HOCHSCHULE MAINZ
Wirtschaft
Business

Grundlagen der Informatik
Hardware und hardwarenahe Programmierung

University of
Applied Sciences

Agenda

1. Typische Komponenten eines Rechners
2. CPU und Assembler
3. CPU-Performance und Moores Law

Agenda

1. **Typische Komponenten eines Rechners**
2. CPU und Assembler
3. CPU-Performance und Moores Law

Hardware

Unter dem Begriff Hardware werden alle *physischen* Komponenten (elektronische und mechanische Bestandteile) eines datenverarbeitenden Systems zusammengefasst:

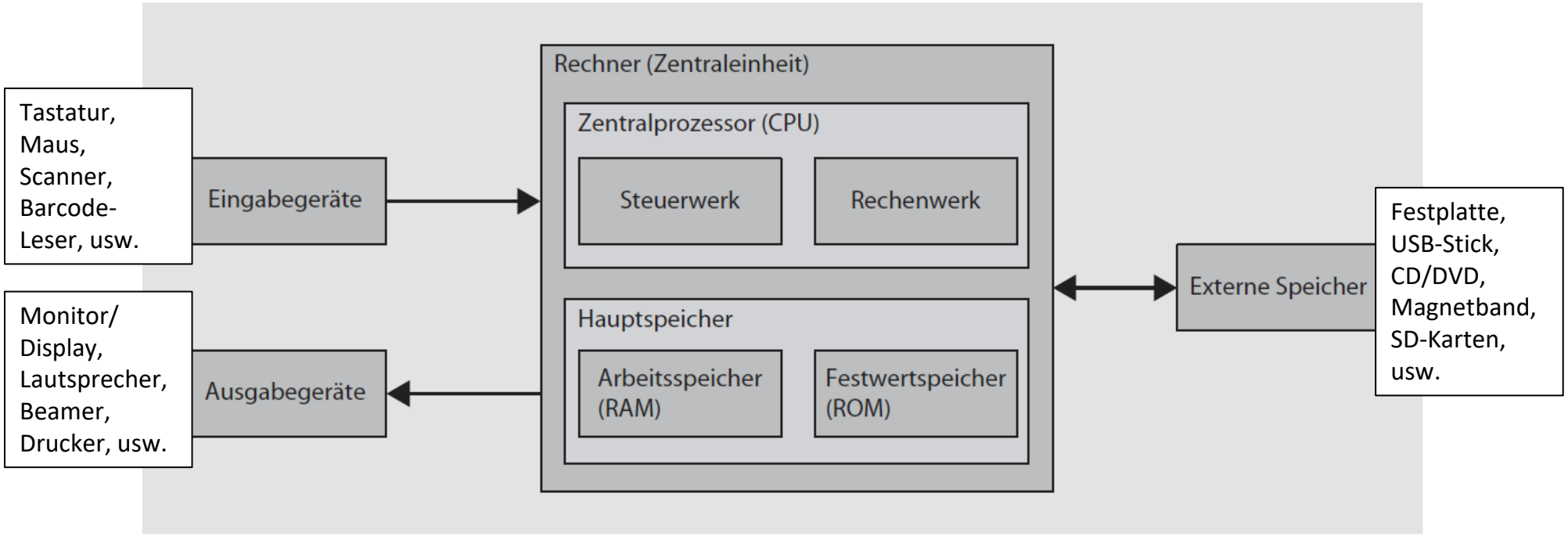
– Endnutzersysteme, z. B. Desktop-Computer, Tablets, Smartphones, …
– Backendsysteme, z. B. Server, Storage, Großrechner (Mainframe), Supercomputer, …
– Netzwerkkomponenten, z. B. Router, Switches, …

Hardware-Komponenten eines Computers

Hardware-Komponenten eines Computers können sein:

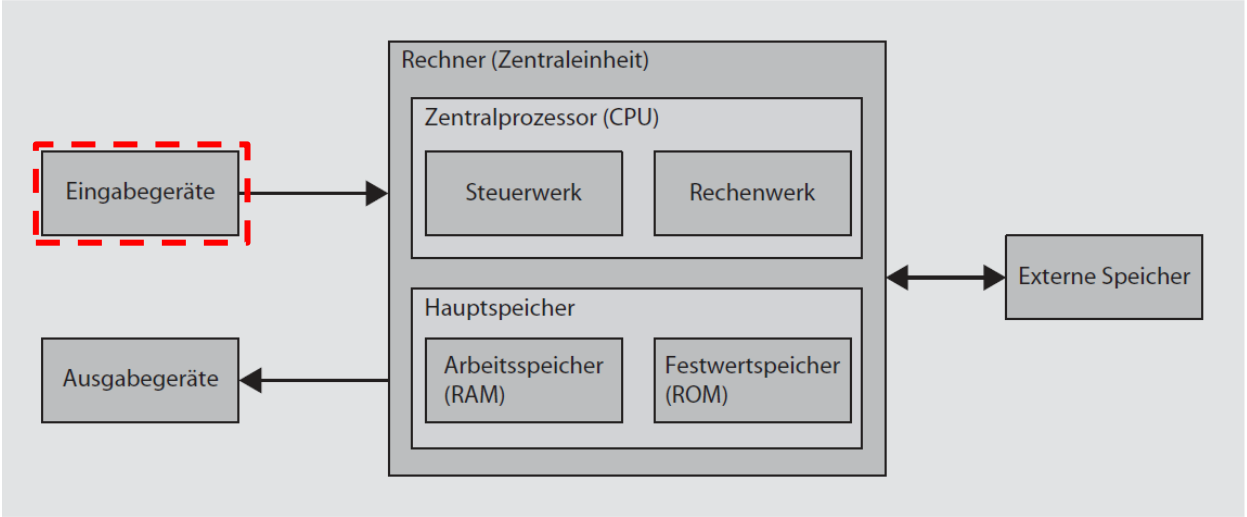
- Hauptplatine (Motherboard bzw. Mainboard)
- Prozessor (Central Processing Unit, CPU)
- Arbeitsspeicher (Random Access Memory, RAM)
- Grafikkarte
- Netzwerkschnittstelle
- Eingabegeräte (z. B. Tastatur, Maus, Gamepad, Scanner, Mikrofon)
- Ausgabegeräte (z. B. Bildschirm, Beamer, Drucker, Lautsprecher)
- optische Laufwerke (z. B. Blu-ray- oder DVD-Laufwerk)
- interne und externe Speichermedien (z. B. Festplatte, USB-Stick)

Hardware-Komponenten eines Computers



Eingabegeräte

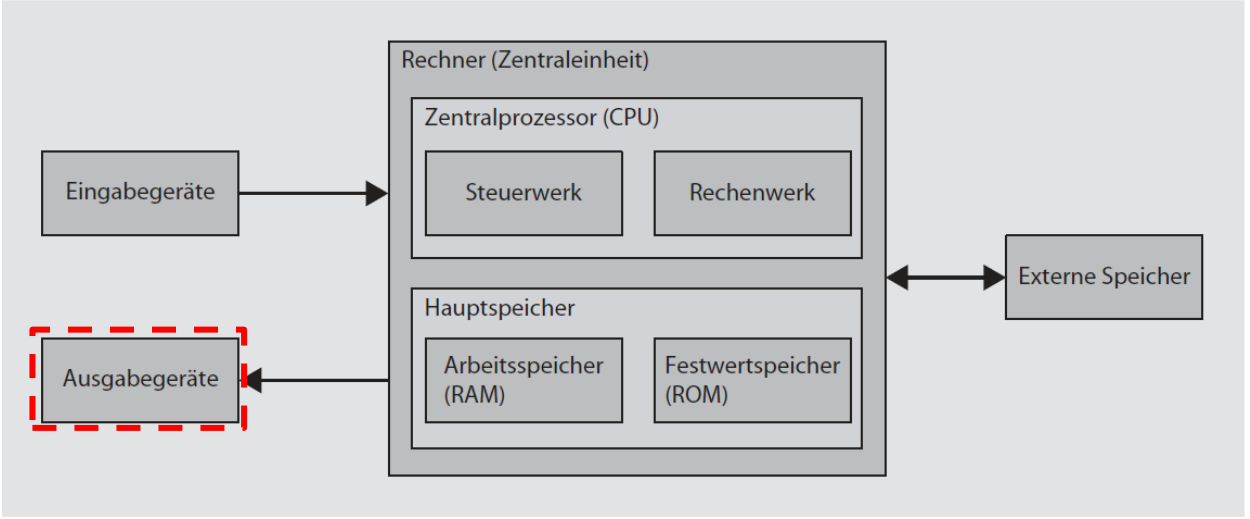
- Maschinelles Einlesen von Urbelegen (z. B. Papierbelege, Verpackungen) oder Daten von Plastikkarten (z. B. Ausweis-, Kunden-, Krankenversicherungs-, Kreditkarten)
- Lesen von Barcodes oder QR-Codes
- Scanner, Lesegeräte oder RFID (bzw. NFC)
- Automatische Direkteingabe; Daten werden von Sensoren erfasst
- Manuelle Direkteingabe über z. B. Tastatur, Maus, Touchscreen, aber auch mobile Datenerfassung
- Akustische Direkteingabe über Mikrofon



Ausgabegeräte

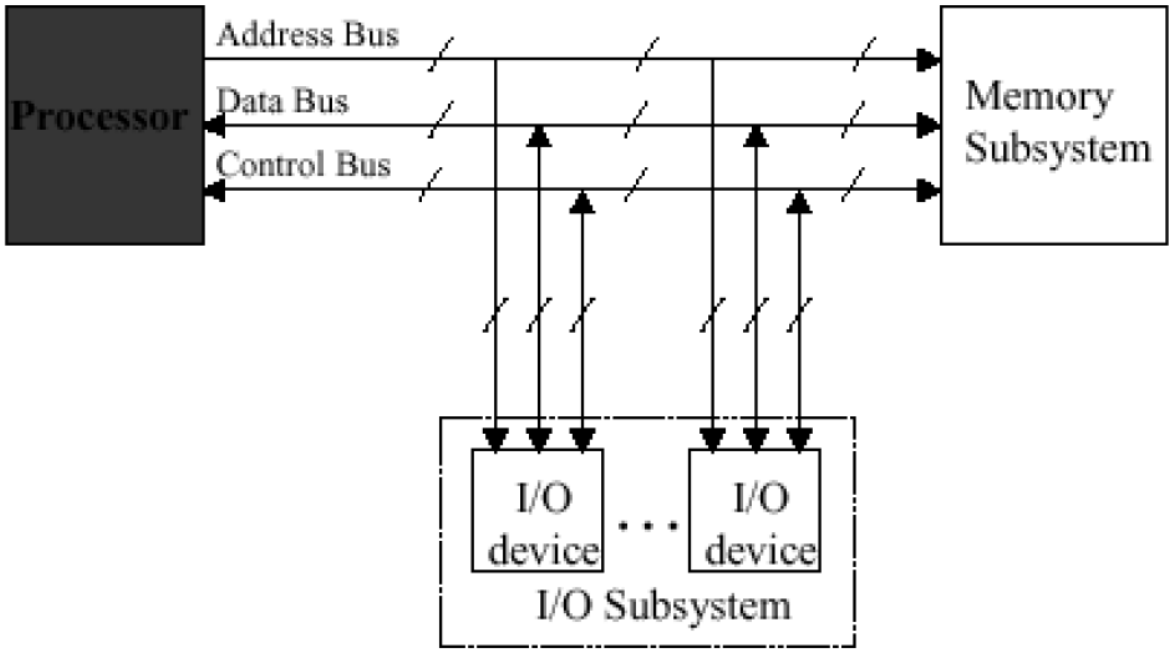
- Indirekte Datenausgabe
  - In maschinell lesbarer Form, (Zwischen-)speicherung für Archivierung, spätere Weiterverarbeitung oder spätere Datenausgabe in visuell lesbarer Form

- Direkte Datenausgabe
  - Visuelle Form über z. B. Bildschirm, Drucker oder Projektion (Beamer)
  - Akustische Datenausgabe über Lautsprecher



Adress-, Daten- und Steuerbus

- Wir unterscheiden 3 Busse:
  - Adressbus: unidirektionale Übermittlung von Adressen zum Speicher (oder zu den Ein-/ Ausgabeeinheiten) und bewirkt das Lesen oder Schreiben eines Datums über den Datenbus von bzw. zu dieser Adresse
  - Datenbus: bidirektionalen Übertragung von Daten
  - Steuerbus: Koordination exklusiver Zugriffe auf den Daten- und Adress-bus (Bus reservieren, freigeben, …)



Externe Speichertechnologien

- Magnetbänder
  - Zugriffszeit: sequenzieller Zugriff, d. h. man muss spulen
    => daher extreme Zugriffszeiten von Sekunden bis Minuten
  - Datentransferrate: akzeptabel, 400 MB/s (sequentiell)
  - Kosten pro GB: sehr günstig (<<0,01€)
  - i. d. R. nur noch zur Archivierung großer Datenmengen (mehrere 1.000 TB!) genutzt; oft bereits ersetzt durch festplattenbasierte Systeme in Form von Virtual-Tape-Libraries
- Optische Speicher
  - z. B. Blu-Ray, DVD, CD-ROM
  - Zugriffszeit: sehr langsam, ca. 80-250 ms
  - Datentransferrate: gering, ca. 8 bis 54 MB/s
  - Kosten pro GB: günstig (0,05 €)

Externe Speichertechnologien

- Magnetplatten
  - Speicherung auf magnetisierbaren Platten
  - Motorlagerung und Lese-Schreib-Mechanik der Laufwerke mit rotierenden Platten ist sehr stoßempfindlich
  - Zugriffszeit: langsam (ca. 4-9 ms)
  - Datentransferrate: gering, 30–250 MB/s
  - Kosten pro GB: günstig (0,02€/GB)
  - Verwendung als Festplatte (Hard Disk Drive, HDD)

Externe Speichertechnologien

- Elektronische Halbleiterspeicher
  - Geräuschlos, etwas kühler, deutlich robuster als HDD
  - Zugriffszeit: Direkter Zugriff und daher deutlich schneller als HDD (ca. 0,05 bis 0,2 ms)
  - Datentransferrate: ca. 500 bis 3.000 MB/s, NVMe-SSD sogar bis zu 15 GB/s
  - Kosten pro GB: zwischenzeitlich auch recht günstig (ab 0,05€/GB); Spitzenmodelle je nach Ausführung etwas teurer (0,12€/GB)
  - Heute genutzt als Festplatten für Smartphones, Tablets und PCs (Solid State Drive, SSD)
  - Zudem in USB-Speichersticks, Speicherkarten (z. B. microSD/miniSD, SD), …

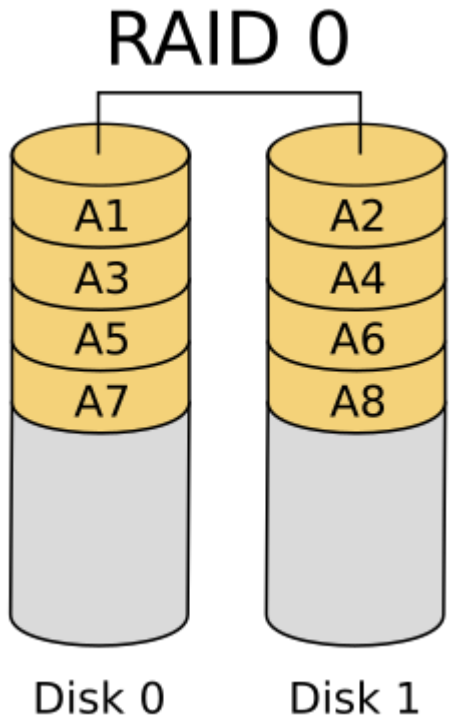
Netzwerkspeicher

- Abzugrenzen von lokalen Massenspeichern („Direct Attached Storage“) sind Netzwerkspeicher:
- NAS – Network Attached Storage:
  - Eigenständiger Mini-Server mit Festplatten, der im Netzwerk verfügbar gemacht wird
  - Zentraler Datenspeicher für viele Geräte
  - Automatische Backups, Medienstreaming, Cloud-Funktion, …
  - Netzwerkgeschwindigkeit limitiert (typisch 1 Gb/s; bis zu 25 Gb/s)
- SAN – Storage Area Network:
  - Professionelles, hochperformantes Speichernetz für Server
  - Geschwindigkeit, als wäre es ein lokales Laufwerk (z. B. über Fibre Channel oder iSCSI mit 25 bis zu 128 Gbit/s)
  - Sehr hohe Performance, Ausfallsicherheit, zentrale Verwaltung
  - sehr teuer, komplexe Einrichtung

Netzwerkspeicher

Netzwerkspeicher wie NAS- und SAN-Systeme nutzen oft RAID (Redundant Array of Independent Disks) zur Steigerung von Sicherheit oder Leistung

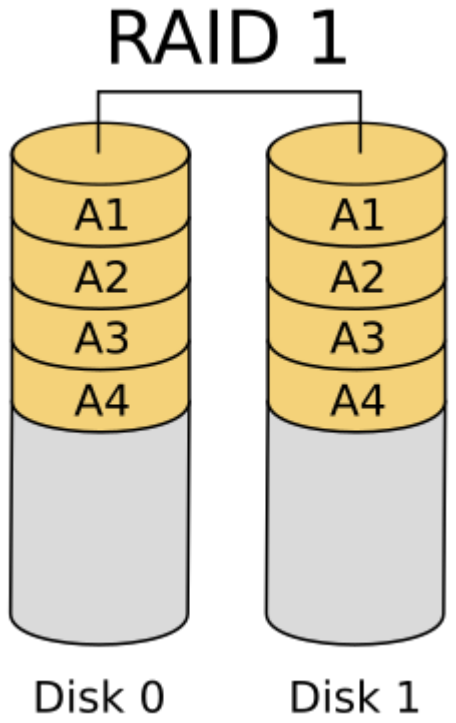
RAID-Level	Beschreibung	Vorteil	Nachteil
RAID 0	Datenstreifen über mehrere Platten	schnell ($\cdot\, n$)	kein zusätzliche Ausfallsicherheit
RAID 1	Spiegelung	hohe Ausfallsicherheit	halbe Kapazität
RAID 5	mehrere Platten+Parität	sicher + effizient	langsam beim Wiederaufbau
RAID 6	mehrere Platten+doppelte Parität	sicher bei 2 Ausfällen	aufwändig
RAID 10	Kombination aus 1 und 0	schnell + sicher	teuer (50 % nutzbar)



Netzwerkspeicher

Netzwerkspeicher wie NAS- und SAN-Systeme nutzen oft RAID (Redundant Array of Independent Disks) zur Steigerung von Sicherheit oder Leistung

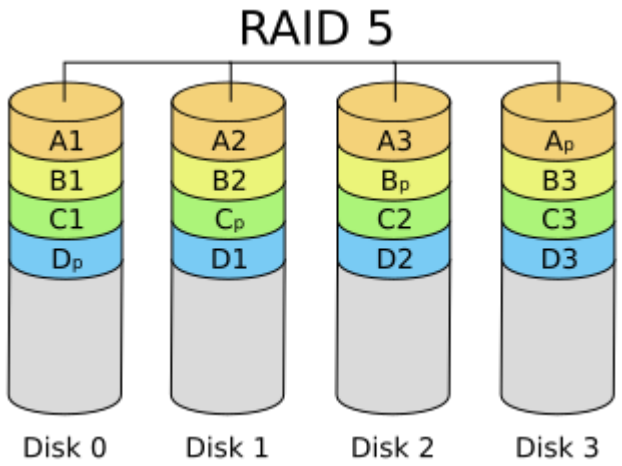
RAID-Level	Beschreibung	Vorteil	Nachteil
RAID 0	Datenstreifen über mehrere Platten	schnell ($\cdot\, n$)	kein zusätzliche Ausfallsicherheit
RAID 1	Spiegelung	hohe Ausfallsicherheit	halbe Kapazität
RAID 5	mehrere Platten+Parität	sicher + effizient	langsam beim Wiederaufbau
RAID 6	mehrere Platten+doppelte Parität	sicher bei 2 Ausfällen	aufwändig
RAID 10	Kombination aus 1 und 0	schnell + sicher	teuer (50 % nutzbar)



Netzwerkspeicher

Netzwerkspeicher wie NAS- und SAN-Systeme nutzen oft RAID (Redundant Array of Independent Disks) zur Steigerung von Sicherheit oder Leistung

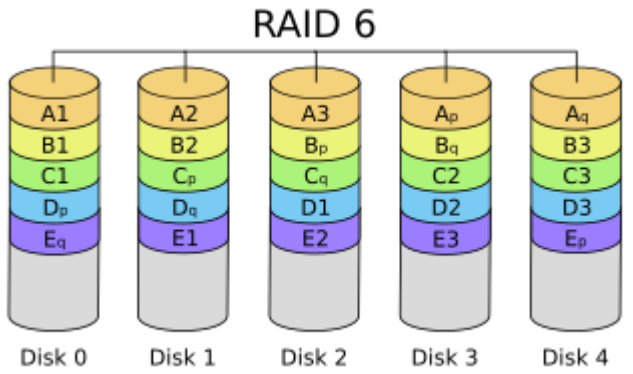
RAID-Level	Beschreibung	Vorteil	Nachteil
RAID 0	Datenstreifen über mehrere Platten	schnell ($\cdot\, n$)	kein zusätzliche Ausfallsicherheit
RAID 1	Spiegelung	hohe Ausfallsicherheit	halbe Kapazität
RAID 5	mehrere Platten+Parität	sicher + effizient	langsam beim Wiederaufbau
RAID 6	mehrere Platten+doppelte Parität	sicher bei 2 Ausfällen	aufwändig
RAID 10	Kombination aus 1 und 0	schnell + sicher	teuer (50 % nutzbar)



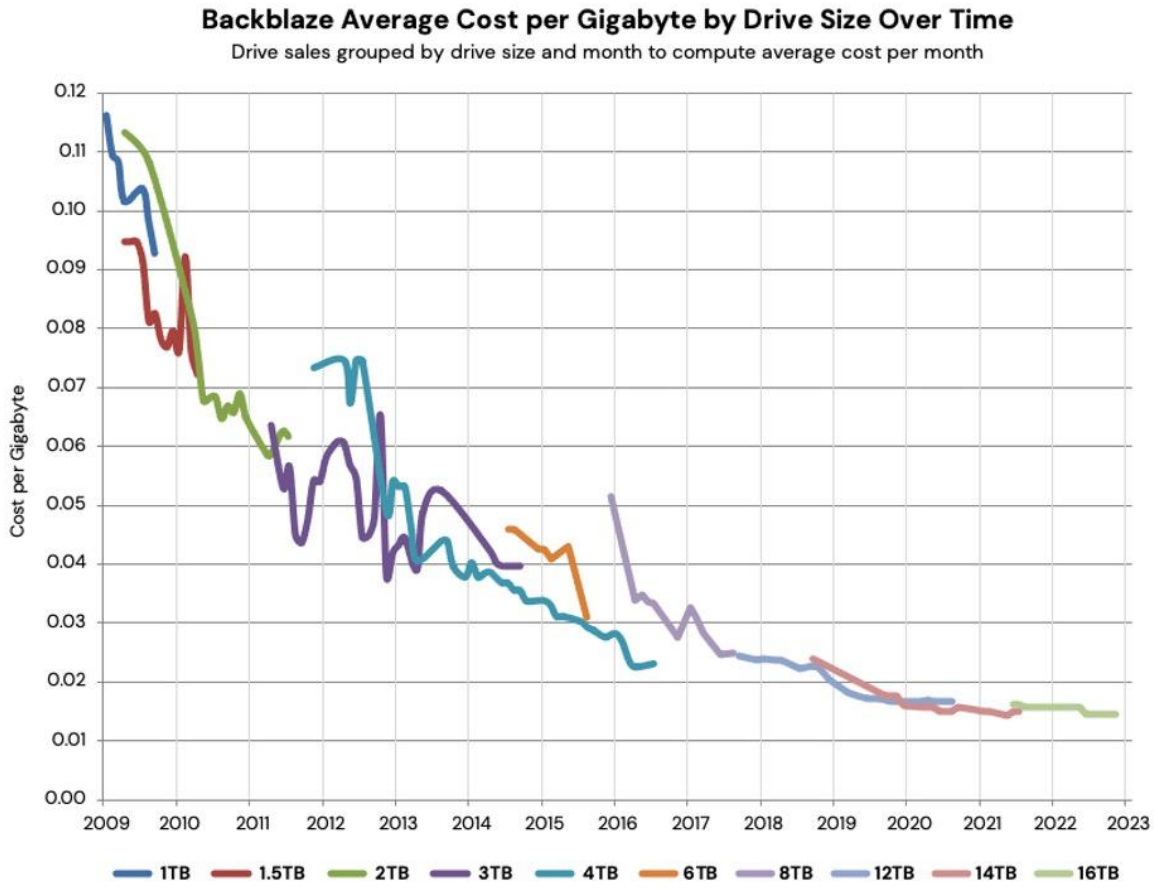
Netzwerkspeicher

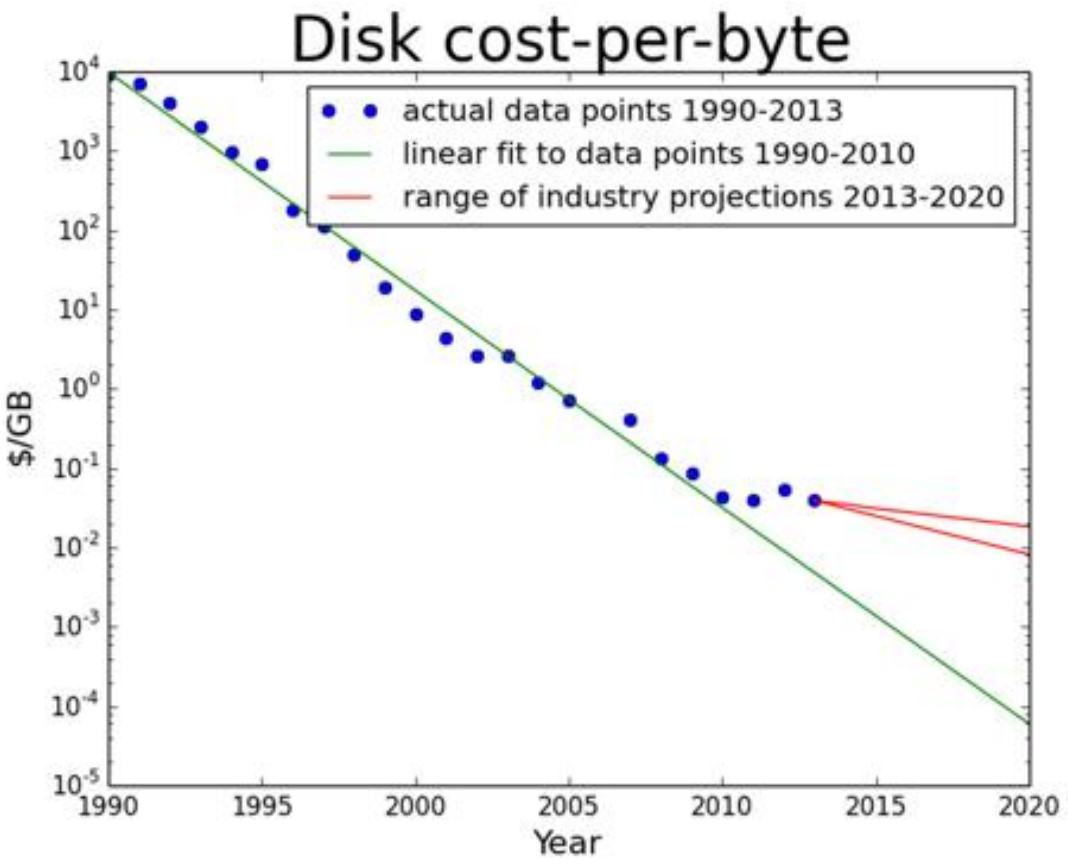
Netzwerkspeicher wie NAS- und SAN-Systeme nutzen oft RAID (Redundant Array of Independent Disks) zur Steigerung von Sicherheit oder Leistung

RAID-Level	Beschreibung	Vorteil	Nachteil
RAID 0	Datenstreifen über mehrere Platten	schnell ($\cdot\, n$)	kein zusätzliche Ausfallsicherheit
RAID 1	Spiegelung	hohe Ausfallsicherheit	halbe Kapazität
RAID 5	mehrere Platten+Parität	sicher + effizient	langsam beim Wiederaufbau
RAID 6	mehrere Platten+doppelte Parität	sicher bei 2 Ausfällen	aufwändig
RAID 10	Kombination aus 1 und 0	schnell + Ausfallsicher	teuer (50 % nutzbar)



Entwicklung der Speicherkosten bei externem Speicher

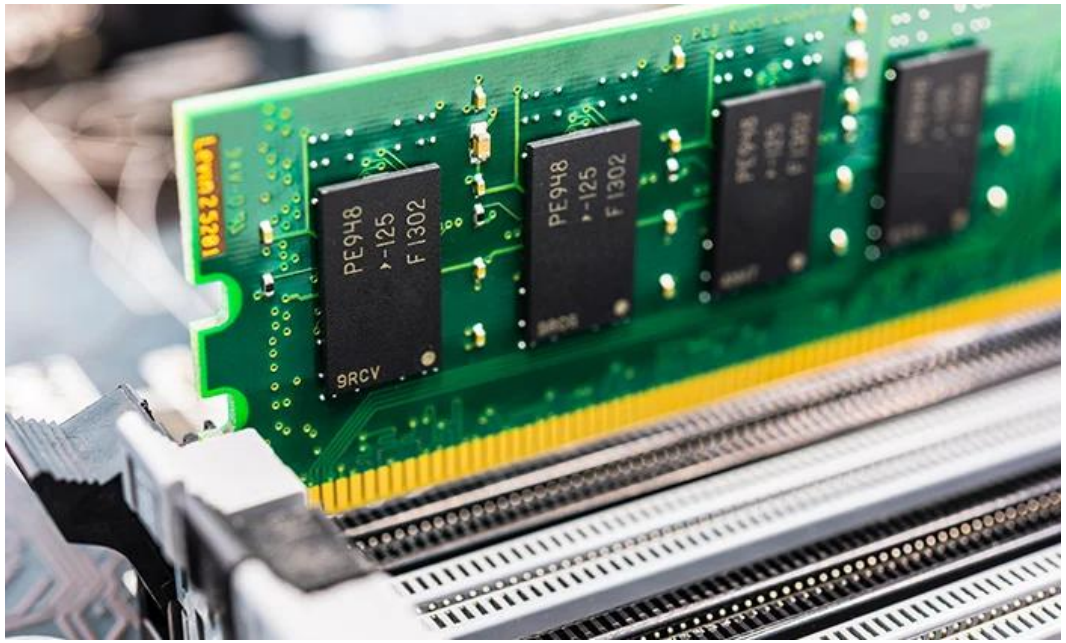


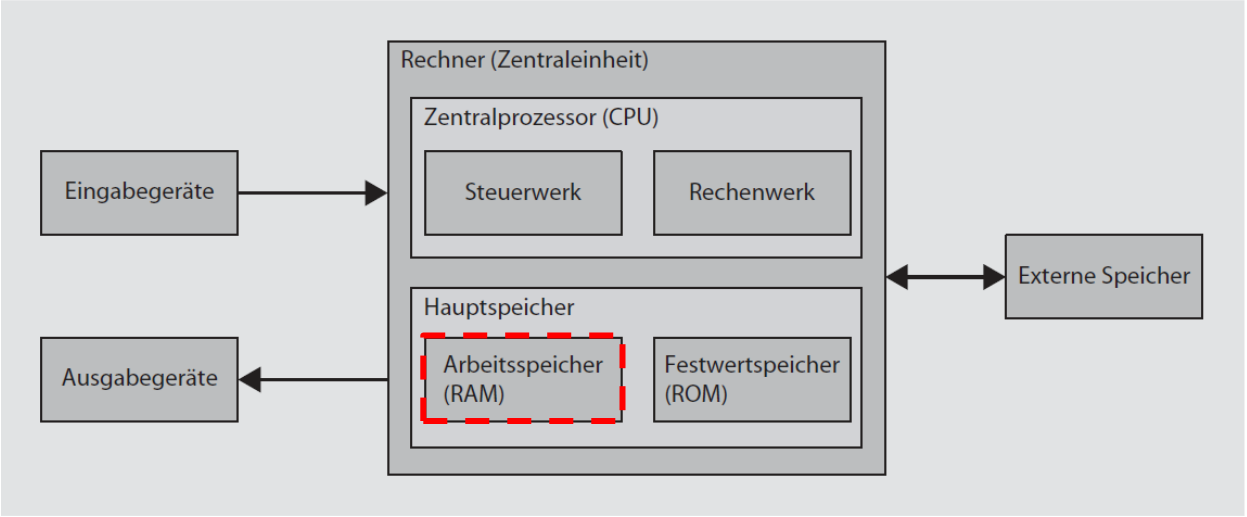


Interne Speichertechnologien

Arbeitsspeicher (Random Access Memory, RAM)

- Elektronisches Speichermedium (Transistoren und Kondensatoren)
- Nicht dauerhafter Schreib-/Lesespeicher
- Nimmt die in Aktion befindlichen Programme auf und hält die Befehle für den (Zentral-) Prozessor bereit, speichert Eingabe-, Zwischen- und Ausgabedaten
- Relevant für die Leistungsfähigkeit eines Rechners
- Zugriffszeit ca. 50-80 ns ≈ 0,00006 ms (Faktor 1.000 schneller als die schnellsten SSD-Festplatten)
- Datentransferrate: 80–90 GB/s (6x schneller als die schnellsten SSD-Festplatten)
- Kosten pro GB: teuer (2-4€/GB)

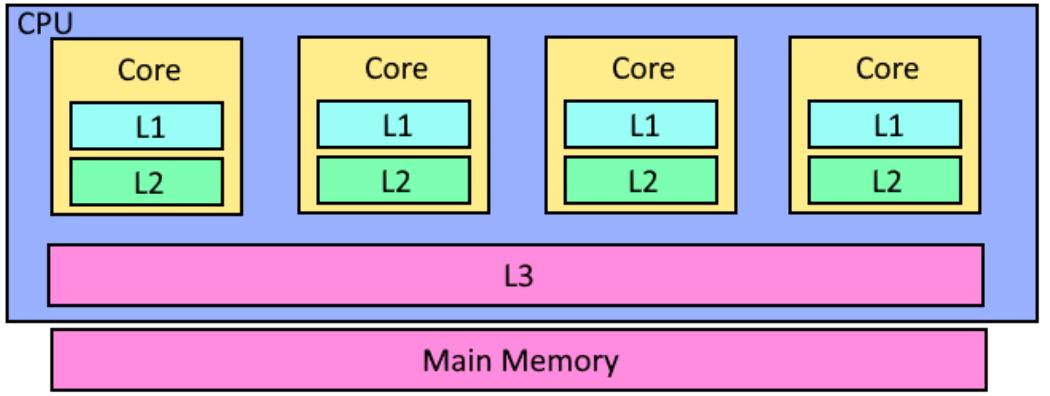




Interne Speichertechnologien

- Cache (frz. „cacher“ - verbergen)
  - kleiner Speicher mit besonders schnellem Zugriff
  - Stellt einen Puffer dar, der die Verarbeitungszeit im Prozessor und die (längere) Zugriffszeit zum Arbeitsspeicher ausgleicht
  - In Cache-Speichern werden z. B. die vom Prozessor zuletzt bearbeiteten Daten zwischengespeichert; werden diese kurzfristig nochmals benötigt, kann der Zugriff darauf nun erheblich schneller durchgeführt werden
  - Cache ist maßgeblich für die Geschwindigkeit der CPU mitverantwortlich

Interne Speichertechnologien



- Cache (frz. „cacher“ - verbergen)
  - Heutige Prozessoren haben in der Regel Multi-level Caches
    - L1 Cache ist einem Rechenkern zugeordnet
      - extrem schnell (wenige Rechentakte), speichert kürzlich genutzte Daten und Instruktionen
      - heute oft schon 1 bis 2 MB
      - Zugriff in 0,5 ns (ca. 120x schneller als RAM)
    - L2 Cache ist einem Kern, manchmal einem Cluster von Kernen zugeordnet
      - größer und etwas langsamer
      - bis zu 20 MB bei 2-4 ns (ca. 30x schneller als RAM)
    - L3 Cache ist häufig „shared“ zwischen allen Kernen
      - bis zu 128 MB bei 8-15 ns (ca. 6x schneller als RAM)

Interne Speichertechnologien

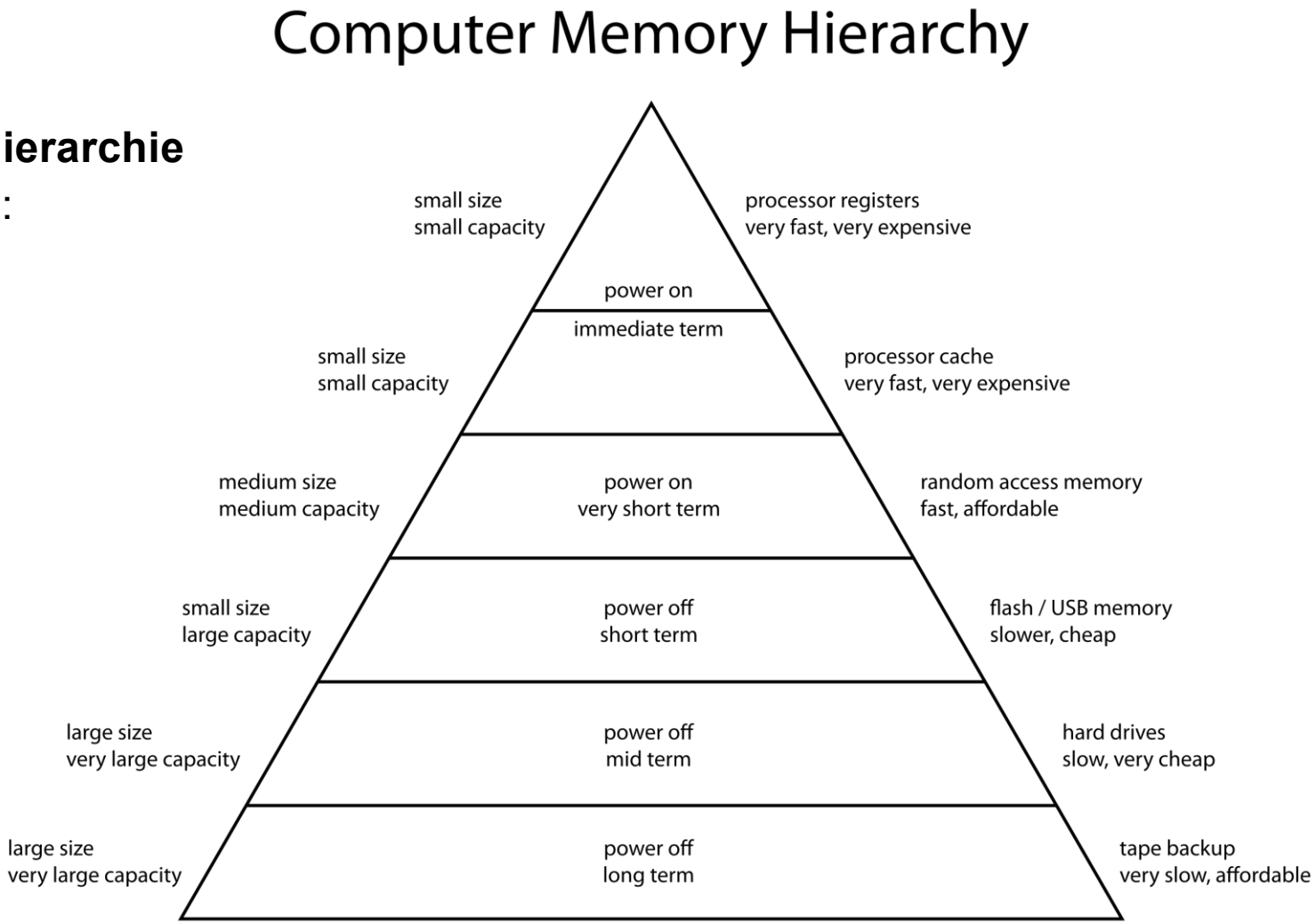
- Register
  - Kleinster und schnellster Speicher des Computers, direkt in der CPU-Pipeline
  - Direkter Zugriff möglich, keine Latenz, d.h. ~0,25 ns (≈ 1 CPU-Takt bei 4 GHz)
  - Nur wenige KB pro CPU

- Festwertspeicher (Read Only Memory, ROM)
  - Nur-Lese-Speicher, z. B. auf dem Mainboard zu finden
  - enthält Programme des Steuer- und Rechenwerks sowie unveränderliche Anwendungsprogramme sowie BIOS/UEFI/u.ä.

Speichertechnologien - Überblick

Kriterien einer **Speicherhierarchie** der Speichertechnologien:

- Größe / Kapazität
- Speicherdauer
- Kosten
- Zugriffszeit

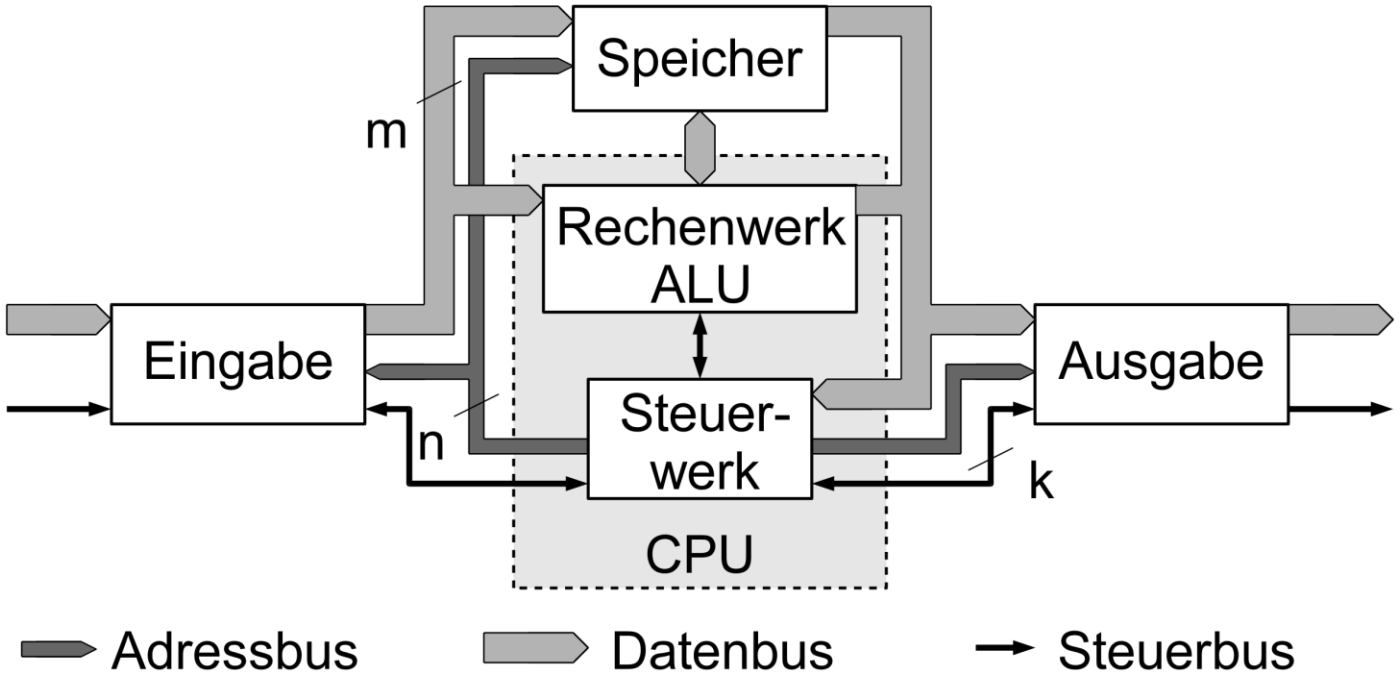


Agenda

1. Typische Komponenten eines Rechners
2. **CPU und Assembler**
3. CPU-Performance und Moores Law

Prozessor

– Die meisten Rechner verwenden die „Von-Neumann-Architektur“:



Steuerwerk / CU

- Das Steuerwerk besteht aus:
  - Befehlszähler (program counter oder instruction pointer)
    - ist ein Register, das die Speicheradresse des nächsten auszuführenden Befehls enthält
    - nach jedem Schritt wird der Zähler erhöht (bzw. bei Sprungbefehlen überschrieben)
  - Befehlsdecoder
    - zerlegt den Maschinenbefehl unter Verwendung der Befehlstabelle (instruction table) in seine Bestandteile (Operation + Operanden)
    - erkennt also welche Operation ausgeführt werden soll (z. B. Addition, Sprung, Vergleich) und welche Register oder Speicheradressen beteiligt sind
  - Befehlsregister
    - der aktuell aus dem Speicher geholte Befehl wird hier zwischengespeichert, während er dekodiert wird
  - Operationensteuerung
    - erzeugt die Steuersignale für andere Komponenten der CPU (ALU, Registerwerk, Speicher)
    - das Steuerwerk sorgt insofern für die Ausführung der Operationen (z. B. leitet den Befehl und die Operanden an das Rechenwerk weiter)

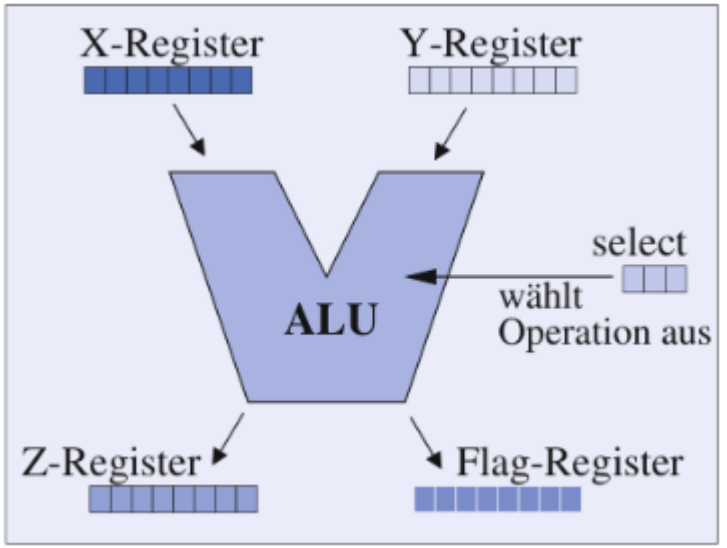
Von-Neumann-Zyklus

Das Steuerwerk eines Rechners mit klassischer Von-Neumann-Architektur folgt bei der Abarbeitung eines Maschinenprogramms dem *Von-Neumann-Zyklus:*

1. **Fetch - Hole den nächsten Befehl:**
   Das Steuerwerk liest die Adresse des nächsten Befehls aus dem Program Counter aus. Diese Adresse wird an den Hauptspeicher übermittelt, der den dort gespeicherten Maschinenbefehl an die CPU zurücksendet. Der Befehl wird im Befehlsregister zwischengespeichert.
2. **Decode - Dekodiere den Befehl:**
   Das Steuerwerk erzeugt aus dem abgelegten Maschinenbefehl die notwendigen Steuersignale (Operation und Operanden). Der Befehlszähler wird auf die Adresse des nächsten Befehls gesetzt (meist +1).
3. **Fetch Operands - Hole benötigte Operanden:**
   Lade die zur Ausführung notwendigen Daten (Operanden) aus dem Hauptspeicher in das Rechenwerk.
4. **Execute - Führe den Befehl aus:**
   Eine arithmetische oder logische Operation wird vom Rechenwerk ausgeführt und das Ergebnis wird in einem Register gespeichert. Bei Sprungbefehlen und erfüllter Sprungbedingung wird an dieser Stelle der Befehlszähler verändert.
5. **Write back - Schreibe das Ergebnis zurück:**
   Sofern notwendig, wird das Ergebnis der Berechnung in den Speicher zurückgeschrieben.

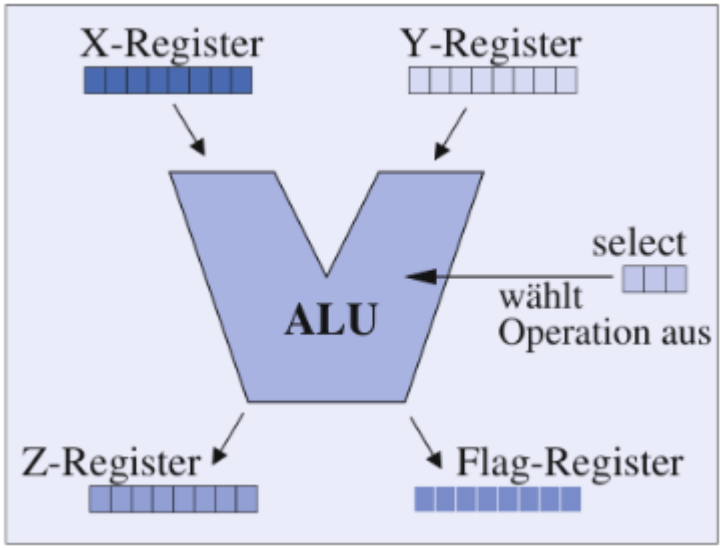
Rechenwerk / ALU

- Das Rechenwerk realisiert sowohl *arithmetische* Operationen (Addition, Subtraktion, …) als auch *logische* Operationen (AND, OR, …)
- X und Y sind Input, Z ist Output
- $select$ legt die auszuführende Operation („Opcode“) fest (z. B. Addieren)
- Da bei der Ausführung von Operationen bestimmte Zusatz-informationen (wie z. B. Übertrag, Fehler) anfallen können, existiert noch ein „*Flag*“-Register



Rechenwerk

- In der Regel werden u. a. folgende Operationen unterstützt:
  - Arithmetisch (immer für Festkommazahlen, heute i.d.R. auch Gleitkommazahlen):
    - Addition
    - Subtraktion
    - Multiplikation
    - Division
  - Logische und andere:
    - NOT, AND, OR, XOR
    - Bit-Shift, Bit-Rotation
    - Register-Manipulationen und Bit-Veränderungen (Bits setzen, löschen und testen)
    - Vergleich
    - …



Weitere Komponenten der ALU

Moderne ALUs haben weitere spezielle Komponenten, z. B.:

- Eigene Logik für Multiplikation, Division und komplexere arithmetische Operationen
- Vektorbasierte Implementierungen wenn viele gleichartige Daten auf gleiche Weise bearbeitet werden sollen
- …

RISC vs. CISC

Reduced Instruction Set Computer (RISC)

- Einfache Prozessor-Struktur mit limitiertem Befehlssatz (zwischen 20 und 100 Maschinenbefehle)
- Maschinenbefehle sind sehr schnell ausführbar (oft in einem Verarbeitungsschritt)
- Große Anzahl von Registern, von denen die meisten frei verwendbar sind
- Interpretation der Befehle direkt durch die Hardware

Complex Instruction Set Computer (CISC)

- Komplexe Prozessor-Struktur, meist mehr als 100 Maschinenbefehle
- komplexe Funktionen werden direkt durch den Prozessor durchgeführt
  => unterschiedliche und teils lange Ausführungszeiten für die einzelnen Befehle
- kleine Anzahl von Registern, von denen die meisten für vorbestimmte Aufgaben eingesetzt werden
- Teilweise Mikroprogramme als Software in der CPU implementiert

RISC vs. CISC

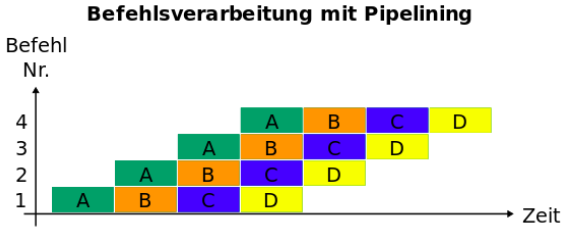
Reduced Instruction Set Computer (RISC)

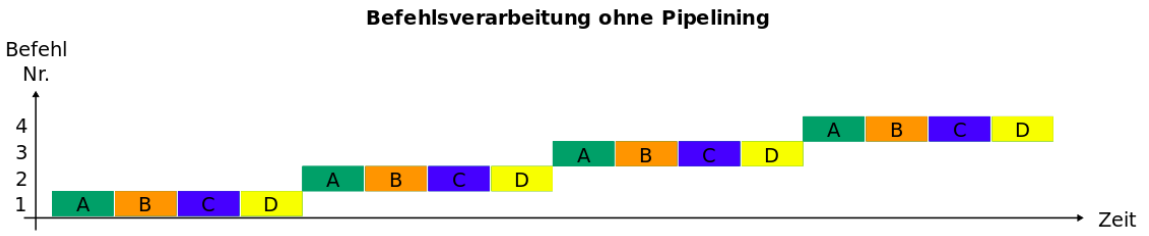
- CPU-Entwurf ist einfacher
- Einfachheit ermöglicht kleinerer Chip-Fläche und somit Platz für zusätzliche Register oder Cache-Speicher auf dem Chip
- Übersetzungsprogramme (Compiler, Interpreter) müssen die erforderlichen Schritte mit den angebotenen sehr einfachen Maschinenbefehlen nachbilden
- i.d.R. schnellere Befehlsausführung (Umsetzung allein auf Hardware, schnellere Befehls-Dekodierung da weniger Befehle, Einsatz von Pipelining)

Complex Instruction Set Computer (CISC)

- CPU-Entwurf wegen der Vielzahl einzelner Befehle wesentlich komplizierter
- Erleichtert den Bau von Compilern oder Interpretern, indem diese vorhandene „mächtige“ CISC-Befehle verwenden

Pipelining





- Bei Pipelining geht es um die parallele Abarbeitung mehrerer Befehle in einem Prozessor
- wird vor allem bei RISC-Prozessoren verwendet, weil Befehle in der Regel eine gleich lange Ausführungszeit benötigen
- Beispiel für 5-stufige Pipeline (6 Befehle in 10 Takten):

Befehl 1:	ID	IF	EX	MEM	WB				
Befehl 2:		ID	IF	EX	MEM	WB			
Befehl 3:			ID	IF	EX	MEM	WB		
Befehl 4:				ID	IF	EX	MEM	WB	
Befehl 5:					ID	IF	EX	MEM	WB
Befehl 6:						ID	IF	EX	MEM	WB
Takt:	1	2	3	4	5	6	7	8	9	10

Pipelining

Spezielle Probleme des Pipelining:

- Data-Hazards:
  - zu dichtes Aufeinanderfolgen von Befehlen
    => benötigte Zwischenergebnisse liegen nicht vor
  - Lösungsmöglichkeiten:
    Einfügen von NoOps, Forwarding (Zusatzhardware erlaubt, das Ergebnis des einen Befehls direkt in die Register zu schreiben), Umsortieren der Befehle durch den Compiler („Out-of-order execution“)
- Control Hazards:
  - Es muss zuerst die Bedingung („if“) ausgewertet werden, bevor entschieden werden kann, an welche Stelle des Programms zu springen ist
    => es ist nicht möglich sofort den nächsten Befehl zu bearbeiten, da ja noch nicht bekannt ist, was der nächste Befehl sein wird
  - Lösungsmöglichkeiten:
    Einfügen von NoOps, Branch Prediction / Sprungvorhersage (wahrscheinlichsten nächsten Befehl bearbeiten; wenn falsch: ganze Pipeline leeren und eventuell falsch gesetzte Werte zurücksetzen)

RISC vs. CISC heute

RISC

- Hauptsächlich ARM-Architektur
- Hauptvorteil: Energieeffizienz und Performance/Watt
- Anwendungen entsprechend:
  Smartphones, Tablets, IoT, Apple-Laptops, Embedded Systeme und Microcontroller
  Seit etwa 2020 auch mehr und mehr Server-Systeme (etwa 20% Marktanteil, stark wachsend)

CISC

- Hauptsächlich x86-64 („x64“)
- Kompatibilität (z. B. Windows, ältere Programme), GPU-Integration
- Anwendungen: klassischen PC- und Serverarchitekturen (Intel und AMD Server, Desktop-PCs und Gaming/Konsolen)

- Moderne CISC-CPUs verstehen immer noch die x86-/x86-64-Befehlssätze (= viele komplexe Instruktionen) dekodieren diese aber in interne Mikro-Operationen, die vom Kern wie bei einer RISC-Pipeline ausgeführt werden

Erhöhung der Wortbreite in den verschiedenen Prozessorgenerationen

- Die Wortbreite eines Prozessors legt fest, aus wie vielen Bits ein Maschinenwort eines Prozessors besteht
- Im Falle des Adressbusses legt die Wortbreite z. B. die maximale Größe von Speicheradressen fest, und somit auch automatisch, wie viel Arbeitsspeicher ein Prozessor überhaupt adressieren kann

Prozessor (Jahr)	Wortbreite	Darstellbare Werte
Intel 4004 (1971)	4 Bit	16
Intel 8080, Zilog Z80 (1974)	8 Bit	256
Intel 8088, Motorola 68000 (1979)	16 Bit	65 536
Intel 80386, PowerPC (1985)	32 Bit	4 294 967 296
Mips R4000 (1991), AMD Opteron (2003), Intel Xeon (2004)	64 Bit	18 446 744 073 709 551 616

Anzahl an Recheneinheiten (Cores)

- Moderne Prozessoren sind Multicore-Prozessoren: sie besitzen mehrere komplette Recheneinheiten, Core oder auch Kern genannt
- Jeder Core kann unabhängig voneinander ein eigenes Programm ausführen
- Es kann sich z. B. ein Core um die Betriebssystemaufgaben kümmern, während der andere die gerade aktive Anwendung abarbeitet
- Die Auslastung von Multicore-Prozessoren hängt wesentlich davon ab, inwiefern sich die aktuellen Aufgaben parallelisieren lassen

Taktfrequenz

- Die Anzahl der Rechentakte pro Sekunde (Taktfrequenz) sagt etwas darüber aus, wie viele Berechnungen ein Prozessor pro Zeiteinheit durchführen kann
- Allerdings: Auch die Implementierung der Befehle in der CPU ist entscheidend (Beispiel: wenn ein Prozessor zur Ausführung bestimmter Befehle mehrere Taktzyklen benötigt, während ein anderer Prozessor diese Befehle in einem Taktzyklus abarbeitet, oder sogar mehrere Befehle gleichzeitig verarbeiten kann)

Assembler

- Binäre Maschinenbefehle, wie sie im Befehlsregister verarbeitet werden, sind für den Menschen praktisch unlesbar
- Daher wurde zur Erleichterung der Programmierung eine symbolische Schreibweise für Maschinenbefehle eingeführt: Assemblersprache
- Jedem binären Maschinenbefehl wird ein einfacher Befehl (z. B. MOVE oder ADD) zugeordnet
- Fast jeder Prozessortyp umfasst unterschiedliche Maschinenbefehle und es gibt somit unterschiedliche Assembler-Befehle („Prozessor-Befehlssatz“)

Assembler

- Der Quelltext eines Assemblerprogramms wird mit Hilfe einer Übersetzungssoftware (Assembler oder Assemblierer) in Maschinencode übersetzt

- Assemblerbefehl: movb $0x61, %al # „move byte mit Wert x nach y“
- Maschinensprache: 10110000 01100001

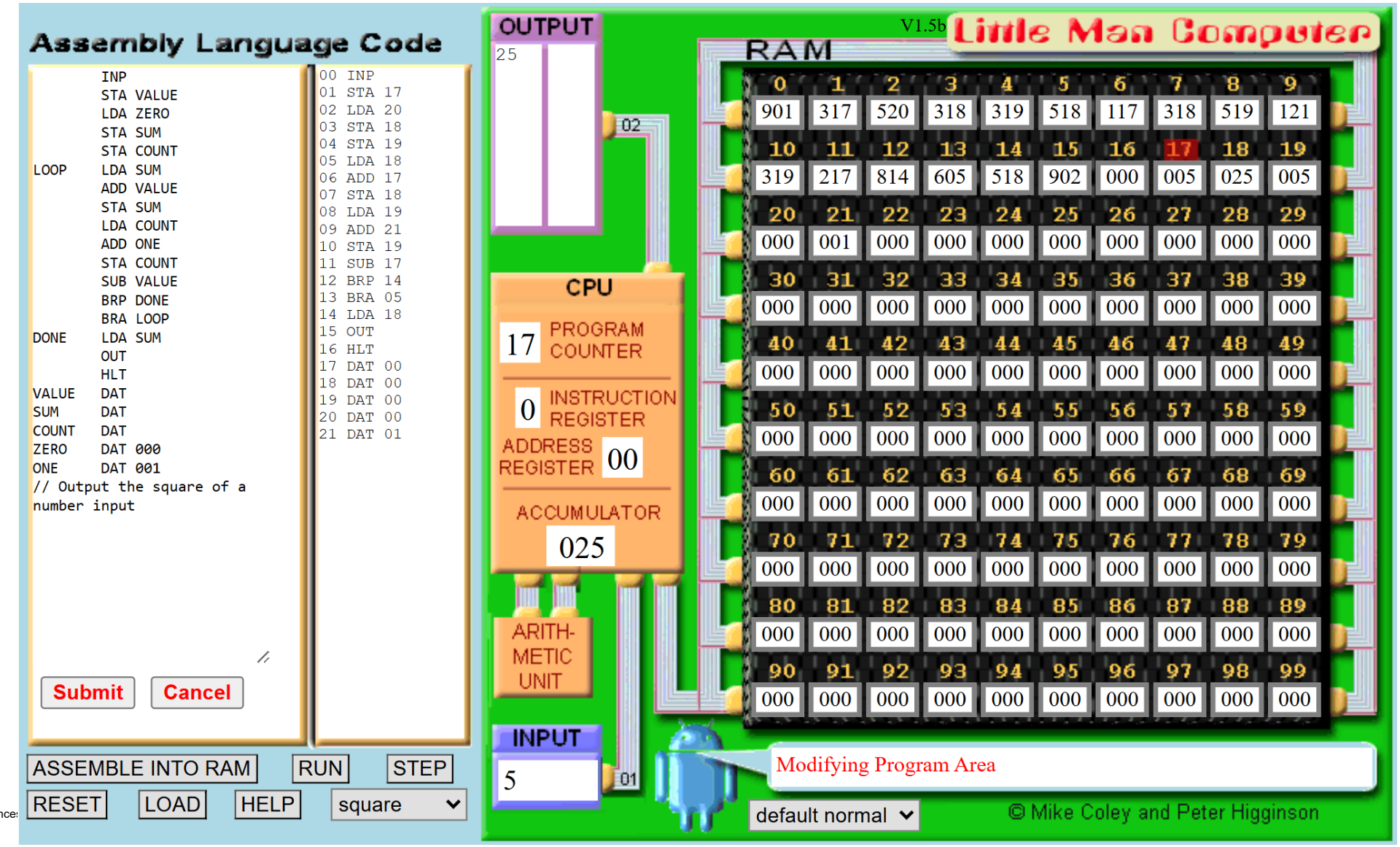
Opcode: lade einen 8Bit-Wert direkt („immediate“) in das CPU-Register „AL“

Operand: 97 (Dezimal)

Prozessor – Little Man Computer

- Der Little Man Computer (LMC) ist ein Modell zur besseren Erklärung der Von-Neumann-Architektur und des Von-Neumann-Zyklus
- stellt die grundlegenden Funktionen eines Computers verständlich dar
- im Jahr 1965 entwickelt
- kann in einer einfachen Assembler-Sprache programmiert werden

Prozessor – Little Man Computer

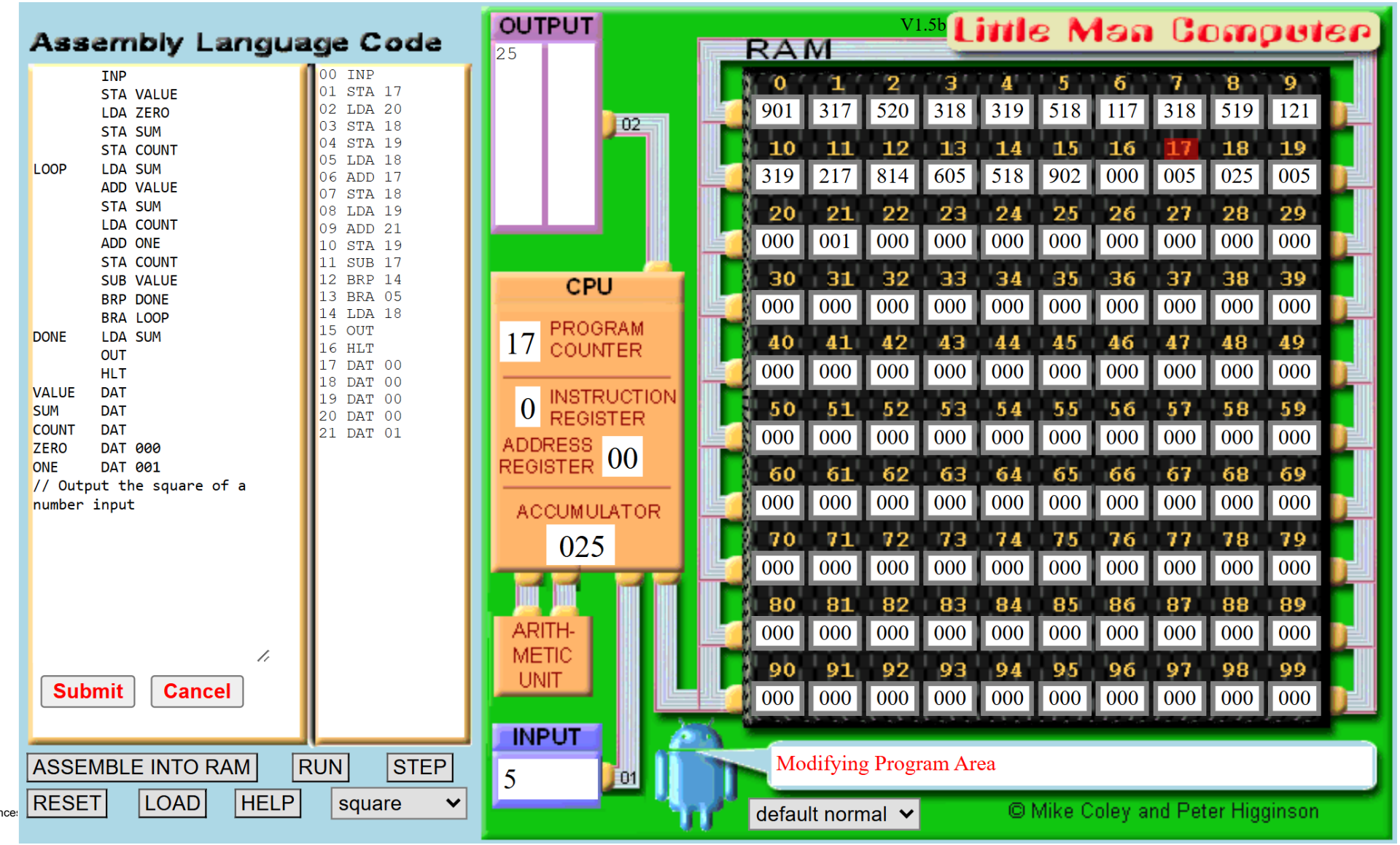


Prozessor – Little Man Computer

- Konzept eines kleinen Mannes, der in einer geschlossenen „Poststelle“ eingesperrt ist
- Empfang und Ausgabe von Daten:
  - 2 „Fächer“ mit der Bezeichnung Input und Output
- Hauptspeicher:
  - 100 „Fächer“, die von 0 bis 99 nummeriert sind
  - enthalten jeweils eine dreistellige Anweisung oder Daten (im Bereich von 000 bis 999)
- Rechenwerk:
  - zwei Funktionen - Addition und Subtraktion
- Programmzähler
  - speichert die Adresse der nächsten Anweisung, die der kleine Mann ausführen wird
  - wird normalerweise nach der Ausführung jeder Anweisung um 1 erhöht
  - Schleifen und Bedingungen möglich (Programmzähler wird auf eine nichtsequentielle Speicheradresse gesetzt wenn eine bestimmte Bedingung erfüllt ist)

Prozessor – Little Man Computer

https://peterhigginson.co.uk/LMC/



Agenda

1. Typische Komponenten eines Rechners
2. CPU und Assembler
3. **CPU-Performance und Moores Law**

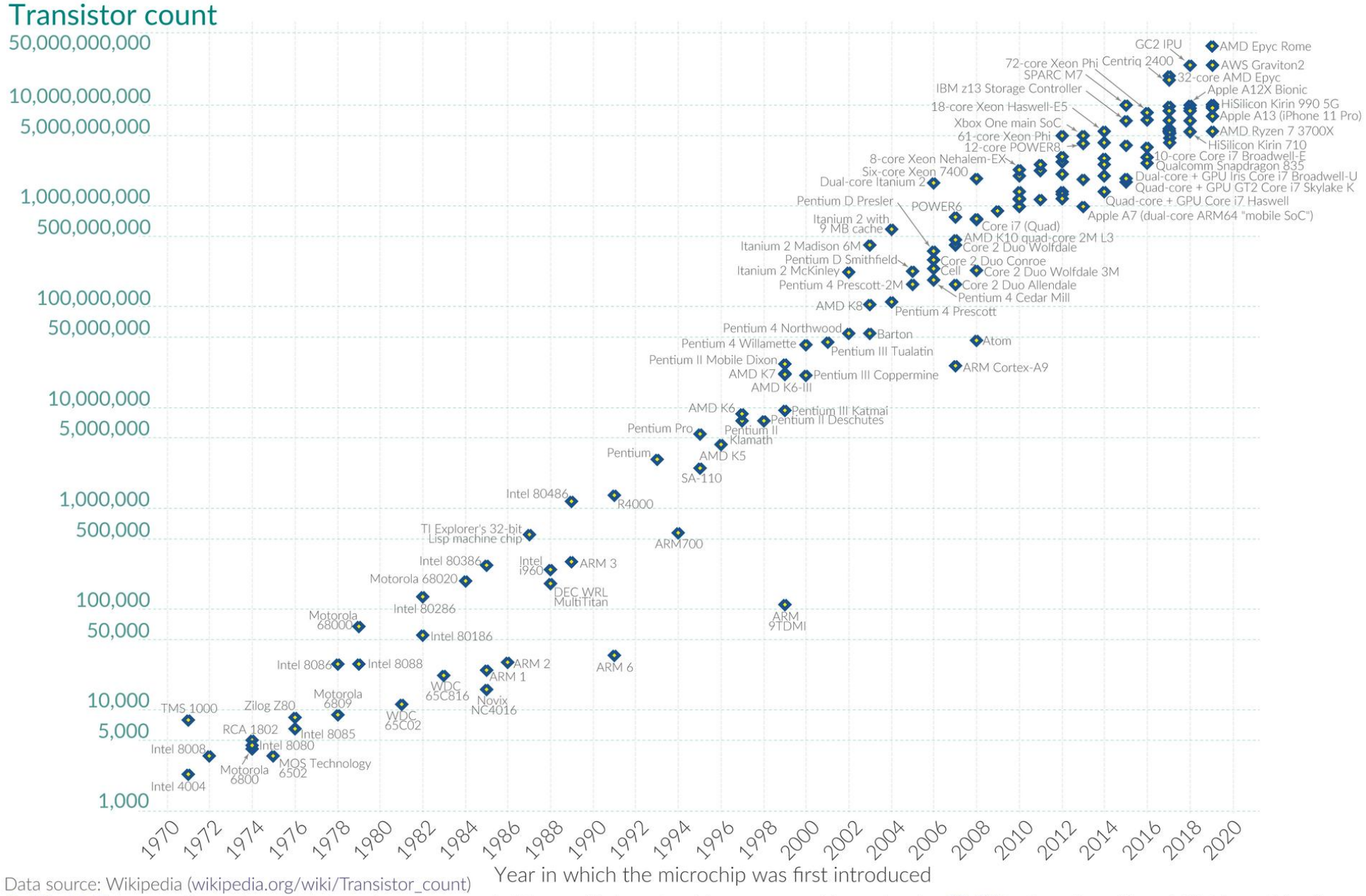
Moores Law

- Die Anzahl der Transistoren integrierter Schaltkreise mit minimalen Komponentenkosten verdoppelt sich etwa alle 2 Jahre
- Stammt von Gordon Moore (Gründer und ehemaliger CEO von Intel) aus der Zeit zwischen 1965 und 1975 und war zumindest bis 2020 gültig
- Bestimmt maßgeblich das Tempo der Entwicklung in der IT

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

Our World in Data

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.



Data source: Wikipedia (wikipedia.org/wiki/Transistor_count)
OurWorldinData.org – Research and data to make progress against the world's largest problems.

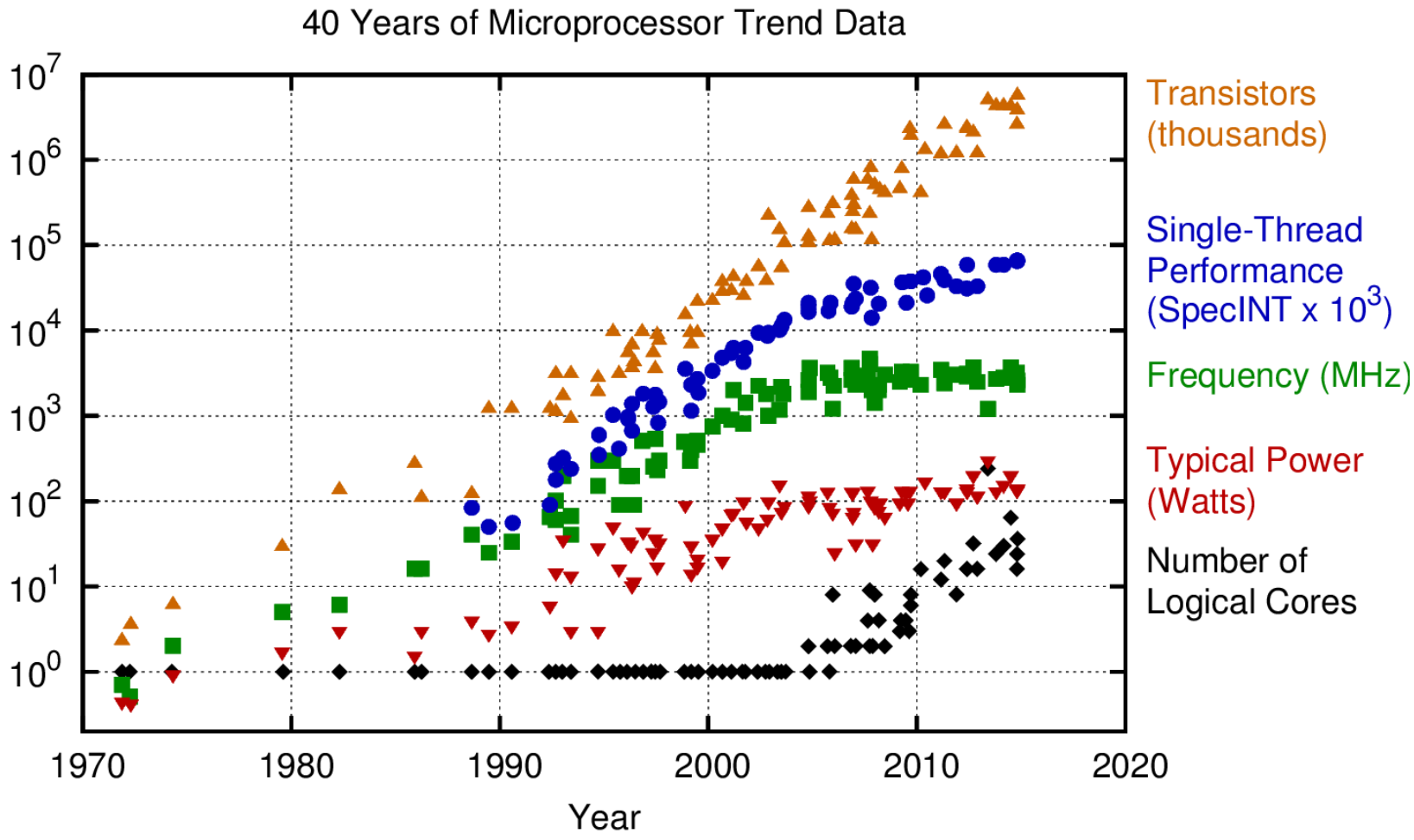
Moores Law

- Aus dem mooreschen Gesetz lässt sich nicht direkt ableiten, dass die Rechenleistung von Computern mit der Anzahl der Transistoren linear zunimmt

- Moderne Prozessoren verwenden z. B. Transistoren für integrierte Cache-Speicher, die lediglich die Datenbereitstellung beschleunigen (nicht die arithmetische Rechenleistung)
- Wikipedia vergleicht beispielsweise 2 Versionen des Intel Pentium III: „Katmai“ mit 500 MHz und „Coppermine“ mit 1000 MHz
  => Bei 3-facher Transistorzahl und doppelter Taktfrequenz stieg die Leistung „nur“ um den Faktor 2,2

- Seit Einführung von Mehrkernprozessoren wird die steigende Transistoranzahl häufig durch zusätzliche Prozessorkerne umgesetzt. Diese erhöhen die parallele Verarbeitungsfähigkeit, aber nicht zwingend die Rechenleistung im gleichen Maß

Historische Trends bei Prozessoren



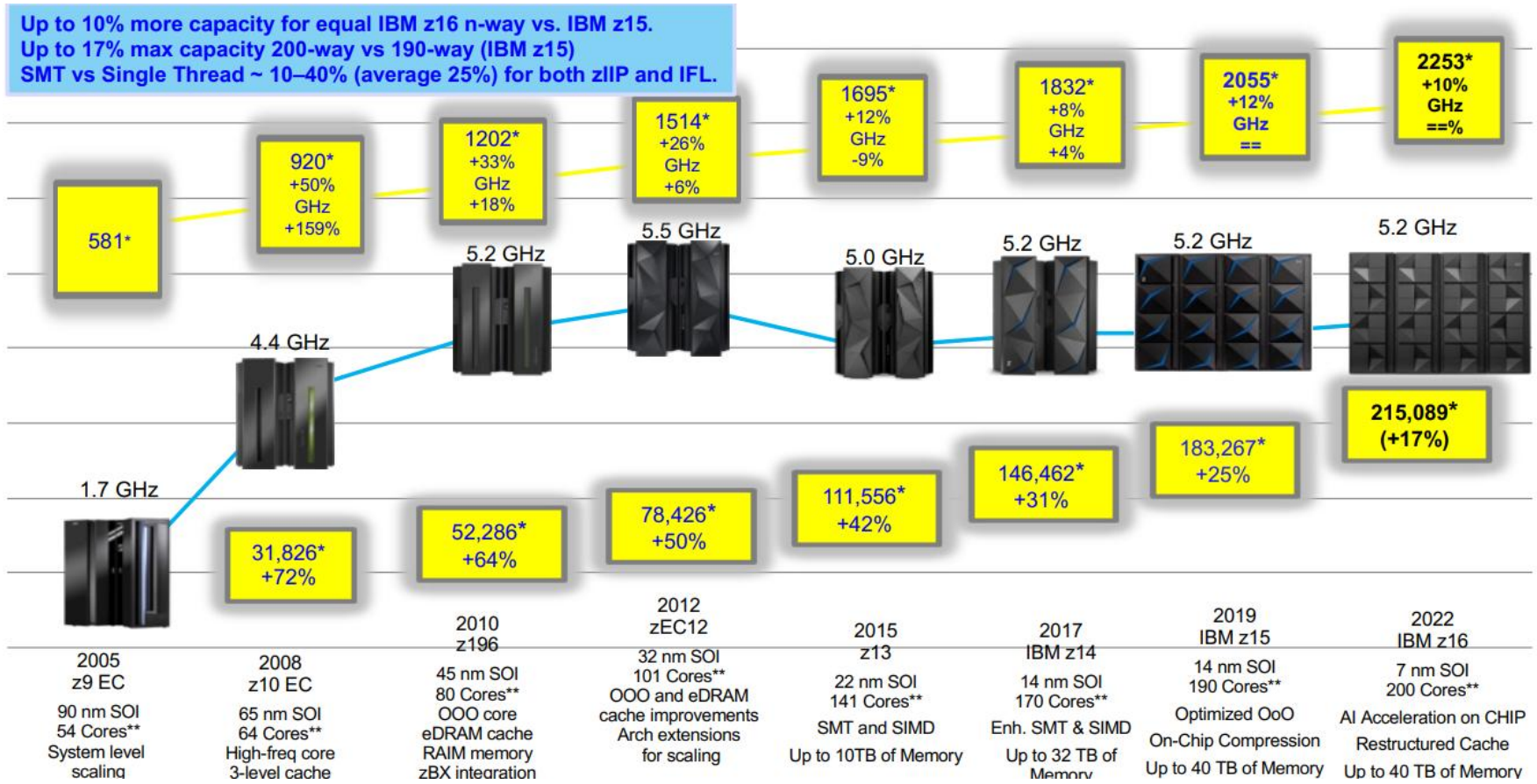
Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2015 by K. Rupp

Clock Rate am Beispiel IBM Mainframe

Clock Rate:

- ENIAC 100 KHz
- Intel 8080 2 MHz
- Intel P5 Pentium 100 MHz
- Intel Core i7 3,9 GHz
- IBM zEC12 5,5 GHz
- IBM z16 5,2 GHz

- Grund:
- Leistungsaufnahme ~ Takt³

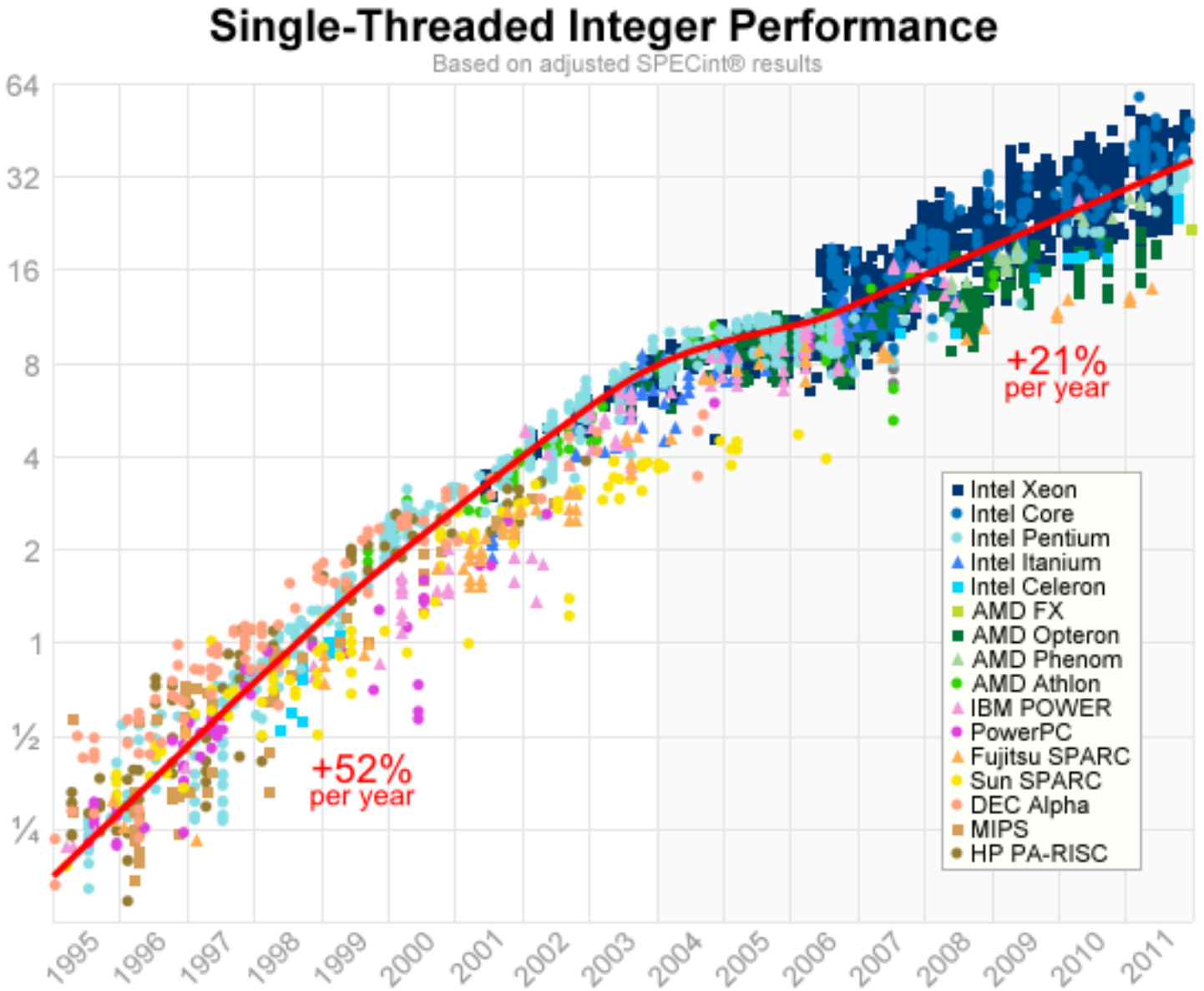


Gründe für konstante Taktraten in den letzten 20 Jahren

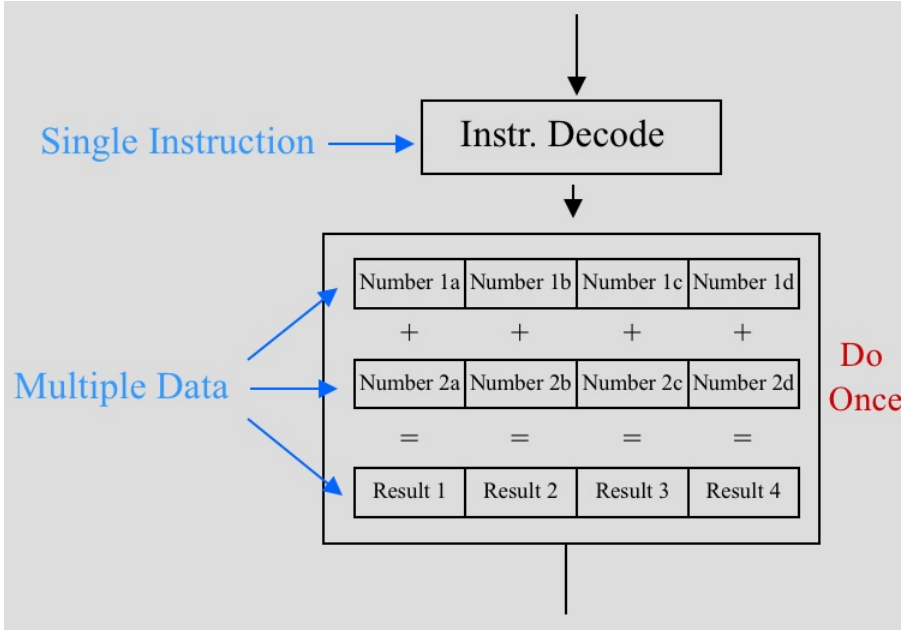
- Um den Takt zu erhöhen, muss man die Spannung ebenfalls erhöhen, denn Transistoren schalten bei höherer Frequenz sonst nicht mehr zuverlässig
- Zusammenhang zwischen Takt f und Spannung V ist empirisch in etwa gleichförmig ($V \sim f$), höhere Taktung, höhere Spannung
- Die Leistungsaufnahme wächst im Ergebnis etwa mit der dritten Potenz der Taktfrequenz ($P \sim f^3$)
- Also: doppelter Takt => 8-fache Verlustleistung, dreifacher Takt => 27-fache Verlustleistung
- Überhitzung!

- Konsequenz: „Thermal Wall“
- Um 2005 erreichte man etwa 3–4 GHz als praktikables Limit

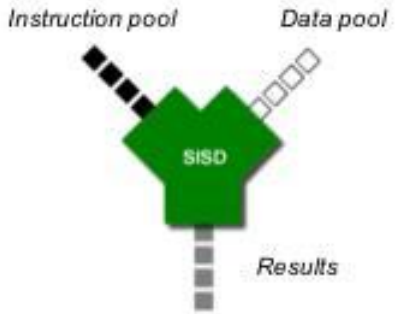
Single-Thread Performance Integer Processing (SPECint)

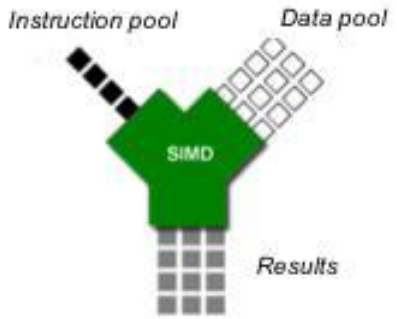


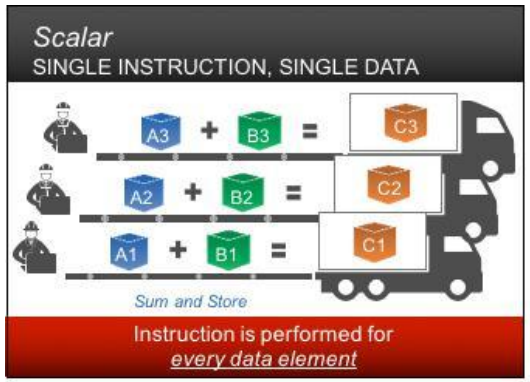
Single Instruction / Multiple Data (SIMD)

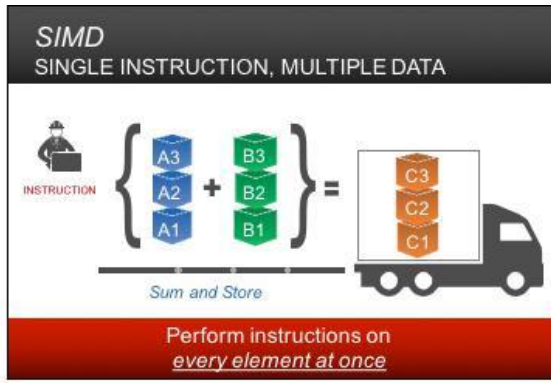


- Verbesserung der Single Thread Performance beispielsweise durch SIMD: eine Instruktion kann mehrere Datenelemente gleichzeitig verarbeiten – typischerweise in einem Vektorregister
- Ein Core hat also eine Vector Unit mit speziellen Maschineninstruktionen
- diese Instruktionen arbeiten auf mehreren Daten gleichzeitig (eine Form der Superskalarität)
- Beispiel: Ein 256-Bit breites Vektorregister kann acht 32-Bit-Ganzzahlen gleichzeitig (elementweise) addieren oder multiplizieren
  => 8 Additionen parallel in einem Takt
- Anwendungen sind z. B. Kryptographie, Analytics und (Big) Data Mining



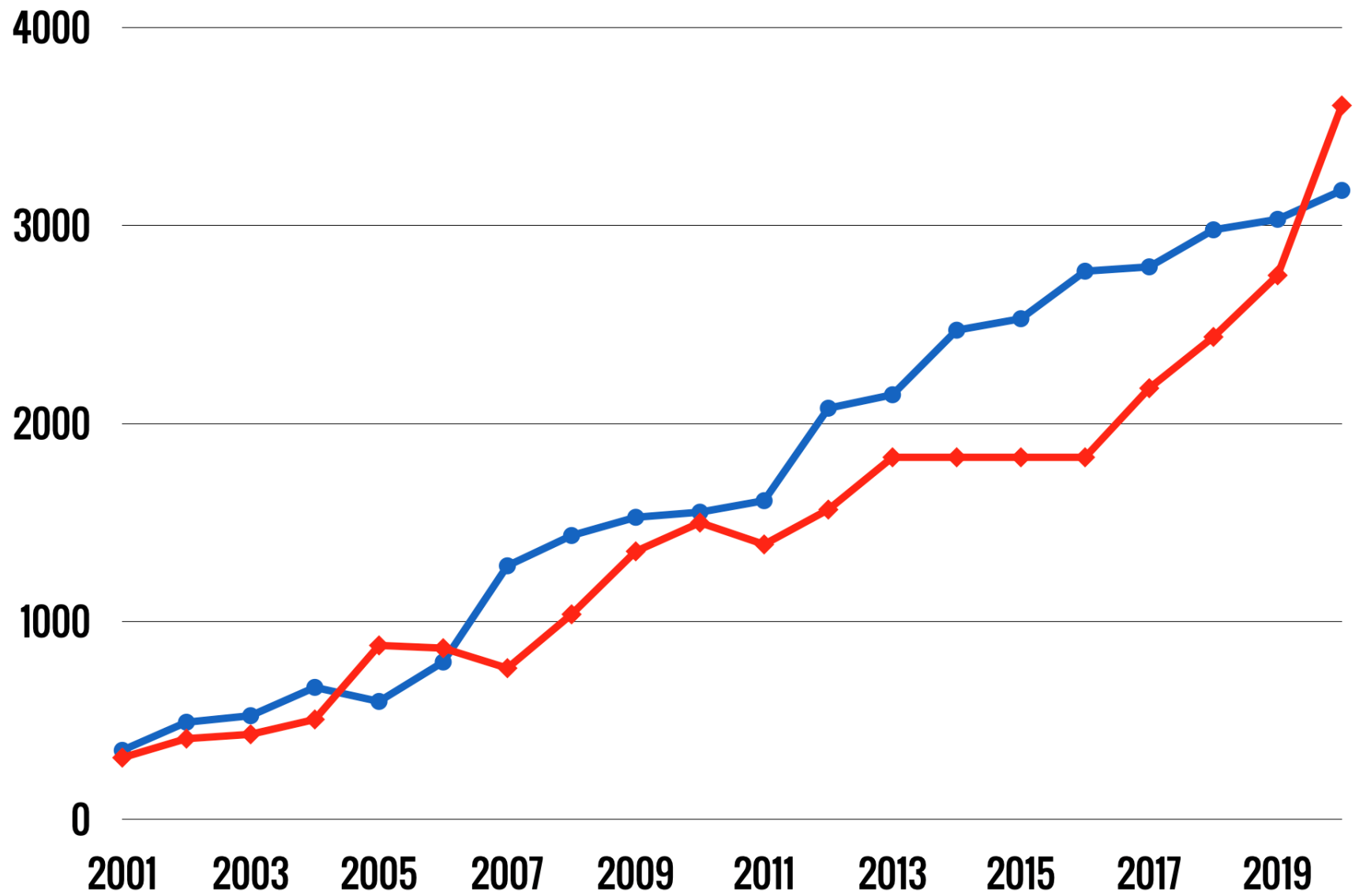






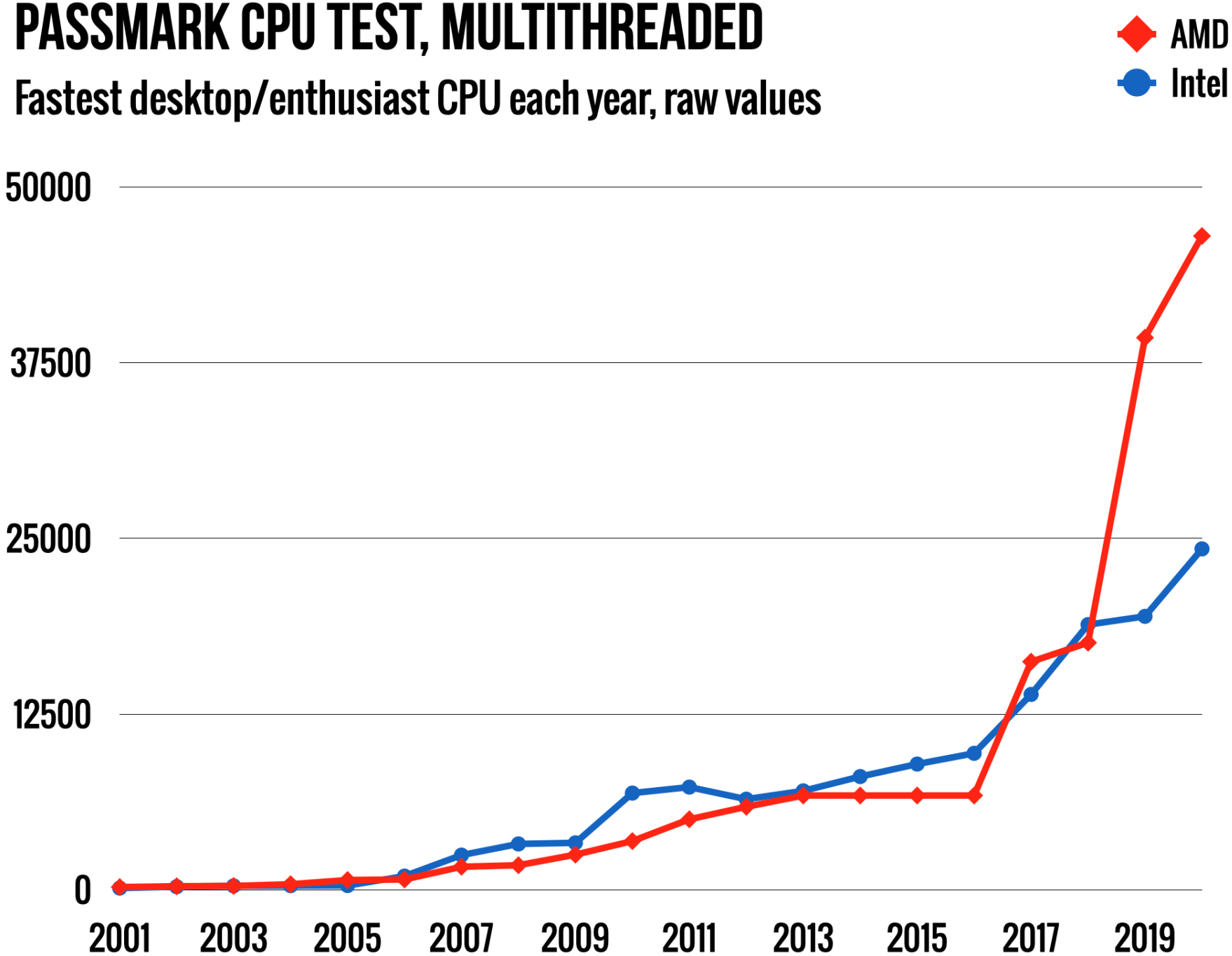
PASSMARK CPU TEST, SINGLETHREADED, 2001-2020

Fastest desktop/enthusiast CPU each year, raw values



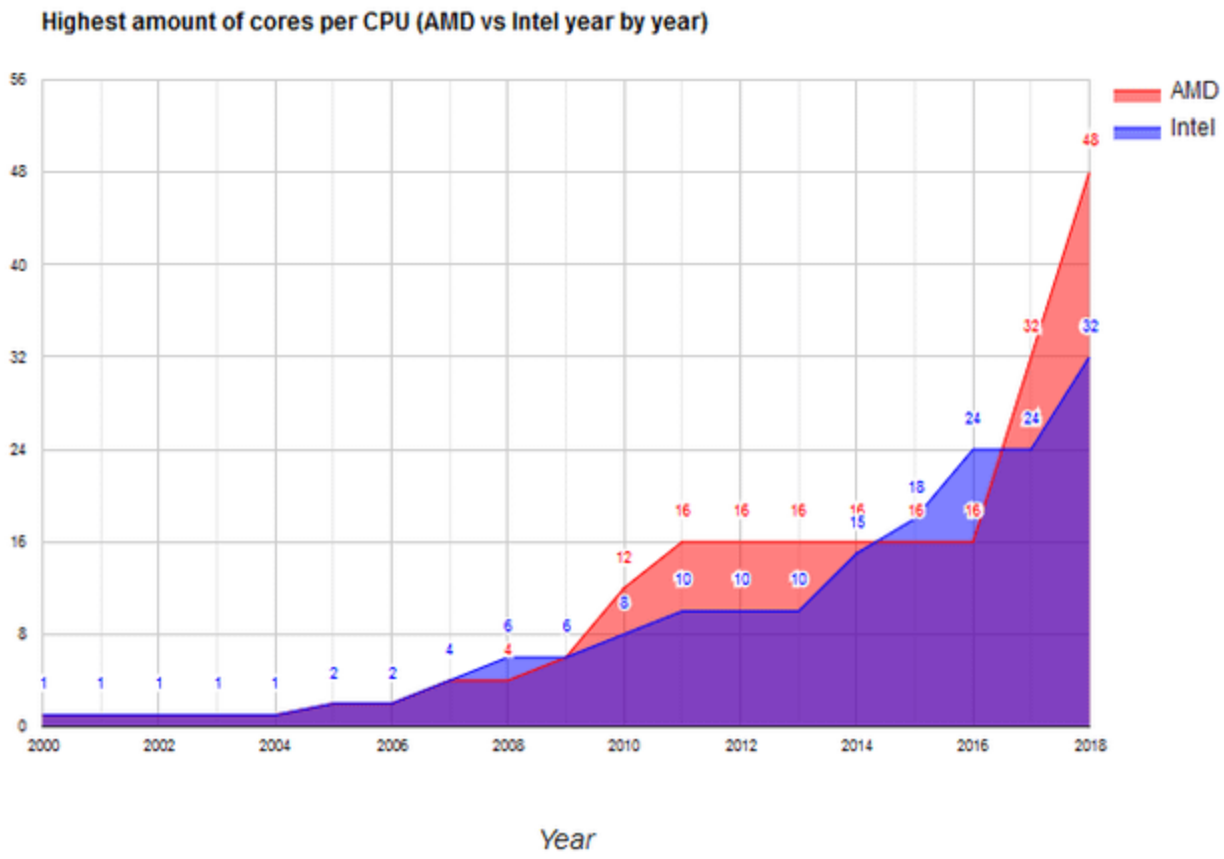
PASSMARK CPU TEST, MULTITHREADED

Fastest desktop/enthusiast CPU each year, raw values



Mehrkernprozessoren

- Auch bei Mehrkernprozessoren stoßen wir aktuell an Grenzen, 8–16 Kerne (bzw. 64 bei Server-Systemen) sind vermutlich der „Sweet Spot“
- Auch hier: thermische Grenzen
- Zudem: Speicher- und Bandbreitenlimits für parallele Zugriffe („Memory Wall“)
- Steigende Transistoranzahl durch zusätzliche Prozessorkerne erhöht die *parallele* Verarbeitungsfähigkeit; Problem: beschränkte Parallelisierbarkeit
- Nach dem amdahlschen Gesetz limitiert bereits ein geringer sequenzieller Anteil des Codes die theoretische Beschleunigung
- Ist z. B. 10 % des Programmcodes nicht parallel ausführbar, so ist bei *beliebig vielen* Kernen nur ein maximaler Speedup von 10 erreichbar



Ausblick

- Aufgabenverteilung: „Effizienz- und Performance-Kerne“

- Spezialbeschleuniger als Co-Prozessoren:
  Zur Bearbeitung spezieller Aufgaben werden auf diesen Workload optimierte spezielle Prozessoren mit entsprechenden Maschinenbefehlen vorgesehen und benötigten erheblich weniger Rechenzeit als ein Universalprozessor
  => z. B. dedizierte Blöcke für Matrix-Multiplikationen / KI-Inferenz (Tensor Cores), Video-Encoding, Verschlüsselung, …

- General-Purpose computing on Graphics Processing Units:
  Grafikprozessoren (GPUs) werden nicht nur für 3D-Grafik, sondern für allgemeine Rechenaufgaben eingesetzt (KI / Machine Learning, High-Performance Computing, Krypto-Mining, …)
  Massive, gleichförmige Rechenoperationen auf tausenden Kernen (GPU-Kern ist viel einfacher, spezialisierter und schwächer als ein CPU-Kern, aber dafür gibt es tausende davon, die massiv parallel rechnen)

Ausblick

– Verbessertes Chip-Design (z. B. in den kommenden Jahren 3D-Stacking: Speicher und Logik in mehreren Ebenen übereinander führt zu kürzeren Wegen und mehr Bandbreite)

– Zielstellung verändert sich wegen mobilen Geräten (Laptops, Smartphones, …) und wegen der hohen Kosten für Energie in Rechenzentren
– Heute: Mehr Leistung pro Watt

=> CPUs wachsen nicht mehr in GHz oder Cores, sondern in **Spezialisierung, Integration und Energieeffizienz**

HOCHSCHULE MAINZ
Wirtschaft
Business

Vielen Dank für Ihre Aufmerksamkeit!

Prof. Dr. Dirk Schweim
Professur für Wirtschaftsinformatik

Hochschule Mainz
University of Applied Sciences
Lucy-Hillebrand-Str. 2
55128 Mainz, Germany

T +49 6131 628-3315
E Schweim@HS-Mainz.de
W www.hs-mainz.de/schweim

University of
Applied Sciences