

Rechnerarchitektur

Einführung

Univ.-Prof. Dr. Rainer Böhme

Wintersemester 2021/22 · 6. Oktober 2021

Ziele dieser Lehrveranstaltung

- Vermittlung von Grundwissen über den **Entwurf** von **Digitalschaltungen**
- Verständnis des **Aufbaus** und der **Arbeitsweise** von Hardware
- Verständnis von **Architekturprinzipien** und **Organisationsformen** moderner Rechner
- Einsicht in das Zusammenspiel von Hardware und Software
- Erstellung von **maschinennahen Programmen** am Beispiel der ARM-Architektur
- Lernen von Ansätzen zur **Bewertung** und zum Vergleich von Rechnersystemen



Organisation dieser Lehrveranstaltung

Vorlesung

- Zeit: mittwochs, 09:15–11:00 Uhr
- Ablauf: Stream auf https://twitch.tv/uibkseclab_de mit Chat-Rückkanal
- Umfang: 3 ECTS-AP European Credit Transfer System-Anrechnungspunkte

Proseminar

- Zeit: donnerstags, 17:00–18:00 Uhr, ab 7.10.
- Ablauf: Stream auf https://twitch.tv/uibkseclab_de
- Interaktion: Chat-Rückkanal und Online-Tests im eCampus LMS („OLAT“)
- Umfang: 2 ECTS-AP – **Anwesenheitspflicht!**

Tutorium

- Zeit: donnerstags, 09:15–10:00 Uhr, ab 14.10., freiwillig
- Ort: HS E
- Tutor: Patrick Aschenbrenner Patrick.Aschenbrenner@student.uibk.ac.at

Bewertung

Vorlesung

- Schriftliche Klausur (Theorie), falls nötig online
- 1. Klausur 02.02.2022, 09:00–11:00 Uhr
- **Anmeldung** bis zum 18.01.2022 möglich.
- Zwei weitere Termine,
voraussichtlich Anfang April und Mitte September

Proseminar

- Bewertung der Einsendeaufgaben
- Wöchentliche, kurze Online-Tests
- Weitere Details im Proseminar

Lesen Sie auch die FAQ zu Prüfungen auf unserer Webseite.

Vorlesungsmaterialien

Folien

- Die Folien und Videoeinheiten stehen nach der Vorlesung im Internet zur Verfügung.
- Verwenden Sie die Seitenzahlen als Referenz für Ihre Mitschrift.
- Die kompletten Streams bleiben ca. 10 Tage online.

Lehrbücher

- D. A. Patterson and J. L. Hennessy, Computer Organization and Design, 5. Auflage, Morgan Kaufmann, 2014. (4. Auflage auch auf Deutsch erschienen)
- D. W. Hoffmann, Grundlagen der Technischen Informatik, 3. Auflage, Hanser, 2013.
- W. Oberschelp und G. Vossen, Rechneraufbau und Rechnerstrukturen, 10. Auflage, Oldenbourg, 2006.

Danksagung

Die Folieninhalte basieren teilweise auf den Materialien meiner Vorgänger:
Univ.-Prof. Drs. Matthias Harders, Falko Dressler, Alfred Strey

Team

Professor



Univ.-Prof. Dr.-Ing.
Rainer Böhme

Sekretärin



Jenifer
Payr

Techniker



Manuel
Knoflach-Schrott

Videoschnitt



Gloria
Dzida

WissenschaftlerInnen



Dr. Svetlana
Abramova



Dr. Paulina
Pesch



Dr. Daniel
Woods



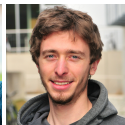
Michael
Fröwis



Maximilian
Hils



Nora
Hofer



Patrik
Keller



Alexander
Schlögl

Bemerkungen zur akademischen Ausbildung

Rolle der Universität

- Kombination von Forschung und Lehre
- Innovation = Idee + Ausführung
- **Lernziel:** Spaß an systematischem Denken
- „Rohstoff für Wachstum“

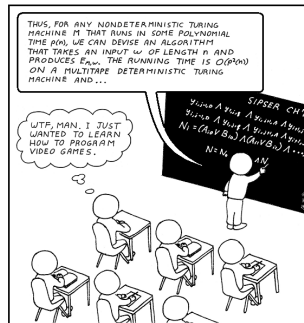
Bedeutung des „Stoffs“ in der Lehre

- Vermittlung von Prinzipien und Denkmethoden, die Innovation fördern
- Bekannte, in der Praxis verbreitete Lösungen sind Anschauungsmaterial
- Praxisrelevantes Faktenwissen als Nebeneffekt (veraltet schnell)



Bücherrad, Agostino Ramelli 1588, Bild: Wikipedia

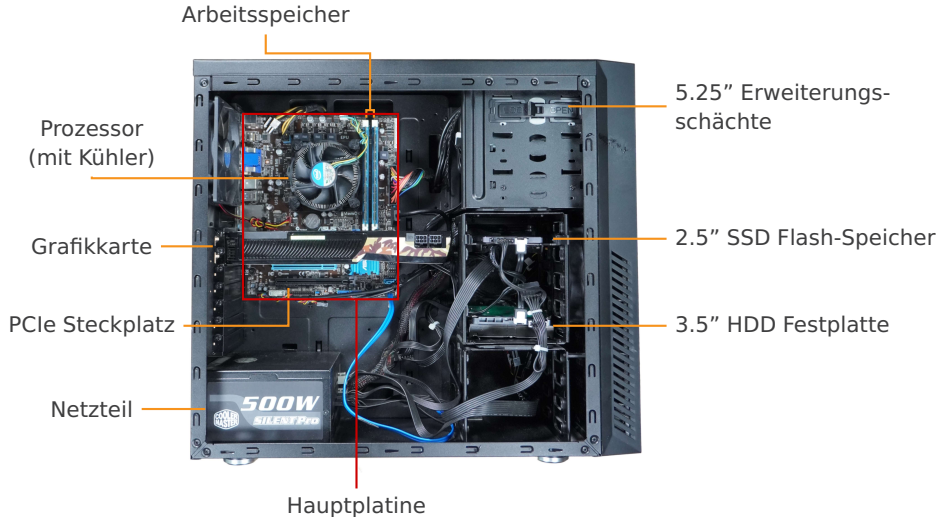
Konsequenzen für die Lehre im Bachelorstudium



Voraussetzungen

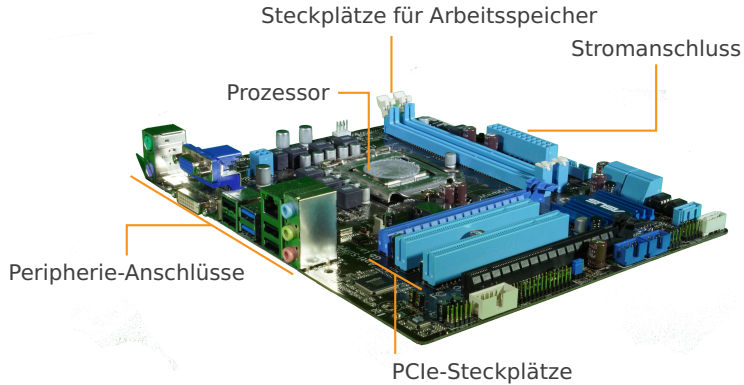
- **Grundlagen** des Fachs
- **Austausch:** Diskurs, Gastvorträge, Auslandssemester (ja, es geht wieder!), etc.
- Fachspezifische **Kommunikation:** Terminologie und Konventionen, Fremdsprachen, wissenschaftliche Literatur („Papers“)
- Fachspezifische **Fertigkeiten**, z. B. Programmieren
- **Ethik:** Standards guter wissenschaftlicher Praxis (z. B. Reproduzierbarkeit)
→ Fehler passieren trotzdem

Innenansicht eines Desktop-PCs



Hauptplatine

(engl. Motherboard)



Populäre Einplatinenrechner



Arduino

8-Bit-Atmel AVR CPU, 16 MHz
32 KB Flash-Speicher, USB
ca. 15 Euro



Raspberry Pi

32-Bit-ARM Cortex-A CPU, 1.2 GHz
1 GB Arbeitsspeicher, HDMI, Ethernet
ca. 35 Euro



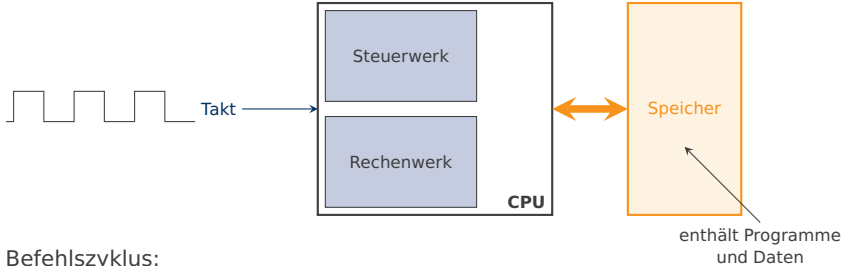
2 €

Micro Bit

32-Bit-ARM Cortex-M0 CPU, 16 MHz
256 KB Flash-Speicher, USB, Bluetooth
Sensoren, LED-Matrix, Tasten
ca. 15 Euro

Prozessor

(engl. Central Processing Unit, CPU)



In jedem Befehlszyklus:

- Holen einer Instruktion aus dem Speicher
- Dekodieren im Steuerwerk
- Austausch benötigter Daten (= Operanden) mit dem Speicher
- Ausführen im Rechenwerk

Die CPU ist die Hauptkomponente jedes Rechners.

Beispiel

Programm in der Programmiersprache C

```
1 #include <stdio.h>
2
3 int main(void)
4 {
5     printf("Was ist Software?\n");
6
7     return 0;
8 }
```

Beispiel

Maschinennahes Assembler-Programm (x86-Befehlssatz, erzeugt vom Compiler)

```
text:00001F44      call    _exit
text:00001F49      hlt
text:00001F49  start      endp
text:00001F49
text:00001F4A      ; ===== SUBROUTINE =====
text:00001F4A      ;
text:00001F4A      ; Attributes: bp-based frame
text:00001F4A
text:00001F4A      public _main
text:00001F4A      proc near          ; CODE XREF: start+30↑p
main:              push    ebp
text:00001F4A      mov     ebp, esp
text:00001F4B      push    ebx
text:00001F4D      sub     esp, 14h
text:00001F4E      call    $+5
text:00001F51      pop     ebx
text:00001F56      lea     eax, (aWasIstSoftware - 1F56h)[ebx] ; "Was ist Software?"
text:00001F57      mov     [esp], eax      ; char *
text:00001F5D      call    _puts
text:00001F60      mov     eax, 0
text:00001F65      add     esp, 14h
text:00001F6A      pop     ebx
text:00001F6D      leave
text:00001F6E      retn
text:00001F6F      main      endp
text:00001F6F
text:00001F6F      __text   ends
text:00001F6F
cstring:00001F70 ; =====
cstring:00001F70
cstring:00001F70 ; Segment type: Pure data
cstring:00001F70      __cstring segment byte public 'DATA' use32
cstring:00001F70      assume cs: __cstring
cstring:00001F70      ;org 1F70h
cstring:00001F70      aWasIstSoftware db 'Was ist Software?',0 ; DATA XREF: _main+D↑o
cstring:00001F70      __cstring ends
cstring:00001F70
symbol_stub:00001F82 ; =====
symbol_stub:00001F82
symbol_stub:00001F82 ; Segment type: Pure code
symbol_stub:00001F82      __symbol_stub segment word public 'CODE' use32
symbol_stub:00001F82      assume cs: __symbol_stub
symbol_stub:00001F82      ;org 1F82h
```

Beispiel

Darstellung als Speicherauszug (Hexadezimaldarstellung der Byte-Werte)

```
00001F0A 00 00 6A 00 89 E5 83 E4 F0 83 EC 10 8B 5D 04 89 ..j.....].  
00001F1A 1C 24 8D 4D 08 89 4C 24 04 83 C3 01 C1 E3 02 01 $.M.L$.  
00001F2A CB 89 5C 24 08 8B 03 83 C3 04 85 C0 75 F7 89 5C .\$......u.\  
00001F3A 24 0C E8 09 00 00 89 04 24 E8 39 00 00 00 F4 $.9....  
00001F4A 55 89 E5 53 83 EC 14 E8 00 00 00 00 5B 8D 83 1A U.S.....[...  
00001F5A 00 00 00 89 04 24 E8 23 00 00 00 B8 00 00 00 00 ..$.#.....  
00001F6A 83 C4 14 5B C9 C3 57 61 73 20 69 73 74 20 53 6F ..[.Was ist So  
00001F7A 66 74 77 61 72 65 3F 00 FF 25 1C 20 00 00 FF 25 ftware?...%. ...%  
00001F8A 20 20 00 00 68 18 20 00 00 FF 25 14 20 00 00 90 ..h....%. ...  
00001F9A 68 0C 00 00 00 E9 EA FF FF FF 68 00 00 00 E9 h.....h.....  
00001FAA E0 FF FF FF 01 00 00 00 1C 00 00 00 00 00 00 ....  
00001FBA 00 00 1C 00 00 00 00 00 00 00 1C 00 00 00 02 00 .....  
00001FCA 00 00 00 00 00 00 34 00 00 00 34 00 00 00 F9 0F .....4...4....  
00001FDA 00 00 00 00 00 00 34 00 00 00 03 00 00 00 0C 00 .....4.....  
00001FEA 01 00 10 00 01 00 00 00 00 00 00 00 00 00 00 .....  
00002000 00 10 00 00 24 20 00 00 28 20 00 00 2C 20 00 00 ....$( ( , , ..  
00002010 30 20 00 00 E8 31 00 00 00 00 00 E0 31 00 00 00 0 ...l.....l..  
00002020 E4 31 00 00 00 00 00 00 00 00 00 00 00 00 00 .l.....  
00002030 00 00 00 00 .....  
00003000 11 40 64 79 6C 64 5F 73 74 75 62 5F 62 69 6E 64 .@dyld_stub bind  
00003010 65 72 00 51 72 14 90 00 72 1C 11 40 5F 65 78 69 er.Qr...r..@_exi  
00003020 74 00 90 00 72 20 11 40 5F 70 75 74 73 00 90 00 t...r ._ puts...  
00003030 00 02 5F 00 0C 73 74 61 72 74 00 4B 00 04 5F 00 ...start.K...  
00003040 27 6D 61 69 6E 00 50 4E 58 41 72 67 00 55 65 6E 'main.PNXArg.Uen  
00003050 76 69 72 6F 6E 00 67 00 02 6D 68 5F 65 78 65 63 viron.g._mh exec  
00003060 75 74 65 5F 68 65 61 64 65 72 00 47 5F 70 72 6F ute_header.G pro  
00003070 67 6E 61 6D 65 00 6C 02 00 00 00 03 00 8C 1E 00 gname.l.....  
00003080 03 00 CA 1E 00 00 02 63 00 5D 76 00 62 03 00 A4 .....c.]v.b...  
00003090 20 00 03 00 A8 20 00 03 00 AC 20 00 03 00 B0 20 .....  
000030A0 00 00 00 00 02 00 00 00 1E 04 00 00 8E 1F 00 00 .....  
000030B0 10 00 00 00 0E 06 00 00 00 20 00 00 17 00 00 00 .....  
000030C0 0F 09 00 00 24 20 00 00 1F 00 00 00 0F 09 00 00 ....$  
000030D0 28 20 00 00 27 00 00 00 0F 09 00 00 30 20 00 00 ( ..'.....0 ..  
000030E0 33 00 00 00 03 00 10 00 00 10 00 00 47 00 00 00 3.....G...  
000030F0 0F 09 00 00 2C 20 00 00 50 00 00 00 0F 01 00 00 ...., ..P.....  
00003100 4A 1F 00 00 56 00 00 00 0F 01 00 00 0C 1F 00 00 J...V.....  
00003110 5C 00 00 00 01 00 01 01 00 00 00 00 62 00 00 00 \.....b...  
00003120 01 00 01 01 00 00 00 00 68 00 00 00 01 00 00 01 .....h.....  
00003130 00 00 00 00 09 00 00 00 0A 00 00 00 00 00 00 40 .....  
00003140 00 00 00 40 09 00 00 00 0A 00 00 00 20 00 20 73 ...@..... s  
00003150 74 75 62 20 68 65 6C 70 65 72 73 00 5F 70 76 61 tub helpers. pva  
00003160 72 73 00 5F 4E 58 41 72 67 63 00 5F 4E 58 41 72 rs._NXArgc. NXAr  
00003170 67 76 00 5F 5F 5F 70 72 6F 67 6E 61 6D 65 00 5F gv._progname.  
00003180 5F 6D 68 5F 65 78 65 63 75 74 65 5F 68 65 61 64 _mh_execute_head
```

Zahlendarstellung



Basis: Anzahl der verwendeten Ziffern in einem Stellenwertsystem

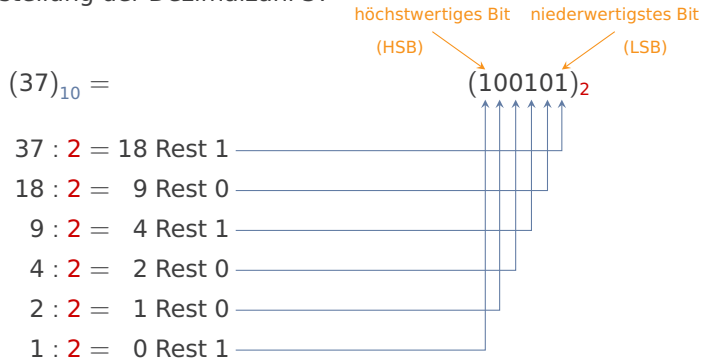
In Digitalschaltungen bietet sich die Basis **2** an: $\{0, 1\}$

$$(101101)_2 = (45)_{10} \leftarrow \text{Basis 10 i. d. R. weggelassen}$$
$$1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 4 \cdot 10^1 + 5 \cdot 10^0$$
$$1 \cdot 32 + 0 \cdot 16 + 1 \cdot 8 + 1 \cdot 4 + 0 \cdot 2 + 1 \cdot 1 = 4 \cdot 10 + 5 \cdot 1$$

Die Zahlendarstellung zur Basis **2** nennt man **binär**.

Umrechnung von dezimal nach binär

Gesucht: Binärdarstellung der Dezimalzahl 37



Die niederwertigste Ziffer (Bit) gibt an, ob die Zahl gerade (0) oder ungerade (1) ist.

Hexadezimalzahlen

Stellenwertsystem zur Basis 16: $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F\}$

Grund: 8 Bit = 1 Byte, $2^4 = 16 \rightarrow$ Darstellung eines Bytes in zwei Ziffern

$$\underbrace{(1101)}_d \underbrace{0110}_6 \bigg)_{16}$$

Umrechnung hexadezimal $\rightarrow$ dezimal: $(d6)_{16} = 13 \cdot 16^1 + 6 \cdot 16^0 = (214)_{10}$

Umrechnung dezimal $\rightarrow$ hexadezimal: Wiederholt durch 16 teilen und Reste notieren.

Konventionen

Verwechslungsgefahr, falls Hexadezimalzahlen keine Ziffern A–F enthalten.

Kennzeichnung durch Präfix 0x oder Suffix h. Bsp.: $0x0400 = 400h = 1024$

Hörsaalfragen

Wir verwenden das ARSnova Audience Response System.

- Benutzen Sie Ihr Smartphone.
- Die URL lautet: `https://arsnova.uibk.ac.at`
- Der Zugangsschlüssel lautet: **24 82 94 16**
- Oder scannen Sie den QR-Kode.



Fragen

1. Welche Ziffernfolge entspricht der Dezimalzahl 13 in Binärdarstellung?
2. Welche Zeichenfolge entspricht der Dezimalzahl 13 im Hexadezimalsystem?

Beispiel einer historischen CPU

MOS 6502 (1975)

- Apple II
- Atari 800 XL
- Commodore C64 (als 6510)

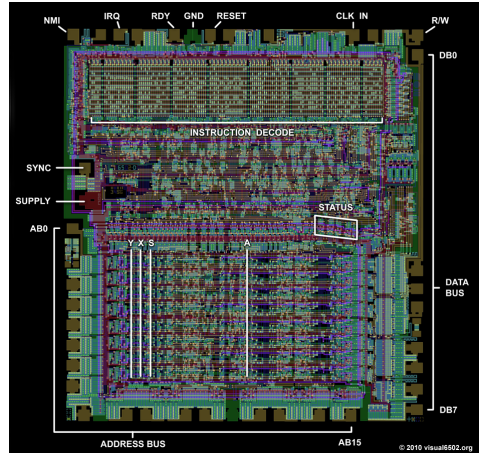
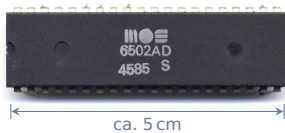
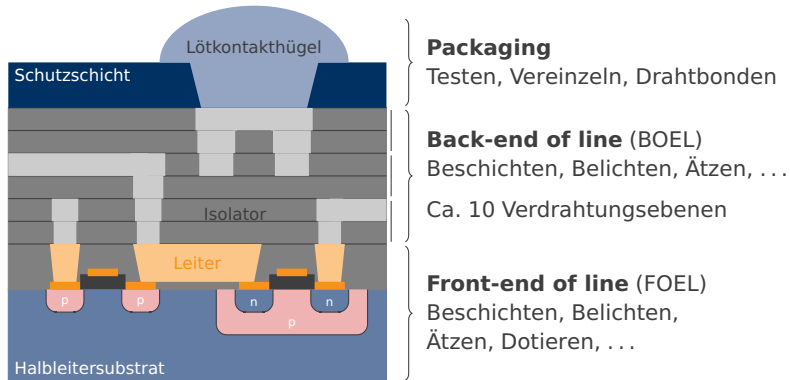


Bild: visual6502.org

Aufbau von Mikrochips

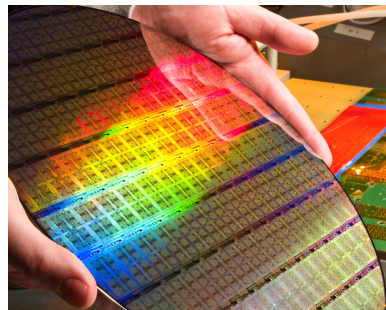
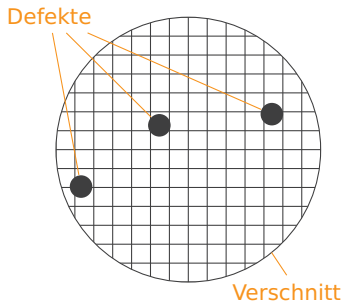
Schnitt durch einen "Die" (engl.), Herstellung im Planarprozess



Moderne Produktionsprozesse haben bis zu 500 Schritte.

Wafer

(engl. für „Waffel“ oder „Oblate“)



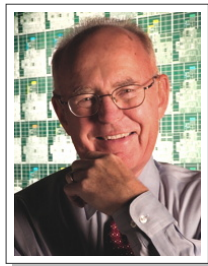
- Wafer-Durchmesser steigt: 50 mm (1970) – 300 mm (2002)
- Strukturgröße sinkt: 10 μm (1971) – 5 nm (2020)
- Die-Fläche steigt: 12 mm² (1971) – 815 mm² (2017)

Bildquelle: IBM (300 mm-Wafer)

Gordon Moore

“The complexity for minimum component costs has increased at a rate of roughly a factor of two per year. Certainly over the short term this rate can be expected to continue, if not to increase.”

Electronics Magazine, 1965

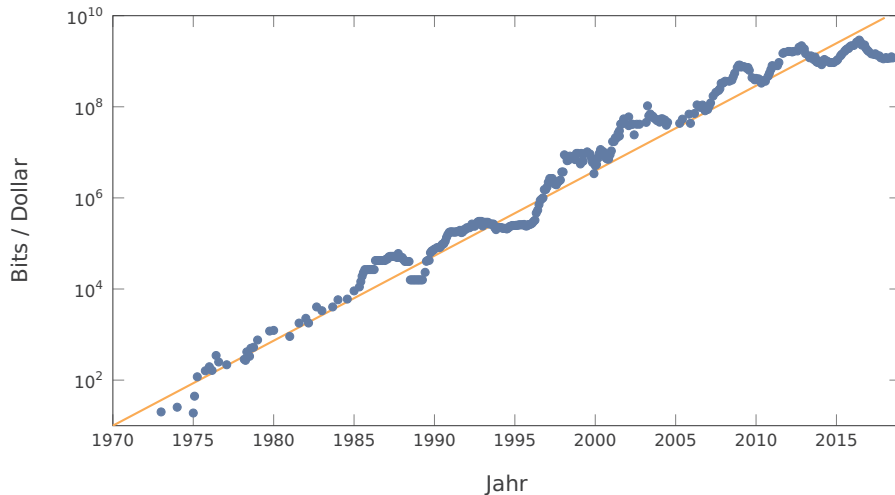


2005

Gordon E. Moore, geb. 1929, Mitgründer von Intel
Urheber des Moore'schen Gesetzes

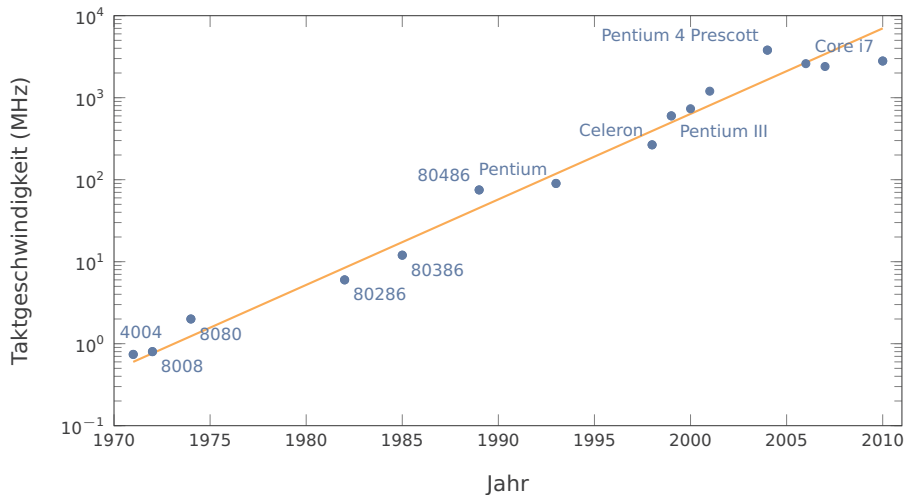
Bild: Intel

Entwicklung der Größe des Arbeitsspeichers



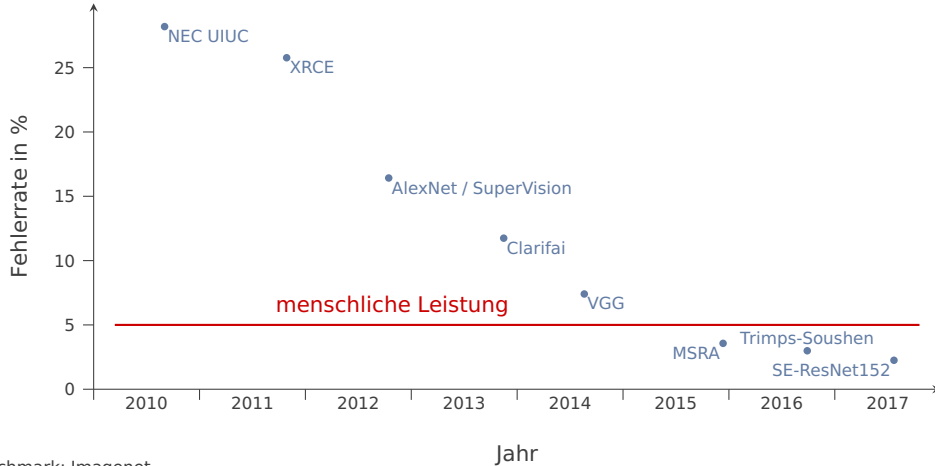
Datenquelle: John C. McCallum 2018, eigene Darstellung

Taktgeschwindigkeit von Intel-Prozessoren



Datenquelle: Wikipedia 2016

Fortschritt in der Bilderkennung



Benchmark: Imagenet

Datenquelle: Electronic Frontier Foundation 2018

Syllabus – Wintersemester 2021/22

06.10.21	1. Einführung
13.10.21	2. Kombinatorische Logik I
20.10.21	3. Kombinatorische Logik II
27.10.21	4. Sequenzielle Logik I
03.11.21	5. Sequenzielle Logik II
10.11.21	6. Arithmetik I
17.11.21	7. Arithmetik II
24.11.21	8. Befehlssatzarchitektur (ARM) I
01.12.21	9. Befehlssatzarchitektur (ARM) II
15.12.21	10. Ein-/Ausgabe
12.01.22	11. Prozessorarchitekturen
19.01.22	12. Speicher
26.01.22	13. Leistung
02.02.22	Klausur (1. Termin)