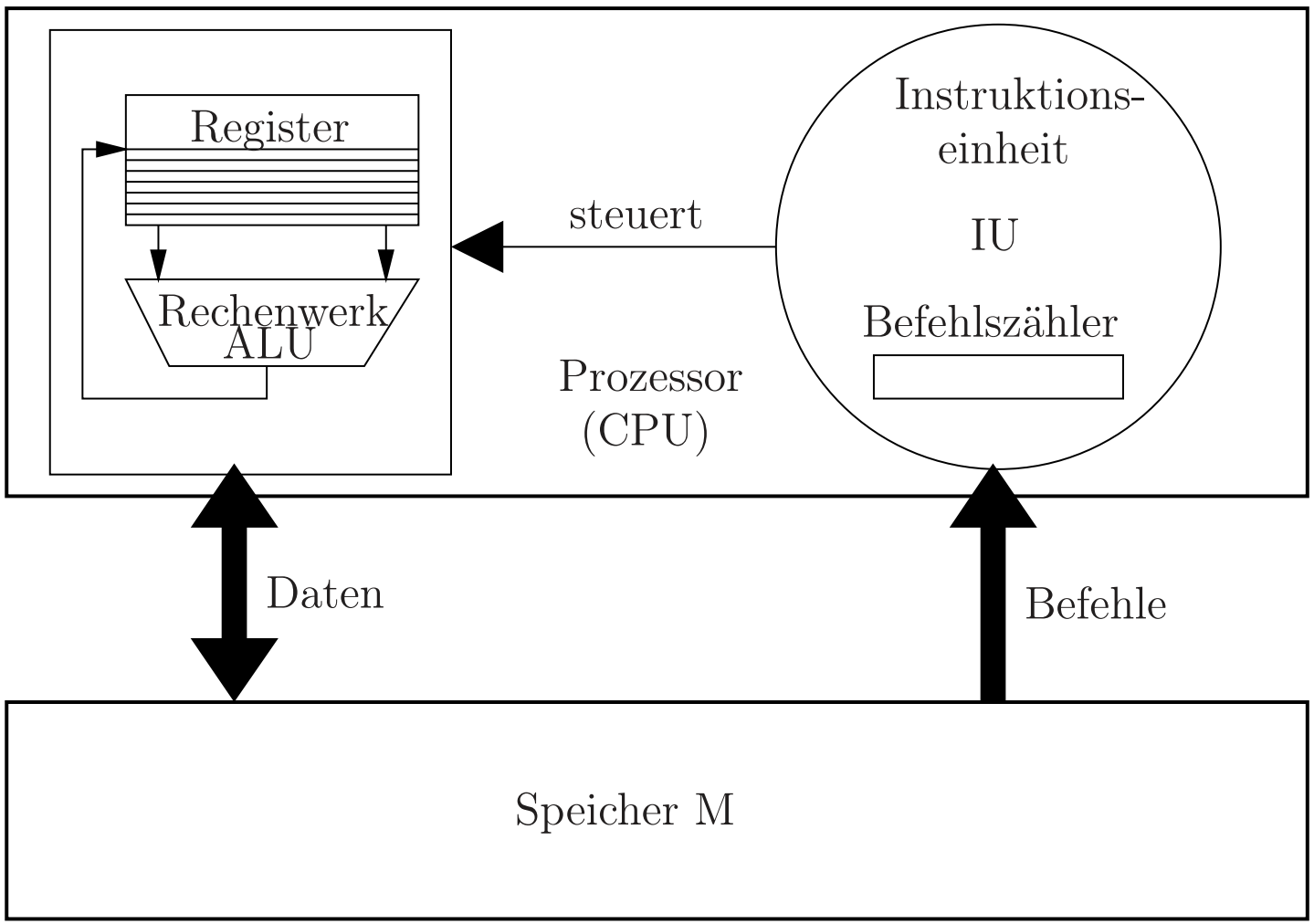
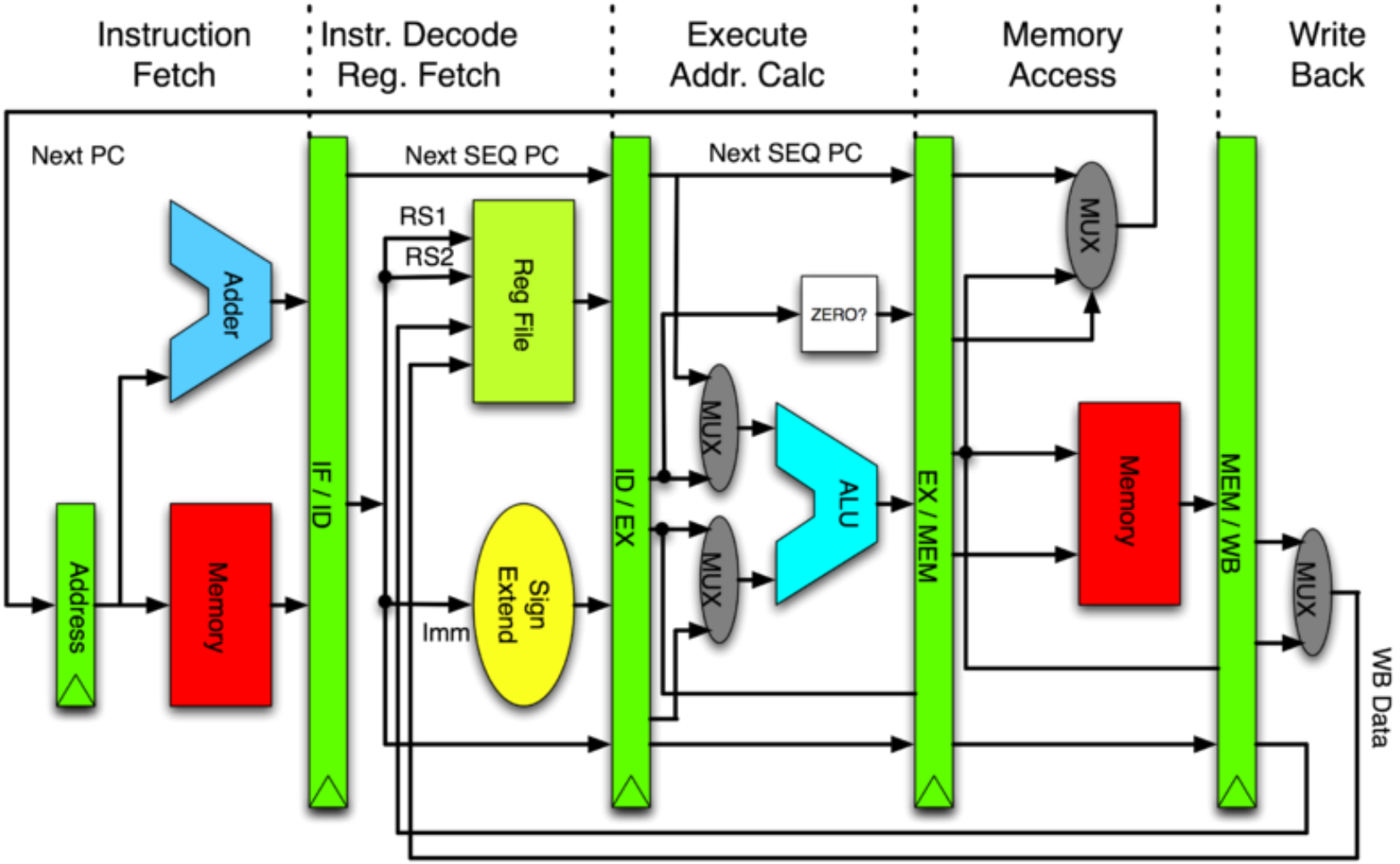
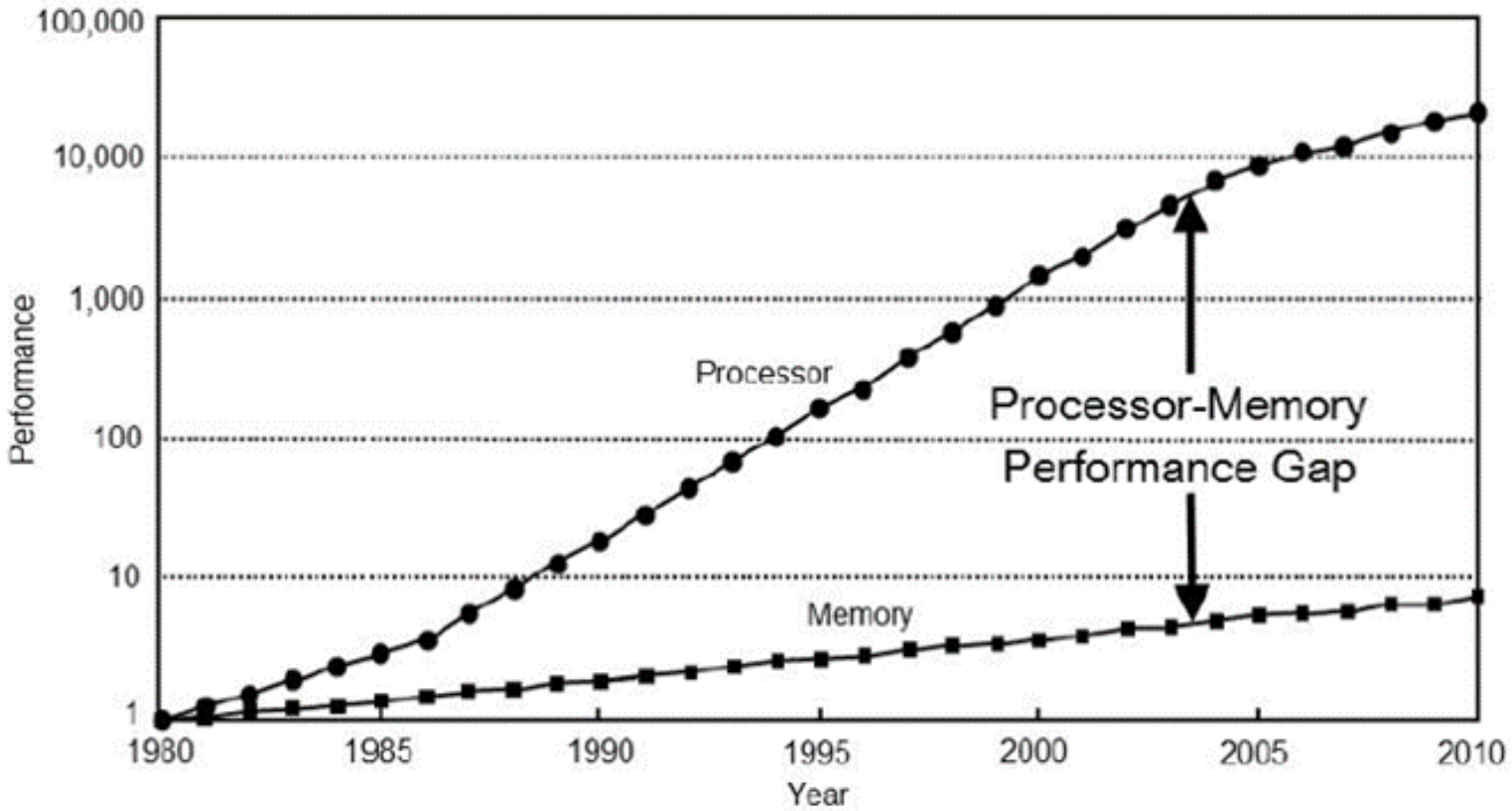
Von-Neumann Rechner



Aufbau des MIPS Prozessors

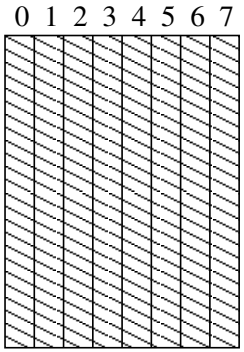


Memory gap

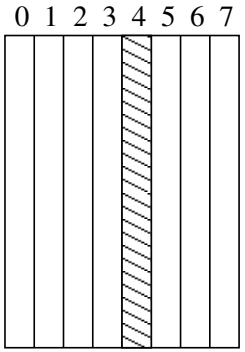


Einstufiger Cache – Platzierung

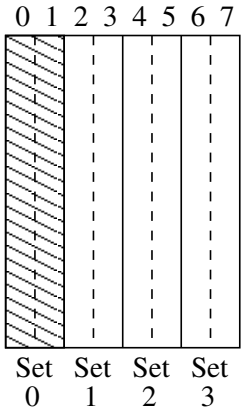
Voll assoziativ:
Block 12 kann in
jeden Cache−−Block



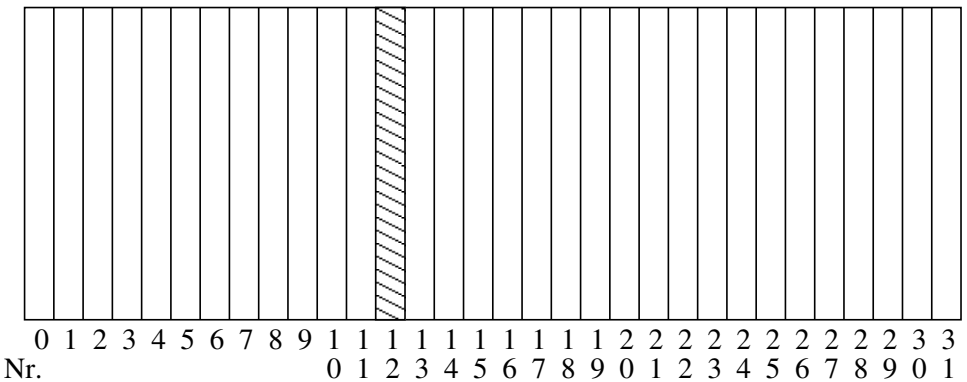
Direkte Zuordnung:
Block 12 kann nur in
Cache−−Block 4
(12 mod 8)



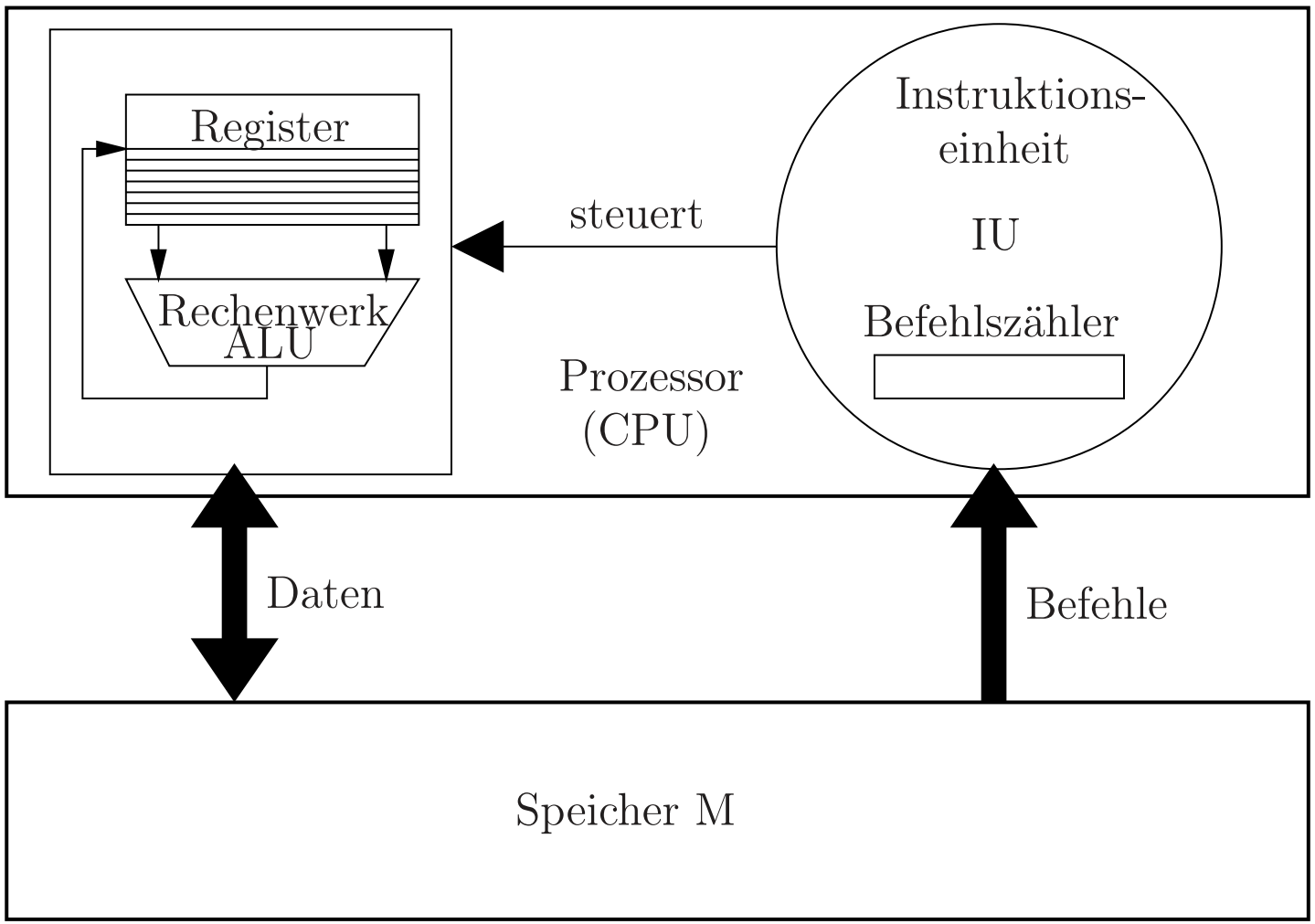
2−−fach assoziativ:
Block 12 geht in
Menge 0 (12 \mod 4)



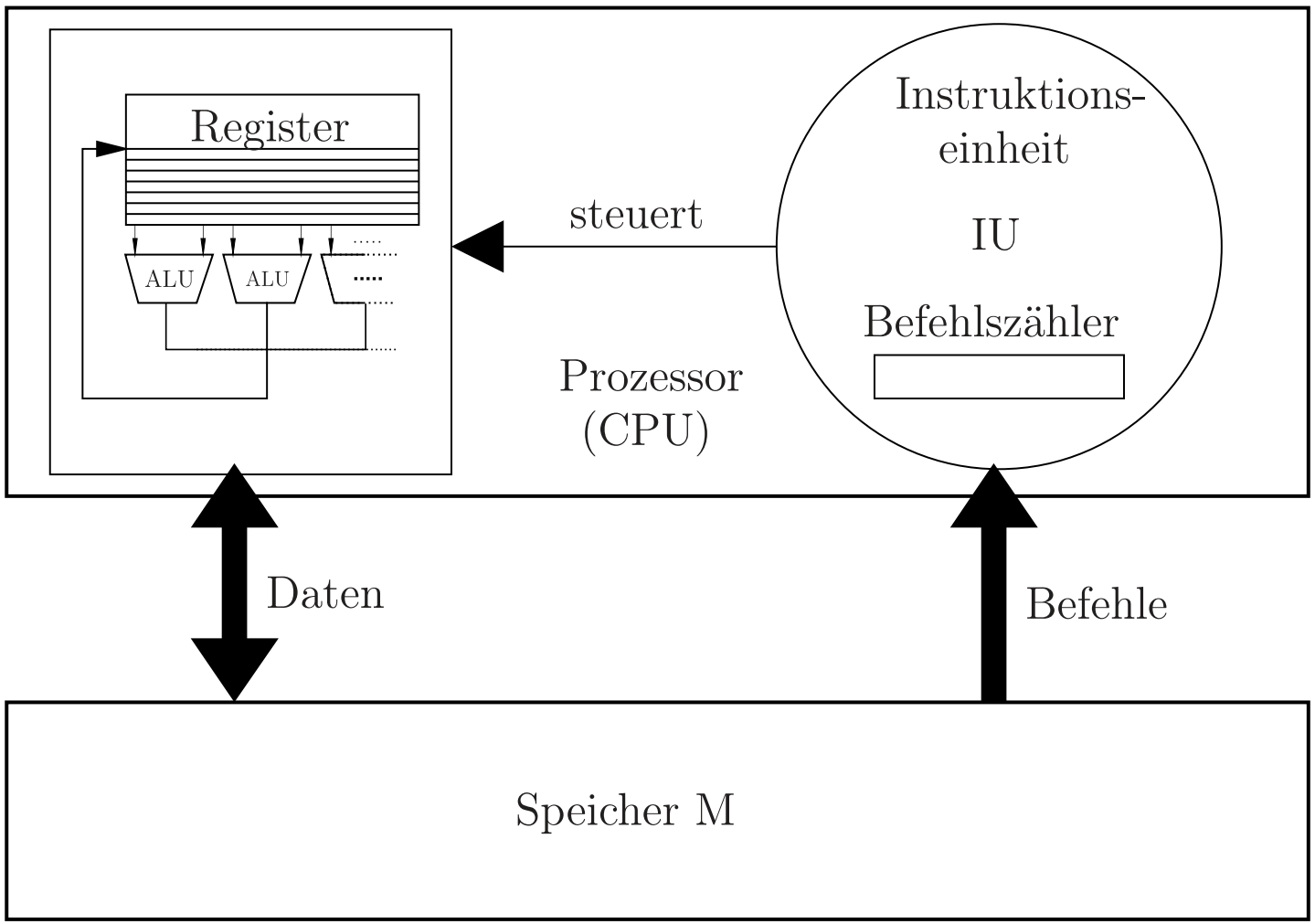
Hauptspeicher:



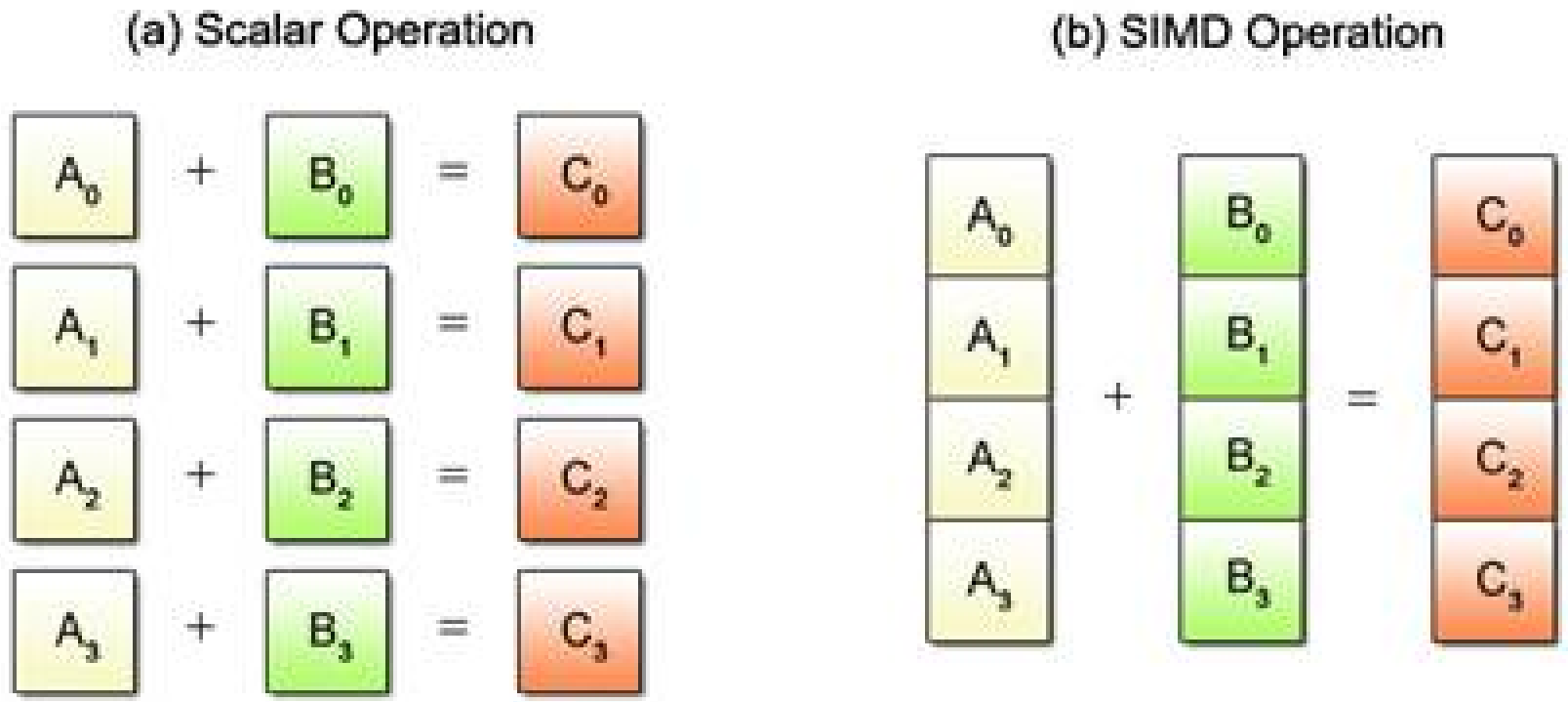
SIMD Prozessor



SIMD Prozessor



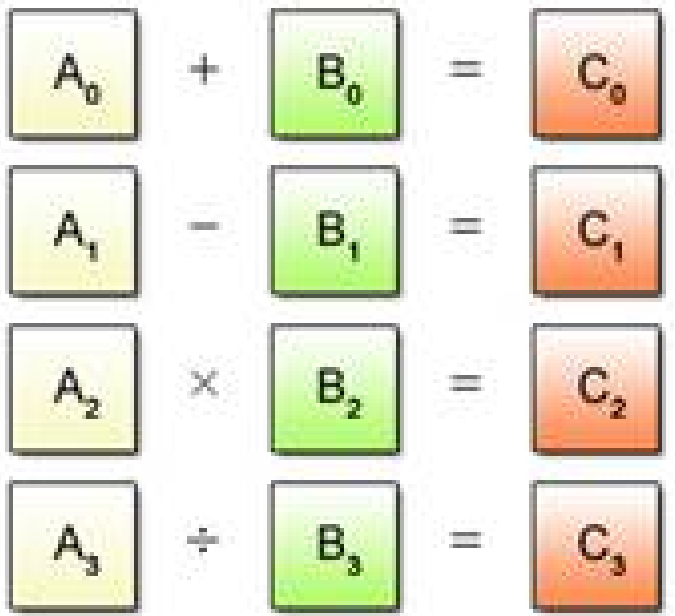
Vektorisierungs Konzept



Bilder: https://www.kernel.org/, Cell Programming Tutorial

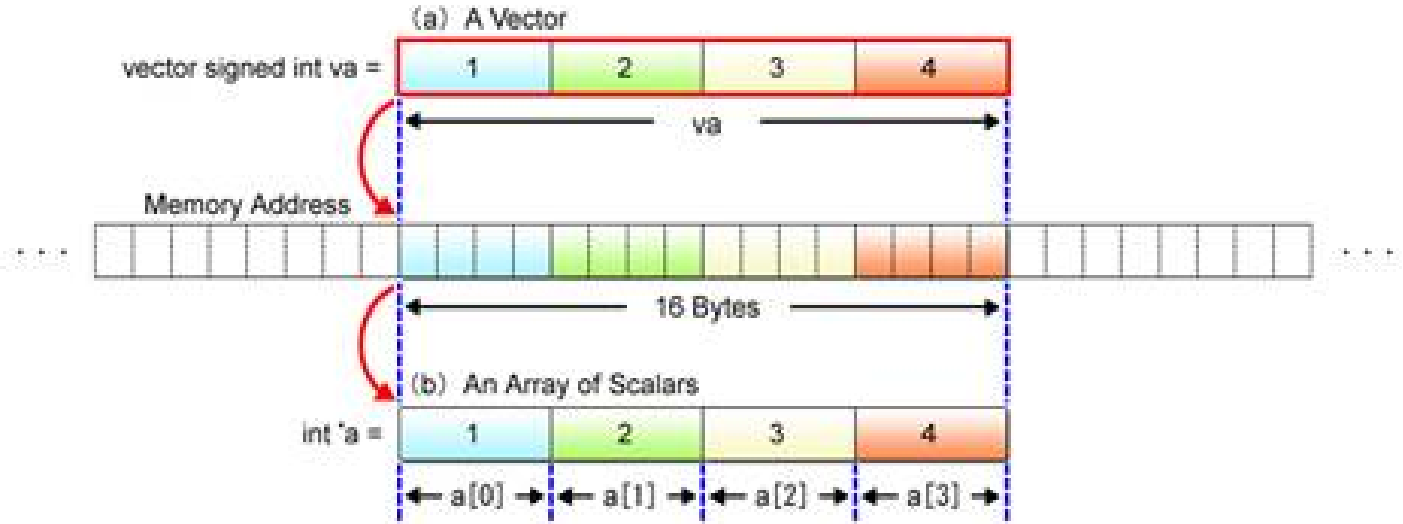
Vektorisierungs Konzept

Nicht möglich!



Bilder: https://www.kernel.org/, Cell Programming Tutorial

Vektorisierungs Konzept



Bilder: https://www.kernel.org/, Cell Programming Tutorial

Herausforderungen moderner Hardware

- Mehrere Ebenen von Parallelität
- MPI-parallel, Multi-core, SIMD
- Memory-wall

Herausforderungen moderner Hardware

- Mehrere Ebenen von Parallelität
- MPI-parallel, Multi-core, SIMD
- Memory-wall

Beispiel Intel Xeon E5-2698 v3 (Haswell)

- Beworbene Peak Performance: 486.4 GFlop/s

Herausforderungen moderner Hardware

- Mehrere Ebenen von Parallelität
- MPI-parallel, Multi-core, SIMD
- Memory-wall

Beispiel Intel Xeon E5-2698 v3 (Haswell)

- Beworbene Peak Performance: 486.4 GFlop/s
- 16 cores $\rightarrow$ ein Core: 30.4 GFlop/s

Herausforderungen moderner Hardware

- Mehrere Ebenen von Parallelität
- MPI-parallel, Multi-core, SIMD
- Memory-wall

Beispiel Intel Xeon E5-2698 v3 (Haswell)

- Beworbene Peak Performance: 486.4 GFlop/s
- 16 cores $\rightarrow$ ein Core: 30.4 GFlop/s
- AVX2+FMA $\rightarrow$ ohne FMA: 15.2 GFlop/s

Herausforderungen moderner Hardware

- Mehrere Ebenen von Parallelität
- MPI-parallel, Multi-core, SIMD
- Memory-wall

Beispiel Intel Xeon E5-2698 v3 (Haswell)

- Beworbene Peak Performance: 486.4 GFlop/s
- 16 cores $\rightarrow$ ein Core: 30.4 GFlop/s
- AVX2+FMA $\rightarrow$ ohne FMA: 15.2 GFlop/s
- 4× SIMD $\rightarrow$ ohne AVX: 3.8 GFlop/s

$\rightarrow$ klassischer, nicht-paralleler Code begrenzt auf 3.8 GFlop/s

$\rightarrow$ man verliert 99% der Leistung