1.4.6 Massiv parallele Computersysteme

Am Beispiel der CM2:

Technische Daten:

SIMD
$\leq$ 64 K Prozessoren (*Bit*-slice)
$\leq$ 10 GFlops
$\leq$ 256 Kbit/Prozessor (d.h. $\leq$ 2 GB) Hauptspeicher
pro 32 Prozessoren je 1 Gleitkommaprozessor,
d.h. $\leq$ 2K Gleitkommaprozessoren à 32 Bit / 64 Bit
$\leq$ 60 GB Data Vault

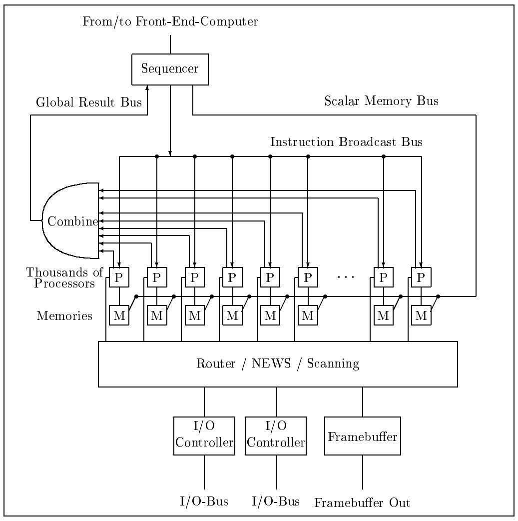
Hier in Wuppertal (zur Zeit: 4 K Prozessoren, kein Data Vault):

8 K Prozessoren, d.h. 256 Gleitkommaprozessoren à 64 Bit
8 K $\cdot$ 256 KBit = 256 MB Hauptspeicher
$\leq$ 1,25 GFlops
10 GB Data Vault

Bemerkungen:

- „message passing“ durch Nachrichten über NEWS
- 12-dim Hypercube
- Datavault: $\underbrace{\underbrace{32}_{\text{Bits/Wort}} + \underbrace{7}_{\text{error correction}} + \underbrace{3}_{\text{Reserve}}}_{\text{„fault tolerant disc system“}} = 42\ HD's$
- Framebuffer: für je 32 Prozessoren existiert 1 Leitung zum Display = 256 Leitungen bei 8 K Prozessoren

Abbildung 1.22: Schaltschema der CM2



Am Beispiel der CM5 (CM5E-LD):

- Zur CM2 kompatible SIMD-Maschine:

  „Vektor“-Operationen:

```
shape [16384]t;
int:t p1,p2,p3;
with (t)
     p3 = p2 + p1
```

- MIMD-Modus: 32-Prozessor-System (hier in Wuppertal)

  32 Prozessoren mit je 4 VU's (Vektorunits à 64 Bit) und einem SuperSPARC-Knoten (40 MHz) für Skalaroperationen,
  $\leq$ 5 GFlops peak-Leistung (realistisch: $\sim$40%),
  1 GB Hauptspeicher (32 MB pro PN – processing node –)
  24GB scalable disk array (SDA)

message passing (broadcast, …) und verschiedene Instruktionsströme für verschiedene Prozessoren

Probleme bei SIMD und MIMD:

Engpaß bei SIMD:

```
with(t)
     where (B(p1))
          ...
     elsewhere
          ...
```

← Gesamtausführungszeit
= Zeit_für_where + Zeit_für_elsewhere

Problem bei MIMD:

Synchronisation!

Als Ausweg ein Kompromiß:

SPMD (= „same program multiple data“) — „data parallel programming“

- außer in where/elsewhere-Zweigen reines SIMD
- where- und elsewhere-Zweige laufen parallel zur selben Zeit
- where/elsewhere-Zweige werden gleichzeitig gestartet
- automatische Synchronisation am Ende von where/elsewhere
- zum Teil wird auch dasselbe Programm auf alle Knoten verteilt, wo es dann jeweils zu einem Zeitpunkt an verschiednen Stellen der Befehlsausführung angekommen sein kann (unterschiedliche Fallunterscheidungszweige …)

Aktuelle massiv parallele Computersysteme sind Z.B. die Cray-T3E und die NEC SX-4 (vergleiche Abschnitt 1.4.3).

1.4.7 Anwendung: Die massiv parallele Simulation

Die massiv parallele Simulation stellt ein leistungsfähiges Tool zu

Analyse, Verifikation, Optimierung und Erforschung

komplexer Systeme in wirtschaftlichen/sozialwirtschaftlichen/... Disziplinen dar; sie ermöglicht Experimente. Beispiele:

- Prognosen für komplexe ökonomische Systeme,
- Optimierung von Fertigungsanlagen, Kraftwerken, Produktionsstätten, ...
- nichtlineare dynamische Systeme als Modelle
  (Hier sind analytische Methoden häufig zu aufwendig oder nicht existent.)
  ↓ chaotische dynamische Systeme

  Lit.: H.-W. Lorenz
  Simulation komplexen Systemverhaltens in der nichtlinearen dynamischen Wirtschaftstheorie,
  Simulationstechnik — 7. Symposium in Hagen, 9/91, Bd.4, S.104
- Lager- und Montagesysteme
- Werkzeugsteuerung
- Prozeßleitsteuerung
- Betriebsleitebene von Kraftwerken (Simulation von Störfällen,...), auch als Lehrinstrument ohne materielle Gefahren

Komplexe Systeme sind meist aus Teilsystemen zusammensetzbar, die einzelnen Prozessoren zugeordnet werden können und deshalb nicht schwierig zu parallelisieren sind.

1.4.8 Klassifikation von parallelen Rechnern

Datenstrom \ Befehlsstrom	ein	mehrere	
ein	SISD (=von Neumann)	MISD	← ?
mehrere	SIMD	MIMD	← CM5
	↑ CM2		

	# Computer \ Memory	unshared	shared
	ein Computer	Nachrichtenaustausch (message passing)	shared memory
vgl. Kapitel 3 ←	n Computer	Message Passing in distributed computing	distributed virtual memory
			↑ LINDA, ...

	Speicher\Kommunikation	gemeinsame Variablen	Nachrichtentausch
global memory →	globaler Speicher	$\text{GM}\underbrace{\text{SV}}_{\text{shared variables}}$	$\text{GM}\underbrace{\text{MP}}_{\text{message passing}}$
distributed memory →	verteilter Speicher	DMSV	DMMP
			↑ CM5

1.5 Speichermodule (SIMM's, ...)

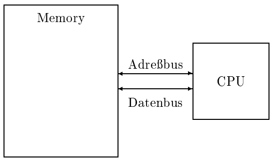
S = single
I = inline
M = memory
M = module

Zum Aussehen und zum Einbau von SIMM's vergleichen Sie bitte den Abschnitt über SIMM's in Kapitel 1.3 „*Field replaceable units*".

...

1.6 Assoziative Speicher

Die sequentielle Adressierung als *Flaschenhals*:



Probleme bei der Suche nach assoziierten Daten mit Hilfe von „von-Neumann Rechnern":

- Suche nach der Telefonnummer von „Meyer, A.":

  a) sequentielle Suche:

Adresse	Name	Nummer	
⋮	⋮	⋮	...
1012	Meyer, A.	237704	
1112	Adler, K.	1244	
1212	Schulte, B.	124466	
⋮	⋮	⋮	...

```
   ⋮
ResetListe;
Found := false;
Repeat
  GetNextRecord(R);
  if R.Name = 'Meyer,A.' then
    Found := true;
until (Found = true) or EndofListe;
   ⋮
```

  Zeitaufwendig, da im schlimmsten Falle alle Listenelemente ausprobiert werden müssen!

  b) Im Falle eines nach dem Schlüsselfeld alphabetisch geordneten Datenbestandes: binäre Suche

55 > 8:

1	...
3	...
7	...
8	...
12	...
55	...
101	...

55 > 12:

8	...
12	...
55	...
101	...

55 = 55:

12	...
55	...
101	...

```
    ⋮
ResetGesamteListe;
Found := false;
Repeat
  GetMiddleListElement(R);
  if R.Name = 'Meyer,A.' then
    Found := true;
  else if 'Meyer,A.' < R.Name then
    Liste := erste Halbliste
  else
    Liste := zweite Halbliste
until (Found = true) or (Liste = leer);
```

In einem sequentiell adressierten Computer ist die Anweisung

`Liste := zweite Halbliste`

nur mit zusätzlichem Verwaltungsaufwand effektiv realisierbar (*Suchbaum*).

Alternative: „Finde **per Hardware** alle *Listenelemente* mit bestimmten Eigenschaften" in nur „einer" Taktperiode — die assoziative Adressierung:

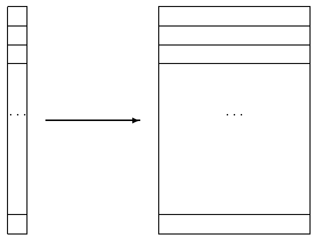
"Meyer,A."
↕ statt einer Adresse wird der „Schlüssel" direkt benutzt

⋮	
Meyer,A.	237704
⋮	

↕ Datenbus (erster, nächster Match)
"237704"

Probleme:

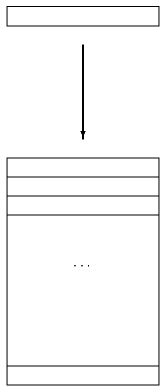
- Schaltungsaufwand
- Inflexibel in der „Schlüssel"-Busbreite und der Datenbus-Breite; deshalb in Worten und Bits zum Teil parallel/zum Teil sequentiell durchgeführt:



Maske Wortbreite
Bitseriell und wortparallel organisierter Assoziativspeicher

In einer Taktperiode wird in einer Serie von Worten dasselbe Bit in all diesen Worten auf eine Übereinstimmung („match") mit der Suchmaske getestet.

Maske



Wortbreite
Wortseriell und bitparallel organisierter Assoziativspeicher

In einer Taktperiode wird in einer Serie von Worten genau ein Wort auf (exakte) Übereinstimmung mit der Suchmaske getestet.

Assoziative Speicher werden im Daten- und Instruktions-Cache benutzt. Daneben finden sie Anwendung in Spezialrechnern („PEPE" von Burroughs beziehungsweise „LUCAS"). Sollten die Probleme zur preisgünstigen Fertigung von großen flexiblen Assoziativspeichern einmal gelöst sein, so ist mit einer Anwendung im Datenbankbereich unbedingt zu rechnen.

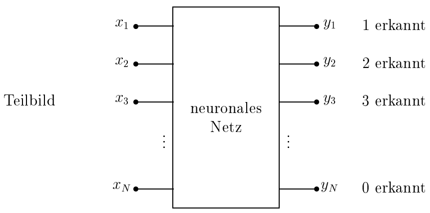
1.7 Neuronale Netze

Komplexe Aufgaben sind zum Teil nicht mehr analytisch (d.h. durch Algorithmen) lösbar bzw. müssen für jede Eingabedatenkonfiguration aufwendig wieder neu gelöst werden.

Beispiel:

Handgeschriebene Ziffern
(größennormierte Darstellung mit einer 16×16 Grauwertmatrix)

Die Erkennung handgeschriebener Ziffern (etwa für die Postleitzahlen-Erkennung) ist analytisch nicht sehr gut beschreibbar. Deshalb wird versucht, ein dem menschlichen Gehirn ähnlich komplexes, „lernendes“ Schaltwerk — ein neuronales Netz — zur Problemlösung einzusetzen.

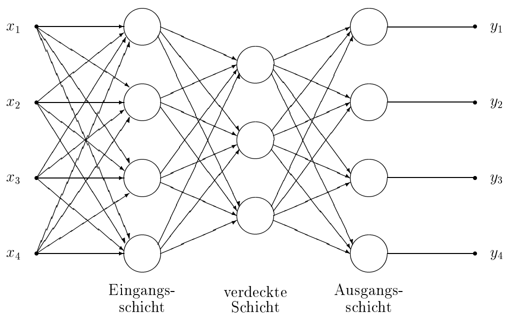


Neuronales Netz mit Ein- und Ausgangsvektoren

Merkmale („künstlicher“) neuronaler Netze:

- *viele* Neuronen
- feste problemspezifische Verschachtelung der Neuronen
- Verbindungen gewichtet (hemmend, fördernd)
- jedes Neuron agiert durch

  ständige gewichtete Aufsummierung aller Eingänge und „Schwellwert“-Ausgabe auf andere Neuronen
- häufig modularer Aufbau: Schichten



Bei einigermaßen komplexen Netzen sind die benötigten Gewichte für die gewünschte Aufgabe nicht mehr berechenbar; deshalb:

Training / Lernen der Gewichte

durch „Rückkopplung“ oder „Backpropagation“.

Vorteile beim Einsatz von neuronalen Netzen:

- Lernfähigkeit (jedoch lange Dauer)
- Generalisierung (erlernt auch das Verhalten für „ähnliche“ Situationen, d.h. auch für nicht gelernte Eingabesituationen werden „sinnvolle“ Ausgaben erzeugt)
- Fehlertoleranz (gegenüber dem Ausfall einzelner Neuronen oder Eingangsdaten)
- Unempfindlichkeit gegen Rauschen

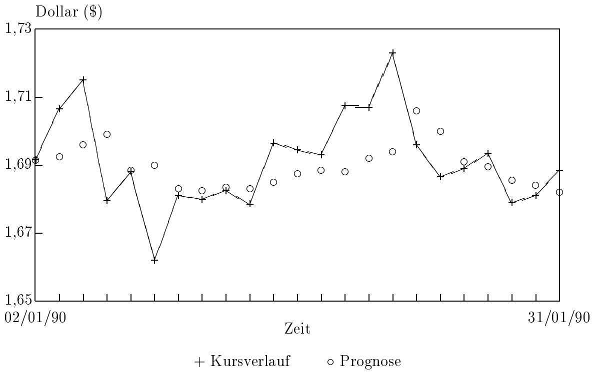
Nachteile:

- Analyse der Klassifikationskriterien unmöglich
- Lernen (= Programmieren) nur durch Training möglich
- das Trainingsprogramm muß auf die zu lernende Aufgabe abgestimmt sein; das Erreichen dieser Abstimmung ist häufig nur unklar zu erkennen
- Lernen sehr zeitaufwendig (dadurch aber keine Analyse der Probleme nötig)
- sehr eingeschränkt einsetzbar, meist Klassifikation, Mustererkennung, Optimierung, Kontrollaufgaben (Robotersteuerung)
- kein universelles neuronales Netz, d.h für jede Aufgabe
  - spezielles Netz
  - spezielle Verbindungsstruktur
  - spezielle Trainingsdaten

In gewissem Rahmen verlangt der Einsatz von Expertensystemen ebenfalls keine analytische Durchdringung des zu lösenden Problems (sondern „nur“ eine Sammlung von „Wissen“ in Form von Fakten und Regeln). Im Gegensatz zu neuronalen Netzen liefert aber der Einsatz von Expertensystemen eine nachvollziehbare Begründung der Problemlösung.

Anwendungen:

Aktienkursprognosen



Prognose des Dollarkurses

- Bei Kursprognosen wurde eine Trefferquote von ∼56% erreicht.
- Mustererkennung beim Lesen von Postleitzahlen.
- Klassifizierung von Gefahrensituationen
- ...

Vergleich: Elektronik – Biologie, künstliche — natürliche Neuronen

	Elektronik	Biologie
Clock-Frequenz	$10^8 s^{-1}$	$10^3 s^{-1}$ Frequenz der Neuronenimpulse
Speicherdichte	10^7 Bit/cm^{-3}	10^{12} Synapsen/cm^{-3}
Verbindungen	10 fan-in, fan-out von digitalen Gattern	10^4 Synapsen pro Neuronen
Anzahl Elemente	10^9 Transistoren	10^{11} Neuronen, 10^{15} Synapsen
Leistungsverbrauch	1 Kilowatt	50 Watt

Eine Auswahl von elektronischen neuronalen Netzen:

		Verbindungen pro Sekunde (x 10^9)	Verbindungen Auflösung	Anzahl Neuronen	Anzahl Verbin-dungen	Anzahl Gewichte	Technologie	Lernen
1	M.Griffin et al. Adaptive Solutions	1,6	9 Bit x 16 Bit fixed point	64	64	16.384	0,8μm CMOS 26,2 mm x 27,5 mm	auf einem Chip verschiedene Algorithmen
2	M.Asai et al. Hitachi	1,5	8 Bit x 8 Bit product	576	576	36.864	0,8 μm CMOS Gate Array, Wafer 5 Inch	nicht auf dem Wafer
3	M.A.Sivilotti et al. Neural Semiconductor	2	4 Bit x 4 Bit product	32	1.024	1.024	?	nicht auf dem Chip
4	H.Graf, D.Henderson AT&T Bell Labs.	80	1 Bit x 4 Bit product flexible	256	8.192	8.192	0,9 μm CMOS 4,5mm x 7 mm	nicht auf dem Chip
5	Y.Arima et al. Mitsubishi Electric	28	1 Bit x 6 Bit product	336	28.000	28.000	1μm CMOS 14,5 mm x 14,5 mm	auf dem Chip Boltzmann
6	R.Allen, J.Alspector and A.Jayakumar, Bellcore	0,1	5 Bit x 5 Bit product	32	992	992	1,25 μm CMOS	auf dem Chip Boltzmann
7	R.Benson et al. Intel Corp.	2	6 Bit x 6 Bit product	64	10.240	10.240	1μm CMOS EEprom	nicht auf dem Chip

Quelle:Weber

Die ersten drei Schaltungen sind digitale Prozessoren, während die folgenden vier zumindest teilweise analoge Schaltungen sind.

1.8 ANHANG: Benchmarks

Relative Rechnersystem-Leistungen sind abhängig von

- der CPU-Leistung,
- der Systemarchitektur (Cache, Bus-Leistung, Hauptspeichergröße und -geschwindigkeit, Swapping-Bereich, Plattengröße und -geschwindigkeit, ...)
- dem Betriebssystem (Art und Version),
- dem Compiler (Sprache, Implementierung und Version, mit/ohne Optimierung, Art der Optimierung, ...)

Deshalb sind Benchmarks nur bei vollständiger Notation der Betriebsumgebung (CPU, Systemarchitektur, Anwendungsprogramm-Typ, Betriebssystem und Compiler) und auch dann nur beschränkt für eine Vorauswahl sinnvoll einsetzbar.

Besser: Test der gewünschten Anwendersoftware unter Einsatzbedingungen.
(Manche Benchmarks – etwa SPEC92 – versuchen einen dedizierten Anwendungs-Mix zu mitteln.)

Beispiele:

- *MIPS* = million instructions per second = $\frac{1}{(\text{cycles per average instruction})\cdot(\text{Zykluszeit in } \mu s)}$
  Beachte: RISC-Mips und CISC-Mips sind unvergleichbar, da die relative Codelänge unterschiedlich ist; die Größe von „cycles per ...“ ist maschinenabhängig.
- *MFLOPS* ähnlich für „million floating point operations per second“.
- *Whetstone* testet relative Laufzeiten für ein Programm mit 106 „Whetstone-Instructions“, die als statistisches Mittel aus 949 Anwenderprogrammen ermittelt wurden (FORTRAN-Anwendungen, Schwerpunkt: numerische Berechnung und Gleitkommaoperationen).
- *Dhrystone* testet ein Programm aus 100 Anweisungen mit 53 Zuweisungstypen, 32 Steueranweisungen, 15 Prozedur- und Funktionsaufrufen (ADA-, C-, Pascal-Anwendungen; Schwerpunkt: Systemprogrammierung).
- *LINPACK-MFLOPS* testen eine Unterprogramm-Sammlung zur „Linearen Algebra“, 100 oder 1000 Gleichungen mit 100 bzw. 1000 Unbekannten in doppelter Gleitkomma-Genauigkeit (FORTRAN-Anwendungen, Schwerpunkt: Vektorrechnen, Supercomputing, Workstations, ...)
- *OLTP* (= OnLine Transactions Processing), Schwerpunkt: System-/Datenbanksoftware, I/O, Kommunikation; TP's = transactions per second.
- *Level 1 BLAS MFLOPS* (= basic linear algebra subroutines, etwa bzgl. saxpy
  — entspricht $A \cdot x + y$ — , ...)
- *SPECint92* mißt das geometrische Mittel von sechs INTEGER-Applikationen (espresso, li, eqntott, compress, sc, gcc) relativ zur Leistung einer VAX 11/780.

- *SPECfp92* mißt das geometrische Mittel von vierzehn Floatingpoint-Applikationen (spice2g6, doduc, mdljdp2, wave5, tomcatv, ora, alvinn, ear, mdljsp2, swm256, su2cor, hydro2d, nasa7, fpppp) relativ zur Leistung einer VAX 11/780.
- *SPECrate int92, SPECrate fp92* mißt ähnlich den relativen Multiprozeß-/Multiprozessordurchsatz (vergangene Zeit, bis mehrere gleichzeitig gestartete Kopien der Benchmark-Programme zu Ende gekommen sind).
- *SDET Peak Throughput, KENBUS1 Peak Throughput, Concurrent Workload* sind relative Größen zum Vergleich des Multiprozeß-/Multiprozessordurchsatzes unter zeitlich ansteigender Last.
- *SPECmark89, SPECint89, SPECfp89, SPECThruput89* sind alte Varianten mit weniger Benchmaks bzw. älteren Versionen dieser Benchmark-Programme.

Auszug aus dem SPEC Newsletter:

Trademark Information:

Here is a list of trademarks mentioned in this newsletter. Other trademarked items that may also appear in this newsletter are trademarks of their respective companies.

UNIX is a registered trademark of Unix Systems Laboratories.

Ethernet is a trademark of Xerox Corporation.

...

SPEC, SPECratio, SPECmark89, SPECint89, SPECfp89, SPECint92, SPECfp92, SPECrate_int92 and SPEC_fp92 are trademarks of the Standard Performance Evaluation Corporation.

...

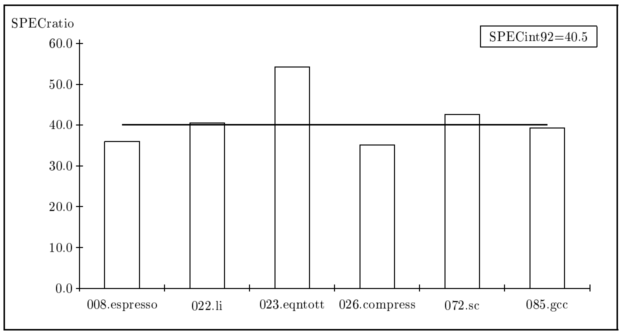
SPEC Member Companies:

- AT&T/NCR
- Auspex Systems, Inc.
- Compaq Computer Corp.
- Control Data Systems Inc.
- Data General Corp.
- Digital Equipment Corp.
- Electronic Data Systems
- Fujitsu Microelectronics
- HaL Computer Systems Inc.
- Hewlett Packard Co.
- IBM Corp.
- Intel Corp.
- Kubota Pacific Computer Inc.
- Intergraph Corp.
- Motorola Inc.
- NeXT Computer Inc.
- Novell Inc.
- Olivetti
- Siemens Nixdorf Informationssysteme AG
- Silicon Graphics Inc.
- Solbourne Computer Inc.
- Sun Microsystems Inc.
- Unisys Corp.

SPEC Associates:

- Leibniz Rechenzentrum, Germany
- Science and Engeneering Research Council (SERC), England
- National Taiwan University, Taiwan
- Center for Scientific Computing (CSC), Finnland

SPEC Benchmark CINT92 Summary					
RESULTS: Benchmark Name	SPEC Reference Time (Sec)	Time (Sec)	SPEC Ratio	IBM Corporation RS/6000 POWERstation 335	
				Hardware	
008.espresso	2270	63.9	35.5	Modell Number	POWERstation 355
022.li	6210	151.4	41.0	CPU	41.67 MHz POWER 4132
023.eqntott	1100	20.8	52.9	FPU	Integrated
026.compress	2770	81.1	34.2	Number of CPU's	1
072.sc	4530	105.8	42.8	Primary Cache	32KBI+32KBD on chip
085.gcc	5460	138.7	39.4	Secondary Cache	None
				Memory	32MB
				Disk Subsystem	1 x 1.0 GB
				Other Hardware	None
				Software	
				O/S & Version	AIX Version 3.2.3
				Compilers & Version	XL C/6000 Version 1.2.1
				Other Software	Kuck & Associates Beta
Geometric Mean: SPECint92			40.5		KAP-C
Hardware Availiability:			Mar-93	File System Type	AIX
Software Availiability:			Dec-92	System State	Multi-user
License #:	11	Tested:	Jan-93	By: IBM	Of: Austin, TX



Kapitel 2

Peripherie

2.1 Scanner, Strichcodeleser, OCR

2.1.1 Historische Entwicklung der Informations-Input-Codes

Historisch entwickelten sich verschiedene Informationscodierungen im Bereich der automatischen Datenverarbeitung:

- Lochkarten/-streifen
- TÜV-Formulare, ... (Markierungen: 1 aus N)
- Strichcodes (Barcodes) auf Waren, Kassenerfassung (Markierungen):

  EAN = europäische Artikel-Nummer, DIN 66236:

13-stelliger Dezimalcode			
1–2	3–7	8–12	13
Landnr.	Betriebsnr.	Artikelnr.	Prüfziffer

- Magnetcodes auf Kreditkarten, Telefonkarten, ... (Codes):

  „passiv, 3 Spuren, DIN 9785“, z.B. Kreditkarten

  oder fälschungssicherer

  „aktiv (Chips)“, z.B. Telefonkarten

- OCR = „optical character recognition“

  $\left\{\begin{array}{l}\text{OCR-A}\\ \text{OCR-B}\\ \text{Maschinenschrift}\end{array}\right.$ mit Drucker, TeX,... erzeugbar

- Klarschrift normiert
- Klarschrift handschriftlich (Blockschrift)
- Schreibschrift (Erkennung durch neuronale Netze, ...)

2.1.2 Strichcodeleser

Strichcodeleser (z.B. in Bibliotheken zur Signatur-Erfassung genutzt) führen die optische Abtastung durch mechanische Führung (Bewegung durch eine Bedienungsperson) über die Vorlage aus. Dazu ist eine angemessene gleichmäßige Bewegung notwendig!
Es gibt (PC-)Tastaturen mit integriertem Anschluß für Strichcode-Lesestifte, Laser-Scanner-Pistolen, Schlitz-Leser und Durchzugs-Leser. Diese liefern die entsprechenden Zeichencodes an den Personal-Computer, als ob sie per Tastatur eingegeben worden wären (es ist also keine spezielle Eingabegeräte-Schnittstelle der benutzten Software nötig).

Typische Einsatzgebiete sind:

1. Produktidentifizierung bei Warenein- und -ausgang
2. Datenerfassung im Produktionsablauf
3. Betriebsdaten-Erfassung
4. Inventur-Eingabe
5. Kassenterminal im Handel
6. Arzneimittelverwaltung
7. ...

Bei der Auswahl von Strichcodeleser-Hardware ist insbesondere auf eine automatische Erkennung der gängigen Strichcodes zu achten: Code 3 of 9, Code 3 of 9 Extended, Code 128, Interleaved 2 of 5, Industrial 2 of 5, Codebar, Code 93, UPC A, UPC E, EAN/JAN 8, EAN/JAN 13, Plessy, ASCII, ...

2.1.3 Scanner

Bei Scannern geschieht die optische Abtastung

- entweder wie beim Strichcodeleser oder
- ganzseitig wie beim Kopierer (d.h. keine Bewegung nötig).

Als Ergebnis ist

- Pixelgraphik oder
- mit Hilfe von OCR-Software auch Schrift (ASCII, Textverarbeitungsformate mit Fettschrift, verschiedenen Schriftgrößen, verschiedenen Schriftarten, ...)

möglich. Alte OCR-Systeme sind nur geeignet für

- Normschrift,
- feste Lage (Drehproblem) der Vorlage.

Neuere OCR-Systeme

- erkennen Texte auch lage- und verzerrungsunabhängig durch Nutzung geometrischer Invarianten statt durch bloßes Pixel-Matching,
- sind lernfähig (d.h. Hinzulernen besonderer Zeichen oder Symbole: Logos, griechische Buchstaben, mathematische Symbole, „verschmierte“ Zeichen, ...).

Fax-Geräte können als Scanner „geringer“ Auflösung genutzt werden. Kopierer mit Computerausgang können als Scanner genutzt werden, solche mit Computereingang als Drucker.

Schriften OCR-A und OCR-B

Strichcodes

OCR Prüfstand

Erkennungsgüte und Erkennungszeiten

	c't-Text		Listing		Stern-Text	
Differenz der Wortzahlen						
Gesamtzahl	670		631		671	
	absolut	relativ	absolut	relativ	absolut	relativ
Optopus*	46	6,87%	136	21,55%	96	14,31%
Omnipage	47	7,01%	40	6,34%	38	5,66%
WordScan	5	0,75%	45	7,13%	35	5,22%
Perceive	48	7,16%	631	100%	53	7,90%
Catchword	21	3,13%	294	46,59%	117	17,44%
GO-CR	50	7,46%	169	26,78%	52	7,75%
Recognita	5	0,75%	371	58,80%	38	5,66%
Readiris	48	7,16%	411	65,13%	75	11,18%
Differenz verschiedener Wörter						
	c't-Text		Listing		Stern-Text	
Gesamtzahl	457		229		418	
	absolut	relativ	absolut[1]	relativ[1]	absolut	relativ
Optopus*	9	1,97%	83	36,24%	74	17,70%
Omnipage	65	14,22%	96	41,92%	50	11,96%
WordScan	31	6,78%	113	49,34%	61	14,59%
Perceive	93	20,35%	229	100%	77	18,42%
Catchword	23	5,03%	237	103,5%	85	20,33%
GO-CR	52	11,38%	237	103,5%	88	21,05%
Recognita	36	7,88%	289	126,2%	70	16,75%
Readiris	53	11,60%	27	11,79%	75	17,94%

* Höhere Werte als die Gesamtzahl ergeben sich durch zerrissene Wörter.

Die *Differenz der Wortzahlen* ergibt sich als Differenz der Gesamtzahl der Worte in der Vorlage und Gesamtzahl der Worte im gescannten Text (Ursache der Differenz: Zeichenfehler werden i.a. durch Benutzung eines Wörterbuches „berichtigt"; Leerräume in Proportionalschrift können fälschlich als Wortende interpretiert werden, ...). Beim heutigen Stand der Technik kann man davon ausgehen, durchschnittlich 10% der gescannten Worte nachbearbeiten zu müssen.

Zeichensätze:

HP Roman8 Zeichensatz (insbesondere die ersten acht Spalten: 7 Bit ACSII):

Landesspezifische Varianten durch Zeichenersatz (ISO-Austauschtabelle):

(„Industriestandard“) PC-8 Zeichensatz:

Windows 3.0 Zeichensatz:

ISO-8859 Latin 1 (ECMA-94 Latin 1) Zeichensatz:

Eingabe von Sonderzeichen / Tastaturen:

Alt- beziehungsweise Compose-Tastensequenzen:

Pen-Computer als elektronisches Notizbuch:
etwa bei der Arztvisite, bei Verwaltungsaufgaben in Firmen, ...
DIN-A5 oder DIN-A4 Notepad, Eingabe mit Stift = $\boxed{\text{Pentops}}$

- mit Handschrifterkennung statt Tastatur
- PCMCIA-Standard 1 Speicherkarten (85,6x59x3,3 mm, 68 Kontakte, 64 MB RAM)
- PCMCIA-Standard 2: I/O, Modem, LAN, Funkmodems, Festplatten, ...

Typ I		3,3 mm hoch
Typ II		5 mm hoch
Typ III		10,5 mm hoch
Typ IV	(etwa für Festplatten)	12,5...16mm hoch

mobiles Computing:
Notebooks (DIN-A4, 2 ... 4 kg)
Subnotebooks (DIN-A5, 1,5... 2 kg)
Palmtops (DIN-A5, bis 1 kg)
Palmpad-Organizer (Größe einer Videokassette, elektronischer Terminplaner)
Notepads (Größe einer Videokassette)
PDAs (Personal Digital Assistant, Schreibblockgröße)