) Heizung
- elektrische Widerstandsheizung:
  Temperaturerhöhung ΔT durch elektrische Leistung $P = U \cdot I$ während einer Zeitspanne Δt

  $P \cdot \Delta t = \Delta Q = c \cdot \Delta T$

5.2.3 Übertragungsglieder

a) Leitungen
- Freie Leitung, R, C, L
- Verdrillte Leitungen, R, C, L, $R = \sqrt{L/C}$
- Coaxialleitung, Wellenwiderstand R, Dämpfung $\delta(\omega)$
- Lichtwellenleiter, Dämpfung $\delta(\lambda)$, Dispersion $n(\lambda)$
- Funk und Ultraschall, Reflexionen

b) Koppelglieder
- Galvanische Trennung durch
  - Optokoppler
  - Transformatoren
- Frequenztrennung durch
  - Filter
  - Hoch-, Tief-, Bandpässe
  - Weichen

c) Umformer
- Verstärker, Abschwächer
- Diskriminatoren, Begrenzer
- Schmitt-Trigger (Hysterese)

Inhaltsverzeichnis