



Universität Hamburg

DER FORSCHUNG | DER LEHRE | DER BILDUNG

MIN-Fakultät
Fachbereich Informatik



64-040 Modul InfB-RSB

Rechnerstrukturen und Betriebssysteme

[https://tams.informatik.uni-hamburg.de/
lectures/2023ws/vorlesung/rsb](https://tams.informatik.uni-hamburg.de/lectures/2023ws/vorlesung/rsb)

– Kapitel 1 –

Andreas Mäder



Universität Hamburg
Fakultät für Mathematik, Informatik und Naturwissenschaften
Fachbereich Informatik

Technische Aspekte Multimodaler Systeme

Wintersemester 2023/2024



Einführung

Exkurs: Geschichte

Personal Computer

von-Neumann Konzept

Moore's Law

System on a chip

Roadmap und Grenzen des Wachstums

Literatur





Brockhaus-Enzyklopädie: „Informatik“

Die Wissenschaft von der systematischen Verarbeitung von Informationen, besonders der automatischen Verarbeitung mit Hilfe von Digitalrechnern (→ Computer). ...





Brockhaus-Enzyklopädie: „Informatik“

Die Wissenschaft von der *systematischen Verarbeitung von Informationen*, besonders der *automatischen Verarbeitung mit Hilfe von Digitalrechnern* (→ Computer). . . .

system. Verarbeitung: von-Neumann Paradigma

- ▶ Wie löst eine Folge elementarer Befehle (Programm) ein Problem?
- ⇒ Softwareentwicklung, Programmierung



Brockhaus-Enzyklopädie: „Informatik“

Die Wissenschaft von der *systematischen Verarbeitung von Informationen*, besonders der *automatischen Verarbeitung mit Hilfe von Digitalrechnern* (→ Computer). . . .

system. Verarbeitung: von-Neumann Paradigma

- ▶ Wie löst eine Folge elementarer Befehle (Programm) ein Problem?
- ⇒ Softwareentwicklung, Programmierung

Digitalrechner: das technische System dazu (Rechnerarchitektur)

- ▶ Wie wird Information (Zahlen, Zeichen) repräsentiert/codiert?
 - ▶ Wie arbeiten technische Schaltungen (Hardware) Befehle ab?
- ⇒ Hardwareentwicklung



- ▶ seit 80er Jahren: unterschiedliche, getrennte Paradigmen
 - SW** ▶ Hardware ist vorgegeben
 - ▶ Abstraktion von der Hardware möglich
 - ▶ Programmierung in Hochsprachen (Produktivität)
 - HW** ▶ technische Werte als Optimierungsziel (Taktfrequenz, Latenz, Durchsatz, Leistungsaufnahme etc.)
 - ▶ getrieben von technischer Entwicklung (*Moore's Law*)
 - ▶ Maschinenbefehl wird auf Hardwarearchitektur ausgeführt
 - ▶ Kontext aus SW und Betriebssystem wird nicht beachtet
- ▶ Trend: technischer Fortschritt langsamer
Leistungssteigerungen durch neue Architekturkonzepte



Trennung von Software und Hardware?

- ▶ seit 80er Jahren: unterschiedliche, getrennte Paradigmen
 - SW** ▶ Hardware ist vorgegeben
 - ▶ Abstraktion von der Hardware möglich
 - ▶ Programmierung in Hochsprachen (Produktivität)
 - HW** ▶ technische Werte als Optimierungsziel (Taktfrequenz, Latenz, Durchsatz, Leistungsaufnahme etc.)
 - ▶ getrieben von technischer Entwicklung (*Moore's Law*)
 - ▶ Maschinenbefehl wird auf Hardwarearchitektur ausgeführt
 - ▶ Kontext aus SW und Betriebssystem wird nicht beachtet
 - ▶ Trend: technischer Fortschritt langsamer
Leistungssteigerungen durch neue Architekturkonzepte
- ⇒ dies funktioniert seit Jahren!

- ▶ seit 80er Jahren: unterschiedliche, getrennte Paradigmen
 - SW** ▶ Hardware ist vorgegeben
 - ▶ Abstraktion von der Hardware möglich
 - ▶ Programmierung in Hochsprachen (Produktivität)
 - HW** ▶ technische Werte als Optimierungsziel (Taktfrequenz, Latenz, Durchsatz, Leistungsaufnahme etc.)
 - ▶ getrieben von technischer Entwicklung (*Moore's Law*)
 - ▶ Maschinenbefehl wird auf Hardwarearchitektur ausgeführt
 - ▶ Kontext aus SW und Betriebssystem wird nicht beachtet
- ▶ Trend: technischer Fortschritt langsamer
Leistungssteigerungen durch neue Architekturkonzepte
- ▶ dies funktioniert seit Jahren ... bis Ende 2017



...

Wechselwirkungen zwischen SW und HW
als „Angriffsszenario“

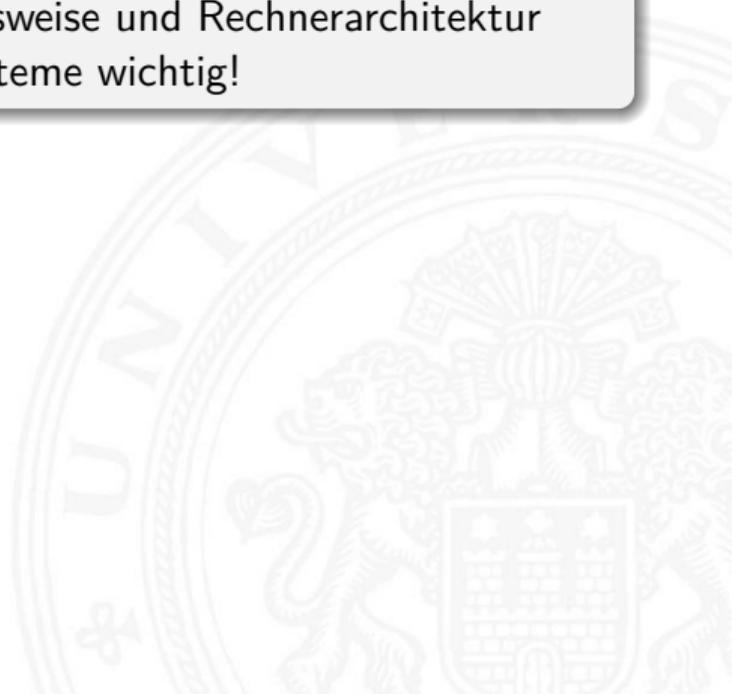


Konsequenz

verschiedene Sichtweisen funktionieren nicht mehr!

Programmierer: Grundverständnis techn. Funktionsweise und Rechnerarchitektur

Hardwaredesigner: Programmabläufe und Betriebssysteme wichtig!





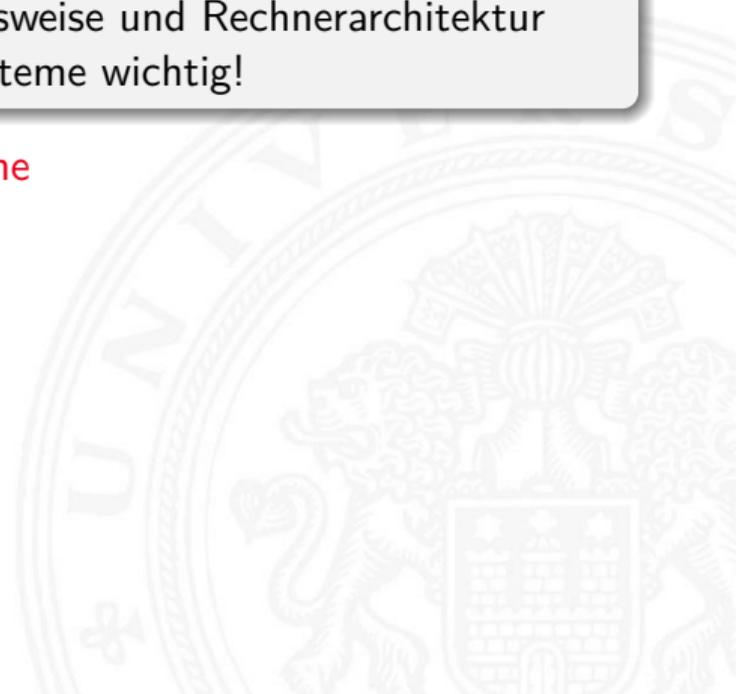
Konsequenz

verschiedene Sichtweisen funktionieren nicht mehr!

Programmierer: Grundverständnis techn. Funktionsweise und Rechnerarchitektur

Hardwaredesigner: Programmabläufe und Betriebssysteme wichtig!

Motivation für **Rechnerstrukturen und Betriebssysteme**



Konsequenz

verschiedene Sichtweisen funktionieren nicht mehr!

Programmierer: Grundverständnis techn. Funktionsweise und Rechnerarchitektur

Hardwaredesigner: Programmabläufe und Betriebssysteme wichtig!

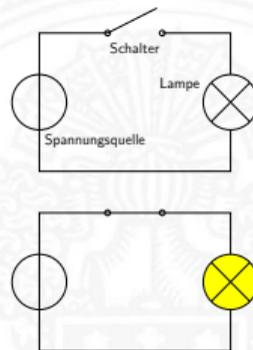
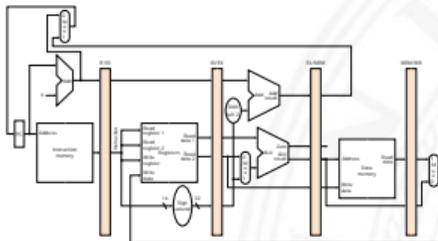
Motivation für **Rechnerstrukturen und Betriebssysteme**

⇒ Interaktion von SW und HW

```
#include <stdio.h>

int main( int argc, char ** argv )
{ printf( "Hello, world!\n" );
  return 0;
}
```

- ▶ „performante“, „sichere“ Software programmieren
- ▶ Systemsicht / Variantenvielfalt von Mikroprozessorsystemen
- ▶ Bewertung von Trends und Perspektiven



Konsequenz

verschiedene Sichtweisen funktionieren nicht mehr!

Programmierer: Grundverständnis techn. Funktionsweise und Rechnerarchitektur

Hardwaredesigner: Programmabläufe und Betriebssysteme wichtig!

Motivation für Rechnerstrukturen und Betriebssysteme

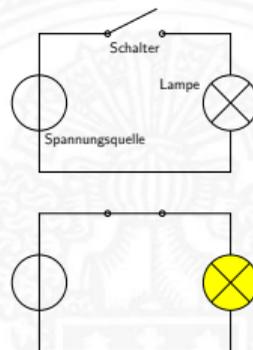
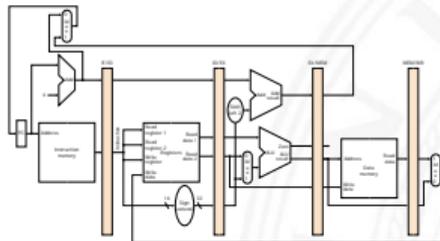
⇒ Interaktion von SW und HW

```
#include <stdio.h>

int main( int argc, char ** argv )
{ printf( "Hello, world!\n" );
  return 0;
}
```

- ▶ „performante“, „sichere“ Software programmieren
- ▶ Systemsicht / Variantenvielfalt von Mikroprozessorsystemen
- ▶ Bewertung von Trends und Perspektiven

⇒ *Wie funktioniert ein Computer?*



► Formel $n! = \prod_{n=1}^n n$ $0! = 1$

► C Code */* Fakultät... - nicht rekursiv */*

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

unsigned int fak(unsigned int n)
{ unsigned int res = 1;
  while (n > 1)
  { res = n * res;
    n = n - 1;
  }
  return res;
}

int main(int argc, char **argv)
{ int arg;

  if ((argc != 2) || ((arg = atoi(argv[1])) < 0))
  { printf ("usage: \"fak <n>\", n >= 0\n");
    exit(1);
  }

  printf ("%d! = %d\n", arg, fak(arg));
  exit(0);
}
```

Wie funktioniert ein Computer?

```
unsigned int fak(unsigned int n)
{ unsigned int res = 1;
  while (n > 1)
  { res = n * res;
    n  = n - 1;
  }
  return res;
}
```

Ablauf 4!

$$\frac{n =}{res =}$$



Wie funktioniert ein Computer?

```
unsigned int fak(unsigned int n)
{ unsigned int res = 1;
  while (n > 1)
  { res = n * res;
    n  = n - 1;
  }
  return res;
}
```

Ablauf 4!

$$\frac{n = 4}{res =}$$



Wie funktioniert ein Computer?

```
unsigned int fak(unsigned int n)
{ unsigned int res = 1;
  while (n > 1)
  { res = n * res;
    n  = n - 1;
  }
  return res;
}
```

Ablauf 4!

$$\frac{n = 4}{res = 1}$$

Wie funktioniert ein Computer?

```
unsigned int fak(unsigned int n)
{ unsigned int res = 1;
  while (n > 1)
  { res = n * res;
    n  = n - 1;
  }
  return res;
}
```

Ablauf 4!

$$\frac{n = 4}{res = 1}$$

Wie funktioniert ein Computer?

```
unsigned int fak(unsigned int n)
{ unsigned int res = 1;
  while (n > 1)
  { res = n * res;
    n  = n - 1;
  }
  return res;
}
```

Ablauf 4!

$$\frac{n = 4}{res = 1 \quad 4}$$

Wie funktioniert ein Computer?

```
unsigned int fak(unsigned int n)
{ unsigned int res = 1;
  while (n > 1)
  { res = n * res;
    n  = n - 1;
  }
  return res;
}
```

Ablauf 4!

n = 4	3
<hr/>	
res = 1	4

Wie funktioniert ein Computer?

```
unsigned int fak(unsigned int n)
{ unsigned int res = 1;
  while (n > 1)
  { res = n * res;
    n  = n - 1;
  }
  return res;
}
```

Ablauf 4!

n =	4	3
res =	1	4

Wie funktioniert ein Computer?

```
unsigned int fak(unsigned int n)
{ unsigned int res = 1;
  while (n > 1)
  { res = n * res;
    n  = n - 1;
  }
  return res;
}
```

Ablauf 4!

n =	4	3
res =	1	4 12

Wie funktioniert ein Computer?

```
unsigned int fak(unsigned int n)
{ unsigned int res = 1;
  while (n > 1)
  { res = n * res;
    n  = n - 1;
  }
  return res;
}
```

Ablauf 4!

n =	4	3	2
res =	1	4	12

Wie funktioniert ein Computer?

```
unsigned int fak(unsigned int n)
{ unsigned int res = 1;
  while (n > 1)
  { res = n * res;
    n  = n - 1;
  }
  return res;
}
```

Ablauf 4!

n =	4	3	2
res =	1	4	12

Wie funktioniert ein Computer?

```
unsigned int fak(unsigned int n)
{ unsigned int res = 1;
  while (n > 1)
  { res = n * res;
    n  = n - 1;
  }
  return res;
}
```

Ablauf 4!

n =	4	3	2	
res =	1	4	12	24

Wie funktioniert ein Computer?

```
unsigned int fak(unsigned int n)
{ unsigned int res = 1;
  while (n > 1)
  { res = n * res;
    n  = n - 1;
  }
  return res;
}
```

Ablauf 4!

n =	4	3	2	1
res =	1	4	12	24

Wie funktioniert ein Computer?

```
unsigned int fak(unsigned int n)
{ unsigned int res = 1;
  while (n > 1)
  { res = n * res;
    n  = n - 1;
  }
  return res;
}
```

Ablauf 4!

n =	4	3	2	1
res =	1	4	12	24

Wie funktioniert ein Computer?

```
unsigned int fak(unsigned int n)
{ unsigned int res = 1;
  while (n > 1)
  { res = n * res;
    n  = n - 1;
  }
  return res;
}
```

- ▶ *Wie werden Programmanweisungen schrittweise abgearbeitet?*

Wie funktioniert ein Computer?

```
unsigned int fak(unsigned int n)
{ unsigned int res = 1;
  while (n > 1)
  { res = n * res;
    n  = n - 1;
  }
  return res;
}
```

- ▶ *Wie werden Programmanweisungen schrittweise abgearbeitet?*
- ▶ *Wie wird Information (hier Zahlen) technisch dargestellt?*

Wie funktioniert ein Computer?

```
unsigned int fak(unsigned int n)
{ unsigned int res = 1;
  while (n > 1)
  { res = n * res;
    n  = n - 1;
  }
  return res;
}
```

- ▶ *Wie werden Programmanweisungen schrittweise abgearbeitet?*
- ▶ *Wie wird Information (hier Zahlen) technisch dargestellt?*
- ▶ *Welche technischen Komponenten werden benötigt?*
Wie funktionieren sie? Wie kann damit gerechnet werden?

Wie funktioniert ein Computer?

```
unsigned int fak(unsigned int n)
{ unsigned int res = 1;
  while (n > 1)
  { res = n * res;
    n  = n - 1;
  }
  return res;
}
```

- ▶ *Wie werden Programmanweisungen schrittweise abgearbeitet?*
- ▶ *Wie wird Information (hier Zahlen) technisch dargestellt?*
- ▶ *Welche technischen Komponenten werden benötigt?
Wie funktionieren sie? Wie kann damit gerechnet werden?*
- ▶ *Welche Mechanismen sorgen dafür, dass mehrere Anwendungen
(und mehrere Benutzer) „gleichzeitig“ arbeiten?*



1. ständige technische Fortschritte in Mikro- und Optoelektronik mit einem weiterhin *exponentiellen* Wachstum

(50% ... 100% pro Jahr)

- ▶ Rechenleistung von Prozessoren / „Performanz“
- ▶ Speicherkapazität Hauptspeicher (DRAM, SRAM, FLASH)
- ▶ Speicherkapazität Langzeitspeicher (Festplatten, FLASH)
- ▶ Übertragungsraten / Bandbreite (Netzwerke)

2. neue Entwurfsparadigmen und -werkzeuge

⇒ Möglichkeiten und Anwendungsfelder

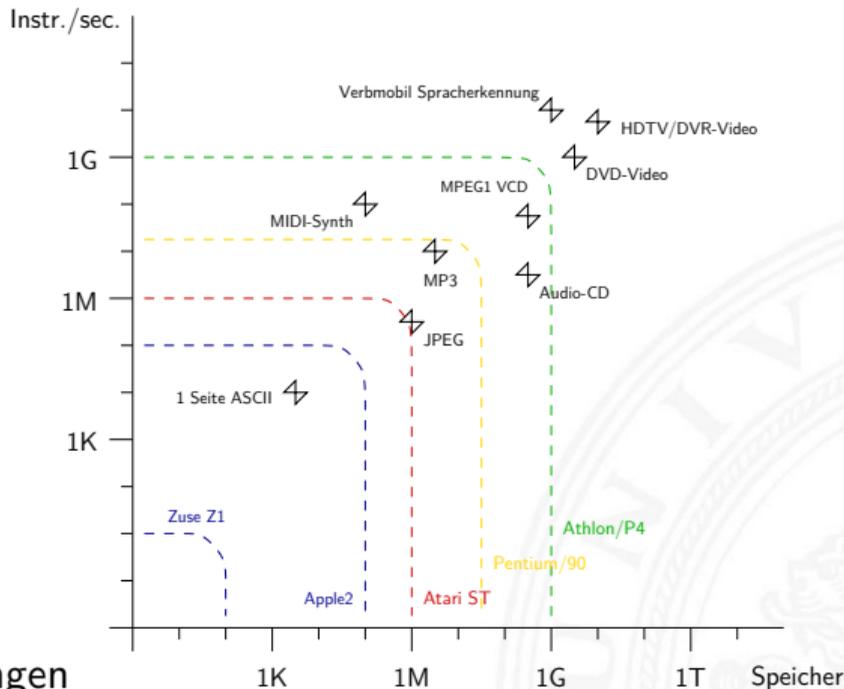
⇒ Produkte und Techniken



Kriterien / Maßgrößen

- ▶ Rechenleistung: MIPS
- ▶ MBytes (RAM, HDD)
- ▶ Mbps
- ▶ MPixel

⇒ jede Rechnergeneration erlaubt neue Anwendungen



Hardware für den Mondflug





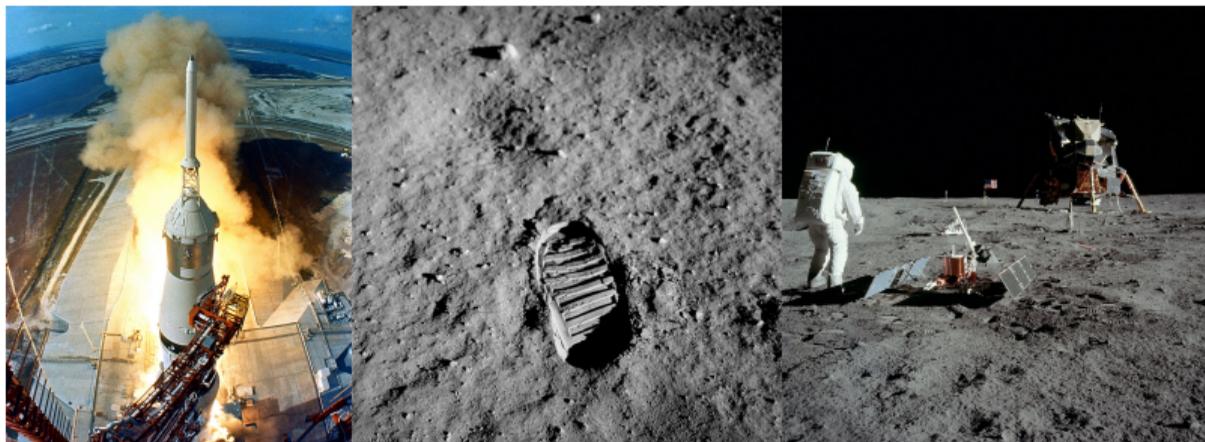
Hardware für den Mondflug



... aber nicht 2023

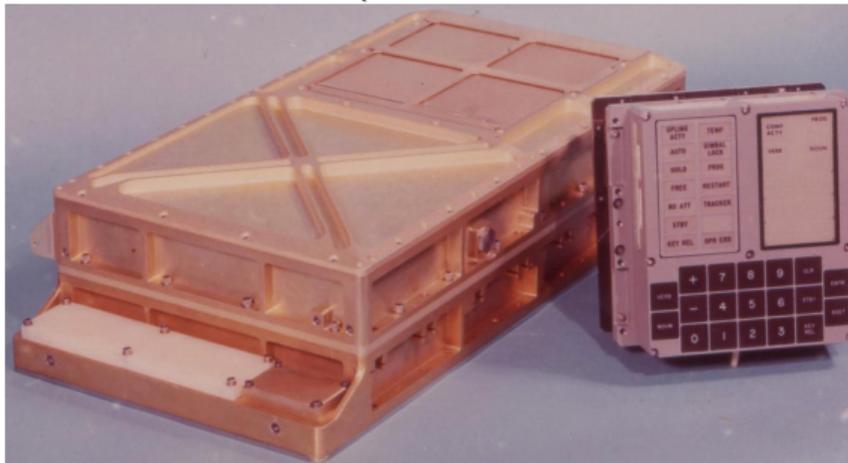


Beispiel: Apollo 11 (1969)



- ▶ bernd-leitenberger.de/computer-raumfahrt1.shtml
- ▶ history.nasa.gov/computers/contents.html
- ▶ en.wikipedia.org/wiki/Apollo_Guidance_Computer
- ▶ en.wikipedia.org/wiki/IBM_System/360
www.computerhistory.org/revolution/mainframe-computers/7

1. Bordrechner: AGC (Apollo Guidance Computer)



Dryden Flight Research Center EC96-43408-1. Photographed 1996
Apollo computer interface box used in the F-8 Digital
Fly-By-Wire program. NASA photo by Dennis Taylor

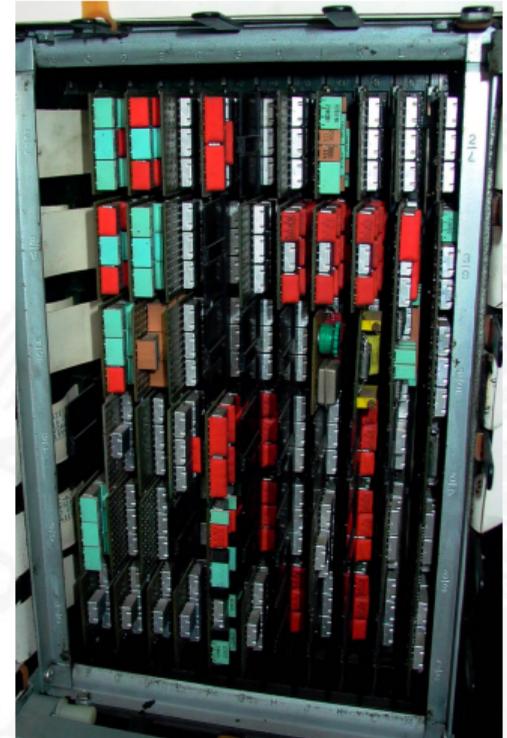
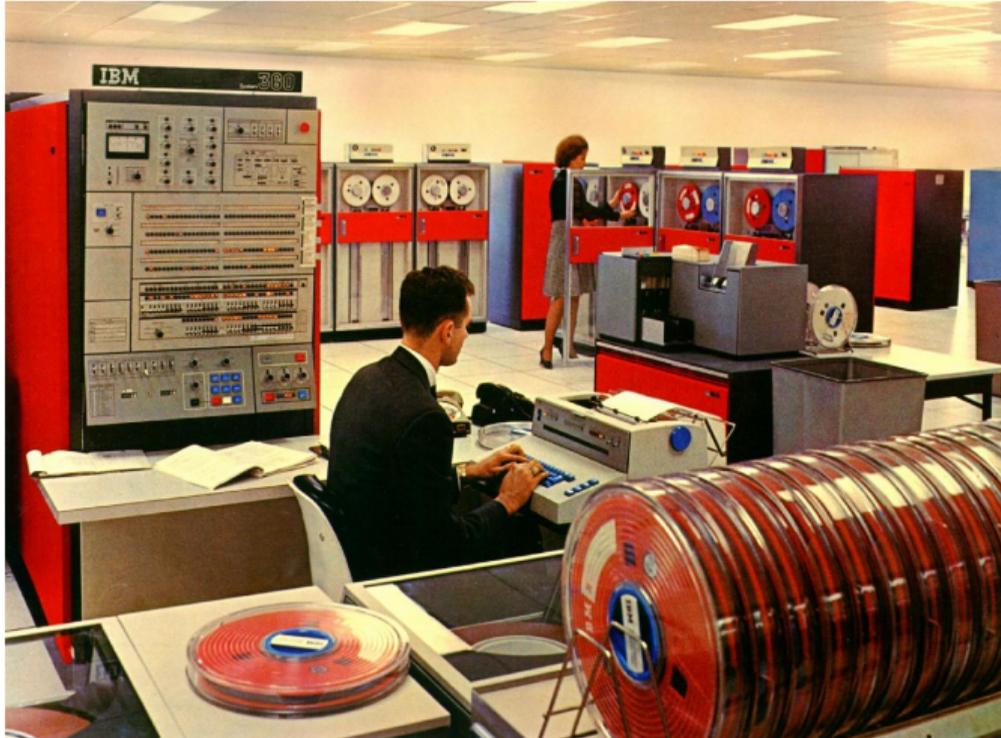
- ▶ Dimension $61 \times 32 \times 15,0$ cm 31,7 kg
20 \times 20 \times 17,5 cm 8,0 kg
- ▶ Taktfrequenz: 1,024 MHz
- ▶ 16-bit Worte, nur Festkomma
- ▶ Speicher ROM 36 KWorte 72 KByte
RAM 2 KWorte 4 KByte

Addition: $\approx 20 \mu\text{s}$, mehrere Takte

Zykluszeit: $11,7 \mu\text{s}$, 12 Takte, 85,3 KHz

2. mehrere Großrechner: IBM System/360 Model 75s

[▶ YouTube Video](#)



- ▶ je nach Ausstattung: Anzahl der „Schränke“
- ▶ Taktfrequenz: bis 5 MHz
- ▶ 32-bit Worte, 24-bit Adressraum (16 MByte)
- ▶ Speicherhierarchie: bis 1 MByte Hauptspeicher (1,3 MHz Zykluszeit)
- ▶ (eigene) Fließkomma Formate
- ▶ Rechenleistung: 0,7 Dhrystone MIPS



- ▶ je nach Ausstattung: Anzahl der „Schränke“
 - ▶ Taktfrequenz: bis 5 MHz
 - ▶ 32-bit Worte, 24-bit Adressraum (16 MByte)
 - ▶ Speicherhierarchie: bis 1 MByte Hauptspeicher (1,3 MHz Zykluszeit)
 - ▶ (eigene) Fließkomma Formate
 - ▶ Rechenleistung: 0,7 Dhrystone MIPS
- ▶ ... und 2016 ¹

	CPU	Cores	[DMIPS]	F_{clk} [GHz]
Smartphone	Exynos 8890	8	47 840	2,3
Desktop PC	Core i7 6950X	10	317 900	3,0

¹Daten aktueller Systeme sind nicht verfügbar – Benchmark ist überholt

- ▶ je nach Ausstattung: Anzahl der „Schränke“
 - ▶ Taktfrequenz: bis 5 MHz
 - ▶ 32-bit Worte, 24-bit Adressraum (16 MByte)
 - ▶ Speicherhierarchie: bis 1 MByte Hauptspeicher (1,3 MHz Zykluszeit)
 - ▶ (eigene) Fließkomma Formate
 - ▶ Rechenleistung: 0,7 Dhrystone MIPS
- ▶ ... und 2016 ¹

	CPU	Cores	[DMIPS]	F_{clk} [GHz]
Smartphone	Exynos 8890	8	47 840	2,3
Desktop PC	Core i7 6950X	10	317 900	3,0

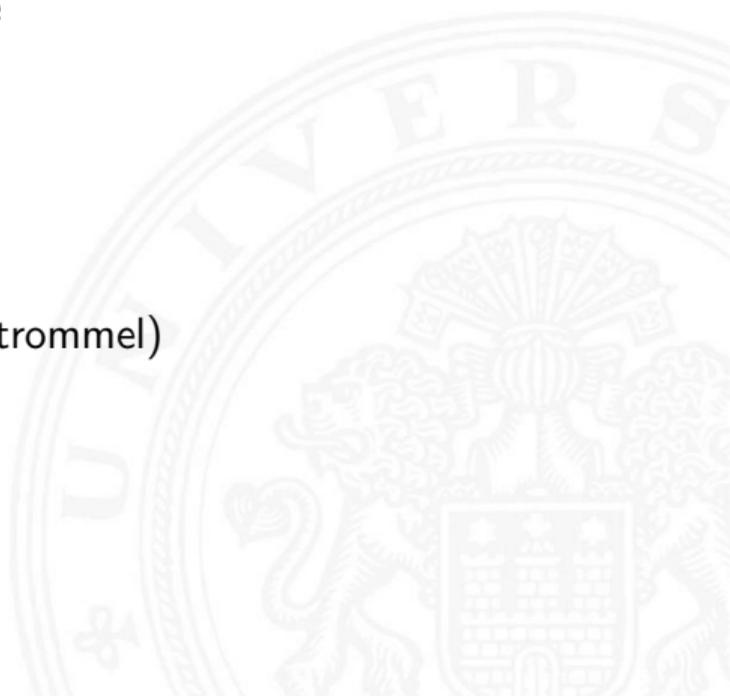
⇒ Moore's Law

¹Daten aktueller Systeme sind nicht verfügbar – Benchmark ist überholt



- ???? Abakus als erste Rechenhilfe
- 1642 Pascal: Addierer/Subtrahierer
- 1671 Leibniz: Vier-Operationen-Rechenmaschine
- 1837 Babbage: Analytical Engine

- 1937 Zuse: Z1 (mechanisch)
- 1939 Zuse: Z3 (Relais, Gleitkomma)
- 1941 Atanasoff & Berry: ABC (Röhren, Magnettrommel)
- 1944 Mc-Culloch Pitts (Neuronenmodell)
- 1946 Eckert & Mauchly: ENIAC (Röhren)
- 1949 Eckert, Mauchly, von Neumann: EDVAC
(erster speicherprogrammierter Rechner)
- 1949 Manchester Mark-1 (Indexregister)





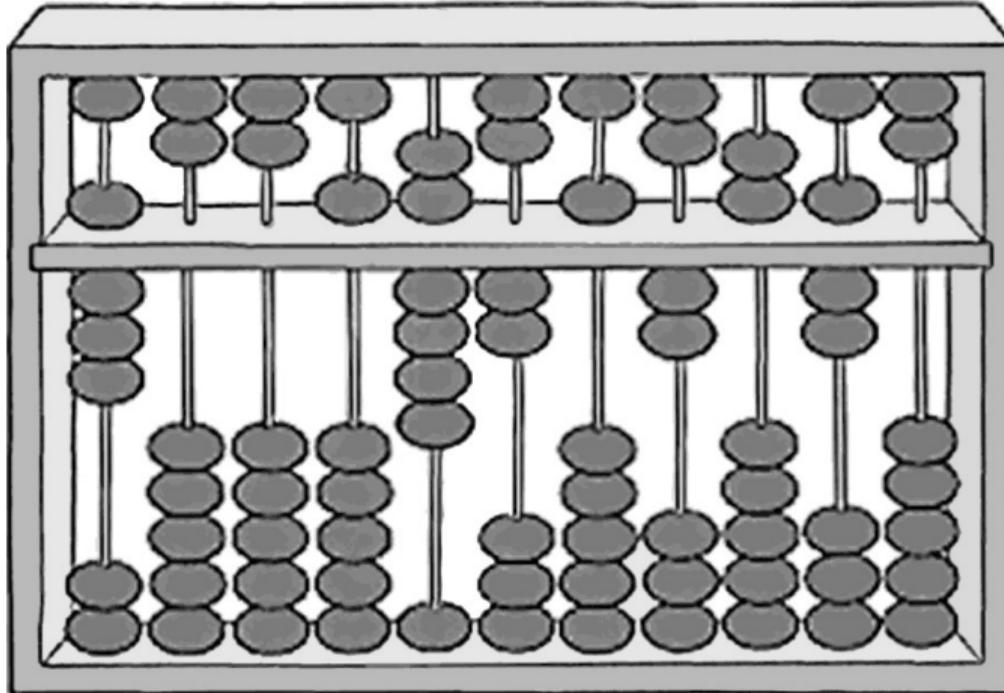
Abakus

Wert in Spalte

8 0 0 5 14 2 5 2 10 7 0

Kugel = 5

Kugel = 1



Zehnerpotenz
der Spalte

10^{10} 10^9 10^8 10^7 10^6 10^5 10^4 10^3 10^2 10^1 10^0



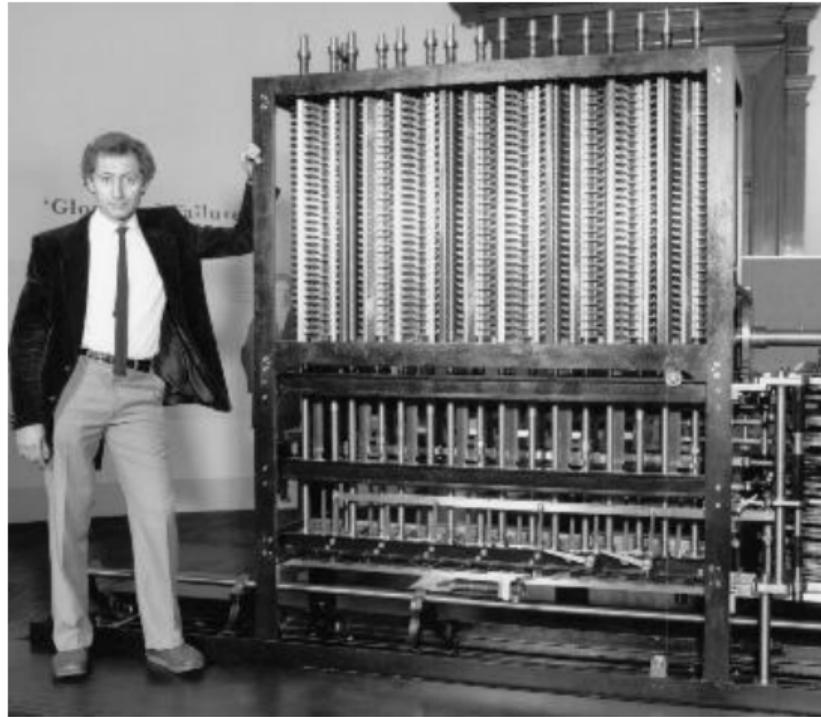
- 1623 Schickard: Sprossenrad, Addierer/Subtrahierer
- 1642 Pascal: „Pascalene“
- 1673 Leibniz: Staffelwalze, Multiplikation/Division
- 1774 Philipp Matthäus Hahn: erste gebrauchsfähige '4-Spezies'-Maschine

Difference Engine

Charles Babbage 1822: Berechnung nautischer Tabellen

1.1 Einführung - Exkurs: Geschichte

64-040 Rechnerstrukturen und Betriebssysteme



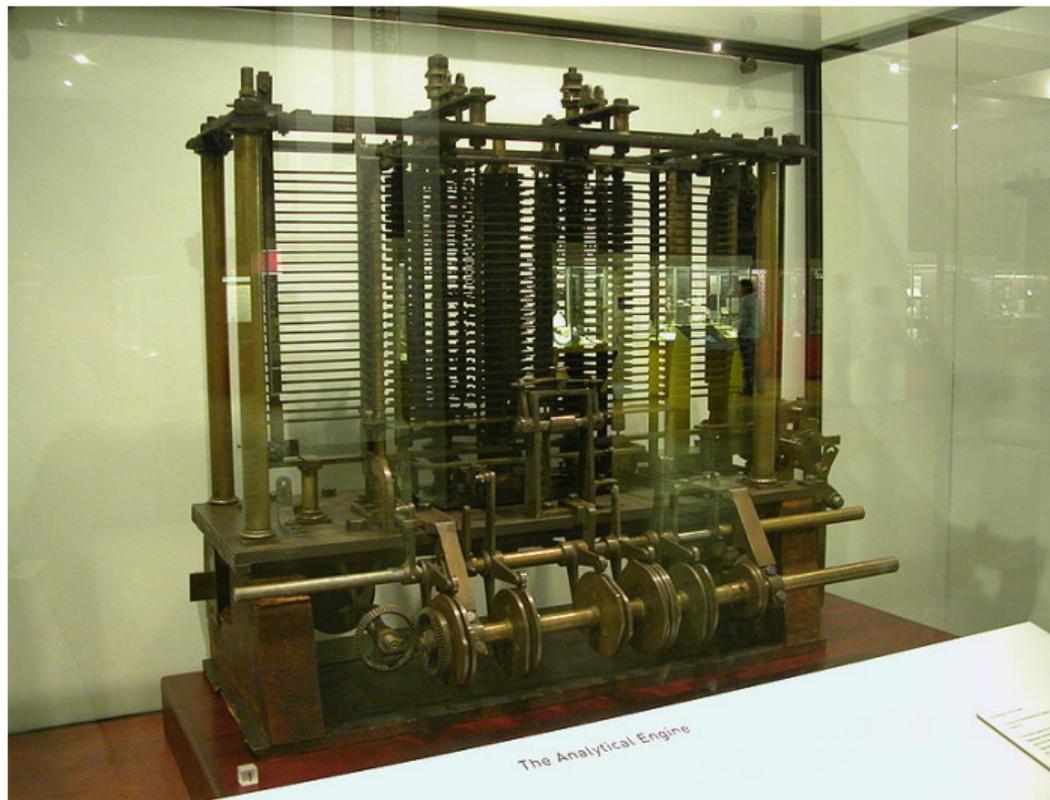
Original von 1832 und Nachbau von 1989, London Science Museum

Analytical Engine

Charles Babbage 1837-1871: frei programmierbar, Lochkarten, unvollendet

1.1 Einführung - Exkurs: Geschichte

64-040 Rechnerstrukturen und Betriebssysteme





- ▶ nicht mehr eine Maschine für eine Aufgabe / ein Problem sondern **feste Hardware**
 - + problemspezifisch wird ein **flexibles Programm** entwickelt

 - ▶ schrittweises Problemlösen \Rightarrow Algorithmen
 - ▶ zentrale Paradigmen der Informatik
- \Rightarrow **von-Neumann Konzept**

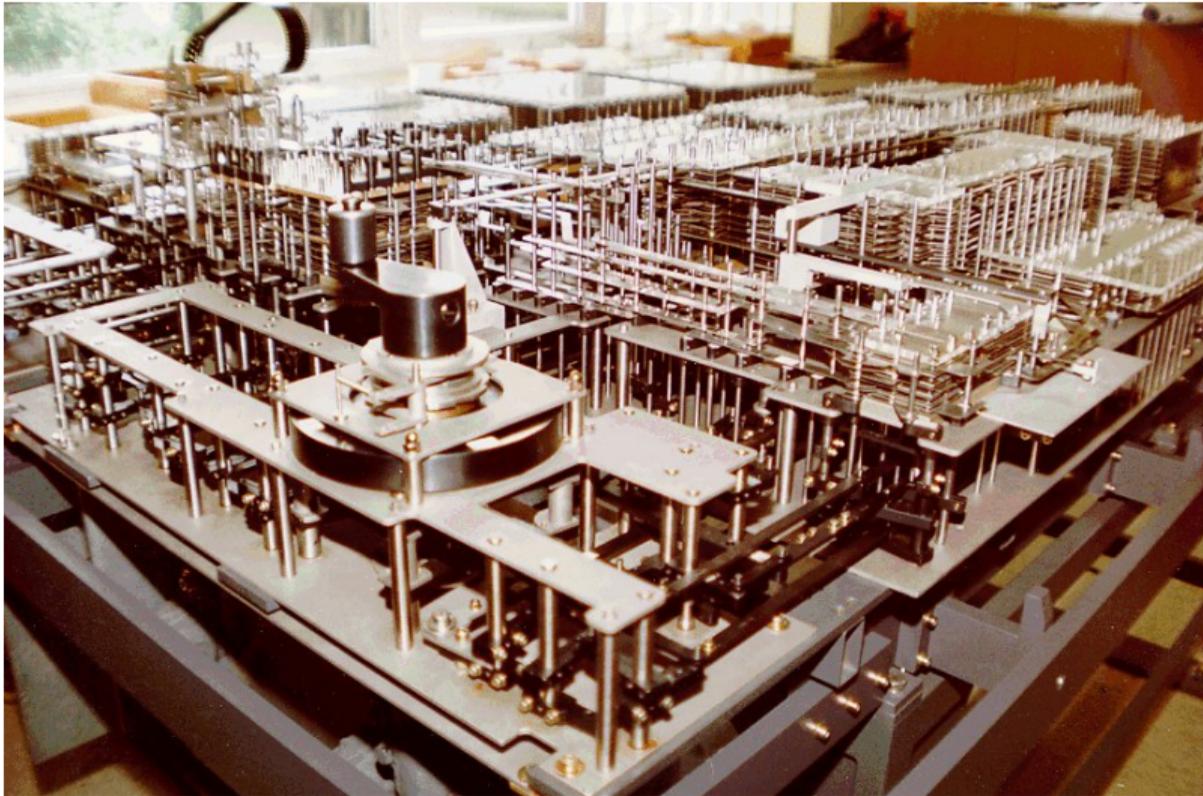


Zuse Z1

Konrad Zuse 1937: 64 Register, 22-bit, mechanisch, Lochfilm

1.1 Einführung - Exkurs: Geschichte

64-040 Rechnerstrukturen und Betriebssysteme

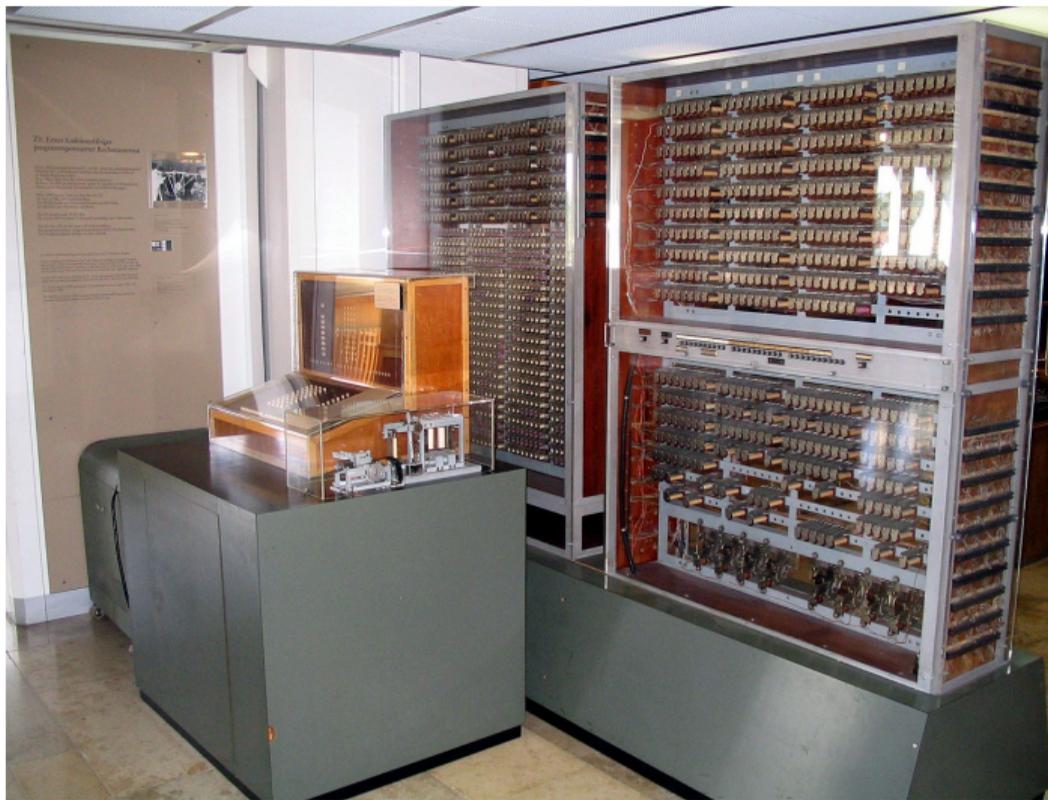


Zuse Z3

Konrad Zuse 1941, 64 Register, 22-bit, 2000 Relays, Lochfilm

1.1 Einführung - Exkurs: Geschichte

64-040 Rechnerstrukturen und Betriebssysteme

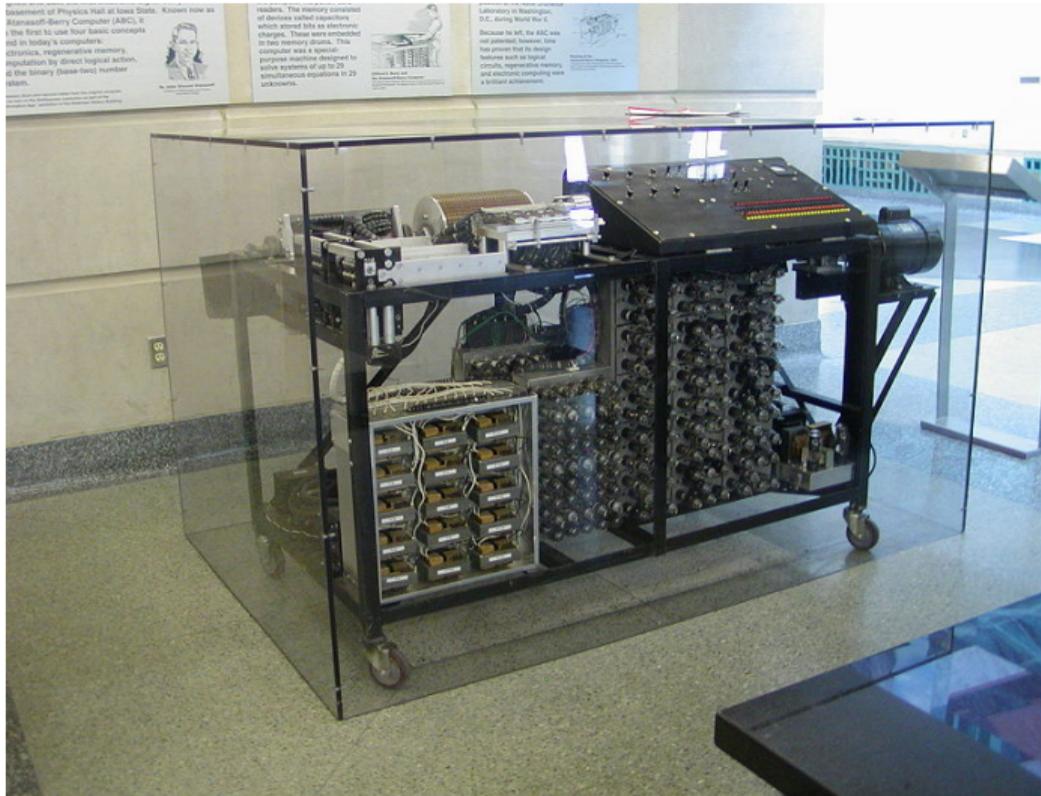


Atanasoff-Berry Computer (ABC)

J.V. Atanasoff 1942: 50-bit Festkomma, Röhren und Trommelspeicher, fest programmiert, erste ALU

1.1 Einführung - Exkurs: Geschichte

64-040 Rechnerstrukturen und Betriebssysteme

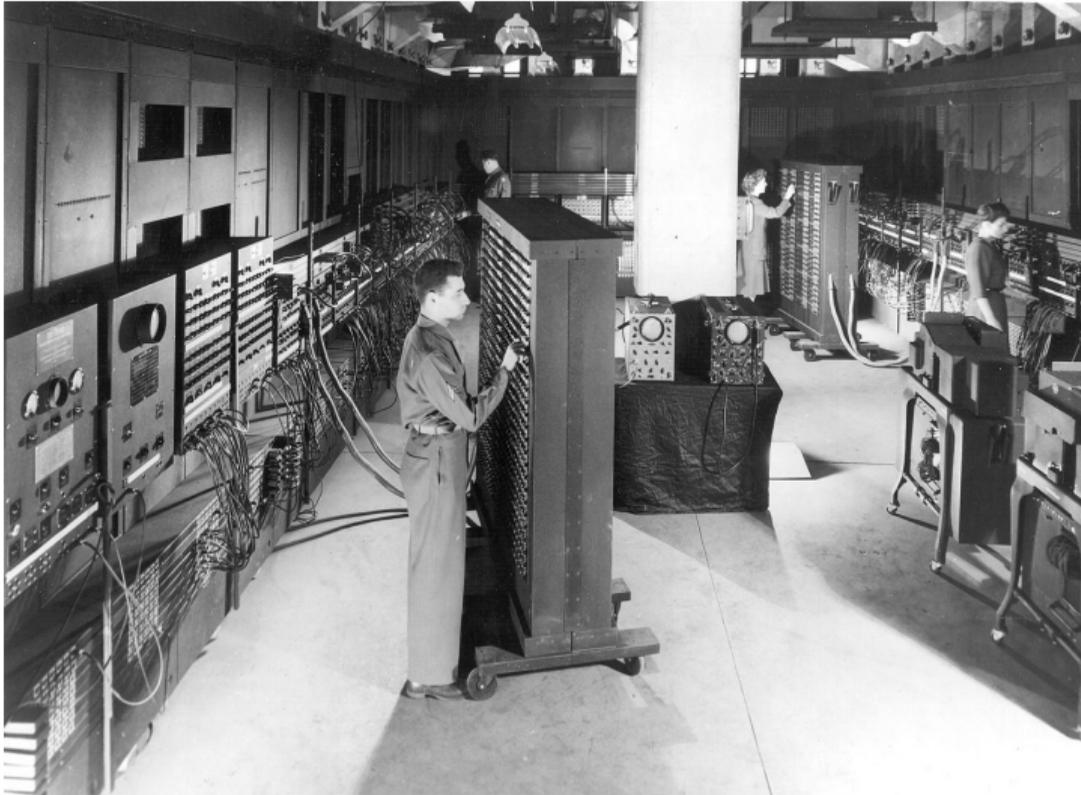


ENIAC – Electronic Numerical Integrator and Computer

J. Mauchly & J.P. Eckert, 1946: Röhren, Steckbrett-Programm

1.1 Einführung - Exkurs: Geschichte

64-040 Rechnerstrukturen und Betriebssysteme



First computer bug

92.

9/9

0800 Antan started { 1.2700 9.037 847 025
1000 " stopped - antan ✓ { 9.037 846 995 correct
13⁰⁰ (032) MP-MC ~~2.130476415~~ 4.615925059(-2)
(033) PRO 2 2.130476415
convd 2.130476415
Relays 6-2 in 033 failed speed speed test
for relay 11.00 test. Relay
214
Relay 3

1100 Started Cosine Tape (Sine check)
1525 Started Multy Adder Test.

1545  Relay #70 Panel F
(moth) in relay.

1650 Antan started.
1700 closed down.

First actual case of bug being found.





EDVAC

J. Mauchly, J.P. Eckert & J. von Neumann, 1949: Röhren, speicherprogrammiert

1.1 Einführung - Exkurs: Geschichte

64-040 Rechnerstrukturen und Betriebssysteme

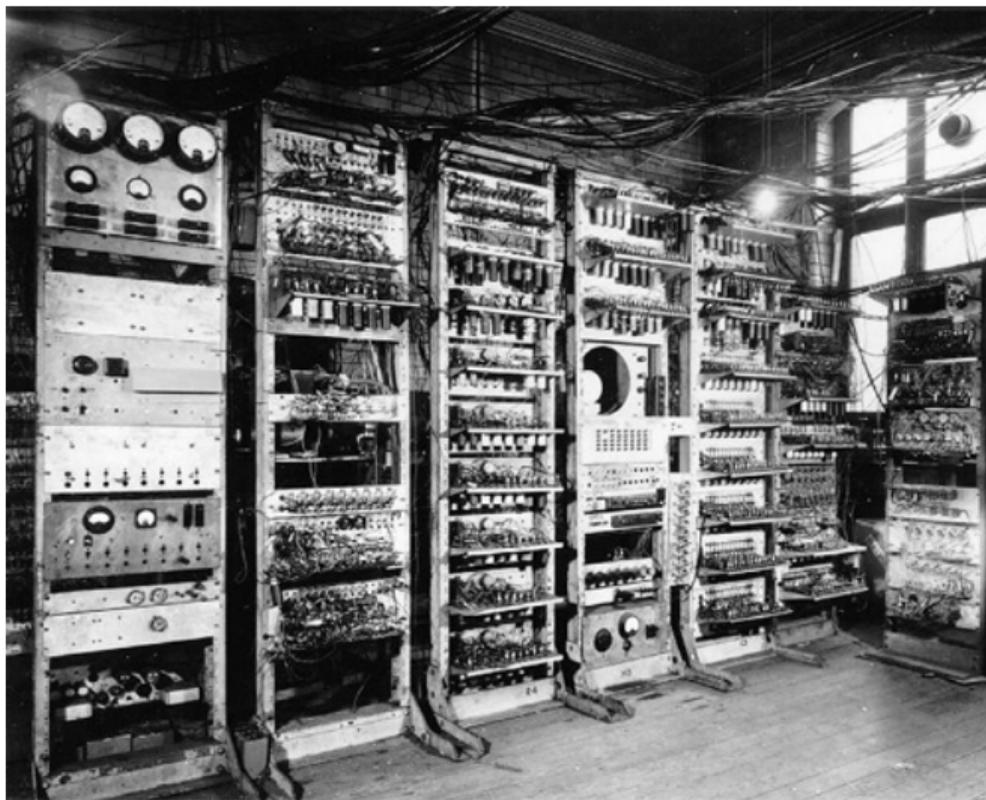


Manchester Mark-1

F.C. Williams & T. Kilburn, 1949: Trommelspeicher, Indexregister

1.1 Einführung - Exkurs: Geschichte

64-040 Rechnerstrukturen und Betriebssysteme

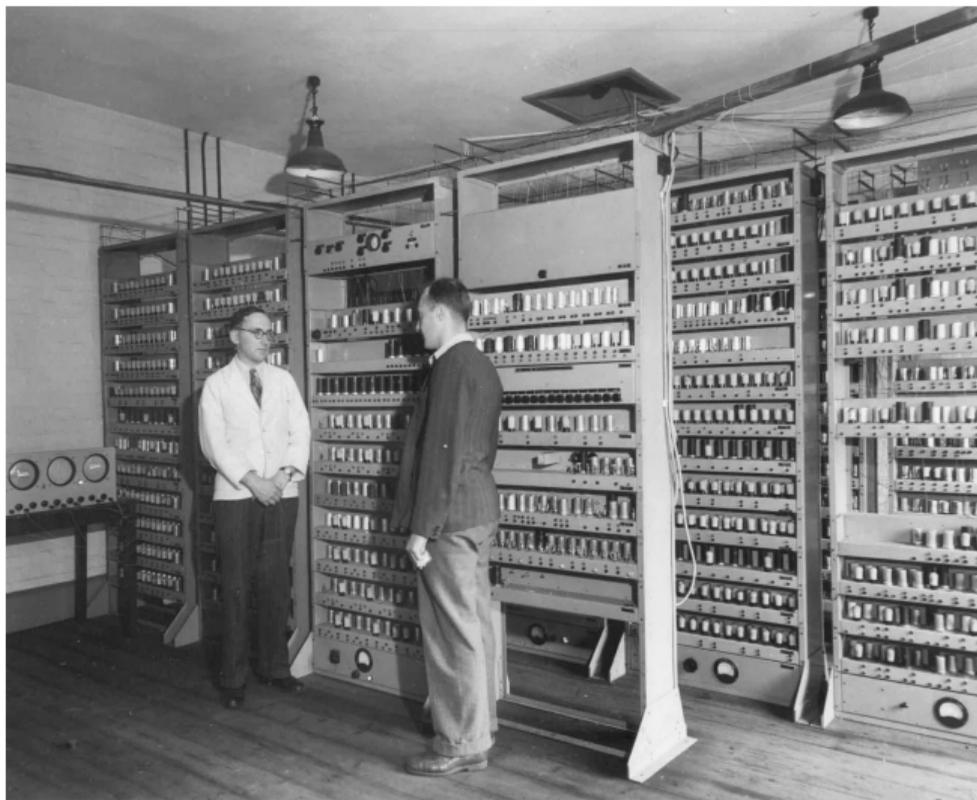


Manchester EDSAC

M. Wilkes 1951: Mikroprogrammierung, Unterprogramme, speicherprogrammiert

1.1 Einführung - Exkurs: Geschichte

64-040 Rechnerstrukturen und Betriebssysteme



- ▶ zunächst noch kaum Softwareunterstützung
- ▶ nur zwei Schichten:
 1. Programmierung in elementarer Maschinsprache (ISA level)
 2. Hardware in Röhrentechnik (device logic level)
 - Hardware kompliziert und unzuverlässig



Mikroprogrammierung (Maurice Wilkes, Cambridge, 1951):

- ▶ Programmierung in komfortabler Maschinsprache
- ▶ Mikroprogramm-Steuerwerk (Interpreter)
- ▶ einfache, zuverlässigere Hardware
- ▶ Grundidee der **CISC**-Rechner: VAX, 68000, 8086 etc.
(**C**omplex **I**nstruction **S**et **C**omputer)



- ▶ erste Rechner jeweils nur von einer Person benutzt
 - ▶ Anwender = Programmierer = Operator
 - ▶ Programm laden, ausführen, Fehler suchen usw.
- ⇒ Maschine wird nicht gut ausgelastet
- ⇒ Anwender mit lästigen Details überfordert

Einführung von **Betriebssystemen**

- ▶ „system calls“
- ▶ Batch-Modus: Programm abschicken, warten
- ▶ Resultate am nächsten Tag abholen



- ▶ Erfindung des Transistors 1948 (J. Bardeen, W. Brattain, W. Shockley)
- ▶ schneller, zuverlässiger, sparsamer als Röhren
- ▶ Miniaturisierung und dramatische Kostensenkung

- ▶ Beispiel: Digital Equipment Corporation PDP-1 (1961)
 - ▶ 4Ki Speicher (4096 Worte à 18-bit)
 - ▶ 200 KHz Taktfrequenz
 - ▶ 120 000 \$
 - ▶ Grafikdisplay: erste Computerspiele
- ▶ Nachfolger PDP-8: 16 000 \$
 - ▶ erstes Bussystem
 - ▶ 50 000 Stück verkauft



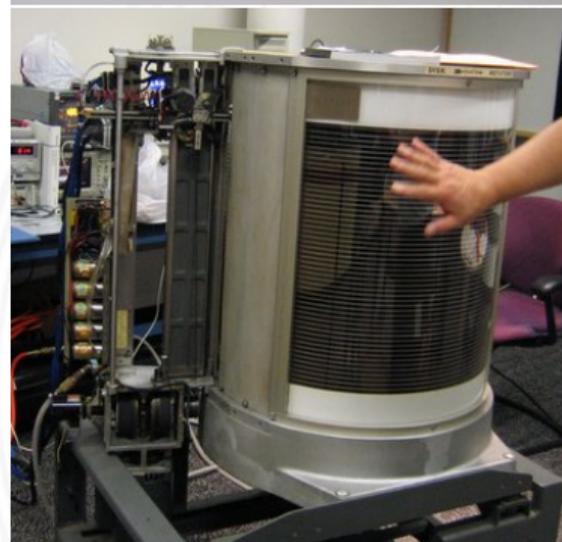
Massenspeicher bei frühen Computern

- ▶ Lochkarten
- ▶ Lochstreifen
- ▶ Magnetband

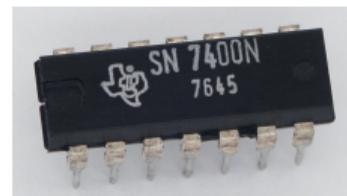
- ▶ Magnettrommel
- ▶ Festplatte

IBM 350 RAMAC (1956)

5 MByte, 600 ms Zugriffszeit

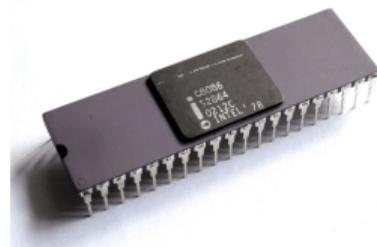


- ▶ Erfindung der integrierten Schaltung 1958 (R. Noyce, J. Kilby)
- ▶ Dutzende... Hunderte... Tausende Transistoren auf einem Chip
- ▶ IBM Serie-360: viele Maschinen, ein einheitlicher Befehlssatz
- ▶ volle Softwarekompatibilität



Eigenschaft	Model 30	Model 40	Model 50	Model 65
Rel. Leistung [Model 30]	1	3,5	10	21
Zykluszeit [ns]	1 000	625	500	250
Max. Speicher [KiB]	64	256	256	512
Pro Zyklus gelesene Byte	1	2	4	16
Max. Anzahl von Datenkanälen	3	3	4	6

- ▶ VLSI = *Very Large Scale Integration*
- ▶ ab 10 000 Transistoren pro Chip
- ▶ gesamter Prozessor passt auf einen Chip
- ▶ steigende Integrationsdichte erlaubt immer mehr Funktionen



1972 Intel 4004: erster Mikroprozessor

1975 Intel 8080, Motorola 6800, MOS 6502 ...

1981 IBM PC („personal computer“) mit Intel 8088

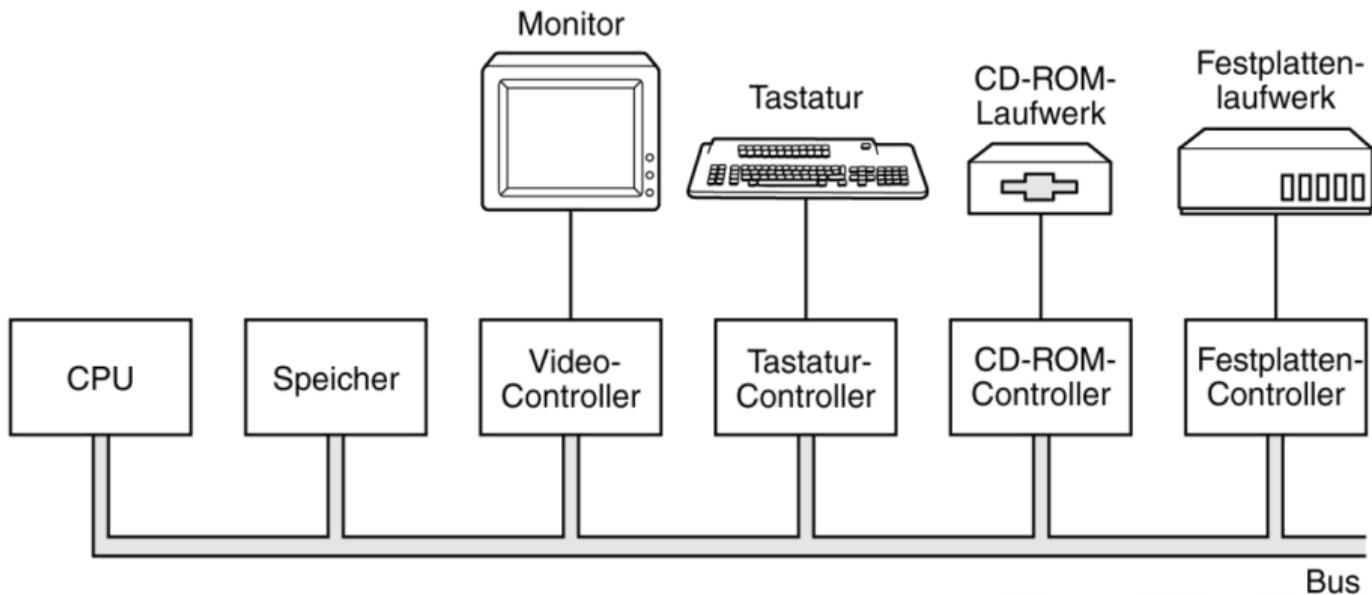
...

- ▶ Massenfertigung erlaubt billige Prozessoren (< 1\$)
- ▶ Miniaturisierung ermöglicht mobile Geräte

Xerox Alto: first workstation (1973)



Personal Computer: Aufbau des IBM PC (1981)

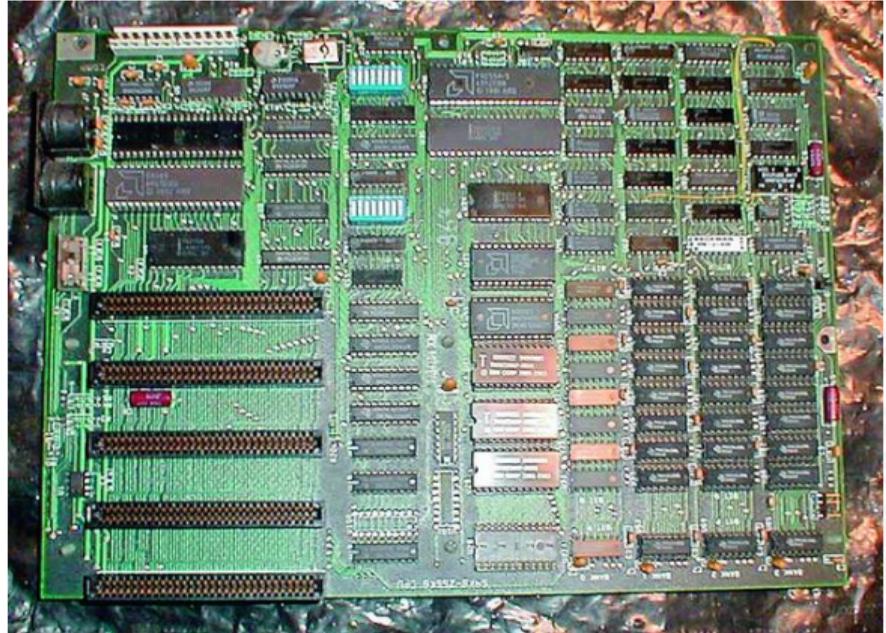
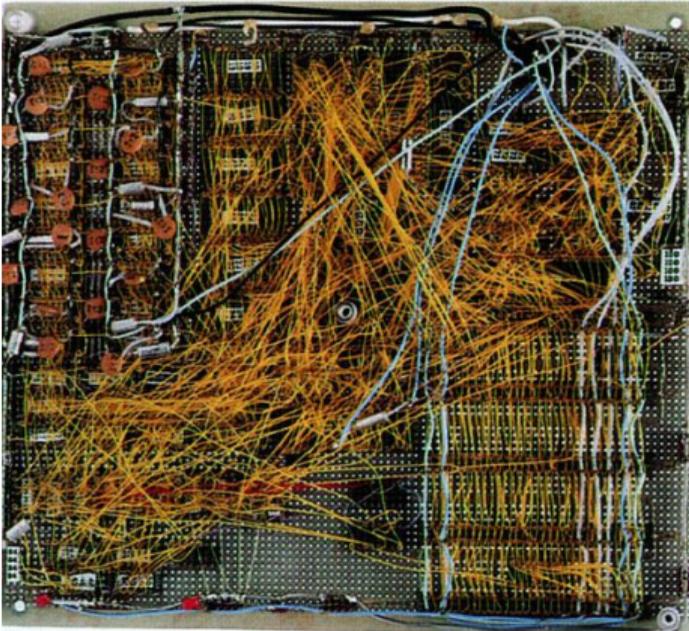


- ▶ Intel 8086/8088, 512 KByte RAM, Betriebssystem MS-DOS
- ▶ alle Komponenten über den zentralen (ISA-) Bus verbunden
- ▶ Erweiterung über Einsteckkarten

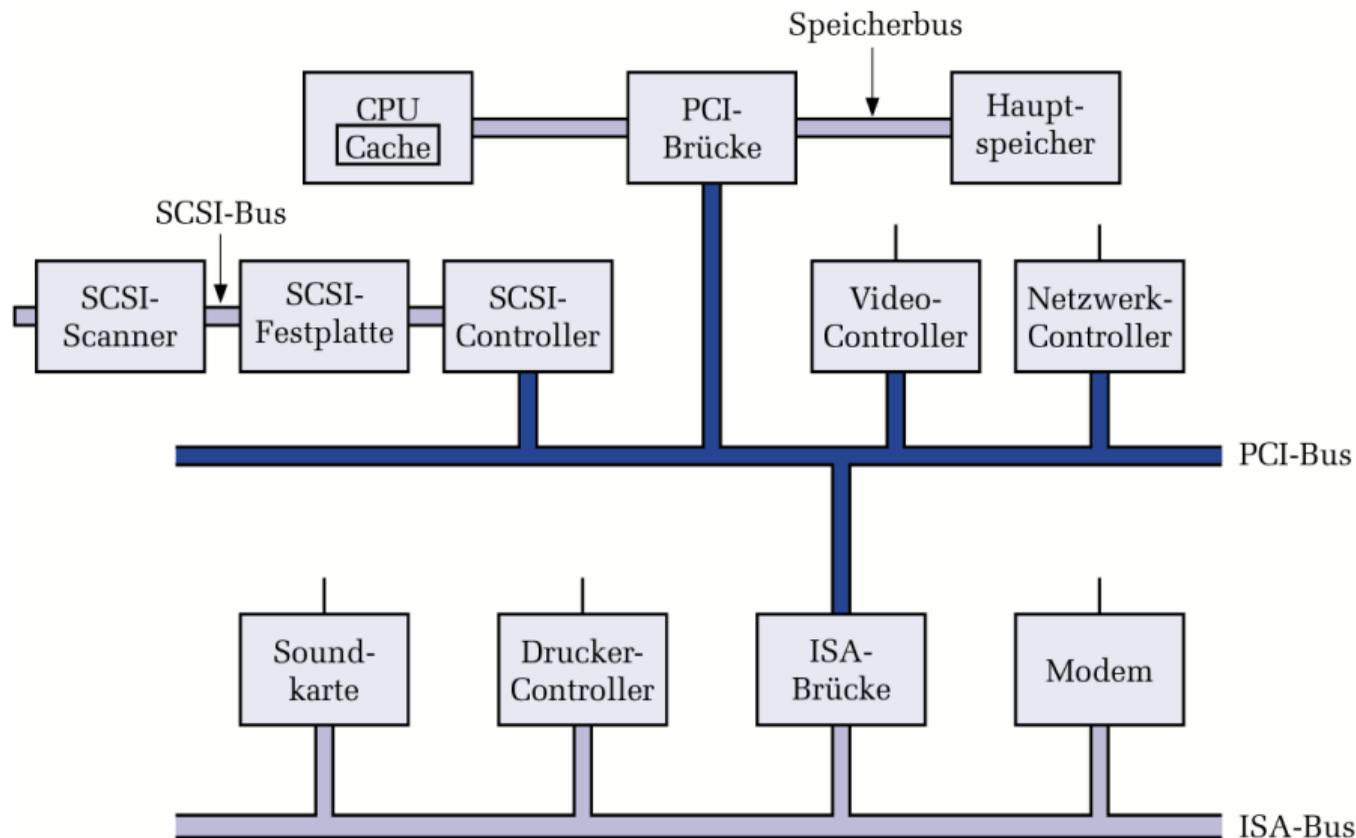
PC Prototyp (1981) und Hauptplatine

1.2 Einführung - Personal Computer

64-040 Rechnerstrukturen und Betriebssysteme

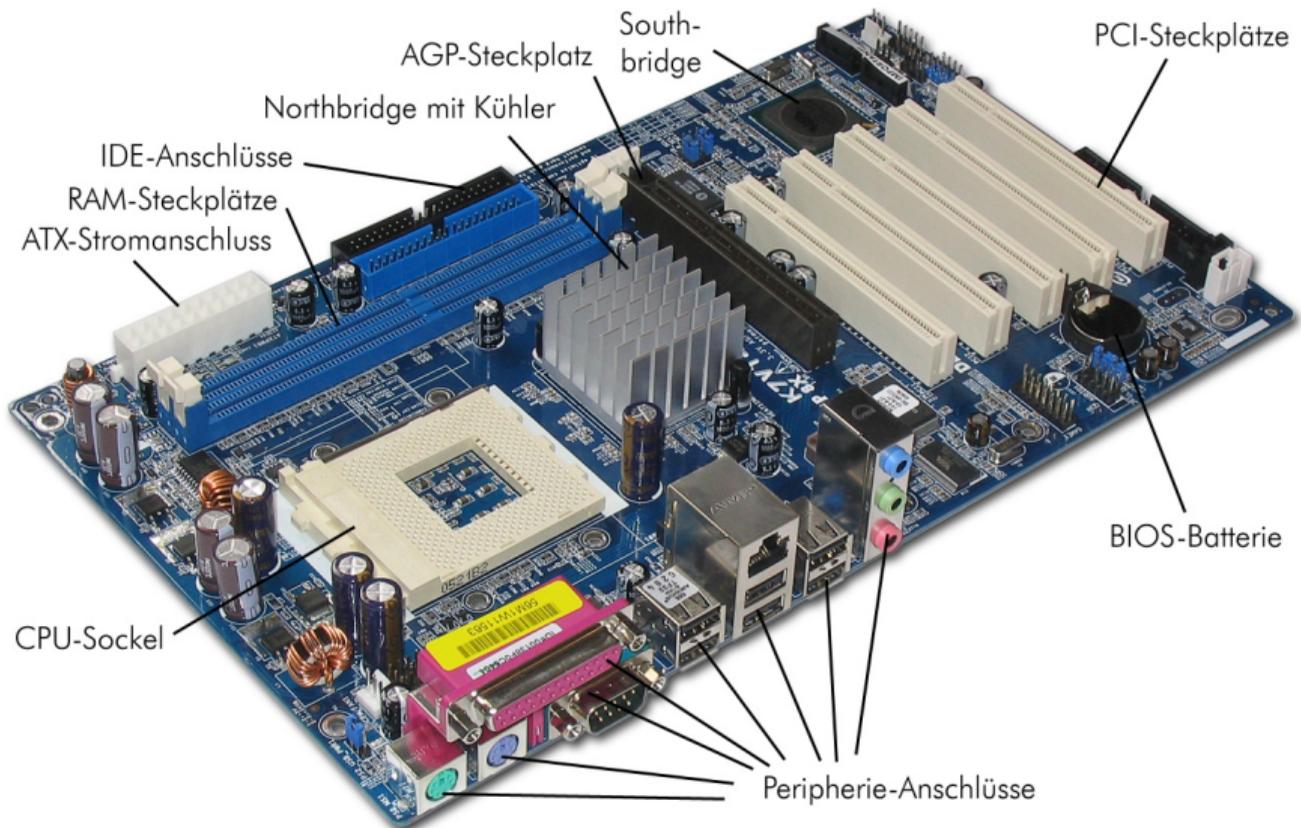


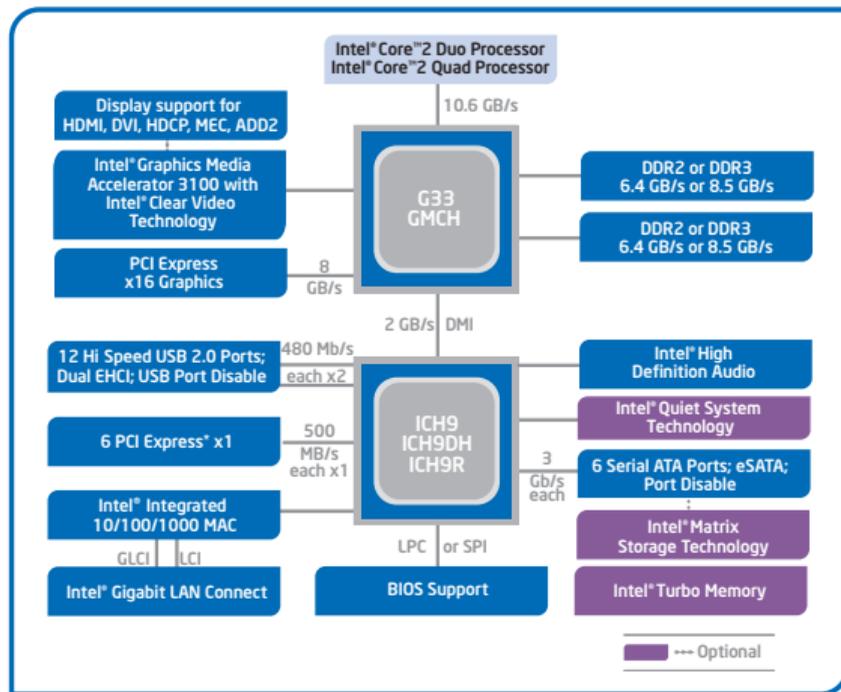
Aufbau mit PCI-Bus (2000)



[TA14]

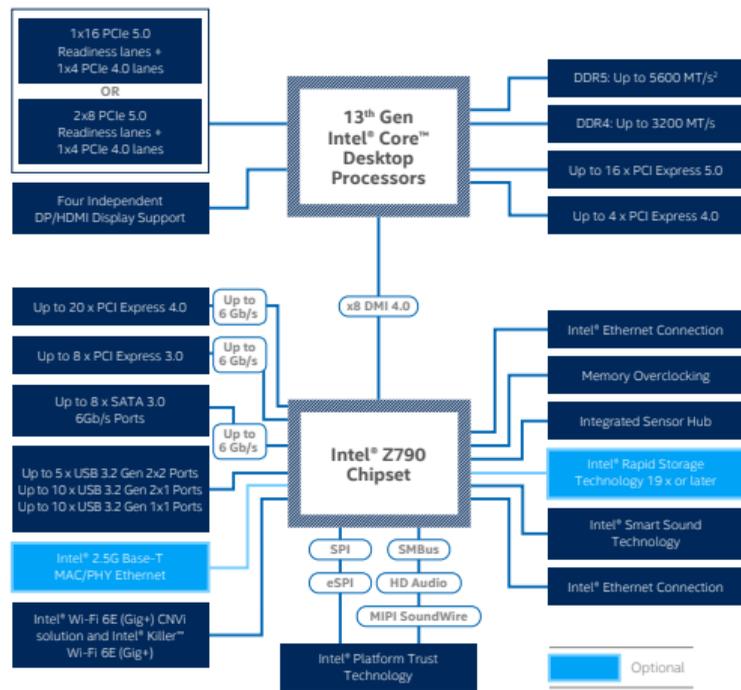
Hauptplatine (2005)





Intel ark.intel.com

- ▶ Mehrkern-Prozessoren („dual-/quad-/octa-core“)
- ▶ schnelle serielle Direktverbindungen statt PCI/ISA Bus



Intel ark.intel.com

- ▶ Speichercontroller und externe Anbindung (PCI Express) in CPU
- ▶ Grafikprozessor in CPU

► Anzahl an Systemen / Prozessoren – weltweit

System	Anzahl (geschätzt!)
PCs, Workstation, Server	2 Milliarden
Tablets	1,3 Milliarden
Smartphones	4,8 Milliarden
„Embedded Systems“	75-100 Milliarden

► Preis des Prozessors

Typ	Preis [\$]	Beispielanwendung
Wegwerfcomputer	0,5	Glückwunschkarten
Mikrocontroller	5	Uhren, Geräte, Autos
Mobile Computer und Spielkonsolen	50	Smartphones, Tablets, Heimvideospiele
Personalcomputer	500	Desktop- oder Notebook-Computer
Server	5 000	Netzwerkserver
Workstation Verbund	50 000 – 500 000	Abteilungsrechner (Minisupercomp.)
Großrechner (Mainframe)	5 Millionen	Batch-Verarbeitung in einer Bank
Supercomputer	> 50 Millionen	Klimamodelle, Simulationen

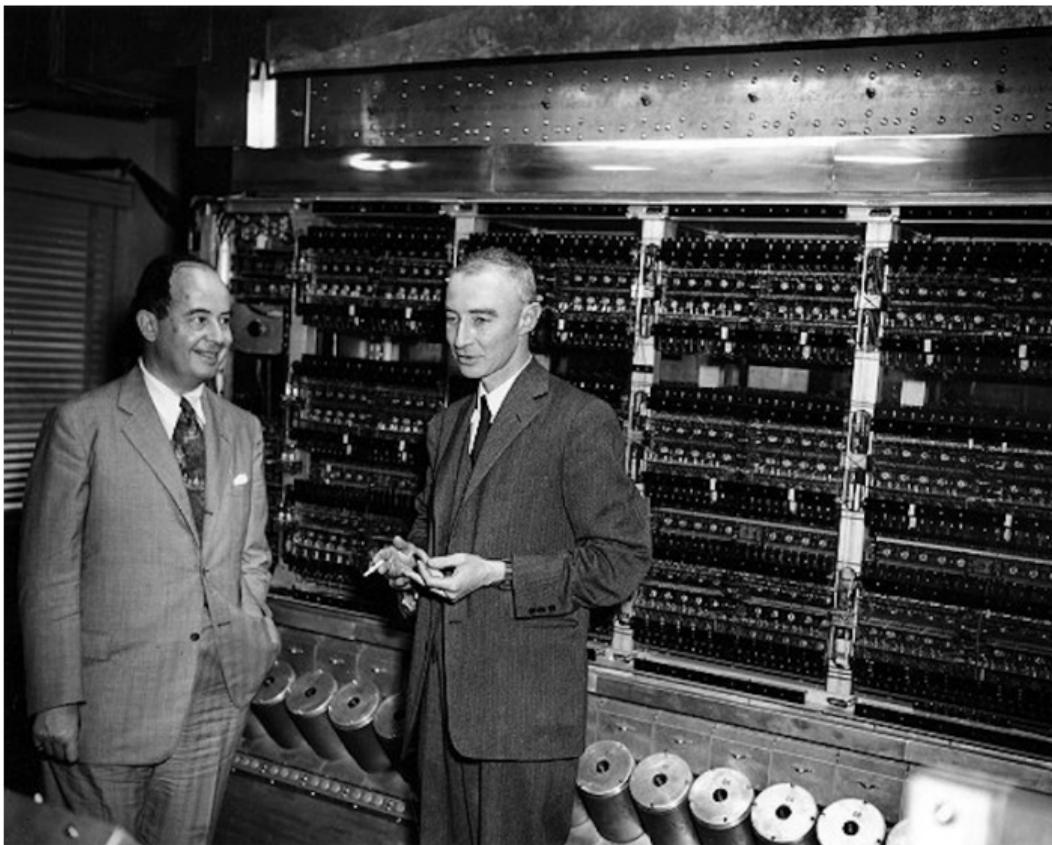


- ▶ J. Mauchly, J.P. Eckert, J. von-Neumann 1945
 - ▶ Abstrakte Maschine mit minimalem Hardwareaufwand
 - ▶ System mit Prozessor, Speicher, Peripheriegeräten
 - ▶ die Struktur ist unabhängig von dem Problem, das Problem wird durch austauschbaren Speicherinhalt (Programm) beschrieben
 - ▶ gemeinsamer Speicher für Programme und Daten
 - ▶ fortlaufend adressiert
 - ▶ Programme können wie Daten manipuliert werden
 - ▶ Daten können als Programm ausgeführt werden
 - ▶ Befehlszyklus: Befehl holen, decodieren, ausführen
- ⇒ enorm flexibel
- ▶ **alle** aktuellen Rechner basieren auf diesem Prinzip
 - ▶ aber vielfältige Architekturvarianten, Befehlssätze usw.

von-Neumann Rechner: IAS Computer

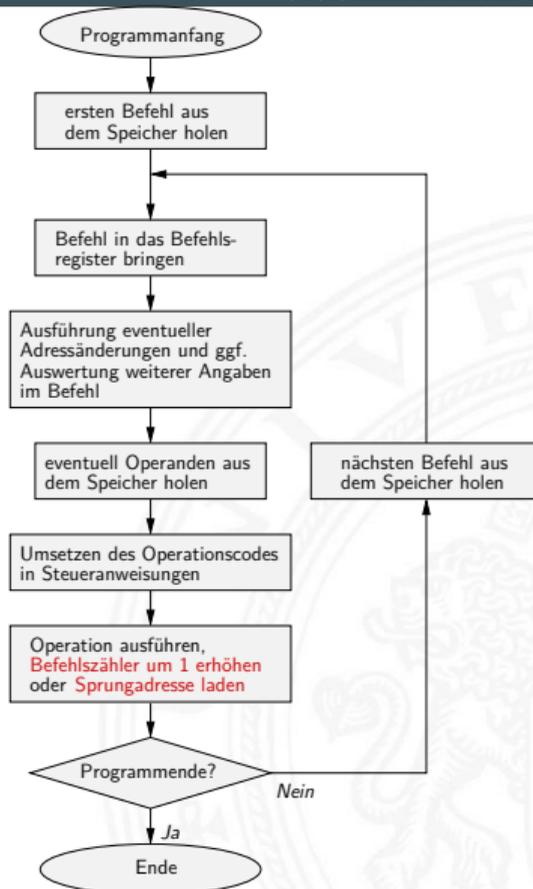
1.3 Einführung - von-Neumann Konzept

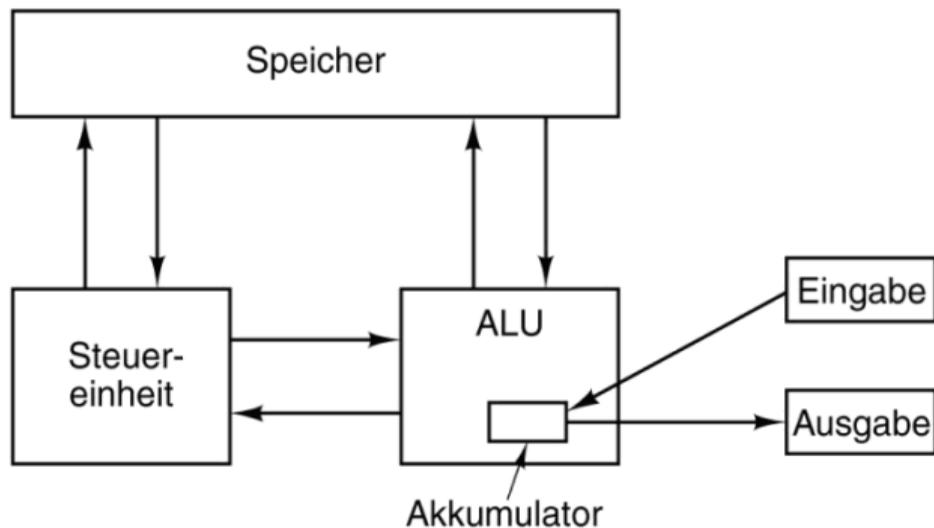
64-040 Rechnerstrukturen und Betriebssysteme



John von Neumann, R. J. Oppenheimer,
IAS Computer Princeton
computerhistory.org

- ▶ Programm als Sequenz elementarer Anweisungen (Befehle)
- ▶ als Bitvektoren im Speicher codiert
- ▶ Interpretation (Operanden, Befehle und Adressen) ergibt sich aus dem Kontext (der Adresse)
- ▶ zeitsequenzielle Ausführung der Instruktionen



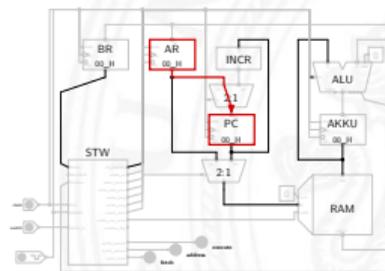
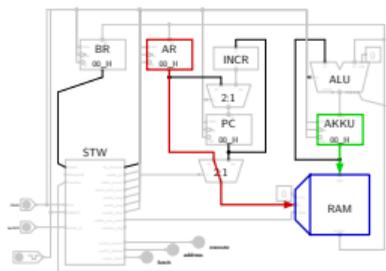
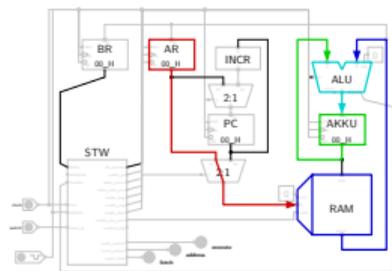
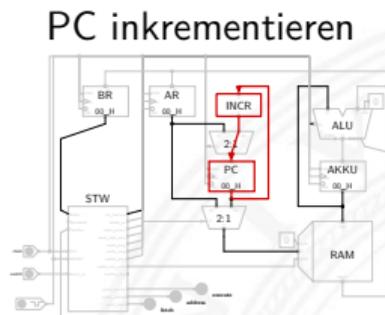
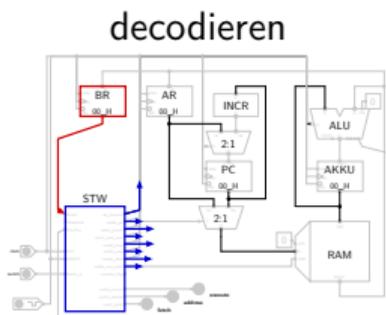
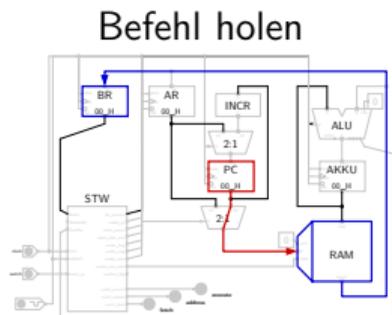


[TA14]

Fünf zentrale Komponenten:

- ▶ Prozessor mit **Steuerwerk** und **Rechenwerk** (ALU, Register)
- ▶ **Speicher**, gemeinsam genutzt für Programme und Daten
- ▶ **Eingabe-** und **Ausgabewerke**
- ▶ verbunden durch Bussystem

- ▶ Verschaltung der Hardwarekomponenten für alle mögl. Datentransfers
- ▶ abhängig vom Befehl werden nur bestimmte Pfade aktiv
- ▶ Ausführungszyklus



rechnen

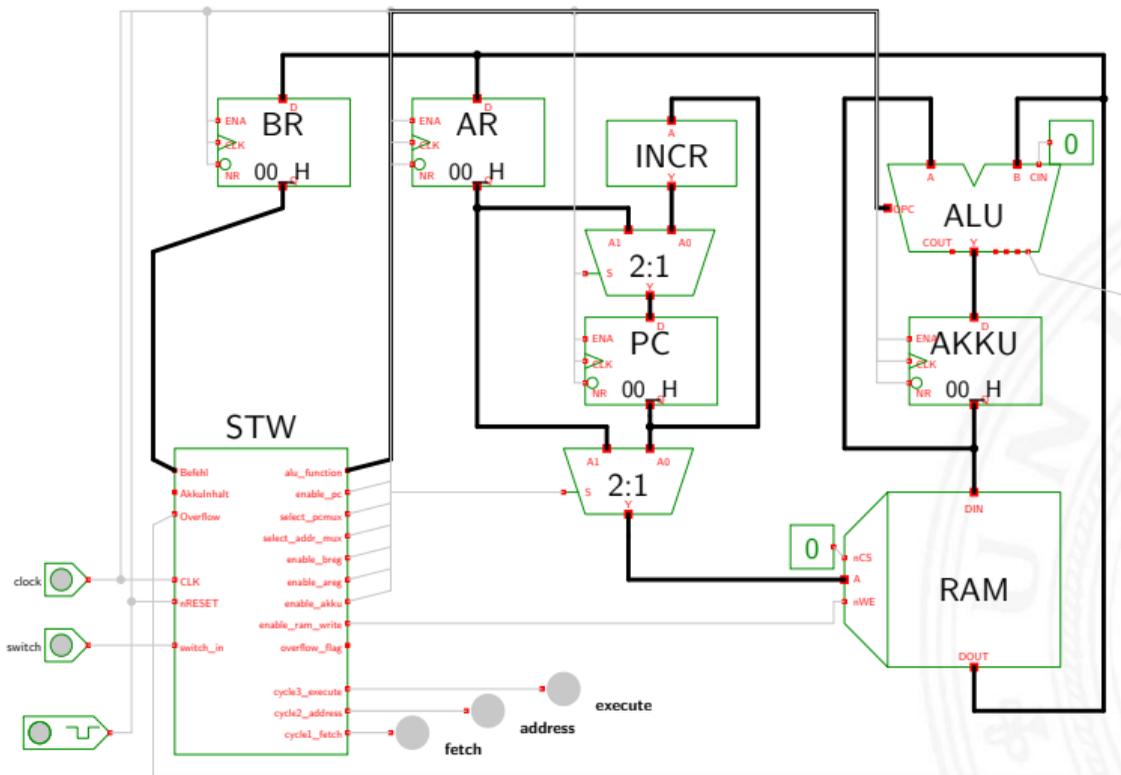
speichern

springen

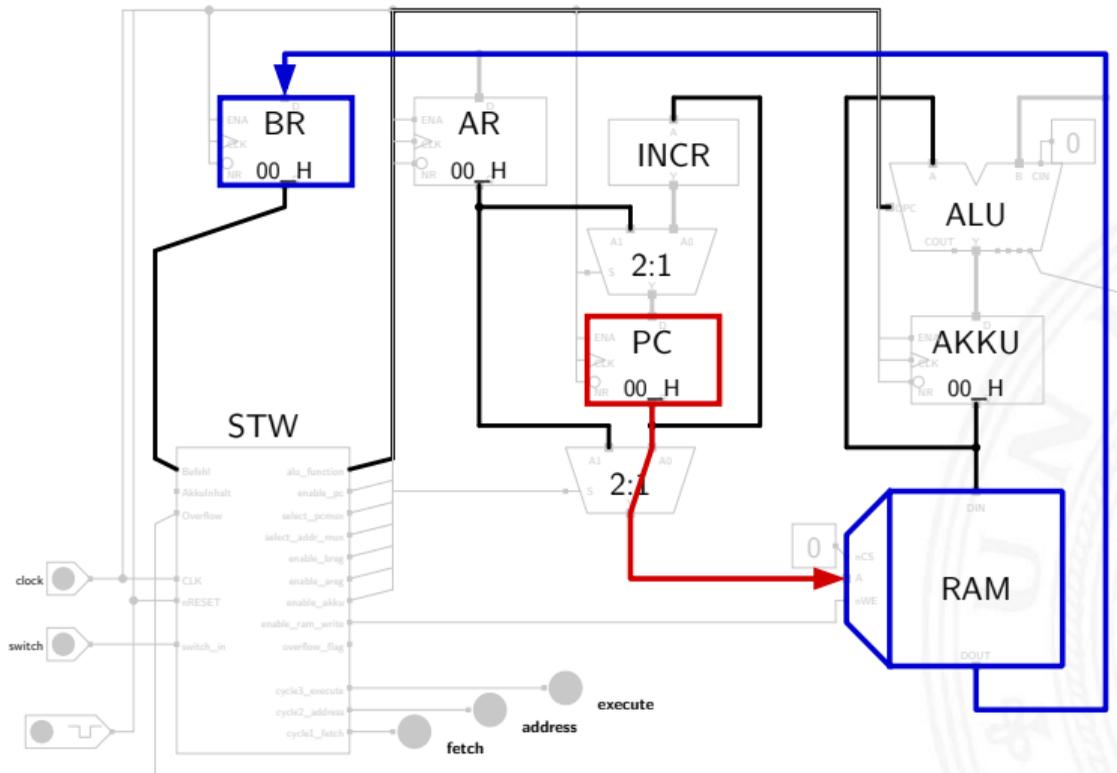
Beispiel: PRIMA (die primitive Maschine)

- ▶ ein (minimaler) 8-bit von-Neumann Rechner

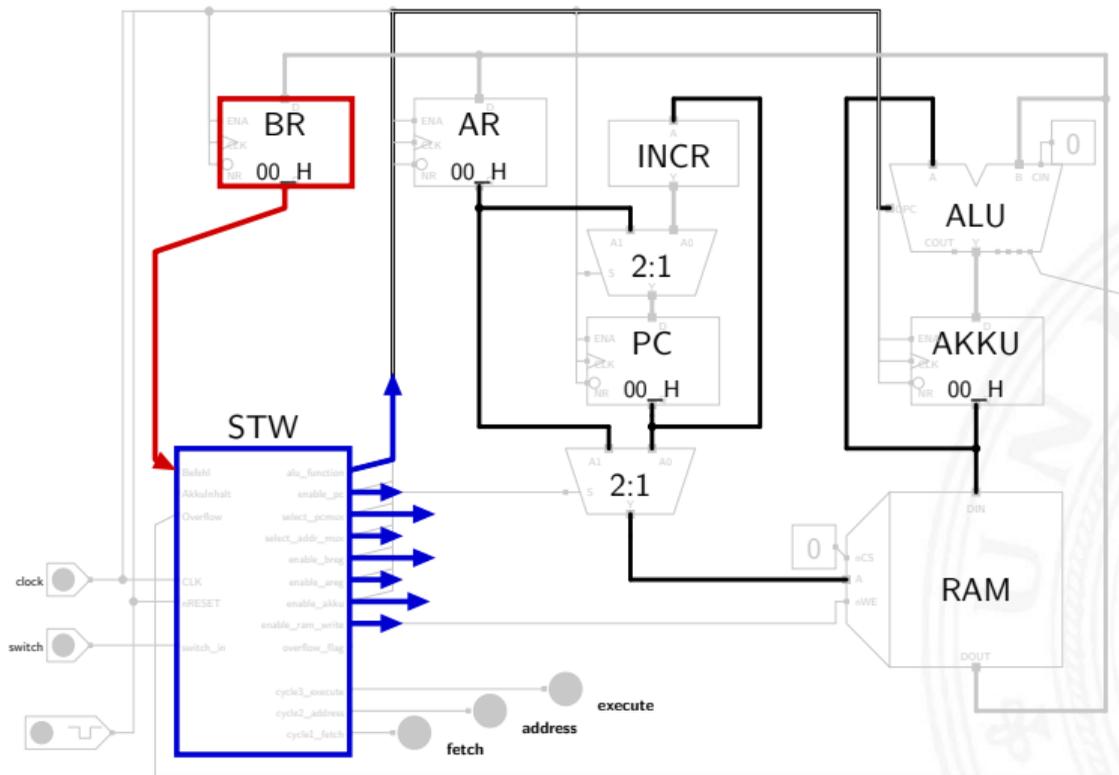
[HenHA] Hades Demo: 50-rtlib/90-prima/prima



► $BR = RAM[PC]$

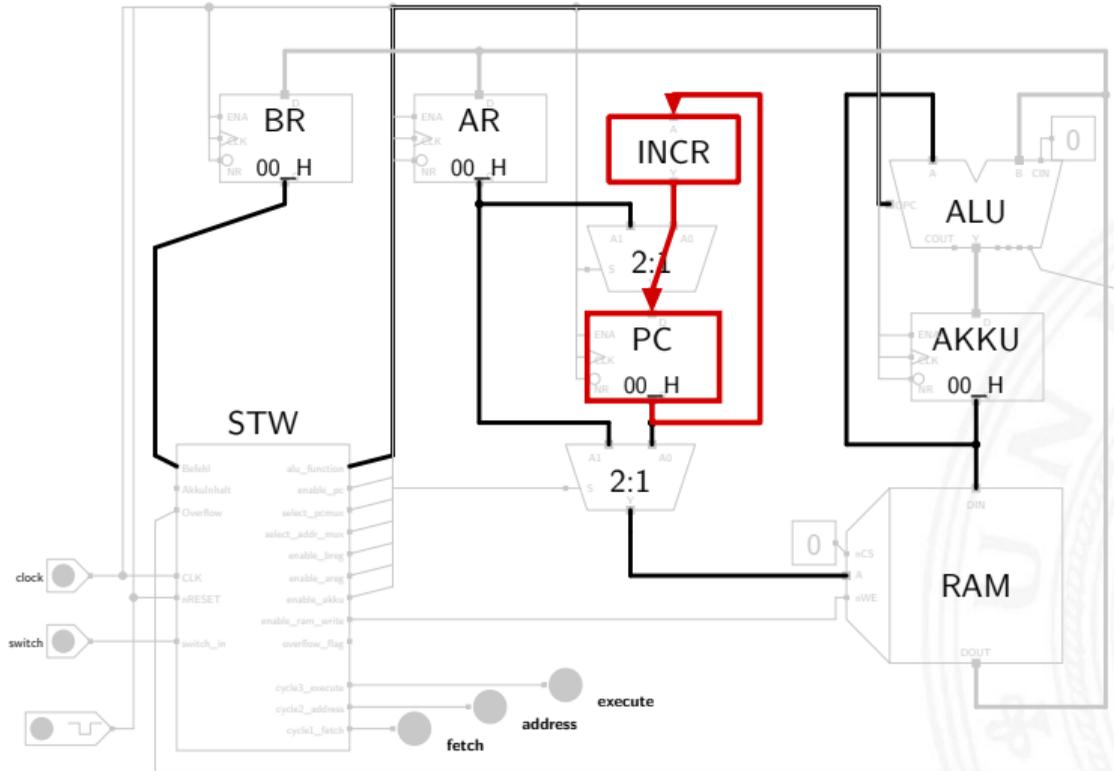


► Steuersignale = decode(BR)

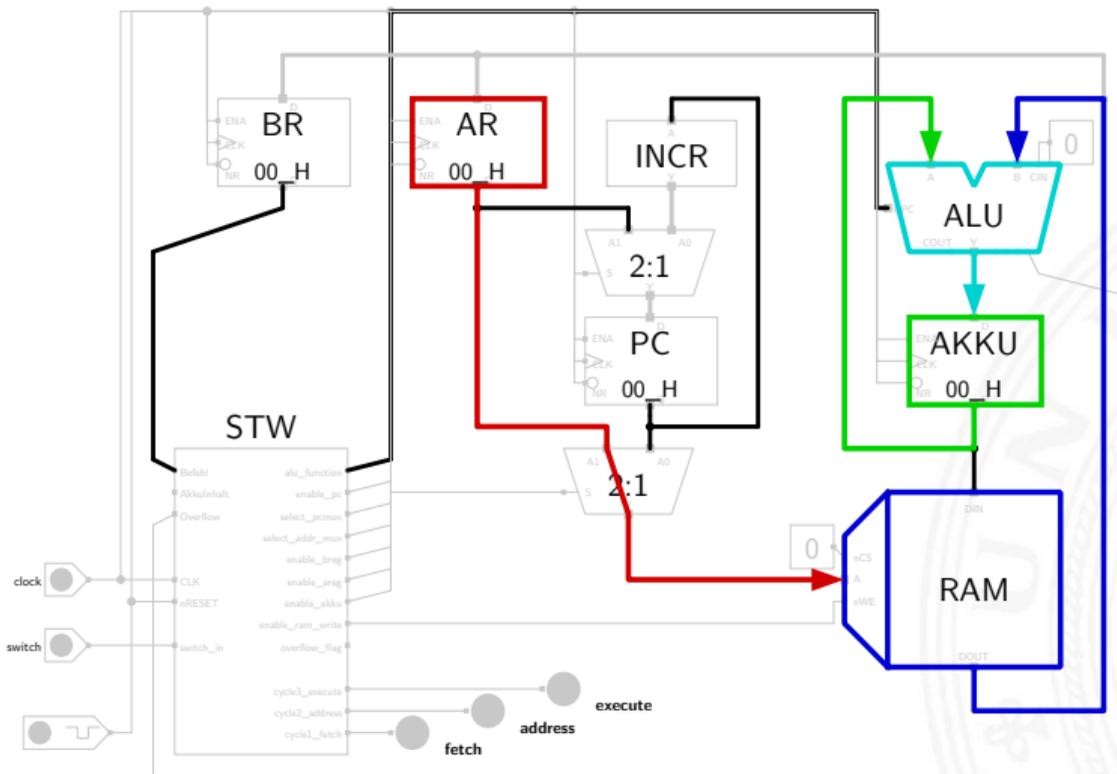


PRIMA: PC inkrementieren

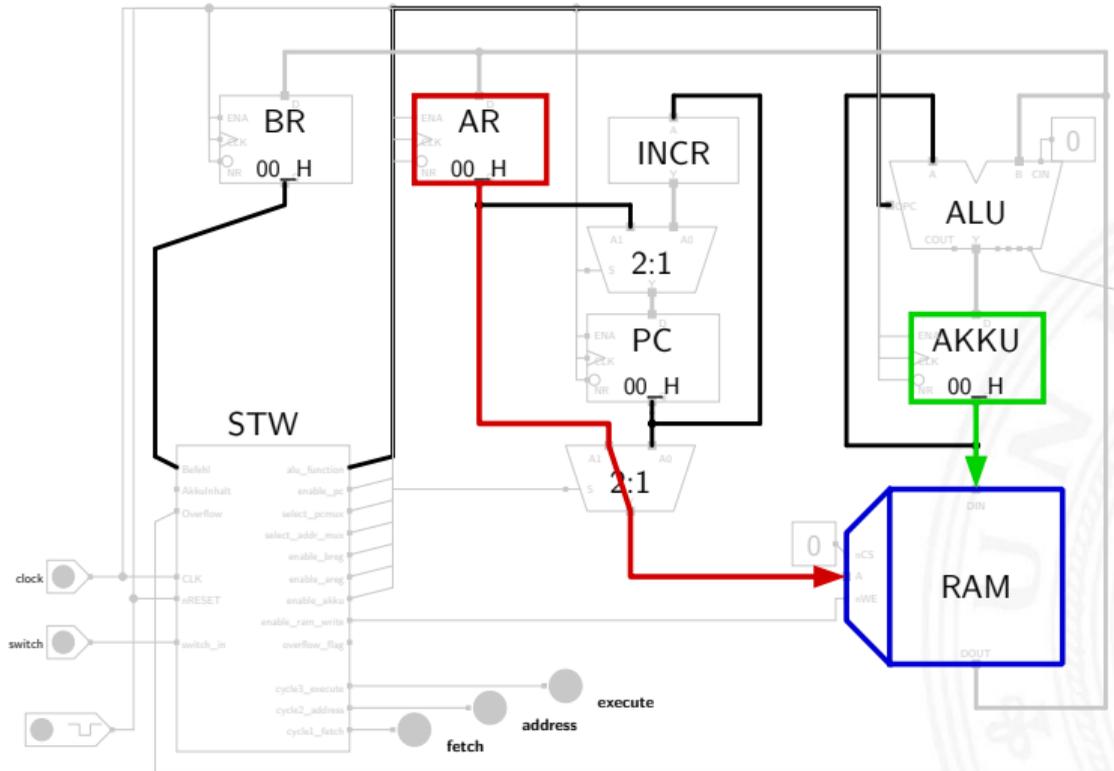
► $PC = PC + 1$



► Akku = Akku + RAM[AR]

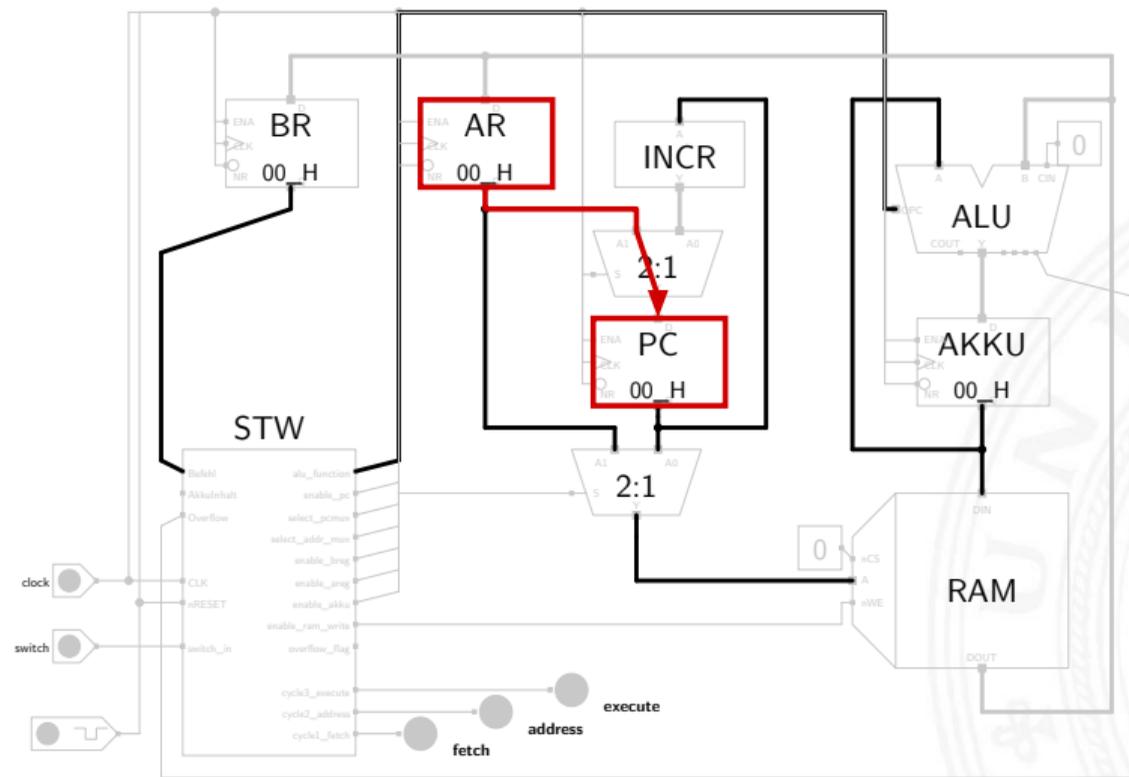


► $RAM[AR] = Akku$



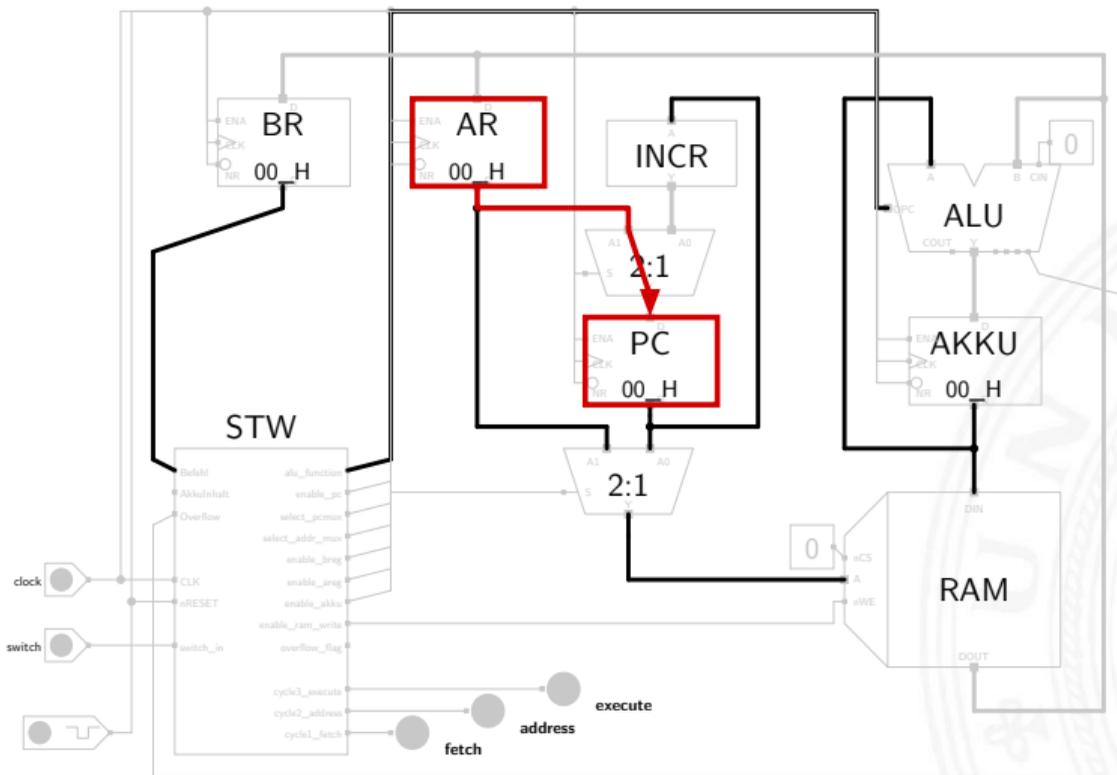
PRIMA: springen

► PC = AR



► PC = AR

später dazu mehr...





- ▶ bessere Technologie ermöglicht immer kleinere Transistoren
 - ▶ Kosten (Material/Entwurf/Fertigung) sind proportional zur Chipfläche
- ⇒ bei gleicher Funktion kleinere und billigere Chips
- ⇒ bei gleicher Größe leistungsfähigere Chips

Moore's Law

Gordon Moore, Mitgründer von Intel, 1965

Speicherkapazität von ICs vervierfacht sich alle drei Jahre

- ⇒ schnelles **exponentielles Wachstum**
- ▶ klares Kostenoptimum bei hoher Integrationsdichte
 - ▶ trifft auch auf Prozessoren zu

Moore's Law (cont.)

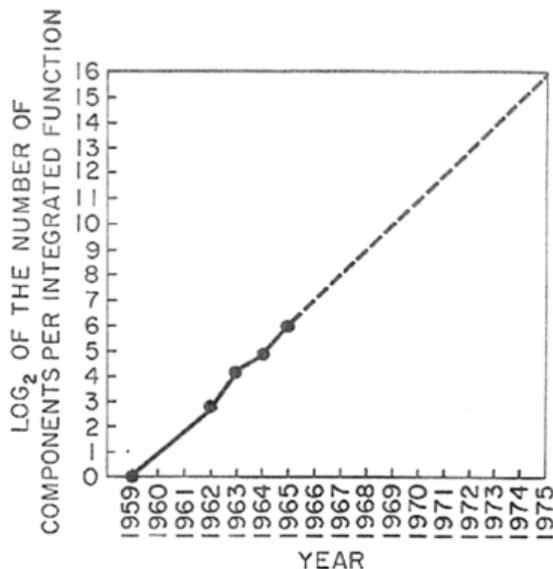


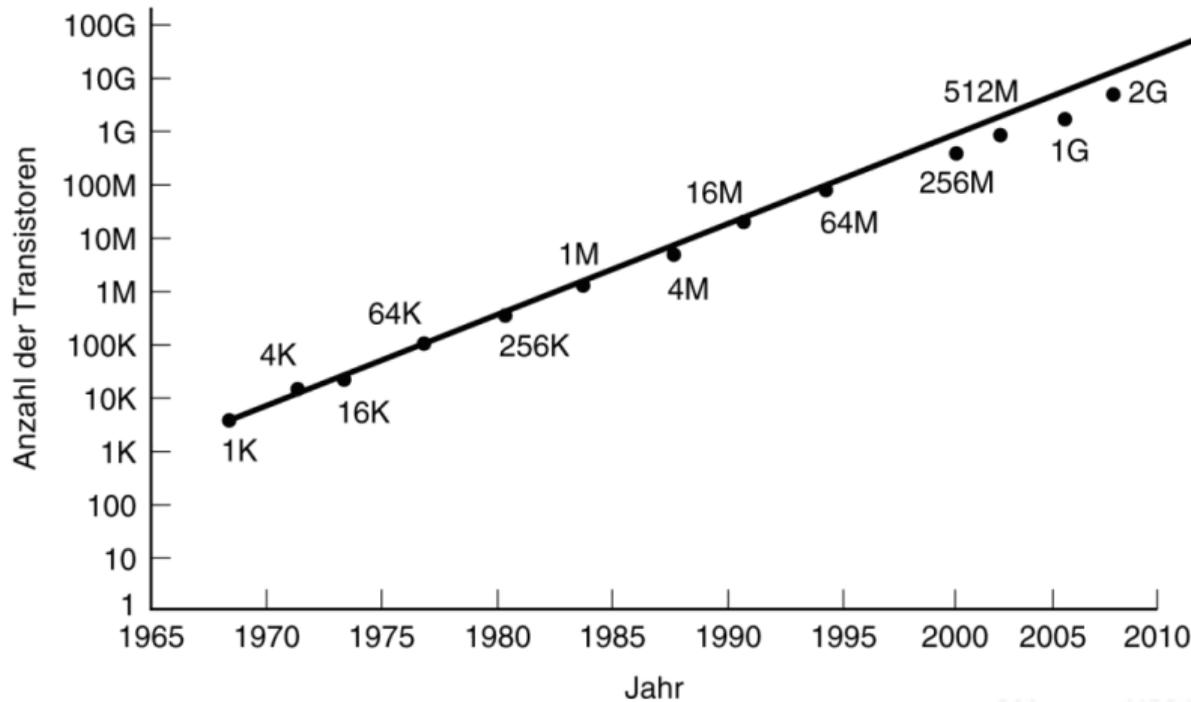
Fig. 2 Number of components per integrated function for minimum cost per component extrapolated vs time.

Gordon Moore, 1965, [Moo65]:
Cramming more components onto integrated circuits

Wird das so weitergehen?

- ▶ Vorhersage gilt immer noch
- ▶ „IRDS“ Prognosen bis zum Jahr 2037 [IRDS22]

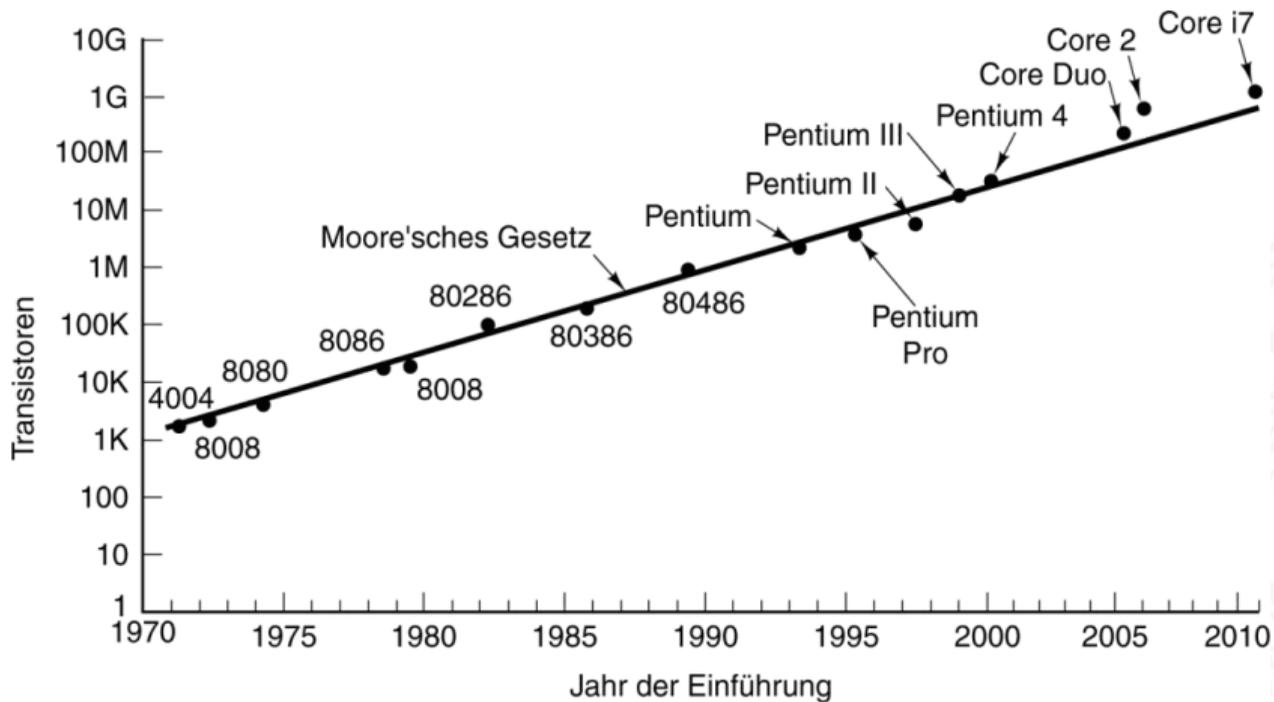
Moore's Law: Transistoren pro Speicherchip



[TA14]

- ▶ Vorhersage: 60% jährliches Wachstum der Transistoranzahl pro IC
Verdopplung alle 18 Monate (12...24 Monate)

Moore's Law: Evolution der Prozessoren



[TA14]

Moore's Law: Evolution der Prozessoren (cont.)



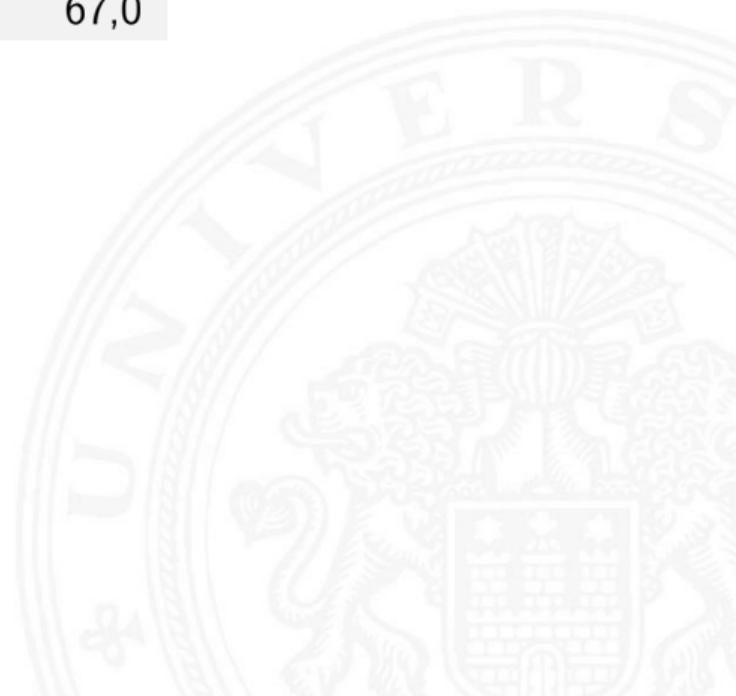
[Intel] Intel Innovation 2023



Moore's Law: Evolution der Prozessoren (cont.)

Transistoren pro IC (monolithisch)

Modell		Typ	Jahr	Trans. [Mrd.]
M2 Max	Apple	CPU	2023	67,0

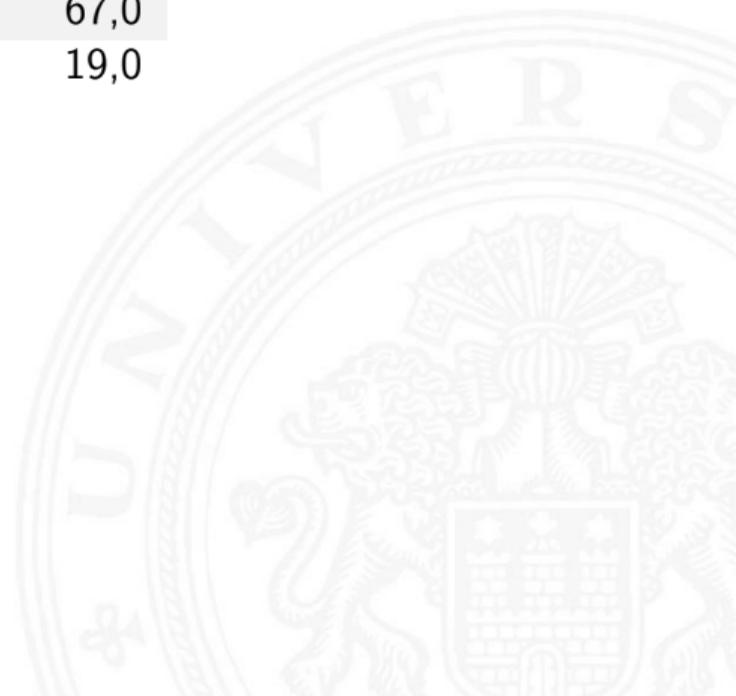




Moore's Law: Evolution der Prozessoren (cont.)

Transistoren pro IC (monolithisch)

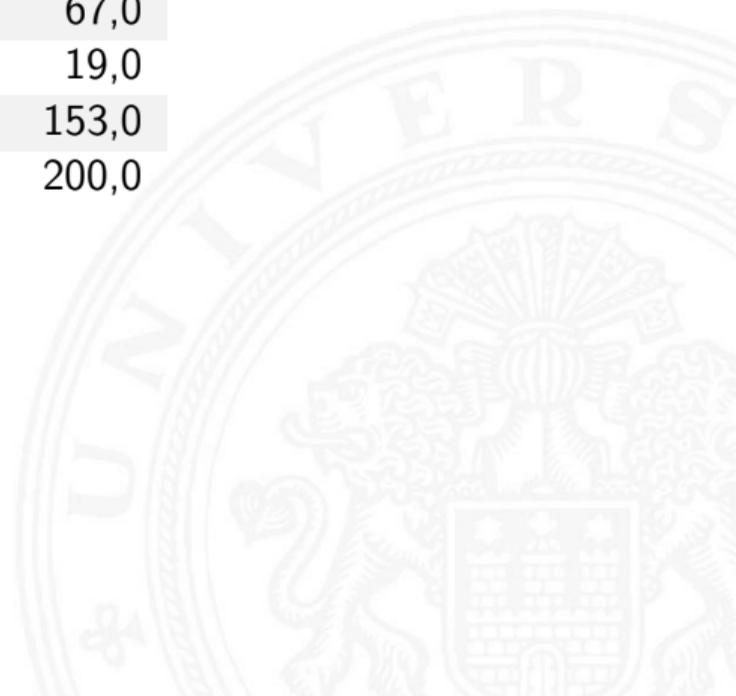
Modell		Typ	Jahr	Trans. [Mrd.]
M2 Max	Apple	CPU	2023	67,0
A17	Apple	SOC	2023	19,0





Transistoren pro IC (monolithisch)

Modell		Typ	Jahr	Trans. [Mrd.]
M2 Max	Apple	CPU	2023	67,0
A17	Apple	SOC	2023	19,0
MI300X	AMD	GPU	2023	153,0
GH200	Nvidia	GPU	2023	200,0

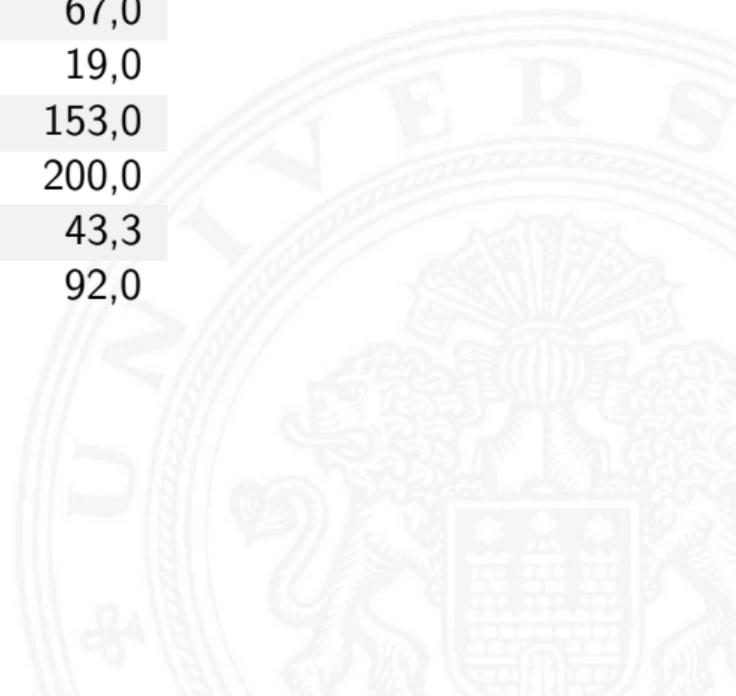




Moore's Law: Evolution der Prozessoren (cont.)

Transistoren pro IC (monolithisch)

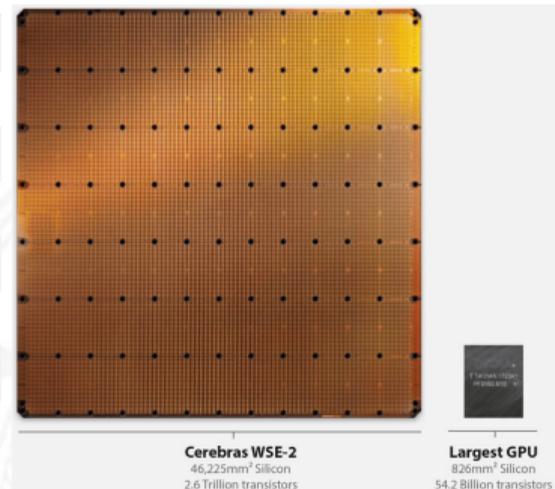
Modell		Typ	Jahr	Trans. [Mrd.]
M2 Max	Apple	CPU	2023	67,0
A17	Apple	SOC	2023	19,0
MI300X	AMD	GPU	2023	153,0
GH200	Nvidia	GPU	2023	200,0
Stratix 10	Intel (Altera)	FPGA	2019	43,3
VP 1802	AMD (Xilinx)	FPGA	2021	92,0



Moore's Law: Evolution der Prozessoren (cont.)

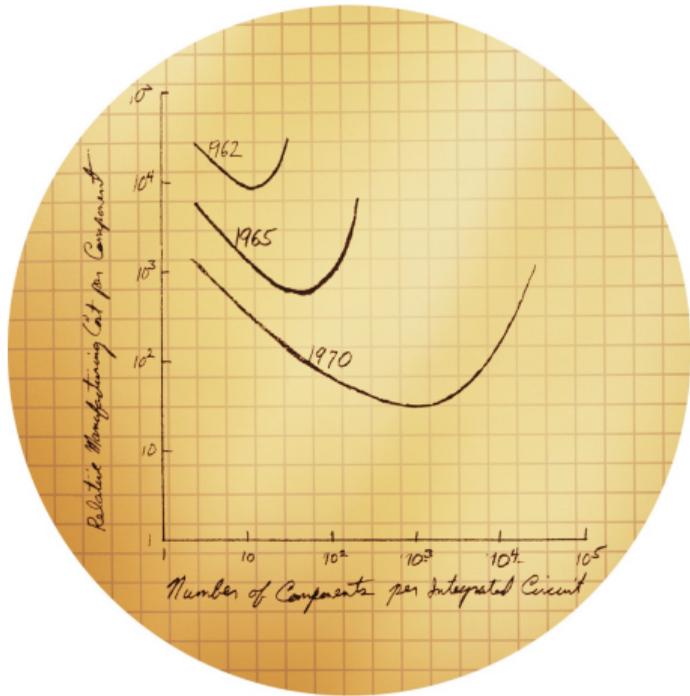
Transistoren pro IC (monolithisch)

Modell		Typ	Jahr	Trans. [Mrd.]
M2 Max	Apple	CPU	2023	67,0
A17	Apple	SOC	2023	19,0
MI300X	AMD	GPU	2023	153,0
GH200	Nvidia	GPU	2023	200,0
Stratix 10	Intel (Altera)	FPGA	2019	43,3
VP 1802	AMD (Xilinx)	FPGA	2021	92,0
WSE-2	Cerebras	WSc	2021	2 600,0



[www.cerebras.net]

Moore's Law: Kosten pro Komponente



Originalskizze von G. Moore [Intel]



$$L(t) = L(0) \cdot 2^{t/18}$$

mit: $L(t)$ = Leistung zum Zeitpunkt t ,
 $L(0)$ = Leistung zum Zeitpunkt 0,
und Zeit t in Monaten.

Einige Formelwerte:

Jahr 1:	1,5874
Jahr 2:	2,51984
Jahr 3:	4
Jahr 5:	10,0794
Jahr 6:	16
Jahr 7:	25,3984
Jahr 8:	40,3175



Leistungssteigerung der Spitzenrechner seit 1993

www.top500.org de.wikipedia.org/wiki/Supercomputer

Jahr	Rechner	CPU	Linpack	[TFlop/s]	Prozessoren
1993	TMC CM-5/1024	(SuperSparc 32MHz)		0,0597	1 024
1994	Intel XP/S140	(80860 50MHz)		0,1434	3 680
1995	Fujitsu NWT	(105 MHz)		0,17	140
1996	Hitachi SR2201/1024	(HARP-1E 120MHz)		0,2204	1 024
1997	Intel ASCI Red	(Pentium Pro 200MHz)		1,068	7 264
1999	Intel ASCI Red	(Pentium Pro 333MHz)		2,121	9 472
2001	IBM ASCI White	(Power3 375MHz)		7,226	8 192
2002	NEC Earth Simulator	(NEC 1GHz)		35,86	5 120
2005	IBM BlueGene/L	(PowerPC 440 2C 700MHz)		136,8	65 536
2006	IBM BlueGene/L	(PowerPC 440 2C 700MHz)		280,6	131 072
2008	IBM Roadrunner	(Opteron 2C 1,8GHz + IBM Cell 9C 3,2 GHz)		1 026,0	122 400
2010	Cray XT5-HE Jaguar	(Opteron 6C 2,6GHz)		1 759,0	224 162
2011	Fujitsu K computer	(SPARC64 VIIIfx 2.0GHz)		8 162,0	548 352
2012	IBM BlueGene/Q Sequoia	(Power BQC 16C 1,6GHz)		16 324,8	1 572 864
2013	NUDT Tianhe-2	(Xeon E5-2692 12C 2,2 GHz + Phi 31S1P)		33 862,7	3 120 000
2016	Sunway TaihuLight	(Sunway SW26010 260C 1,45 GHz)		93 014,6	10 649 600
2018	Summit	(IBM Power9 22C 3,07 GHz + NVIDIA GV100)		143 500,0	2 397 824
2021	Fugaku	(Fujitsu A64FX 48C 2,2 GHz)		442 010,0	7 630 848
2023	JUWELS [D #13]	(Epyc 7402 24C 2,8 GHz + NVIDIA A100)		44 120,0	449 280
2023	Frontier	(AMD Epyc 64C 2,0 GHz + Instinct)		1 194 000,0	8 699 904

Leistungssteigerung der Spitzenrechner seit 1993

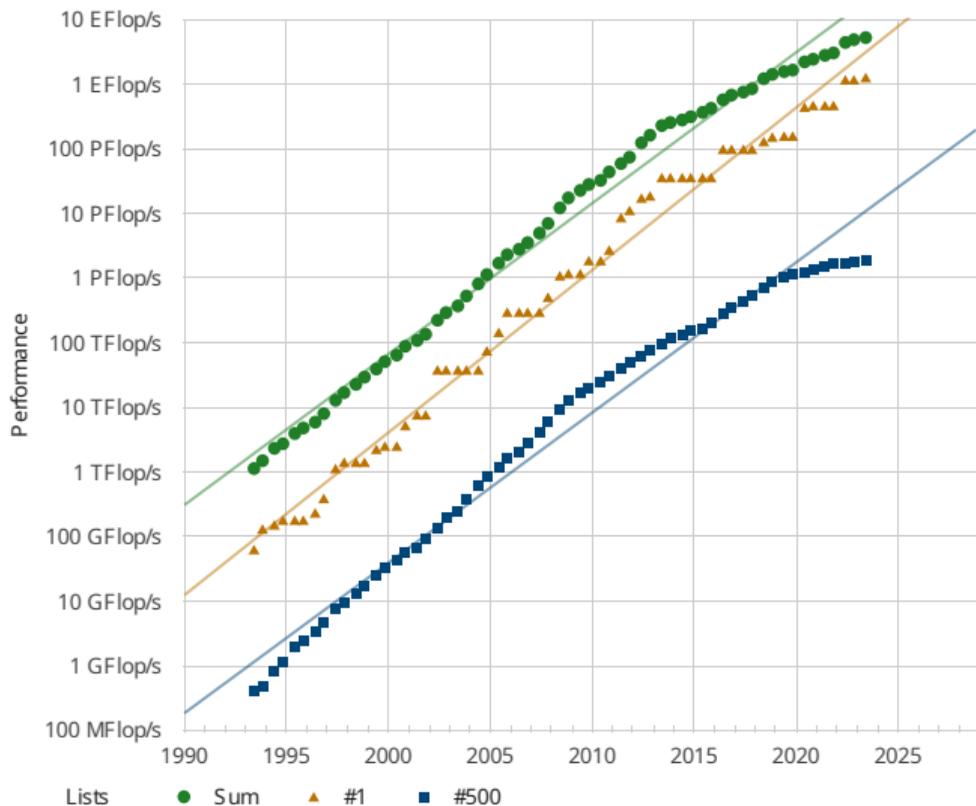
www.top500.org de.wikipedia.org/wiki/Supercomputer

Jahr	Rechner	CPU	Linpack	[TFlop/s]	Prozessoren	Power [KW]
1993	TMC CM-5/1024	(SuperSparc 32MHz)		0,0597	1 024	
1994	Intel XP/S140	(80860 50MHz)		0,1434	3 680	
1995	Fujitsu NWT	(105 MHz)		0,17	140	
1996	Hitachi SR2201/1024	(HARP-1E 120MHz)		0,2204	1 024	
1997	Intel ASCI Red	(Pentium Pro 200MHz)		1,068	7 264	
1999	Intel ASCI Red	(Pentium Pro 333MHz)		2,121	9 472	
2001	IBM ASCI White	(Power3 375MHz)		7,226	8 192	
2002	NEC Earth Simulator	(NEC 1GHz)		35,86	5 120	3 200
2005	IBM BlueGene/L	(PowerPC 440 2C 700MHz)		136,8	65 536	716
2006	IBM BlueGene/L	(PowerPC 440 2C 700MHz)		280,6	131 072	1 433
2008	IBM Roadrunner	(Opteron 2C 1,8GHz + IBM Cell 9C 3,2 GHz)		1 026,0	122 400	2 345
2010	Cray XT5-HE Jaguar	(Opteron 6C 2,6GHz)		1 759,0	224 162	6 950
2011	Fujitsu K computer	(SPARC64 VIIIfx 2.0GHz)		8 162,0	548 352	9 899
2012	IBM BlueGene/Q Sequoia	(Power BQC 16C 1,6GHz)		16 324,8	1 572 864	7 890
2013	NUDT Tianhe-2	(Xeon E5-2692 12C 2,2 GHz + Phi 31S1P)		33 862,7	3 120 000	17 808
2016	Sunway TaihuLight	(Sunway SW26010 260C 1,45 GHz)		93 014,6	10 649 600	15 371
2018	Summit	(IBM Power9 22C 3,07 GHz + NVIDIA GV100)		143 500,0	2 397 824	9 783
2021	Fugaku	(Fujitsu A64FX 48C 2,2 GHz)		442 010,0	7 630 848	29 899
2023	JUWELS [D #13]	(Epyc 7402 24C 2,8 GHz + NVIDIA A100)		44 120,0	449 280	1 764
2023	Frontier	(AMD Epyc 64C 2,0 GHz + Instinct)		1 194 000,0	8 699 904	22 703



Leistungssteigerung der Spitzenrechner seit 1993 (cont.)

www.top500.org de.wikipedia.org/wiki/Supercomputer





- ▶ Miniaturisierung schreitet weiter fort
- ▶ Taktraten physikalisch limitiert
 - ▶ Technologie / Strukturgrößen
 - ▶ Leistungsaufnahme \Rightarrow Spannungsversorgung + Kühlung

Entwicklungen

- ▶ seit 2011: CPU plus Grafikeinheit
- ▶ Integration mehrerer CPUs auf einem Chip (2-...64-Cores)
- ▶ Cache Speicher (SRAM als schneller Zwischenspeicher) auf dem Die
- ▶ Integration von Peripheriegeräten (Speicherinterface, PCIe, ...)
- ▶ heterogene Architekturen: „performance“ / „efficiency“
- ▶ multi-Chip Module (CPU, IO, Cache) + 3D Stapel
- ▶ **SoC**: „System on a chip“

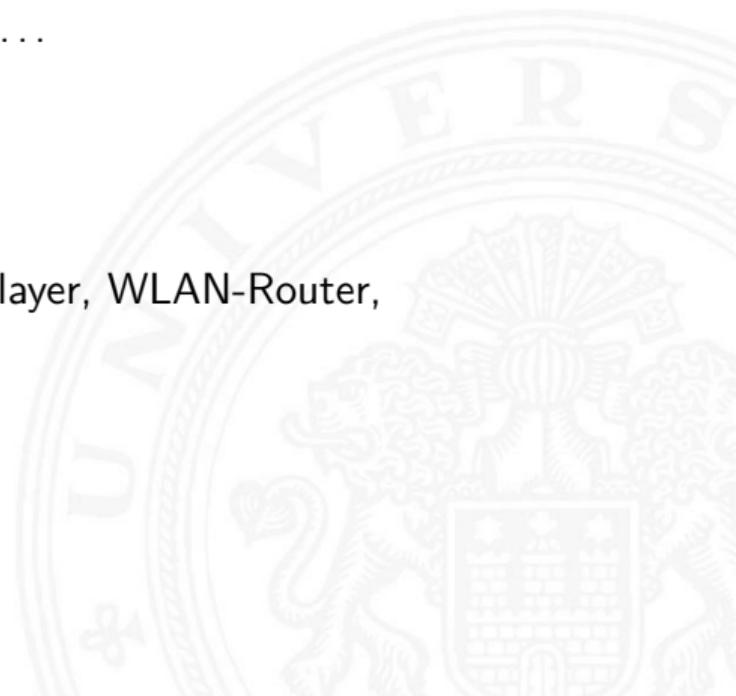


Gesamtes System auf einem Chip integriert:

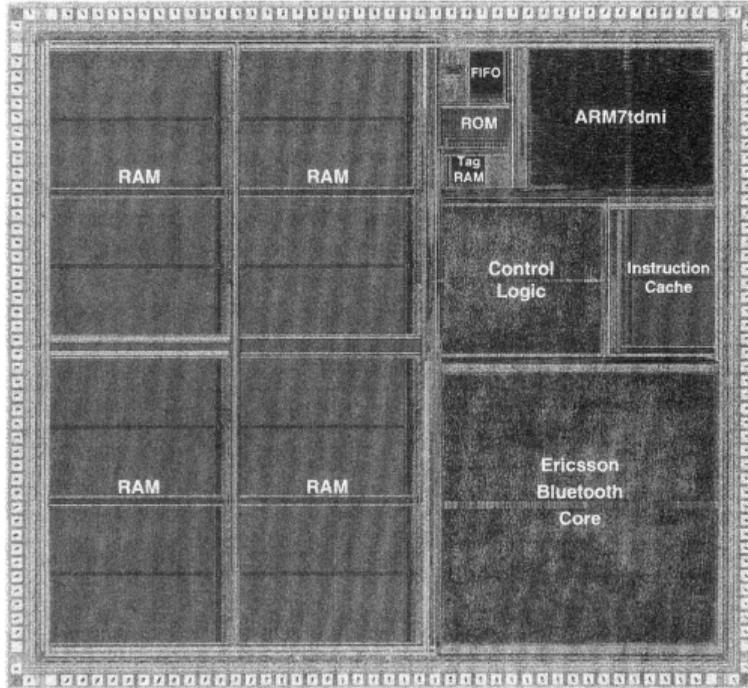
- ▶ ein oder mehrere Prozessoren, z.T. verschiedene Typen
 - ▶ hohe Rechenleistung
 - ▶ energieeffizient
- ⇒ z.B. ARM mit *big.LITTLE* Konzept; aktuell Laptop, PC: Intel, AMD
- ▶ Cache Hierarchie: 1st Level (D+I pro CPU) / 2nd (pro CPU) / 3rd (alle Kerne)
- ▶ dedizierte Prozessoren: Grafik, Video(de)codierung, DSP, AI ...
- ▶ Peripherieschnittstellen: Bussysteme, Speichercontroller
- ▶ Hauptspeicher (speziell "Embedded"), Grafikspeicher
- ▶ weitere Speicher für Medien/Netzwerkoperationen



- ▶ Peripherieblöcke nach Kundenwunsch konfiguriert:
 - ▶ Displaysteuerung: DP, HDMI ...
 - ▶ A/V-Schnittstellen: Kamera, Mikrofone, Audio ...
 - ▶ serielle und parallele Schnittstellen, SPI, I/O-Pins ...
 - ▶ Feldbusse: I²C, CAN ...
 - ▶ PC-like: USB, Firewire, SATA ...
 - ▶ Netzwerk kabelgebunden (Ethernet)
 - ▶ Funkschnittstellen: WLAN, Bluetooth, 5G ...
- ▶ Smartphones, Tablet-Computer, Medien-/DVD-Player, WLAN-Router, NAS-/Home-Server ...



► Bluetooth-Controller (2000)



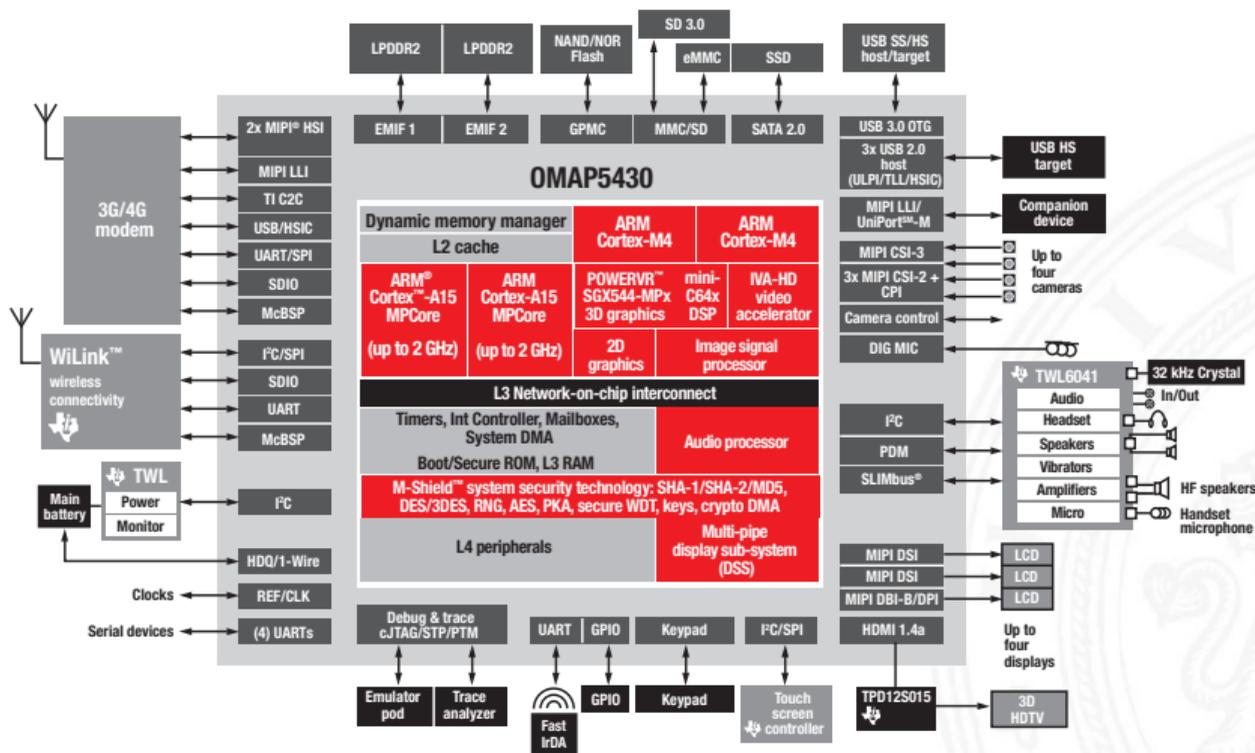
© VLSI Technology, Inc.

[Fur00]

Prozess	0,25 μm
Metall	3-Layer
V_{DD}	2,5 V
Transistoren	4,3 Mill.
Chipfläche	20 mm ²
Taktrate	0 ... 13 MHz
MIPS	12
Power	75 mW
MIPS/W	160

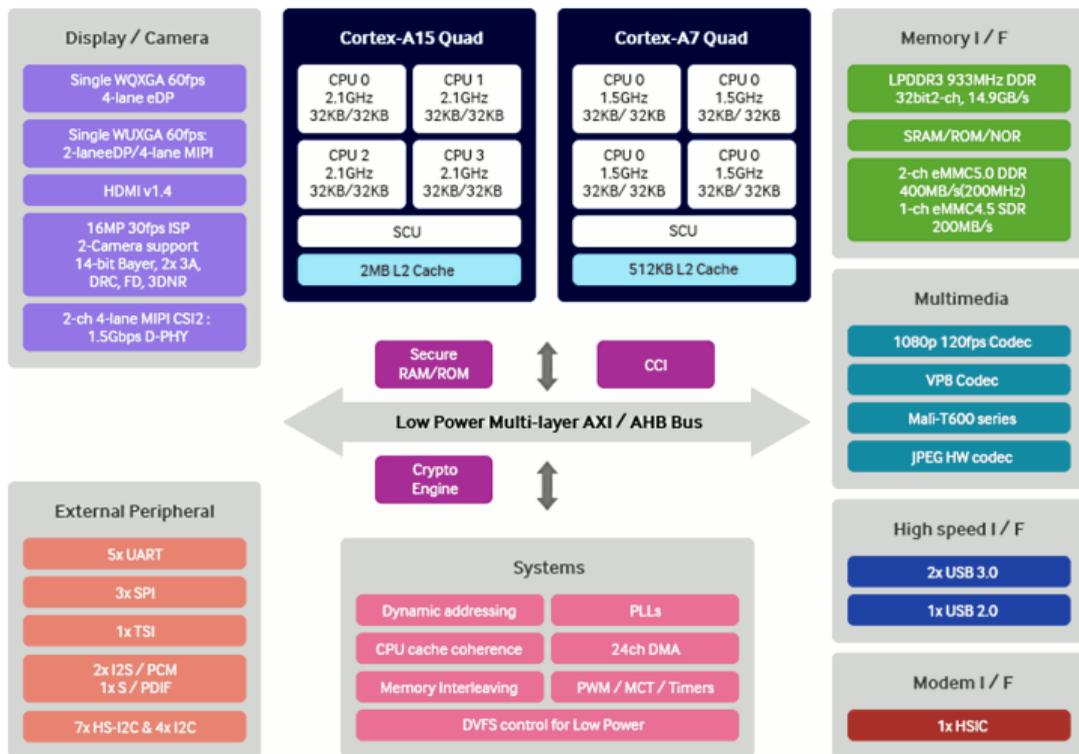
► Texas Instruments OMAP 5430 (2011)

[T1]



► Samsung Exynos-5422 (2014)

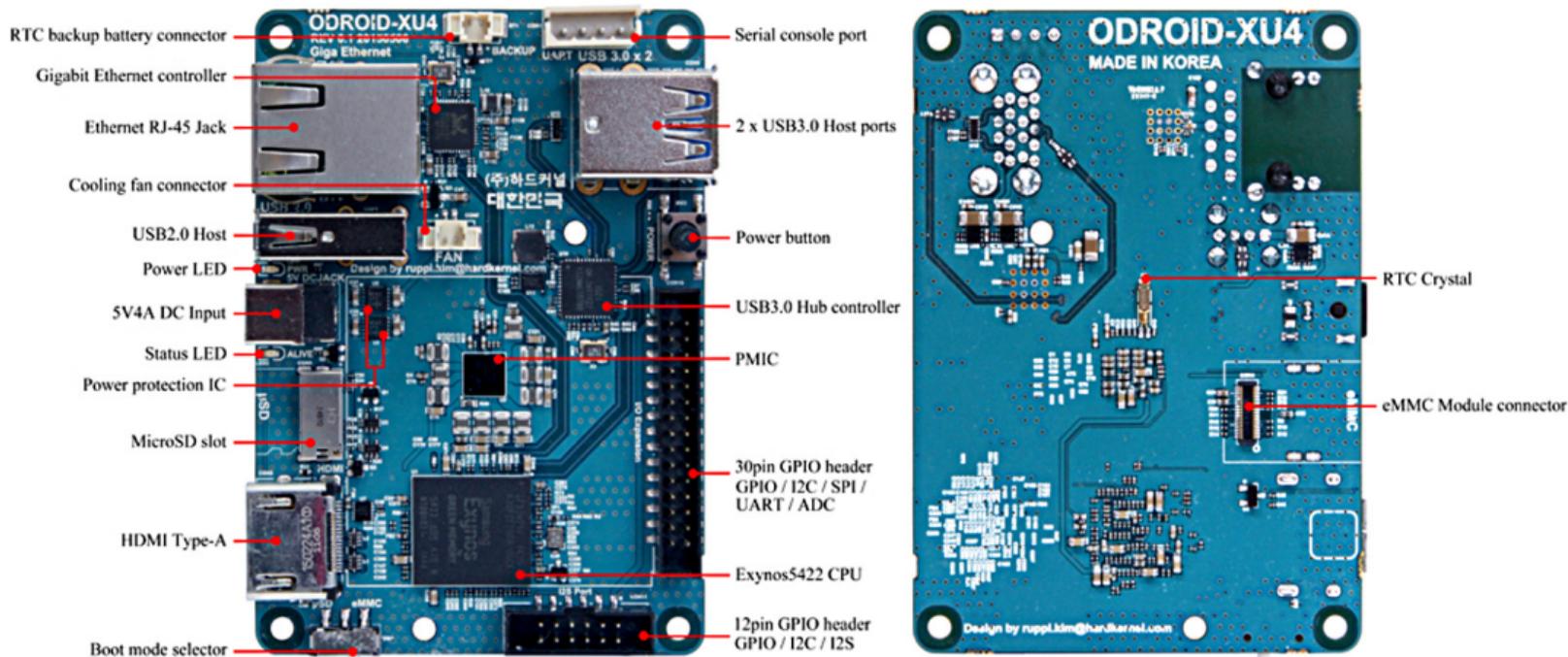
[Samsung]



▶ Beispiel: Odroid XU4

[Hardkernel]

- ▶ vollständiger 8-Kern Mikrocomputer
- ▶ Betriebssystem: Android oder Linux





- ▶ Jeder exponentielle Verlauf stößt irgendwann an natürliche oder wirtschaftliche Grenzen
 - ▶ Beispiel: physikalische Limits
 - ▶ Eine DRAM-Speicherzelle speichert etwa 200 Elektronen (2012)
Skalierung: es werden mit jeder neuen Technologiestufe weniger
 - ▶ Offensichtlich ist die Grenze spätestens dann erreicht, wenn nur noch ein einziges Elektron für 1-bit gespeichert wird
 - ▶ Ab diesem Zeitpunkt gibt es bessere Performanz nur noch durch bessere Algorithmen / Architekturen!
- ⇒ Annahme: 50 % Skalierung pro Jahr, 200 Elektronen/Speicherzelle
gesucht: $x \hat{=}$ Jahre Fortschritt
- ⇒ $200 / (1,5^x) \geq 1$
 $x = \ln(200) / \ln(1,5) \approx 13$ Jahre

$$a^b = \exp(b \cdot \ln a)$$



IEEE International Roadmap for Devices and Systems

irds.ieee.org/editions/2022

- ▶ IEEE: Institute of Electrical and Electronics Engineers
- ▶ Beteiligung von
 - ▶ Halbleiterherstellern
 - ▶ Geräte-Herstellern
 - ▶ Universitäten und Forschungsinstituten
 - ▶ Fachverbänden aus USA, Europa, Asien
- ▶ Publikation von langjährigen Vorhersagen
- ▶ Zukünftige Entwicklung der Halbleitertechnologie
- ▶ Prognosen zu Fertigungsprozessen, Modellierung, Simulation, Entwurf etc.
- ▶ für Chips (Speicher, Prozessoren, SoC ...) und Systeme
- ▶ neue Technologien: Quantencomputing etc.

Roadmap: IRDS (cont.)

Table MM01 – More Moore – Logic Core Device Technology Roadmap (Ausschnitt, 2017)

YEAR OF PRODUCTION	2017	2019	2021	2024	2027	2030	2033
Logic industry "Node Range" Labeling (nm)	P54M36	P48M28	P42M24	P36M21	P32M14	P32M14T2	P32M14T4
IDM-Foundry node labeling	"10"	"7"	"5"	"3"	"2.1"	"1.5"	"1.0"
Logic device structure options	i10-f7	i7-f5	i5-f3	i3-f2.1	i2.1-f1.5	i1.5-f1.0	i1.0-f0.7
Logic device main stream device	finFET	finFET	LGAA	LGAA	LGAA	VGAA, LGAA	VGAA, LGAA
DEVICE STRUCTURES	FDSOI	LGAA	finFET	VGAA	VGAA	3DVL SI	3DVL SI
DEVICE STRUCTURES	finFET	finFET	LGAA	LGAA	LGAA	VGAA	VGAA
DEVICE STRUCTURES							
LOGIC TECHNOLOGY ANCHORS							
Patterning technology inflection for Mx interconnect	193i, EUV	193i, EUV DP	193i, EUV DP	193i, High-NA EUV	193i, High-NA EUV+(DSA)	193i, High-NA EUV+(DSA)	193i, High-NA EUV+(DSA)
Channel material technology inflection	Si	SiGe25%	SiGe50%	Ge, IIIV (TFET?), 2D Mat	Ge, IIIV (TFET?), 2D Mat	Ge, IIIV (TFET?), 2D Mat	Ge, IIIV (TFET?), 2D Mat
Process technology inflection	Conformal deposition	Conformal Doping, Contact	Channel, RMG	Stacked-device Non-Cu Mx	Stacked-device Non-Cu Mx	Steep-SS, 3D	Steep-SS, 3D
Stacking generation inflection	2D	2D	3D-stacking: W2W D2W	3D-device: P-over-N Hetero	3D-device: Mem-on-Logic Hetero	3D-device: Mem-on-Logic Hetero	3D-device: Logic-on-Logic Hetero
LOGIC TECHNOLOGY INTEGRATION CAPACITY							
Design scaling factor for standard cell	-	0,98	1,09	0,96	1,03	2,00	1,00
Design scaling factor for SRAM (111) bitcell	-	1,00	1,00	1,00	1,00	1,25	1,00
POWER AND PERFORMANCE SCALING FACTORS							
Vdd (V)	0,75	0,70	0,65	0,65	0,65	0,60	0,55
Physical gate length for HP Logic (nm)	20,0	18,0	16,0	14,0	12,0	12,0	12,0
Datapath speed improvement at Vdd - relative	1,00	1,19	1,21	1,34	1,56	1,60	1,70
Power density of logic path cube at fmax - relative	1,00	1,20	1,21	1,82	2,69	4,49	8,00
fmax of a single CPU core at Vdd (GHz)	2,5	3,0	3,0	3,3	3,9	4,0	4,2
favg at constant power density and Vdd (GHz)	2,50	2,48	2,51	1,84	1,45	0,89	0,53
CPU SiP throughput at fmax (TFLOPS/sec)	0,16	0,27	0,46	0,79	1,34	2,27	3,86
INTERCONNECT TECHNOLOGY							
Conductor	Cu, non-Cu	Cu, non-Cu	Cu, non-Cu	Cu, non-Cu	Cu, non-Cu	Cu, non-Cu	Cu, non-Cu
Number of wiring layers	14	16	18	20	20	20	20



Moore's Law

Beispiel für die Auswirkung von Moore's Law

Angenommen die Lösung einer Rechenaufgabe dauert derzeit vier Jahre und die Rechenleistung wächst jedes Jahr um 60%.

Wie lösen wir das Problem ?





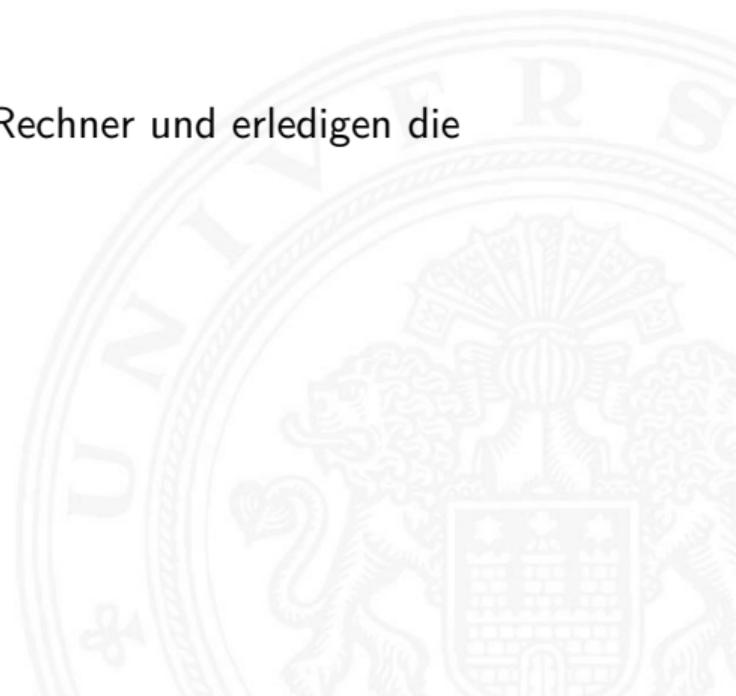
Moore's Law: Schöpferische Pause

Beispiel für die Auswirkung von Moore's Law

Angenommen die Lösung einer Rechenaufgabe dauert derzeit vier Jahre und die Rechenleistung wächst jedes Jahr um 60%.

Ein mögliches Vorgehen ist dann das folgende:

- ▶ Wir warten drei Jahre, kaufen dann einen neuen Rechner und erledigen die Rechenaufgabe in einem Jahr.
- ▶ *Wie das ?*





Moore's Law: Schöpferische Pause

Beispiel für die Auswirkung von Moore's Law

Angenommen die Lösung einer Rechenaufgabe dauert derzeit vier Jahre und die Rechenleistung wächst jedes Jahr um 60%.

Ein mögliches Vorgehen ist dann das folgende:

- ▶ Wir warten drei Jahre, kaufen dann einen neuen Rechner und erledigen die Rechenaufgabe in einem Jahr.
- ⇒ Nach einem Jahr können wir einen Rechner kaufen, der um den Faktor 1,6 Mal schneller ist, nach zwei Jahren bereits $1,6 \cdot 1,6$ Mal schneller, und nach drei Jahren (also am Beginn des vierten Jahres) gilt $(1 + 60\%)^3 = 4,096$.
- ▶ Wir sind also sogar ein bisschen schneller fertig, als wenn wir den jetzigen Rechner die ganze Zeit durchlaufen lassen.



Ab jetzt erst mal ein *bottom-up* Vorgehen:

Start mit grundlegenden Aspekten

- ▶ Informationsverarbeitung und -repräsentation
- ▶ Darstellung von Zahlen und Zeichen
- ▶ arithmetische und logische Operationen
- ▶ Schaltnetze, Schaltwerke, endliche Automaten

dann Kennenlernen aller Basiskomponenten des Digitalrechners

- ▶ Gatter, Flipflops ...
- ▶ Register, ALU, Speicher ...

und Konstruktion eines Rechners (HW) mit seinen Betriebsmitteln (SW)

- ▶ Befehlssatz, -abarbeitung, Assembler
- ▶ Pipelining, Speicherhierarchie
- ▶ Prozesskontrolle, Locking, Interrupts, Scheduling
- ▶ virtueller Speicher, Dateisystem, Ein- / Ausgabe
- ▶ ...



- [TA14] A.S. Tanenbaum, T. Austin: *Rechnerarchitektur – Von der digitalen Logik zum Parallelrechner*.
6. Auflage, Pearson Deutschland GmbH, 2014. ISBN 978-3-8689-4238-5
- [HenHA] N. Hendrich: *HADES — HAMBURG DEsign System*.
Universität Hamburg, FB Informatik, Lehrmaterial.
tams.informatik.uni-hamburg.de/applets/hades/webdemos
- [Fur00] S. Furber: *ARM System-on-Chip Architecture*.
2nd edition, Pearson Education Limited, 2000. ISBN 978-0-201-67519-1
- [Moo65] G.E. Moore: *Cramming More Components Onto Integrated Circuits*.
in: *Electronics* 38 (1965), April 19, Nr. 8

[IRDS22] *International Roadmap for Devices and Systems (IRDS) 2022 Edition.*
IEEE International Roadmap for Devices and Systems, 2022.

irds.ieee.org/editions/2022

[Intel] Intel Corp.; Santa Clara, CA.

www.intel.com ark.intel.com

www.intel.com/content/www/us/en/history/museum-gordon-moore-law.html

www.intel.de/content/www/de/de/silicon-innovations

[TI] Texas Instruments Inc.; Dallas, TX.

www.ti.com

[Samsung] Samsung Electronics Co., Ltd.; Suwon, Südkorea.

www.samsung.com

[Hardkernel] Hardkernel co., Ltd.; AnYang, Südkorea.

www.hardkernel.com