# Kapitel 2: Rechner im Überblick

Rechnersichten

Rechnerorganisation: Aufbau und Funktionsweise

Assembler

# Lernziele

- Die Begriffe Hardware, Software und Firmware kennen und unterscheiden können
- Die Prinzipien eines von-Neumann-Rechners kennen und erläutern
- Den groben Aufbau eines von-Neumann-Rechners skizzieren und erläutern können unter Verwendung der Begriffe
  - Arbeitsspeicher, Speicherzelle, Byte, Bit
  - Prozessor , Steuerwerk, Rechenwerk, Register
  - Memory – I/O Busse

# Hardware, Software und Firmware

## Hardware

- elektronische Schaltungen,
- I/O-Geräte,
- Speicher, ...

## Software

- durch Programme virtueller Rechner oberhalb der Maschinensprache beschriebene Algorithmen

## Firmware

- Programme, die bei der Fabrikation eines Rechners in Hardware eingebettet wurden (sind in der Regel in Nur-Lesespeicher (ROM, PLA, ...) abgespeichert)

# Hardware vs. Software (1)

- Früher Software, heute Hardware:
  - ganzzahlige Multiplikation, Division
  - Gleitkommaarithmetik
  - ...

- Aber auch: RISC-Konzept (Reduced Instruction Set)
  - Teile wieder mit Software ausgeführt
  - Beispiel: ganzzahlige Division in RISC-Prozessoren

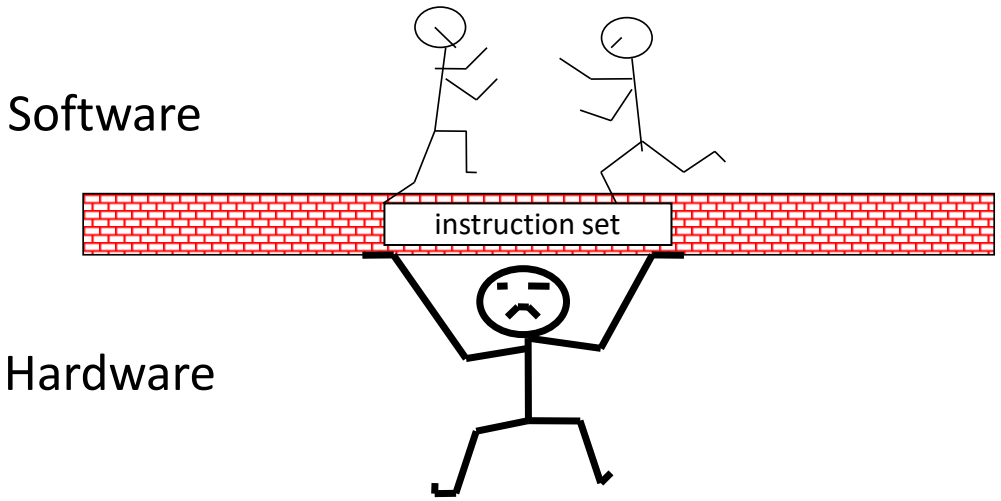
# Hardware vs. Software (2)

Festlegung des Befehlssatzes eines Rechners entscheidet, welche Teilaufgaben durch Hardware und welche durch Software gelöst werden können.

**Beispiel:** Realisiert man die Division als Hardware oder als Software?

Beantwortung hängt an

- Kosten
- Performanz
- Nutzung



# Exkurs: Hardware / Software Co-Design

- Anwendung insbesondere bei „**eingebetteten Systemen**“, die nicht frei programmierbar sein müssen
- Welche Komponenten in Hardware realisiert bzw. in Software/Firmware implementiert werden, entscheidet der „Systemarchitekt“ oder wird (halb-)automatisch anhand von Laufzeitanalysen der Software entschieden.
- Auslagern rechenzeitintensiver Teile in Hardware!

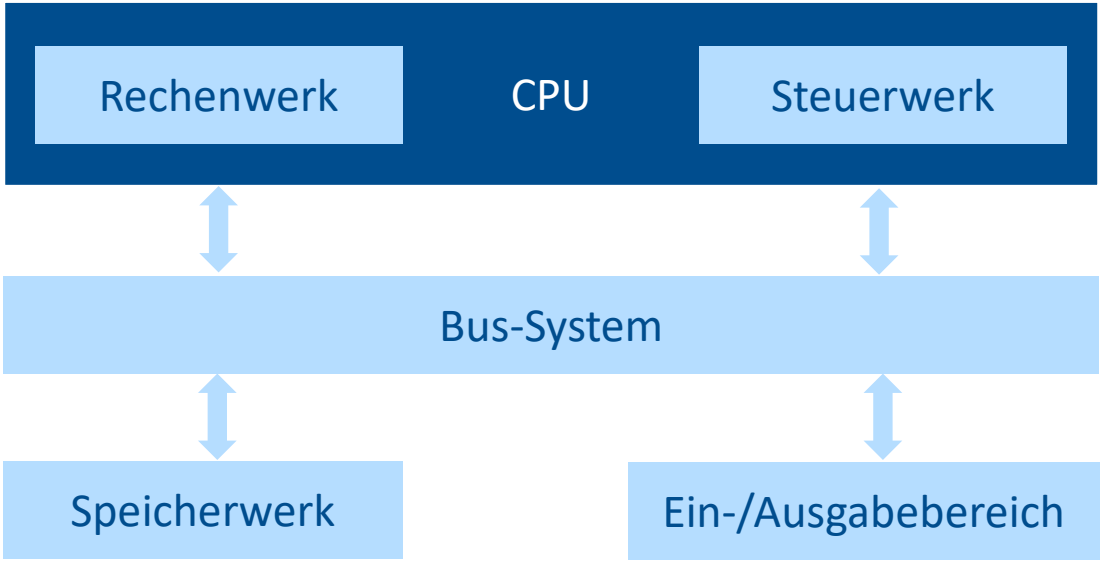
# John von Neumann (1903 - 1957)

- Der Aufbau moderner Rechner wurde stark von dem Mathematiker John von Neumann beeinflusst
- Biographie:
  - 1903 in Budapest geboren
  - Studium in Budapest und Zürich
  - 1933-1957 am „Institute for Advanced Studies“, Princeton
  - 1945 mitverantwortlich für den Bau der ENIAC

John von Neumann, 1932

# Das von-Neumann Prinzip

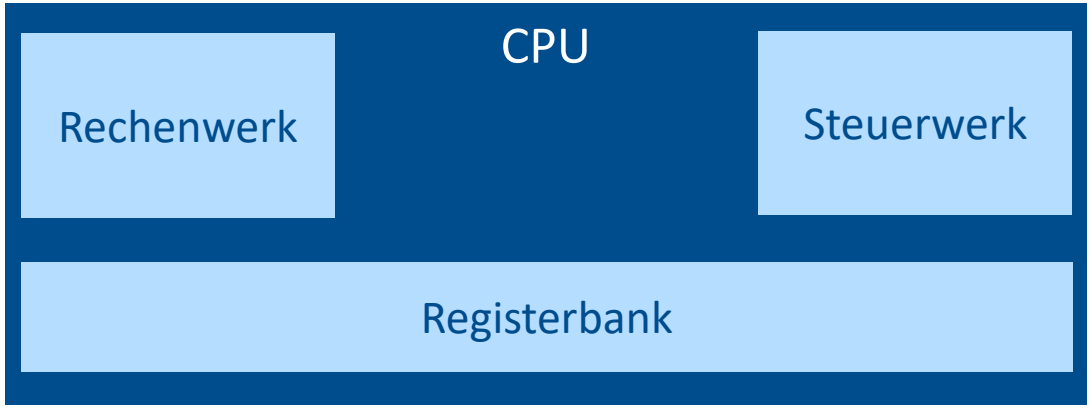
- Voll-elektrischer Rechner
- Binäre Darstellung der Daten
- Benutzung einer arithmetischen Einheit
- Benutzung eines Steuerwerkes
- Interner Daten- und Programmspeicher



# Prinzipieller Aufbau eines Rechners

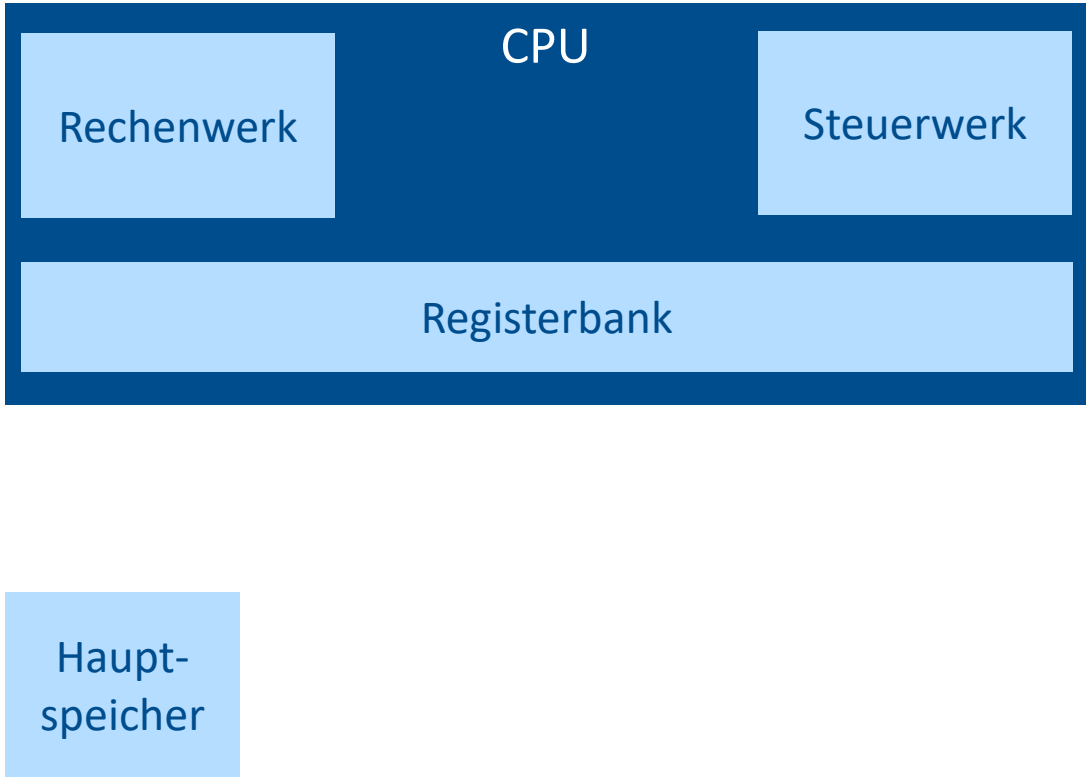
# Prinzipieller Aufbau eines Rechners

- Prozessor (CPU)



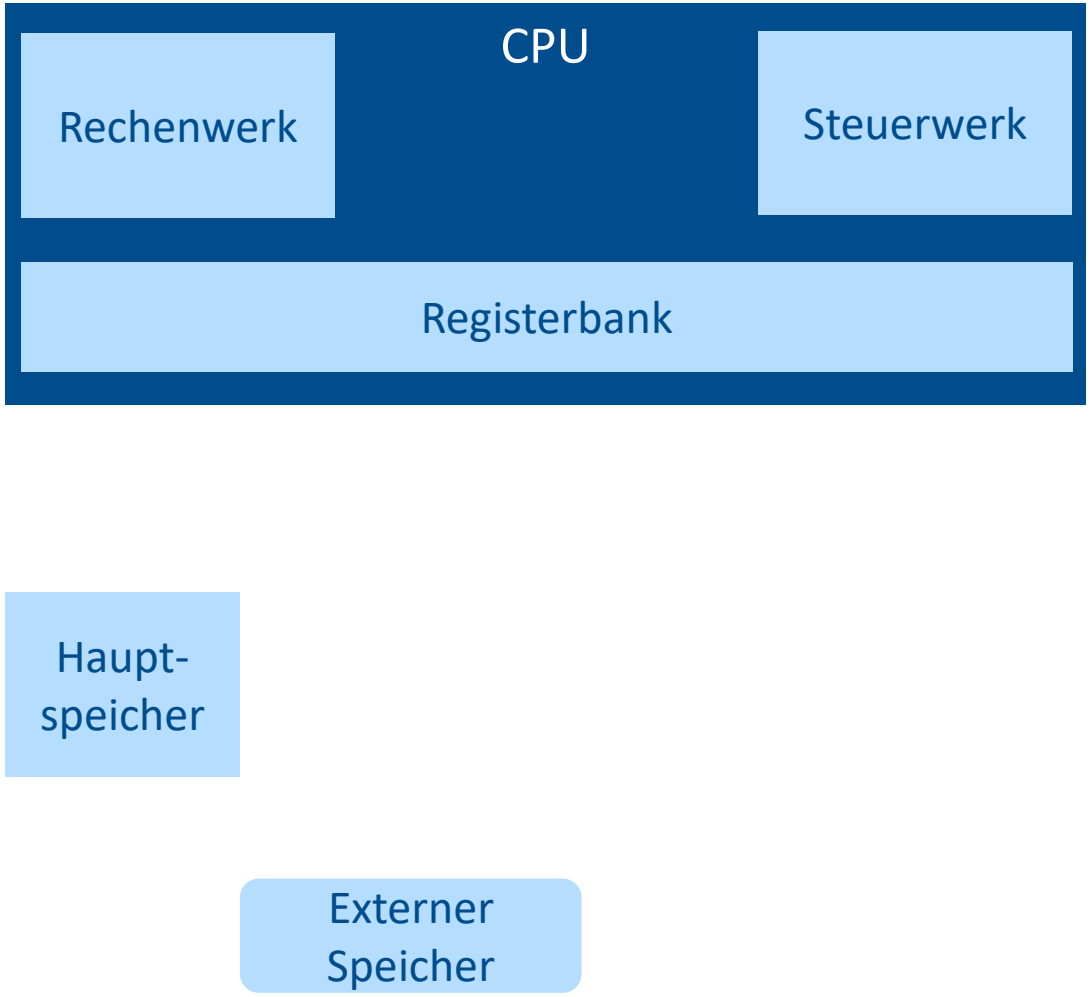
# Prinzipieller Aufbau eines Rechners

- Prozessor (CPU)
- Hauptspeicher



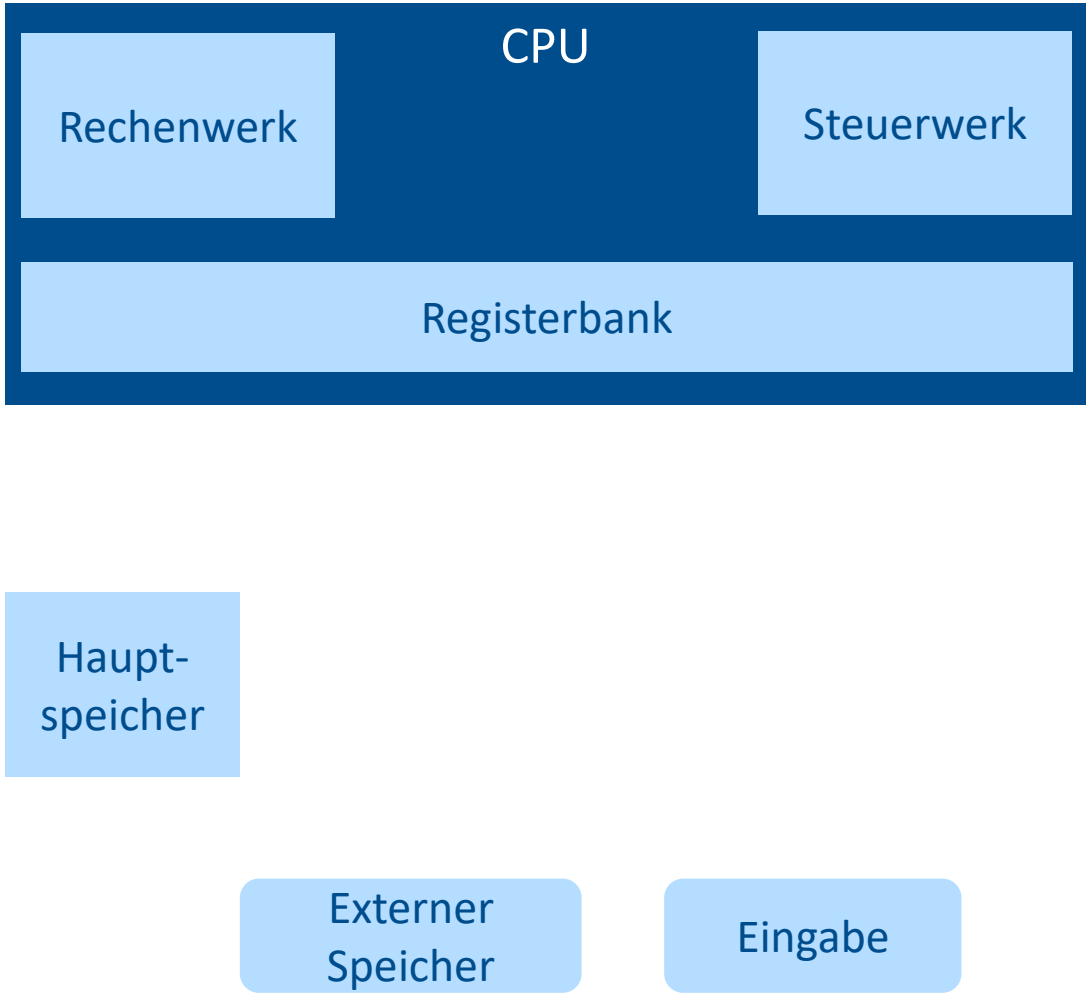
# Prinzipieller Aufbau eines Rechners

- Prozessor (CPU)
- Hauptspeicher
- Externe Speicher



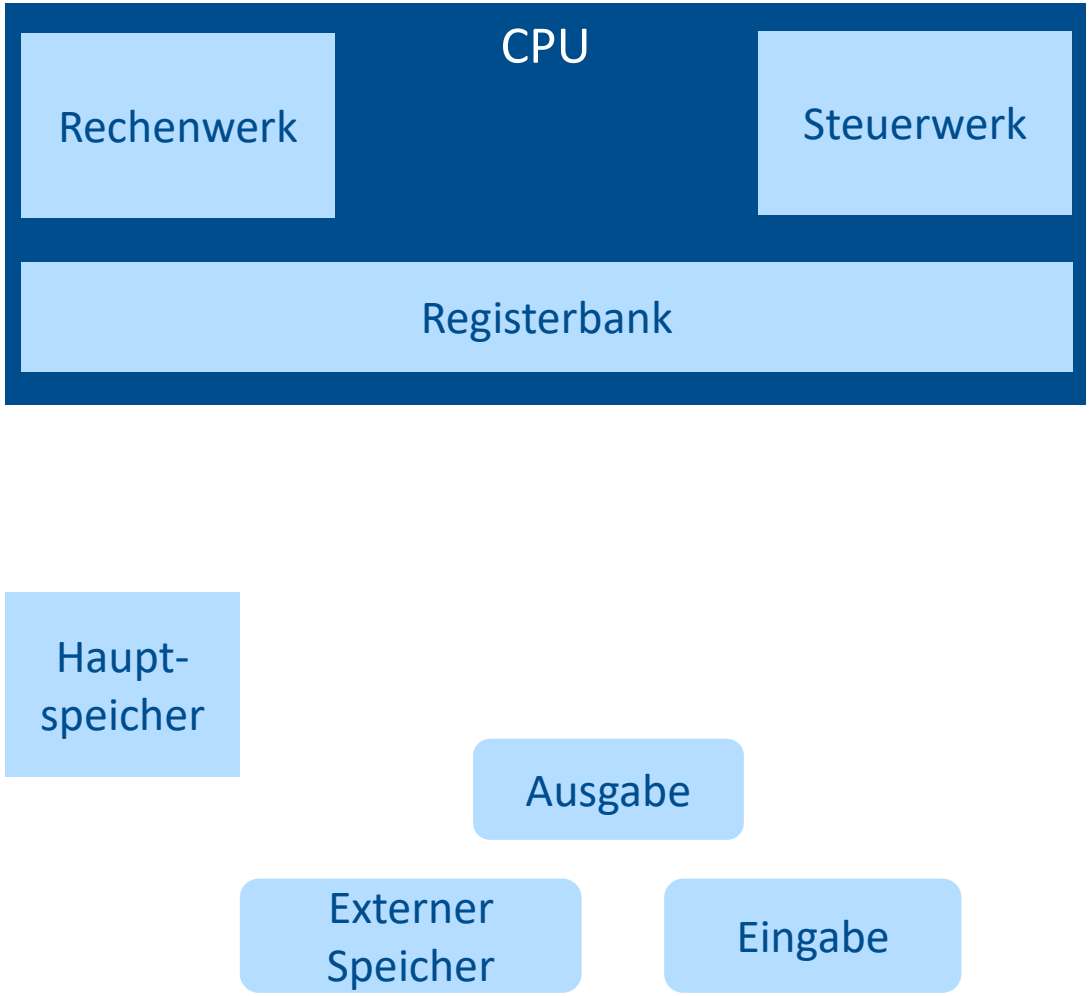
# Prinzipieller Aufbau eines Rechners

- Prozessor (CPU)
- Hauptspeicher
- Externe Speicher
- Eingabegeräte (Tastatur, Maus)



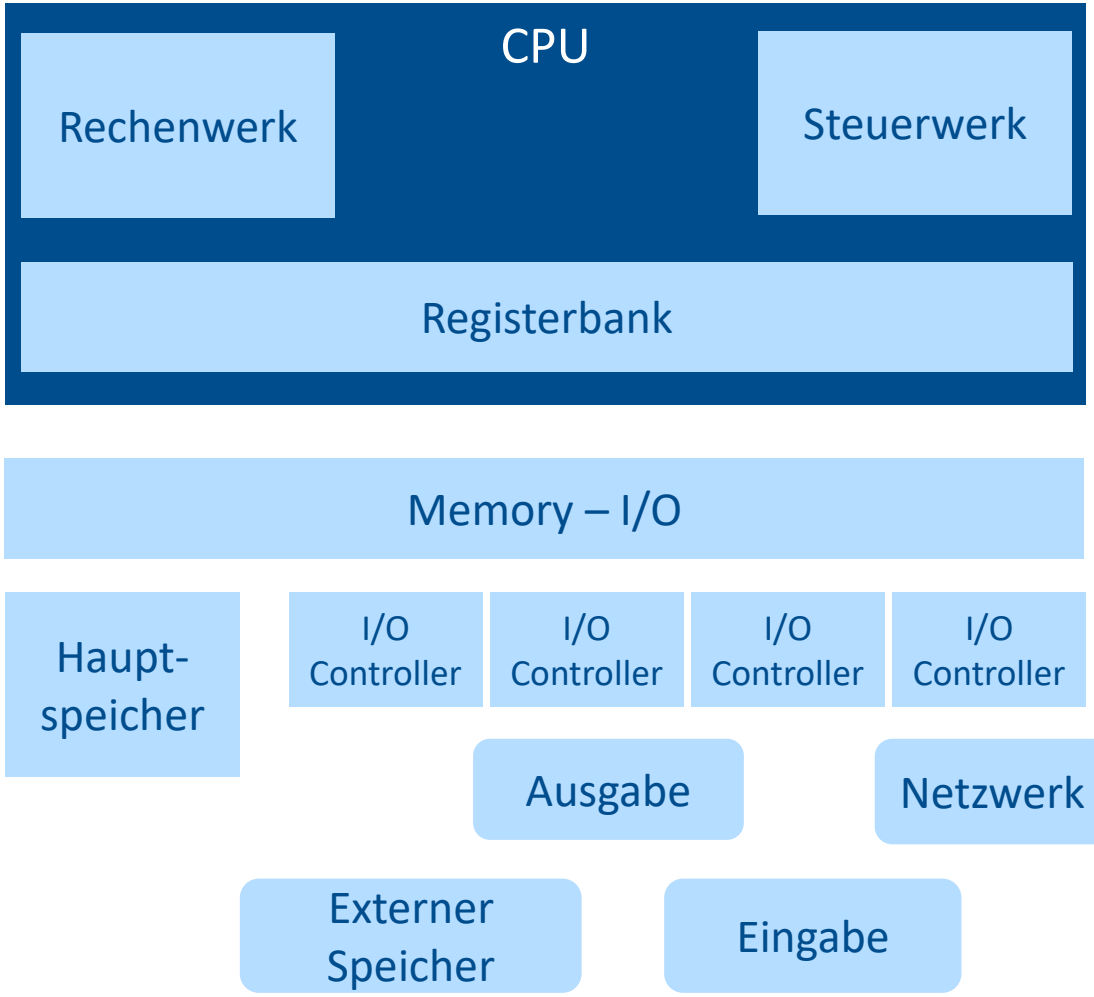
# Prinzipieller Aufbau eines Rechners

- Prozessor (CPU)
- Hauptspeicher
- Externe Speicher
- Eingabegeräte (Tastatur, Maus)
- Ausgabegeräte (Bildschirm, Drucker, Plotter)



# Prinzipieller Aufbau eines Rechners

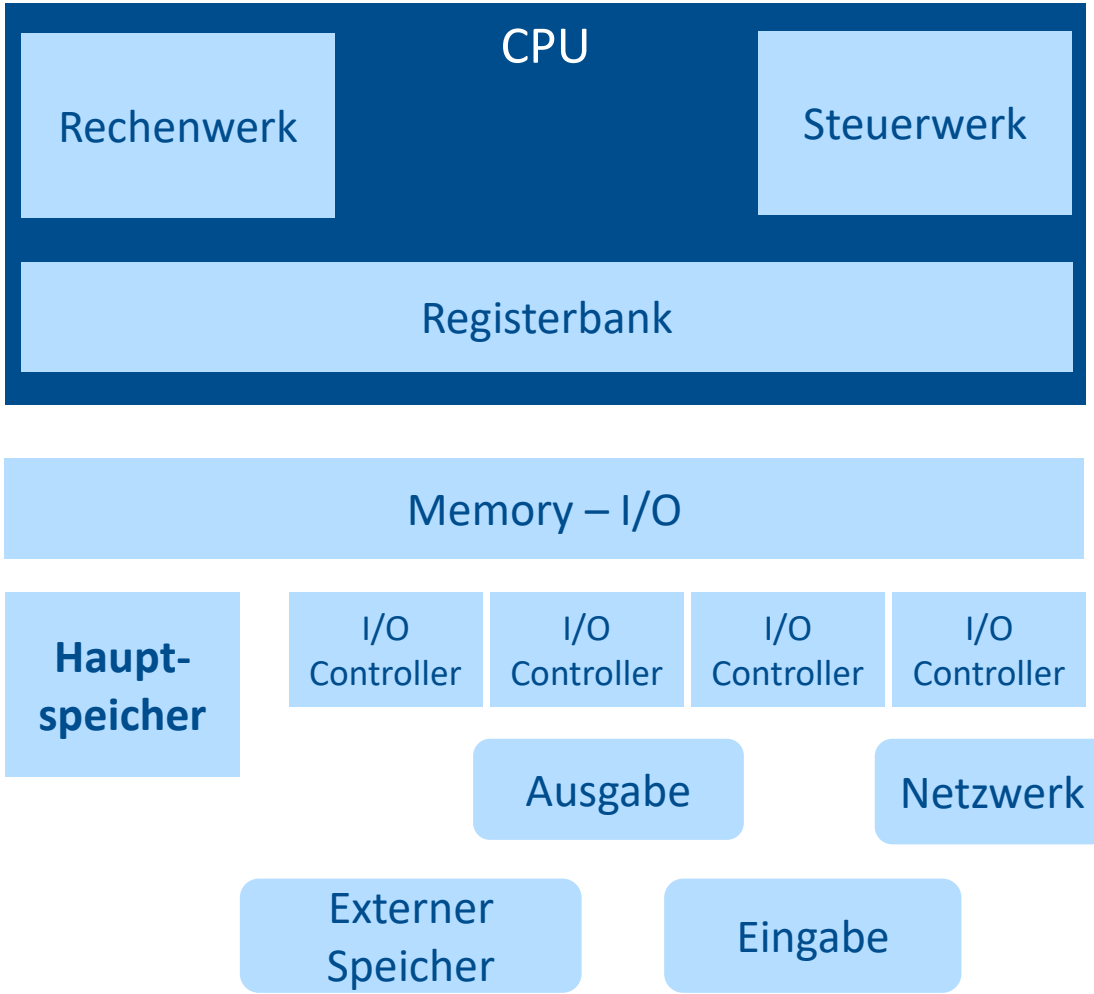
- Prozessor (CPU)
- Hauptspeicher
- Externe Speicher
- Eingabegeräte (Tastatur, Maus)
- Ausgabegeräte (Bildschirm, Drucker, Plotter)
- Busse



# Der Arbeitsspeicher

- erlaubt Abspeicherung
  - Von Programmen
  - „normalen Daten“
- besteht aus $n$ Speicherzellen
- Eine Speicherzelle
  - speichert eine Information bestehend aus $k$ Bit. Der Wert $k$ wird Wortbreite des Speichers genannt ($k = 8,16,32,64$)
  - wird über eine Adresse (entspricht Hausnummer) aus dem Bereich $[0..n-1]$ angesprochen
  - ist kleinste adressierbare Einheit des Speichers

Kerngrößen
- Anzahl der Speicherzellen
- Wortbreite des Speichers
- Zugriffszeit



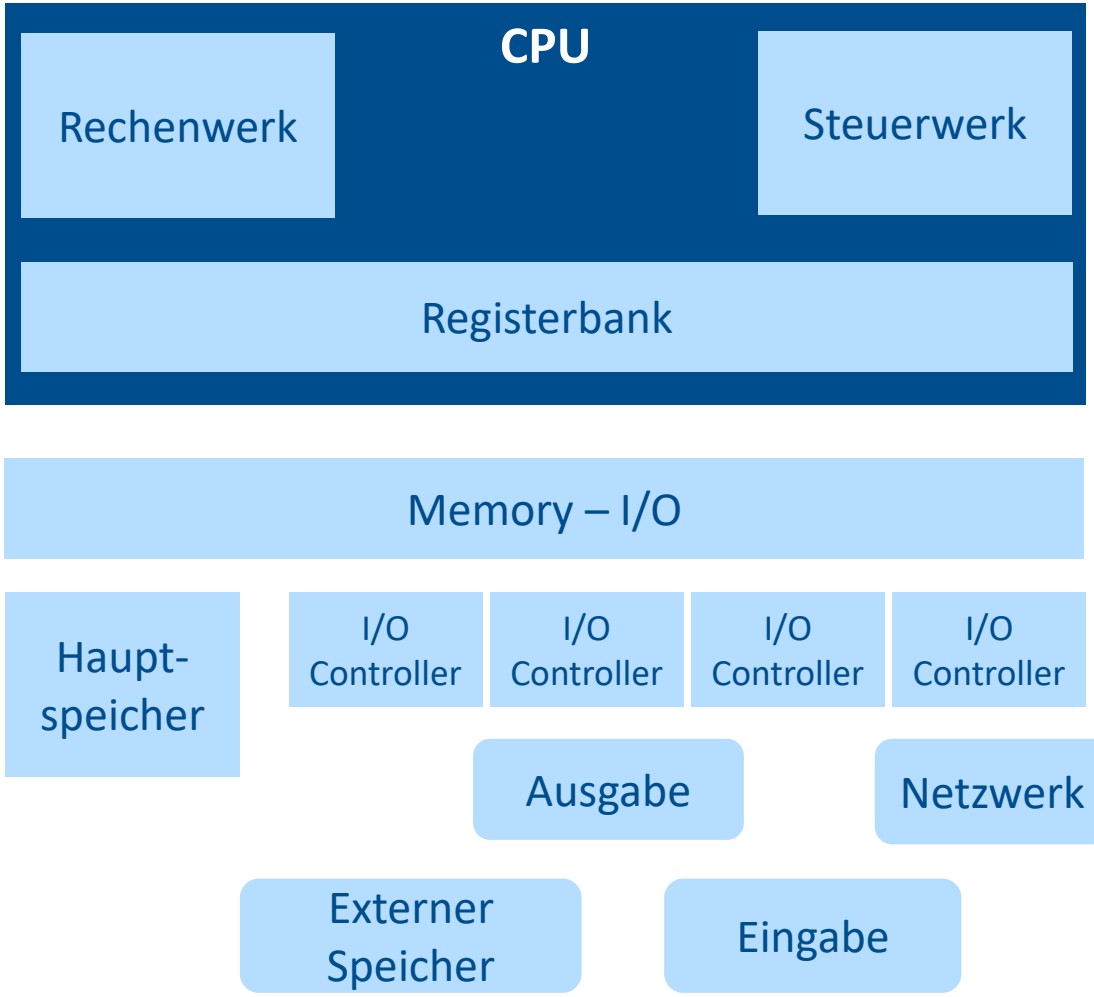
# Die Einheiten des Computers – Bits und Bytes

- Bit
  - Kleinste Informationsmenge, die in einem System mit zwei Werten gespeichert werden kann
  - 0 oder 1
  - Belegung des Bits wird durch angelegte Spannung definiert
- Byte
  - Nächstgrößere Einheit
  - 1 Byte = 8 Bit

# Der Prozessor

besteht in der einfachsten Form aus

- **Steuerwerk (Control Unit)**
  - holt durch Setzen geeigneter Signale den nächsten Maschinenbefehl aus dem Hauptspeicher in ein Register
  - analysiert durch Setzen geeigneter Signale das gelesene Wort
  - führt durch Setzen geeigneter Signale den gelesenen Maschinenbefehl aus
- **Recheneinheit (Arithmetic Logic Unit - ALU, Datenpfad)**
  - führt die Teilschritte (Vergleiche, Addition, ...) gemäß der gesetzten Signale aus
- **Registerbank**
  - Speicherelemente, sogenannte Register, auf die das Rechenwerk und das Steuerwerk ohne Wartezyklus schnell zugreifen können.



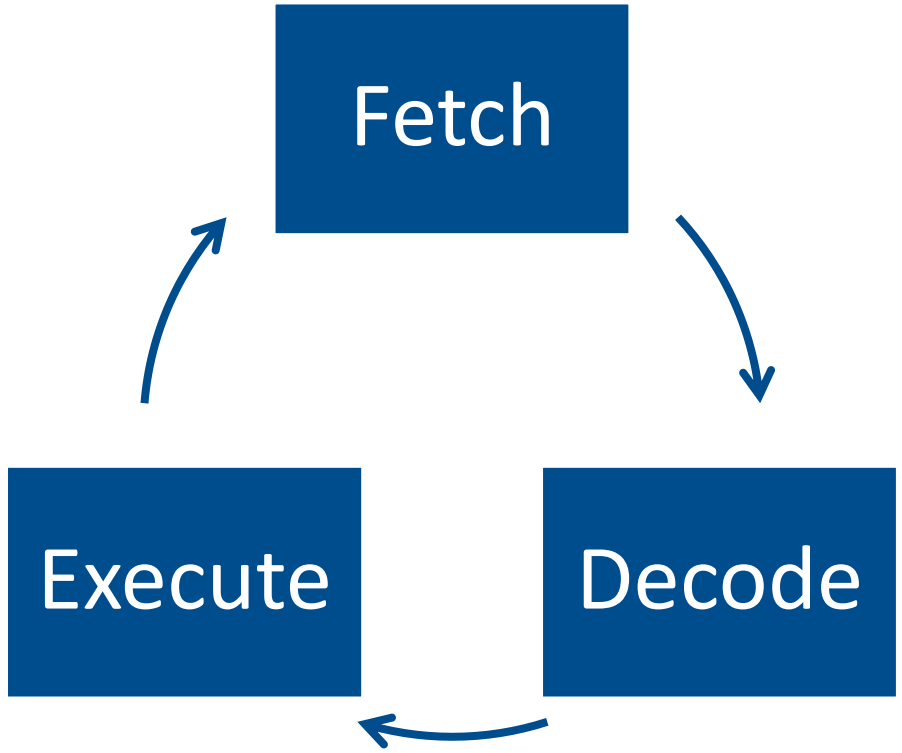
# Register des Prozessors

Vom Prozesser unbedingt benötigte Informationen

- Aktuell abzuarbeitende Befehl (Befehlsregister – **I**nstruction **R**egister (IR))
- Adresse des nächsten abzuarbeitenden Befehls im Hauptspeicher (Befehlszähler – **P**rogram **C**ounter (PC))
- Adresse im Hauptspeicher, ab der Daten und Informationen gespeichert werden können (**S**tack**p**ointer (SP))
- Von der ALU zu verarbeitende Daten (Arbeitsregister – General Purpose Register)

# Prinzipielle Arbeitsweise eines Prozessors

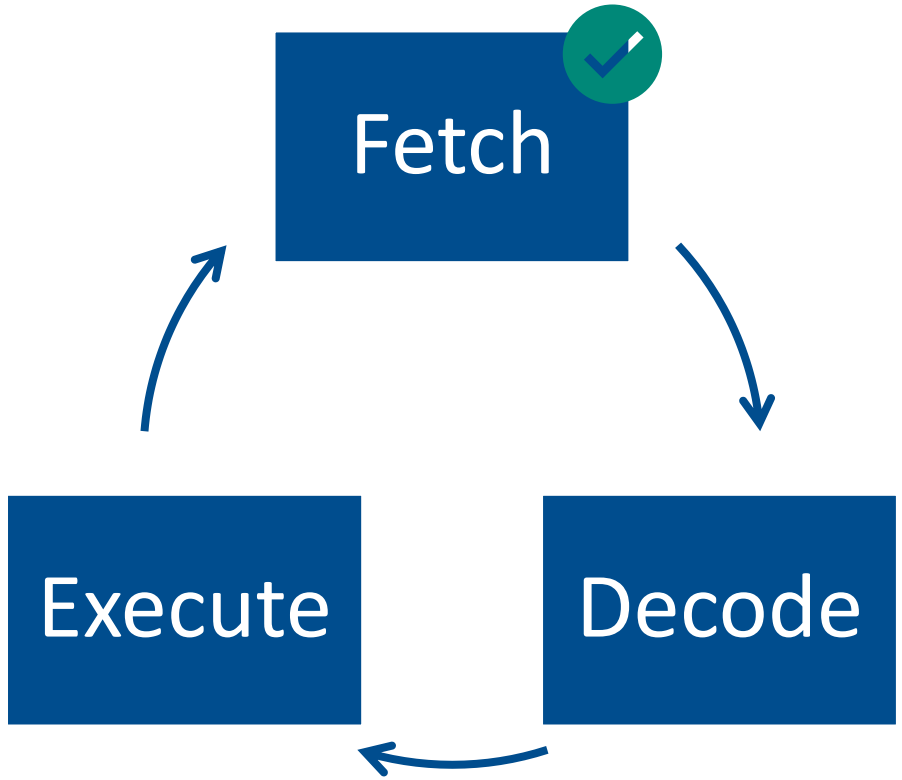
Der Prozessor arbeitet im Fetch-Decode-Execute Zyklus



# Prinzipielle Arbeitsweise eines Prozessors

Der Prozessor arbeitet im Fetch-Decode-Execute Zyklus

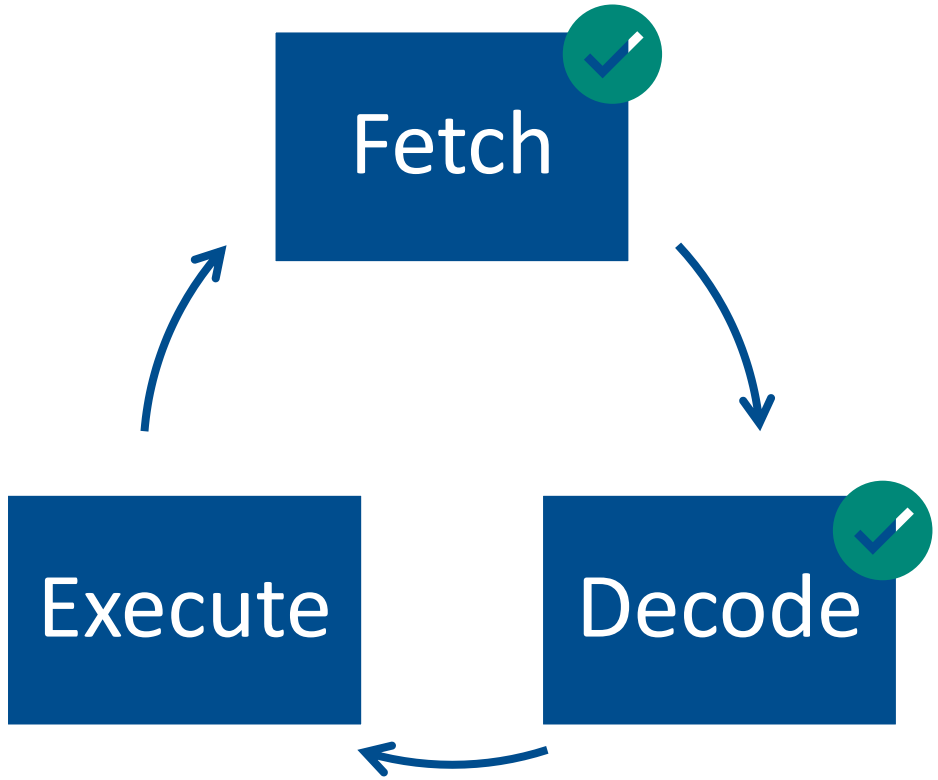
- **Fetch**: Hole den nächsten Maschinensprachebefehl aus dem Arbeitsspeicher und speichere ihn im Befehlsregister IR ab. Die benötigte Adresse steht im Befehlszähler PC.



# Prinzipielle Arbeitsweise eines Prozessors

Der Prozessor arbeitet im Fetch-Decode-Execute Zyklus

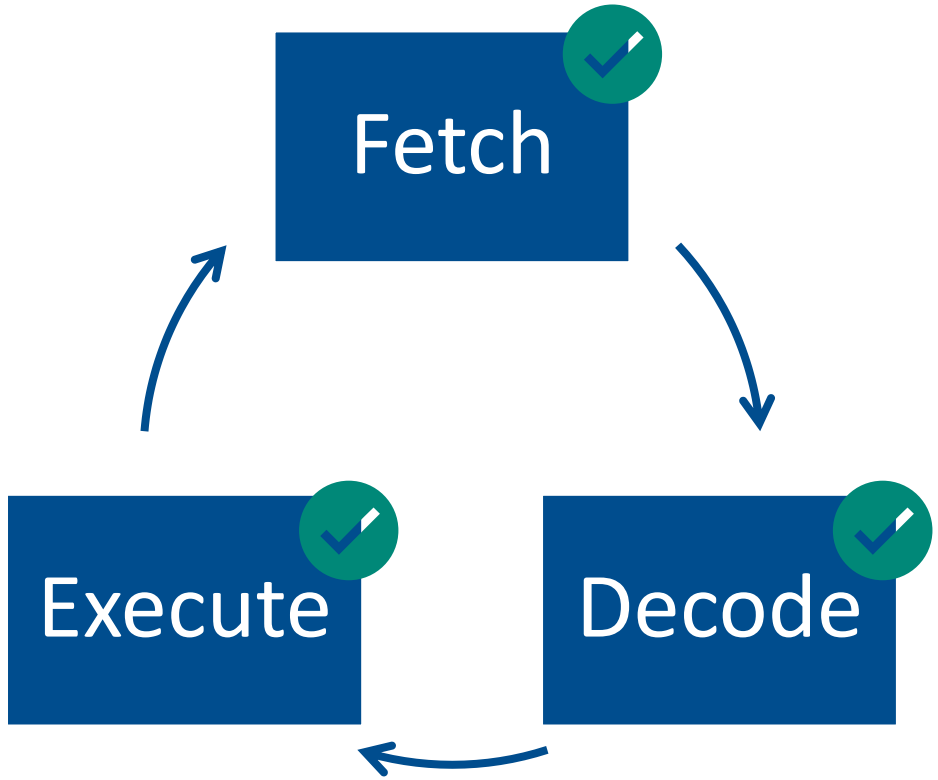
- **Fetch**: Hole den nächsten Maschinensprachebefehl aus dem Arbeitsspeicher und speichere ihn im Befehlsregister IR ab. Die benötigte Adresse steht im Befehlszähler PC.
- **Decode**: Analysiere den Befehl und lade die benötigten Daten.



# Prinzipielle Arbeitsweise eines Prozessors

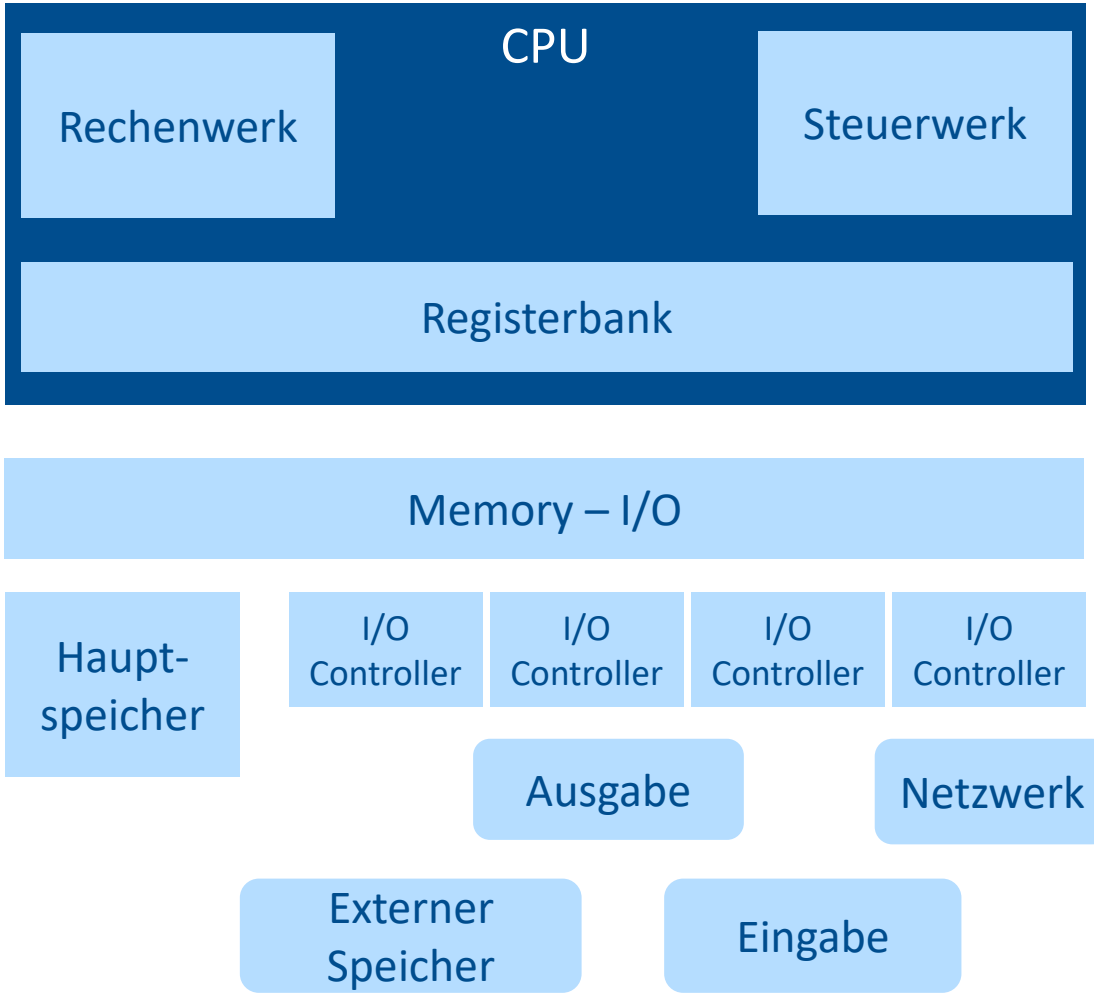
Der Prozessor arbeitet im Fetch-Decode-Execute Zyklus

- **Fetch**: Hole den nächsten Maschinensprachebefehl aus dem Arbeitsspeicher und speichere ihn im Befehlsregister IR ab. Die benötigte Adresse steht im Befehlszähler PC.
- **Decode**: Analysiere den Befehl und lade die benötigten Daten.
- **Execute**: Führe den Befehl aus und speichere das Ergebnis ab.

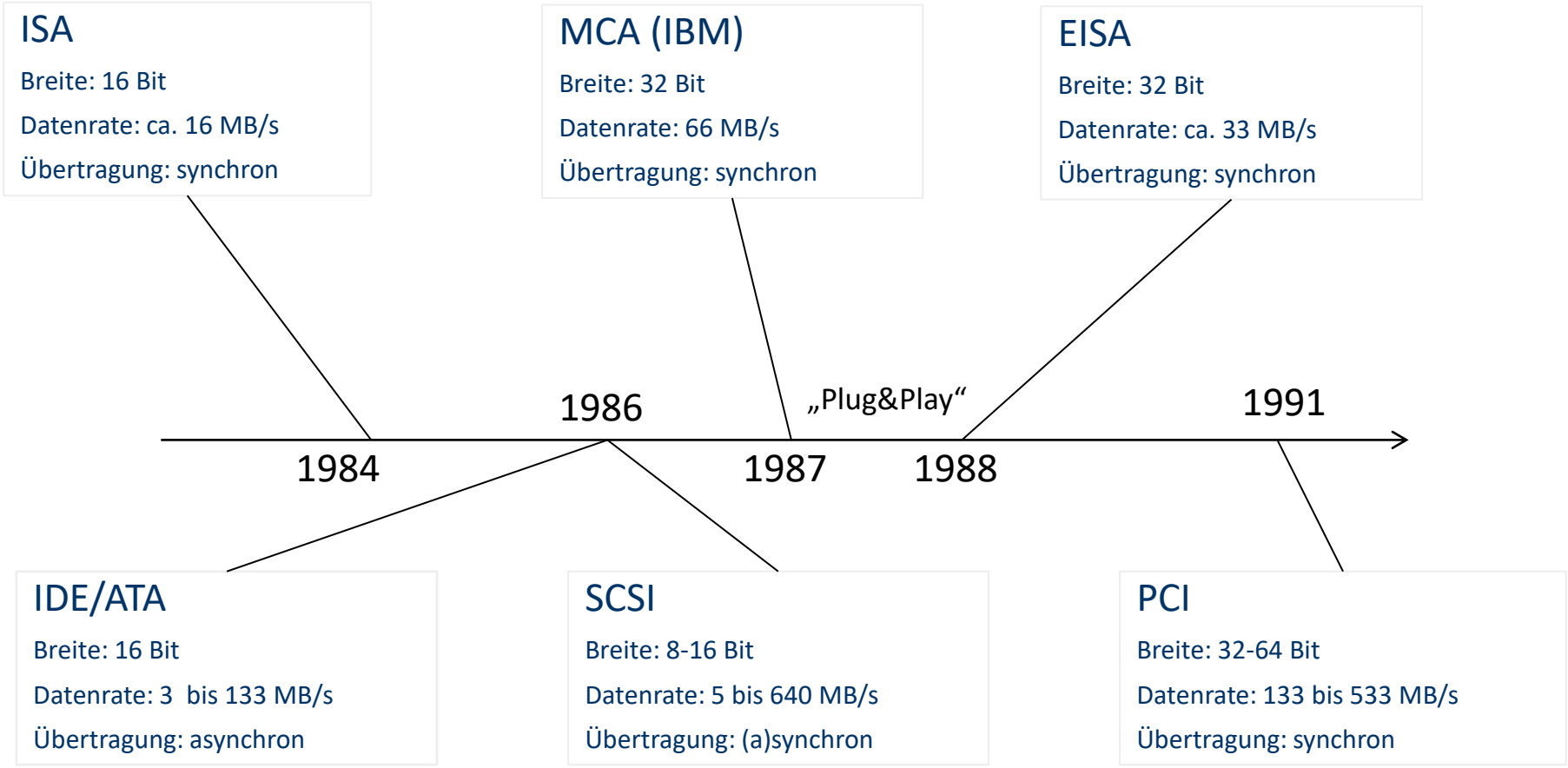


# Memory - I/O Busse

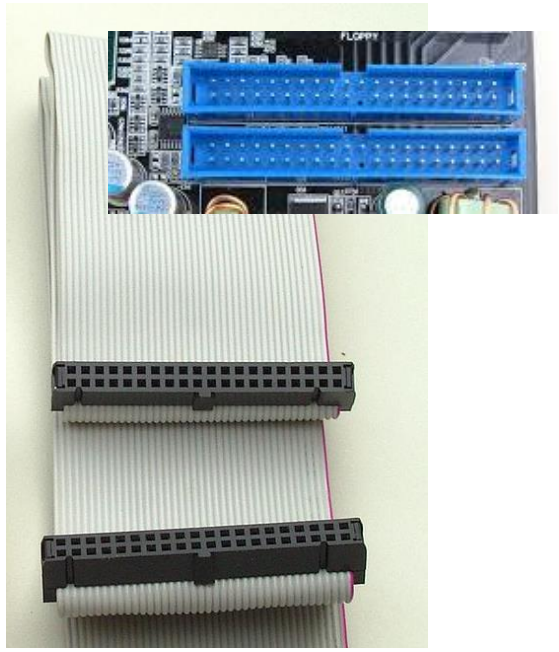
- Elektrische Leitungsbündel in einem digitalen Rechner
- Kommunikationsmedium, auf dem Daten fließen können
  - Zwischen CPU und Hauptspeicher
  - Zwischen CPU und I/O-Geräten
  - Zwischen Hauptspeicher und I/O-Geräten
- Normierte Stecker
- Standardisierte Interpretation der einzelnen Leitungen



# Bussysteme vorgestern …



# Bussysteme vorgestern …

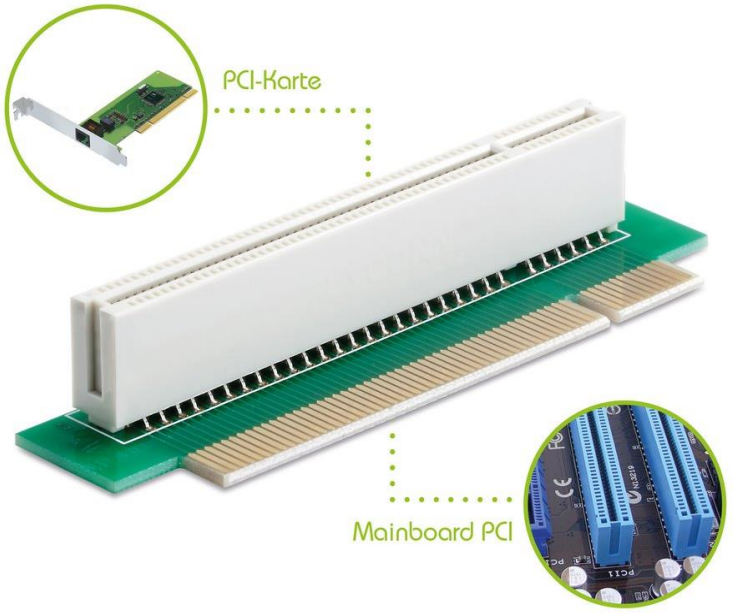


## IDE/ATA

Integrated Drive Electronics/
AT Attachment
IBM PC/AT (Advanced Technology)

## SCSI

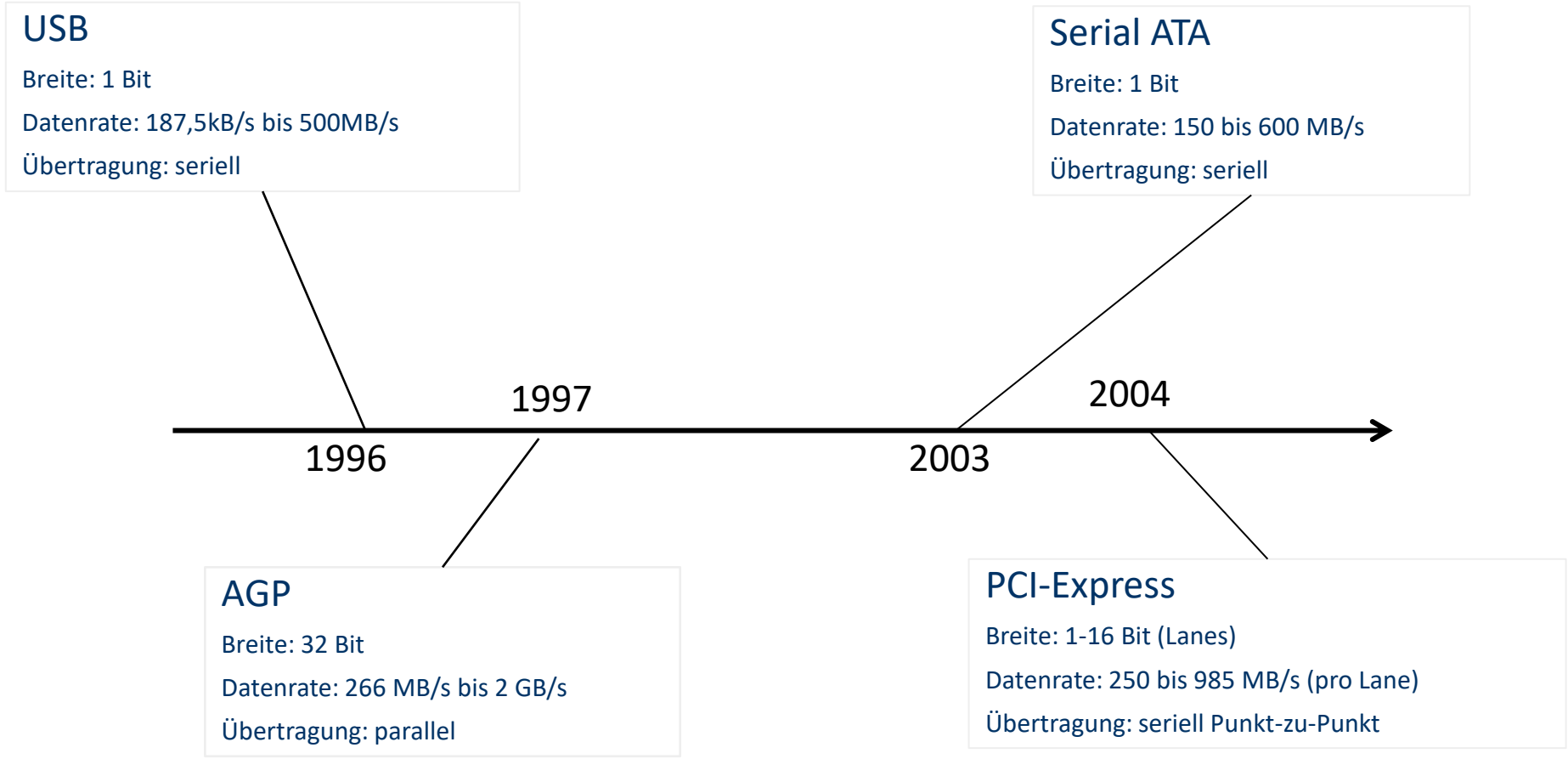
Small Computer System Interface
(sprich: ska-zi)



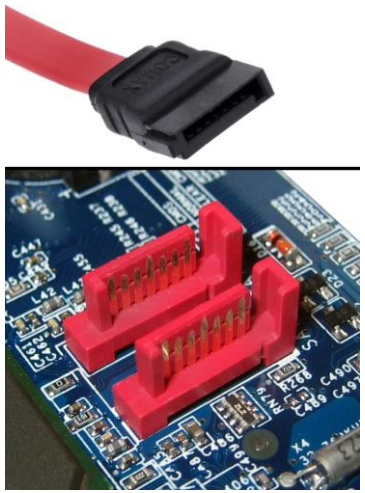

## PCI

Peripheral Component Interconnect

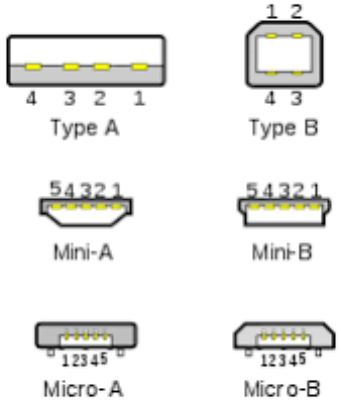
# … gestern


USB
Breite: 1 Bit
Datenrate: 187,5kB/s bis 500MB/s
Übertragung: seriell
Serial ATA
Breite: 1 Bit
Datenrate: 150 bis 600 MB/s
Übertragung: seriell
1997
2004
1996
2003
AGP
Breite: 32 Bit
Datenrate: 266 MB/s bis 2 GB/s
Übertragung: parallel
PCI-Express
Breite: 1-16 Bit (Lanes)
Datenrate: 250 bis 985 MB/s (pro Lane)
Übertragung: seriell Punkt-zu-Punkt

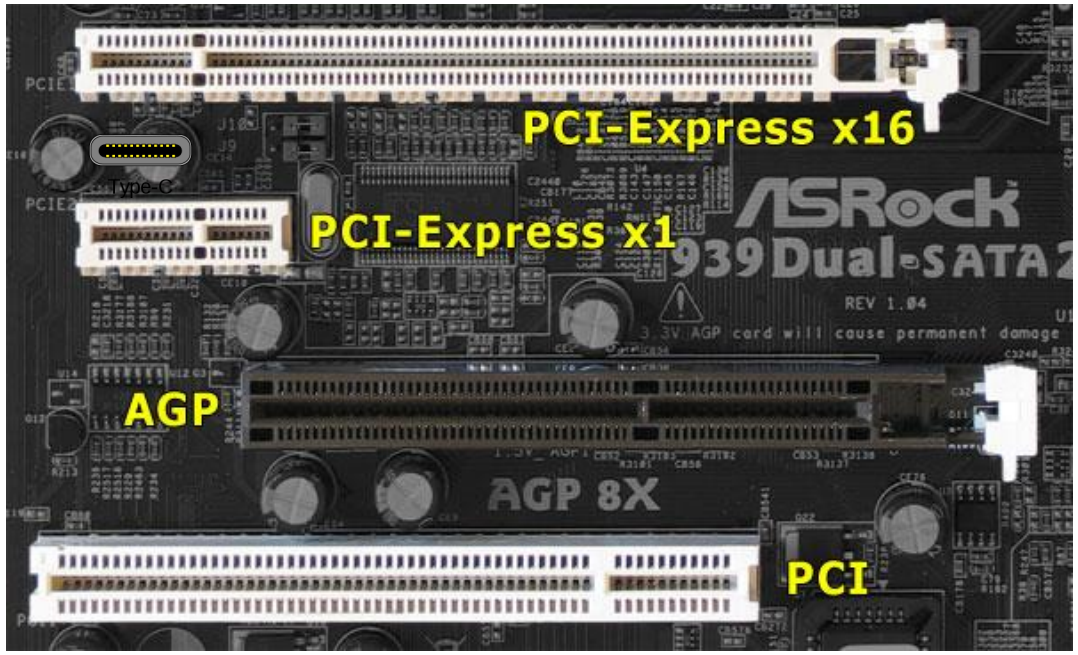
# … gestern



Serial ATA



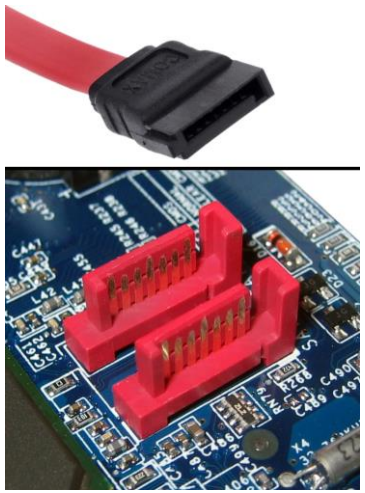

USB - Universal Serial Bus

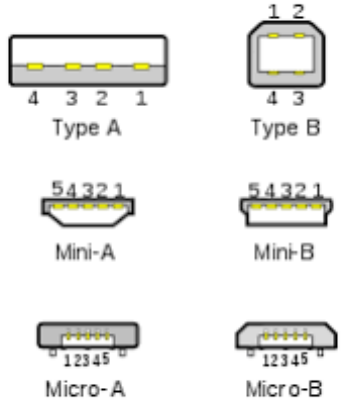


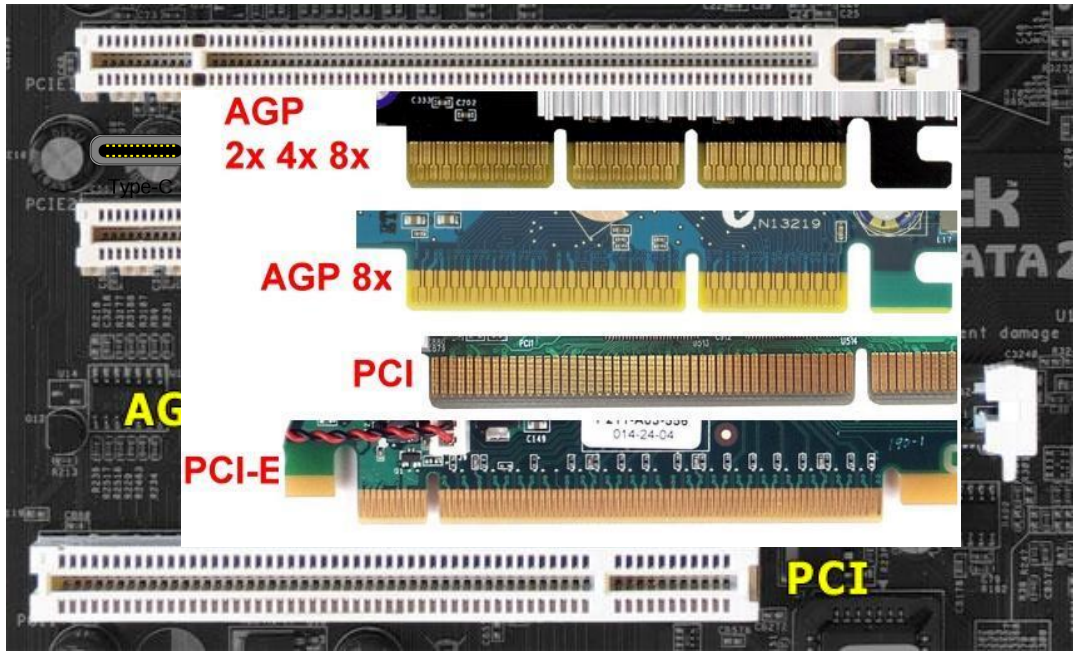

AGP -
Accelerated Graphics Port

PCI-Express
Peripheral Component Interconnect Express

# … gestern





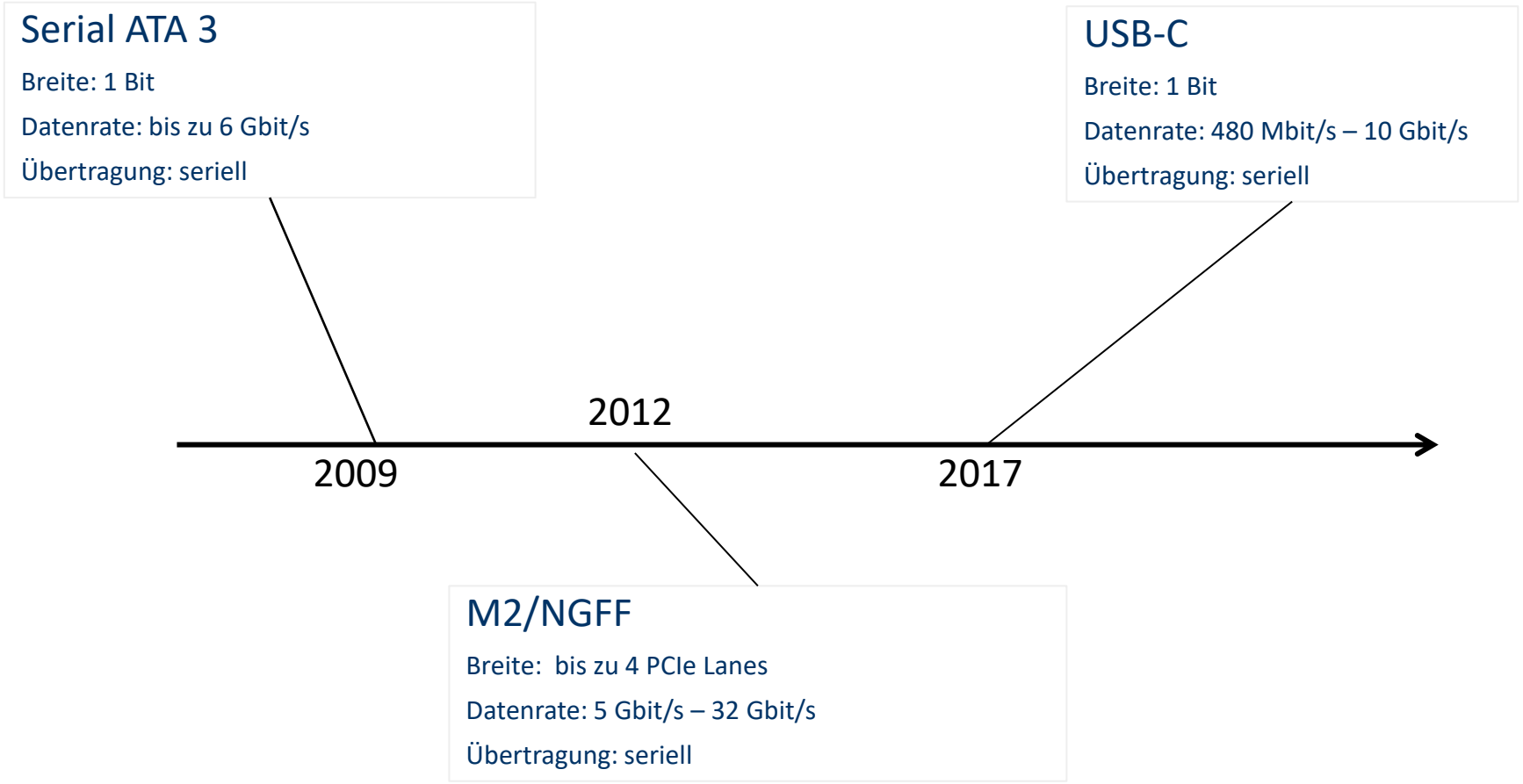

Serial ATA

USB - Universal Serial Bus

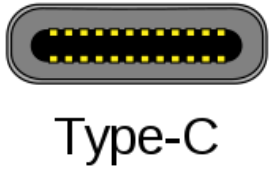
AGP -
Accelerated Graphics Port

PCI-Express
Peripheral Component Interconnect Express

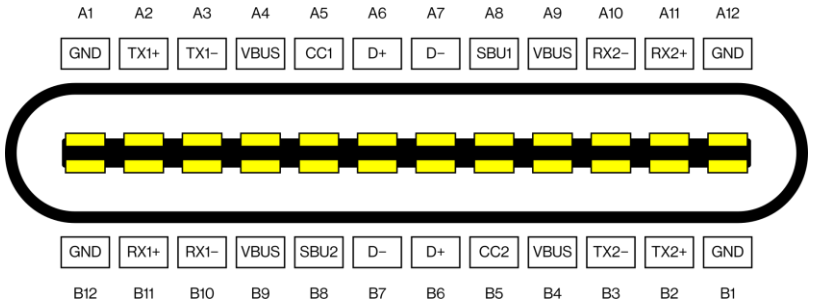
# … und heute


Serial ATA 3
Breite: 1 Bit
Datenrate: bis zu 6 Gbit/s
Übertragung: seriell
USB-C
Breite: 1 Bit
Datenrate: 480 Mbit/s – 10 Gbit/s
Übertragung: seriell
2012
2009
2017
M2/NGFF
Breite: bis zu 4 PCIe Lanes
Datenrate: 5 Gbit/s – 32 Gbit/s
Übertragung: seriell

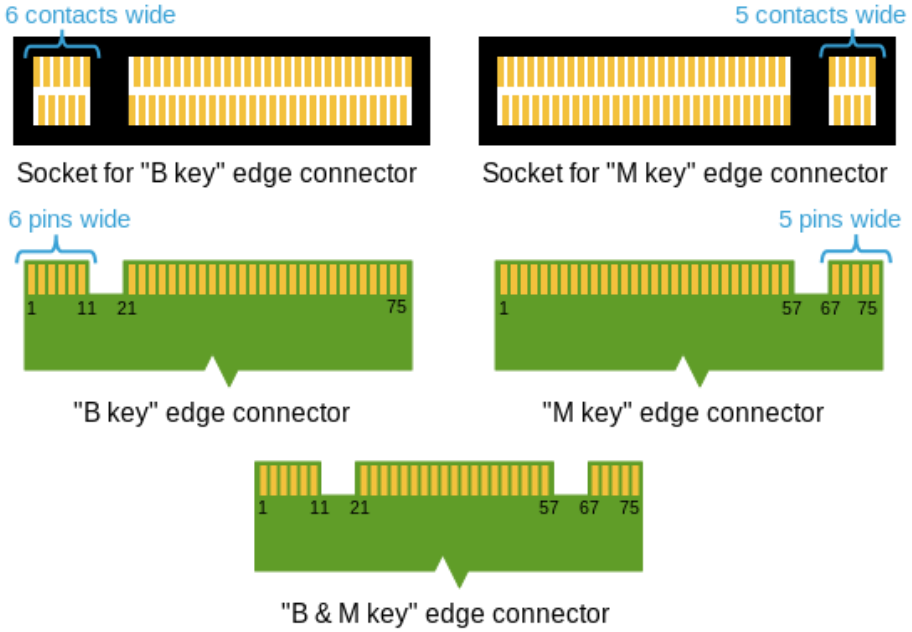
# … und heute



Type-C



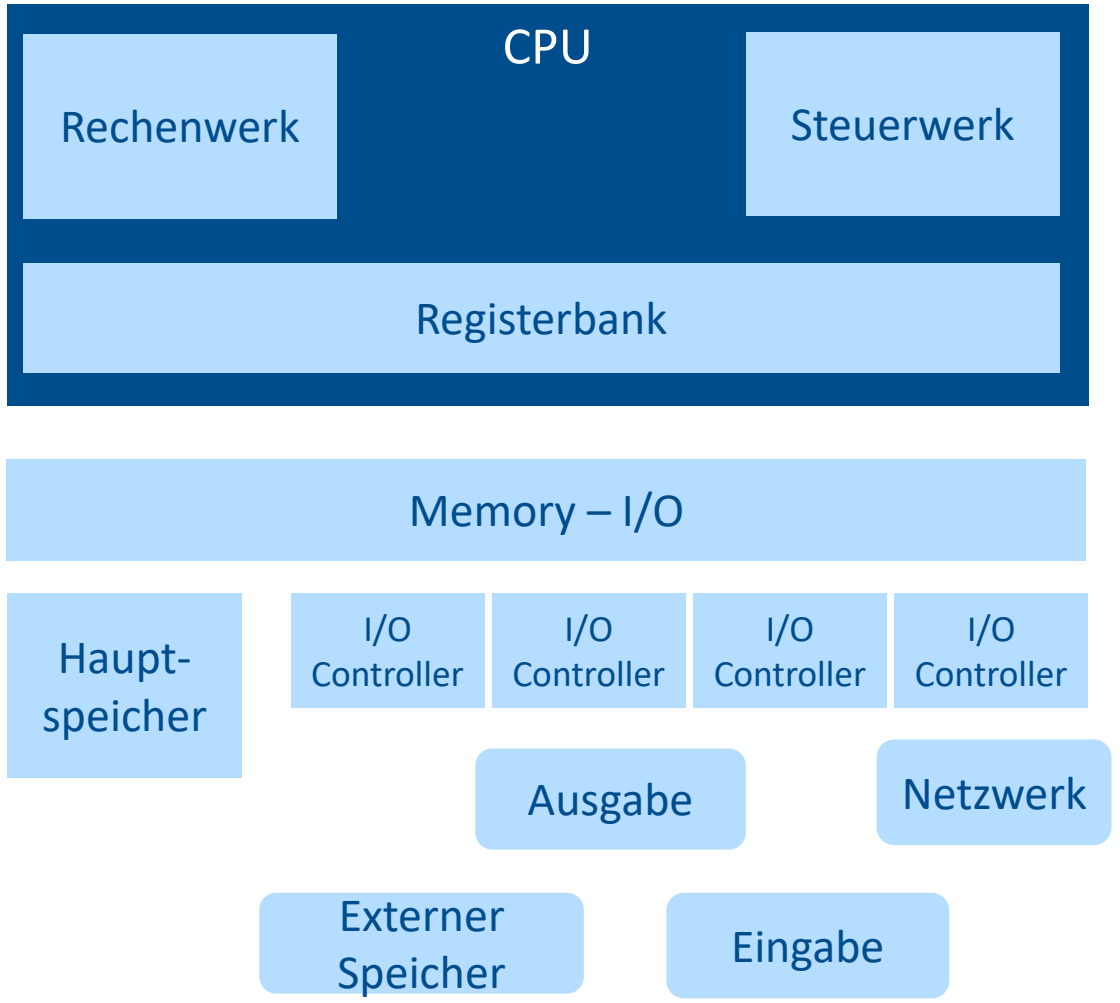

USB –C - Universal Serial Bus




M2/NGFF

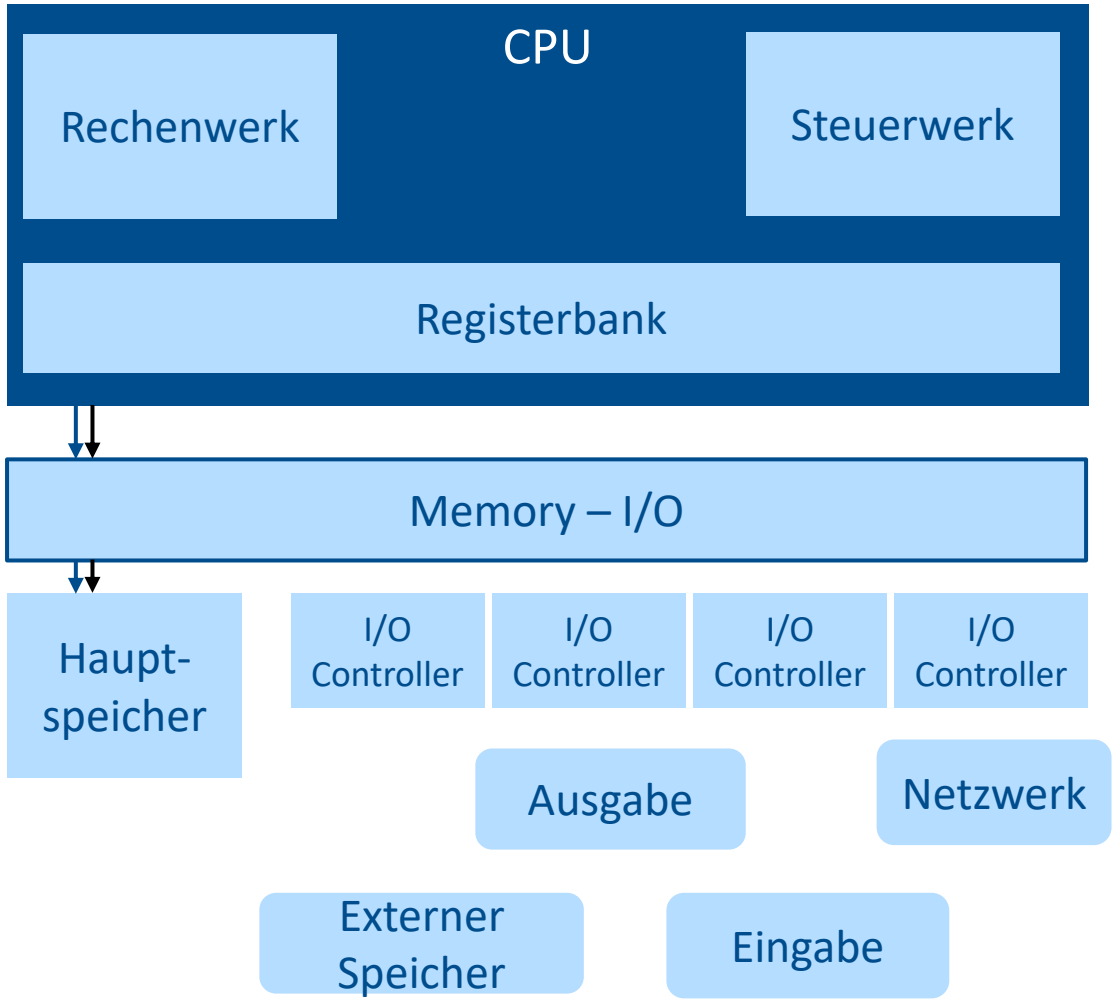
# Prinzipieller Ablauf einer Kommunikation

- Datentransfer vom Hauptspeicher zu einer Festplatte (externer Speicher)



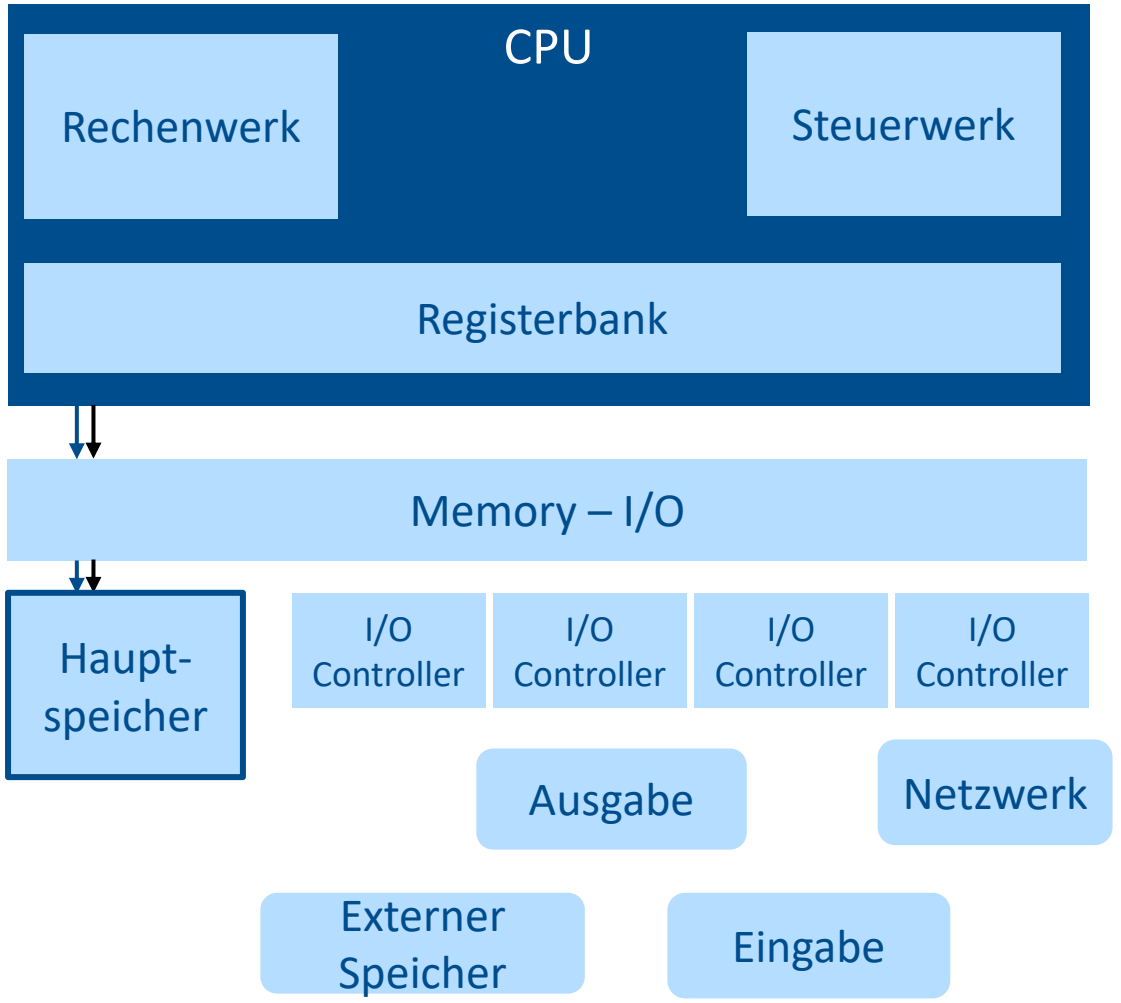
# Prinzipieller Ablauf einer Kommunikation

- Datentransfer vom Hauptspeicher zu einer Festplatte (externer Speicher)
  - Speicher anfragen
    - **Kontrollfluss** (Memory Read Request)
    - **Datenfluss** (Memory Address)



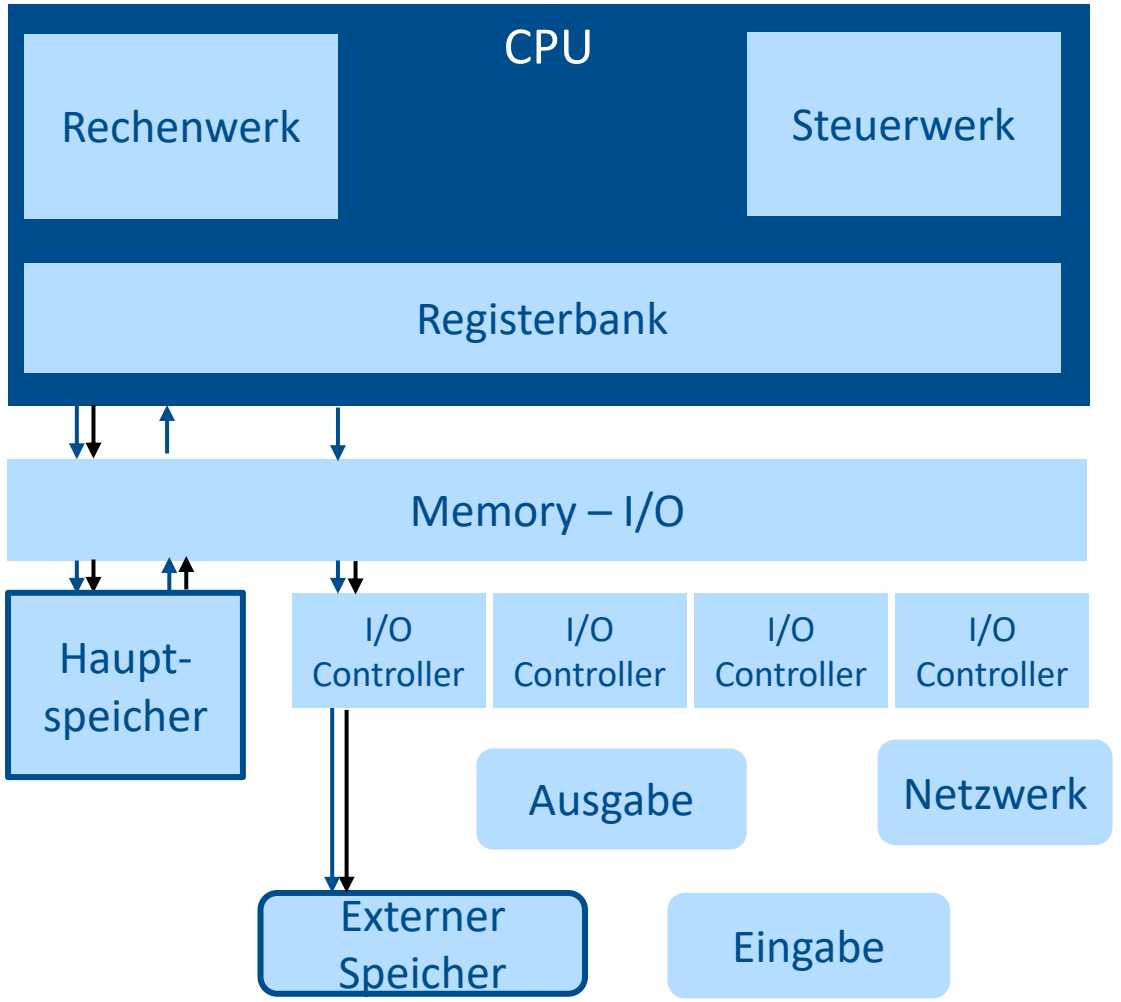
# Prinzipieller Ablauf einer Kommunikation

- Datentransfer vom Hauptspeicher zu einer Festplatte (externer Speicher)
  - Speicher anfragen
    - **Kontrollfluss** (Memory Read Request)
    - **Datenfluss** (Memory Address)
  - Speicher auslesen



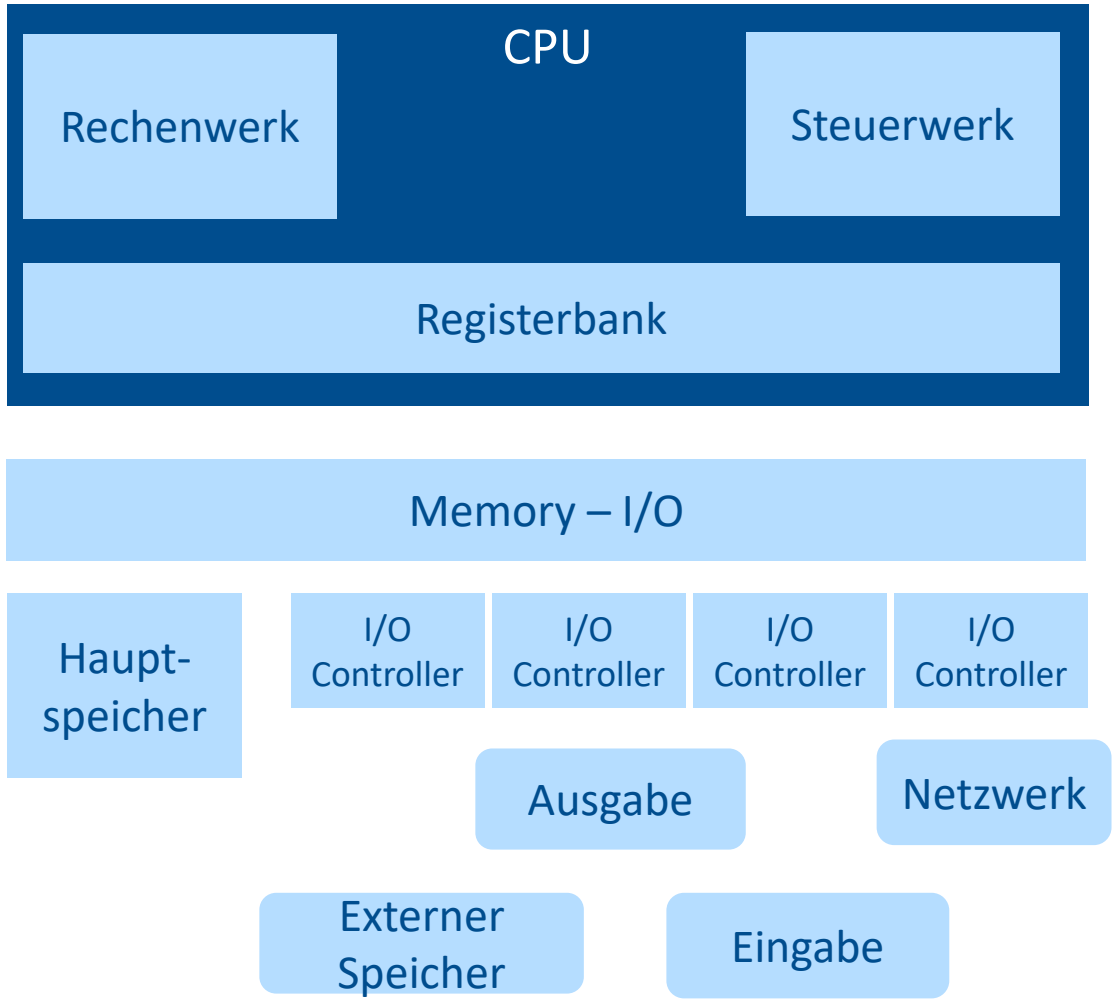
# Prinzipieller Ablauf einer Kommunikation

- Datentransfer vom Hauptspeicher zu einer Festplatte (externer Speicher)
  - Speicher anfragen
    - **Kontrollfluss** (Memory Read Request)
    - **Datenfluss** (Memory Address)
  - Speicher auslesen
  - Daten an das I/O Gerät (Festplatte) senden
    - **Kontrollfluss** (Device Write Request)
    - **Datenfluss** (I/O Device Address and Data)



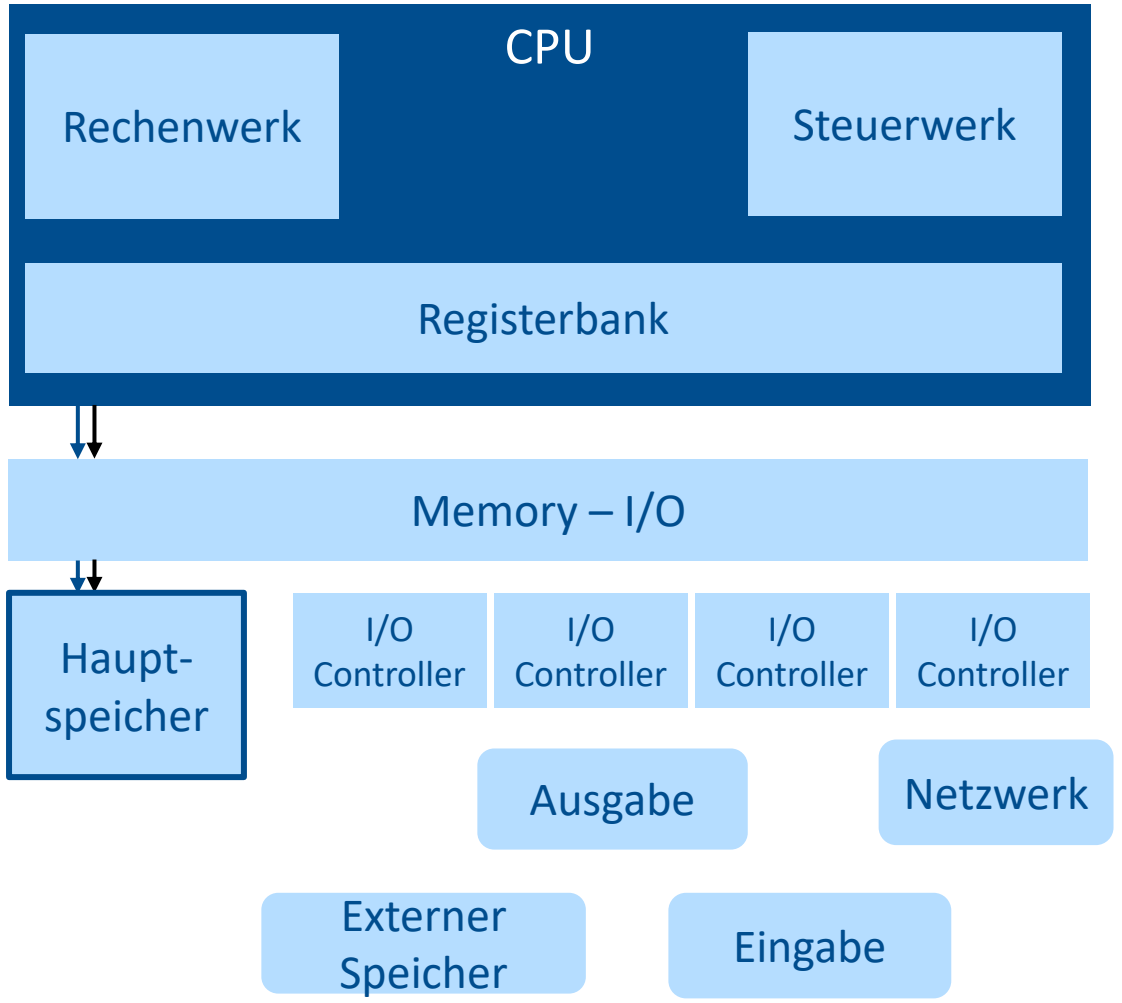
# Prinzipieller Ablauf einer Kommunikation

- Datentransfer von einer Festplatte (externer Speicher) zum Hauptspeicher



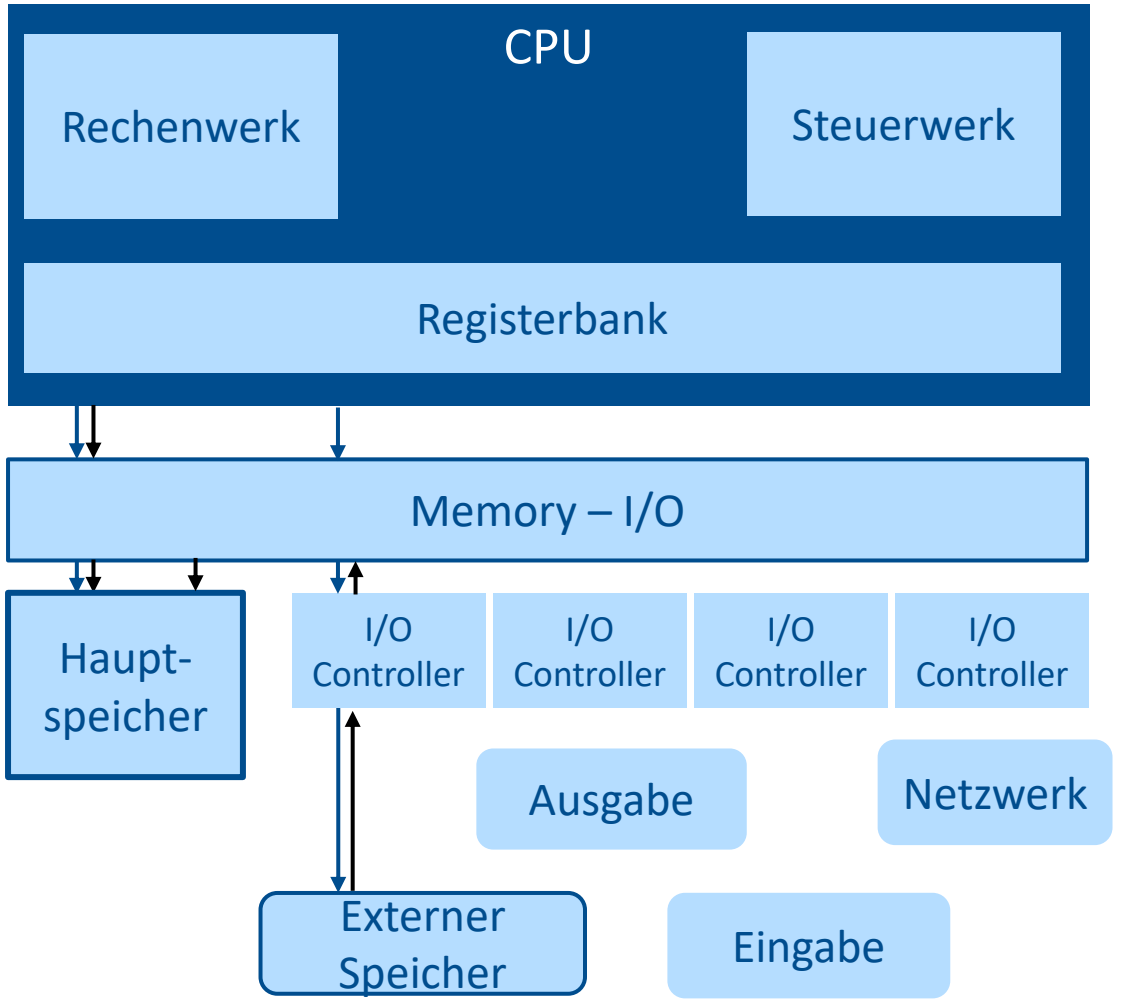
# Prinzipieller Ablauf einer Kommunikation

- Datentransfer von einer Festplatte (externer Speicher) zum Hauptspeicher
  - Speicher anfragen
    - **Kontrollfluss** (Memory Write Request)
    - **Datenfluss** (Memory Address)



# Prinzipieller Ablauf einer Kommunikation

- Datentransfer von einer Festplatte (externer Speicher) zum Hauptspeicher
  - Speicher anfragen
    - **Kontrollfluss** (Memory Write Request)
    - **Datenfluss** (Memory Address)
  - Daten empfangen
    - **Kontrollfluss** (I/O Read Request)
    - **Datenfluss** (I/O Device Address and Data)



# Optionen eines Busses

## Parallele Busse

- Separate Adress- und Datenleitungen sind schneller
- Breiter ist schneller
- Mehrere Busmaster erlaubt (erfordert Schiedsrichter)
- synchrone Übertragung

## Serielle Busse

- Multiplexen von Adress- und Datenleitungen ist billiger
- Schmaler ist billiger
- Nur ein Busmaster (erfordert keinen Schiedsrichter)
- asynchrone Übertragung

# Wie geht es weiter?

- Optimierung des Verarbeitung
  - Pipelining (Kap. 3)
  - Speicherhierarchie, Cache (Kap. 4)
  - Parallelverarbeitung (Kap. 5)
- Sprache zwischen Rechner und Mensch
  - Assembler als letzte menschenlesbare Sprachebene