technische universität
dortmund

Rechnerstrukturen

Gernot A. Fink und Peter Marwedel

TU Dortmund, Fakultät für Informatik
WS 2011/12

- Einleitung
  - Historische Entwicklung von Rechnern
  - Einordnung
- Repräsentation von Daten
  - Repräsentation natürlicher Zahlen
- Organisatorisches
  - Vorlesung
  - Übungen
  - Gruppeneinteilung

Historische Entwicklung von Rechnern

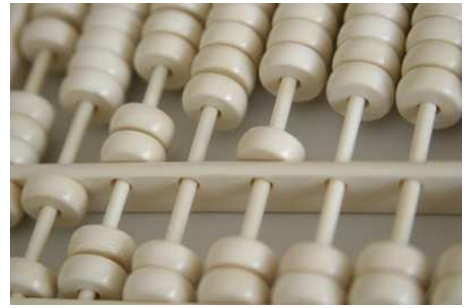
Zielsetzung:

ursprünglich immer Automatisierung von Berechnungen ("schneller", "fehlerfreier") in der Verwaltung, beim Militär, in der Wissenschaft

→ siehe auch Heinz Nixdorf MuseumsForum, Paderborn (http://www.hnf.de/)

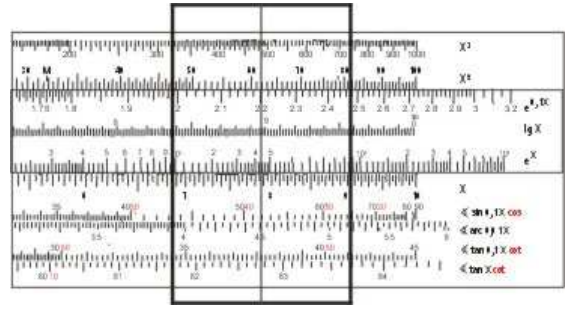
Rechenhilfsmittel

- Abakus (ca. 200 n.Chr. In China)



Quelle: Arithmeum, Bonn

- Rechenschieber (ca. 1620)



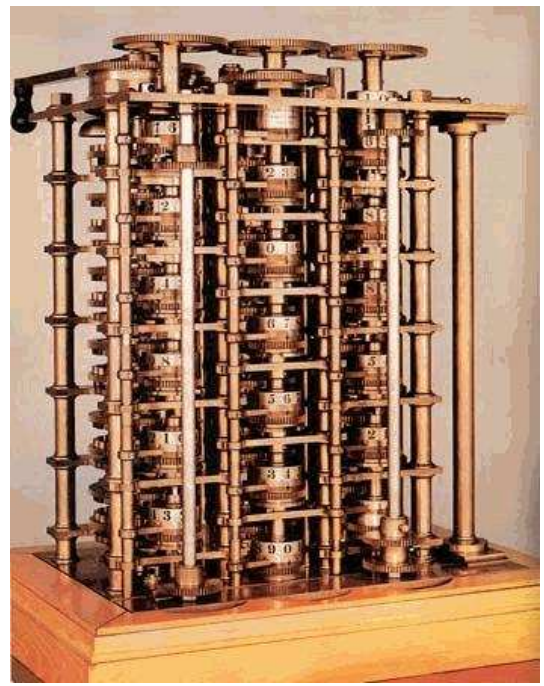
Historische Entwicklung von Rechnern II

Mechanische Rechenmaschinen (1642-1945)

- *Blaise Pascal* (1623–1662)
  Erste funktionsfähige Rechenmaschine für Addition/Subtraktion
- *Gottfried Wilhelm Leibniz* (1646–1716)
  Rechenmaschine für 4 Grundrechenarten, Zahlen in Binärdarstellung

Quelle: Deutsches Museum, München

- *Charles Babbage* (1791–1871)
  Difference Engine, Analytical Engine
  (beide nie fertiggestellt)



Quelle: Univ. of Alabama

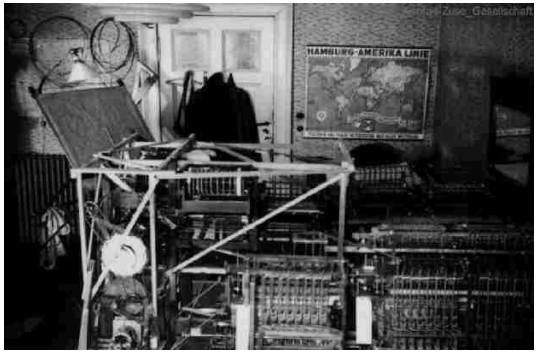
Historische Entwicklung von Rechnern III

... mechanische Rechenmaschinen (1642-1945)

- *Herman Hollerith* (1860-1929)
  Lochkarten zur Datenspeicherung
  (erstmals um 1800 zur Steuerung mechanischer Webstühle)

Quelle: HNF, Paderborn

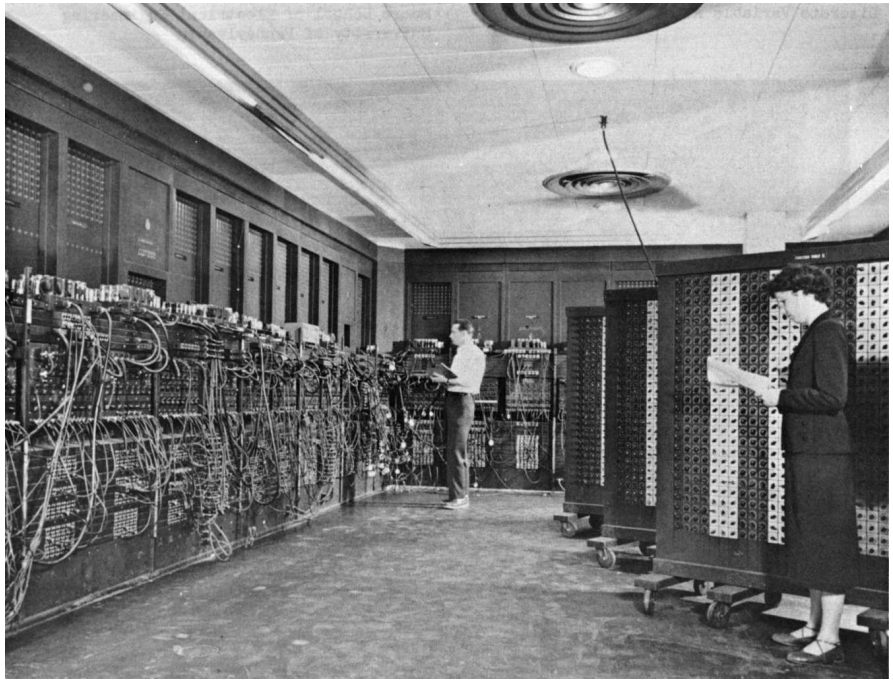
- *Konrad Zuse* (1910-1995)
  Z1 (1937/38): erster programmgesteuerter Rechenautomat



Historische Entwicklung von Rechnern IV

Die 1. Generation: Vakuumröhren (1945-1955)

- *COLOSSUS* (1943)
  unter Mitwirkung von Alan Turing: Britisches Geheimprojekt zur Entschlüsselung des ENIGMA-Codes
- *ENIAC* (1946): erster elektronischer Universalrechner
  18000 Röhren, 1500 Relais, 30t, 140kW



Quelle: worldpress.com

Historische Entwicklung von Rechnern V

... die 1. Generation: Vakuumröhren (1945-1955)

- *John von Neumann*
  Konzeption eines speicherprogrammierbaren Digitalrechners
  Speicher, Rechen-, Steuer- und E/A-Einheiten
- *IBM 701* (1953): erster wissenschaftlicher Rechner

Quelle: IBM

Historische Entwicklung von Rechnern VI

Die 2. Generation: Transistoren (1955-1965)

- TX-0 als erster transistor-basierter Rechner am M.I.T. entwickelt
- Digital Equipment Corporation gegründet (1957):

  PDP-1: erster Minicomputer
  PDP-8: verwendet erstmals ein Bus-System

- Control Data Corporation (CDC, *Seymour Cray*) Modell 6600 (1964): erster Supercomputer
  10-mal schneller als IBM-Top-Modell, parallele Funktionseinheiten,

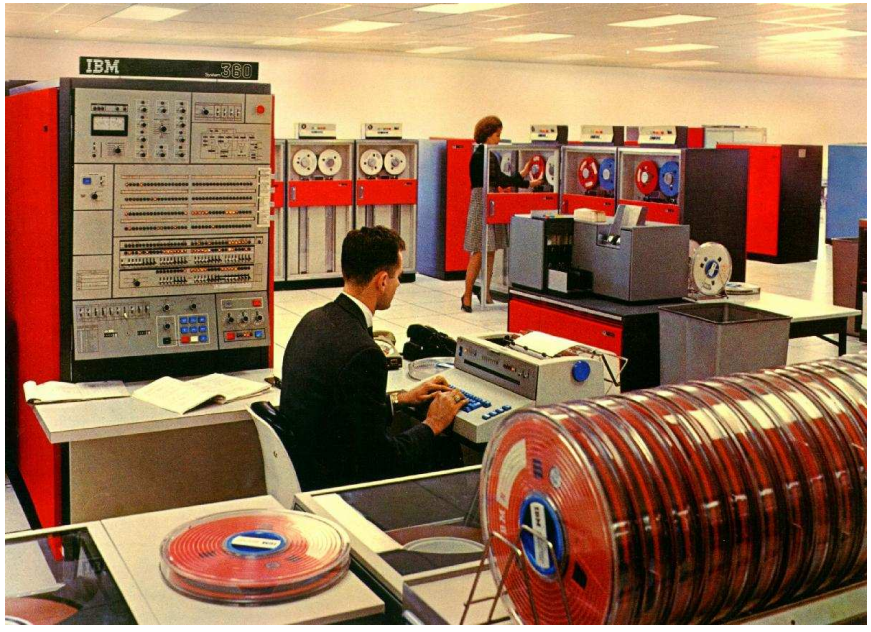
  speziell für number crunching
- Burroughs B5000: Rechner speziell für ALGOL 60

CDC 6600

Historische Entwicklung von Rechnern VII

Die 3. Generation: Integrierte Schaltungen (1965-1980)

- IBM führt mit System/360 eine einzige Produktlinie ein
  Mehrprogrammbetrieb, Emulation anderer Rechner
- PDP-11: erfolgreichster Minicomputer im wissenschaftlichen Bereich



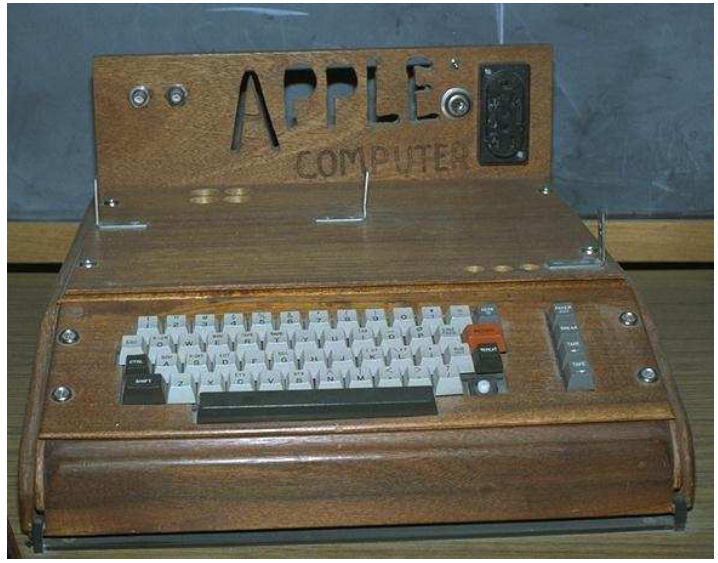
IBM 360 (Quelle: Univ. of Nottinhamn)

Digital Equipment PDP-11

Historische Entwicklung von Rechnern VIII

Die 4. Generation: VLSI-Integration (seit 1980)

höhere Integrationsdichte macht persönliche Rechner möglich



Quelle: Smithsonian Museum

- Erster PC als Bausatz auf Basis des Intel 8080
- CP/M Betriebssystem für 8080-Rechner (diskettenbasiert)
- Steve Jobs & Steve Wozniak bauen den Apple
- IBM-PC auf Intel 8088-Basis mit Standardkomponenten (1981)

  offener, dokumentierter Standard wird von anderen Firmen geklont, Betriebssystem MS-DOS [Rechte verbleiben bei Microsoft]
- Mitte der 80er Jahre: RISC-Rechner entstehen (SPARC, MIPS)

  extrem kleiner Instruktionsvorrat, Mikroprogrammebene wird abgeschafft

Historische Entwicklung von Rechnern IX

Pervasive / Ubiquitous Computing (heute)

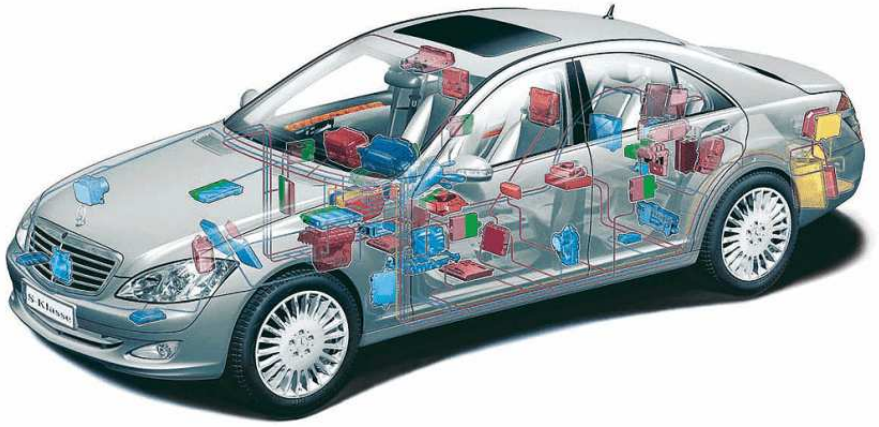
extreme Integrationsdichte ermöglicht die Einbettung von Rechnern in quasi beliebige Artefakte des täglichen Lebens

- Mobiltelephone und persönliche Assistenten

Quelle: websites.am

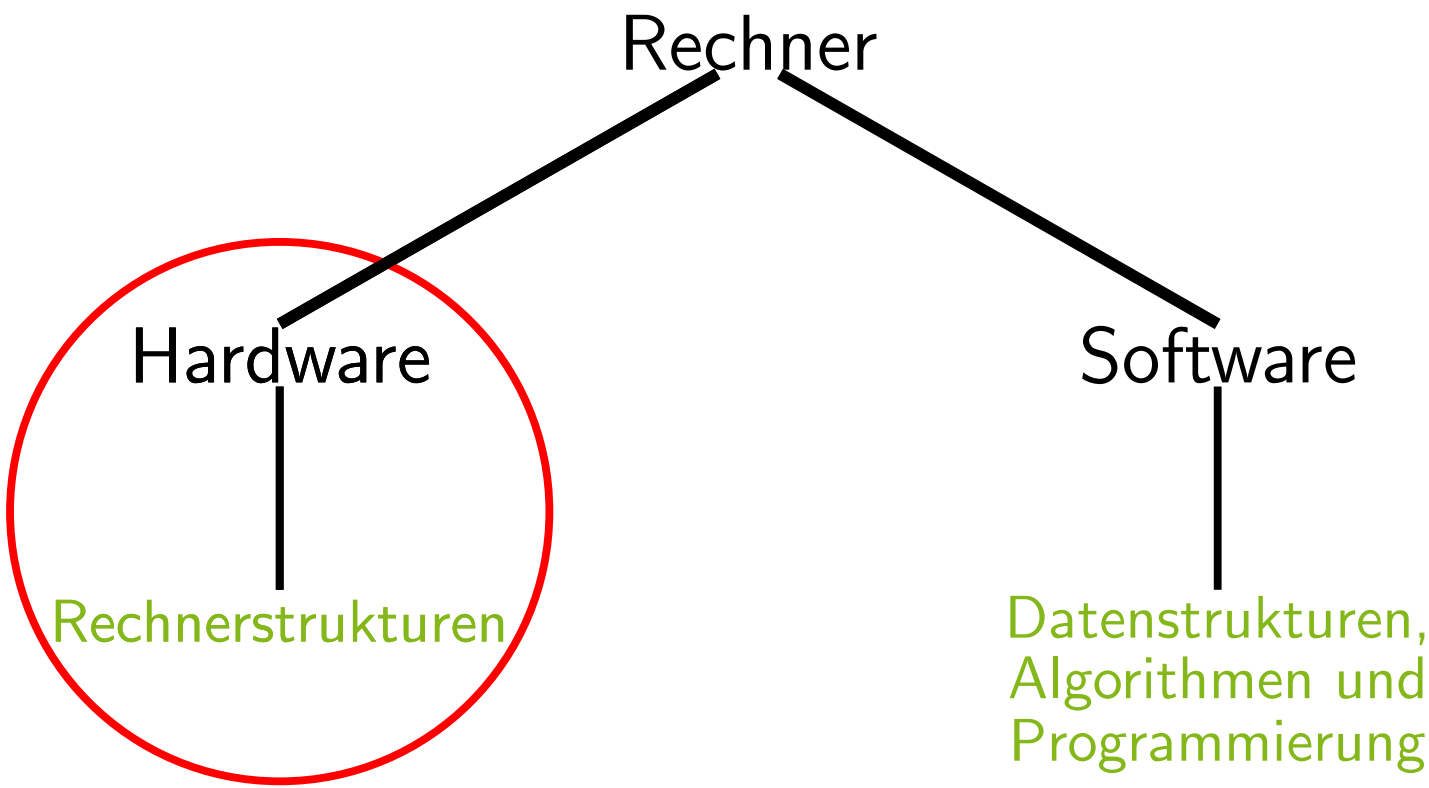
- Kraftfahrzeuge

  ca. 100 Steuergeräte [= Rechner] in Oberklasse-Fahrzeug

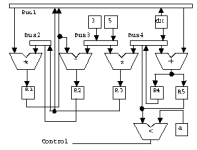
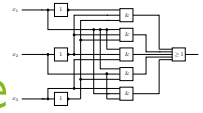
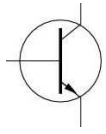
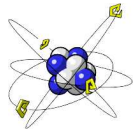


Quelle: freenet.de

Rechner


Rechner
Hardware
Software
Rechnerstrukturen
Datenstrukturen,
Algorithmen und
Programmierung

Hardware: Abstraktionsebenen

5	Programmiersprachenebene (z. B. Java, C++)	DAP 1
4	Assemblerprogrammebene (z. B. mov ebx, d00ah)	RS
3	Mikroarchitekturebene	RS
2	Registertransferebene 	RS
1	digital-logische Ebene 	RS
0	Transistor-Ebene 	ET& NRT E-Technik
−1	physikalische Ebene 	Physik

Informatik

Repräsentation von Daten

Erinnerung: Moderne Rechner arbeiten binär ($\rightarrow$ *Digital*rechner)

Grund:

- technisch einfacher zu realisieren
- damit billiger zu realisieren
- bieten wohl-definierte Genauigkeit
- sind leichter zu programmieren

Problem: Wie repräsentiert man mit nur zwei Grundzuständen ...

- Zahlen? ... natürliche, ganze, reelle Zahlen?
- Texte?
- andere Daten?

Darstellung natürlicher Zahlen

Braucht man für unendlich viele Zahlen unendliche viele Symbole?

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, ...

Ziffern $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ reichen aus.

Stellenwertsystem

$$13072 = 2 \cdot 1 + 7 \cdot 10 + 0 \cdot 100 + 3 \cdot 1000 + 1 \cdot 10000$$

$10\,000$ $= 10^4$	$1\,000$ $= 10^3$	100 $= 10^2$	10 $= 10^1$	1 $= 10^0$
1	3	0	7	2

Stellenwertsysteme

10 Ziffern $\{0,1,2,3,4,5,6,7,8,9\}$ — **Basis** 10

Muss das so sein?

Behauptung Es geht mit jeder Basis $b \in \mathbb{N} \setminus \{1\}$.

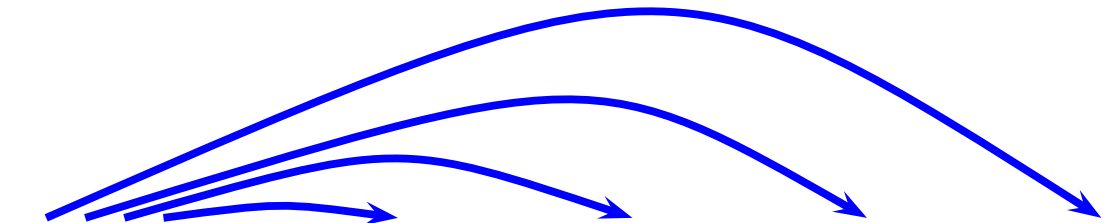
Basis b „Ziffern“ $\{0,1,\ldots,b-1\}$

Schreibweise $(z_{l-1}\, z_{l-2}\, \cdots\, z_1\, z_0)_b$

Beispiel $(1\,3\,0\,7\,2)_{10} = 13072$

bei Basis b $>$ 10 manchmal „Ziffern“ A für 10, B für 11, ...

Umrechnung Basis $b \rightsquigarrow$ Basis 10



$$(3\,7\,4\,8)_{10} = 8 \cdot 10^0 + 4 \cdot 10^1 + 7 \cdot 10^2 + 3 \cdot 10^3$$

$$= 8 \cdot 1 + 4 \cdot 10 + 7 \cdot 100 + 3 \cdot 1000$$

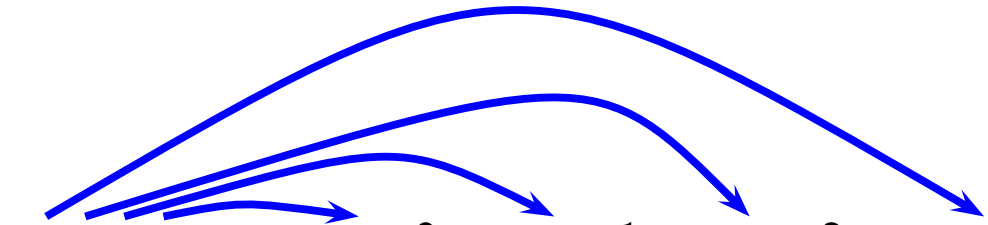
$$= 8 + 40 + 700 + 3000$$

$$= 3748$$

Basis 10 $\rightsquigarrow$ Basis 10 — natürlich witzlos (trivial ;-)

Umrechnung Basis $b \rightsquigarrow$ Basis 10

ein anderes Beispiel Basis 7



$$(2\,1\,6\,4)_7 = 4 \cdot 7^0 + 6 \cdot 7^1 + 1 \cdot 7^2 + 2 \cdot 7^3$$

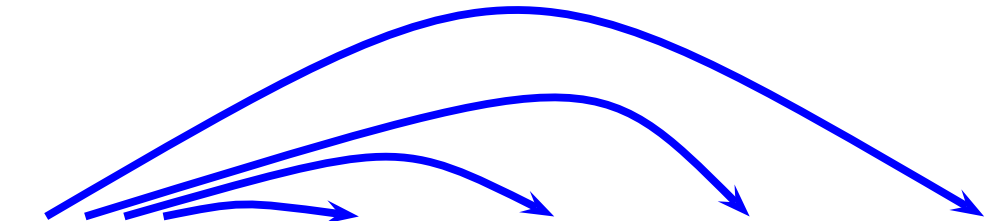
$$= 4 \cdot 1 + 6 \cdot 7 + 1 \cdot 49 + 2 \cdot 343$$

$$= 4 + 42 + 49 + 686$$

$$= 781$$

Umrechnung Basis $b \rightsquigarrow$ Basis 10

und noch ein Beispiel Basis 2



$$(1\,0\,1\,1)_2 = 1 \cdot 2^0 + 1 \cdot 2^1 + 0 \cdot 2^2 + 1 \cdot 2^3$$

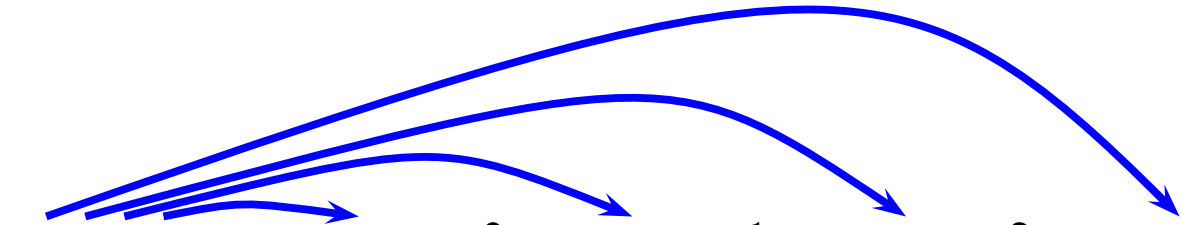
$$= 1 \cdot 1 + 1 \cdot 2 + 0 \cdot 4 + 1 \cdot 8$$

$$= 1 + 2 + 0 + 8$$

$$= 11$$

Umrechnung Basis $b \rightsquigarrow$ Basis 10

und noch ein Beispiel Basis 16



$$(3\,\mathrm{E}\,\mathrm{C}\,9)_{16} = 9 \cdot 16^0 + 12 \cdot 16^1 + 14 \cdot 16^2 + 3 \cdot 16^3$$

$$= 9 \cdot 1 + 12 \cdot 16 + 14 \cdot 256 + 3 \cdot 4\,096$$

$$= 9 + 192 + 3\,584 + 12\,288$$

$$= 16\,073$$

Umrechnung Basis $b \rightsquigarrow$ Basis 10

ein letztes Beispiel Basis 60

$$(2\,48)_{60} = 48 \cdot 60^0 + 2 \cdot 60^1$$

$$= 48 \cdot 1 + 2 \cdot 60$$

$$= 48 + 120$$

$$= 168$$

für uns besonders wichtig Basis 2 (und Basis 10 natürlich)

auch wichtig Basis 16 Warum?

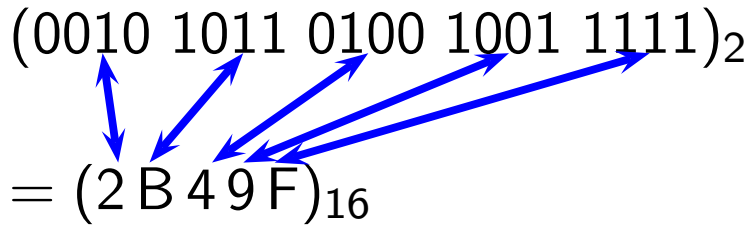
Basis 16

$16 = 2^4$

$(0)_{10}$	$=$	$(0)_{16}$	$=$	$(0)_2$	$(8)_{10}$	$=$	$(8)_{16}$	$=$	$(1000)_2$
$(1)_{10}$	$=$	$(1)_{16}$	$=$	$(1)_2$	$(9)_{10}$	$=$	$(9)_{16}$	$=$	$(1001)_2$
$(2)_{10}$	$=$	$(2)_{16}$	$=$	$(10)_2$	$(10)_{10}$	$=$	$(A)_{16}$	$=$	$(1010)_2$
$(3)_{10}$	$=$	$(3)_{16}$	$=$	$(11)_2$	$(11)_{10}$	$=$	$(B)_{16}$	$=$	$(1011)_2$
$(4)_{10}$	$=$	$(4)_{16}$	$=$	$(100)_2$	$(12)_{10}$	$=$	$(C)_{16}$	$=$	$(1100)_2$
$(5)_{10}$	$=$	$(5)_{16}$	$=$	$(101)_2$	$(13)_{10}$	$=$	$(D)_{16}$	$=$	$(1101)_2$
$(6)_{10}$	$=$	$(6)_{16}$	$=$	$(110)_2$	$(14)_{10}$	$=$	$(E)_{16}$	$=$	$(1110)_2$
$(7)_{10}$	$=$	$(7)_{16}$	$=$	$(111)_2$	$(15)_{10}$	$=$	$(F)_{16}$	$=$	$(1111)_2$

Basis 2 $\longleftrightarrow$ Basis 16

0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111



Namenskonventionen

Basis 10	**Dezimaldarstellung**
Basis 2	**Binärdarstellung** (Ziffern heißen **Bits**)
Basis 16	**Hexadezimaldarstellung** (korrekt eigentlich: *Sedezimalsystem*)

Beobachtung Binärdarstellung für die Repräsentation natürlicher Zahlen im Rechner geeignet

Umrechnung Basis 10 $\rightsquigarrow$ Basis b

Beweis (Korrektheit) im [↗ Anhang]

Algorithmus 1

Eingabe Basis $b \in \mathbb{N} \setminus \{1\}$, Betragszahl $n \in \mathbb{N}$

Ausgabe Repräsentation $(n_l\, n_{l-1} \cdots n_1\, n_0)_b$

1. $l := -1$
2. So lange $n > 0$ gilt
3. $\quad l := l + 1$
4. $\quad n_l := n - b \cdot \lfloor n/b \rfloor$
5. $\quad n := \lfloor n/b \rfloor$

b	16	16	16	16	16
n	38 835	2 427	151	9	0
l	-1	0	1	2	3
n_0		3	3	3	3
n_1			B	B	B
n_2				7	7
n_3					9

also $38\,835 = (9\,7\,\mathrm{B}\,3)_{16}$

Umrechnung Basis 10 $\rightsquigarrow$ Basis b

Beweis (Korrektheit) im [↗ Anhang]

Algorithmus 1

Eingabe Basis $b \in \mathbb{N} \setminus \{1\}$, Betragszahl $n \in \mathbb{N}$

Ausgabe Repräsentation $(n_l\, n_{l-1} \cdots n_1\, n_0)_b$

1. $l := -1$
2. So lange $n > 0$ gilt
3. $\quad l := l + 1$
4. $\quad n_l := n - b \cdot \lfloor n/b \rfloor$
5. $\quad n := \lfloor n/b \rfloor$

b	16	16	16	16	16
n	38.835	2.427	151	9	0
n_3	9				

Quelle: Wikipedia

Definition

Unter einem *Algorithmus* (auch Lösungsverfahren) versteht man eine genau definierte Handlungsvorschrift zur Lösung eines Problems oder einer bestimmten Art von Problemen in endlich vielen Schritten.

b-adische Darstellung – Eigenschaften

Theorem

Für jede Basis $b \in \mathbb{N} \setminus \{1\}$ und jede natürliche Zahl $n \in \mathbb{N}$ lässt sich n eindeutig darstellen als

$$n = n_0 \cdot b^0 + n_1 \cdot b^1 + \cdots + n_l \cdot b^l = \sum_{i=0}^{l} n_i \cdot b^i$$

mit $l \in \mathbb{N}_0$, $n_0, n_1, \ldots, n_l \in \{0, 1, \ldots, b-1\}$ und $n_l > 0$.

Beweis (Existenz + Eindeutigkeit) im [↗ Anhang]

Erstes Fazit

gesehen Einschränkung auf binäre Digitalrechner
in Bezug auf natürliche Zahlen

kein Problem ✓

als nächstes Repräsentation von

- Texten
- ganzen Zahlen (also auch negativen)
- rationalen Zahlen
- anderen Daten

Organisatorisches

Organisatorisches: Vorlesung

Rechnerstrukturen (4V+2Ü)

Termine Mo 16.15–17.45 Uhr HG II, HS 3
Mi 16.15–17.45 Uhr HG II, HS 3

- regelmäßige Teilnahme – erwünscht
- regelmäßige Nachbereitung – erforderlich
  Faustregel Vorlesung : Nachbereitung $\widehat{=}$ 1 : 1
- Skript begleitend lesen – empfehlenswert
- Übungen dazu – erforderlich

Organisatorisches: Vorlesung II

Informationen
http://ls12-www.cs.tu-dortmund.de//de/teaching/courses/ws1011/rs/

bei Fragen **Fragen!**

in der Vorlesung, nach der Vorlesung, oder

gerne	Sprechstunde: Di 11-12 Uhr, OH 16, R.E23
nicht so gerne	Gernot.Fink@tu-dortmund.de
mit *viel* Glück	0231/755-6151

Organisatorisches: Übungen

Veranstalter
Winfried Jansen
Timon Kelter
Alexander Munteanu
+ studentische Tutorinnen und Tutoren

Anmeldung/Einteilung **gleich** im Anschluss an die Vorlesung

Regelmäßige und aktive Teilnahme sehr wichtig!

Übungsschein erforderlich zur Klausurteilnahme („Studienleistung“)

Studienleistungen

Anforderungen für erfolgreiche Übungsteilnahme

- regelmäßige Teilnahme
- erfolgreiche und regelmäßige Bearbeitung der Übungsaufgaben
  - $\geq$ 30% aller Punkte der Übungsblätter 1–3
  - $\geq$ 30% aller Punkte der Übungsblätter 4–6
  - $\geq$ 30% aller Punkte der Übungsblätter 7–9
  - $\geq$ 30% aller Punkte der Übungsblätter 10–12

Hinweis Gruppenabgaben möglich
Gruppengröße ≤ 3

Hinweis: Lernraumbetreuung zusätzlich zu Übungen
(Details werden i.d. Übungen angekündigt)

Übungsgruppeneinteilung

Anmeldeformular für die Übungen in Rechnerstrukturen im WS 2010/2011

Sie können sich einzeln oder als Team (bis zu drei Personen) anmelden. Falls Sie ein Team bilden, benutzen Sie bitte nur ein Anmeldeformular. **Bitte leserlich schreiben!**

Name	Vorname	m/w	Mat.Nr.	Studienrichtung	Nebenfach	Fachsem.

m/w = männlich/weiblich;
Studienrichtung: BA für Bachelor, D für Diplom, KI für (Kern-)Informatik, AI für Angewandte Inf. und LI für Lehramt Inf. (also z.B. BA KI oder D AI), oder Abkürzungen für andere Fachbereiche, z.B, M, Ph, BWL, falls Sie Informatik als Nebenfach studieren;
Nebenfach: Bitte Ihr Nebenfach angeben (falls schon bekannt).
Fachsemester: Bitte das aktuelle Semester eintragen (für die meisten das Erste).

Terminwünsche: Tragen Sie bitte unten in den weißen Tabellenfeldern möglichst viele Wünsche 1, 2, 3 und 4 ein mit der Bedeutung:

1: der Termin wäre mir am liebsten
2: der Termin ist auch ganz gut
3: der Termin geht auch noch
4: wenn es unbedingt sein muss, geht es auch dann

Sie können auch weitere Wünsche 5, 6 usw. eintragen, oder aber denselben Wunsch mehrfach vergeben.

Uhrzeit	Montag	Dienstag	Mittwoch	Donnerstag	Freitag
8-10	xxxxxxxxxx		xxxxxxxxxx	Mathe I	
10-12					xxxxxxxxxx
12-14		DAP Vorlesung	Mathe I	xxxxxxxxxx	xxxxxxxxxx
14-16		xxxxxxxxxx	Mentoring	DAP Vorlesung	xxxxxxxxxx
16-18	RS Vorlesung	xxxxxxxxxx	RS Vorlesung	xxxxxxxxxx	xxxxxxxxxx
18-20	xxxxxxxxxx	xxxxxxxxxx	xxxxxxxxxx	xxxxxxxxxx	xxxxxxxxxx

Abgabe: am Ende der Vorlesung

Die Bekanntgabe der Gruppeneinteilungen erfolgt durch eine Liste im Internet am Freitag, den 15. Oktober 2010, spätestens ab 12.00 Uhr. Informationen zu der Veranstaltung "Übungen zu

Anmeldeformular für die Übungen in Rechnerstruk

Sie können sich einzeln oder als Team (bis zu drei Personen) anmelden. Fa
Sie bitte nur ein Anmeldeformular. **Bitte leserlich schreiben!**

Name	Vorname	m/w	Mat.Nr.	Studienricl

m/w = männlich/weiblich;
Studienrichtung: BA für Bachelor, D für Diplom, KI für (Kern-)Informatik, AI für
Inf. (also z.B. BA KI oder D AI), oder Abkürzungen für andere Fachbereiche, z.B,
Nebenfach studieren;
Nebenfach: Bitte Ihr Nebenfach angeben (falls schon bekannt).
Fachsemester: Bitte das aktuelle Semester eintragen (für die meisten das Erste).

Terminwünsche: Tragen Sie bitte unten in den weißen Tabellenfeldern mö
4 ein mit der Bedeutung:

Anhang

Beweis: Korrektheit von Algorithmus 1!

Ist Algorithmus 1 korrekt? Warum?

1. Algorithmus 1 terminiert, weil n in jedem Schleifendurchlauf immer echt kleiner wird, ganzzahlig bleibt und nie negativ wird.
2. Algorithmus 1 ist korrekt, weil immer

$$n_{\text{Eingabe}} = n_0 \cdot b^0 + n_1 \cdot b^1 + \cdots + n_l \cdot b^l + n_{\text{aktuell}} \cdot b^{l+1}$$

gilt und am Ende $n_{\text{aktuell}} = 0$ ist.

Das muss man noch beweisen, weil es nicht offensichtlich ist!

Beweis der Behauptung

Behauptung

Es gilt immer:

$n_{\text{Eingabe}} = n_0 \cdot b^0 + n_1 \cdot b^1 + \cdots + n_l \cdot b^l + n_{\text{aktuell}} \cdot b^{l+1}$

am Anfang klar, weil $n_{\text{aktuell}} = n_{\text{Eingabe}}$ und $l = -1$

Wir behaupten

wenn es am Anfang eines Schleifendurchlaufs stimmt

dann auch am Ende dieses Schleifendurchgangs.

Begriff Invariante

Beweis Schleifendurchgang erhält Invariante

am Anfang des Durchgangs (Voraussetzung)

$$n_{\text{Eingabe}} = n_0 \cdot b^0 + n_1 \cdot b^1 + \cdots + n_l \cdot b^l + n_{\text{aktuell}} \cdot b^{l+1}$$

im Durchgang

$$l \rightsquigarrow l+1 \qquad n_{l_{\text{neu}}} = n_{\text{aktuell}} - b \cdot \lfloor n_{\text{aktuell}}/b \rfloor \qquad n_{\text{aktuell}} \rightsquigarrow \lfloor n_{\text{aktuell}}/b \rfloor$$

also

$$\underbrace{n_0 \cdot b^0 + n_1 \cdot b^1 + \cdots + n_{l_{\text{alt}}} \cdot b^{l_{\text{alt}}}}_{\text{alte Teildarstellung}}$$

$$+ \underbrace{(n_{\text{aktuell alt}} - b \cdot \lfloor n_{\text{aktuell alt}}/b \rfloor) \cdot b^{l_{\text{neu}}}}_{\text{neue Ziffer}} + \underbrace{\lfloor n_{\text{aktuell alt}}/b \rfloor}_{\text{neues } n_{\text{aktuell}}} \cdot b^{l_{\text{neu}}+1}$$

$$= n_0 \cdot b^0 + n_1 \cdot b^1 + \cdots + n_{l_{\text{alt}}} \cdot b^{l_{\text{alt}}}$$

$$+ b^{l_{\text{neu}}} \cdot (n_{\text{aktuell alt}} - b \cdot \lfloor n_{\text{aktuell alt}}/b \rfloor + \lfloor n_{\text{aktuell alt}}/b \rfloor \cdot b)$$

$$= n_0 \cdot b^0 + n_1 \cdot b^1 + \cdots + n_{l_{\text{alt}}} \cdot b^{l_{\text{alt}}} + b^{l_{\text{neu}}} \cdot n_{\text{aktuell alt}} = n_{\text{Eingabe}} \qquad \square$$

Beweisverfahren „gilt anfangs“ und „bleibt erhalten“
heißt vollständige Induktion

Beweis: b-adische Darstellung – Eigenschaften

Was muss überhaupt bewiesen werden?

1. Existenz Es gibt so eine Darstellung.
   Das beweist Algorithmus 1.
2. Eindeutigkeit Es gibt nur eine Darstellung dieser Art.

Beweismethode Beweis durch Widerspruch

1. Annahme des Gegenteils
2. logisch korrekte Folge von Schlüssen bis zu einer
3. falschen Aussage

b-adische Darstellung – Beweis der Eindeutigkeit

Voraussetzungen Sei $b \in \mathbb{N} \setminus \{1\}$ Basis., Sei $n \in \mathbb{N}$.

Annahme Seien $(n_l\, n_{l-1} \cdots n_0)_b \neq (n'_{l'}\, n'_{l'-1} \cdots n'_0)_b$ verschiedene Darstellungen von n.

O. B. d. A. $l = l'$

sonst kürzere Zahl vorne mit Nullen aufgefüllt.

Schluss $k := \max\{i \in \{0, 1, \ldots, l\} \mid n_i \neq n'_i\}$

ist wohldefiniert, weil Darstellungen verschieden

O. B. d. A. $n_k > n'_k$

sonst alle n_i und n'_i vertauschen

Schluss
$$n = \sum_{i=0}^{l} n_i \cdot b^i > \sum_{i=0}^{l} n'_i \cdot b^i = n$$

weil $\sum_{i=0}^{j} (b-1) \cdot b^i = b^{j+1} - 1 < b^{j+1}$ gilt

für alle $b, j \in \mathbb{N}$

Widerspruch $n > n$ □