



max planck institut
informatik

Ideen und Konzepte der Informatik

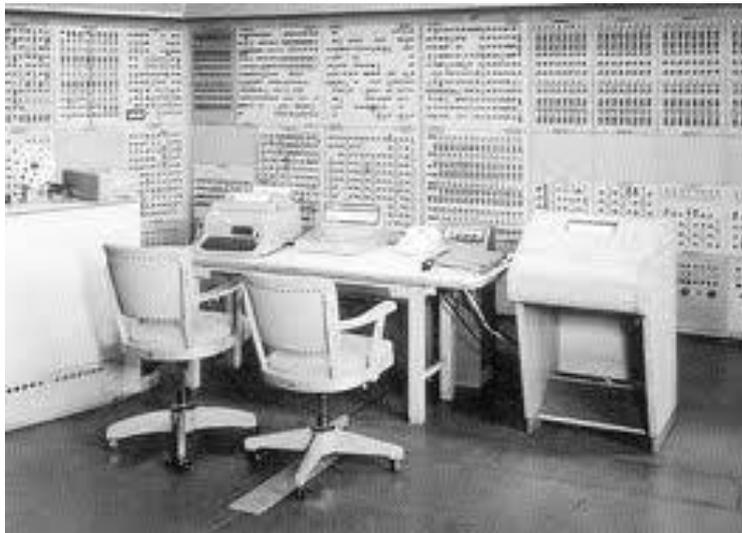
Wie funktionieren Computer?

Kurt Mehlhorn

Übersicht

- Wie funktionieren Computer?
 - Der Von-Neumann-Rechner
- Universalität von Rechnern: Basis für Siegeszug der Informatik
 - Universalität = jeder Rechner kann jedes Programm aufführen
- Geschichtlicher Rückblick
- Alan Turing

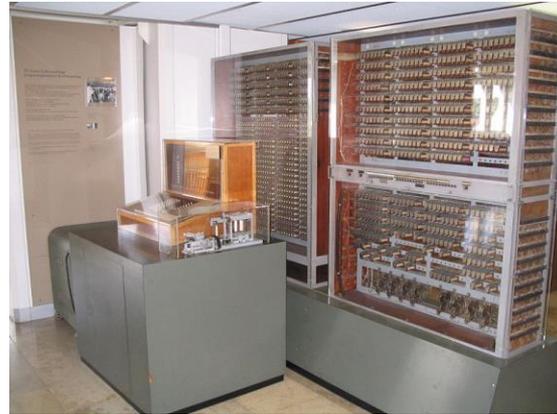
Was ist ein Computer?



Frühe Computer (Konrad Zuse)



Zuse Z1 (1937)



Z3 (1941)



Z4 (1945)

- Z1 ist mechanisch, Z3 und Z4 arbeiten mit Relais
- Z3 und Z4 sind programmierbar (Programm extern) und universell

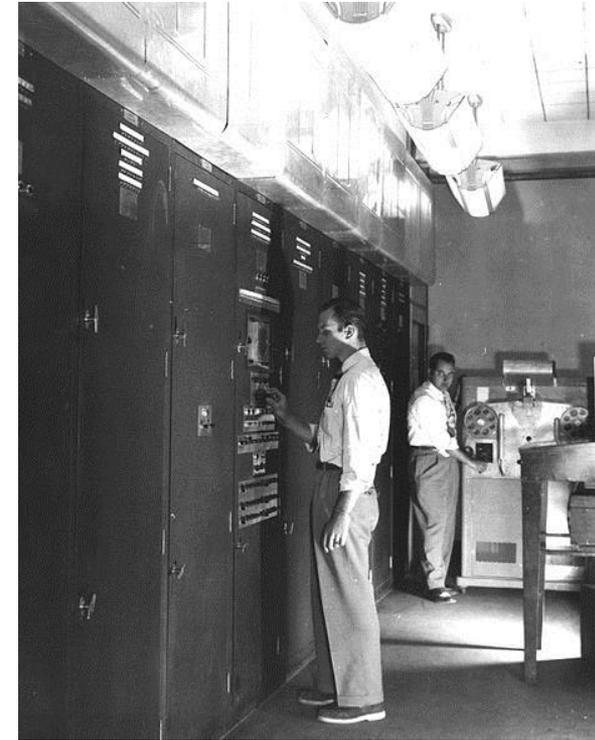
ENIAC (1946)

EDVAC (Electronic ... Computer)

First Draft of a Report on the EDVAC by John von Neumann, 1945

EDVAC, fertiggestellt in 1951

- Stored program
- Speicher: 5.5 kilobytes
- multiplication time: 2.9 milliseconds
- 6,000 vacuum tubes
- consumed 56 kW of power
- 45.5 m² of floor space and weighed 7,850 kg
- operating personnel was thirty people for each eight-hour shift
- Kosten: 500,000 Dollar (entspricht etwa 6 Millionen in 2010)

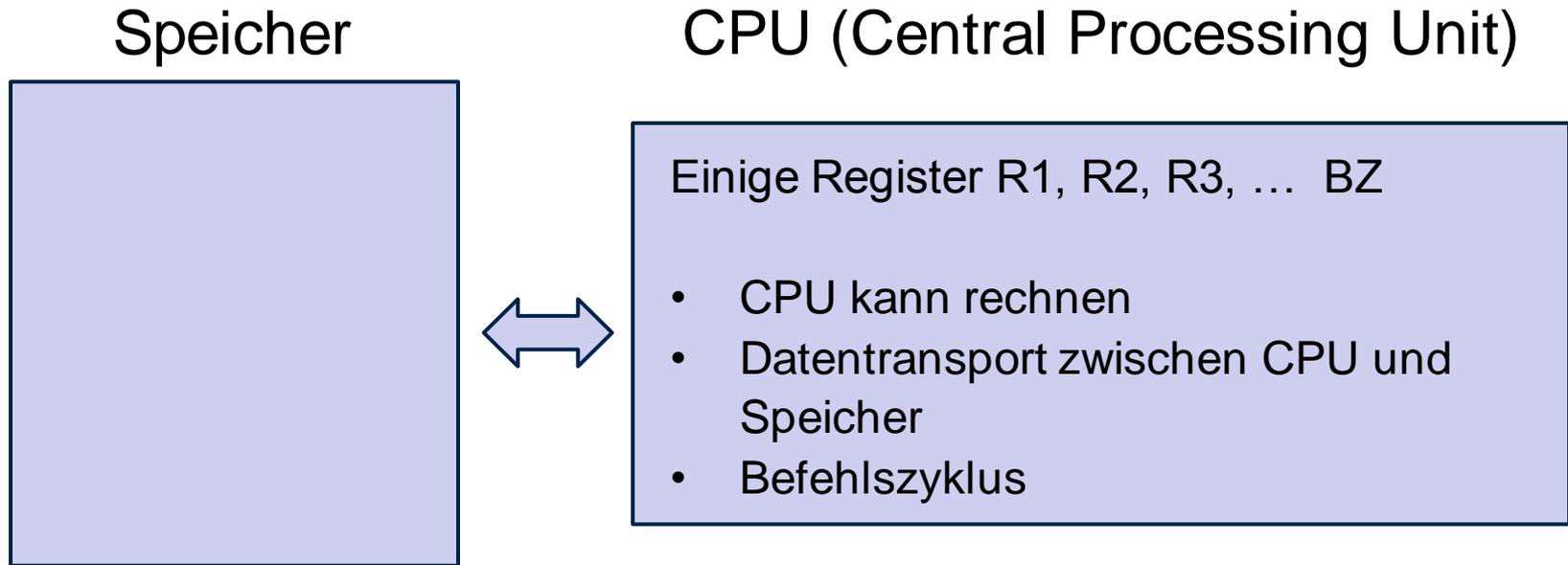


Das Vorbild für alle modernen Rechner

Bitstrings und Speicher

- Bitstring = Folge von Nullen und Einsen, z.B. 00110110
- Bitstring kann man interpretieren als
 - Zahl, z. B. $1010 \equiv 1 \times 8 + 0 \times 4 + 1 \times 2 + 0 \times 1 = 10$
 - Buchstabe, z. B. $01000000 \equiv @$
 - $01100001 \equiv a$
- Speicher besteht aus Speicherzellen
 - sind nummeriert: 0, 1, 2, 3, 4, 5
 - speichern Bitstring der Länge 64, früher: 32, 16

Von-Neumann-Rechner



- Speicherzellen in der CPU heißen Register
- CPU kann rechnen (Befehlsumfang = billiger Taschenrechner)
- Speicher enthält Daten und Programm
- Befehlszyklus sorgt für die Ausführung des Programms,
BZ = Befehlszähler

Typische Befehle

Transport	$R3 \leftarrow M[5]$ $M[5] \leftarrow R2$	$R1 \leftarrow M[R4]$ $M[R2] \leftarrow R1$
Rechnen	$R1 \leftarrow 0$	$R1 \leftarrow R2 + R3$
Sprung	$BZ \leftarrow 7$	$BZ \leftarrow R1$
Bed. Sprung	if $R1 > 0$, $BZ \leftarrow n$, else $BZ \leftarrow BZ + 1$	
Stop	STOP	

Programme und Befehlszyklus

Programm ist eine Folge
von Befehlen

Befehl 1

Befehl 2

Befehl 3

Befehl 4

Befehl 5

Befehl 6

Befehlszyklus,

BZ = Befehlszähler

1. $BZ \leftarrow 1$
2. Führe Befehl mit der Nummer BZ aus. Falls STOP, halte an.
3. Erhöhe BZ um eins (außer bei Sprungbefehl, der BZ setzt)
4. Gehe nach 2.

In $M[1]$ steht eine Zahl n , bilde $1 + \dots + n$

1. $R1 \leftarrow 0$
2. $M[2] \leftarrow R1$
3. $R1 \leftarrow M[2]$
4. $R1 \leftarrow R1 + M[1]$
5. $M[2] \leftarrow R1$
6. $R1 \leftarrow M[1]$
7. $R1 \leftarrow R1 - 1$
8. $M[1] \leftarrow R1$
9. IF $R1 > 0$, $BZ \leftarrow 3$
10. STOP

Ausführung für $n = 4$

BZ

R1

M[1]

M[2]

Laufzeit =

In $M[1]$ steht eine Zahl n , bilde $1 + \dots + n$

1. $R1 \leftarrow 0$
2. $M[2] \leftarrow R1$
3. $R1 \leftarrow M[2]$
4. $R1 \leftarrow R1 + M[1]$
5. $M[2] \leftarrow R1$
6. $R1 \leftarrow M[1]$
7. $R1 \leftarrow R1 - 1$
8. $M[1] \leftarrow R1$
9. IF $R1 > 0$, $BZ \leftarrow 3$
10. STOP

Korrektheit

Bevor Befehl 3 ausgeführt wird, gilt:

- In $M[1]$ steht eine Zahl i mit $n \geq i \geq 1$.
- In $M[2]$ steht $n + \dots + (i + 1)$

Höhere Programmiersprachen

1. $R1 \leftarrow 0$
2. $M[2] \leftarrow R1$
3. $R1 \leftarrow M[2]$
4. $R1 \leftarrow R1 + M[1]$
5. $M[2] \leftarrow R1$
6. $R1 \leftarrow M[1]$
7. $R1 \leftarrow R1 - 1$
8. $M[1] \leftarrow R1$
9. IF $R1 > 0$, $BZ \leftarrow 3$
10. STOP

```
sum ← 0;  
i ← n;  
while (i > 0)  
    sum ← sum + i;  
    i ← i - 1;
```

Produktivitätsgewinn

Java, C, C++, Python,

Compiler übersetzen



Hardware

- Der Speicher, die CPU (central processing unit), die Peripherie (Bildschirm, Tastatur, Maus, Netzanbindung, ...)
- Führt Befehle aus und realisiert den Befehlszyklus.
- Reagiert auf Peripherie.
- Kauft man im Laden.
- Kann jedes Programm ausführen.

Software

- Die Summe der installierten Programme.
- Programme sind im Speicher abgelegt und werden durch die Hardware ausgeführt.
- Der Fantasie sind kaum Grenzen gesetzt.
- Lädt man aus dem Netz.
- Erstellen von guter Software ist teuer.
- Vervielfältigen ist billig, Grenzkosten \approx Null

Kenngrößen und Neuerungen

- Hauptspeicher: 10^9 Worte a 64 Bit
- Befehlszyklus: 10^9 Befehle pro Sekunde
- Eine Million mal leistungsfähiger (Geschwindigkeit, Speicher, Größe), Tausend mal billiger als 1950
- Neuerungen seit 1950
 - Interrupts (Unterbrechungen), mehrere Programme gleichzeitig
 - Speicherhierarchie: Cache, Main, Disk
 - Bildschirme, Grafik, Maus, Sound, Touch, Mikro
 - Netze
 - Preis und Leistung
 - **Software, Nutzerfreundlichkeit**

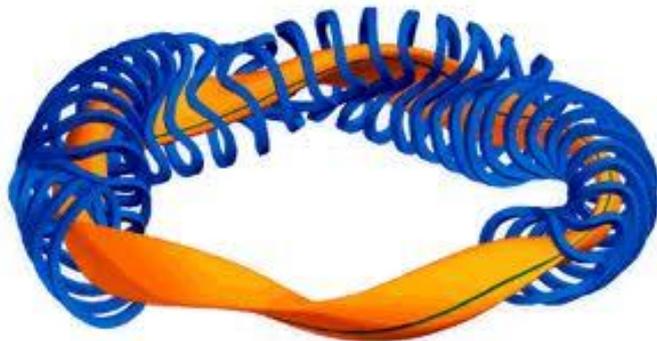
Bildschirme und Graphik

- Dieser Schirm: 1366 x 768 Bildpunkte (Pixels)
- Die CPU kann für jedes Pixel Farbe und Helligkeit einstellen



Supercomputer

- 72 Schränke
- 73000 PowerPCs mit je 2 Gbyte RAM
- Simulation: Physik, Klima, Chemie, Strömung
- 13 Mio Euro



Universalität von Rechnern

- Rechner sind **universell**, d.h., sie können **jedes Programm ausführen**, sofern es nur in ihre Maschinsprache übersetzt ist und es die Ressourcen des Rechners nicht sprengt
- Universalität von Rechnern ist wesentlich für den Erfolg der Informatik
 - Ein Programm auf vielen Rechnern
 - Viele Programme auf einem Rechner

Rechner sind universell

- Normale Werkzeuge sind nicht universell: Hammer, Feile, Zange, Auto,
- Mein Smartphone ist Telefon, aktiver Kalender, Fitnessstrainer, Bankterminal, Wetterauskunft, Browser, Suchmaschine, Musikspieler, Spielzeug, ...
- Office läuft auf Rechnern von IBM, Lenovo, Toshiba, Samsung, Apple, ...

Laufzeit von Programmen

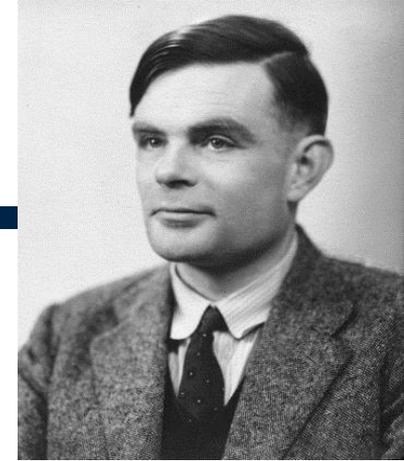
- Ausführungszeit in Sekunden
 - $n = 10^8$, 0.19 sec $n = 10^9$, 1.23 sec
- Für theoretische Überlegungen: Anzahl der ausgeführten Befehle
- Hier: $3 + 7n = O(n)$
- $O()$ = Landausymbol für asymptotisches Wachstum: gibt nur den am schnellsten wachsenden Anteil wieder und ignoriert konstante Faktoren

Geschichtlicher Rückblick

- Charles Babbage (1791 – 1871) + Ada Lovelace: Maschine zur Auswertung von Polynomen, Logarithmentafeln, nie fertig
- Alan Turing (1936) entwirft einfachen universellen Rechner als Gedankenexperiment.
- Konrad Zuse (1941): erster funktionierender programmierbarer Rechner
- Mauchly und Presper (43/44) bauen ENIAC
- Grace Hopper (53): erste Programmiersprache (Cobol)

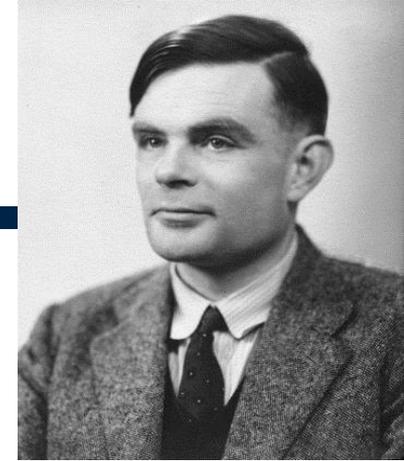


Alan Turing (1912 – 1952)



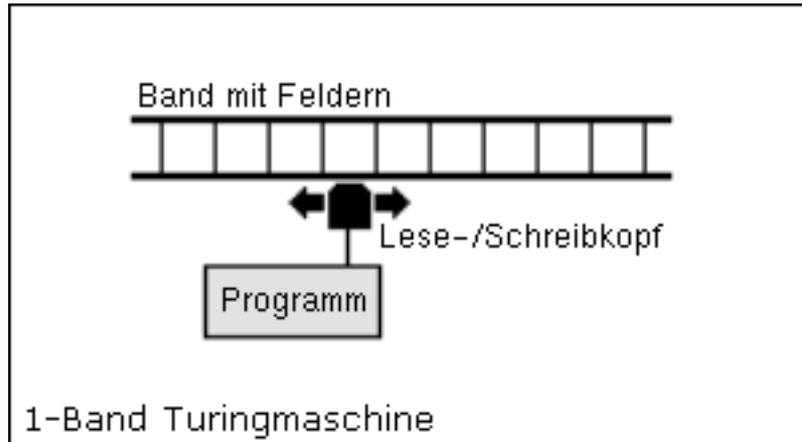
- Britischer Mathematiker
- On Computable Numbers, with an Application to the Entscheidungsproblem
- David Hilbert (1928): kann Mathematik mechanisiert werden?
- Kurt Goedel (1931): NEIN
- Turings Arbeit von 36 vereinfacht den Beweis ganz wesentlich und führt TM ein

Alan Turing (1912 – 1952)



- Kryptanalyse: Churchill stated that Alan Turing made the single biggest contribution to Allied victory in the war against Nazi Germany
- Muster in der Natur
- Sehr guter Sportler, Marathon in 2:46 (Olympiasieger in 48, 2:35)
- Verurteilung wegen Homosexualität in 52, chemische Kastration
- Selbstmord durch Blausäure in 54

Die Turingmaschine



Auf jedem Bandquadrat steht ein Buchstabe (Symbol, Zeichen) in A, ..., Z, a, ..., z, 0, ..., 9, \$, §, ..., leer

Endliches Alphabet

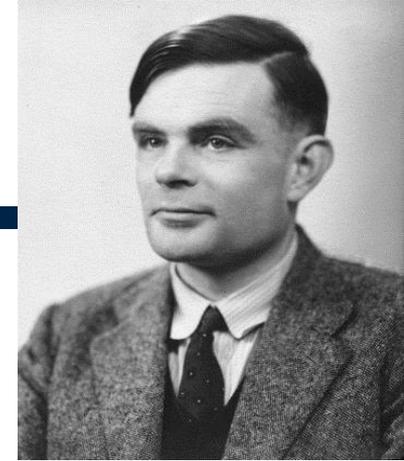
Steuereinheit ist in einem von **endlichen** vielen Zuständen $p, q, q_0, q_1, q_2, \dots$

Turingbefehl	Zustand	Zeichen	neuer Zustand	neues Zeichen	Bewegung
	q_1	a	q_2	b	R

Wenn du im Zustand q_1 ein a liest, dann gehe in den Zustand q_2 über, drucke ein b und bewege den Kopf nach rechts

Turingprogramm = Menge (Folge) von Turingbefehlen, je zwei unterscheiden sich in den ersten Spalten

Turingthese (1936)



- Turing hat TM in 1936 eingeführt
- These: Turing Maschine fasst den Begriff „nach Regeln berechenbar“
- 4 Argumente
 - Menschliche Rechner, siehe nächste Folie
 - Beispiele, zählen, Dezimaldarstellung von π
 - Universelle Turingmaschine
 - Äquivalenz zur Formalisierung von Church
- These ist allgemein akzeptiert

AEG Rechnerraum (1920)



Zusammenfassung

- Rechner sind (im Prinzip) recht einfach aufgebaut: Recheneinheit und Speicher
- Befehlzyklus: wiederhole bis STOP-Befehl
 - Führe Befehl aus und erhöhe Befehlszähler um 1
 - Bei Sprungbefehl setze BZ auf die genannte Adresse
- Übersetzung von höherer Programmiersprache in Maschinensprache erfolgt maschinell
- **Rechner sind universell: viele Programme auf einem Rechner, das gleiche Programm auf vielen Rechnern**