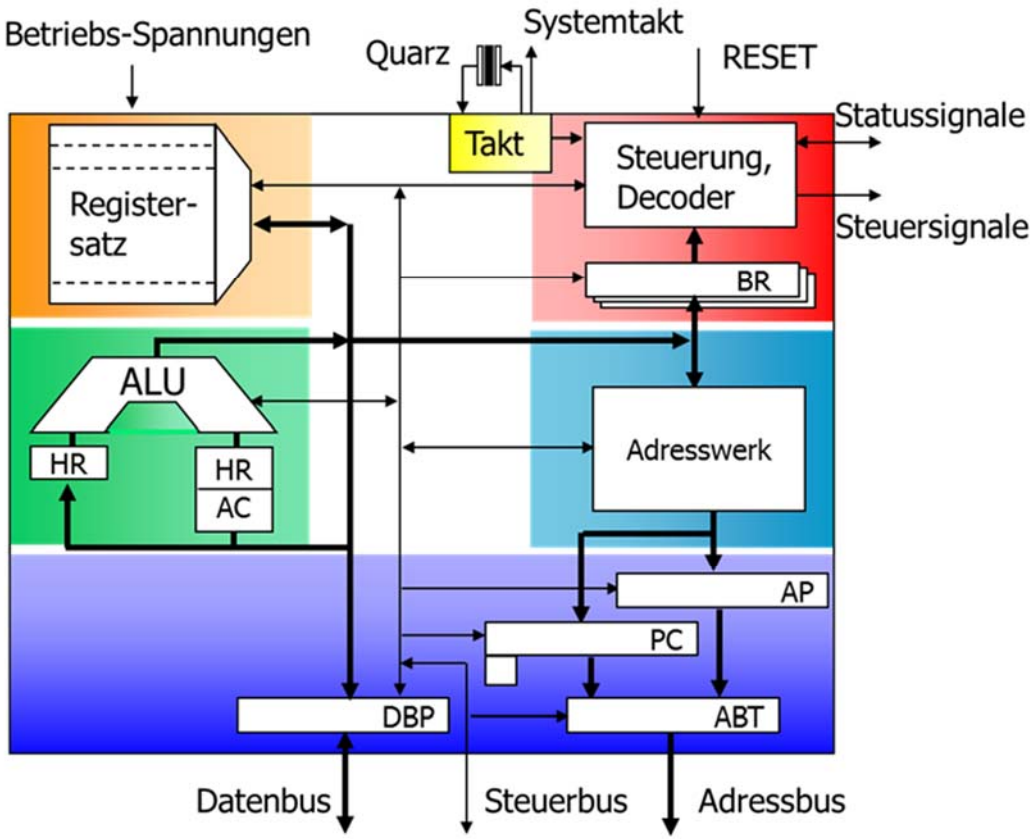
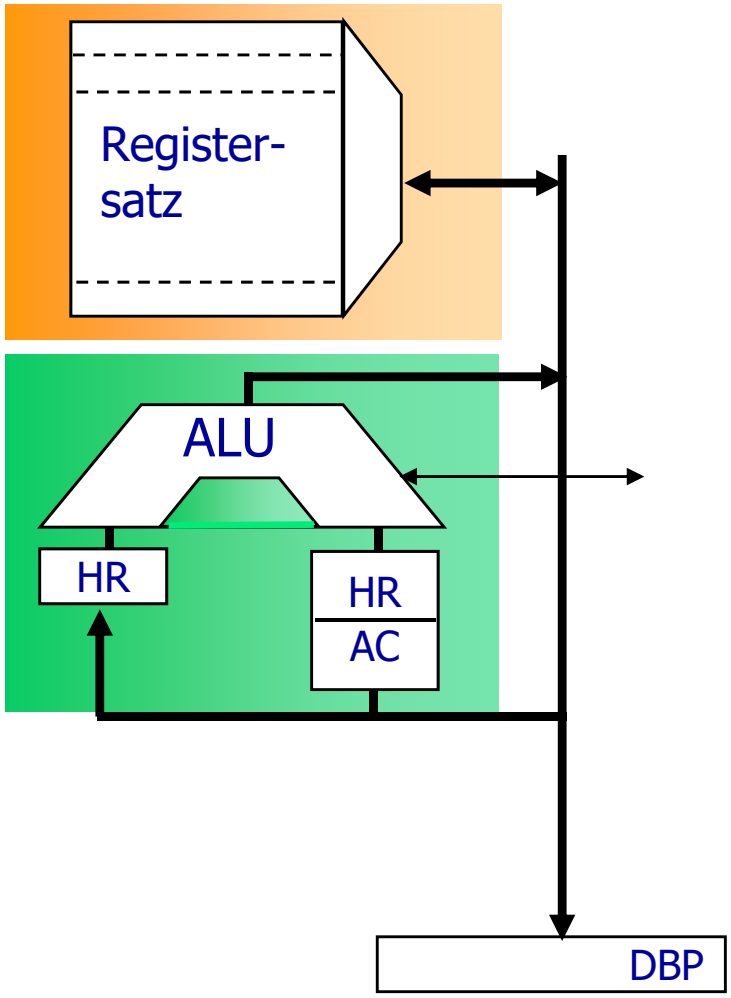
Aufbau eines einfachen µP

- Steuerwerk
- Rechenwerk
- Registersatz
- Adresswerk
- Systembusschnittstelle
- Interne Busse



3.2.3 Der Registersatz

- **Registersatz**
  - Häufig benutzte Daten (Operanden, Ergebnisse) können dort zwischengespeichert werden
  - schnellerer Zugriff als auf den Hauptspeicher

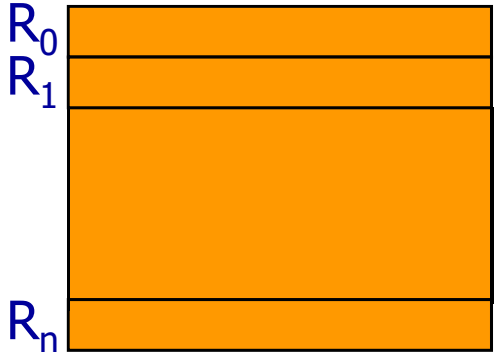


Der Registersatz

- **Registersatz**
  - Üblicherweise als Dual Port Speicher zwischen Eingangs- und Ausgangsbus realisiert
    - Schreiben eines Registers und gleichzeitiges Lesen eines anderen Registers möglich
    - Heutige superskalar-Prozessoren: pro Takt mehrere allgemeine Register schreiben und lesen
  - Register mit Zusatzfunktionen:
    - Inkrementieren / Dekrementieren
    - auf Null setzen
    - Inhalt verschieben

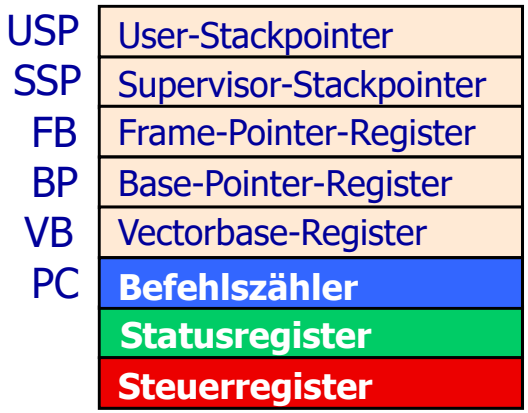
Der Registersatz

- **Registersatz**



universelle, allgemeine Register
(*general purpose register*)

Daten- und Adressregister



Spezialregister
(special purpose register)

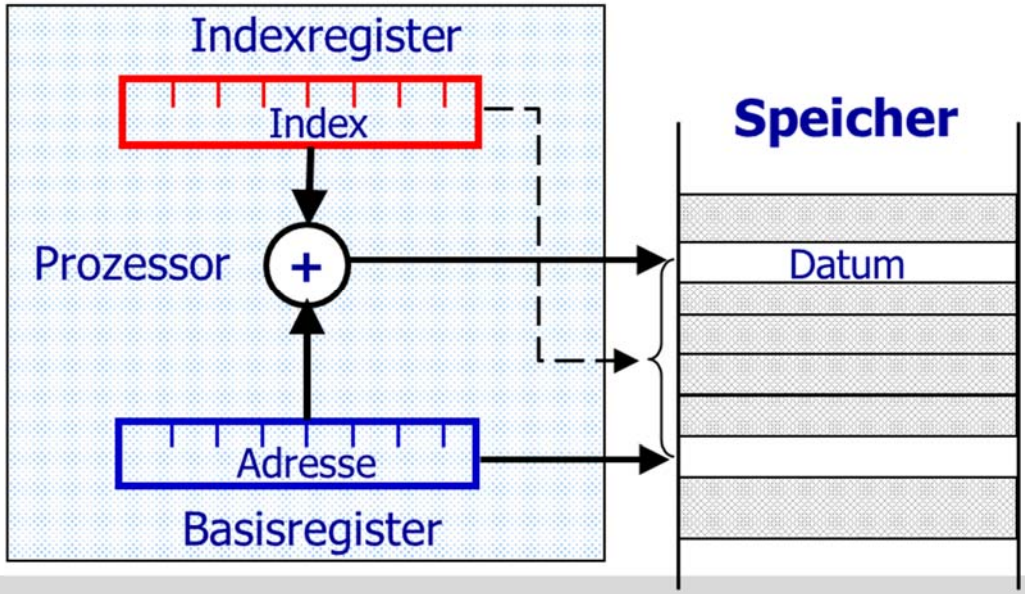
Registersatz

Der Registersatz

- **Daten- und Adressregister**
  - **Datenregister:**
    - Zwischenspeichern von Operanden / Ergebnissen
    - schneller Zugriff auf häufig benutzte Daten
    - bei modernen Prozessoren sind mehrere Datenregister als Akkumulator nutzbar
  - **Adressregister:**
    - Speichern von Adressinformationen eines Speicheroperanden (Operanden im Hauptspeicher)
      - Basisregister
      - Indexregister

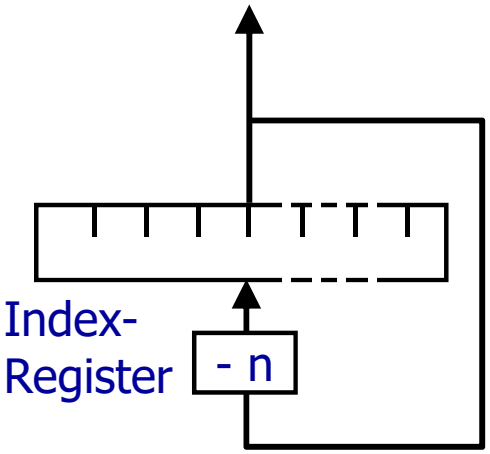
Der Registersatz

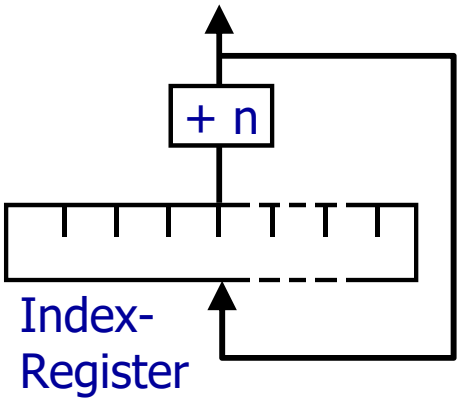
- **Adressregister**
  - **Basisregister**
    - enthält die Anfangsadresse eines Speicherbereichs
    - diese bleibt während der Bearbeitung des Speicherbereichs unverändert
  - **Indexregister**
    - enthält eine Distanz (Offset, Displacement) zu einer Basisadresse und dient zur Auswahl eines bestimmten Datums des Speicherbereichs

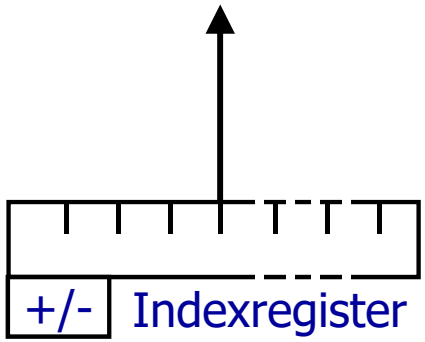


Der Registersatz

- **Adressregister**
  - **Indexregister:** Automatische Modifikation
    - Pre-Dekrement: Dekrementieren des Registerwerts um n vor Adressierung einer Speicherzelle
    - Post-Inkrement: Inkrementieren des Registerwerts um n nach Adressierung einer Speicherzelle
    - Auto-Inkrement / Auto-Dekrement Register

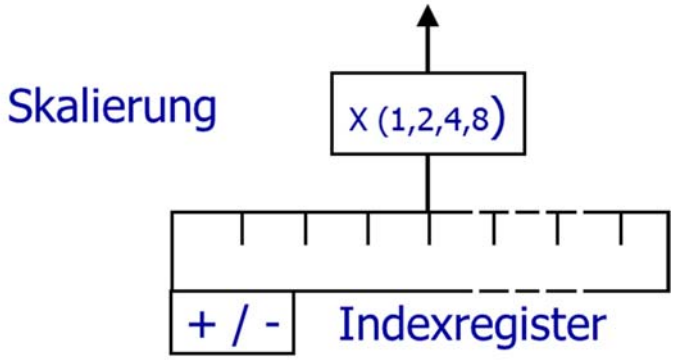






Der Registersatz

- **Adressregister**
  - **Indexregister:** Skalierungsfaktor
    - Das Indexregister wird vor der Auswertung je nach aktueller Datenlänge (1 Byte, 2 Byte, 4 Byte, 8 Byte) mit dem Faktor 1, 2, 4 oder 8 multipliziert
    - Vorteil:
      - bessere Ausnutzung der Registerbreite, da das Register selbst nur noch um 1 inkrementiert bzw. dekrementiert werden muss



Der Registersatz

- **Spezialregister**
  - Befehlszähler (instruction pointer)
  - Steuerregister und Statusregister
  - Interrupt Vector Base Register
    - Enthält die Basisadresse der Vektortabelle
    - Vektortabelle enthält die Interruptvektoren
    - Interruptvektor ist die Adresse der Unterbrechungsbehandlungsroutine, die bei einer Ausnahmesituation angesprungen wird.
  - Stackregister (user und supervisor Stackpointer)

Der Registersatz

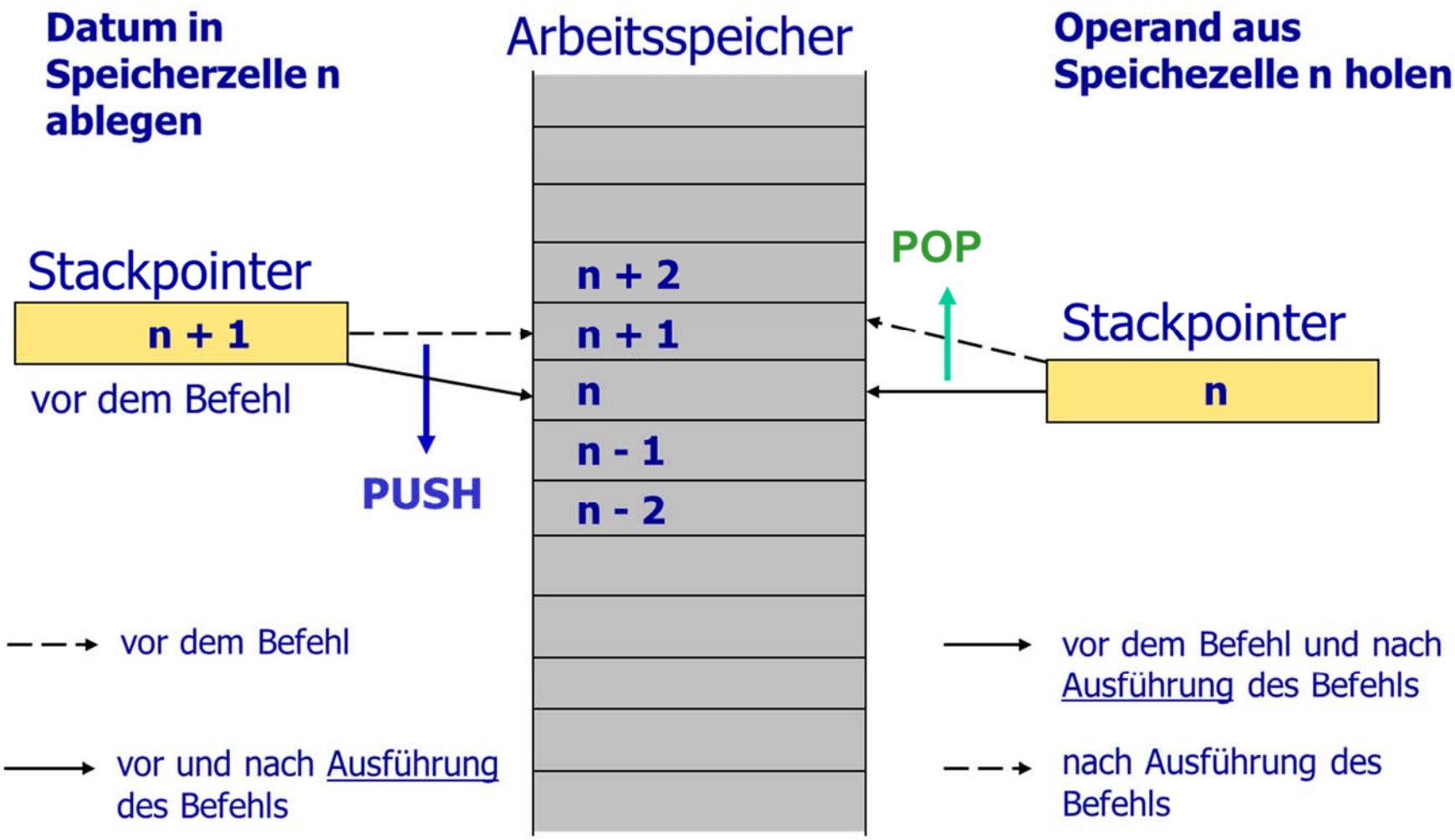
- **Spezialregister**
  - **Stackregister (Stapelzeiger, Stack Pointer SP):**
    - enthält die Adresse des zuletzt in den Stack eingetragenen Datums
    - Spezielle Befehle zur Datenübertragung in den bzw. aus dem Stack
      - PUSH: Inhalt eines Registers wird in den Stack übertragen
      - POP (pull): Inhalt eines Registers wird vom Stack geladen
    - Manche Prozessoren setzen stattdessen die Pre-Dekrement / Post-Inkrement Technik ein

Der Registersatz

- **Laufzeitkeller / Stack**
  - Ein besonderer Speicherbereich, der normalerweise im Arbeitsspeicher angelegt ist (software stack).
  - Ist nach dem Kellerprinzip (LIFO Last-in-first-out) organisiert
  - Wächst in Richtung niederwertiger Adressen
  - Funktion:
    - Ablegen des Inhalts des Befehlszählers (Rücksprungadresse) und der Übergabeparameter und der lokalen Variablen beim Unterprogrammaufruf
    - Ablegen des Prozessorstatus und des Programmzählers beim Aufruf von Unterbrechungs-Routinen
  - Bei modernen Prozessoren häufig mehrere getrennte Laufzeitkeller (Stacks): System Stack, User Stack, Data Stack

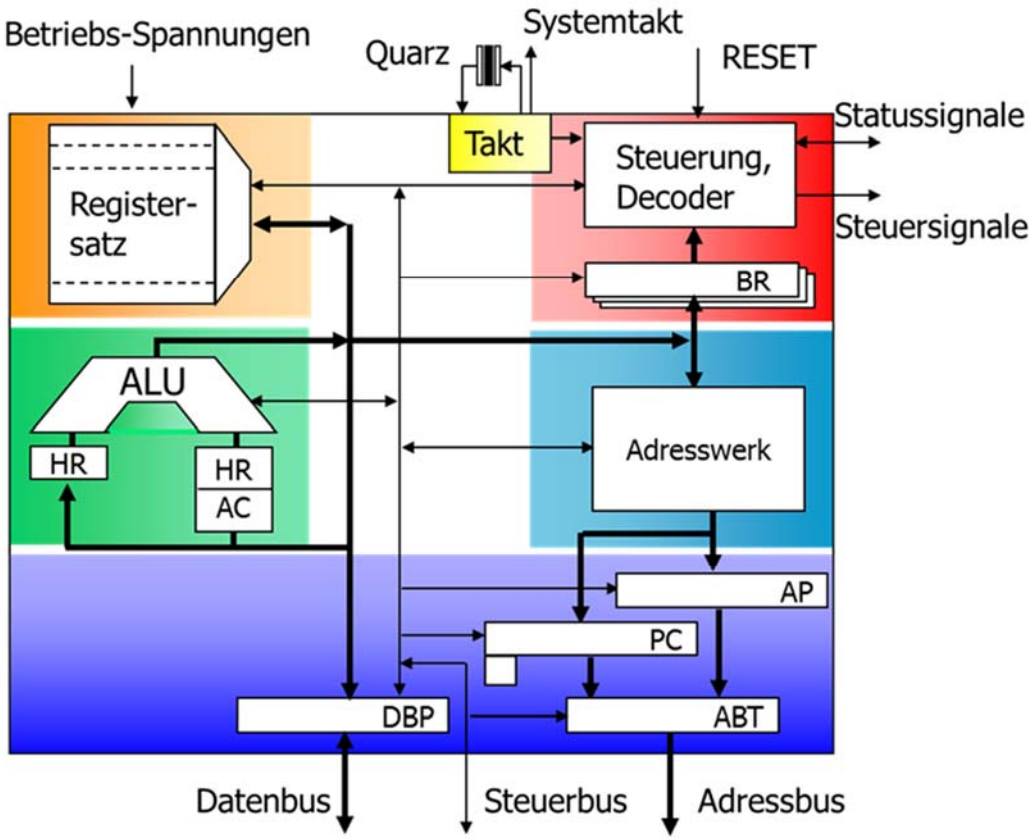
Der Registersatz

- **Verwaltung des Laufzeitkellers / Stacks**



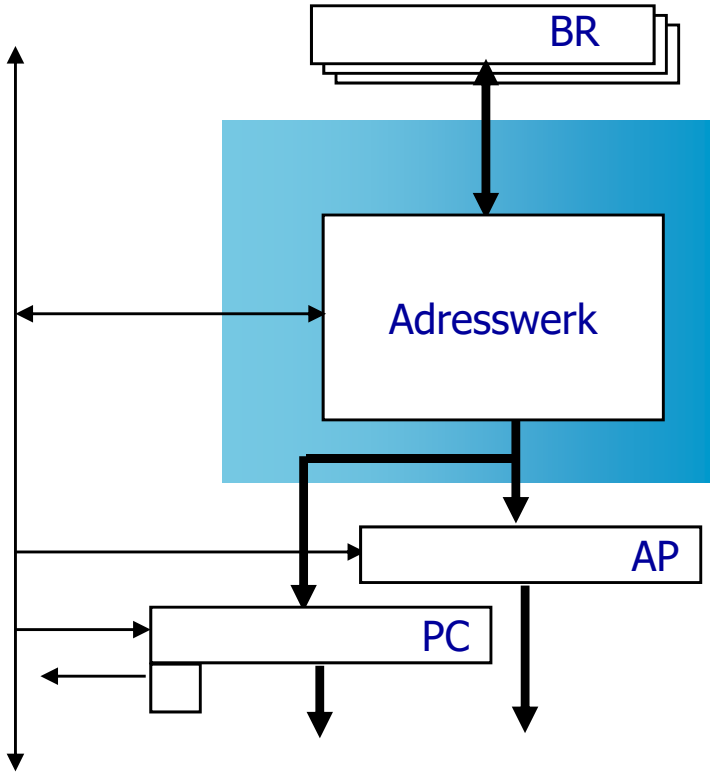
Aufbau eines einfachen µP

- Steuerwerk
- Rechenwerk
- Registersatz
- Adresswerk
- Systembusschnittstelle
- Interne Busse



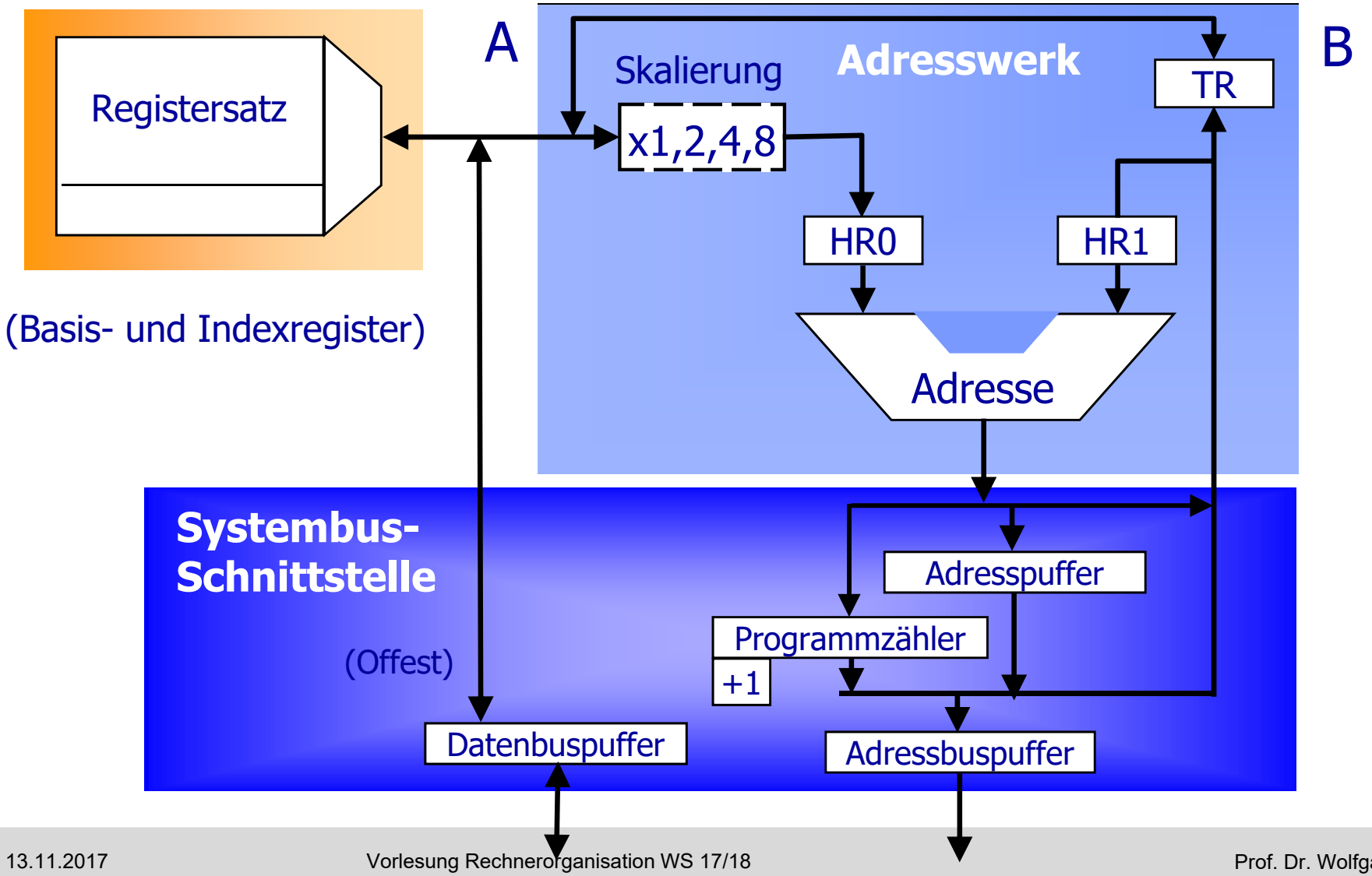
3.2.4 Das Adresswerk

- Führt die vom Steuerwerk angestoßenen Adressberechnungen für Befehle und Daten durch
- Eigenständiges Adresswerk ermöglicht unabhängig vom Rechenwerk Adressberechnungen durchzuführen
  - Operationen im Rechenwerk und Adressberechnung können parallel und ohne Konflikte durchgeführt werden
- Erlaubt komplexe Adressberechnungen



3.2.4 Das Adresswerk

- **Aufbau eines einfachen Adresswerks**

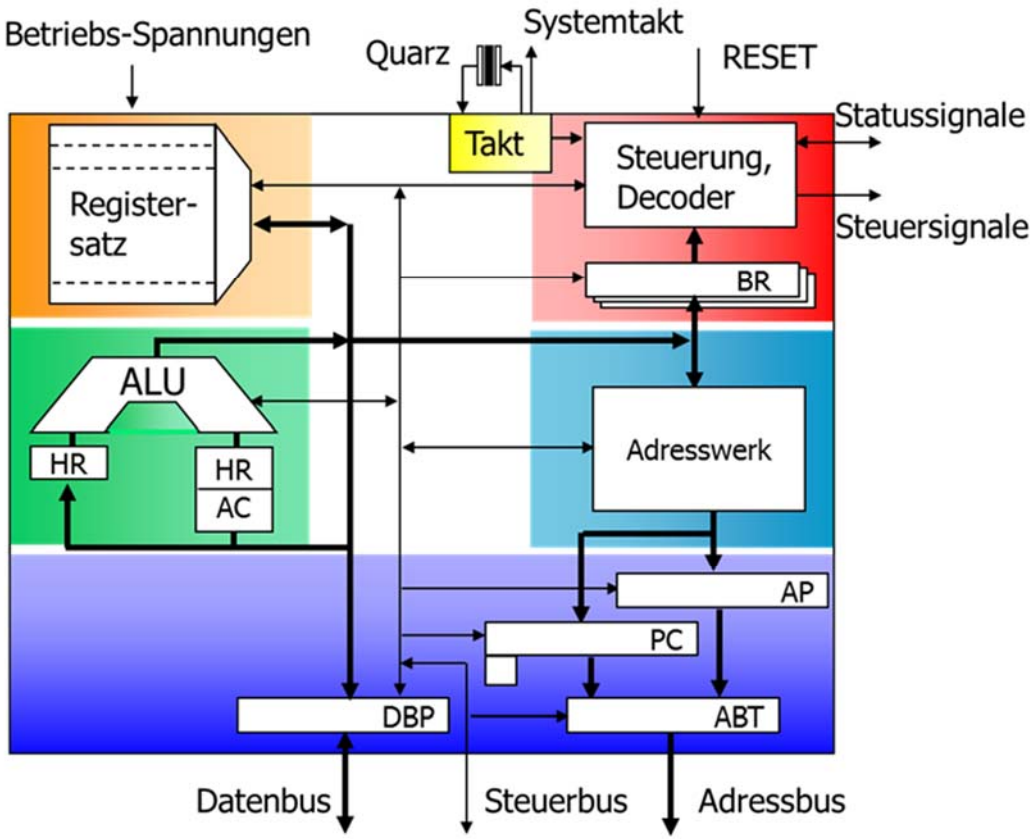


3.2.4 Das Adresswerk

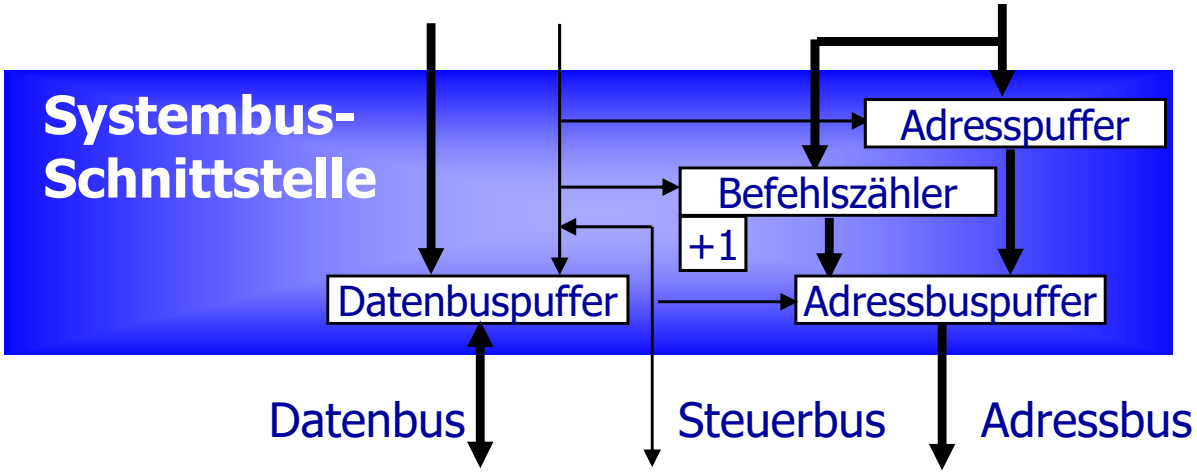
- **Funktionsweise eines einfachen Adresswerks**
  - **Adress-Addierer mit 2 Eingängen**
    - Eingang A:
      - Registersatz (Adresse aus Basis- oder Indexregister)
      - Datenbuspuffer (absolute Adresse im Befehl oder absolute Adressdistanz zu einem Basisregister)
    - Eingang B:
      - Befehlszähler (Adresse des aktuellen Befehls)
      - Adresspuffer (Adresse des aktuellen Operanden)
    - Ausgang
      - kann über den Adresspuffer oder den Programmzähler auf Eingang B rückgekoppelt werden
      - Adresspuffer und Programmzähler als Akkumulatoren

Aufbau eines einfachen µP

- Steuerwerk
- Rechenwerk
- Registersatz
- Adresswerk
- Systembusschnittstelle
- Interne Busse



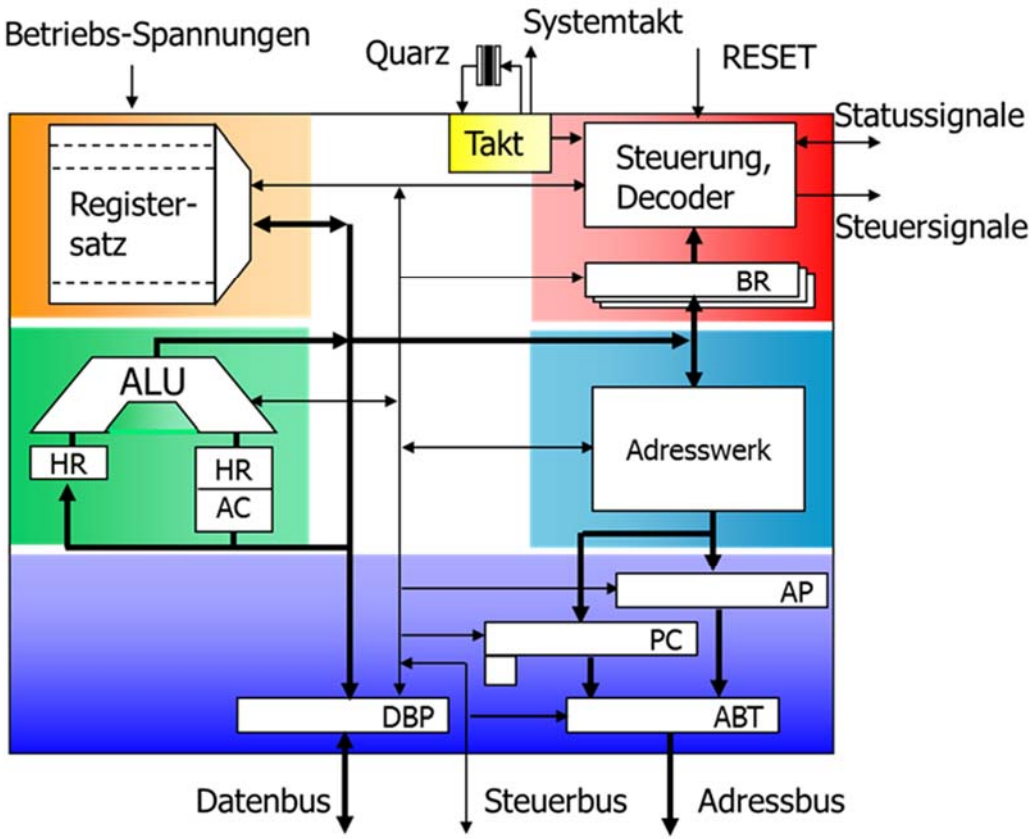
3.2.5 Systembus-Schnittstelle



- Enthält diverse Zwischenspeicher-Register (Puffer) zur kurzfristigen Speicherung von Adressen und Daten, z.B.
  - Befehlszähler: Adresse des nächsten auszuführenden Befehls (Instruction Pointer, Program Counter)
  - Adressbuspuffer: Adresse des zu ladenden oder zu speichernden Datums im Hauptspeicher
  - Datenbuspuffer: puffert den zu speichernden oder zu ladenden Wert
  - Enthält Aus- und Eingangstreiber (Tristate-Treiber)

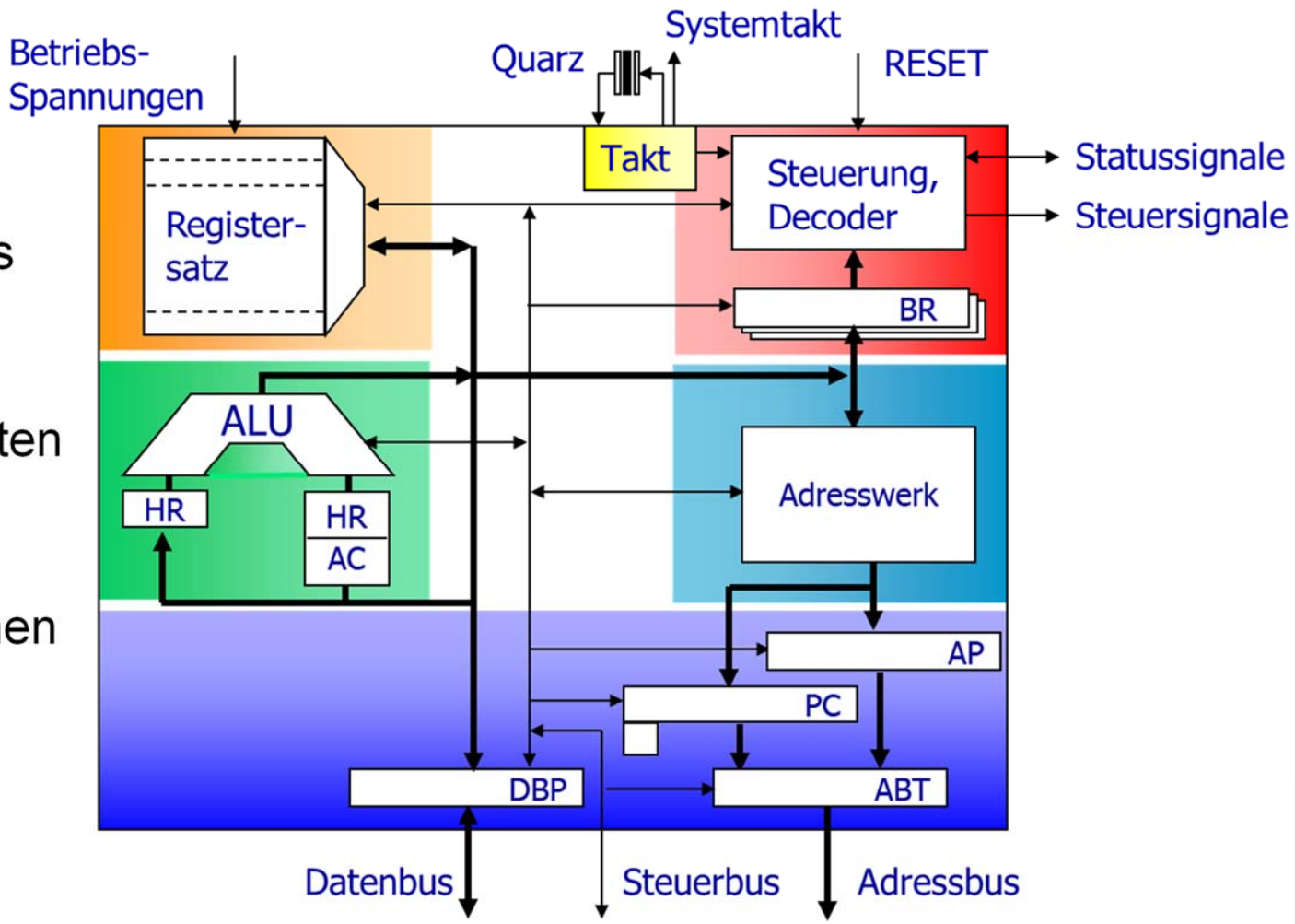
Aufbau eines einfachen µP

- Steuerwerk
- Rechenwerk
- Registersatz
- Adresswerk
- Systembusschnittstelle
- Interne Busse



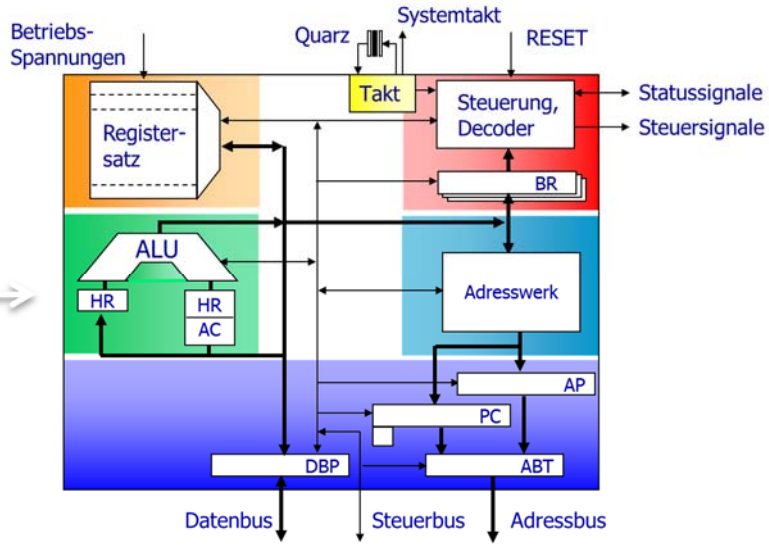
3.2.6 Interne Busse

- Interne Busse oder Datenpfade
  - Verbinden die Komponenten des Prozessors
  - Es werden Adressen und Daten übertragen
  - Status- / Steuerinformationen



Moderne Mikroprozessoren

- Weitere Komponenten auf dem Chip:
- Beispiele:
  - Speicherverwaltungseinheit (Memory Management Unit, MMU)
  - Rechenwerke (Gleitkommaverarbeitung, FPU, Multimediaeinheiten)
  - Cache-Speicher

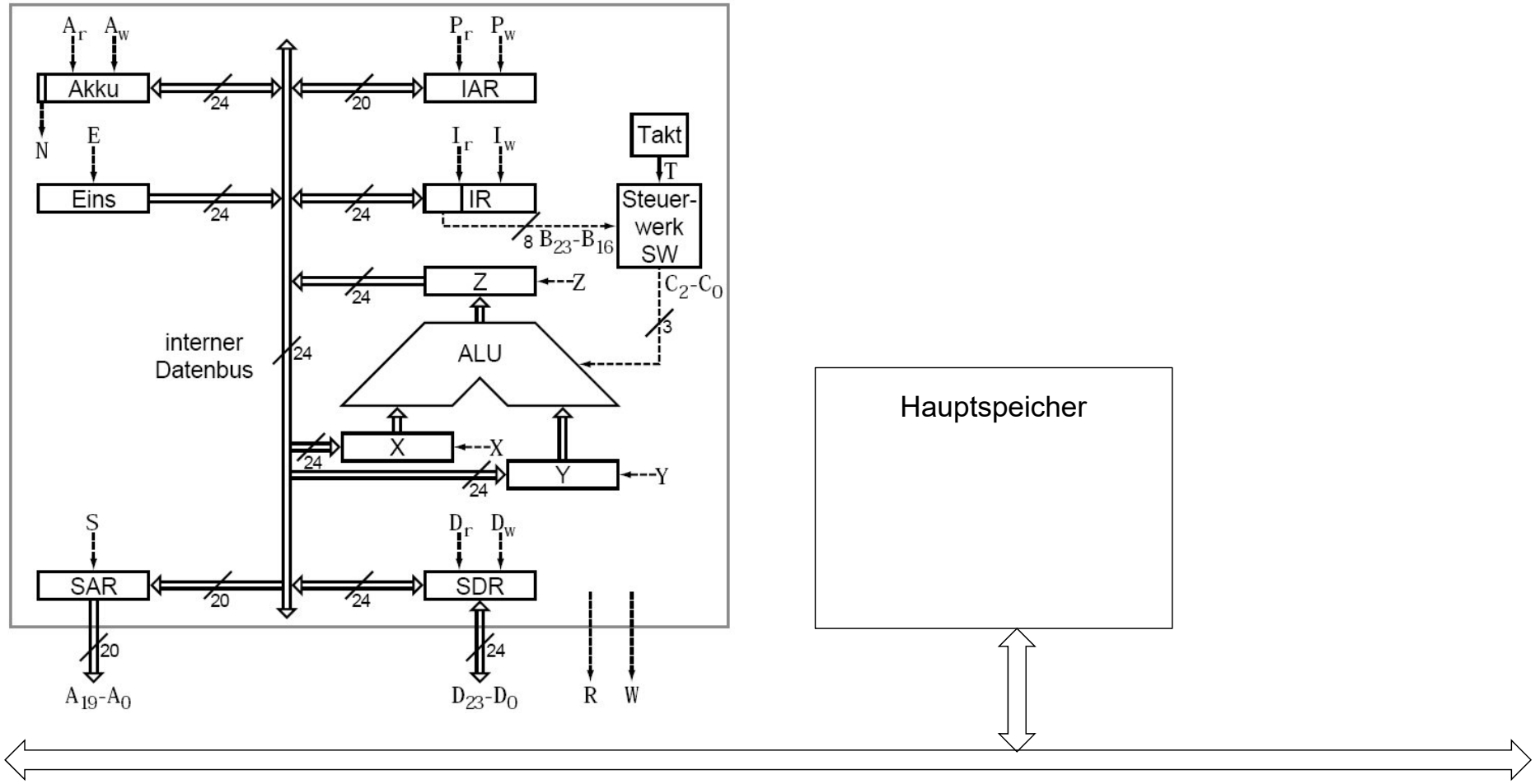


MIMA: Mikroprogrammierbarer Rechner

- **MIMA: Beispielrechner (Übungen)**
  - Mikroprogrammierbare Minimalmaschine
  - SW mit 10 Meldesignalen, 18 Steuersignale und Mikroprogrammspeicher für maximal 256 Mikrobefehle
  - Befehlsabarbeitung:
    - Lese-Phase
    - Dekodierphase
    - Ausführungsphase
  - 3 Taktzyklen für Lese- und Schreibzugriffe

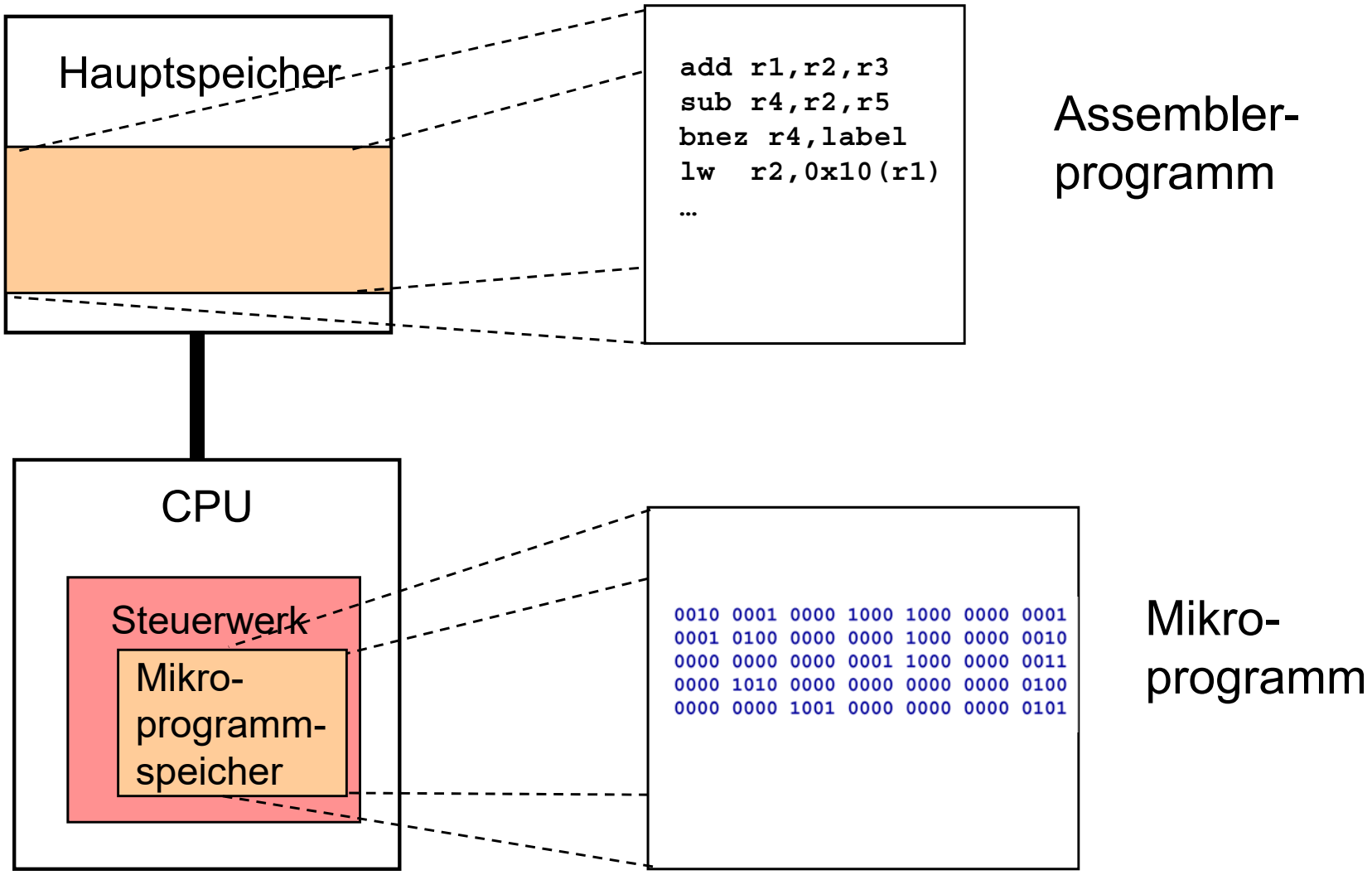
MIMA: Mikroprogrammierbarer Rechner

MIMA: Beispielrechner (Übungen)



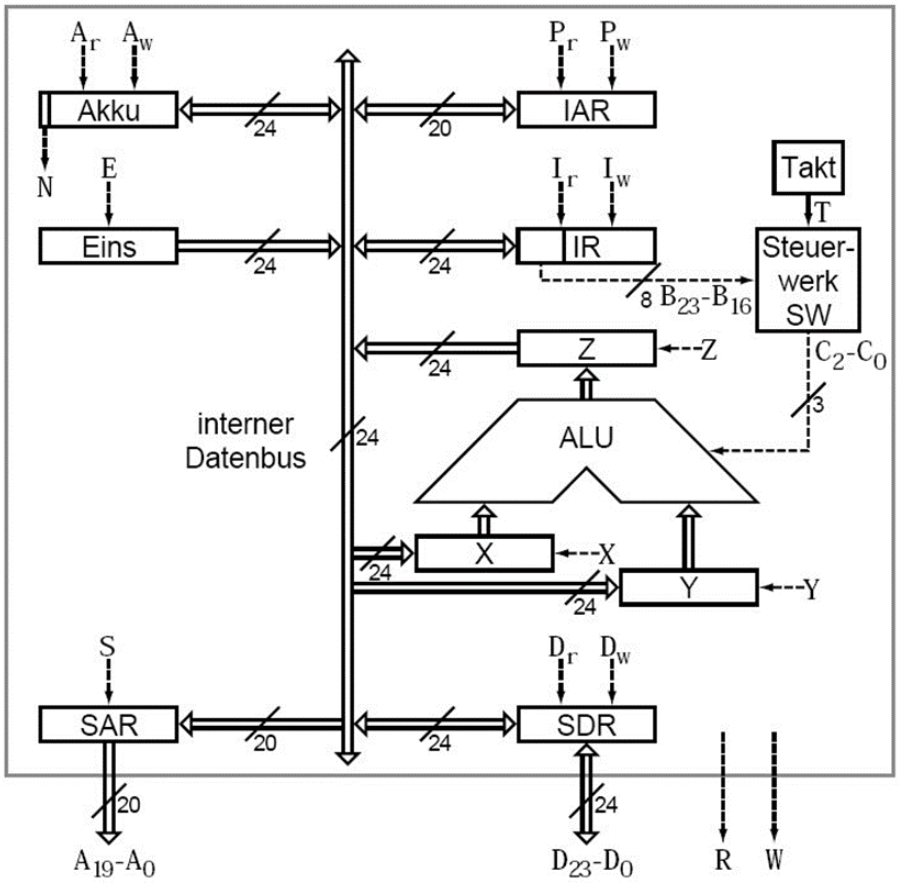
MIMA: Mikroprogrammierbarer Rechner

MIMA: Mikroprogrammsteuerung



MIMA: Mikroprogrammierbarer Rechner

MIMA: Mikroprogrammsteuerung

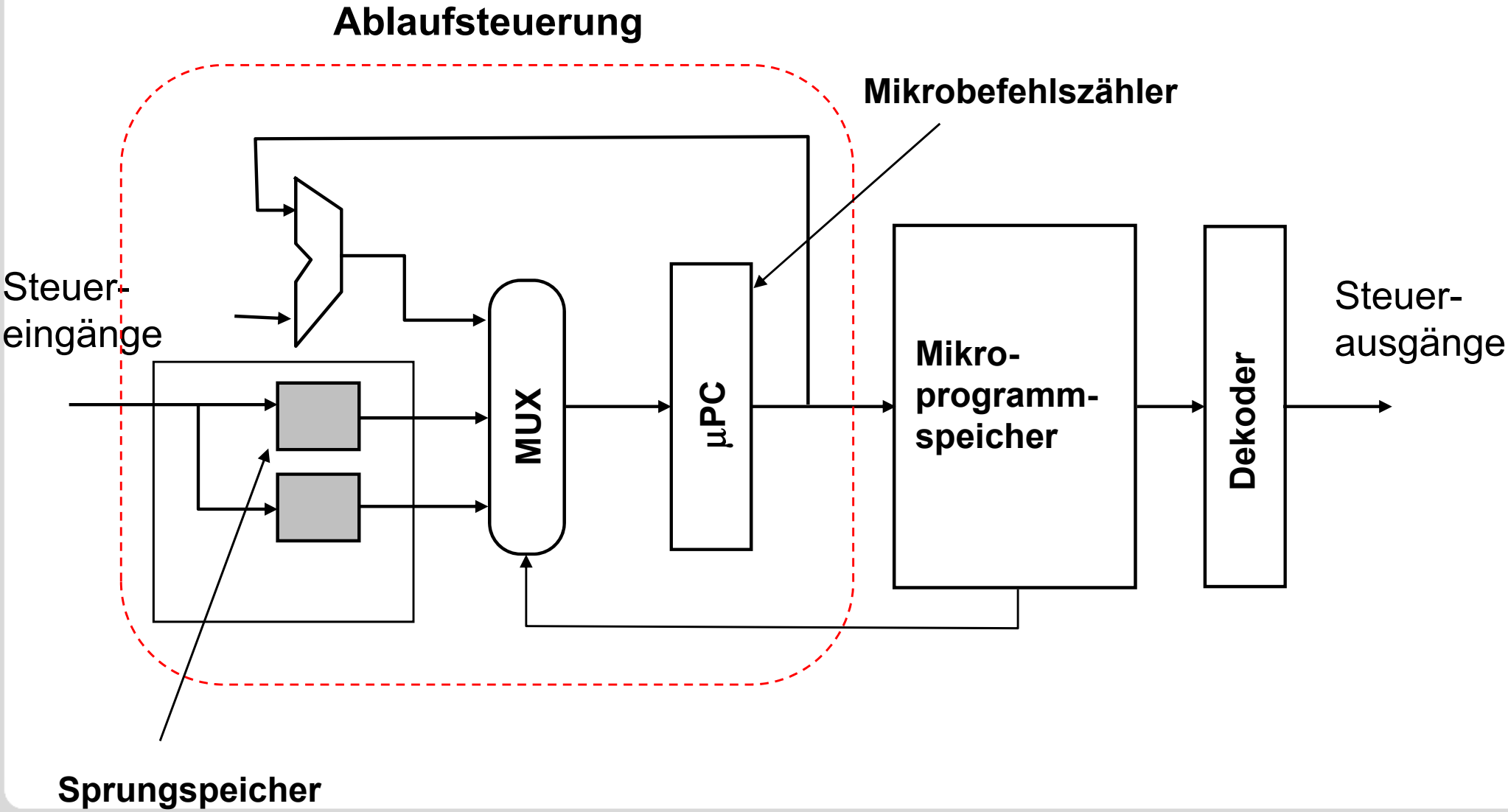


Mikrobefehlsformat

A_r	A_w	X	Y	Z	E	P_r	P_w	I_r	I_w	D_r	D_w	S	C_2	C_1	C_0	R	W	0 0	Folgeadresse F
27			24				20				16				12			9 8	0

MIMA: Mikroprogrammierbarer Rechner

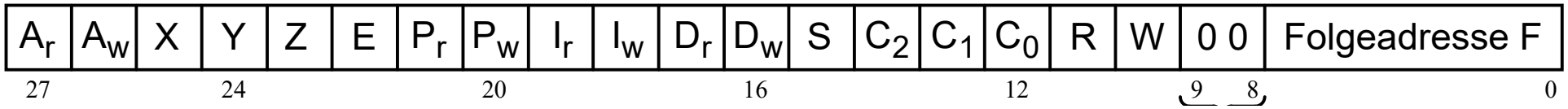
MIMA: Ablaufsteuerung



MIMA: Mikroprogrammierbarer Rechner

MIMA: Beispielrechner (Übungen)

Mikrobefehlsformat



Fetch-Phase bei der MIMA:

```
1. Takt:  IAR -> SAR;   IAR -> X;   R = 1
2. Takt:  Eins -> Y;  R = 1
3. Takt:  ALU auf addieren;   R = 1
4. Takt:  Z -> IAR
5. Takt:  SDR -> IR
```

Mikroprogramm der Fetch Phase

```
0010 0001 0000 1000 1000 0000 0001
0001 0100 0000 0000 1000 0000 0010
0000 0000 0000 0001 1000 0000 0011
0000 1010 0000 0000 0000 0000 0100
0000 0000 1001 0000 0000 0000 0101
```