

# Kapitel 0

## Einführung

Albert-Ludwigs-Universität Freiburg

Prof. Dr. Christoph Scholl

Institut für Informatik  
WS 2015/16

UNI  
FREIBURG

# Einleitung (1)

---

- Technische Informatik beschäftigt sich allgemein mit Hardware.
  - Programmierbare Universalrechner
  - Programmierbare Eingebettete Rechner
  - Spezialhardware
- Technische Informatik beschäftigt sich mit
  - Theoretischen Grundlagen zum Entwurf von Hardware,
  - der Methodik zum Entwurf von Hardware,
  - dem Aufbau von Rechnern,
  - ...

# Einleitung (2)

---

- **Hardware** und **Software** (Algorithmen, Programme) bilden zusammen ein **Gesamtsystem**.
  - **Gesamtsystem** :=  
Die Gesamtheit von Komponenten, die miteinander durch Beziehungen verbunden sind und gemeinsam eine **Funktion** erfüllen.
  - Einführung in Hardware → **Technische Informatik**
  - Einführung in Software
    - → **Informatik I** (Programmierung)
    - → **Systeme I** (Betriebssysteme = hardwarenahe Software, Schnittstelle zwischen Hard- und Software)

# Technische Informatik

---

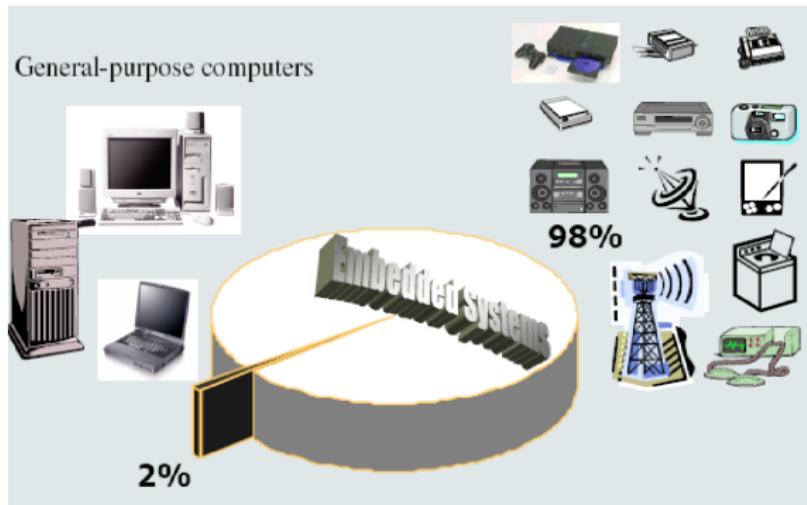
- Universalrechner (general purpose computer): PC, Arbeitsplatzrechner, Workstation, Großrechner
- eingebetteter Rechner *- eingebettet in ein großes technisches System* in Auto, Flugzeug, Waschmaschine, Fernseher, CD-Spieler, Spielkonsole, Navigationsgerät



## Marktanteile

Universalrechner  $\approx 2\%$

Eingebettete Rechner  $\approx$  98%



# Verstehen / Entwerfen von (Rechen-)Systemen

---

- Wie versteht / entwirft man ein System, welches aus sehr vielen Komponenten besteht:
  - einem Rechner, der aus 2 Milliarden Transistoren besteht
  - darauf laufender Software, Betriebssysteme, Übersetzer, Datenbanken, Kommunikationssysteme, Anwendungsprogramme mit Millionen Zeilen Programmcode
- Erkennen der **Struktur:** (*tieradiologer Aufbau*)
  - Aufbau aus **Komponenten**
  - **Zusammenwirken** dieser Komponenten zur Realisierung ihrer Funktion

# Inhalt der Vorlesung

---

## Teil 1:

Entwurf und Analyse eines einfachen Rechners

- Mathematische Grundlagen ←
- Abstrakter Rechner „ReTI“
- Kodierung von Zeichen, Zahlen
- Kombinatorische Logik
- Sequentielle Logik (*+ speichernde Elemente*)
- Formale Spezifikation und Verifikation von Hardware
- Fehlertoleranz und -erkennung

# Inhalt der Vorlesung

---

## Teil 2:

### Architekturkonzepte und ihre Umsetzung

- Befehlssätze
- Pipelining (*Fließbandverarbeitung*)
- Speicherbeschleunigung
- Parallelverarbeitung
- Spezialarchitekturen

# Literatur

---

- Bernd Becker, Paul Molitor  
„Technische Informatik: Eine einführende Darstellung“  
Oldenbourg Wissenschaftsverlag München,  
2008, ISBN 978-3-486-58650-3, 435 Seiten.
  
- Jörg Keller, Wolfgang J. Paul  
„Hardware-Design:  
Formaler Entwurf digitaler Schaltungen“  
Teubner, 2005.  
← ReTi!

# Ergänzende Literatur

---

- A. Tanenbaum  
Structured Computer Organization. Prentice Hall International, 2005.
- D. Patterson, J. Hennessy  
Computer Organization & Design - The Hardware/Software Interface.  
Morgan Kaufmann Publishers, 2008.
- J. Hennessy, D. Patterson  
Computer Architecture: A Quantitative Approach. Morgan Kaufman  
Publishers, 2011.
- R. Drechsler, B. Becker  
Graphenbasierte Funktionsdarstellungen. Teubner, 1998.
- P. Molitor, C. Scholl  
Datenstrukturen und effiziente Algorithmen für die Logiksynthese  
kombinatorischer Schaltungen. Teubner, 1999.
- G. Hotz  
Einführung in die Informatik. Teubner, 1990.

- Historische Entwicklung
  - von den Anfängen zum Multi-Core
  - Chips mit 2 Milliarden Transistoren
  - Probleme und Herausforderungen

# Wirtschaftliche Treiber der Computerentwicklung

---

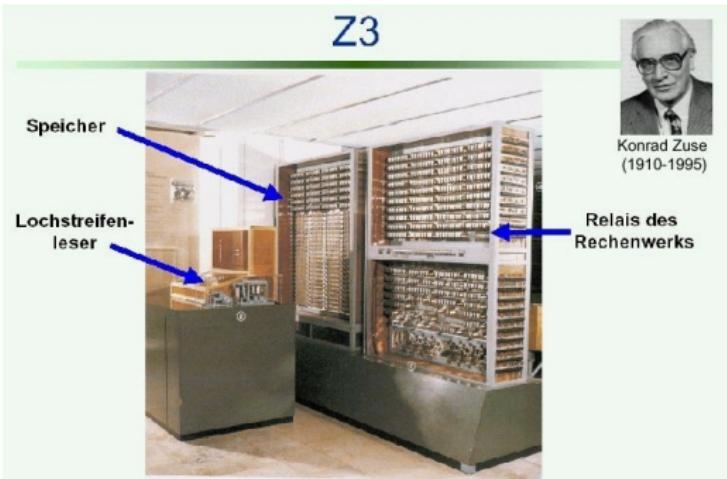
- **1930-60:** Wenige, sehr teure Rechner.
  - Wettervorhersagen, militärische Anwendungen
- **1960-80:** EDV für die Infrastruktur.
  - Banken, Verkehrs-, Energiesysteme
- **1980-2000:** Rechnerbasierte Konsumgüter.
  - PCs, Unterhaltungselektronik, Autos, Handys
- **Heute:** Ubiquitous Computing. *→ allgegenwärtiges Computing*
  - Smartphones/Internet Devices, Geräte zur medizinischen Überwachung

# Technischer Fortschritt als Basis der Entwicklung

---

- Hardware-Fertigungstechnologie
  - Moore's Law: Verdoppelung der Anzahl Transistoren pro Flächeneinheit alle 18 Monate
- Neue Algorithmenklassen
  - Bilderkennung, schnelle Kommunikation, Data Mining / Suchmaschinen, ...
- Neue Entwurfsverfahren
  - Entwurf deutlich komplexerer Systeme in gleicher / unterproportional erhöhter Zeit

# Zuse Z1 / Z3 (1934-1941)



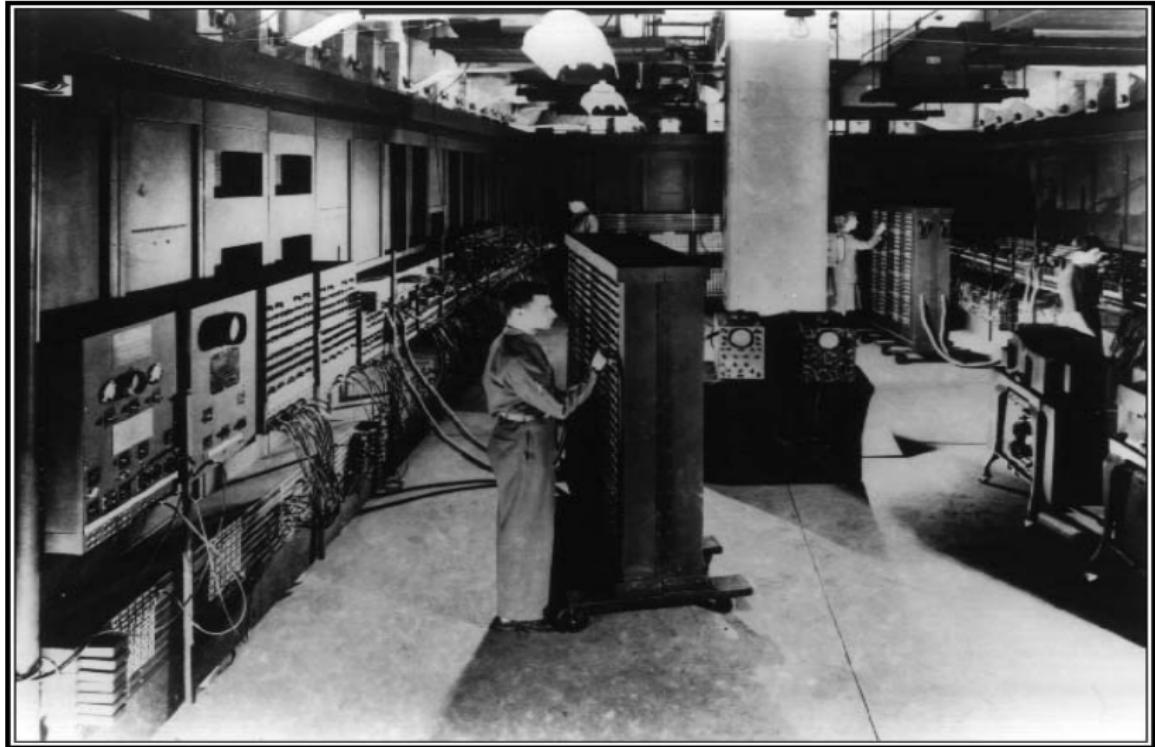
- Bis zu 10 Hz Taktfrequenz
- 64 Speicherzellen der Breite 22
- 4000 Watt Leistung
- Gewicht: 1000kg
- Programmierung über gelochte Filmstreifen

# Historische Entwicklung: ENIAC (1946)

---

- 30.000 kg schwer, 3 m hoch, 24 m breit
- 17.648 Vakuumröhren, 70.000 Widerstände, 1.500 Relais, 10.000 Kondensatoren
- 150.000 Watt Leistung
- Multiplikationszeit: 3 ms
- Programmierung über Schalttafeln

# Eniac



# Historische Entwicklung: Zitate

---

- Thomas Watson (IBM, 1943):

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

- Popular Mechanics (1949):

*„Computers in the future may weigh no more than 1.5 tons.“*

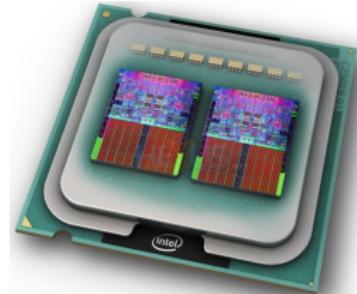
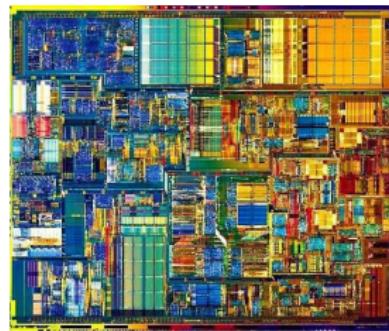
- Ken Olsen (DEC, 1977):

*„There is no reason for any individual to have a computer in their home.“*

# Heutiger Stand: Mikroprozessoren

---

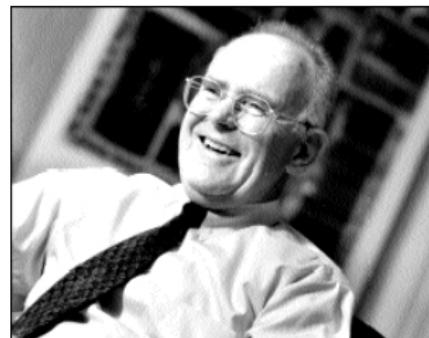
- 2000: Pentium 4
  - 42 Millionen Transistoren
  - Taktfrequenz > 1,5 GHz
- 2014: Intel Core i7 Haswell-E
  - $\approx$  2,6 Milliarden Transistoren
  - Taktfrequenz  $\approx$  4 GHz
- „Multi-Core“ (> 60 Kerne)
  - teilweise > 5 Milliarden Transistoren
  - Parallelausführung mehrerer Programme
  - Parallelausführung von Teilen desselben Programms



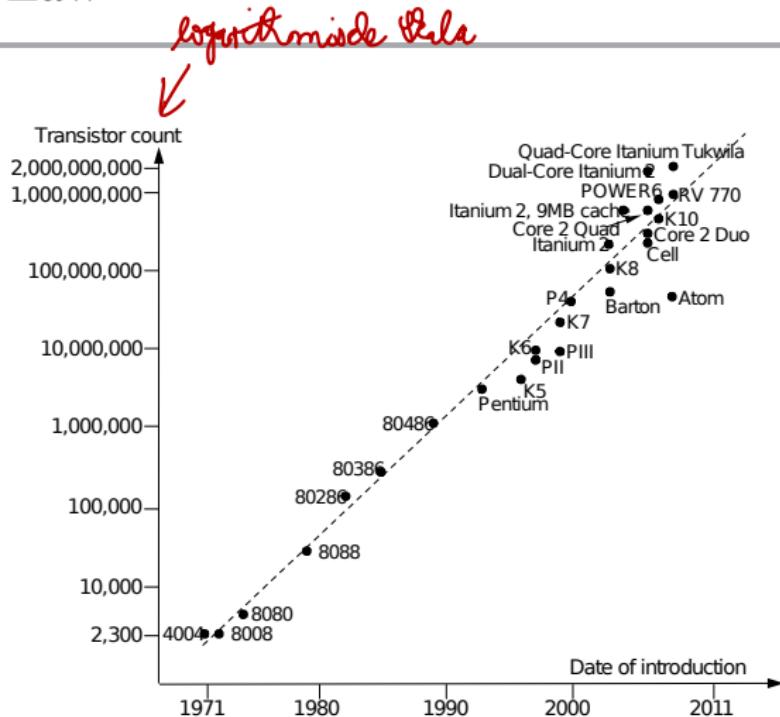
# Fortschritt der Halbleiterfertigung: Moore's Law



- Verdoppelung der Transistor-Dichte alle 18 Monate.  
(Gordon Moore, Mitbegründer von Intel, 1965)



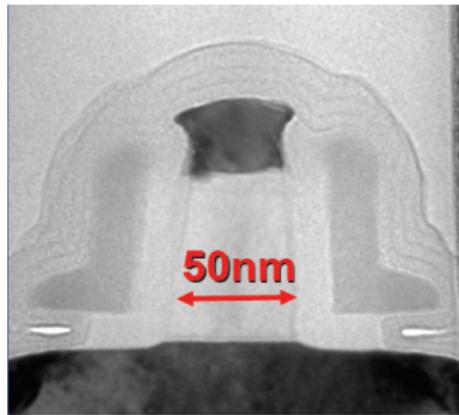
# Moore's Law



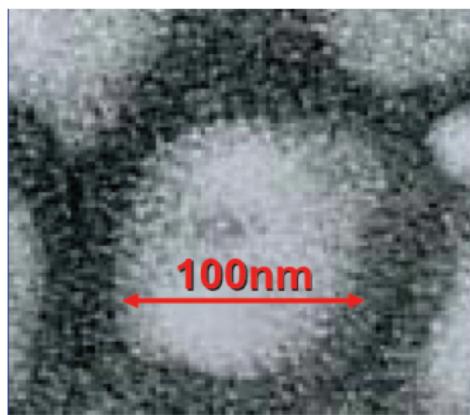
Kein Gesetz, sondern Voraussage, was technologisch möglich sein wird!

# Transistor

---



Transistor



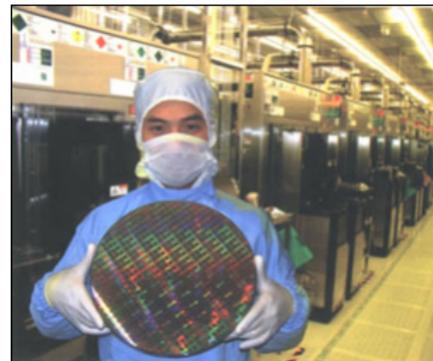
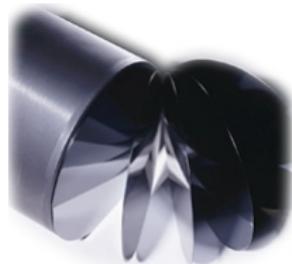
Grippevirus

- Intel arbeitet inzwischen mit 22nm Technologie (Ivy Bridge)

# Halbleiterfertigung im Überblick

---

- Siliziumkristalle werden gezüchtet und in dünne runde Scheiben (Wafers) geschnitten.
- Transistoren werden auf einen Wafer gefertigt (Frontend).
  - Transistoren bilden Logikgatter
- Metallische Verbindungsleitungen (Interconnects) werden über den Transistoren gefertigt (Backend).
  - Interconnects verbinden Logikgatter miteinander und stellen Stromversorgung bereit.



# Einige Fakten über Halbleiterfertigung

---

- Eine Intel-Fab neuester Generation kostet insgesamt ca. 3-5 Milliarden US-Dollar.  
(Memory-Fab von Samsung > 10 Milliarden US-Dollar.)
  - Die teuersten jemals gebauten Fabriken
  - Über 1.000 Maschinen, die mehr als 1 Mio \$ kosten
- Über 500 Fertigungsschritte.



# Fortschritt bei neuen Algorithmen

---

- Anwendungen, die auf der Basis früherer Hardware unmöglich wären.
- Beispiele:
  - Intelligente autonome Rettungsroboter
  - Identifikation von Sehenswürdigkeiten über Handy (lädt automatisch den Reiseführer)
  - Verteilter Zugriff auf größte Datenmengen
  - Wikipedia, Telemedizin, Routenplaner
- Mehr dazu in anderen Vorlesungen.

# Fortschritt bei Entwurfsverfahren

---

- Systeme bestehen aus **Hardware**, **Software** und gegebenenfalls **Zwischenschichten** (Betriebssystem und ähnliches).
- Für alle diese Bereiche gibt es effiziente **Entwurfsabläufe** („**Flows**“).
- Sie sind notwendig, um die Komplexität heutiger Systeme zu beherrschen und die Milliarden Transistoren auch zu nutzen.

# Vorgehen

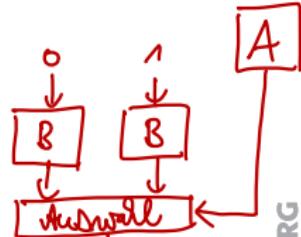
- Es sind **Entwurfsziele** zu erreichen, die oft einander widersprechen und gegeneinander abgewogen werden müssen (**Trade-Offs**).

- Systemabmessungen (physikalische Größe)
- Geschwindigkeit
- Energieverbrauch
- Entwicklungszeit (Time to Market)
- Kosten (Entwicklung / Fertigung)

*Variante 1:*



*Variante 2:*



*Illustration für Trade-off zwischen  
Räume und Zeit*

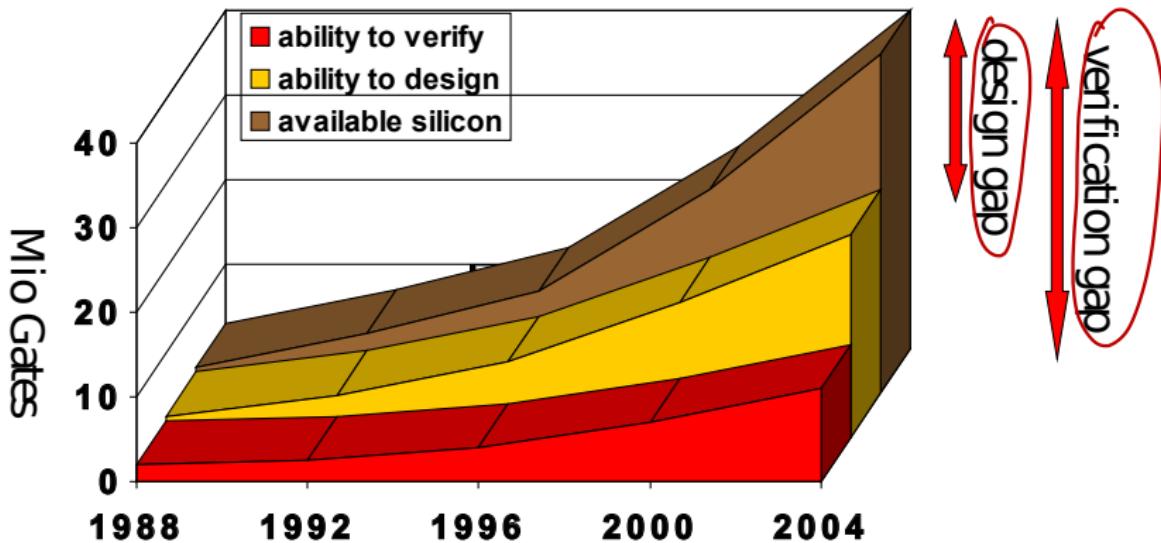
# Allgemeine Entwurfsprinzipien

---

- **Hierarchie:** Partitionierung des Systems in kleinere Blöcke; Entwurf der Blöcke und ihre spätere Zusammensetzung.
- **Entwurfsautomatisierung:** Verwende automatische Verfahren für möglichst viele Entwurfsschritte.
- **Wiederverwendung** von existierenden Designs (früher entworfen oder von Außen als „intellectual property core“ zugekauft).

*Bibliotheken* ← *Software*  
                             *Hardware*

# Design und Verification Gap



# Einige berühmte Beispiele von Entwurfsfehlern

---

- Pentium-Bug (*Fehler in der Hardware*).

1994

<http://www.intel.com/support/processors/pentium/fdiv/wp>

- Mars-Pathfinder-Mission (*Fehler auf Systemebene*).

*Eintritt =  
betriebszeitme*

[http://research.microsoft.com/en-us/um/people/mbj/mars\\_pathfinder/authoritative\\_account.html](http://research.microsoft.com/en-us/um/people/mbj/mars_pathfinder/authoritative_account.html)

- Ariane-Trägerraketen-Explosion (*Software der Vorgängerversion übernommen*).

<http://www.ima.umn.edu/~arnold/disasters/ariane5rep.html>

- Therac-25 Strahlentherapiemaschine (*Softwarefehler, Entwickler = Tester*).

[http://ieeexplore.ieee.org/xpls/abs\\_all.jsp?arnumber=274940](http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=274940)

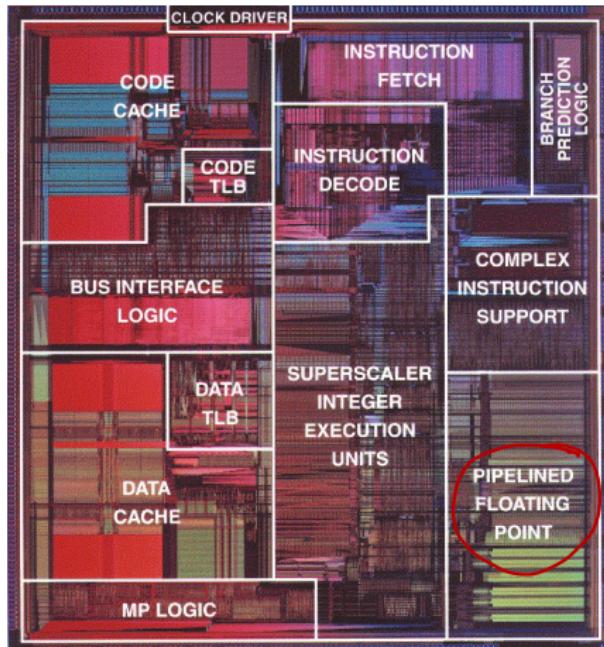
# Pentium Bug (1/2)

$$x = \underline{4195835},\underline{0}$$

$$y = \underline{3145727},\underline{0}$$

$$z = \underline{x} - (\underline{x}/\underline{y}) \times \underline{y}$$

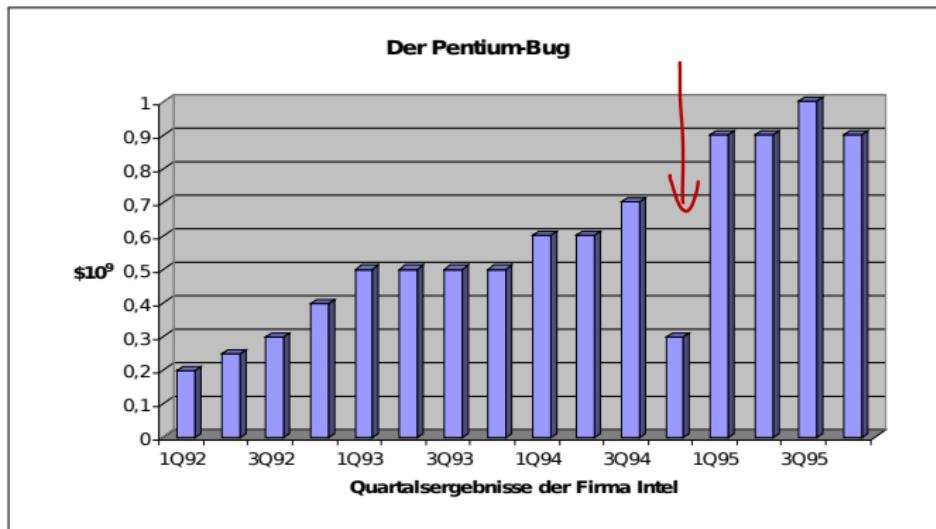
$$z = 256!$$



# Pentium Bug (2/2)

## ■ Fehler in Floating-Point-Einheit

- ⇒ Imageverlust für Intel
- ⇒ 480 Mio \$ zusätzliche Kosten für Intel im 4. Quartal 1994



# Herausforderungen (1/2)

---

- Wie entwerfen? Entwurfsproblematik - „Design-Gap“

→ VLSI-Entwurf, Entwurfsmethodik

Anforderungen: *VLSI = Very large scale integration*

- Entwurfszeit (Time to market)

- Arbeitszeit der Designer
  - Laufzeit der Algorithmen

- Energieverbrauch

- Entwurfskorrektheit (siehe z.B. Pentium-Bug)

- Effizienzanforderungen / Performanz

*Fläche  
Raufseite*

- Was tun mit all den Transistoren?

Sinnvolle Ausnutzung der verfügbaren Ressourcen

→ Architekturkonzepte, Rechnerarchitektur

# Herausforderungen (2/2)

---

## VLSI-Entwurf, Entwurfsmethodik

- Wird bei ReTI angedeutet ← TI !
- Rechnerarchitektur, erster Teil
- Spezialvorlesungen
  - Testen
  - Verifikation
  - Eingebettete Systeme
  - ...
- Projekte, Praktika (z.B. FPGA-Design)

## Architekturkonzepte

- Kapitel 8 ← TI
- Vorlesung Rechnerarchitektur, 2. Teil
- ...

Field Programmable Gate Arrays