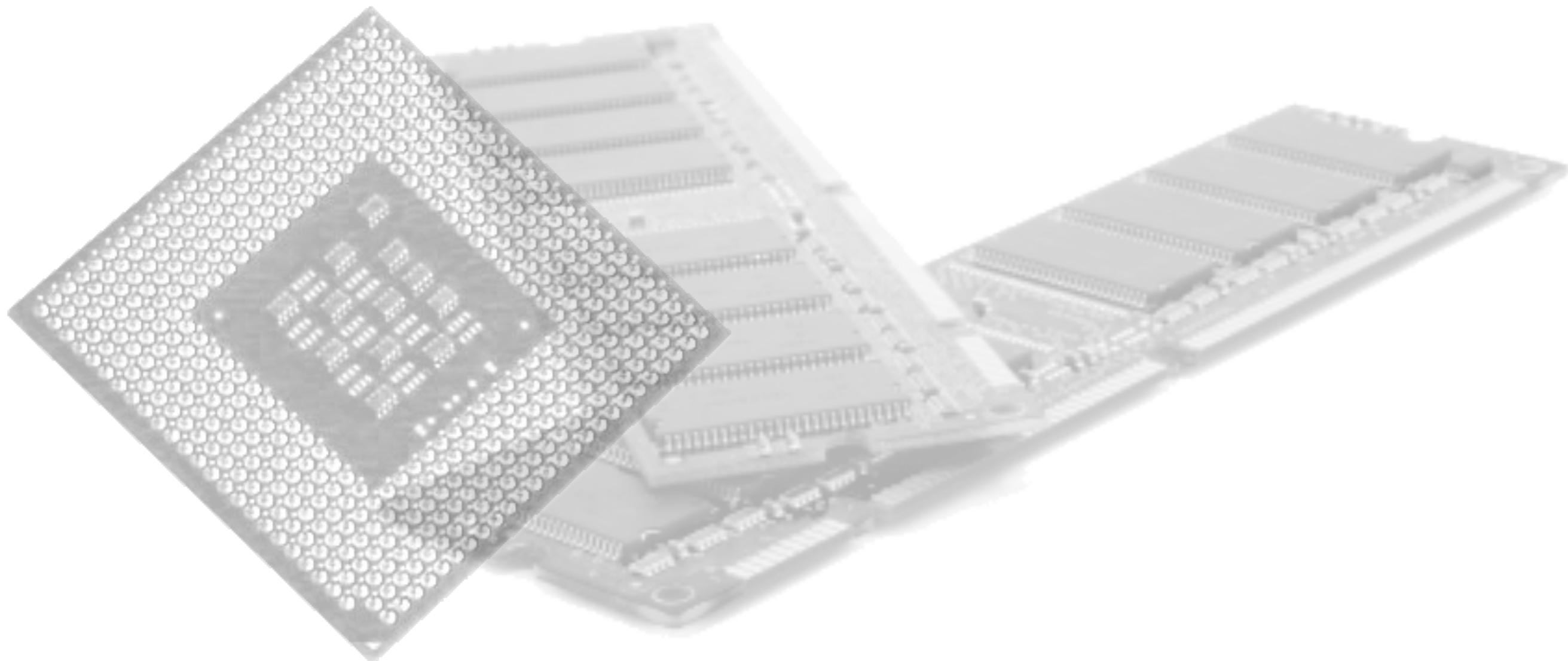
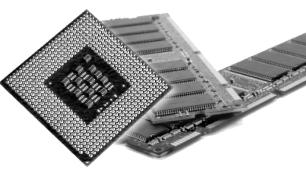


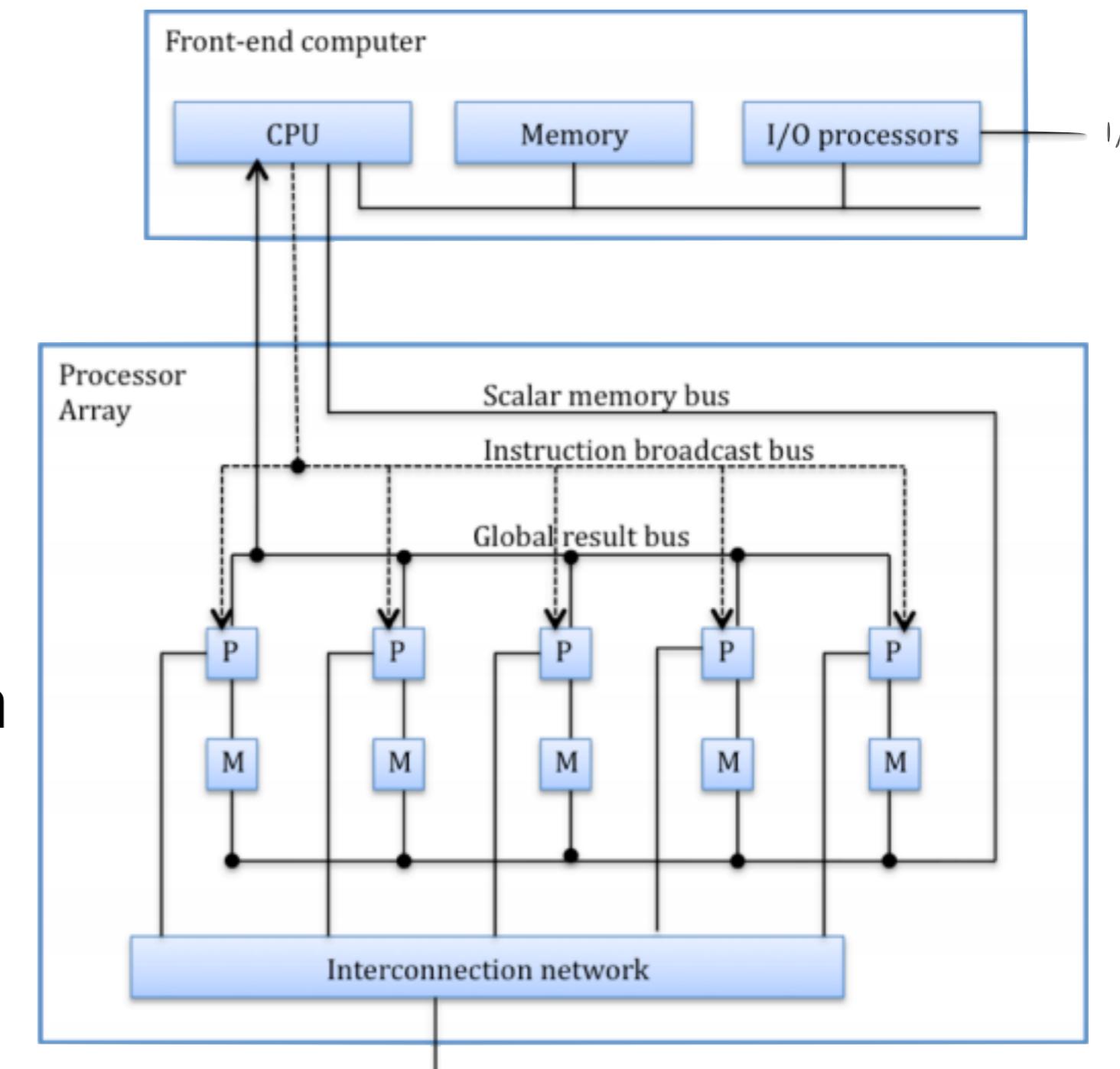
Parallele Rechnerarchitekturen

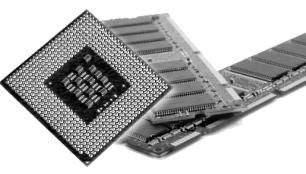




Prozessor Arrays

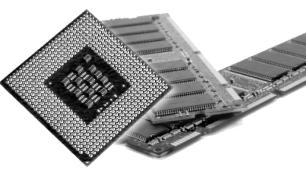
- ein Prozessor (Computer) für die Steuerung
- identische Prozessoren
- die gleiche Operation soll mit mehreren Zahlen gleichzeitig ausgeführt werden





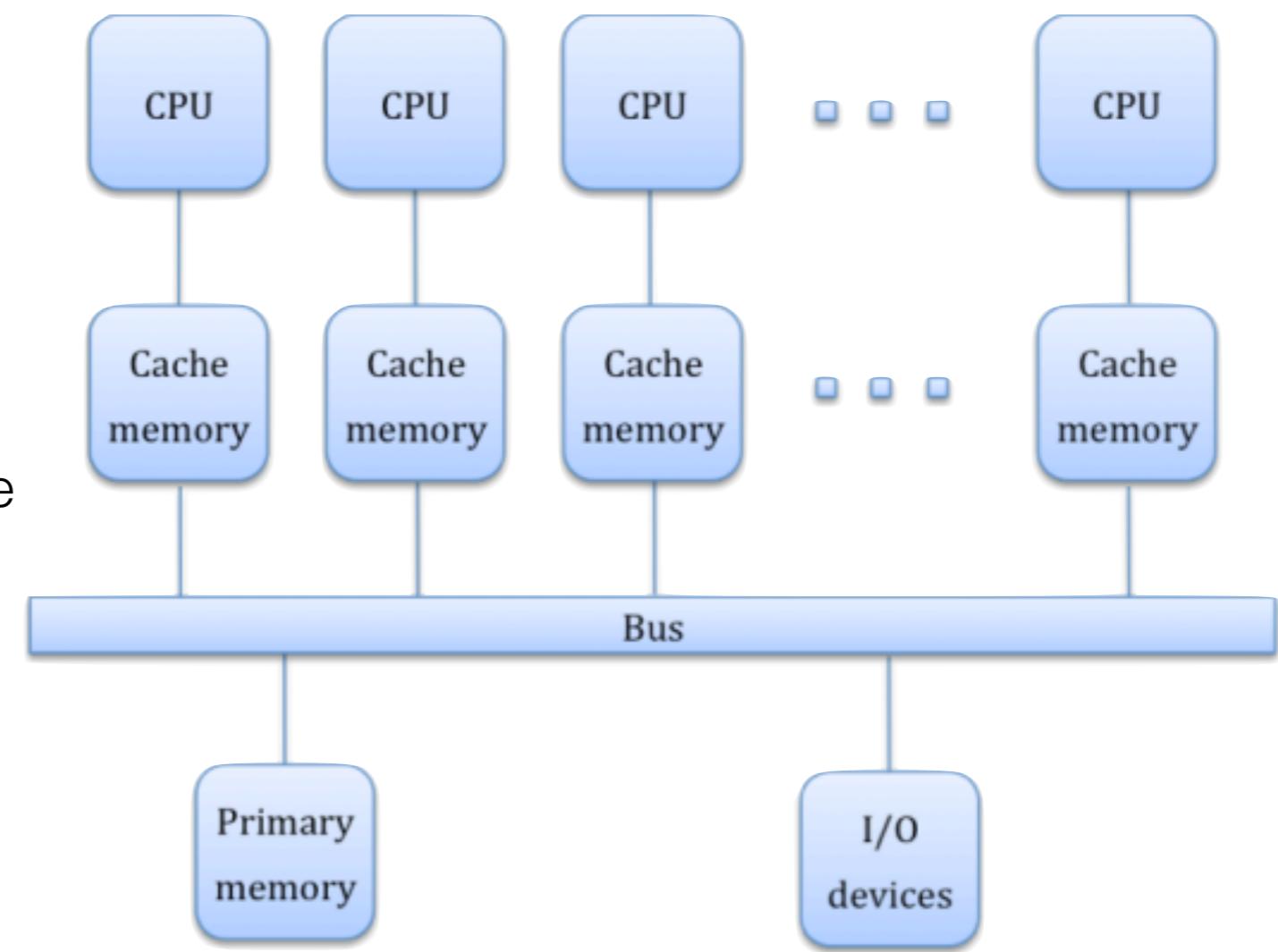
Multiprozessoren

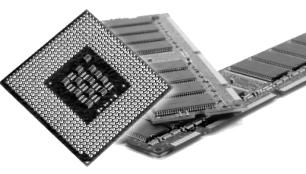
- Mehrere Prozessoren teilen sich einen Speicher
- eine Speicheradresse zeigt immer auf den gleichen Speicherplatz, egal welcher Prozessor sie anspricht
- 2 verschiedene Arten von Multiprozessoren:
 - zentraler Multiprozessor
 - verteilter Multiprozessor



zentraler Multiprozessor (UMA)

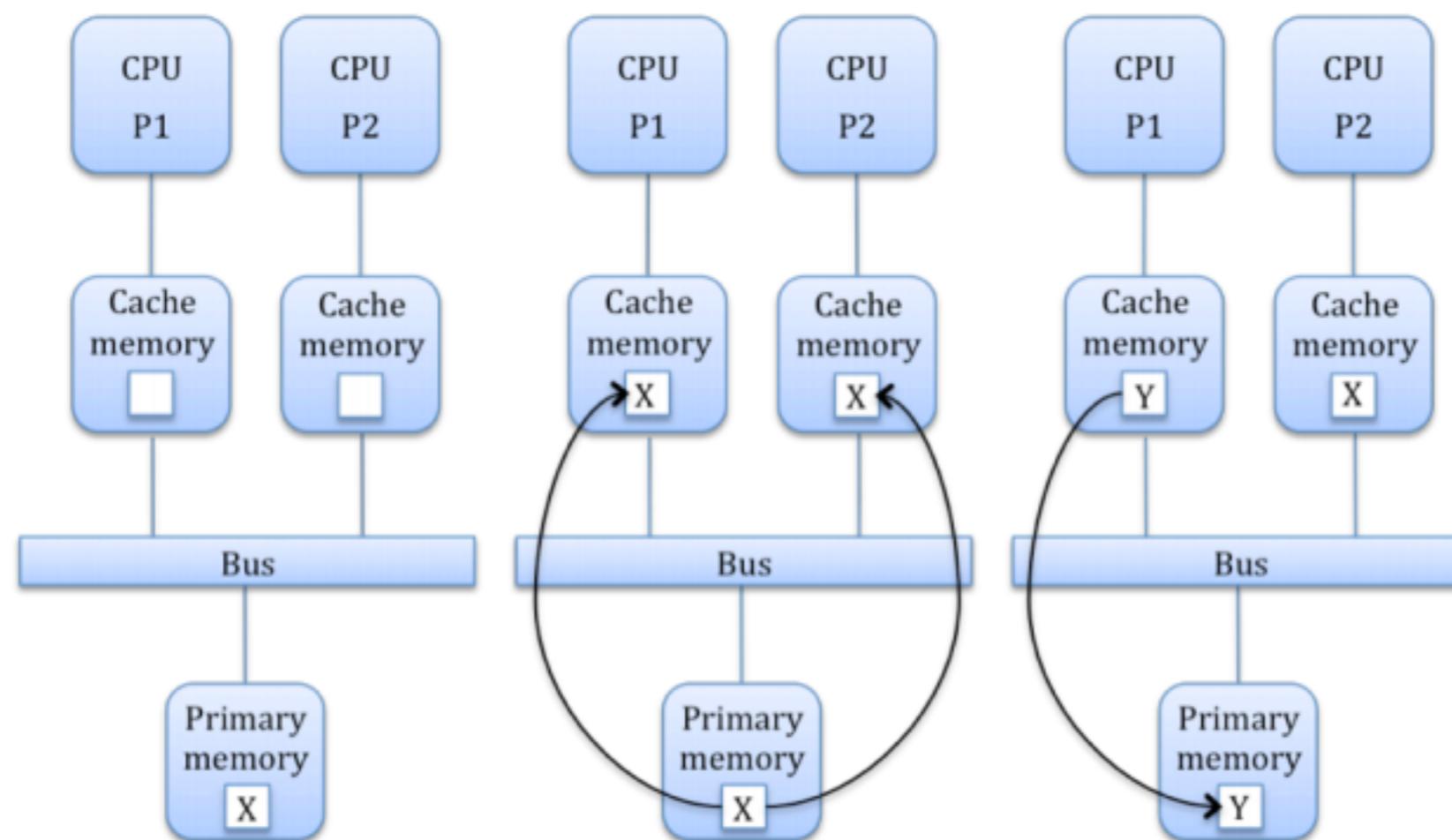
- mehrere CPUs sind an einen Hauptspeicher angeschlossen
- uniform memory access (UMA)
- oder Symmetric multiprocessing (SMP)
- keine Netzwerkverbindung wird zwischen den CPUs benötigt, da alle über einen Bus auf den Speicher zugreifen
- private data: nur ein Prozessor benutzt die Daten
- shared data: alle Prozessoren arbeiten mit den Daten

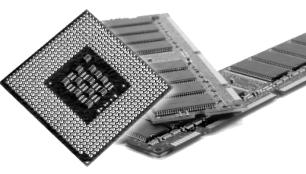




zentraler Multiprozessor (UMA)

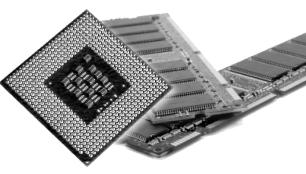
- Problem wenn mehrere Prozessoren mit den gleichen Daten arbeiten:





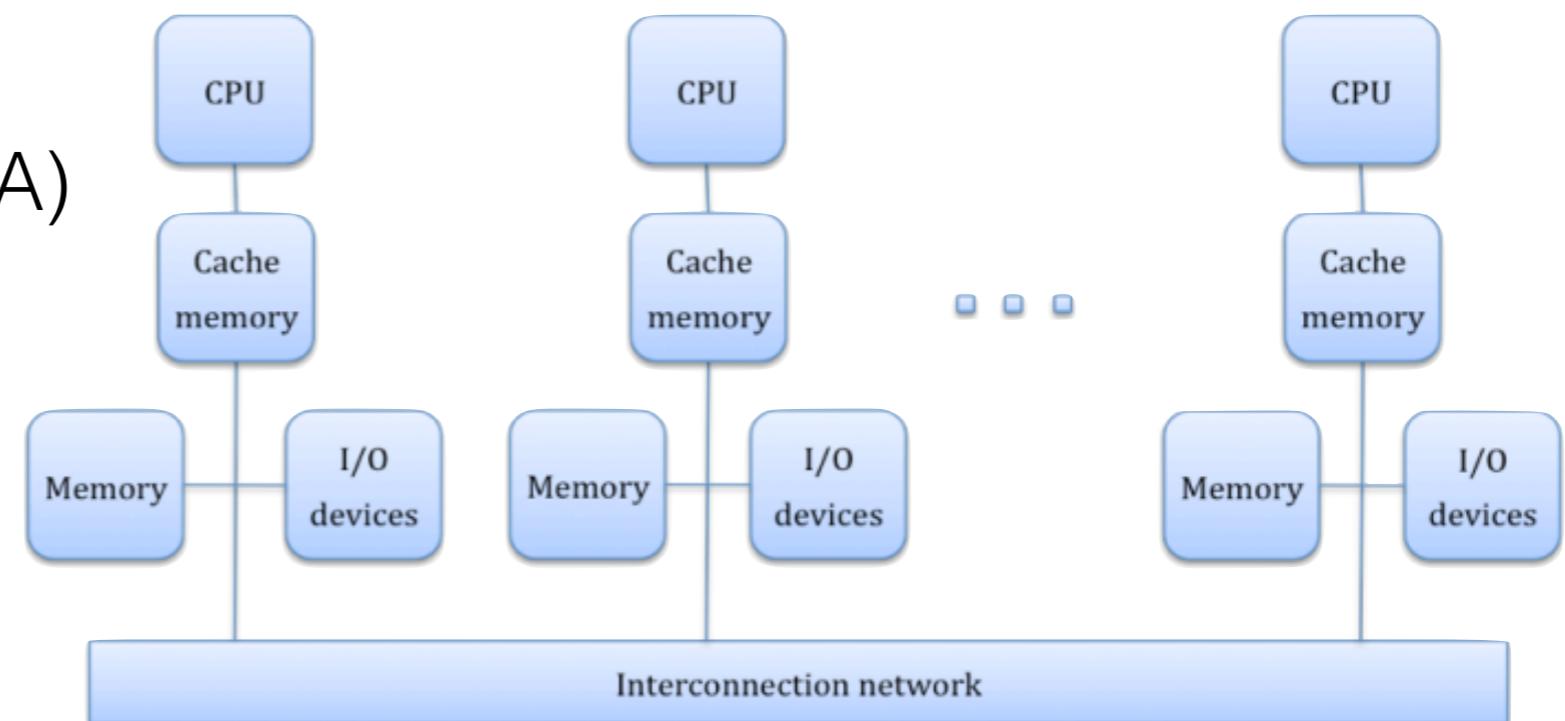
zentraler Multiprozessor (UMA)

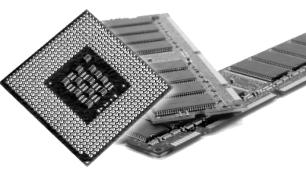
- Lösungsmöglichkeit: Informationen über den “Zustand” der Daten (MESI)
 - **M**odified
 - die Daten wurden verändert und müssen erst wieder von der Cache in den Speicher kopiert werden
 - **E**xclusive
 - die Daten wurden nicht verändert und es ist die einzige Kopie
 - **S**hared
 - Mehr als eine Kopie der Daten ist vorhanden, aber keine bisher verändert
 - **I**nvalid
 - die Kopie ist ungültig



verteilter Multiprozessor (NUMA)

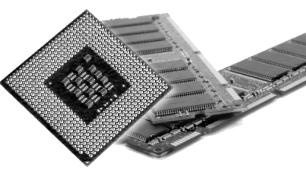
- verteilter Hauptspeicher (jeder Prozessor hat seinen eigenen)
- non-uniform memory access (NUMA)
- schnellerer Speicherzugriff
- Probleme:
 - Kommunikation für den Datenaustausch
 - Es wird schwierig, wenn mehrere Prozessoren mit den gleichen Daten arbeiten sollen





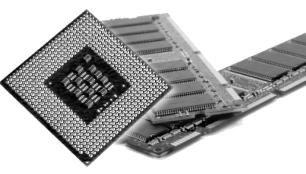
Multicomputer

- Jeder Prozessor hat seinen eigenen Speicher (lokaler Speicher)
- andere Daten müssen über ein Netzwerk verteilt werden
- auch wieder 2 verschiedene Arten:
 - asymmetrischer Multicomputer
 - symmetrischer Multicomputer

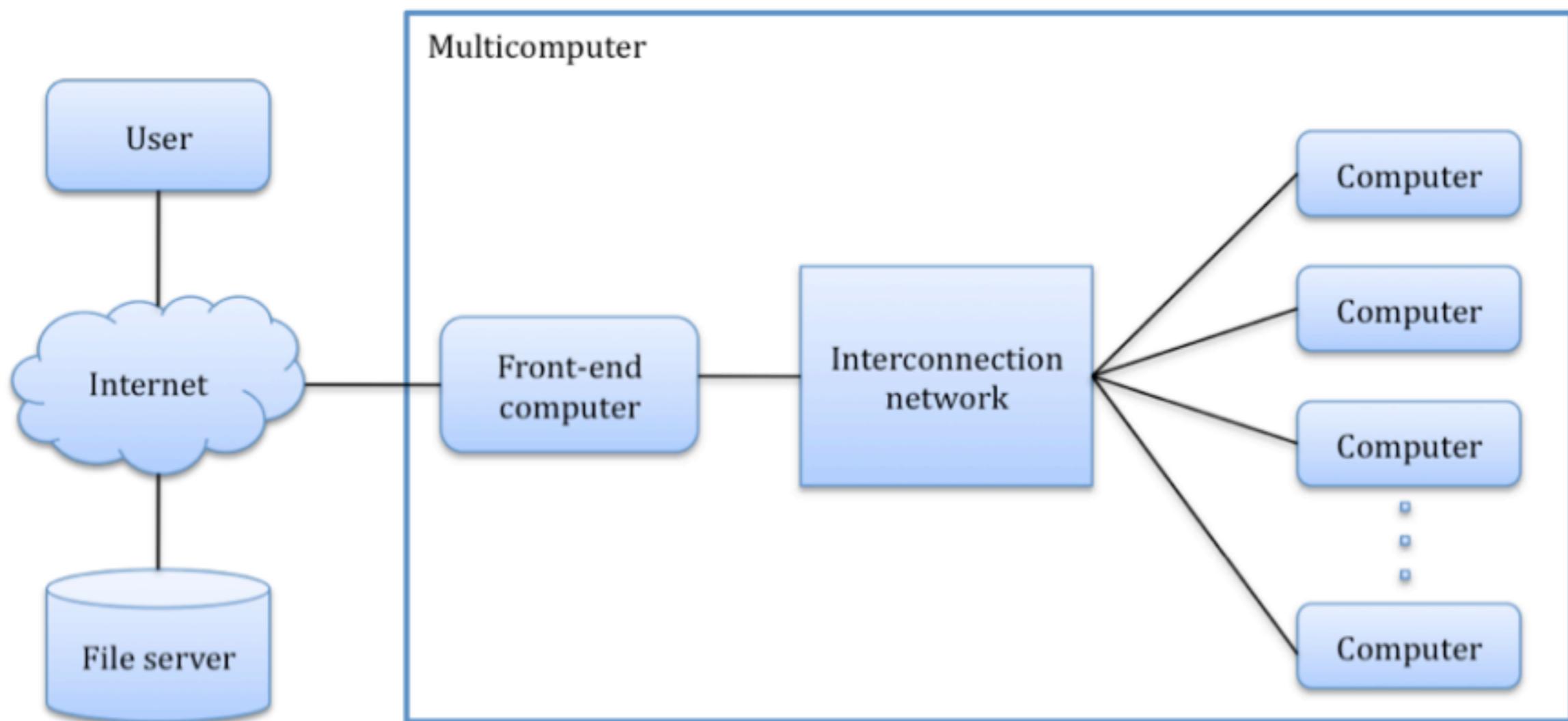


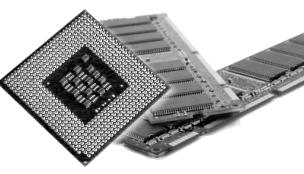
asymmetrischer Multicomputer

- Zugriff auf die vernetzen Computer über einen **Front-end Computer**
- Front-end Computer benötigt die vollen Funktionen und ein vollwertiges Betriebssystem
- die anderen Computer müssen nur die Befehle ausführen können
 - einfaches Betriebssystem
 - kein Bildschirm + I/O Verbindungen



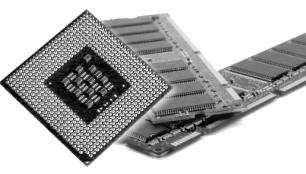
asymmetrischer Multicomputer



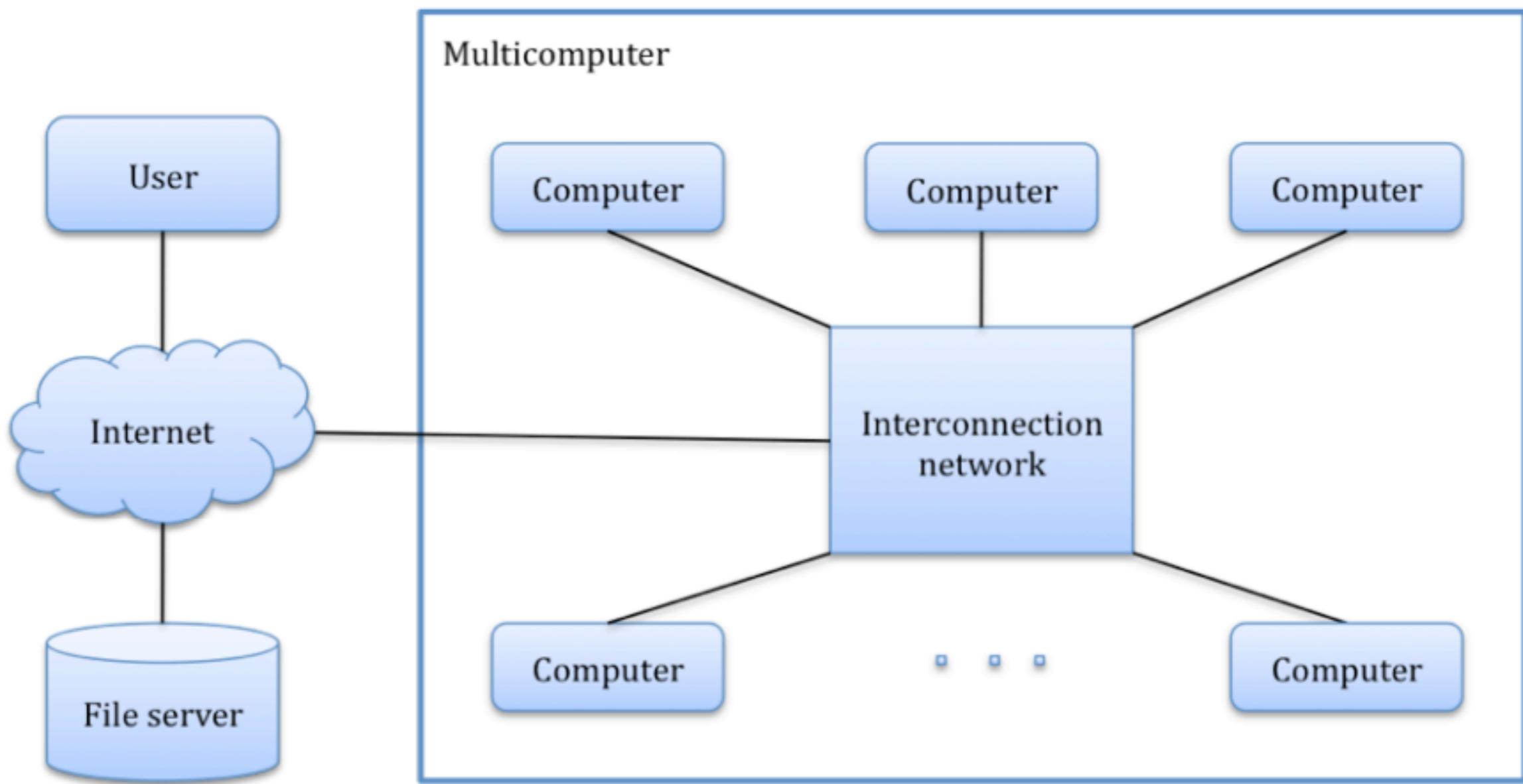


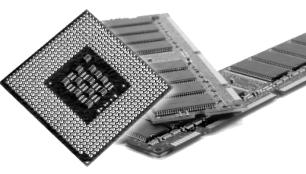
symmetrischer Multicomputer

- alle Computer haben die gleiche Funktionalität
- von jedem Computer aus kann das komplette System bedient werden
- die meisten Supercomputer funktionieren nach diesem Prinzip



symmetrischer Multicomputer





Vergleich Multicomputer

asymmetrischer Multicomputer

Back-end Computer sind einfacher
(d.h. auch kostengünstiger)

wenn der Front-end Computer
ausfällt, ist das ganze System offline

der Userzugriff muss immer über
den Front-end Computer erfolgen,
es kann dadurch schwerer zu
debuggen sein

der Front-end Computer limitiert ggf. die volle Geschwindigkeit von jedem
Computer kann benutzt werden

einfach zum Aufbauen und zu
Warten

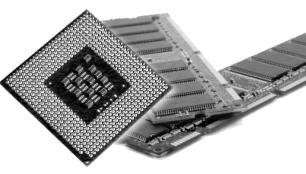
symmetrischer Multicomputer

alle Computer haben die gleiche
Funktionalität

wenn ein Computer ausfällt, dann ist
das System nur etwas langsamer

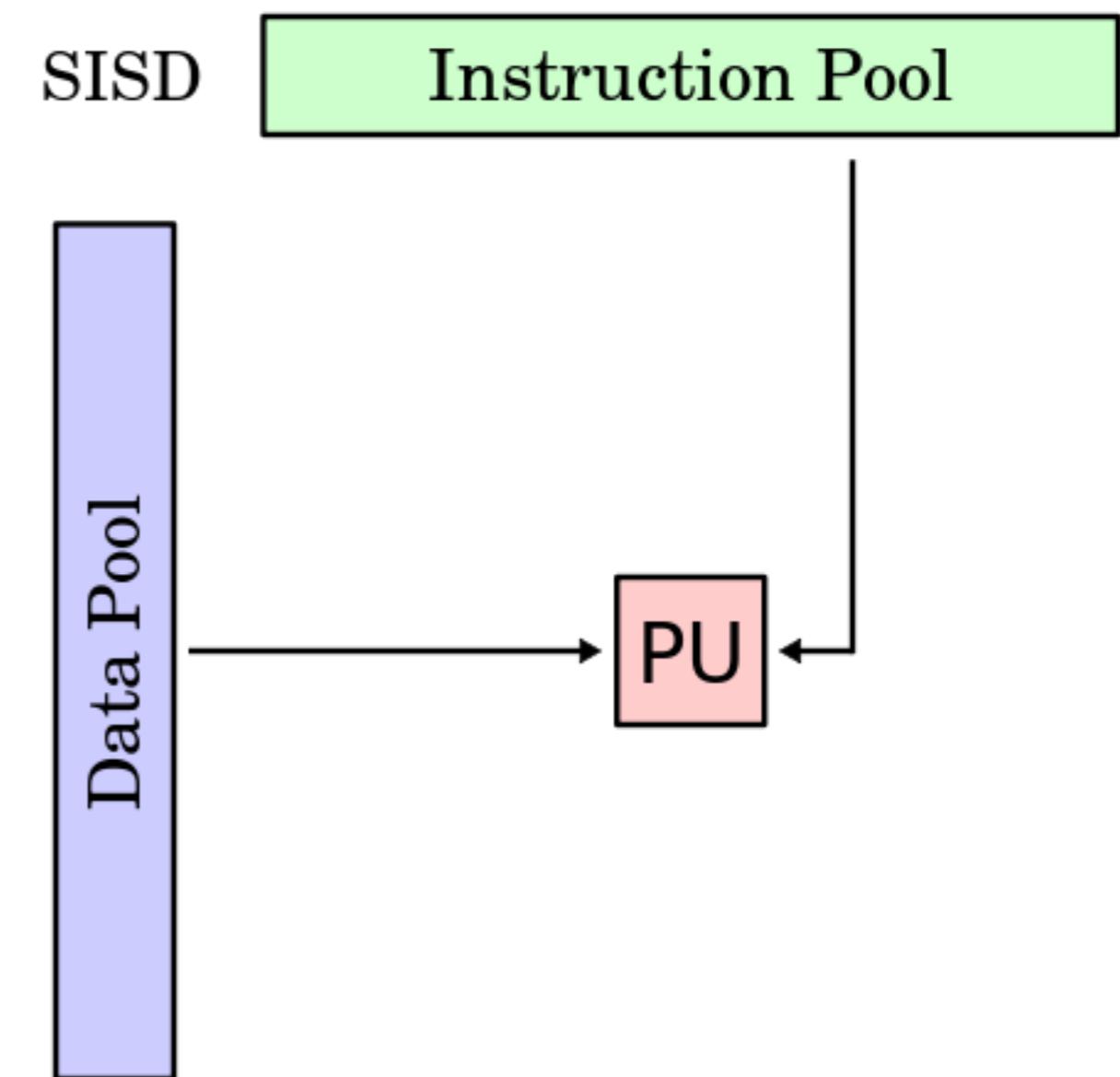
mit jedem Computer kann der User
auf das System zugreifen

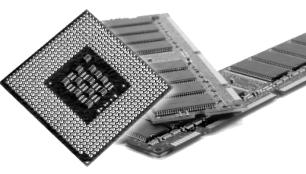
aufwändiges System:
Arbeitsverteilung, usw.



SISD

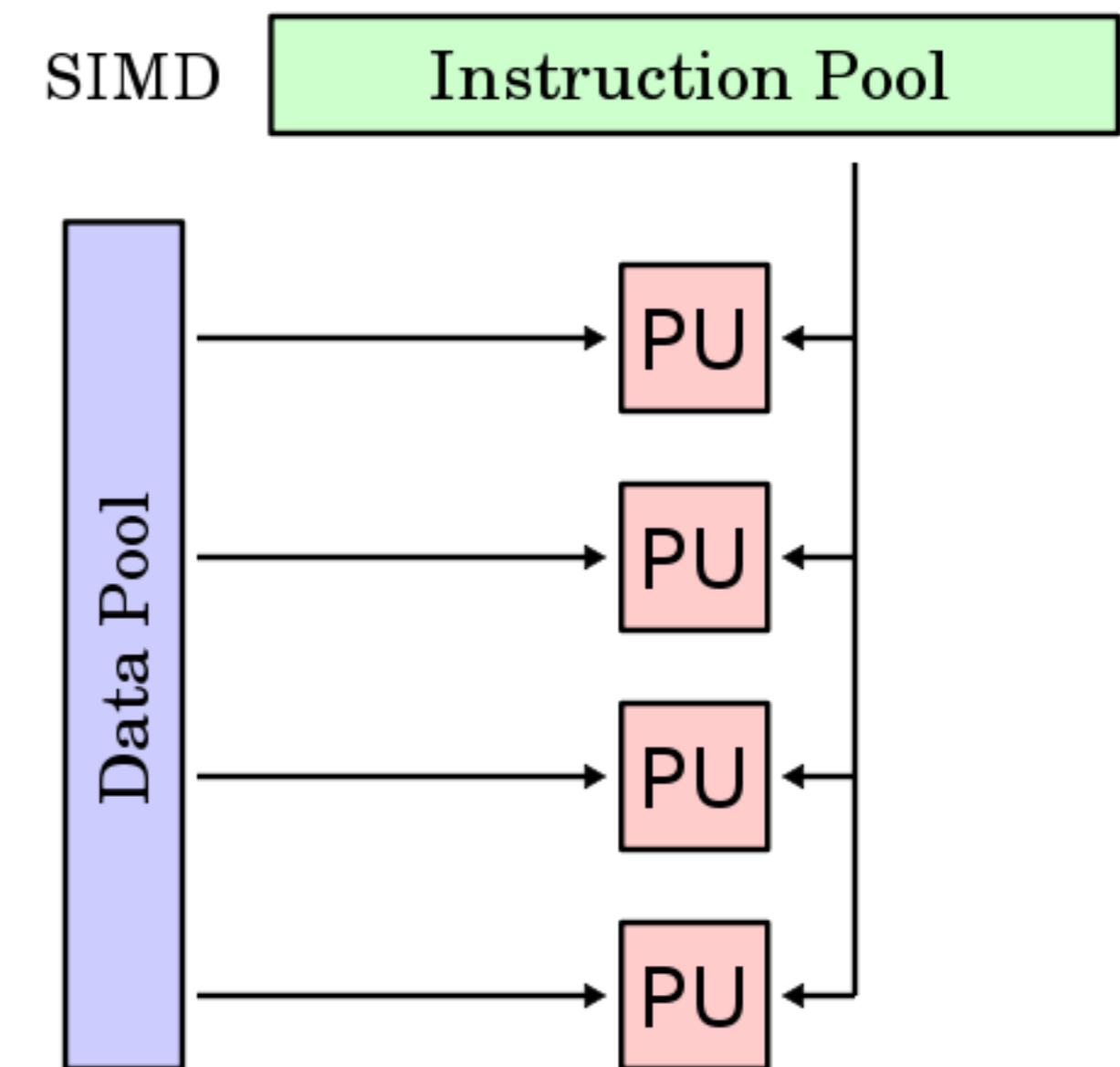
- single instruction, single data
- immer nur ein Befehl kann ausgeführt werden (keine parallele Bearbeitung)
- nur ein Datenzugriff pro Befehl
- klassische von-Neumann Architektur

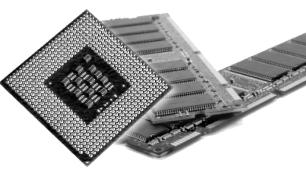




SIMD

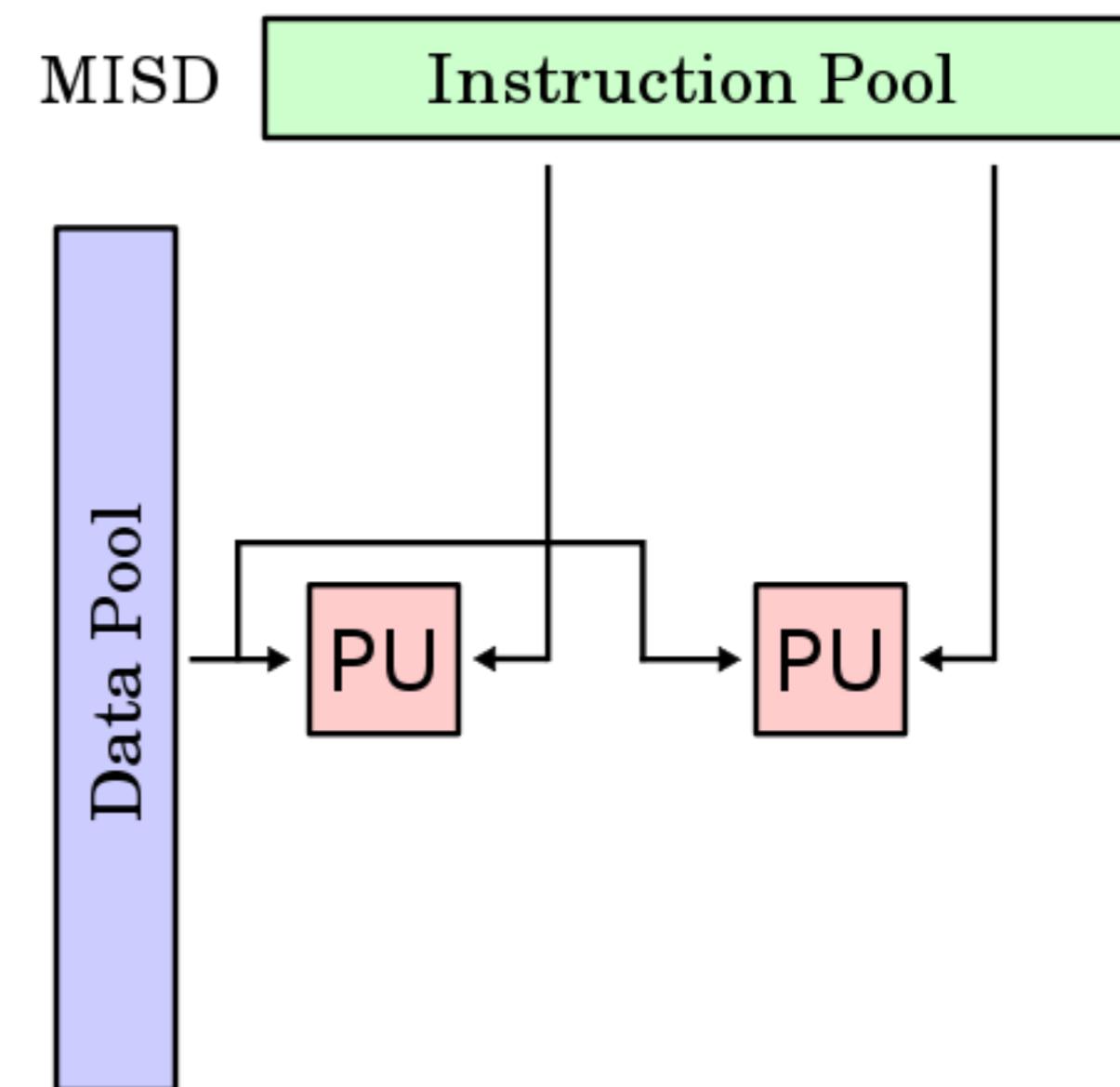
- single instruction, multiple data
- Prozessor Arrays
- nur ein Befehl kann ausgeführt werden, allerdings mit mehreren (unterschiedlichen) Daten
- war für Supercomputer sehr verbreitet, ist aber inzwischen veraltet

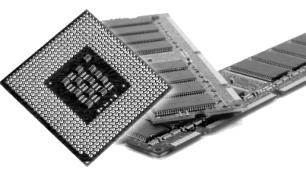




MISD

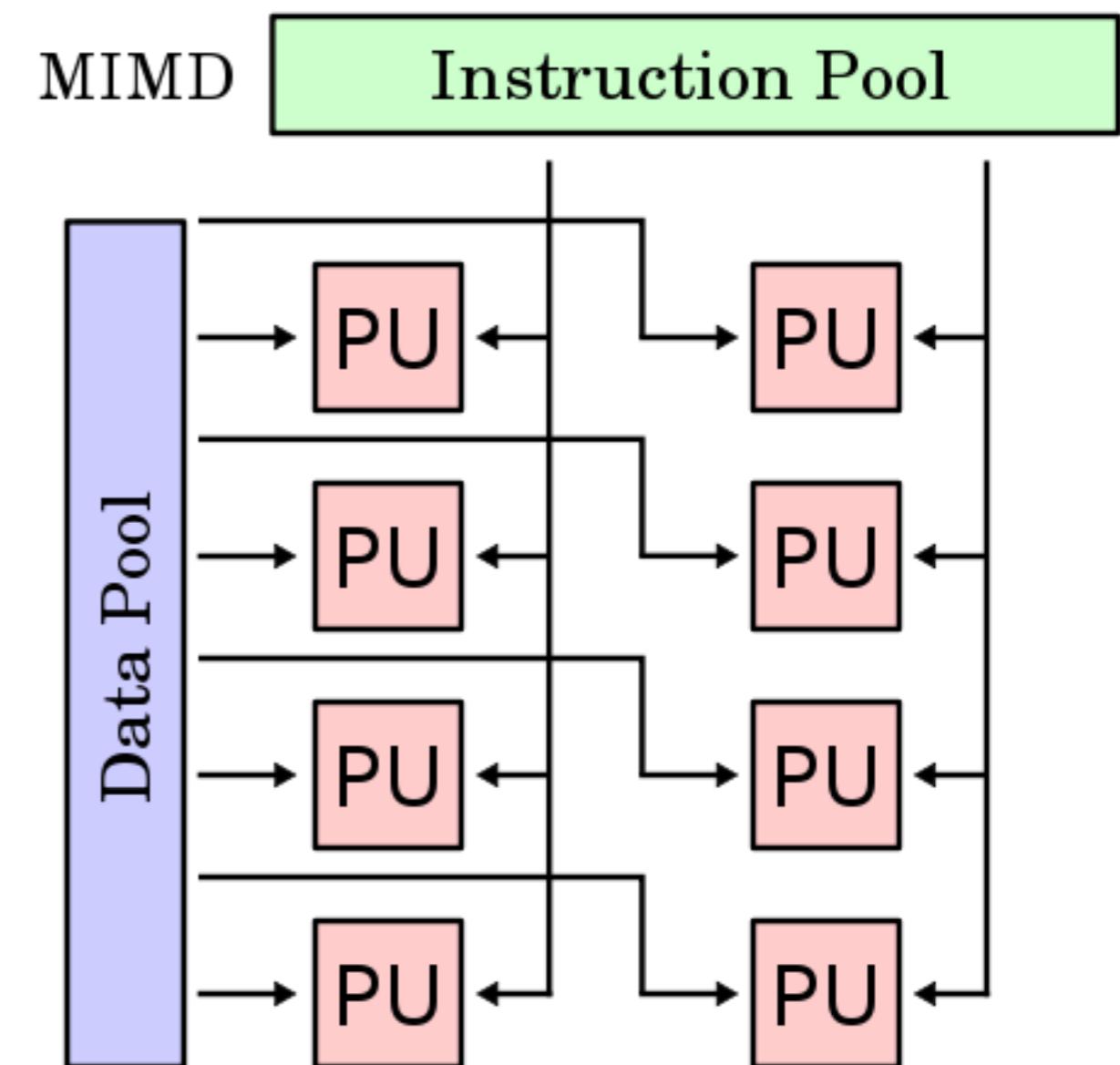
- multiple instruction, single data
- mit den gleichen Daten werden mehrere Befehle gleichzeitig ausgeführt
- kommt in der Praxis fast nie zum Einsatz

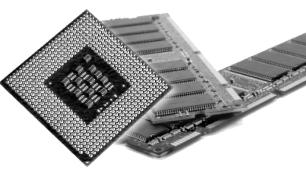




MIMD

- multiple instruction,
multiple data
- paralleles Ausführen von
mehreren Befehlen mit
unterschiedlichen Daten
- UMA und NUMA fallen in
die Kategorie





Parallele Rechnerarchitekturen

- Moderne Architekturen gehören in der Regel zu den MIMD Systemen
- Mehr Leistung durch Parallelle Berechnungen

