

# GTI - Grundlagen der Technischen Informatik Einführung

Thomas Studer  
Institut für Informatik  
Universität Bern

# Vorbemerkungen

---

- > Programmieren 1  
Algorithmen (Software)
- > Grundlagen der Technischen Informatik  
Hardware (Wie baut man Computer?)
- > Algorithmus sagt: Addiere zwei Zahlen!

Wie realisiert ein Computer diese Addition technisch?

- Dabei interessieren wir uns für die Logik, nicht für die Physik (Elektrotechnik).

# GTI Team

---

## Vorlesung:

- Thomas Studer  
[thomas.studer@unibe.ch](mailto:thomas.studer@unibe.ch)

## Übungsbetrieb:

- Tobias Kohler
- Maya Nedir
- Nicolas Wyss

# Termine

---

Vorlesung:

- Mittwoch, 14-16

Übungen:

- Mittwoch, 16-17

Repetitionsserie:

- 1. November, anstelle Vorlesung  
Repetitionsaufgaben, Prüfungsvorbereitung

# Übungsbetrieb

---

**Jede Woche** eine Übungsserie mit Aufgaben

Lösungen müssen wöchentlich abgegeben werden,  
jeweils am **Mittwoch vor der Übungsstunde**

Abgabe schriftlich **auf Papier** (keine elektronische Abgabe)

**Alle Aufgaben** müssen gelöst werden.

Die Übungen müssen in **2er Gruppen** gelöst werden.

Details: **Infoblatt zu den Übungen auf Ilias**

# Prüfung

---

## Prüfung:

- Montag, 8. Januar 2024 um 10:15 Uhr

## Bedingung zur Prüfungszulassung (Testat):

- 70% der Punkte aus den Übungen erreicht.

## Anmeldung:

- in KSL, Details werden noch bekanntgegeben

## Repetition und Fragestunde:

- in der letzten Semesterwoche

# Ilias

---

Ilias:

- <https://www.ilias.unibe.ch>
- Materialien zur Vorlesung

Forum auf Ilias

- für alle Fragen zur Vorlesung und zu den Übungen

# Die Veranstaltung wird aufgezeichnet

## Hinweise zur Podcastnutzung



### Verwendung der Aufzeichnungen

Die Aufnahmen dürfen nur für den Privatgebrauch verwendet werden. Eine Weiterverbreitung in welcher Form auch immer, ganz oder in Auszügen, ist ohne Einverständnis der Dozentin/des Dozenten nicht erlaubt und kann disziplinarisch und anderweitig geahndet werden.

### Widersprüche

Bei inhaltlichen Widersprüchen haben Skripte oder anderes als prüfungsrelevant deklariertes Material Vorrang vor den Podcasts. Bei Unklarheiten kontaktieren Sie bitte umgehend die Dozentin/den Dozenten.

### Lückenlose Aufzeichnung nicht garantiert

Aufgrund technischer Störungen kann es vorkommen, dass einzelne Vorlesungen nicht oder nicht störungsfrei aufgezeichnet werden. Die Studierenden verzichten deshalb auf eigenes Risiko auf den Besuch einer Veranstaltung oder auf das Erstellen eigener Notizen

### Verfügbarkeit nicht garantiert

Die ununterbrochene Verfügbarkeit der Podcast kann aus technischen Gründen nicht garantiert werden.

# Die Vorlesung wird live gestreamt

## Einen Live-Stream in ILIAS aufrufen



### Live-Stream aufrufen

- Studierende sehen nur geplante Live-Stream-Termine, nicht aber geplante Podcast-Termine.
- Bei laufende Live-Streams erscheint ein Button «Beitreten».
- Ein Klick darauf öffnet ein neues Tab. Der Live-Stream muss dann über einen Klick auf das Play-Symbol gestartet werden.

### Häufiger Fehler

Der Button «Beitreten» erscheint nur im Veranstaltungszeitraum und erst, wenn die Startzeit verstrichen ist (im Beispiel «06:13 Uhr»). Wenn Sie die Opencast Serie vorher aufrufen, wird der Button nach Verstreichen der Startzeit NICHT automatisch angezeigt, sondern erst nach einem Reload des Browser (z.B. Klick auf «Aufzeichnungen neu laden»).

# Die Veranstaltung wird live gestreamt

## Hinweise zur Live-Stream Nutzung



### Verwendung von Live-Stream und Aufzeichnung

Der Live-Stream und dessen Aufzeichnung dürfen nur für den Privatgebrauch verwendet werden. Eine Weiterverbreitung in welcher Form auch immer, ganz oder in Auszügen, ist ohne Einverständnis der Dozentin/des Dozenten nicht erlaubt und kann disziplinarisch und anderweitig geahndet werden.

### Widersprüche

Bei inhaltlichen Widersprüchen haben Skripte oder anderes als prüfungsrelevant deklariertes Material Vorrang vor den Aufzeichnungen der Live-Streams. Bei Unklarheiten kontaktieren Sie bitte umgehend die Dozentin/den Dozenten.

### Lückenlose Übertragung & Aufzeichnung nicht garantiert

Aufgrund technischer Störungen kann es vorkommen, dass einzelne Vorlesungen nicht oder nicht störungsfrei per Live-Stream übertragen und aufgezeichnet werden. Die Studierenden verzichten deshalb auf eigenes Risiko auf den Besuch einer Veranstaltung oder auf das Erstellen eigener Notizen.

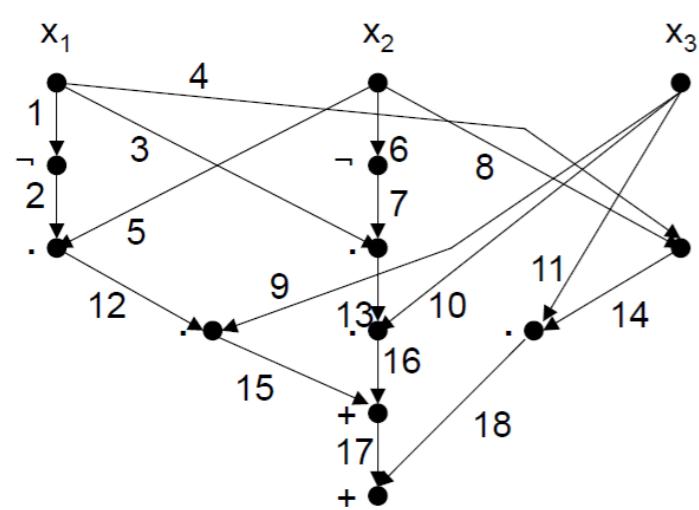
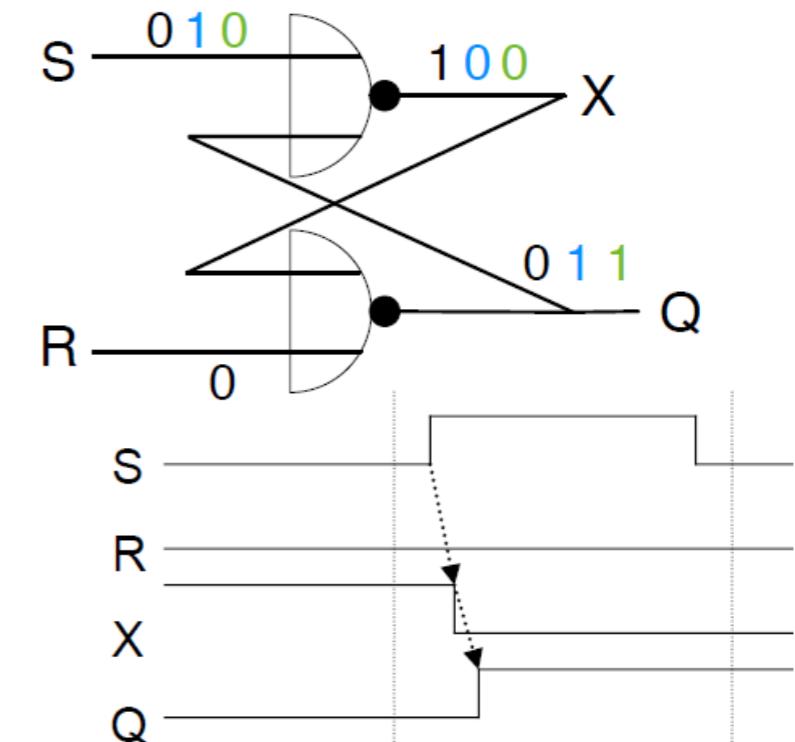
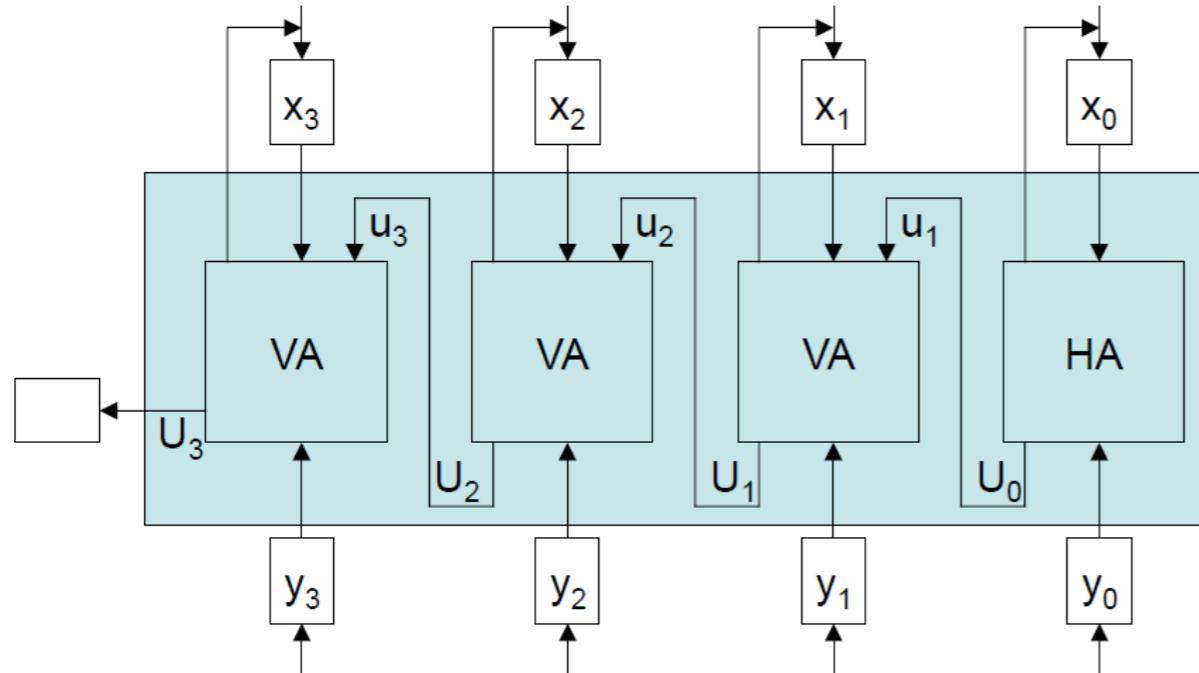
### Verfügbarkeit nicht garantiert

Die ununterbrochene Verfügbarkeit der Live-Streams und anschliessend der Aufzeichnungen kann aus technischen Gründen nicht garantiert werden.

# GTI - Grundlagen der Technischen Informatik

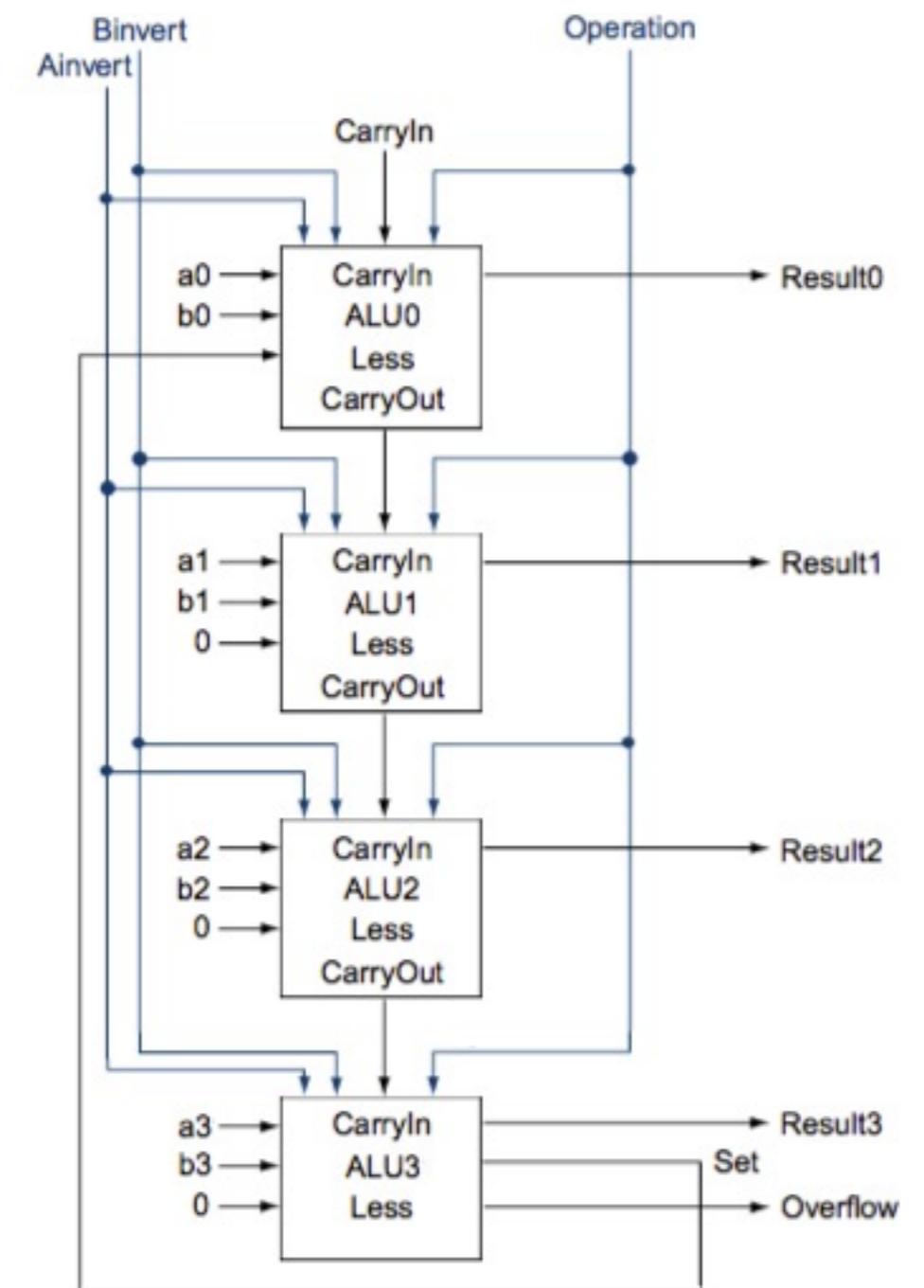
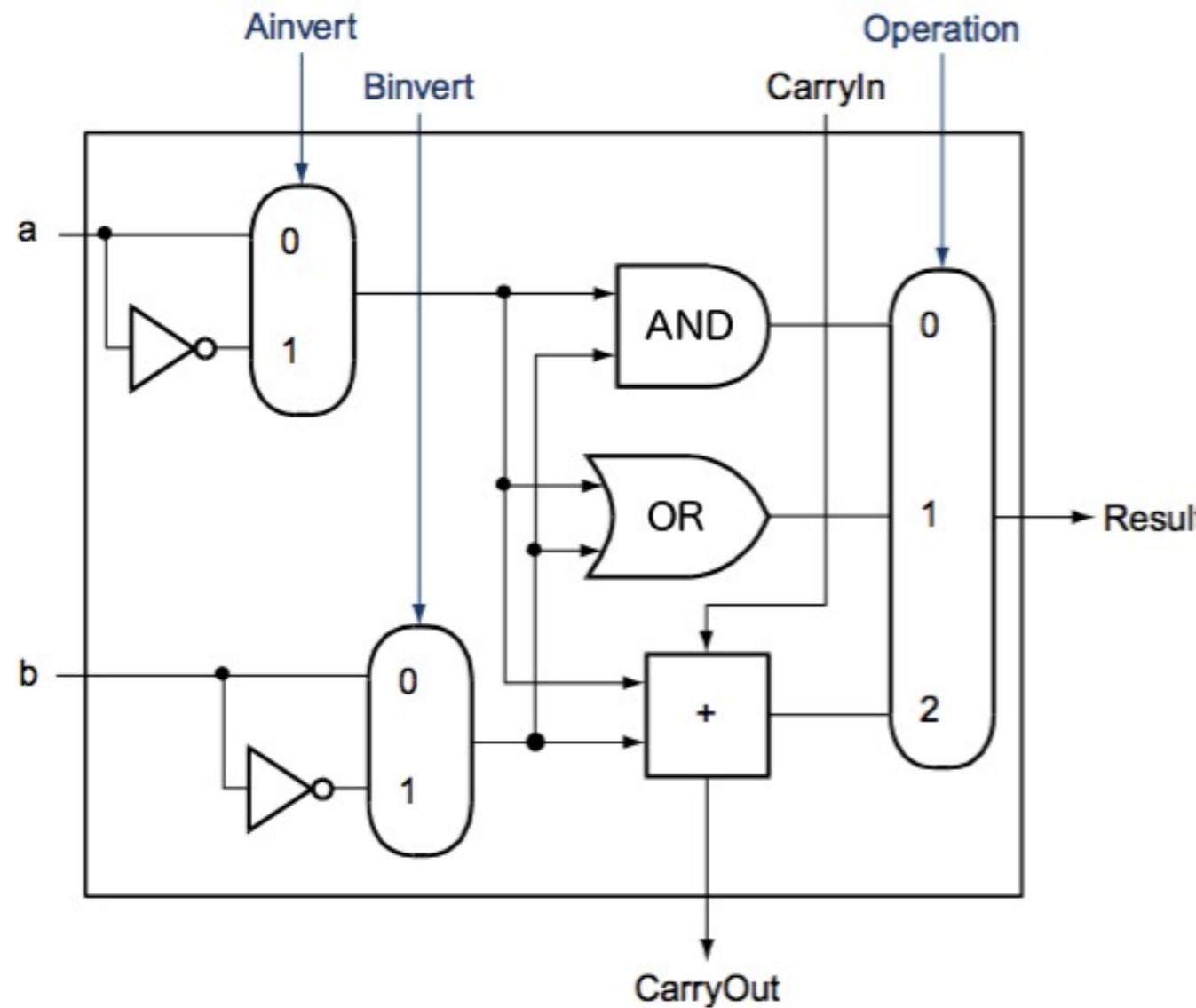
## Ziele der Vorlesung

# Addieren, Speichern, Berechnen



$$f(X_1, X_2, X_3) = m_3 + m_5 + m_7 = \neg X_1 \neg X_2 X_3 + X_1 \neg X_2 X_3 + X_1 X_2 X_3$$

# Arithmetic Logic Unit



# Am Anfang war...

---

0

1

---

# Literatur



- > W. Oberschelp, G. Vossen  
Rechneraufbau und Rechnerstrukturen  
ISBN 3-486-57849-9  
R. Oldenbourg Verlag, München, Wien  
***einige Kapitel sind in Ilias verfügbar!***



- > B. Becker, P. Molitor  
Technische Informatik - Eine einführende  
Darstellung  
ISBN 3-486-58650-5  
Oldenbourg

# Diskrete Mathematik

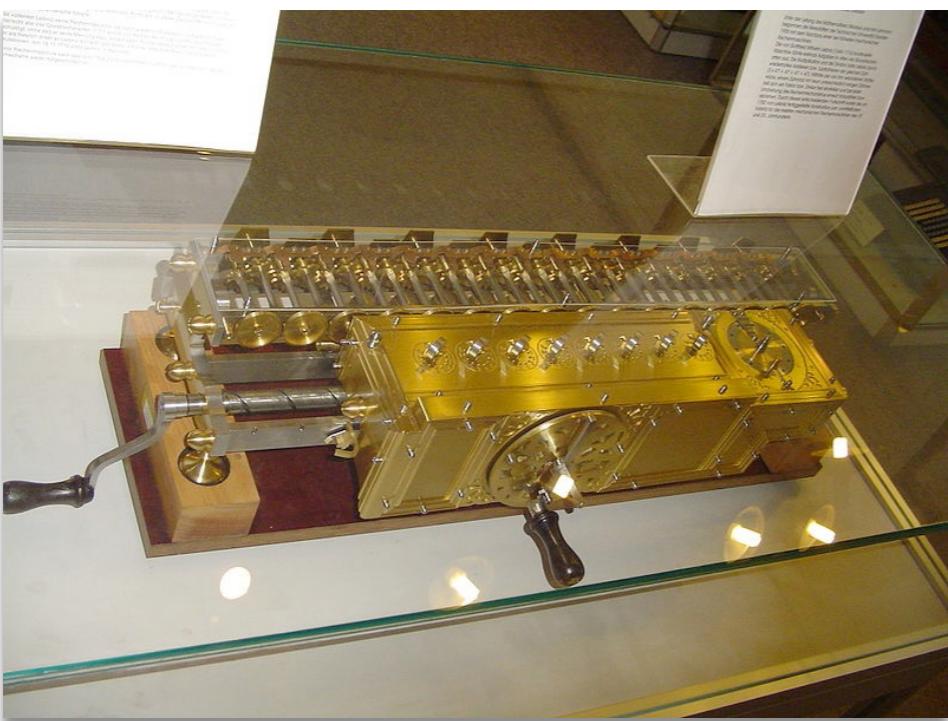
---

- > Es wird ein paar inhaltliche Überschneidungen zwischen GTI und der Vorlesung *Diskrete Mathematik* geben
- > Dies betrifft vor allem die nächsten beiden Wochen
- > Verwenden Sie in GTI die Definitionen aus der GTI Vorlesung und in DM diejenigen aus der DM Vorlesung

# GTI - Grundlagen der Technischen Informatik Geschichte

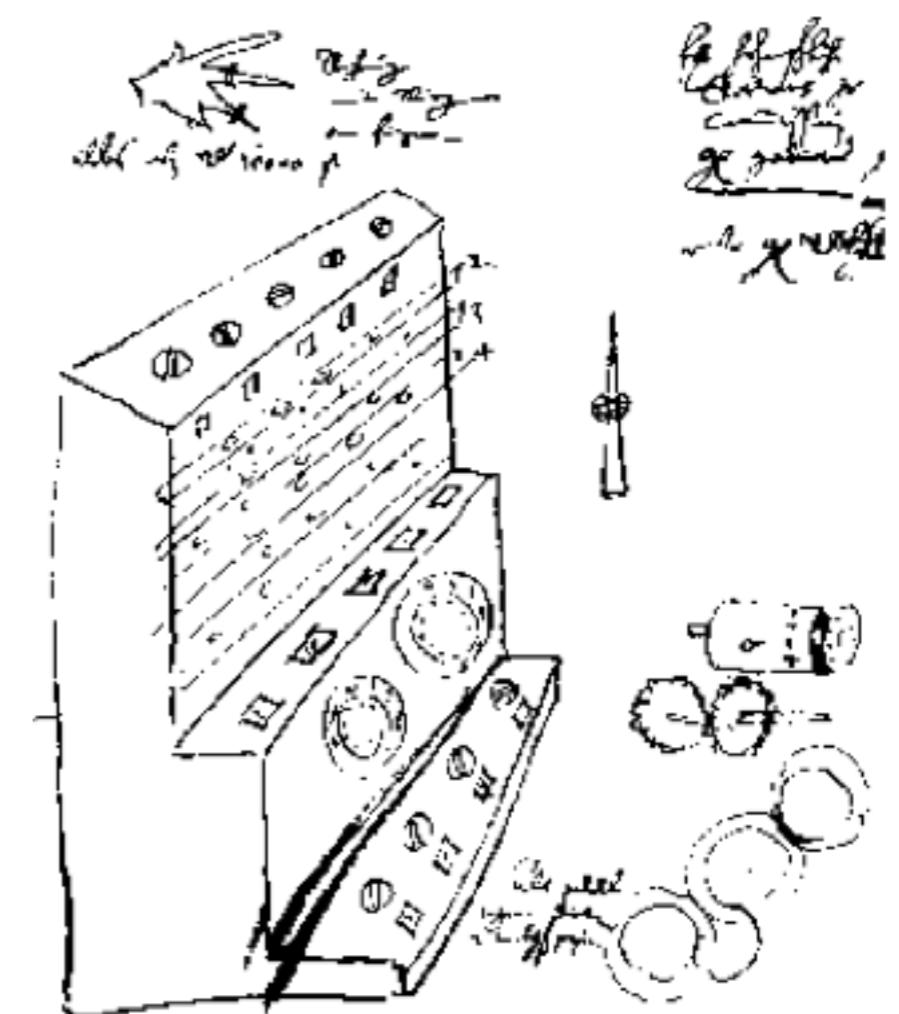
# Gottfried Wilhelm Leibniz

- > “Es ist ausgezeichneter Männer unwürdig, wie Sklaven Stunden bei Berechnungen zu verlieren, die man problemlos jemand anderem überlassen könnte, wenn es nur Maschinen dafür gäbe.”



# Die Rechenuhr von Schickard

- > Diese Maschine aus dem Jahre 1623 gestattet die Durchführung aller vier Grundrechenarten.



# Die Rechenuhr von Schickard

---

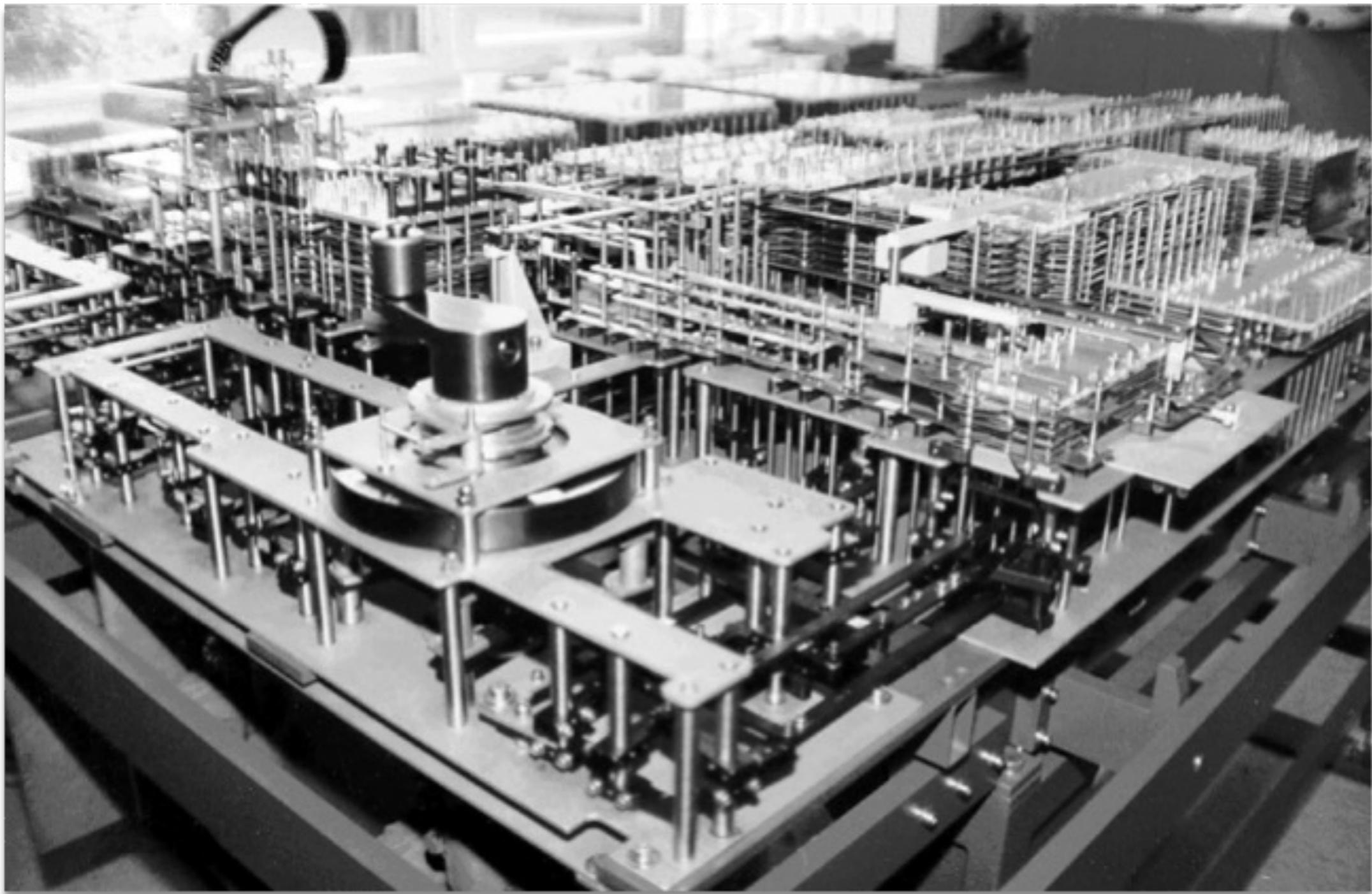
- > Schickard war stolz auf seine großartige Erfindung. So schrieb er in einem Brief vom 20. September 1623 an Kepler:

*"Dasselbe, was Du auf rechnerischem Weg gemacht hast, habe ich kürzlich mechanisch versucht und eine aus 11 vollständigen und 6 verstümmelten Rädchen bestehende Maschine gebaut, welche gegebene Zahlen im Augenblick automatisch zusammenrechnet: addiert, subtrahiert, multipliziert und dividiert. Du würdest hell auflachen, wenn Du da wärest und sehen könntest, wie sie, so oft es über einen Zehner oder Hunderter weggeht, die Stellen zur Linken ganz von selbst erhöht oder ihnen beim Subtrahieren etwas wegnimmt."*

- > Leider stand Schickards Leben und Werk im Schatten des Dreißigjährigen Krieges. Er starb 1653 an Pest, und die Erinnerung an seine Großtat erlosch.

# Z1, Z2, Z3: Konrad Zuse (1938-1941)

---



## Z1

---

- › Mechanische Konstruktion aus Blech
- › Taktfrequenz: 1 Hz
- › Stromverbrauch des Elektromotors zur Taktgebung: 1'000 Watt
- › Gewicht: 500kg
- › Erster frei programmierbarer Rechner
- › Verwendung binäre Zahlen
- › Klare Trennung zwischen Programm und den ausführenden Komponenten

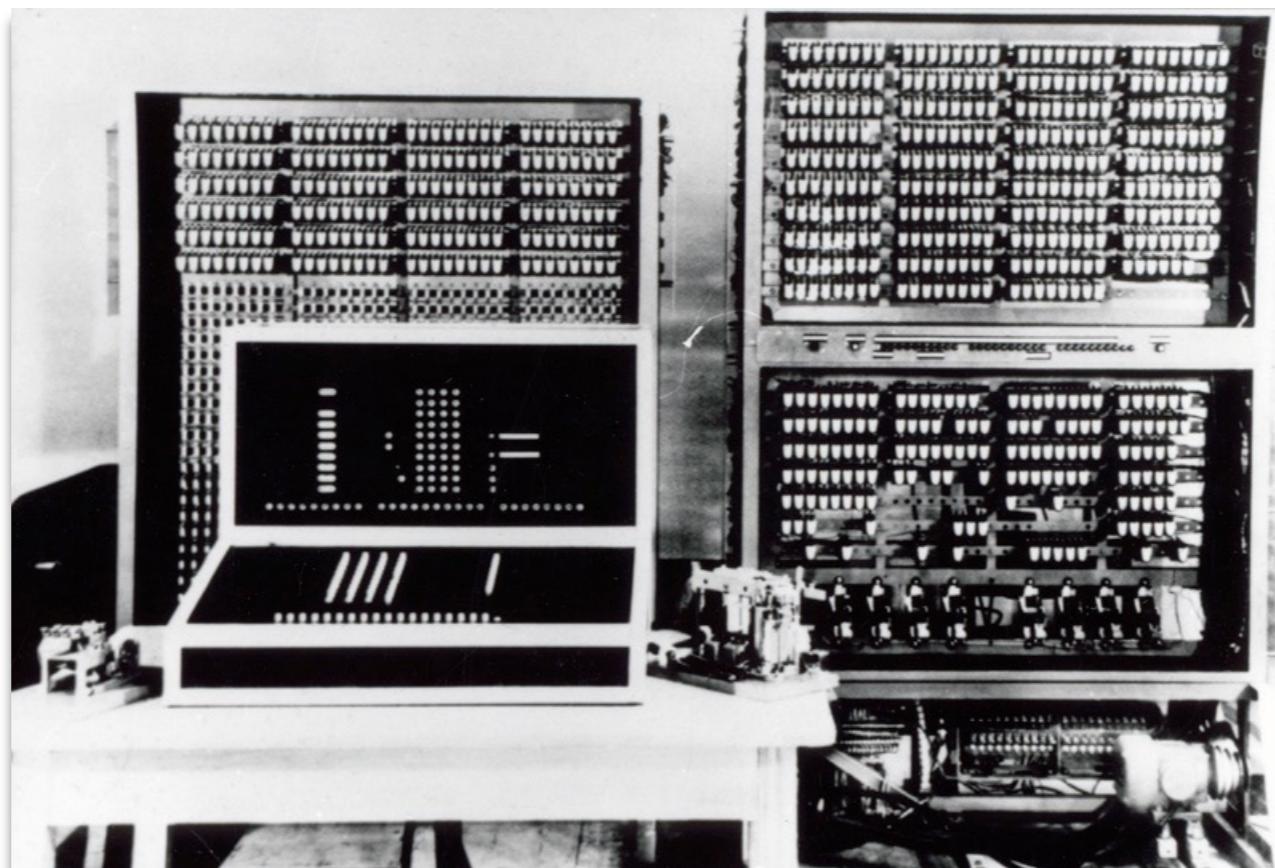
## Z2

- > Arithmetische Einheit und Steuerwerk mit Relais
- > Mechanischer Speicher
- > Taktfrequenz: 3Hz
- > Relais: Schalter, der mit Elektromagneten betätigt wird

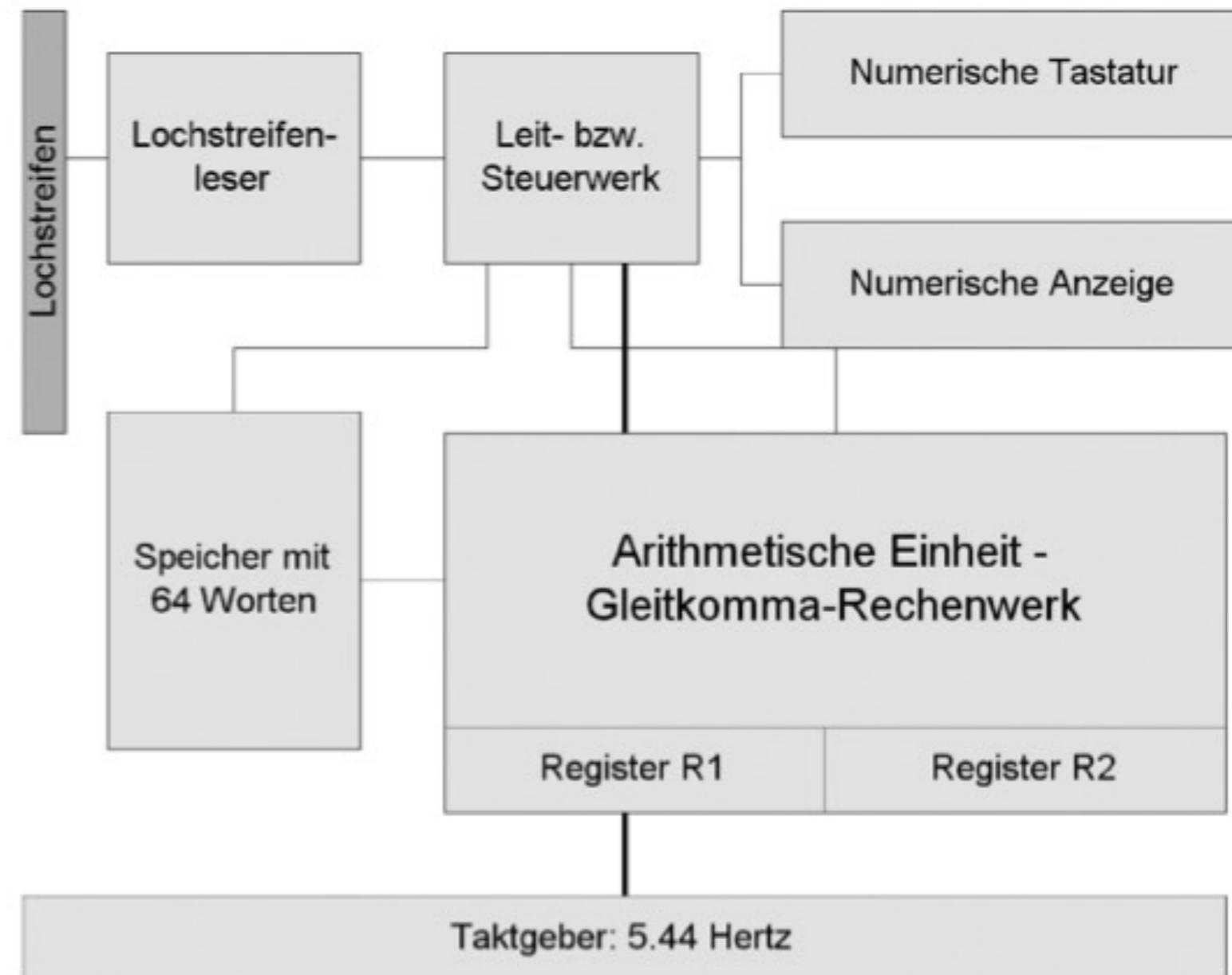


# Z3

- › Auch Speicher mit Relais
- › 64 Speicherzellen mit 22 bit
- › Insgesamt 2600 Relais
- › Stromverbrauch: 4'000 Watt
- › 3s für Multiplikation, 0.6s für Addition



# Blockschaltbild der Z3



## Zitat von Thomas Watson, IBM (1943)

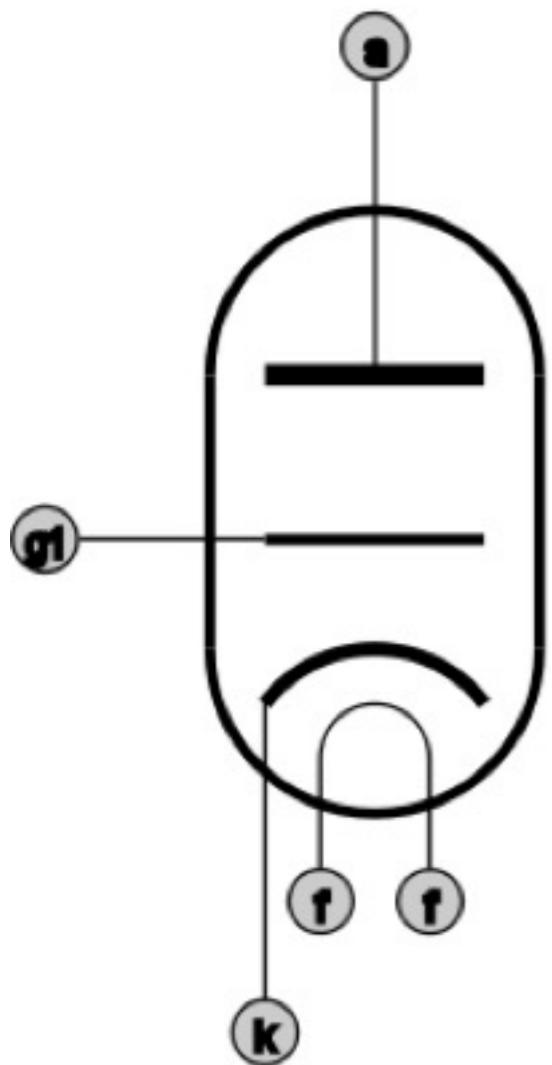
---

> „I think there is a world market for maybe five computers.“

(Thomas Watson, IBM)



# Vakuumröhre



a: Anode  
g1: Gitter  
k: Kathode  
f: Heizung

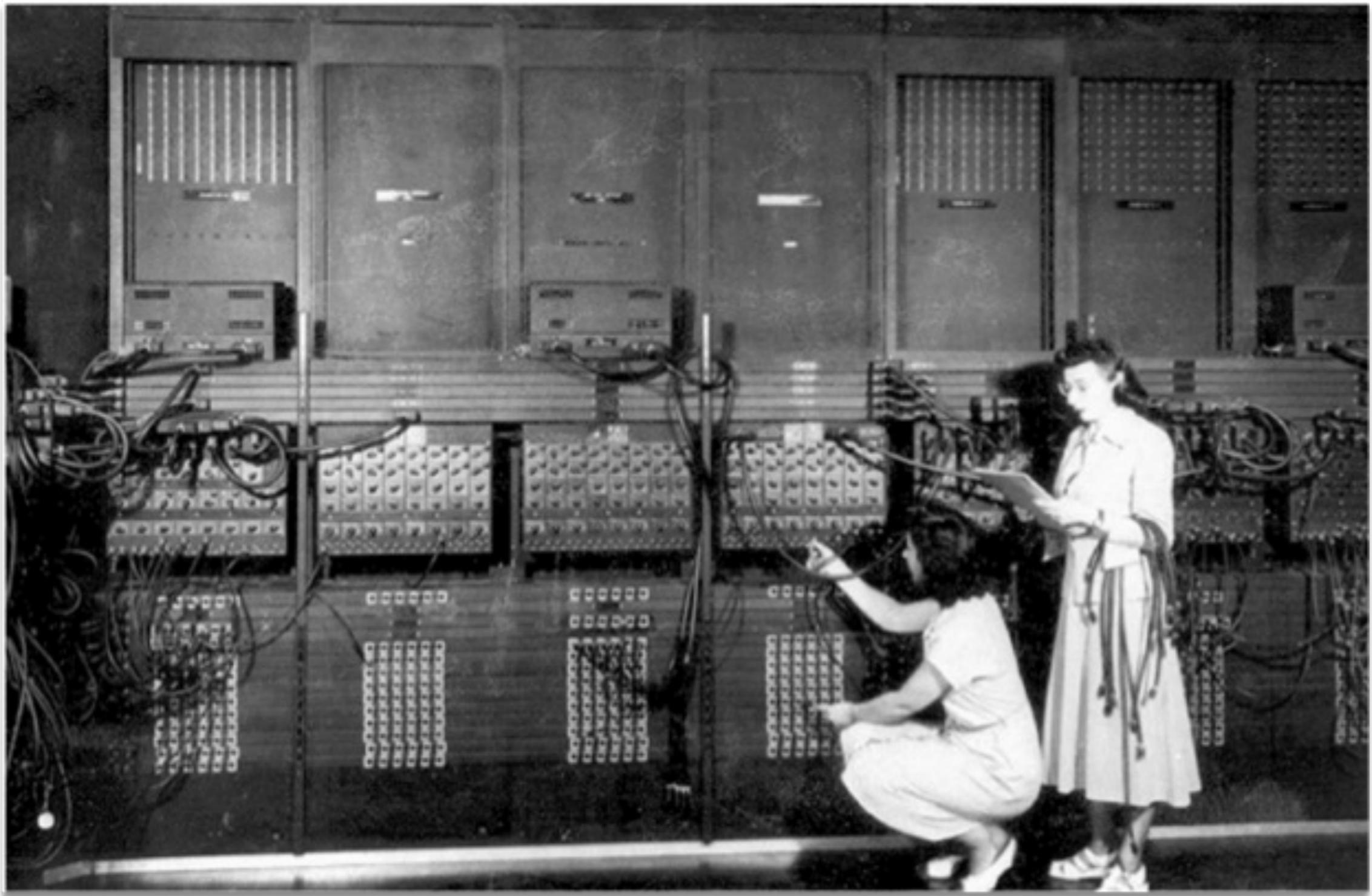


# 1946 ENIAC

---

- > Electronic Numerical Integrator and Computer
- > 17'468 Vakuumröhren (Schalter ohne Mechanik) anstelle von Relais
- > 1'000 mal schneller als Z3
- > Verwendet Dezimalsystem
- > Grösse: 24 x 3 x 1 Meter
- > Gewicht: 30 Tonnen
- > Programmieren via Verdrahtung
- > Vakuumröhren sind sehr anfällig (pro Stunde ein Ausfall)
- > 1951: Nachfolger UNIVAC, 46 verkaufte Rechner (erster kommerzieller Rechner)
- > Preis: 1'000'000 US\$

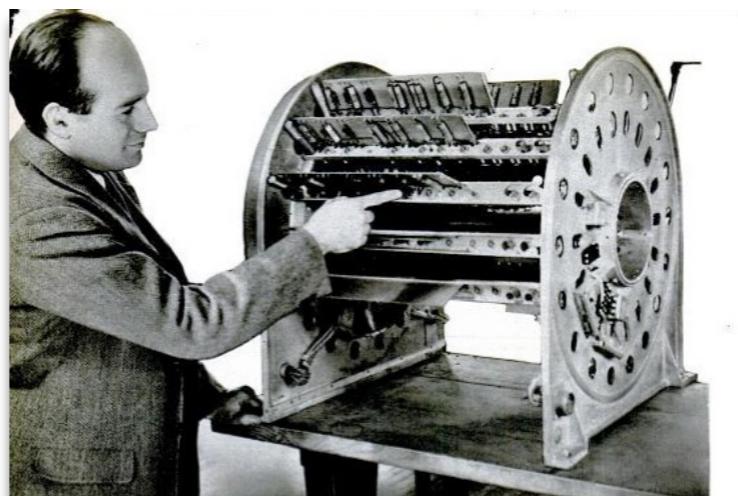
# ENIAC



# Zitat (1949)

> „Computers in the future may weight no more than 1.5 tons.“

(Popular Mechanics, March 1949, Volume 91, nr. 3)



## Brains that Click

By Andrew Hamilton

YOU KNOW HOW it is when you have some figuring to do—your income tax, plans for a new house, or perhaps next year's vacation. You work yourself into a frustrated frenzy and usually come up with several different answers. There's addition, subtraction, multiplication, division, fractions, percentages, square roots, decimals...

More than one weary pencil pusher has wished aloud: "Boy, if I only had a machine to do this!"

For 300 years scientists have been working on mechanical devices to take the drudgery out of mathematics. But it's only recently that they've come up with anything worth shouting about.

Today, in the mathematical and engineering laboratories of America's universities and research centers, giant robots are being built that can do complicated problems thousands of times faster than the human brain, store up long strings of figures in their "memories," flash red lights and ring bells when they get out of whack, and tackle problems so complex it staggers the imagination.

"Some of these machines are almost human," a mathematician said recently. "It gives you the creeps to think what they can do."

The high-speed calculators of science, however, don't remotely resemble the popular notion of a mechanical robot. Instead, they are huge mechanical and electronic machines that cost up to \$750,000, weigh as much as 100 tons, fill whole rooms and are equipped with thousands of vacuum tubes and millions of feet of wire.

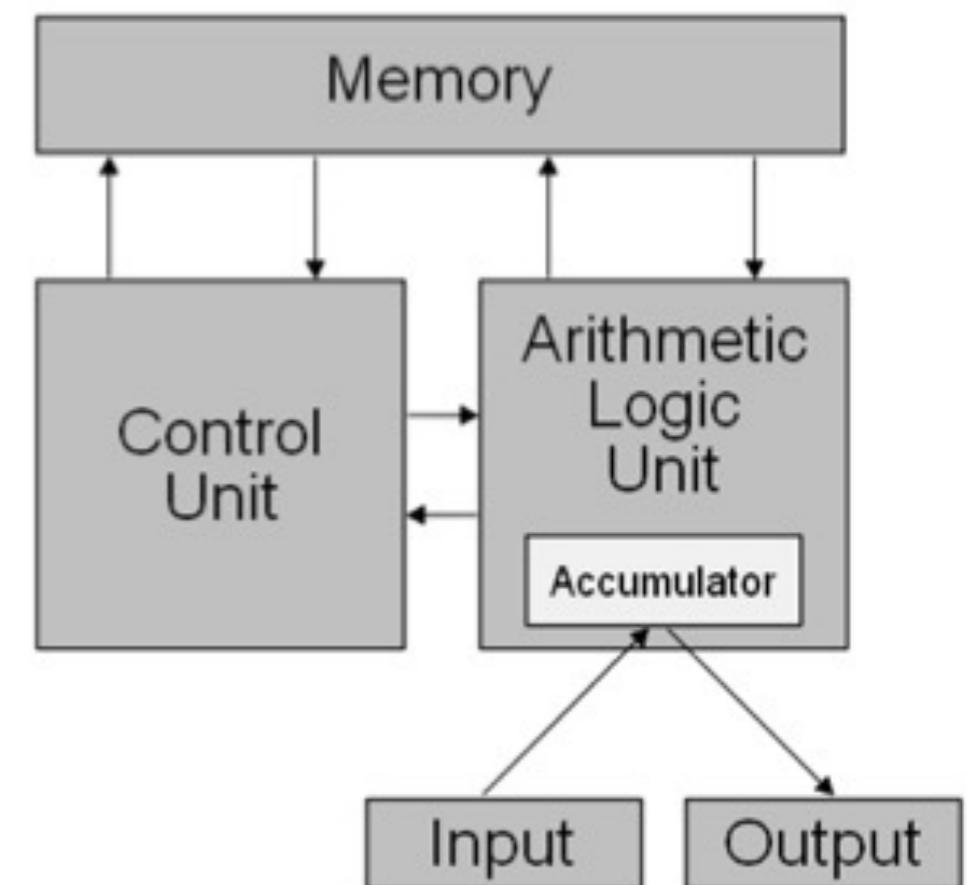
If you can imagine a Ruba Goldberg combination radio-phonograph-telephone switchboard-organ console-typewriter-newspaper teletype-pin-ball machine, you will have a faint notion of a high-speed calculator's appearance.

These machines are unbelievably fast. An ordinary desk calculator will multiply two 10-digit numbers, such as 3,650,678,548 by 5,477,648,971 in about 10 seconds. A "thinking machine" will give you the multiplication in  $\frac{1}{1000}$  second. Here's another example: About 400,000 customs declarations are made each month and it takes

Engineers and mathematicians are like airplane designers. Models in use are already long outmoded by those on the drawing boards. Where a calculator like the ENIAC today is equipped with 18,000 vacuum tubes and weighs 30 tons, computers in the future may have only 1000 vacuum tubes and perhaps weigh only 1½ tons.

# 1952: IAS, von Neumann Rechner

- > John von Neumann (1903-1957)
- > Institute for Advanced Study (Princeton University)
- > Rechnungen können nur schnell ausgeführt werden, wenn der Maschine auch schnell Befehle erteilt werden können
- > von Neumann Architektur
  - Interne Speicherung der Programme
  - Gemeinsamer Speicher für Daten und Programme
  - Fetch-Decode-Execute Zyklus



# IAS-Rechner

---



## Zusatzliteratur auf Ilias:

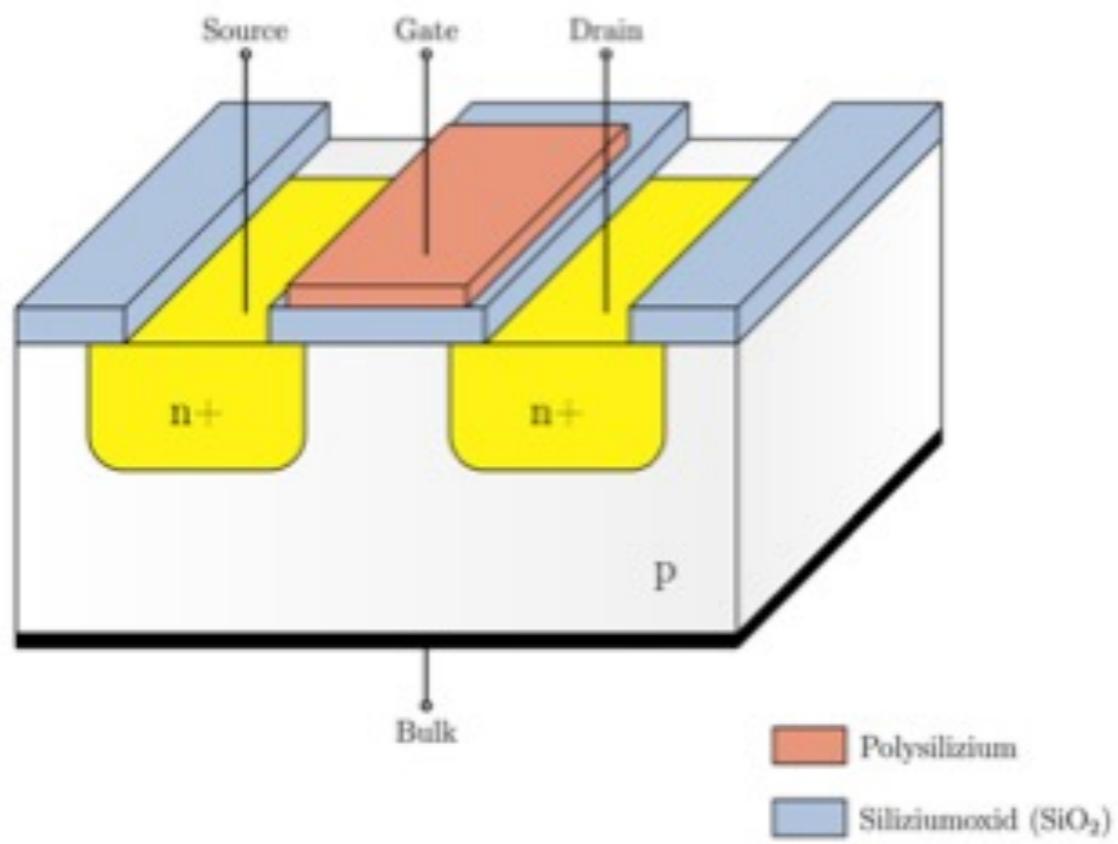
---

### The Stored-Program Universal Computer: Did Zuse Anticipate Turing and von Neumann?

B. Jack Copeland and Giovanni Sommaruga

**Abstract** This chapter sets out the early history of the stored-program concept. The several distinct ‘onion skins’ making up the concept emerged slowly over a ten-year period, giving rise to a number of different programming paradigms. A notation is developed for describing different aspects of the stored-program concept. Theoretical contributions by Turing, Zuse, Eckert, Mauchly, and von Neumann are analysed, followed by a comparative study of the first practical implementations of stored-programming, at the Aberdeen Ballistic Research Laboratory in the US and the University Manchester in the UK. Turing’s concept of universality is also examined, and an assessment is provided of claims that various historic computers—including Babbage’s Analytical Engine, Flowers’ Colossus and Zuse’s Z3—were universal. The chapter begins with a discussion of the work of the great German pioneer of computing, Konrad Zuse.

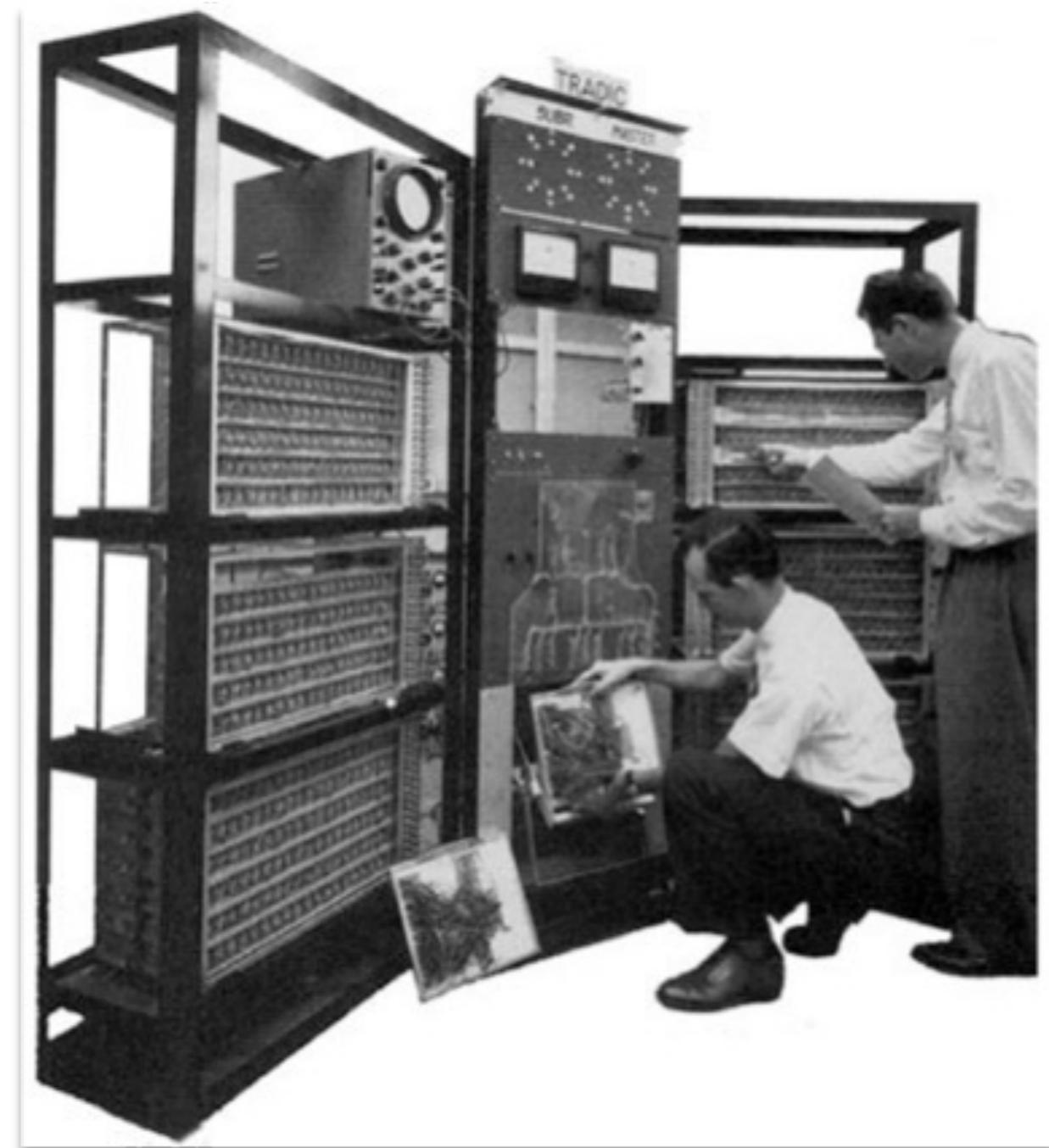
# Transistoren



- > Unter einem **Halbleiter** versteht man einen Festkörper, den man hinsichtlich seiner elektrischen Leitfähigkeit sowohl als Leiter als auch als Nichtleiter betrachten kann.

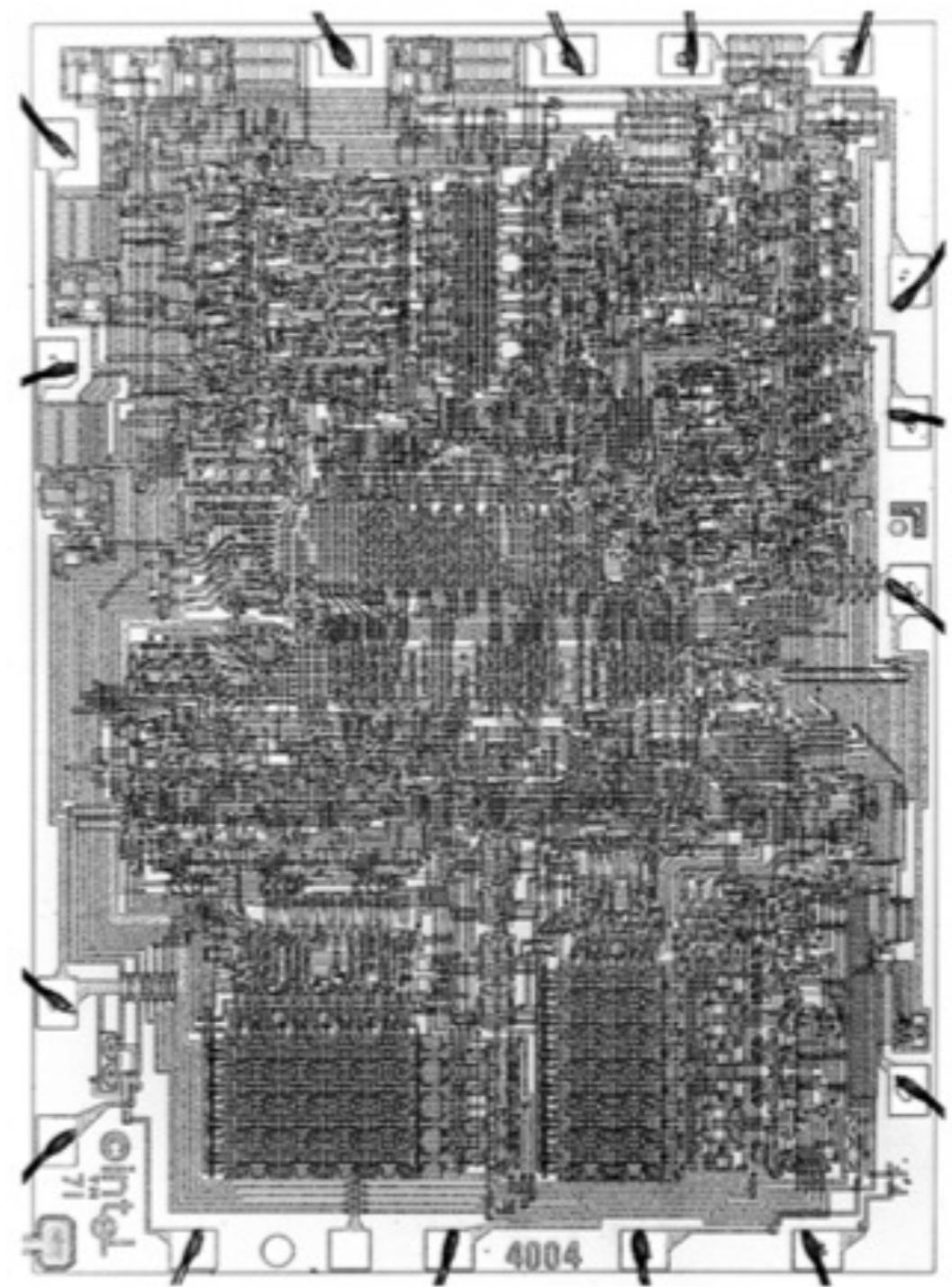
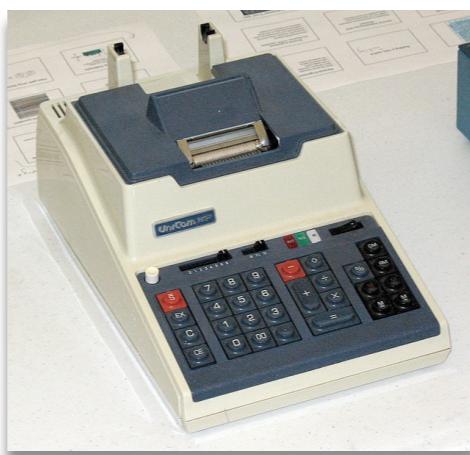
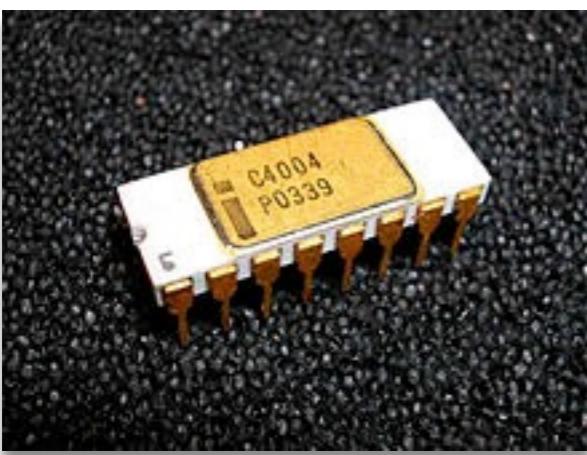
# 1953: TRADIC

- > Transistoren anstelle von Vakuumröhren
- > 700 Transistoren + 10'000 Dioden
- > Zuverlässiger
- > Schneller
- > Brauchen etwa 10mal weniger Strom
- > 100 Watt



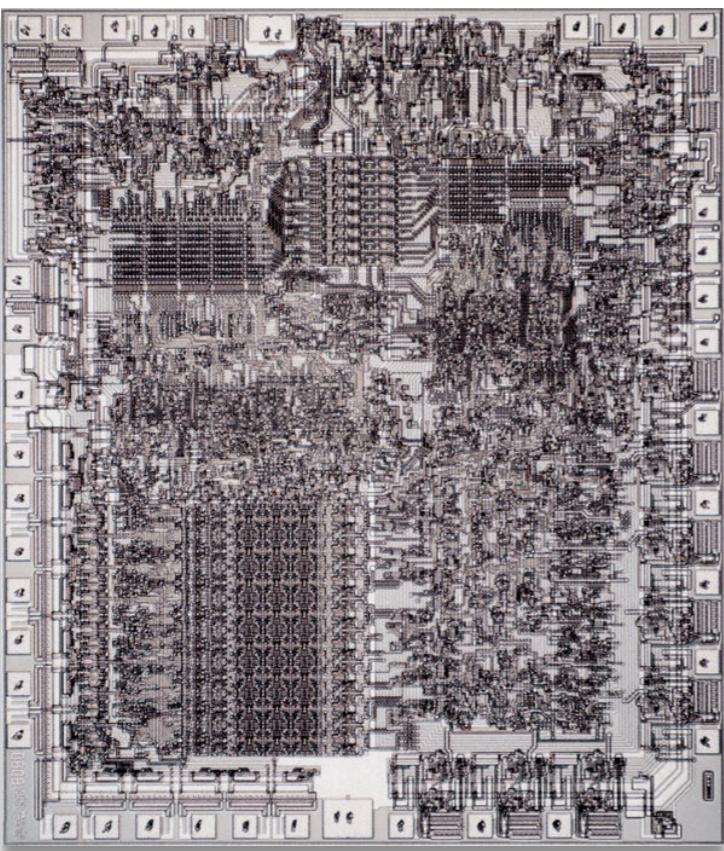
# 1971: Intel 4004

- > Erster Mikroprozessor (auf einem einzigen Chip integriert)
- > 46 Befehle
- > 2300 Transistoren
- > 108 kHz
- > Spezifikation von Busicom (Schaltung für Tischrechner) ignoriert  
-> general-purpose-Prozessor



# 1974: Intel 8080

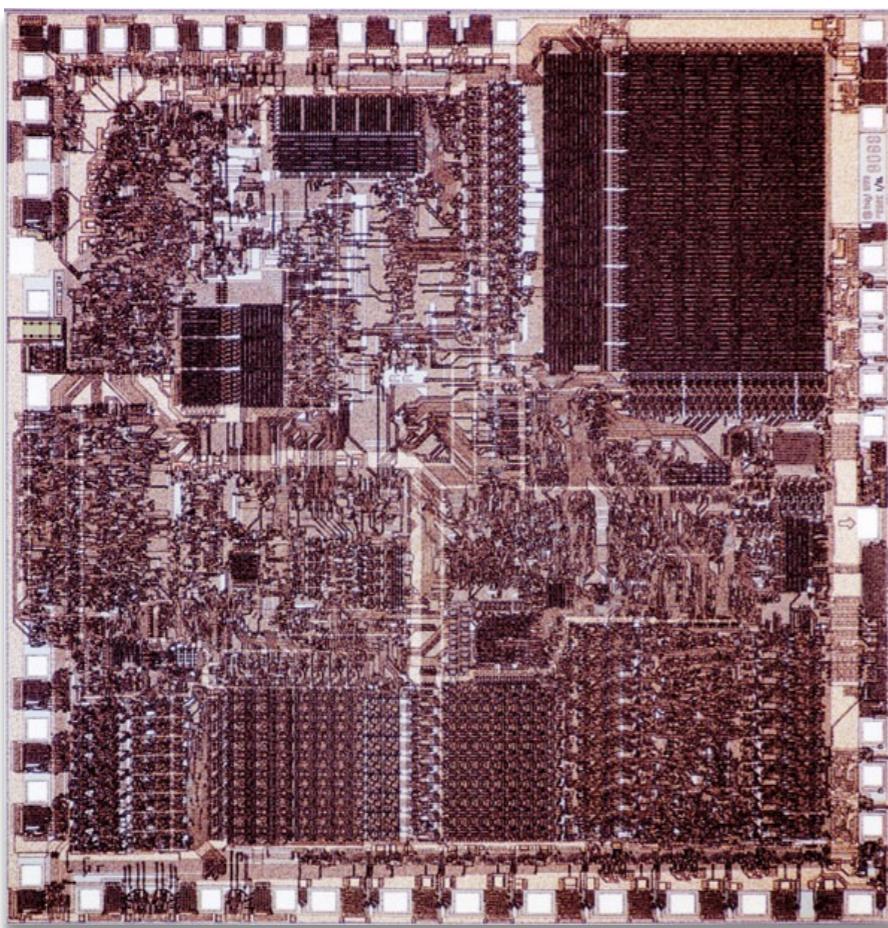
- › 8-Bit
- › 2 MHz
- › 5'000 Transistoren
- › Benutzt z.B. in Altair 8800



# 1978: Intel 8086

---

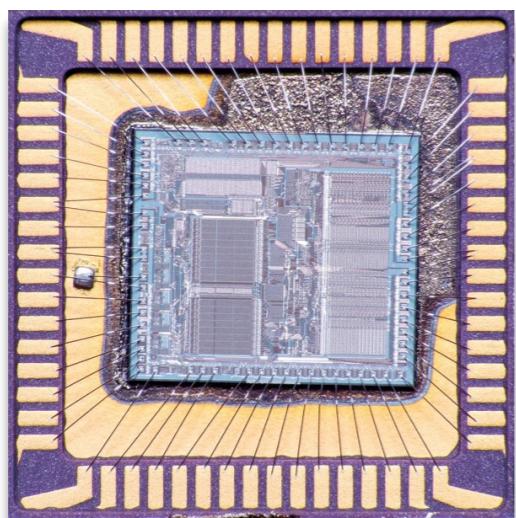
- › 16-Bit
- › 5 MHz
- › 29'000 Transistoren
- › Complex Instruction Set Computer (**CISC**)



# 1979: Motorola 68000

---

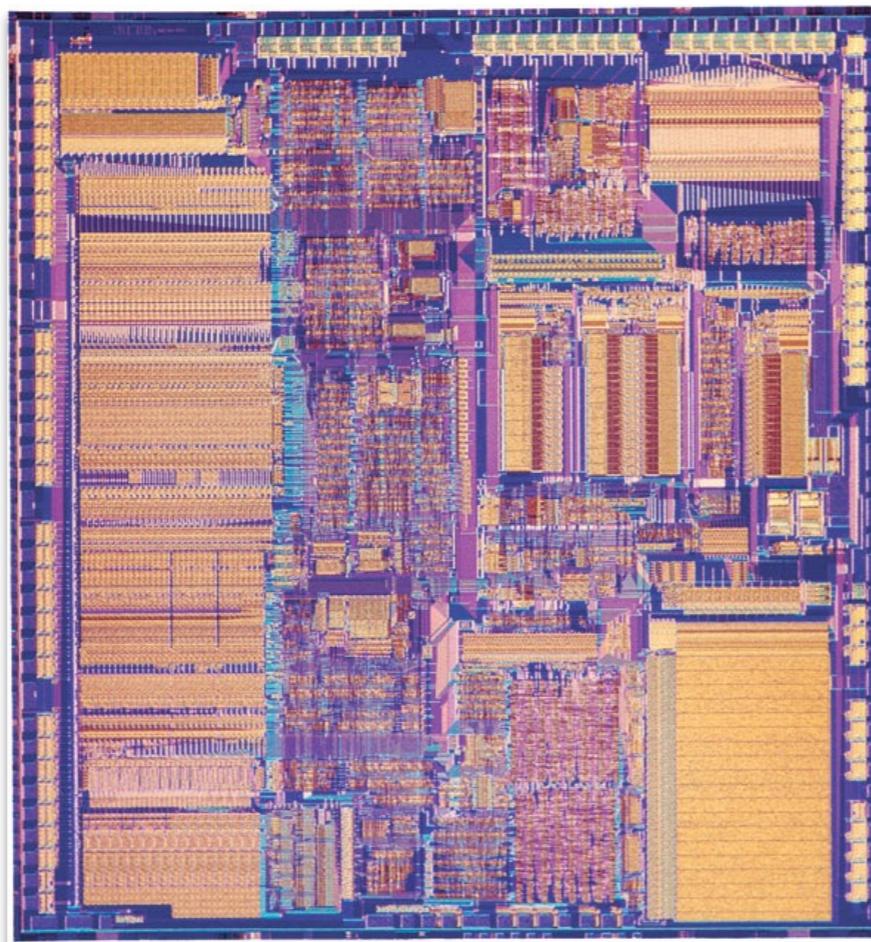
- › 16/32-Bit
- › 8 MHz
- › **CISC**
- › 68'000 Transistoren
- › Benutzt z.B. in Apple Macintosh 128K (1983)



# 1985: Intel 80386

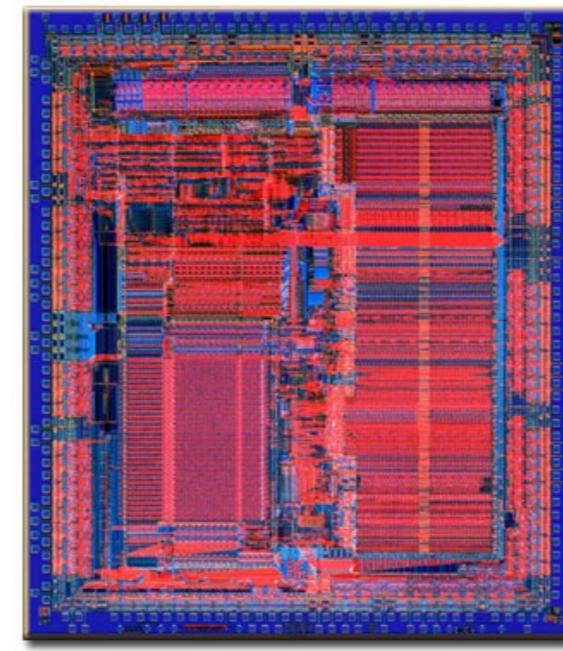
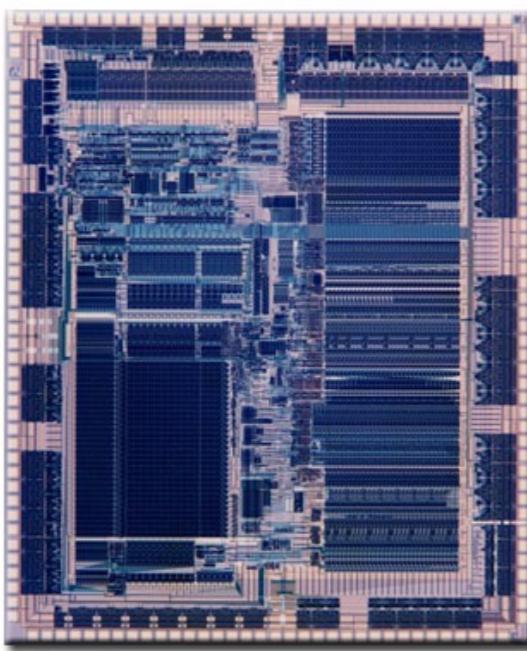
---

- › 32-Bit
- › 12 MHz
- › CISC
- › 275'000 Transistoren



# 1986: MIPS R2000, RISC Prozessor

- › Microprocessor without Interlocked Pipeline Stages
- › Reduced Instruction Set Computer
- › MIPS R2000: 32-Bit, 110'000 Transistoren, 16,7 MHz
- › Idee: Rechner beschleunigen, indem nur einfache Maschinenbefehle zur Verfügung stehen, die alle gleichviel Zeit benötigen.
- › Dadurch: effiziente Befehlspipeline



# Pipeline

---

**Waschen, Trocknen, Bügeln**

**Ohne Pipeline**



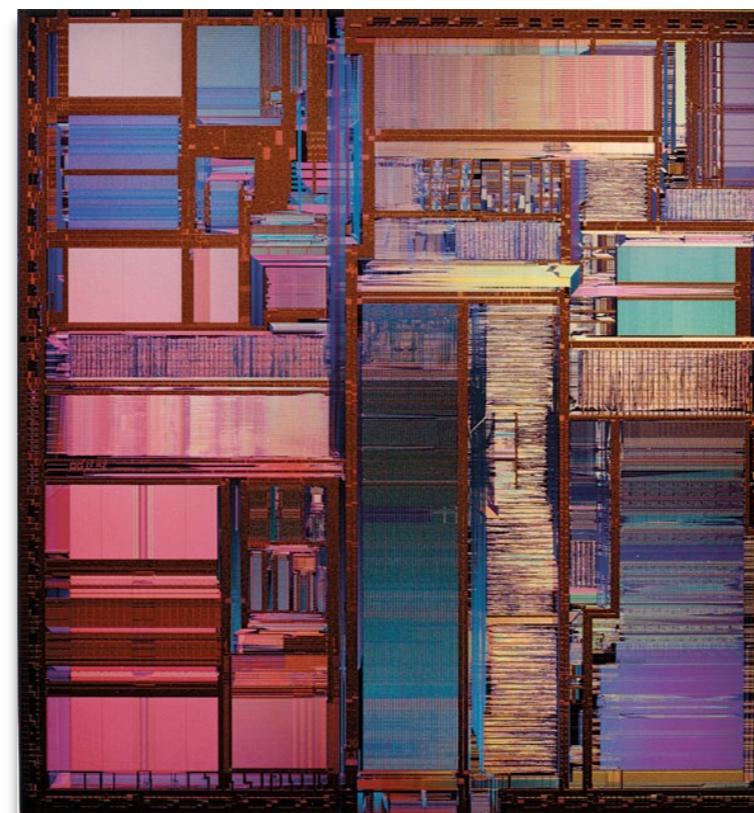
**Mit Pipeline**



# 1993: Intel Pentium

---

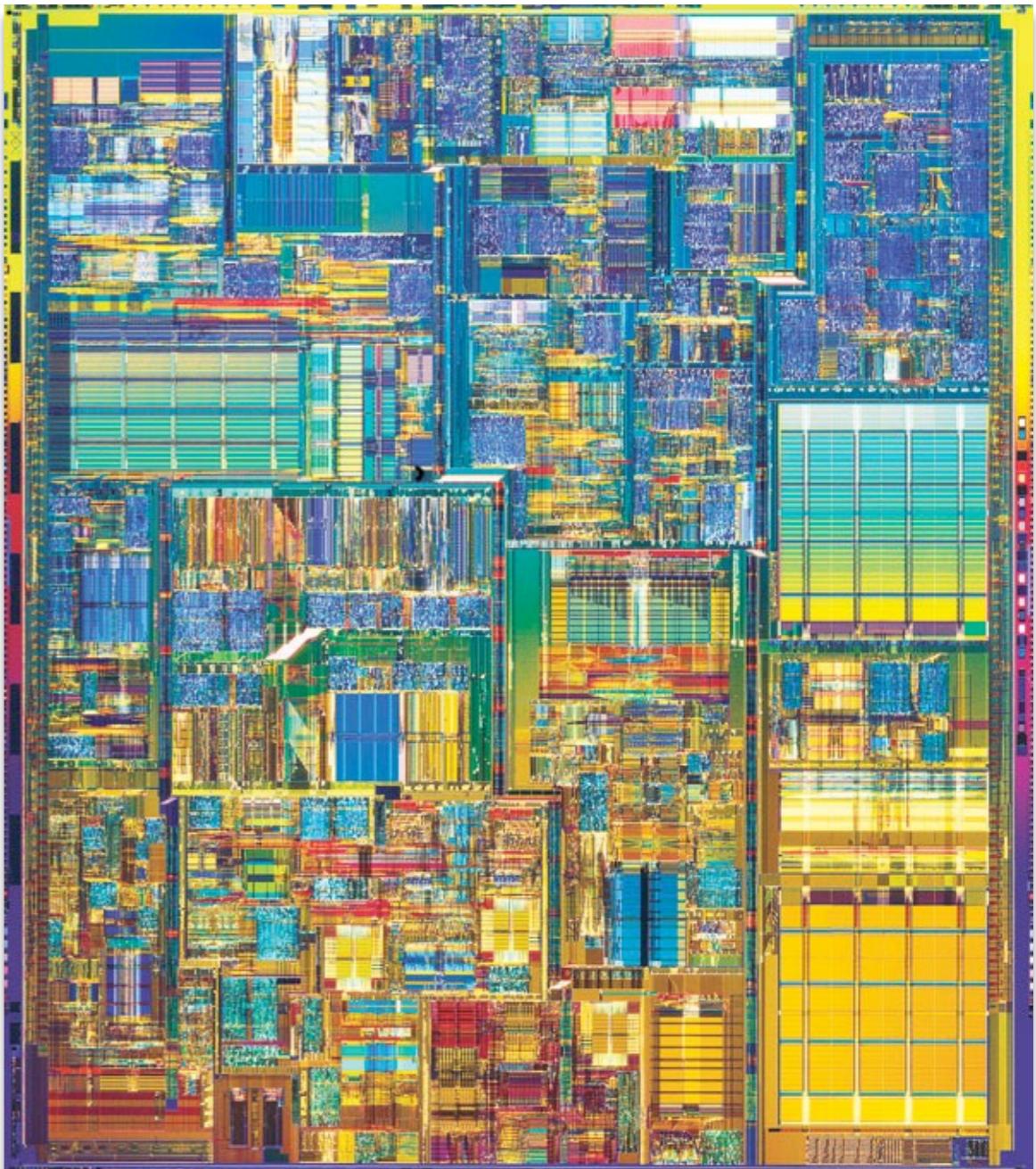
- › 3'100'000 Transistoren
- › 60 MHz
- › 32-Bit
- › Erster superskalarer CISC-Mikroprozessor
- › Dynamische Sprungvorhersage (Branch Prediction)



## 2000: Intel Pentium 4

42'000'000 Transistoren  
1.4 GHz

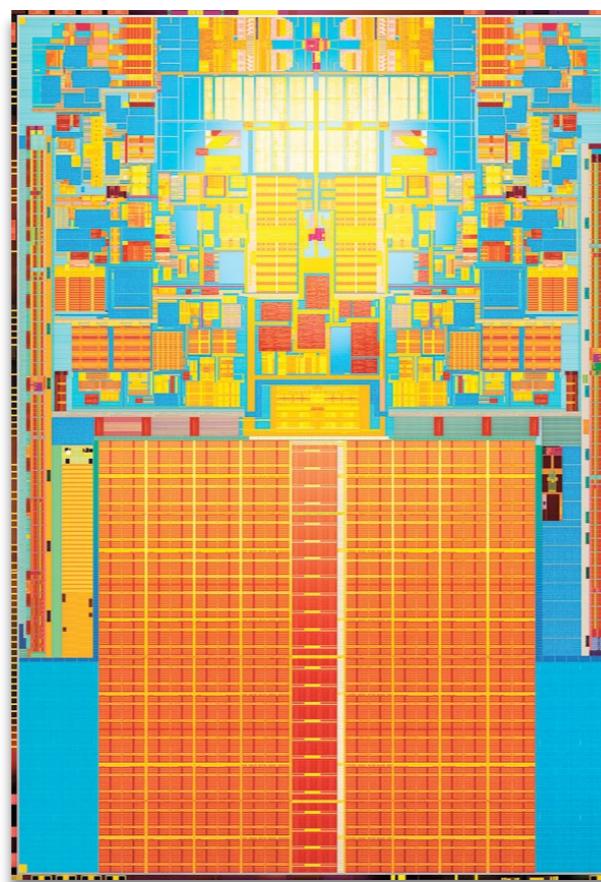
„Wenn die Automobilindustrie ein ähnliches Tempo vorgelegt hätte wie die Halbleiterindustrie, würde ein Rolls Royce heute pro Liter 200'000 Kilometer weit kommen und es wäre billiger, ihn wegzuwerfen als ihn zu parken.“  
(Gordon Moore, Intel Corporation)



## 2007: Intel Core 2 Duo / Quad

---

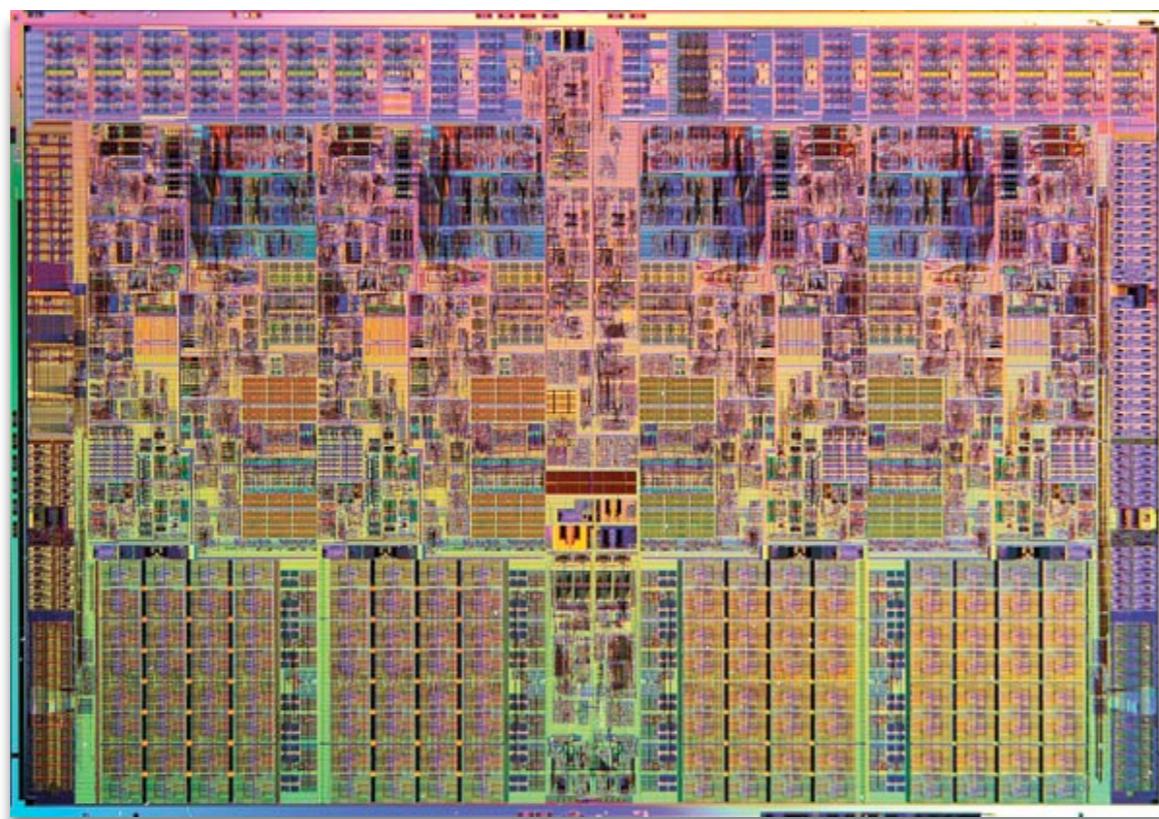
- › 410'000'000 Transistoren
- › 64-Bit
- › 1.06 - 3.33 GHz
- › Zwei-/Vierkernprozessor



## 2009: Intel Core i7

---

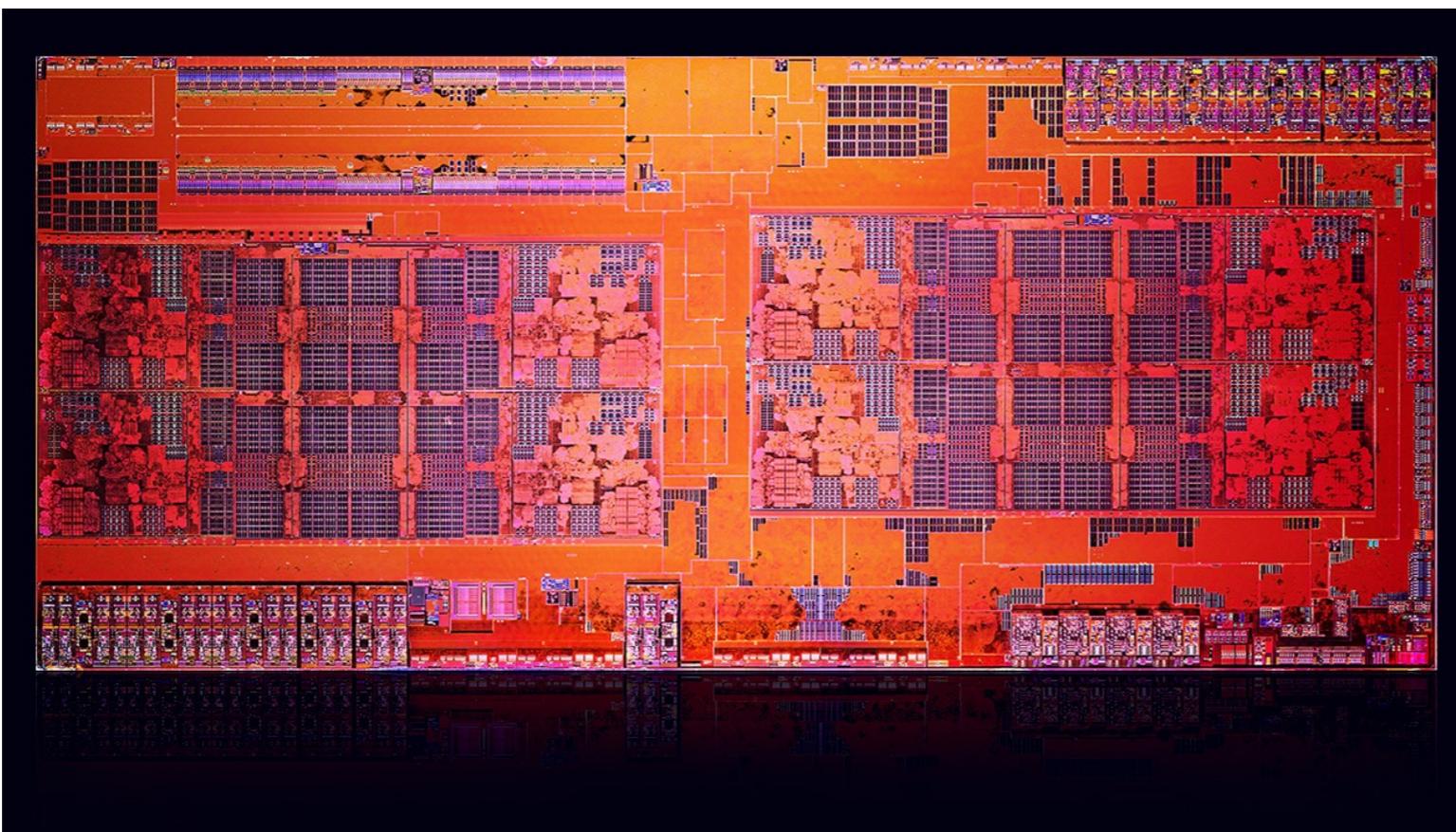
- > 731'000'000 Transistoren
- > 64-Bit
- > 4 Prozessorkerne
- > 2.67 - 3.33 GHz



# 2017: AMD Ryzen

---

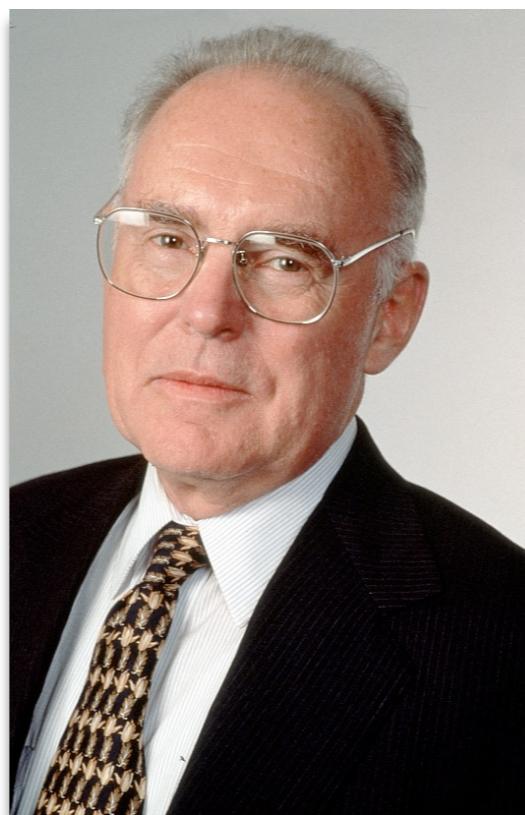
- > 4'800'000'000 Transistoren
- > 22'000'000 Transistoren pro mm<sup>2</sup> (geschätzt)
- > 16 Prozessorkerne / 32 Threads



# Mooresches Gesetz (Moore's Law)

---

- › Gordon E. Moore (Mitgründer von Intel, 1968)
- › Mooresches Gesetz
  - Ursprüngliche Formulierung von 1965: jährliche Verdopplung der Anzahl Bauteile pro Chip
  - 1975: Verdoppelung der Anzahl Bauteile alle 2 Jahre



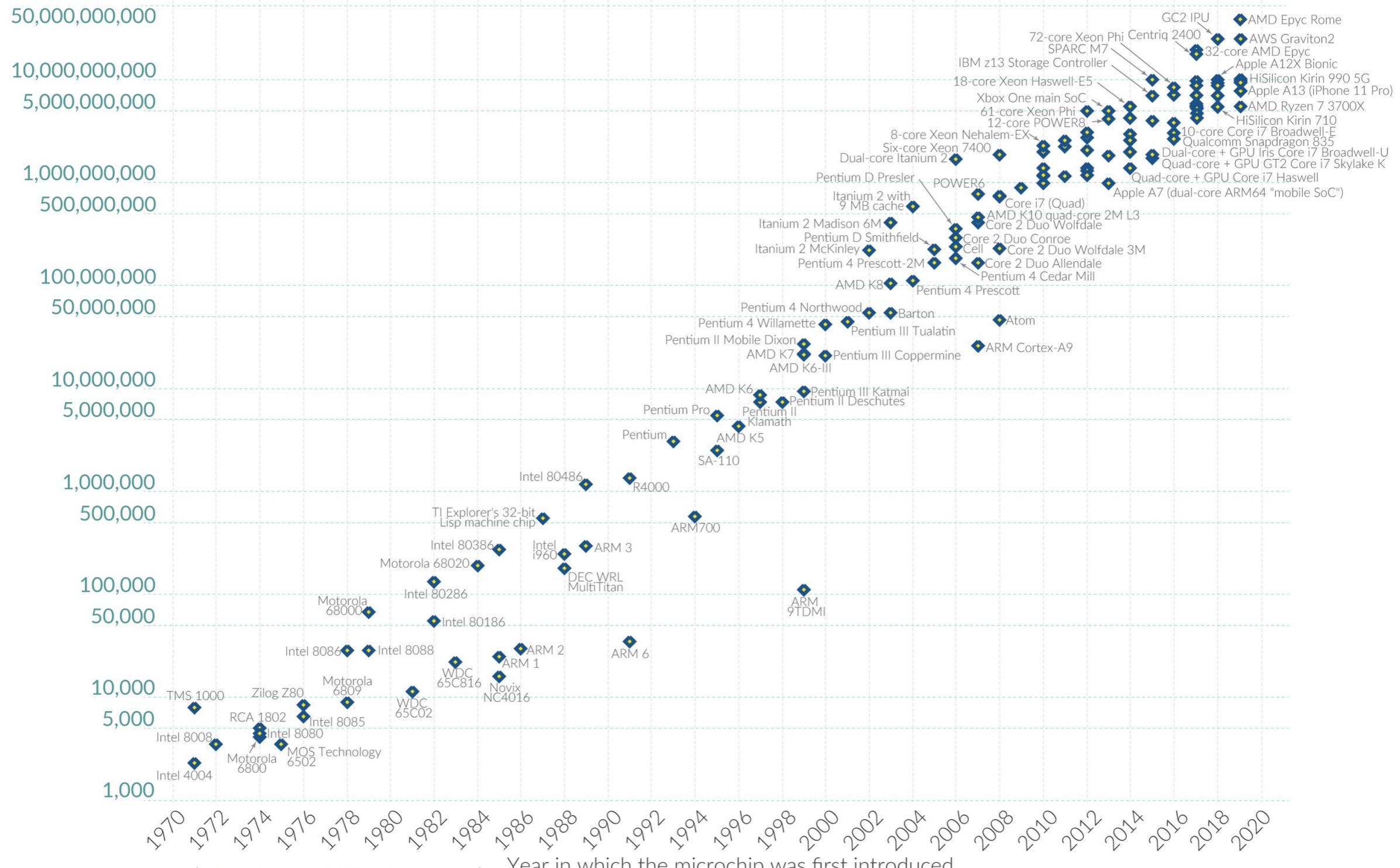
# Entwicklung von Mikroprozessoren

## Moore's Law: The number of transistors on microchips doubles every two years

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important for other aspects of technological progress in computing – such as processing speed or the price of computers.

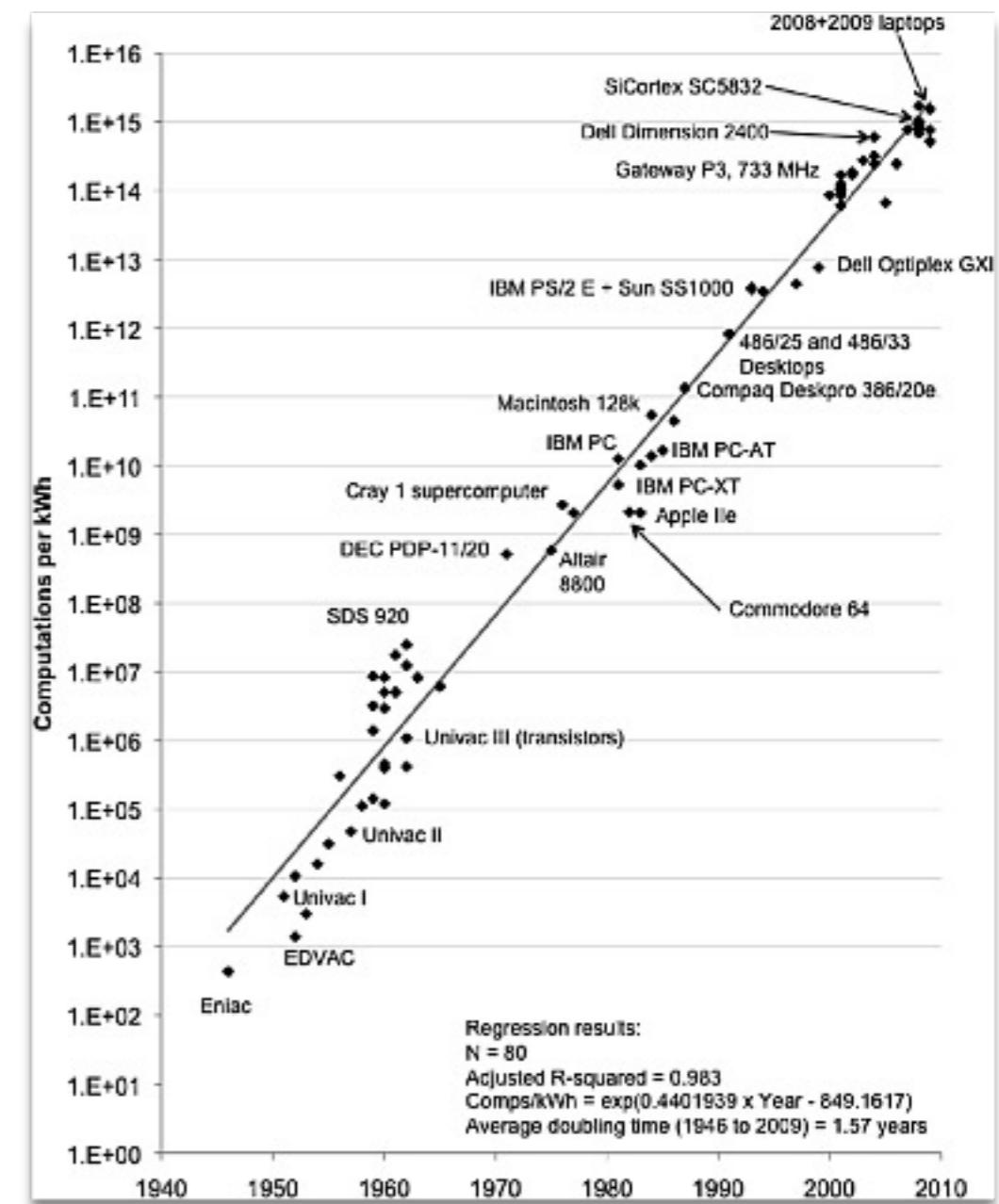
Our World  
in Data

### Transistor count



# 2011: „Koomey’s Law“

- > „What most observers do not know, however, is that the electrical efficiency of computing (the number of computations that can be completed per kilowatt-hour of electricity) also doubled about every 1.5 years over that period.“  
(J. G. Koomey, Stanford University)



# GTI - Grundlagen der Technischen Informatik

## Boolesche Logik

# George Boole (1815 - 1864)

---

- > The Laws of Thought (1854):

Logisches Denken wird auf das Lösen von Gleichungen reduziert.



# Boolesche Logik

---

- >  $x, y$  seien Kollektionen von Objekten
- >  $x + y$  ist die Kollektion von allen Objekten, die zu  $x$  oder zu  $y$  gehören (Vereinigung)
- >  $xy$  ist die Kollektion von allen Objekten, die zu  $x$  und zu  $y$  gehören (Schnitt)
- >  $0$  ist die leere Kollektion

# Boolesche Logik

---

- > Alle s sind p s = sp
- > Kein s ist p sp = 0
- > Einige s sind p sp ≠ 0
- > Einige s sind nicht p s ≠ sp
  
- > Rechenregeln:  
 $(xy)z = x(yz)$   
 $x0 = 0$

# Boolesche Logik

---

- > Beispiel:
  - b: Berner
  - s: Schweizer
  - m: auf dem Mond gewesen
- > Alle b sind s                     $b = bs$
- > Kein s ist m                     $sm = 0$
- >  $bm = (bs)m = b(sm) = b0 = 0,$
- > Wir haben  $bm = 0$ , also kein b ist m

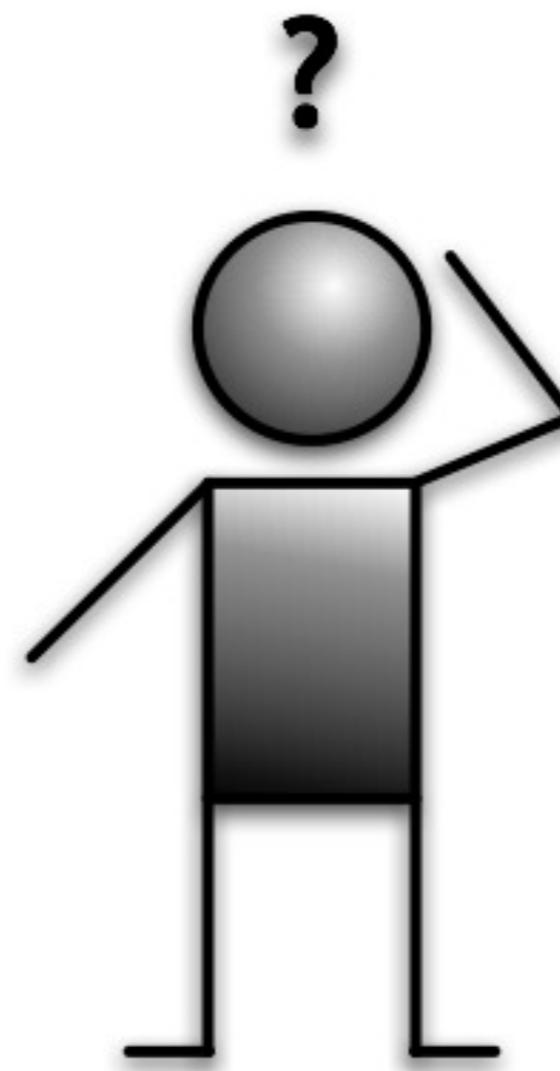
# Rückblick

---

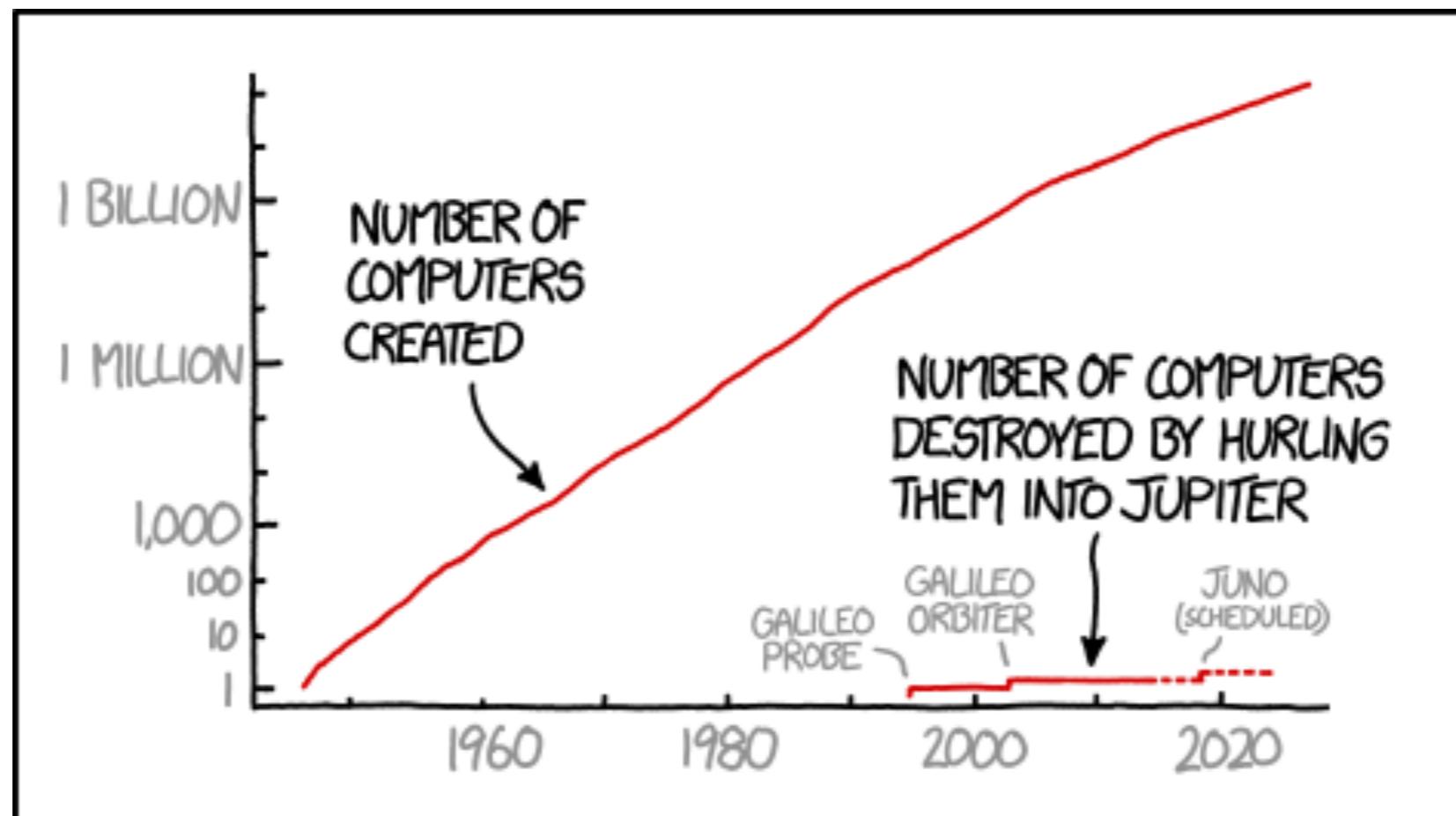
- > Organisatorisches
- > Geschichte
- > Boole'sche Logik

# Fragen?

---



# Zum Schluss



NASA NEEDS TO PICK UP THE PACE IF  
THEY EVER WANT TO FINISH THE JOB.

Quelle: <https://xkcd.com/1727/>