Vom Programm zum Computer

- Aufbau von Computern, die PRO verarbeiten können
- Architektur
- maschinennahe Sprachen
- Ebenenmodell der Rechnerarchitektur

Vom Programm zum Computer
Beseitigung höherer Sprachelemente

- 1. Ansatz: C kann PRO direkt ausführen
  - C wird riesengroß, unübersichtlich, kann dann keine andere Programmiersprache mehr
- 2. Ansatz: Sprachelemente von PRO schrittweise auf immer einfachere Elemente zurückführen, bis Niveau erreicht ist, das als gegeben betrachtet werden kann (z.B. Niveau elektronischer Schaltungen).
- Beispiel: ein solcher *Reduktionsschritt*

Vom Programm zum Computer
Beispielprogramm

Problem: Einlesen einer positive Zahl n, einer bel. lange Folge von positiven ganzen Zahlen. In "größer" und "kleiner" summiere Zahlen, die ≥n bzw. ≤n sind. Wird Null eingegeben, stoppt das Programm und gibt Werte größer und kleiner aus.

Programm in PRO

```
def x, n, größer, kleiner: Zahl
größer←0
kleiner←0
lies(n)
lies(x)
solange x≠0 tue
    wenn x≤n dann
        kleiner←kleiner+x
    sonst
        größer←größer+x
    ende
    lies(x)
ende
zeige(größer)
zeige(kleiner).
```

Vom Programm zum Computer

Beispielprogramm

Sprachelemente
zu komplex, um sie unmittelbar einer Maschine zu übertragen. "Einfachere" (= leichter zu realisierende) Sprachelemente suchen, mit deren Hilfe man die komplexeren Sprachelemente präzise beschreiben kann

Programm in PRO

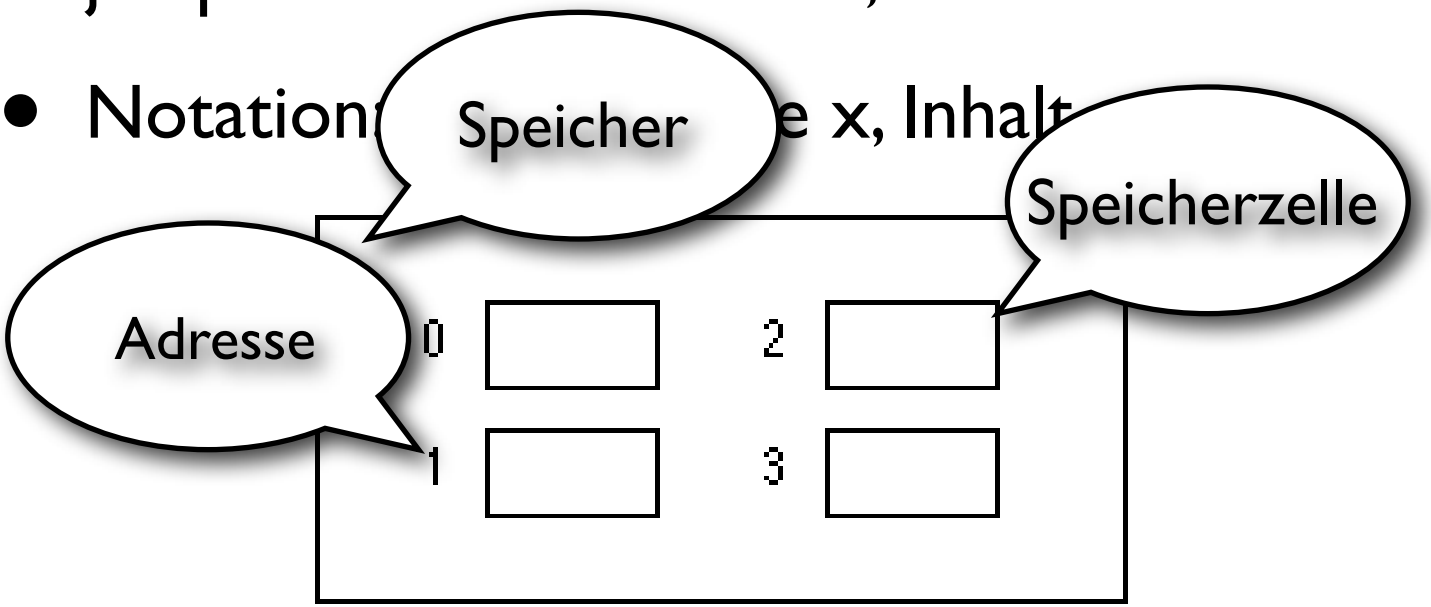
```
def x, n, größer, kleiner: Zahl
größer←0
kleiner←0
lies(n)
lies(x)
solange x≠0 tue
    wenn x≤n dann
        kleiner←kleiner+x
    sonst
        größer←größer+x
    ende
    lies(x)
ende
zeige(größer)
zeige(kleiner).
```

Vom Programm zum Computer
Daten

- Daten erfordern Ablage
  - Ablage = *Speicher* unterteilt in *Speicherzellen* identifizierbar durch *Adressen*
  - je Speicherzelle ein Wert, also 4
  - Notation: Speicherzelle x, Inhalt «x».

Vom Programm zum Computer
Daten

- Daten erfordern Ablage
  - Ablage = *Speicher* unterteilt in *Speicherzellen* identifizierbar durch *Adressen*
  - je Speicherzelle ein Wert, also 4
  - Notation: [illegible] x, Inhalt



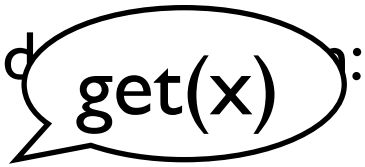
Vom Programm zum Computer
elementare Anweisungen

elementare Anweisungen am Programm identifizieren:

- Einlesen einer Zahl in eine Speicherzelle.
- Transportiere in eine Speicherzelle die Summe des Inhalts zweier Speicherzellen.
- Transportiere in eine Speicherzelle eine Konstante.
- Führe Vergleich "≠" mit Inhalt einer Speicherzelle und einer Konstante aus.
- Führe Vergleich "≤" mit Inhalt zweier Speicherzellen aus.
- Ausgeben einer Speicherzelle

Vom Programm zum Computer
elementare Anweisungen

elementare Anweisungen am Programm id[illegible]:



- Einlesen einer Zahl in eine Speicherzelle.
- Transportiere in eine Speicherzelle die Summe des Inhalts zweier Speicherzellen.
- Transportiere in eine Speicherzelle eine Konstante.
- Führe Vergleich "≠" mit Inhalt einer Speicherzelle und einer Konstante aus.
- Führe Vergleich "≤" mit Inhalt zweier Speicherzellen aus.
- Ausgeben einer Speicherzelle

Vom Programm zum Computer
elementare Anweisungen

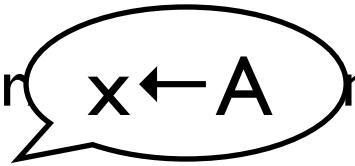
elementare Anweisungen am Programm identifizieren:

- Einlesen einer Zahl in eine Speicherzelle.
- Transportiere in eine Speicherzelle die Summe des Inhalts zweier Speicherzellen.
- Transportiere in eine Speicherzelle eine Konstante.
- Führe Vergleich "≠" mit Inhalt einer Speicherzelle und einer Konstante aus.
- Führe Vergleich "≤" mit Inhalt zweier Speicherzellen aus.
- Ausgeben einer Speicherzelle

Vom Programm zum Computer
elementare Anweisungen

elementare Anweisungen am Programm identifizieren:

- Einlesen einer Zahl in eine Speicherzelle.
- Transportiere in eine Speicherzelle die Summe des Inhalts zweier Speicherzellen.



- Transportiere in eine Speicherzelle eine Konstante.
- Führe Vergleich "≠" mit Inhalt einer Speicherzelle und einer Konstante aus.
- Führe Vergleich "≤" mit Inhalt zweier Speicherzellen aus.
- Ausgeben einer Speicherzelle

Vom Programm zum Computer
elementare Anweisungen

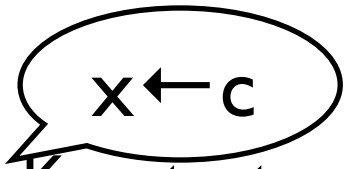
elementare Anweisungen am Programm identifizieren:

- Einlesen einer Zahl in eine Speicherzelle.
- Transportiere in eine Speicherzelle die Summe des Inhalts zweier Speicherzellen.
- Transportiere in eine Speicherzelle eine Konstante.
- Führe Vergleich "≠" mit Inhalt einer Speicherzelle und einer Konstante aus.
- Führe Vergleich "≤" mit Inhalt zweier Speicherzellen aus.
- Ausgeben einer Speicherzelle

Vom Programm zum Computer
elementare Anweisungen

elementare Anweisungen am Programm identifizieren:

- Einlesen einer Zahl in eine Speicherzelle.
- Transportiere in eine Speicherzelle die Summe des Inhalts zweier Speicherzellen.
- Transportiere in eine Speicherzelle eine Konstante.



- Führe Vergleich "≠" mit Inhalt einer Speicherzelle und einer Konstante aus.
- Führe Vergleich "≤" mit Inhalt zweier Speicherzellen aus.
- Ausgeben einer Speicherzelle

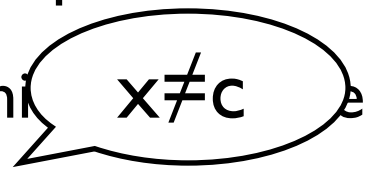
Vom Programm zum Computer
elementare Anweisungen

elementare Anweisungen am Programm identifizieren:

- Einlesen einer Zahl in eine Speicherzelle.
- Transportiere in eine Speicherzelle die Summe des Inhalts zweier Speicherzellen.
- Transportiere in eine Speicherzelle eine Konstante.
- Führe Vergleich "≠" mit Inhalt einer Speicherzelle und einer Konstante aus.
- Führe Vergleich "≤" mit Inhalt zweier Speicherzellen aus.
- Ausgeben einer Speicherzelle

Vom Programm zum Computer
elementare Anweisungen

elementare Anweisungen am Programm identifizieren:

- Einlesen einer Zahl in eine Speicherzelle.
- Transportiere in eine Speicherzelle die Summe des Inhalts zweier Speicherzellen.
- Transportiere in eine Speicherzelle eine Konstante.
- Führe Vergleich "≠" mi 

er Speicherzelle und einer Konstante aus.
- Führe Vergleich "≤" mit Inhalt zweier Speicherzellen aus.
- Ausgeben einer Speicherzelle

Vom Programm zum Computer
elementare Anweisungen

elementare Anweisungen am Programm identifizieren:

- Einlesen einer Zahl in eine Speicherzelle.
- Transportiere in eine Speicherzelle die Summe des Inhalts zweier Speicherzellen.
- Transportiere in eine Speicherzelle eine Konstante.
- Führe Vergleich "≠" mit Inhalt einer Speicherzelle und einer Konstante aus.
- Führe Vergleich "≤" mit Inhalt zweier Speicherzellen aus.
- Ausgeben einer Speicherzelle

Vom Programm zum Computer
elementare Anweisungen

elementare Anweisungen am Programm identifizieren:

- Einlesen einer Zahl in eine Speicherzelle.
- Transportiere in eine Speicherzelle die Summe des Inhalts zweier Speicherzellen.
- Transportiere in eine Speicherzelle eine Konstante.
- Führe Vergleich "≠" mit Inhalt einer Speicherzelle und einer Konstante aus.
- Führe Vergleich "≤" mit Inhalt zweier Speicherzellen aus.
- Ausgeben einer Speicherzelle

Vom Programm zum Computer
elementare Anweisungen

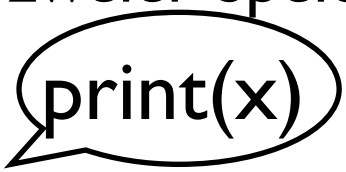
elementare Anweisungen am Programm identifizieren:

- Einlesen einer Zahl in eine Speicherzelle.
- Transportiere in eine Speicherzelle die Summe des Inhalts zweier Speicherzellen.
- Transportiere in eine Speicherzelle eine Konstante.
- Führe Vergleich "≠" mit Inhalt einer Speicherzelle und einer Konstante aus.
- Führe Vergleich "≤" mit Inhalt zweier Speicherzellen aus.
- Ausgeben einer Speicherzelle

Vom Programm zum Computer
elementare Anweisungen

elementare Anweisungen am Programm identifizieren:

- Einlesen einer Zahl in eine Speicherzelle.
- Transportiere in eine Speicherzelle die Summe des Inhalts zweier Speicherzellen.
- Transportiere in eine Speicherzelle eine Konstante.
- Führe Vergleich "≠" mit Inhalt einer Speicherzelle und einer Konstante aus.
- Führe Vergleich "≤" mit Inhalt zweier Speicherzellen aus.
- Ausgeben einer Speicherzelle



Vom Programm zum Computer
elementare Anweisungen

elementare Anweisungen am Programm identifizieren:

- Einlesen einer Zahl in eine Speicherzelle.
- Transportiere in eine Speicherzelle die Summe des Inhalts zweier Speicherzellen.
- Transportiere in eine Speicherzelle eine Konstante.
- Führe Vergleich "≠" mit Inhalt einer Speicherzelle und einer Konstante aus.
- Führe Vergleich "≤" mit Inhalt zweier Speicherzellen aus.
- Ausgeben einer Speicherzelle

Vom Programm zum Computer
Speicherzelle und Inhalt

- Beachte Schreibweisen:
  - x←0: x ist die Speicherzelle mit dem Namen x
  - "x=y": Inhalt von x und y, also eigentlich "«x»=«y»".

Vom Programm zum Computer
Konstruktoren

- Konkatenation → kann bleiben
- bedingte Schleife/Anweisung → Durchbrechen der normalen Reihenfolge (*Heraussprungen*) - eine Art Pfeile
  - Lösung durch Sprunganweisungen und Marken

Vom Programm zum Computer
Konstruktoren

Programm in PRO

```
def x, n, größer, kleiner: Zahl
größer←0
kleiner←0
lies(n)
lies(x)
solange x≠0 tue
    wenn x≤n dann
        kleiner←kleiner+x
    sonst
        größer←größer+x
    ende
    lies(x)
ende
zeige(größer)
zeige(kleiner).
```

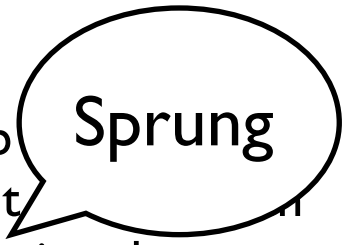
```
        größer←0
        kleiner←0
        get(n)
        get(x)
schleife: berechne, ob x≠0 ist
        falls das nicht zutrifft, dann
            weiter bei ende
        berechne, ob x≤n ist
        falls das zutrifft, dann
            kleiner←kleiner+x
        weiter bei next
        größer←größer+x
next:   get(x)
        weiter bei schleife
ende:   print(größer)
        print(kleiner).
```

Vom Programm zum Computer
Konstruktoren

Programm in PRO

```
def x, n, größer, kleiner: Zahl
größer←0
kleiner←0
lies(n)
lies(x)
solange x≠0 tue
    wenn x≤n dann
        kleiner←kleiner+x
    sonst
        größer←größer+x
    ende
    lies(x)
ende
zeige(größer)
zeige(kleiner).
```

```
        größer←0
        kleiner←0
        get(n)
        get(x)
schleife: berechne, ob
        falls das nicht
            weiter bei ende
        berechne, ob x≤n ist
        falls das zutrifft, dann
            kleiner←kleiner+x
        weiter bei next
        größer←größer+x
next:   get(x)
        weiter bei schleife
ende:   print(größer)
        print(kleiner).
```



Vom Programm zum Computer
Konstruktoren

Programm in PRO

```
def x, n, größer, kleiner: Zahl
größer←0
kleiner←0
lies(n)
lies(x)
solange x≠0 tue
    wenn x≤n dann
        kleiner←kleiner+x
    sonst
        größer←größer+x
    ende
    lies(x)
ende
zeige(größer)
zeige(kleiner).
```

```
        größer←0
        kleiner←0
        get(n)
        get(x)
schleife: berechne, ob
        falls das nicht
            weiter bei ende
        berechne, ob x≤n ist
        falls das zutrifft, dann
            ner←kleiner+x
            ei next
            ←größer+x
next:   get(x)
        weiter bei schleife
ende:   print(größer)
        print(kleiner).
```

Sprung

Marke

Vom Programm zum Computer
Konstruktoren

Programm in PRO

```
def x, n, größer, kleiner: Zahl
größer←0
kleiner←0
lies(n)
lies(x)
solange x≠0 tue
    wenn x≤n dann
        kleiner←kleiner+x
    sonst
        größer←größer+x
    ende
    lies(x)
ende
zeige(größer)
zeige(kleiner).
```

```
        größer←0
        kleiner←0
        get(n)
        get(x)
schleife: berechne, ob x≠0 ist
        falls das nicht zutrifft, dann
            weiter bei ende
        berechne, ob x≤n ist
        falls das zutrifft, dann
            kleiner←kleiner+x
            weiter bei next
        größer←größer+x
next:   get(x)
        weiter bei schleife
ende:   print(größer)
        print(kleiner).
```

Vom Programm zum Computer
Konstruktoren

Programm in PRO

```
def x, n, größer, kleiner: Zahl
größer←0
kleiner←0
lies(n)
lies(x)
solange x≠0 tue
    wenn x≤n dann
        kleiner←kleiner+x
    sonst
        größer←größer+x
    ende
    lies(x)
ende
zeige(größer)
zeige(kleiner).
```

```
        größer←0
        kleiner←0
        get(n)
        get(x)
schleife: berechne, ob x≠0 ist
        falls das nicht zutrifft, dann
            weiter bei ende
        berechne, ob x≤n ist
        falls das zutrifft, dann
            kleiner←kleiner+x
        weiter bei next
        größer←größer+x
next:   get(x)
        weiter bei schleife
ende:   print(größer)
        print(kleiner).
```

Vom Programm zum Computer
Konstruktoren

Programm in PRO

```
def x, n, größer, kleiner: Zahl
größer←0
kleiner←0
```

Ergebnis:
Programm besteht nur noch aus aneinandergereihten Bedingungen, Wertzuweisungen, Sprüngen

```
zeige(kleiner).
```

```
        größer←0
        kleiner←0
        get(n)
        get(x)
schleife: berechne, ob x≠0 ist
        falls das nicht zutrifft, dann
              weiter bei ende
        berechne, ob x≤n ist
        falls das zutrifft, dann
              kleiner←kleiner+x
        weiter bei next
        größer←größer+x
next:   get(x)
        weiter bei schleife
ende:   print(größer)
        print(kleiner).
```

Vom Programm zum Computer
Maschinenaufbau

- Wie sieht eine Maschine aus, die dieses einfachere Programm ausführen kann?
- Daten**speicher**: klar
- **Rechenwerk** für arithmetische Ausdrücke
- **Steuerwerk**: korrekte Abarbeitung des Programms
- **Programmzettel**: Speicher für Programm
- **Ein-/Ausgabeeinheit**: für Bildschirm, Tastatur

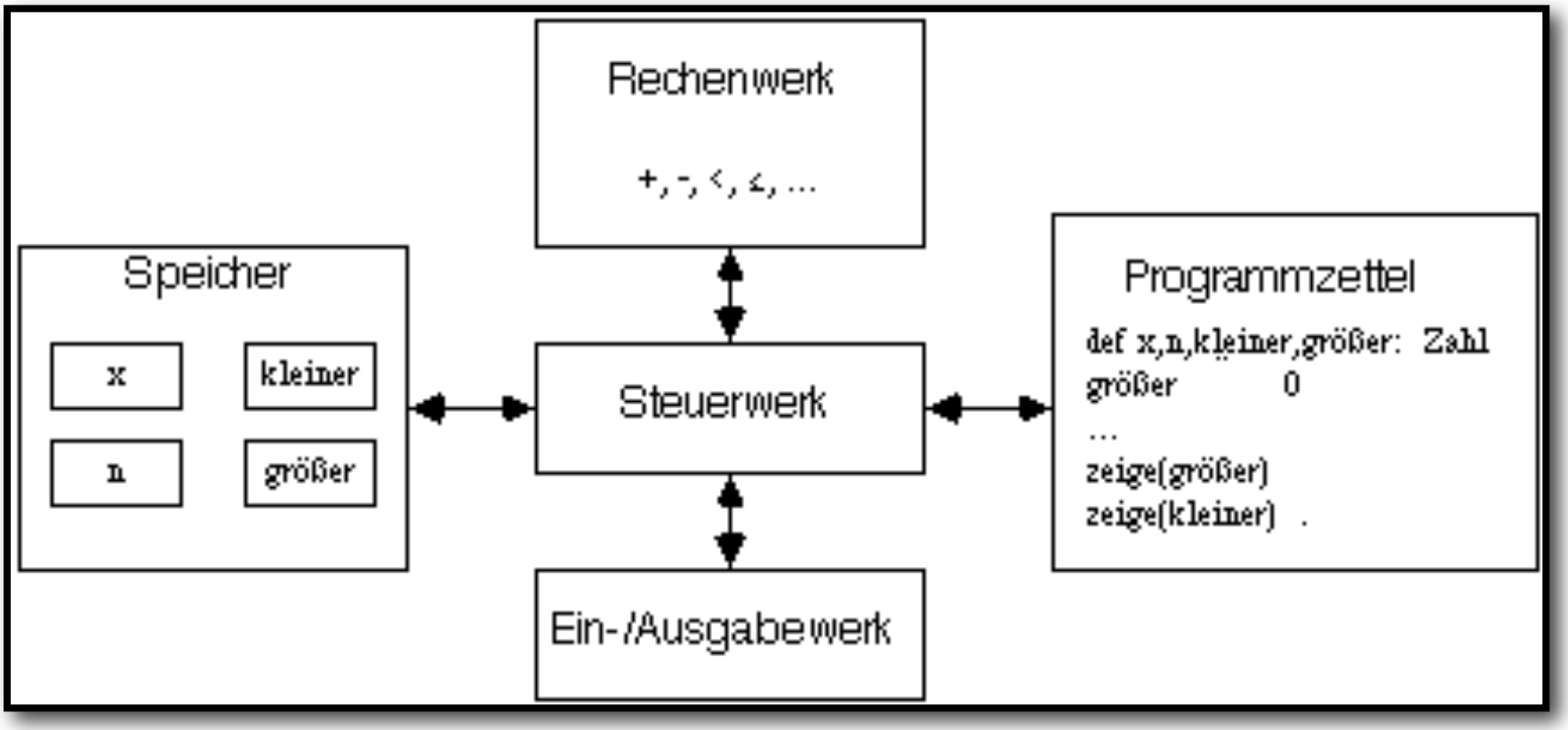
Vom Programm zum Computer
Maschinenaufbau

- Wie sieht eine Maschine aus, die dieses einfachere Programm ausführen kann?
- Daten**speicher**: klar
- **Rechenwerk** für arit e
- **Steuerwerk**: korrekt Programms
- **Programmzettel**: Speicher für Programm
- **Ein-/Ausgabeeinheit**: für Bildschirm, Tastatur

Vom Programm zum Computer

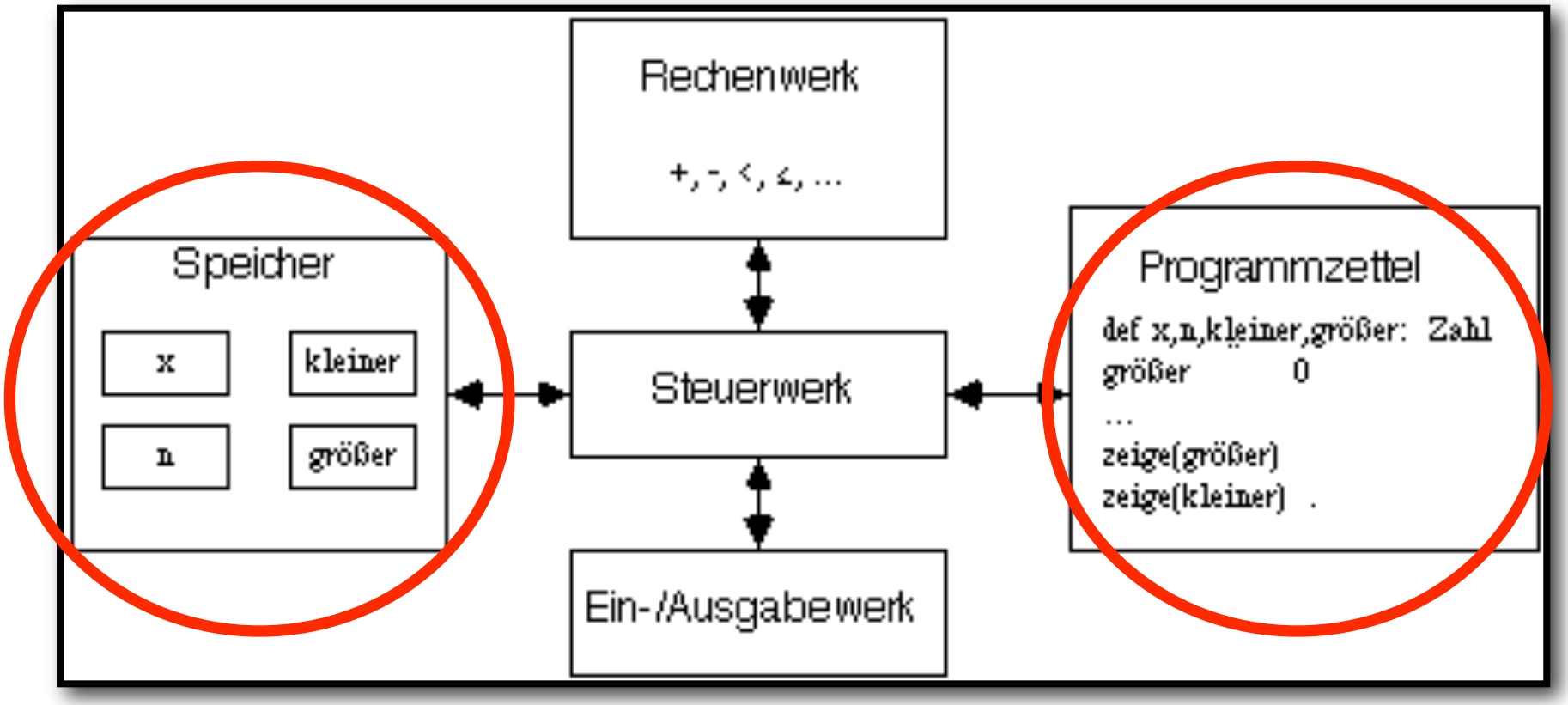
Maschinenaufbau

Vom Programm zum Computer
Maschinenaufbau

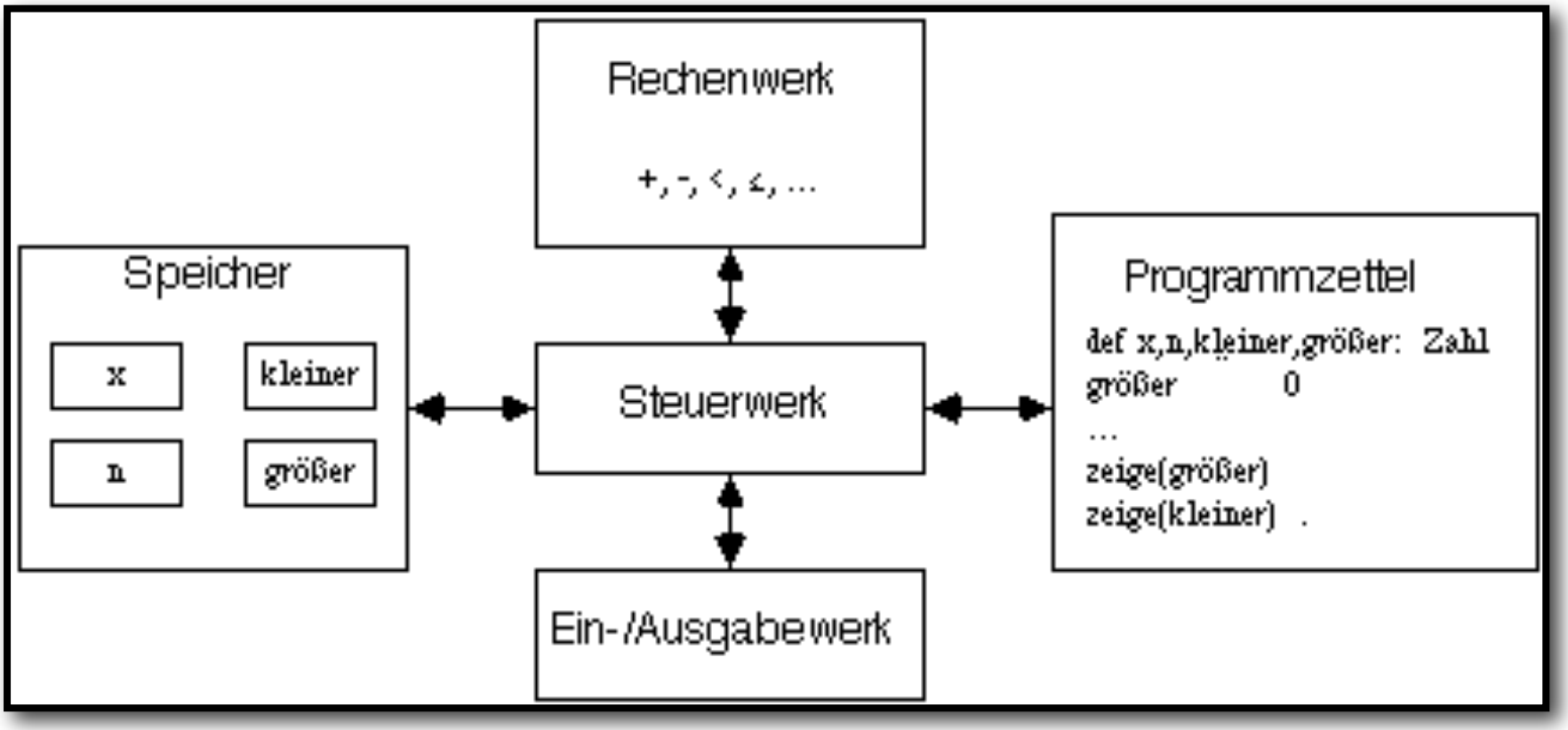


Vom Programm zum Computer

Maschinenaufbau



Vom Programm zum Computer
Maschinenaufbau

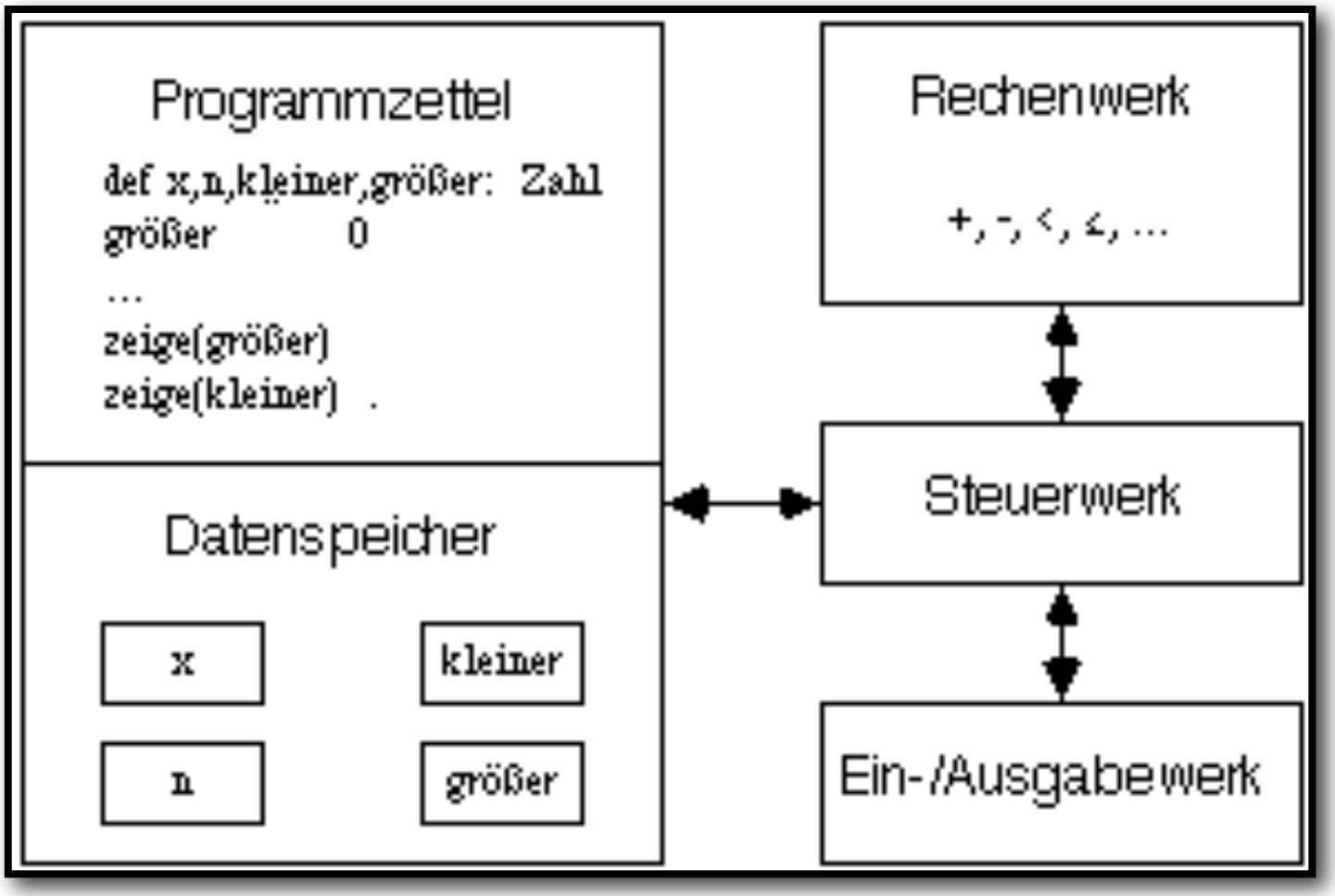


Vom Programm zum Computer

Maschinenaufbau

Vom Programm zum Computer

Maschinenaufbau



Vom Programm zum Computer
Ergebnis

- Zeige: alle Sprachelemente von PRO mit drei (evtl. mit Marke versehenen) Grundbefehlen "Vergleich", "Wertzuweisung" und "bedingter/unbedingter Sprung" simulierbar
- einziger Konstruktor: Konkatenation
- Prozeduraufrufe problematischer: Lösung mit Hilfe einer besonderen Datenstruktur, dem sog. **Keller** (später)

Vom Programm zum Computer
Rechenwerk

- Auswertung bel. langer arithmetischer Ausdrücke nicht möglich → Zerlegung langer Ausdrücke in einfachere
- Beispiel:

  x←a+27*(a+b)-c*d

  Vereinfachung:

```
h1←a+b;
h2←27*h1;
h3←a+h2;
h4←c*d;
x←h3-h4.
```

oder noch einfacher:
spart Hilfsbezeichner

```
h1←a+b;
h1←27*h1;
h1←a+h1;
x←c*d;
x←h1-x.
```

Vom Programm zum Computer
Vereinfachung von Ausdrücken

Es gilt: Jede Wertzuweisung der Form

$$x \leftarrow \text{Ausdruck}$$

läßt sich durch eine Folge von Wertzuweisungen der Form

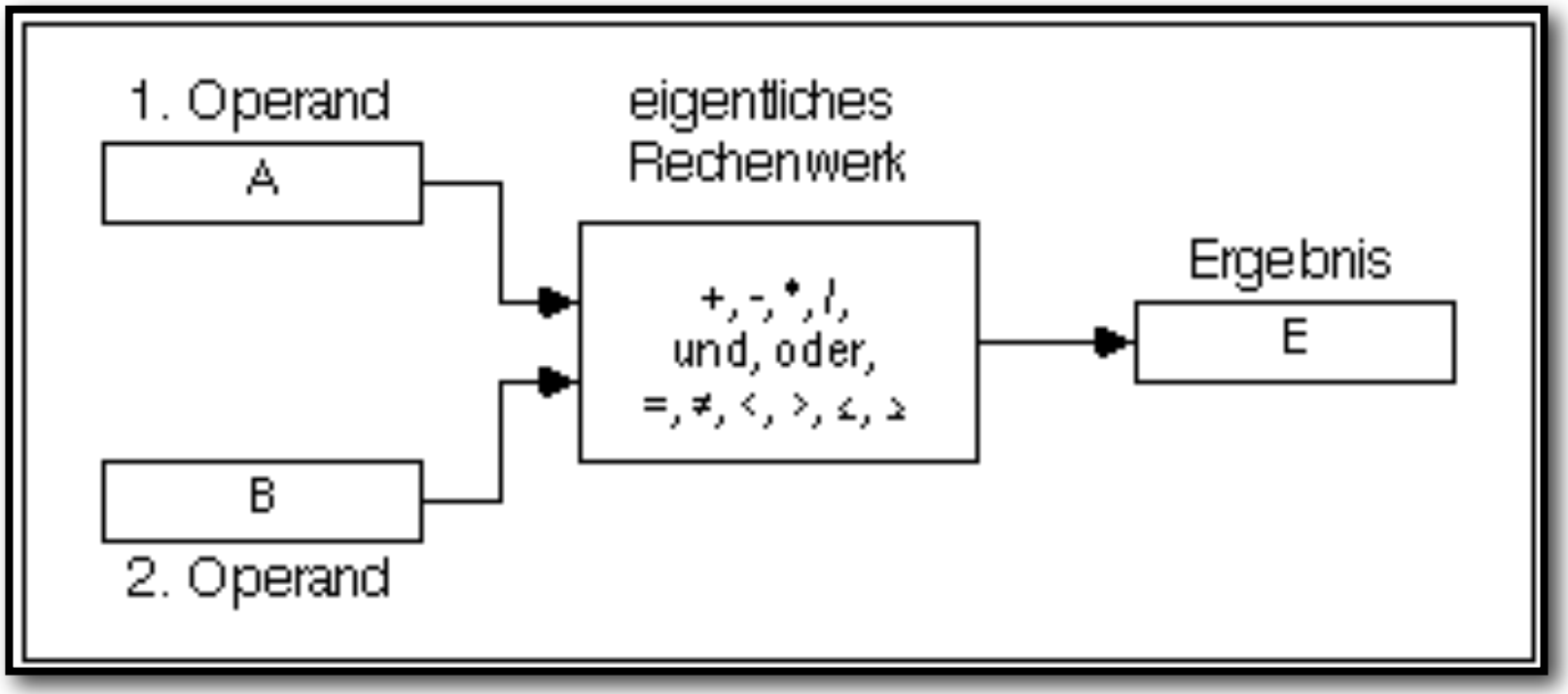
$$u \leftarrow v \quad \text{oder} \quad u \leftarrow v \text{ op } w$$

ersetzen, wobei u ein Bezeichner, v und w Bezeichner oder Konstanten sind und op genau einen Operator (z.B. +, -, *, /) bezeichnet.

Die Umwandlung geht *algorithmisch*, also *maschinell*.

Vom Programm zum Computer
Vereinfachung des Rechenwerks

- Rechenwerk kann mit 3 Speicherzellen auskommen

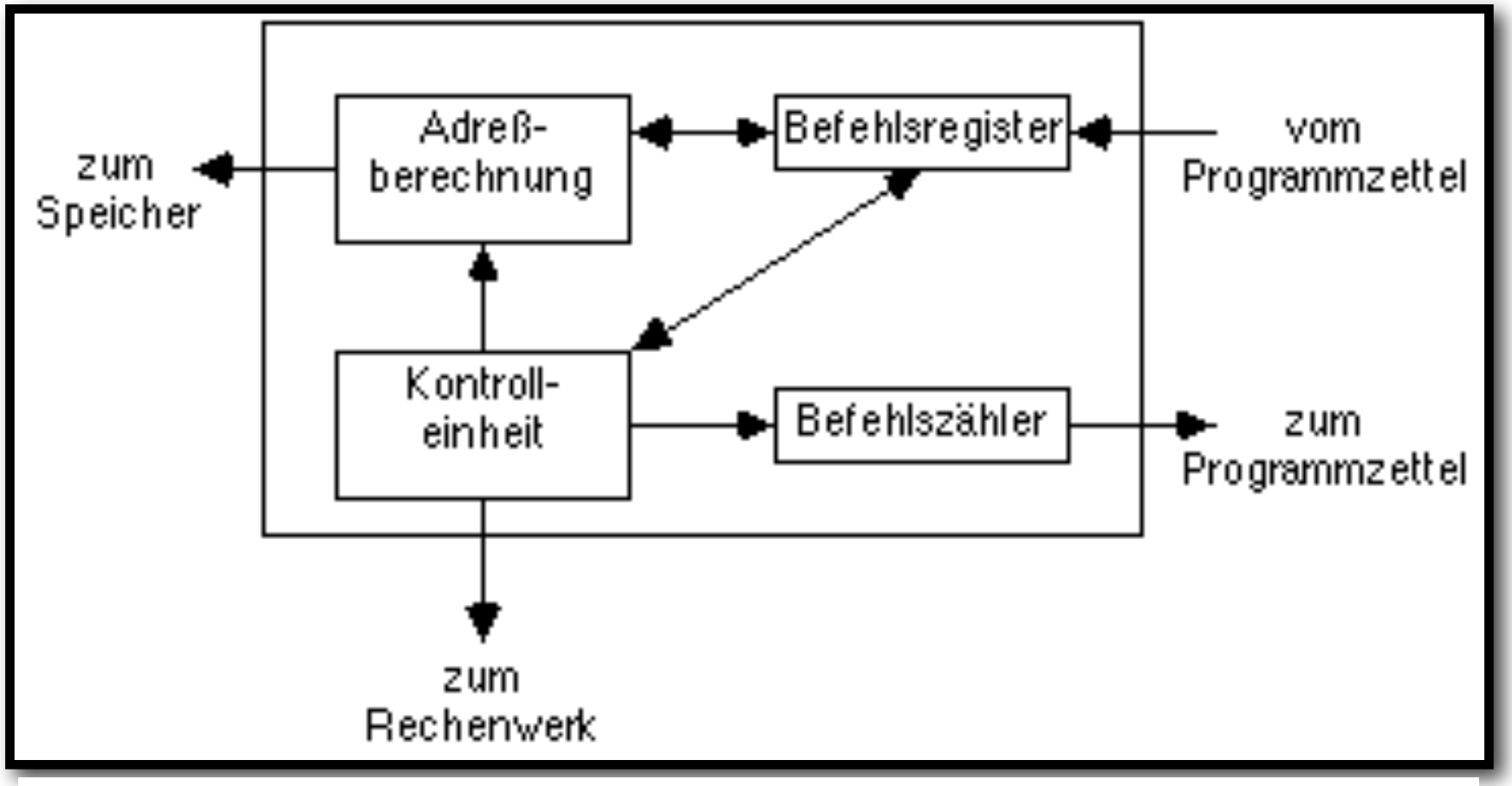


Vom Programm zum Computer
Steuerwerk

Aufgaben

- Elementaranweisung (**Befehl**) vom Programmzettel lesen und im *Befehlsregister* speichern
- nächsten Befehl ermitteln (Nachfolger oder Sprungziel); Befehle des Programms mit 1,2,3 usw. durchnumerieren; *Befehlszähler* enthält Nummer des aktuell bearbeiteten Befehls
- Operanden vom Speicher zum Rechenwerk und Ergebnisse zurück in den Speicher transportieren (*Lesen* und *Schreiben*)
- Speicherzellen, in denen sich Operanden befinden, ermitteln (*Adreßberechnung*).

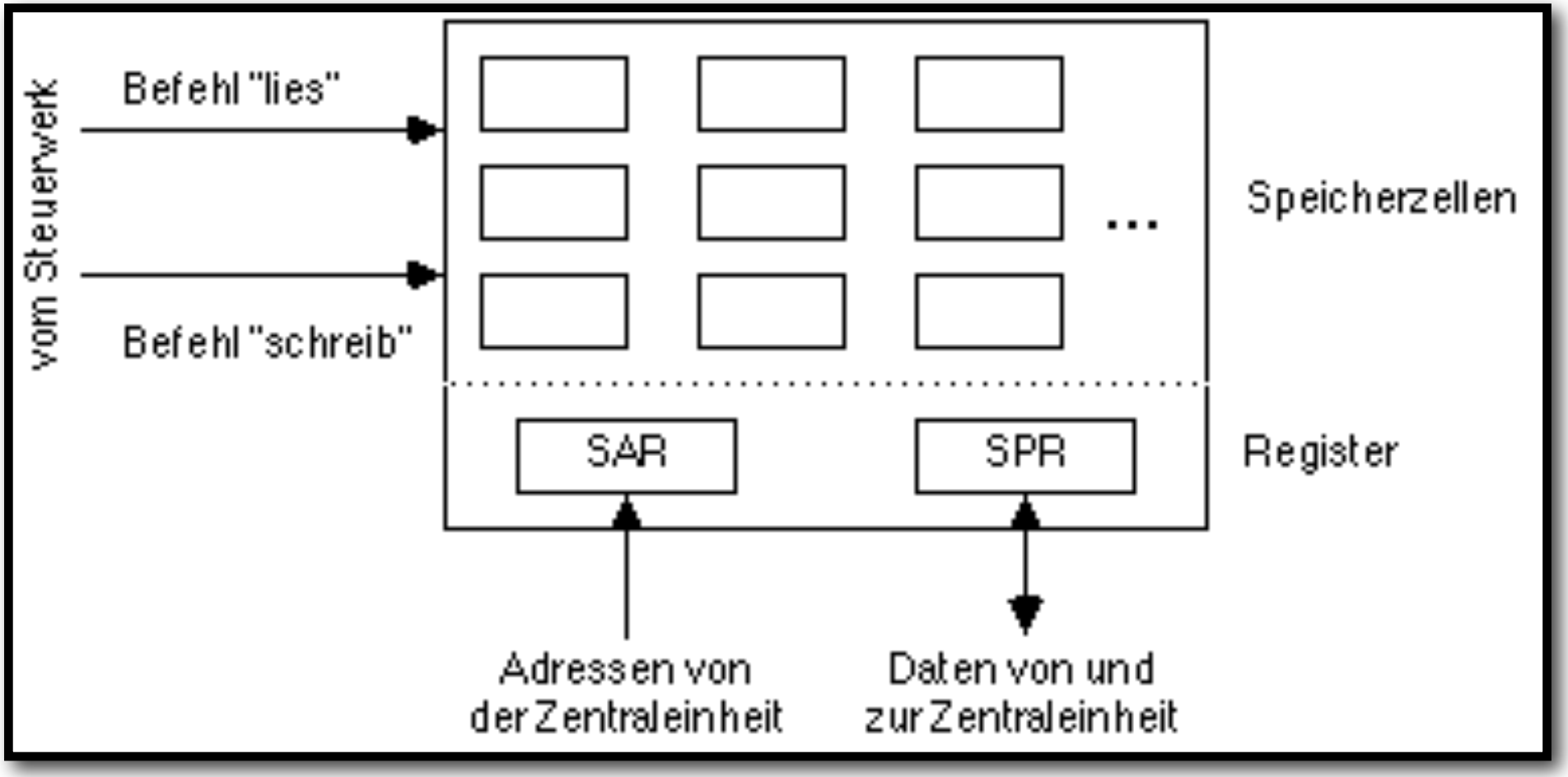
Vom Programm zum Computer
Architektur des Steuerwerks



Speicherzellen, die einem speziellen Zweck dienen, bezeichnet man als **Register**.

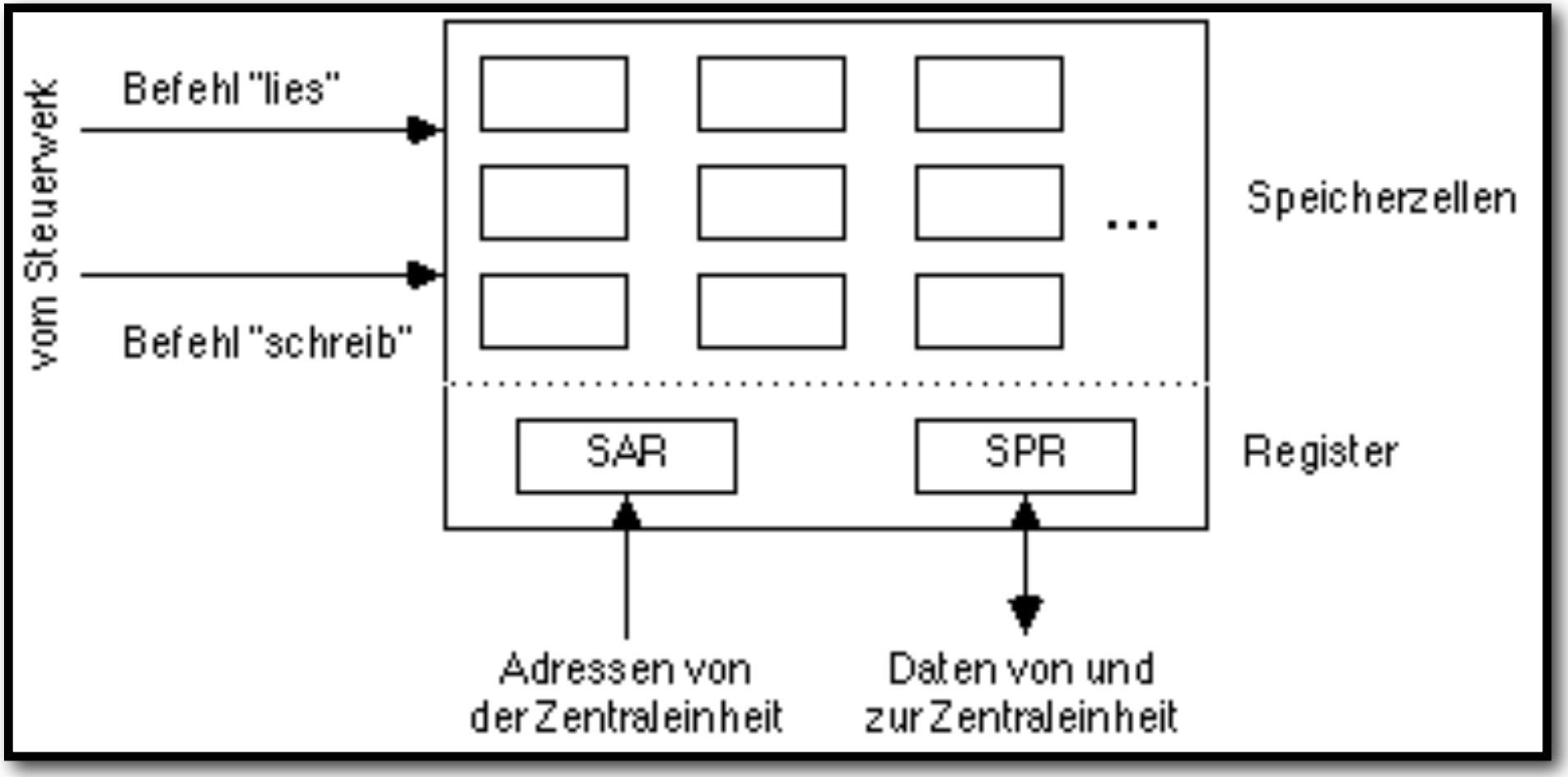
Vom Programm zum Computer

Speicher



Vom Programm zum Computer

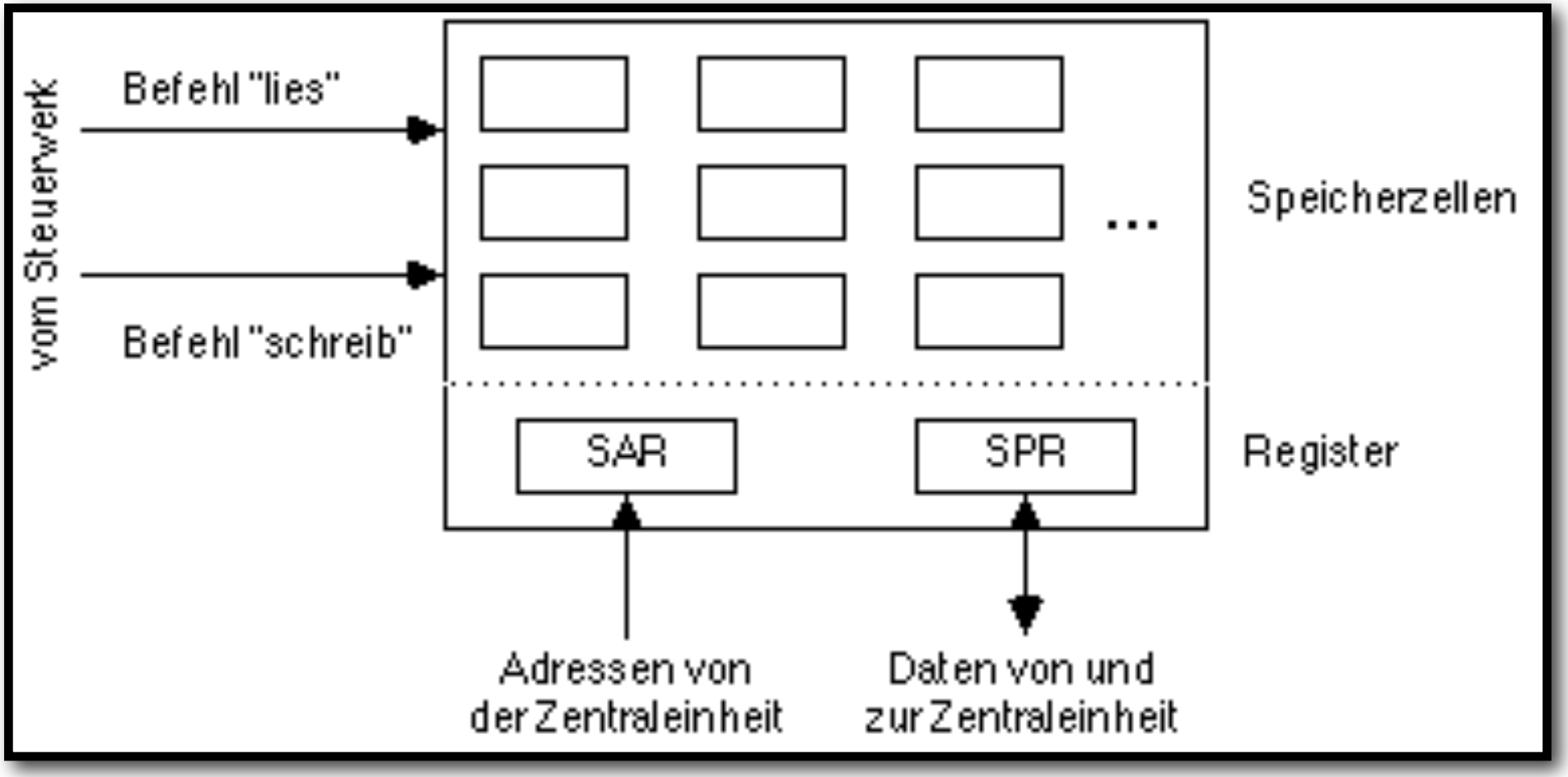
Speicher



Wertzuweisung x←y

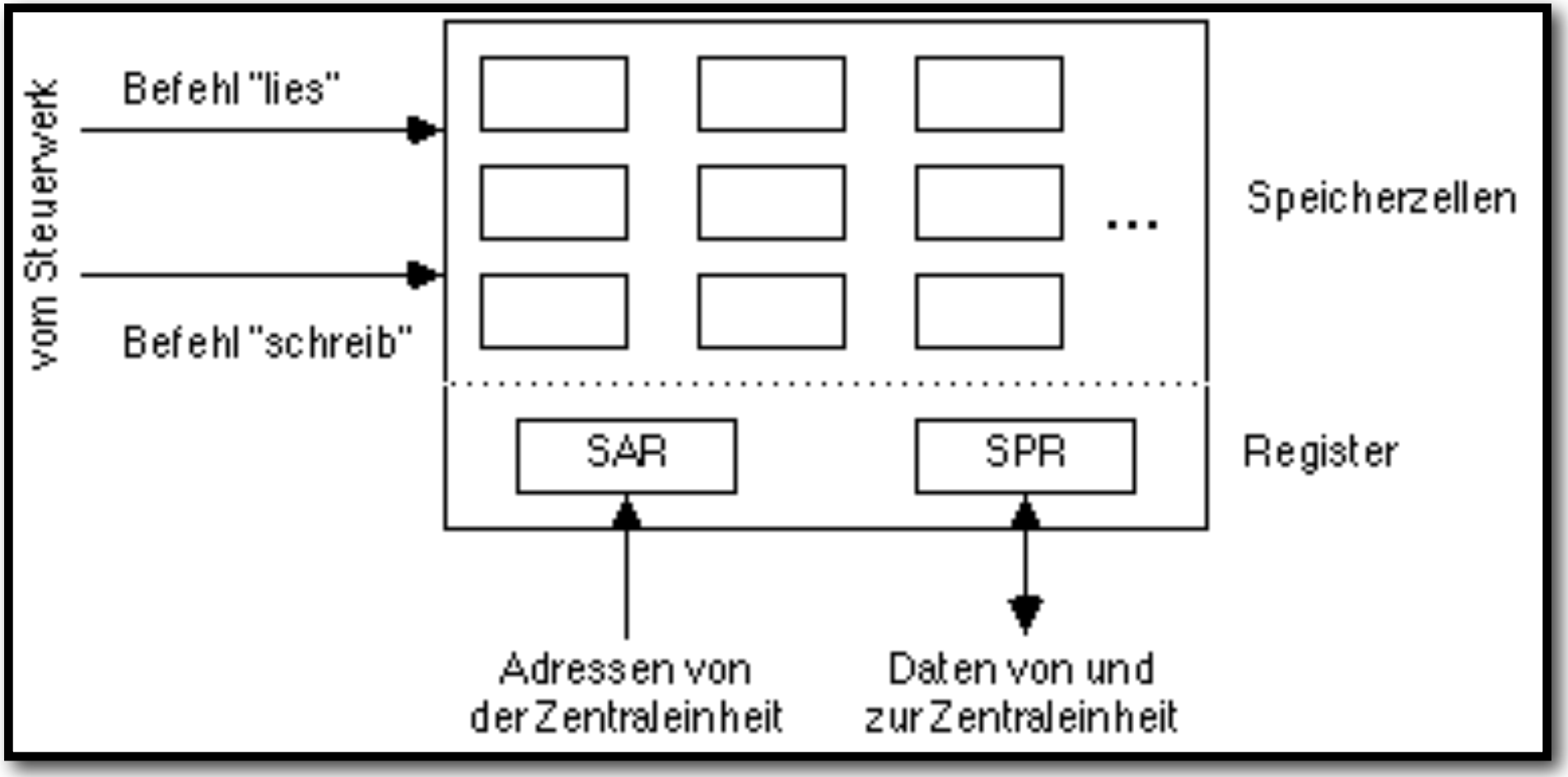
Vom Programm zum Computer

Speicher



Vom Programm zum Computer

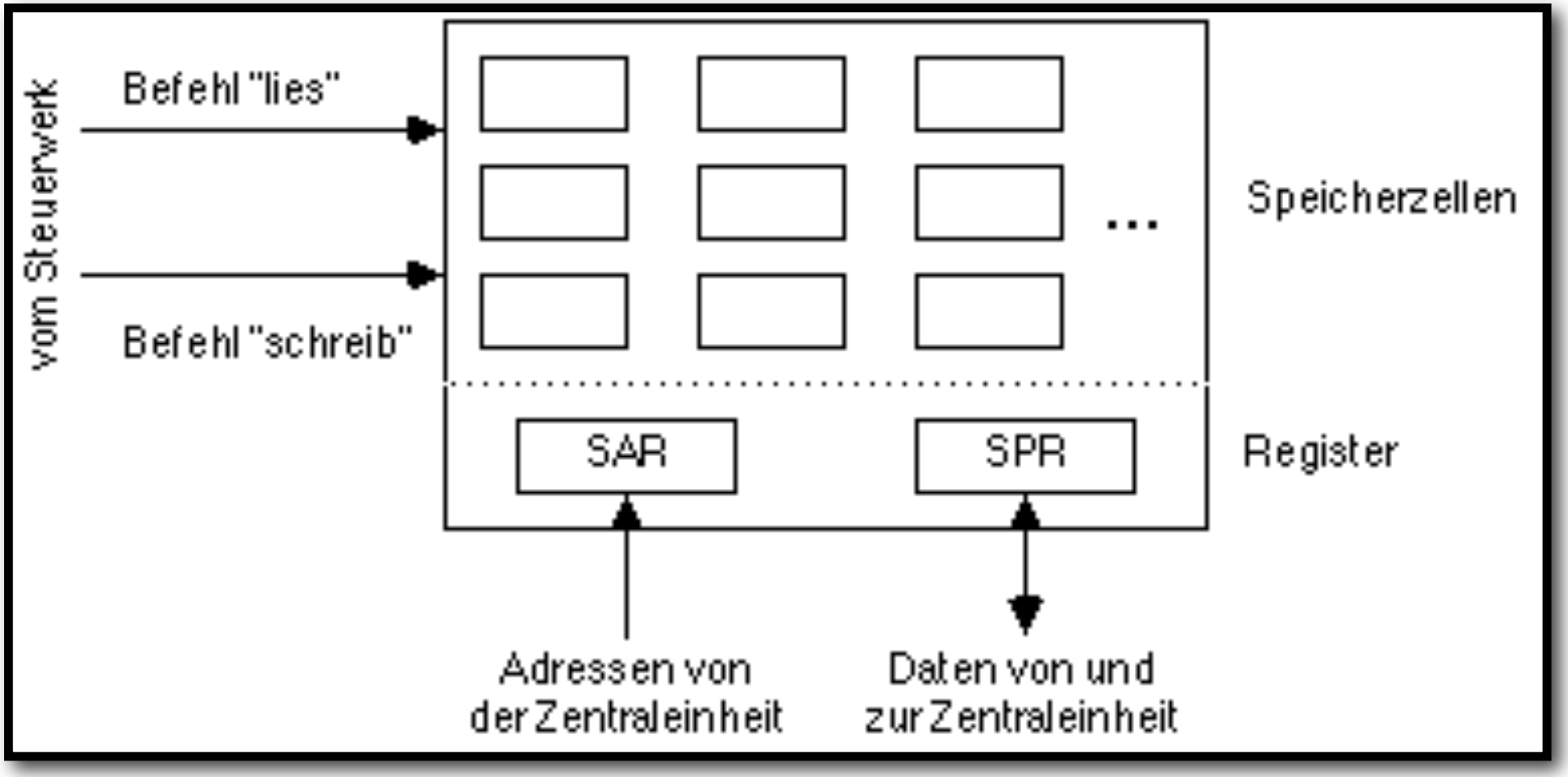
Speicher



Berechne Adresse der Speicherzelle, die zur Variablen y gehört

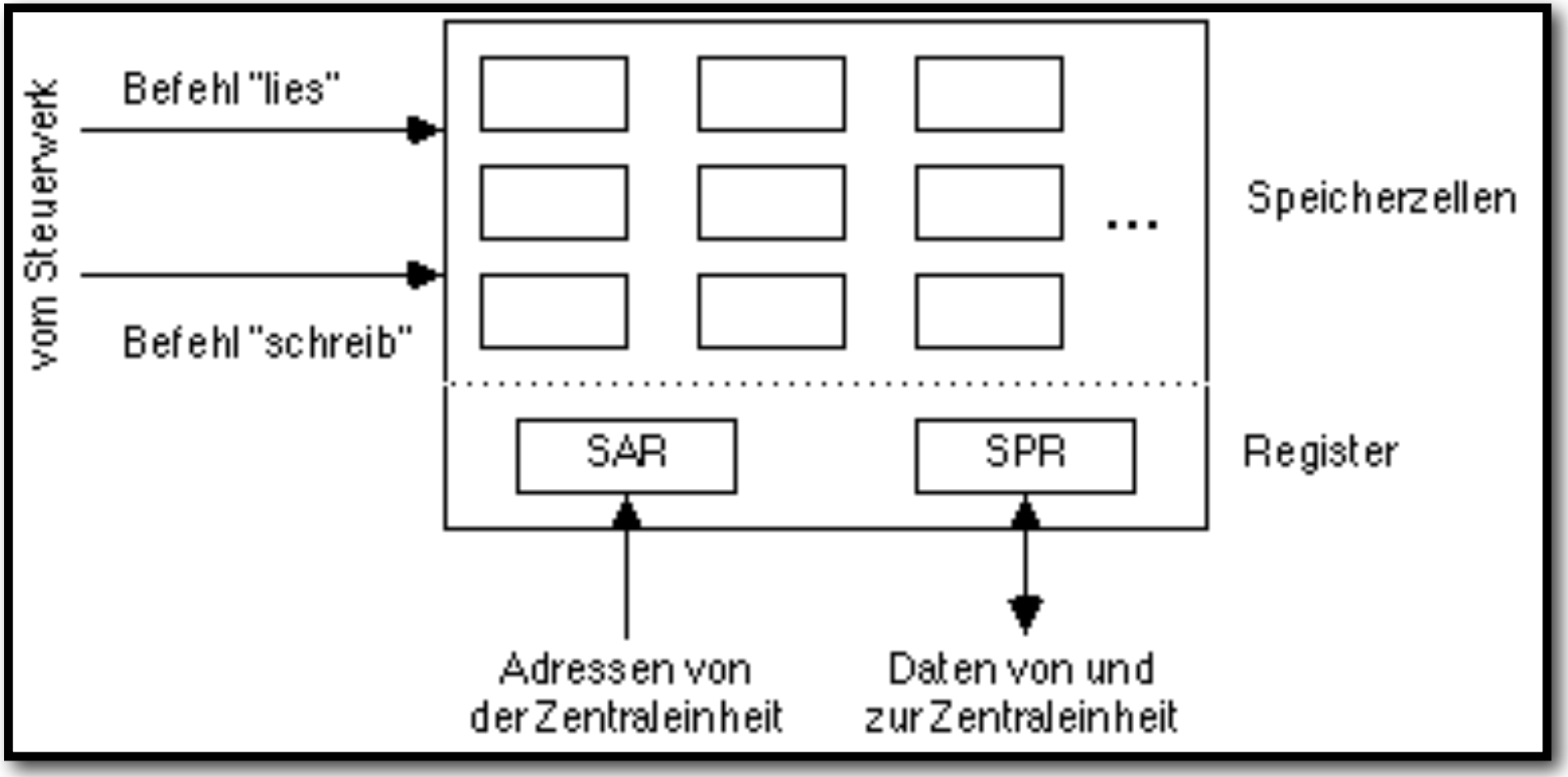
Vom Programm zum Computer

Speicher



Vom Programm zum Computer

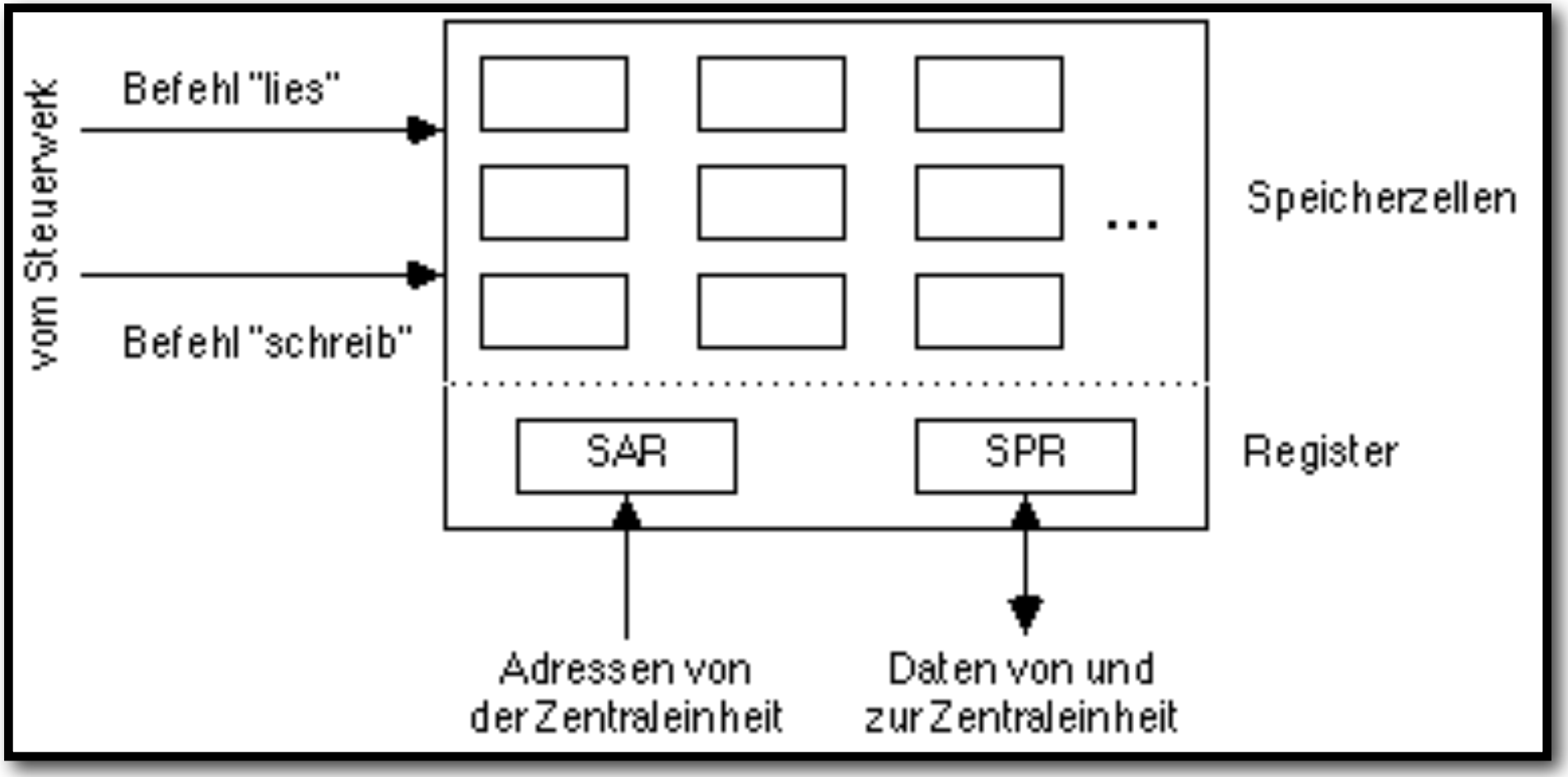
Speicher



Schreibe diese Adresse ins Register SAR

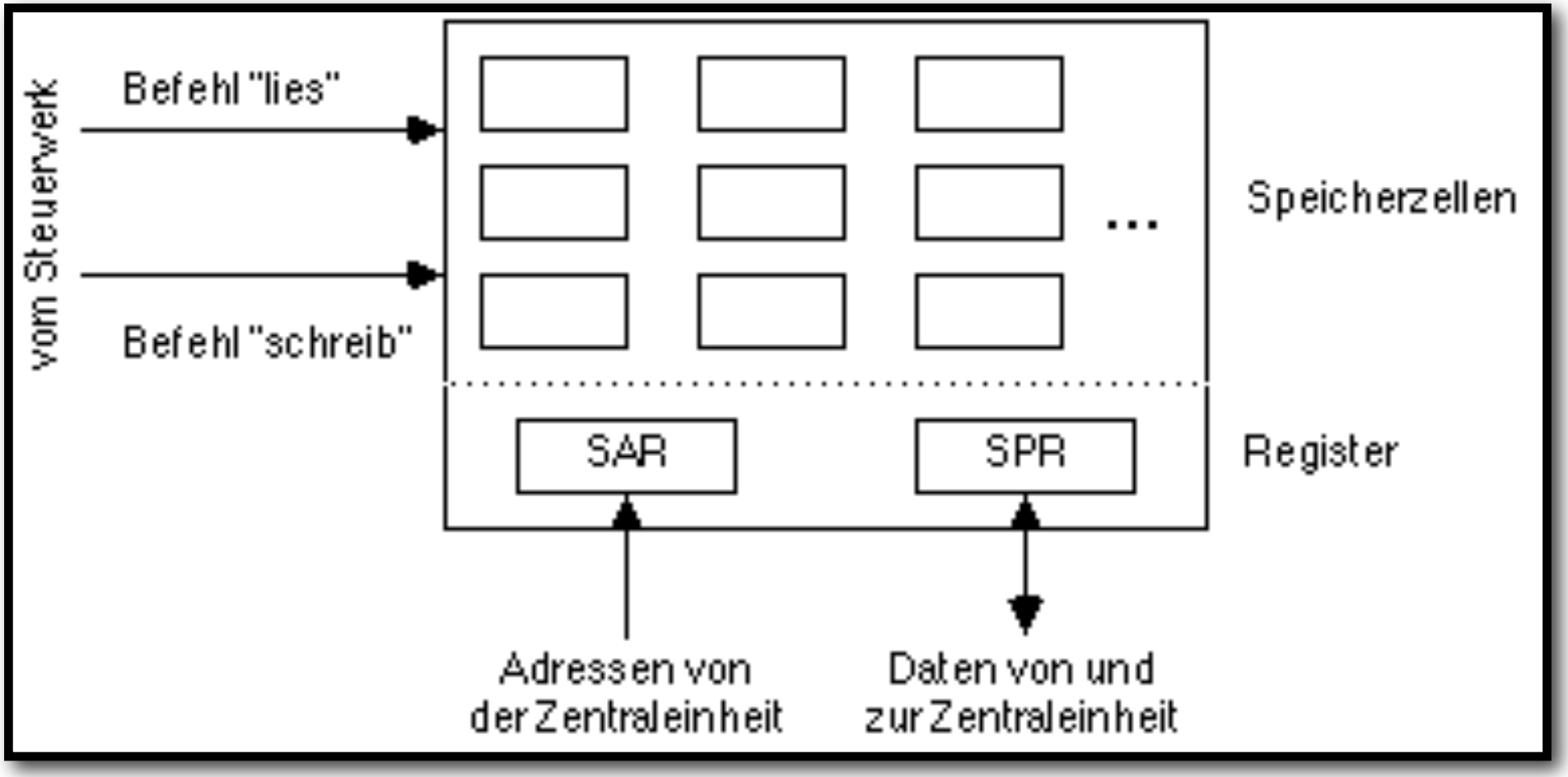
Vom Programm zum Computer

Speicher



Vom Programm zum Computer

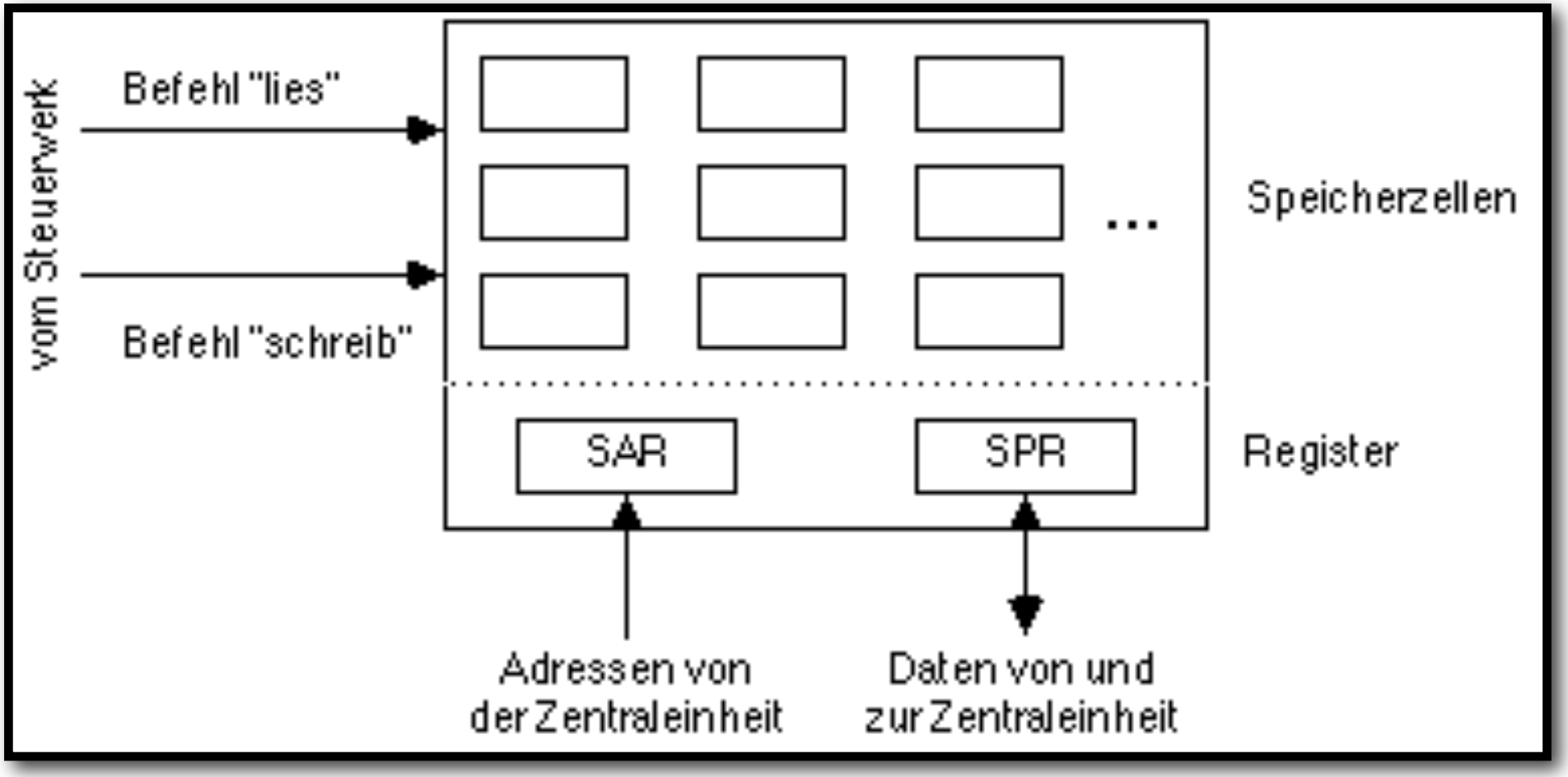
Speicher



Stoße Speicher durch "lies" an (danach steht Wert von y im SPR).

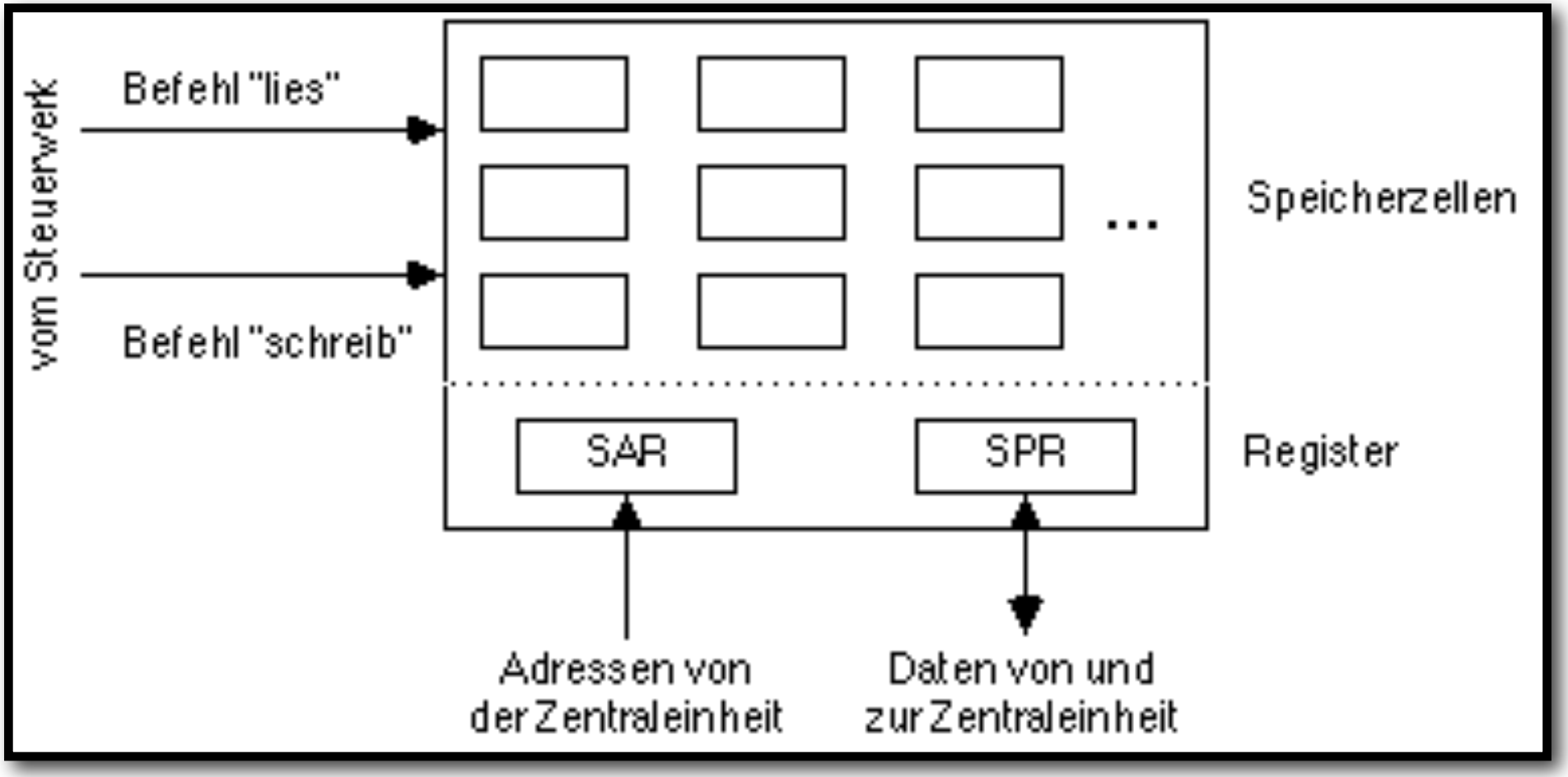
Vom Programm zum Computer

Speicher



Vom Programm zum Computer

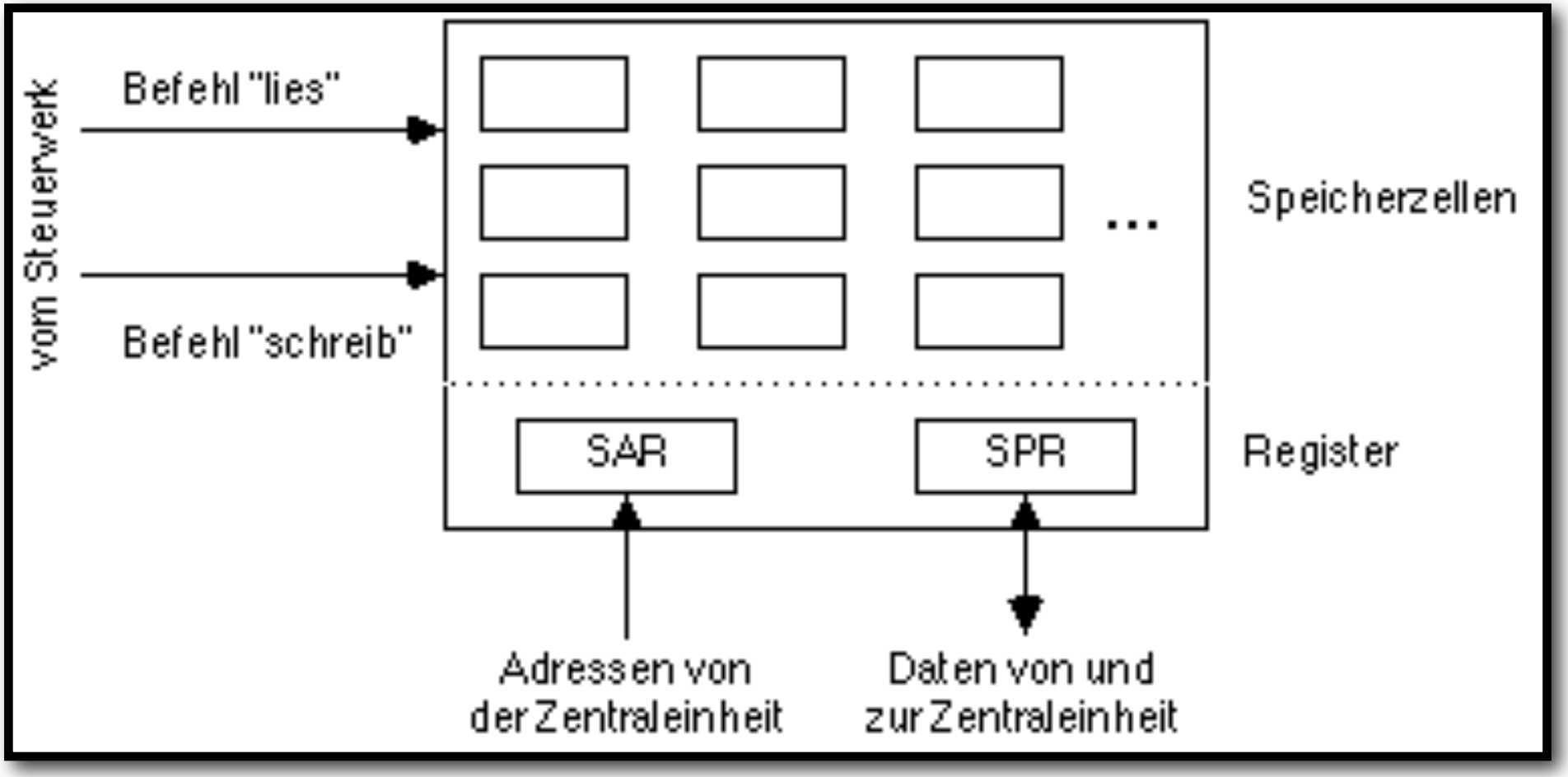
Speicher



Berechne Adresse der Speicherzelle, die zur Variablen x gehört

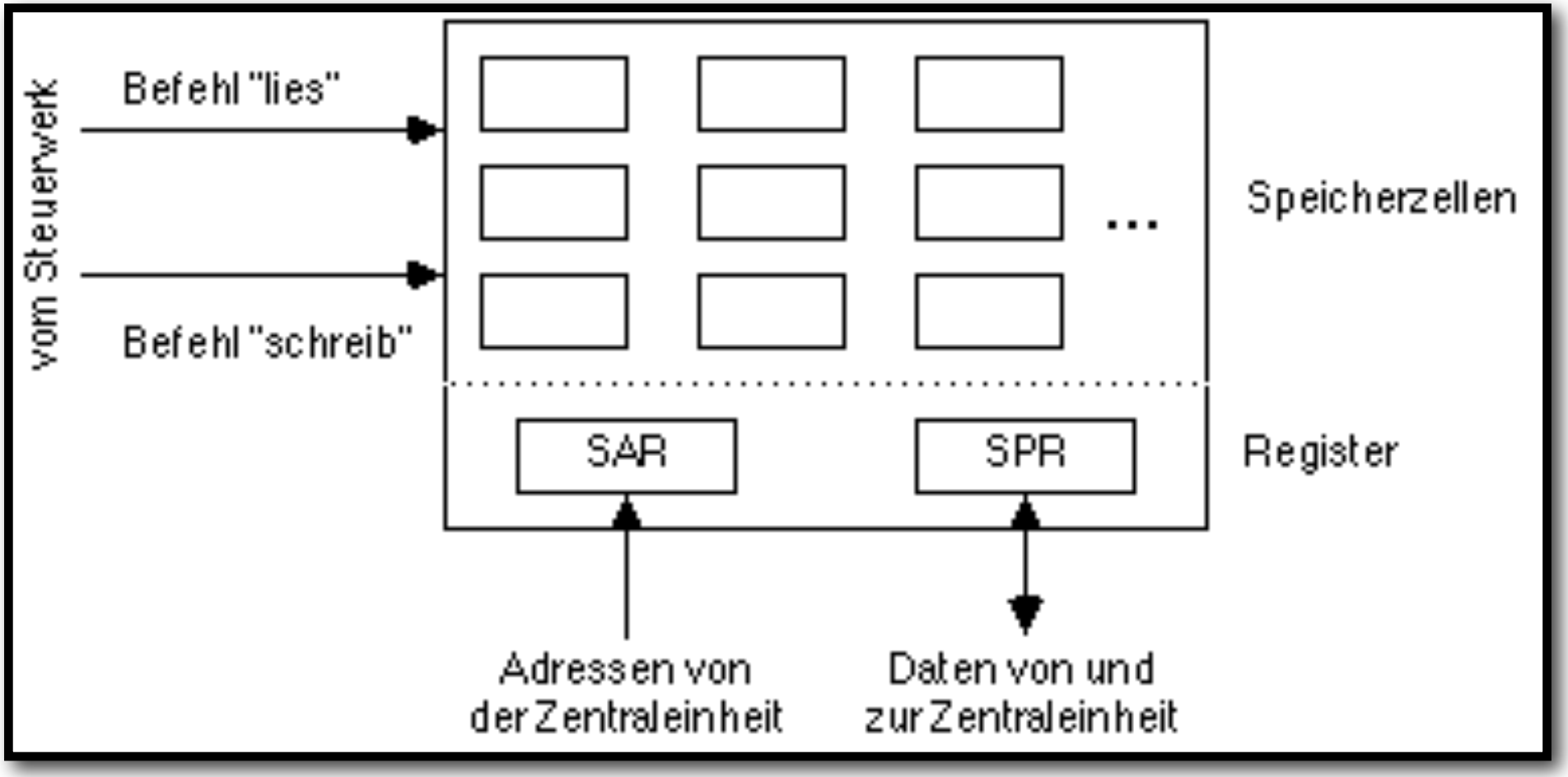
Vom Programm zum Computer

Speicher



Vom Programm zum Computer

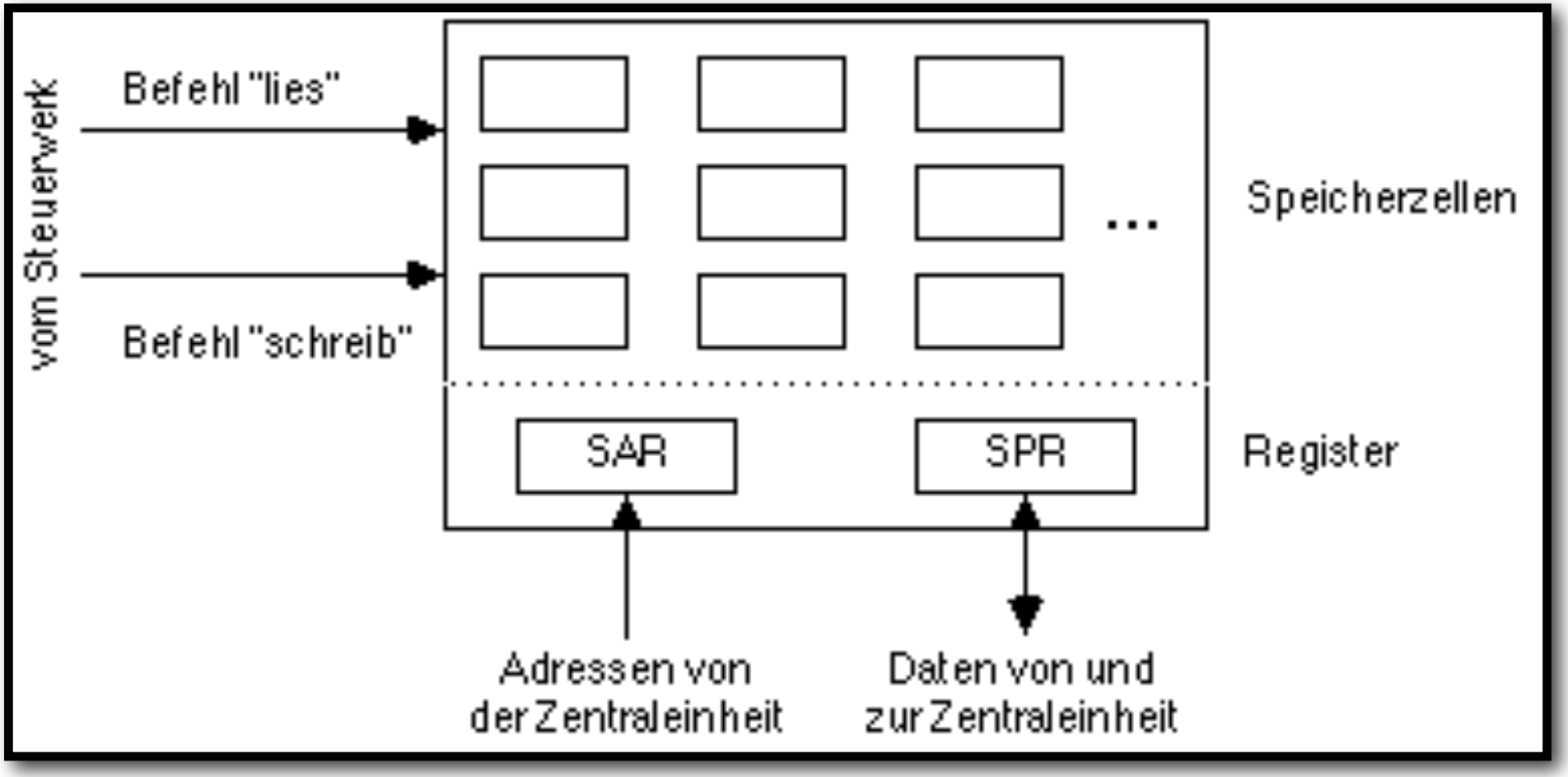
Speicher



Schreibe diese Adresse ins Register SAR

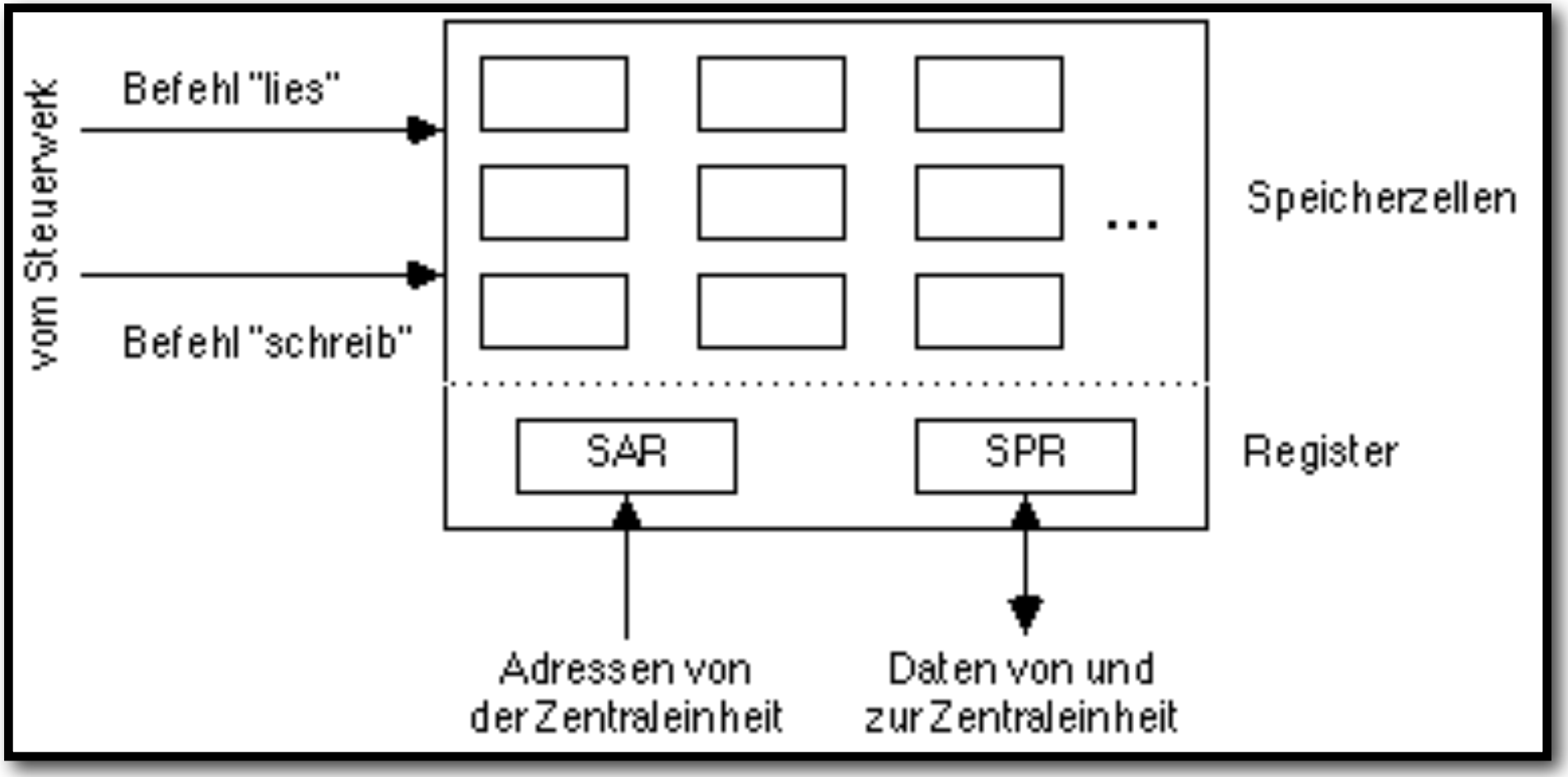
Vom Programm zum Computer

Speicher



Vom Programm zum Computer

Speicher

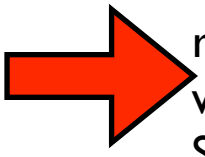


Stoße Speicher durch "schreib" an (y wird durch SPR ersetzt).

Vom Programm zum Computer
maschinennahe Programmierung

read	Speicherzelle, deren Adresse in SAR angegeben ist, nach SPR
write	Inhalt von SPR in Speicherzelle, deren Adresse in SAR angegeben ist
SPR→A	Inhalt des SPR in das Register A des Rechenwerks
A→SPR	Inhalt von Register A des Rechenwerks nach SPR
B→A	Inhalt von Register B des Rechenwerks in Register A
E→A	Inhalt von Register E des Rechenwerks in Register A
n→A	ganzzahlige Konstante n in Register A
add	Addiere A und B und schreibe Ergebnis nach E
sub	Subtrahiere B von A und schreibe Ergebnis nach E
less	Ermittle «A»<«B» und schreibe Ergebnis (wahr oder falsch) nach E
lesseq	Ermittle «A»≤«B» und schreibe Ergebnis (wahr oder falsch) nach E
iftruegoto M	Setze die Verarbeitung bei dem mit M markierten Befehl fort, sofern in E der Wert wahr steht, anderenfalls gehe zum nächsten Befehl
goto M	Setze die Verarbeitung bei der mit M markierten Anweisung fort
get	Eingabe eines Datums in das Register A
print	Drucke den Inhalt des Registers A
print "..."	Drucke den Text zwischen den Anführungszeichen
stop	Ende der Berechnung.

Vom Programm zum Computer
maschinennahe Programmierung

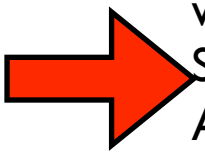


read	Speicherzelle, deren Adresse in SAR angegeben ist, nach SPR
write	Inhalt von SPR in Speicherzelle, deren Adresse in SAR angegeben ist
SPR→A	Inhalt des SPR in das Register A des Rechenwerks
A→SPR	Inhalt von Register A des Rechenwerks nach SPR
B→A	Inhalt von Register B des Rechenwerks in Register A
E→A	Inhalt von Register E des Rechenwerks in Register A
n→A	ganzzahlige Konstante n in Register A
add	Addiere A und B und schreibe Ergebnis nach E
sub	Subtrahiere B von A und schreibe Ergebnis nach E
less	Ermittle «A»<«B» und schreibe Ergebnis (wahr oder falsch) nach E
lesseq	Ermittle «A»≤«B» und schreibe Ergebnis (wahr oder falsch) nach E
iftruegoto M	Setze die Verarbeitung bei dem mit M markierten Befehl fort, sofern in E der Wert wahr steht, anderenfalls gehe zum nächsten Befehl
goto M	Setze die Verarbeitung bei der mit M markierten Anweisung fort
get	Eingabe eines Datums in das Register A
print	Drucke den Inhalt des Registers A
print "..."	Drucke den Text zwischen den Anführungszeichen
stop	Ende der Berechnung.

Vom Programm zum Computer
maschinennahe Programmierung

read	Speicherzelle, deren Adresse in SAR angegeben ist, nach SPR
write	Inhalt von SPR in Speicherzelle, deren Adresse in SAR angegeben ist
SPR→A	Inhalt des SPR in das Register A des Rechenwerks
A→SPR	Inhalt von Register A des Rechenwerks nach SPR
B→A	Inhalt von Register B des Rechenwerks in Register A
E→A	Inhalt von Register E des Rechenwerks in Register A
n→A	ganzzahlige Konstante n in Register A
add	Addiere A und B und schreibe Ergebnis nach E
sub	Subtrahiere B von A und schreibe Ergebnis nach E
less	Ermittle «A»<«B» und schreibe Ergebnis (wahr oder falsch) nach E
lesseq	Ermittle «A»≤«B» und schreibe Ergebnis (wahr oder falsch) nach E
iftruegoto M	Setze die Verarbeitung bei dem mit M markierten Befehl fort, sofern in E der Wert wahr steht, anderenfalls gehe zum nächsten Befehl
goto M	Setze die Verarbeitung bei der mit M markierten Anweisung fort
get	Eingabe eines Datums in das Register A
print	Drucke den Inhalt des Registers A
print "..."	Drucke den Text zwischen den Anführungszeichen
stop	Ende der Berechnung.

Vom Programm zum Computer
maschinennahe Programmierung

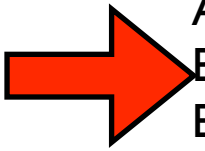


read	Speicherzelle, deren Adresse in SAR angegeben ist, nach SPR
write	Inhalt von SPR in Speicherzelle, deren Adresse in SAR angegeben ist
SPR→A	Inhalt des SPR in das Register A des Rechenwerks
A→SPR	Inhalt von Register A des Rechenwerks nach SPR
B→A	Inhalt von Register B des Rechenwerks in Register A
E→A	Inhalt von Register E des Rechenwerks in Register A
n→A	ganzzahlige Konstante n in Register A
add	Addiere A und B und schreibe Ergebnis nach E
sub	Subtrahiere B von A und schreibe Ergebnis nach E
less	Ermittle «A»<«B» und schreibe Ergebnis (wahr oder falsch) nach E
lesseq	Ermittle «A»≤«B» und schreibe Ergebnis (wahr oder falsch) nach E
iftruegoto M	Setze die Verarbeitung bei dem mit M markierten Befehl fort, sofern in E der Wert wahr steht, anderenfalls gehe zum nächsten Befehl
goto M	Setze die Verarbeitung bei der mit M markierten Anweisung fort
get	Eingabe eines Datums in das Register A
print	Drucke den Inhalt des Registers A
print "..."	Drucke den Text zwischen den Anführungszeichen
stop	Ende der Berechnung.

Vom Programm zum Computer
maschinennahe Programmierung

read	Speicherzelle, deren Adresse in SAR angegeben ist, nach SPR
write	Inhalt von SPR in Speicherzelle, deren Adresse in SAR angegeben ist
SPR→A	Inhalt des SPR in das Register A des Rechenwerks
A→SPR	Inhalt von Register A des Rechenwerks nach SPR
B→A	Inhalt von Register B des Rechenwerks in Register A
E→A	Inhalt von Register E des Rechenwerks in Register A
n→A	ganzzahlige Konstante n in Register A
add	Addiere A und B und schreibe Ergebnis nach E
sub	Subtrahiere B von A und schreibe Ergebnis nach E
less	Ermittle «A»<«B» und schreibe Ergebnis (wahr oder falsch) nach E
lesseq	Ermittle «A»≤«B» und schreibe Ergebnis (wahr oder falsch) nach E
iftruegoto M	Setze die Verarbeitung bei dem mit M markierten Befehl fort, sofern in E der Wert wahr steht, anderenfalls gehe zum nächsten Befehl
goto M	Setze die Verarbeitung bei der mit M markierten Anweisung fort
get	Eingabe eines Datums in das Register A
print	Drucke den Inhalt des Registers A
print "..."	Drucke den Text zwischen den Anführungszeichen
stop	Ende der Berechnung.

Vom Programm zum Computer
maschinennahe Programmierung

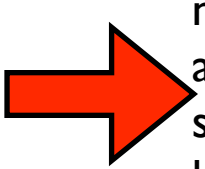


read	Speicherzelle, deren Adresse in SAR angegeben ist, nach SPR
write	Inhalt von SPR in Speicherzelle, deren Adresse in SAR angegeben ist
SPR→A	Inhalt des SPR in das Register A des Rechenwerks
A→SPR	Inhalt von Register A des Rechenwerks nach SPR
B→A	Inhalt von Register B des Rechenwerks in Register A
E→A	Inhalt von Register E des Rechenwerks in Register A
n→A	ganzzahlige Konstante n in Register A
add	Addiere A und B und schreibe Ergebnis nach E
sub	Subtrahiere B von A und schreibe Ergebnis nach E
less	Ermittle «A»<«B» und schreibe Ergebnis (wahr oder falsch) nach E
lesseq	Ermittle «A»≤«B» und schreibe Ergebnis (wahr oder falsch) nach E
iftruegoto M	Setze die Verarbeitung bei dem mit M markierten Befehl fort, sofern in E der Wert wahr steht, anderenfalls gehe zum nächsten Befehl
goto M	Setze die Verarbeitung bei der mit M markierten Anweisung fort
get	Eingabe eines Datums in das Register A
print	Drucke den Inhalt des Registers A
print "..."	Drucke den Text zwischen den Anführungszeichen
stop	Ende der Berechnung.

Vom Programm zum Computer
maschinennahe Programmierung

read	Speicherzelle, deren Adresse in SAR angegeben ist, nach SPR
write	Inhalt von SPR in Speicherzelle, deren Adresse in SAR angegeben ist
SPR→A	Inhalt des SPR in das Register A des Rechenwerks
A→SPR	Inhalt von Register A des Rechenwerks nach SPR
B→A	Inhalt von Register B des Rechenwerks in Register A
E→A	Inhalt von Register E des Rechenwerks in Register A
n→A	ganzzahlige Konstante n in Register A
add	Addiere A und B und schreibe Ergebnis nach E
sub	Subtrahiere B von A und schreibe Ergebnis nach E
less	Ermittle «A»<«B» und schreibe Ergebnis (wahr oder falsch) nach E
lesseq	Ermittle «A»≤«B» und schreibe Ergebnis (wahr oder falsch) nach E
iftruegoto M	Setze die Verarbeitung bei dem mit M markierten Befehl fort, sofern in E der Wert wahr steht, anderenfalls gehe zum nächsten Befehl
goto M	Setze die Verarbeitung bei der mit M markierten Anweisung fort
get	Eingabe eines Datums in das Register A
print	Drucke den Inhalt des Registers A
print "..."	Drucke den Text zwischen den Anführungszeichen
stop	Ende der Berechnung.

Vom Programm zum Computer
maschinennahe Programmierung

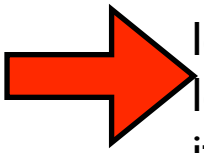


read	Speicherzelle, deren Adresse in SAR angegeben ist, nach SPR
write	Inhalt von SPR in Speicherzelle, deren Adresse in SAR angegeben ist
SPR→A	Inhalt des SPR in das Register A des Rechenwerks
A→SPR	Inhalt von Register A des Rechenwerks nach SPR
B→A	Inhalt von Register B des Rechenwerks in Register A
E→A	Inhalt von Register E des Rechenwerks in Register A
n→A	ganzzahlige Konstante n in Register A
add	Addiere A und B und schreibe Ergebnis nach E
sub	Subtrahiere B von A und schreibe Ergebnis nach E
less	Ermittle «A»<«B» und schreibe Ergebnis (wahr oder falsch) nach E
lesseq	Ermittle «A»≤«B» und schreibe Ergebnis (wahr oder falsch) nach E
iftruegoto M	Setze die Verarbeitung bei dem mit M markierten Befehl fort, sofern in E der Wert wahr steht, anderenfalls gehe zum nächsten Befehl
goto M	Setze die Verarbeitung bei der mit M markierten Anweisung fort
get	Eingabe eines Datums in das Register A
print	Drucke den Inhalt des Registers A
print "..."	Drucke den Text zwischen den Anführungszeichen
stop	Ende der Berechnung.

Vom Programm zum Computer
maschinennahe Programmierung

read	Speicherzelle, deren Adresse in SAR angegeben ist, nach SPR
write	Inhalt von SPR in Speicherzelle, deren Adresse in SAR angegeben ist
SPR→A	Inhalt des SPR in das Register A des Rechenwerks
A→SPR	Inhalt von Register A des Rechenwerks nach SPR
B→A	Inhalt von Register B des Rechenwerks in Register A
E→A	Inhalt von Register E des Rechenwerks in Register A
n→A	ganzzahlige Konstante n in Register A
add	Addiere A und B und schreibe Ergebnis nach E
sub	Subtrahiere B von A und schreibe Ergebnis nach E
less	Ermittle «A»<«B» und schreibe Ergebnis (wahr oder falsch) nach E
lesseq	Ermittle «A»≤«B» und schreibe Ergebnis (wahr oder falsch) nach E
iftruegoto M	Setze die Verarbeitung bei dem mit M markierten Befehl fort, sofern in E der Wert wahr steht, anderenfalls gehe zum nächsten Befehl
goto M	Setze die Verarbeitung bei der mit M markierten Anweisung fort
get	Eingabe eines Datums in das Register A
print	Drucke den Inhalt des Registers A
print "..."	Drucke den Text zwischen den Anführungszeichen
stop	Ende der Berechnung.

Vom Programm zum Computer
maschinennahe Programmierung



read	Speicherzelle, deren Adresse in SAR angegeben ist, nach SPR
write	Inhalt von SPR in Speicherzelle, deren Adresse in SAR angegeben ist
SPR→A	Inhalt des SPR in das Register A des Rechenwerks
A→SPR	Inhalt von Register A des Rechenwerks nach SPR
B→A	Inhalt von Register B des Rechenwerks in Register A
E→A	Inhalt von Register E des Rechenwerks in Register A
n→A	ganzzahlige Konstante n in Register A
add	Addiere A und B und schreibe Ergebnis nach E
sub	Subtrahiere B von A und schreibe Ergebnis nach E
less	Ermittle «A»<«B» und schreibe Ergebnis (wahr oder falsch) nach E
lesseq	Ermittle «A»≤«B» und schreibe Ergebnis (wahr oder falsch) nach E
iftruegoto M	Setze die Verarbeitung bei dem mit M markierten Befehl fort, sofern in E der Wert wahr steht, anderenfalls gehe zum nächsten Befehl
goto M	Setze die Verarbeitung bei der mit M markierten Anweisung fort
get	Eingabe eines Datums in das Register A
print	Drucke den Inhalt des Registers A
print "..."	Drucke den Text zwischen den Anführungszeichen
stop	Ende der Berechnung.

Vom Programm zum Computer
maschinennahe Programmierung

read	Speicherzelle, deren Adresse in SAR angegeben ist, nach SPR
write	Inhalt von SPR in Speicherzelle, deren Adresse in SAR angegeben ist
SPR→A	Inhalt des SPR in das Register A des Rechenwerks
A→SPR	Inhalt von Register A des Rechenwerks nach SPR
B→A	Inhalt von Register B des Rechenwerks in Register A
E→A	Inhalt von Register E des Rechenwerks in Register A
n→A	ganzzahlige Konstante n in Register A
add	Addiere A und B und schreibe Ergebnis nach E
sub	Subtrahiere B von A und schreibe Ergebnis nach E
less	Ermittle «A»<«B» und schreibe Ergebnis (wahr oder falsch) nach E
lesseq	Ermittle «A»≤«B» und schreibe Ergebnis (wahr oder falsch) nach E
iftruegoto M	Setze die Verarbeitung bei dem mit M markierten Befehl fort, sofern in E der Wert wahr steht, anderenfalls gehe zum nächsten Befehl
goto M	Setze die Verarbeitung bei der mit M markierten Anweisung fort
get	Eingabe eines Datums in das Register A
print	Drucke den Inhalt des Registers A
print "..."	Drucke den Text zwischen den Anführungszeichen
stop	Ende der Berechnung.

Vom Programm zum Computer
maschinennahe Programmierung

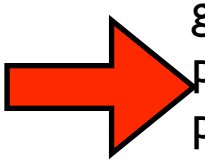
read	Speicherzelle, deren Adresse in SAR angegeben ist, nach SPR
write	Inhalt von SPR in Speicherzelle, deren Adresse in SAR angegeben ist
SPR→A	Inhalt des SPR in das Register A des Rechenwerks
A→SPR	Inhalt von Register A des Rechenwerks nach SPR
B→A	Inhalt von Register B des Rechenwerks in Register A
E→A	Inhalt von Register E des Rechenwerks in Register A
n→A	ganzzahlige Konstante n in Register A
add	Addiere A und B und schreibe Ergebnis nach E
sub	Subtrahiere B von A und schreibe Ergebnis nach E
less	Ermittle «A»<«B» und schreibe Ergebnis (wahr oder falsch) nach E
lesseq	Ermittle «A»≤«B» und schreibe Ergebnis (wahr oder falsch) nach E
iftruegoto M	Setze die Verarbeitung bei dem mit M markierten Befehl fort, sofern in E der Wert wahr steht, anderenfalls gehe zum nächsten Befehl
goto M	Setze die Verarbeitung bei der mit M markierten Anweisung fort
get	Eingabe eines Datums in das Register A
print	Drucke den Inhalt des Registers A
print "..."	Drucke den Text zwischen den Anführungszeichen
stop	Ende der Berechnung.

Vom Programm zum Computer
maschinennahe Programmierung

read	Speicherzelle, deren Adresse in SAR angegeben ist, nach SPR
write	Inhalt von SPR in Speicherzelle, deren Adresse in SAR angegeben ist
SPR→A	Inhalt des SPR in das Register A des Rechenwerks
A→SPR	Inhalt von Register A des Rechenwerks nach SPR
B→A	Inhalt von Register B des Rechenwerks in Register A
E→A	Inhalt von Register E des Rechenwerks in Register A
n→A	ganzzahlige Konstante n in Register A
add	Addiere A und B und schreibe Ergebnis nach E
sub	Subtrahiere B von A und schreibe Ergebnis nach E
less	Ermittle «A»<«B» und schreibe Ergebnis (wahr oder falsch) nach E
lesseq	Ermittle «A»≤«B» und schreibe Ergebnis (wahr oder falsch) nach E
iftruegoto M	Setze die Verarbeitung bei dem mit M markierten Befehl fort, sofern in E der Wert wahr steht, anderenfalls gehe zum nächsten Befehl
goto M	Setze die Verarbeitung bei der mit M markierten Anweisung fort
get	Eingabe eines Datums in das Register A
print	Drucke den Inhalt des Registers A
print "..."	Drucke den Text zwischen den Anführungszeichen
stop	Ende der Berechnung.

Vom Programm zum Computer
maschinennahe Programmierung

read	Speicherzelle, deren Adresse in SAR angegeben ist, nach SPR
write	Inhalt von SPR in Speicherzelle, deren Adresse in SAR angegeben ist
SPR→A	Inhalt des SPR in das Register A des Rechenwerks
A→SPR	Inhalt von Register A des Rechenwerks nach SPR
B→A	Inhalt von Register B des Rechenwerks in Register A
E→A	Inhalt von Register E des Rechenwerks in Register A
n→A	ganzzahlige Konstante n in Register A
add	Addiere A und B und schreibe Ergebnis nach E
sub	Subtrahiere B von A und schreibe Ergebnis nach E
less	Ermittle «A»<«B» und schreibe Ergebnis (wahr oder falsch) nach E
lesseq	Ermittle «A»≤«B» und schreibe Ergebnis (wahr oder falsch) nach E
iftruegoto M	Setze die Verarbeitung bei dem mit M markierten Befehl fort, sofern in E der Wert wahr steht, anderenfalls gehe zum nächsten Befehl
goto M	Setze die Verarbeitung bei der mit M markierten Anweisung fort
get	Eingabe eines Datums in das Register A
print	Drucke den Inhalt des Registers A
print "..."	Drucke den Text zwischen den Anführungszeichen
stop	Ende der Berechnung.

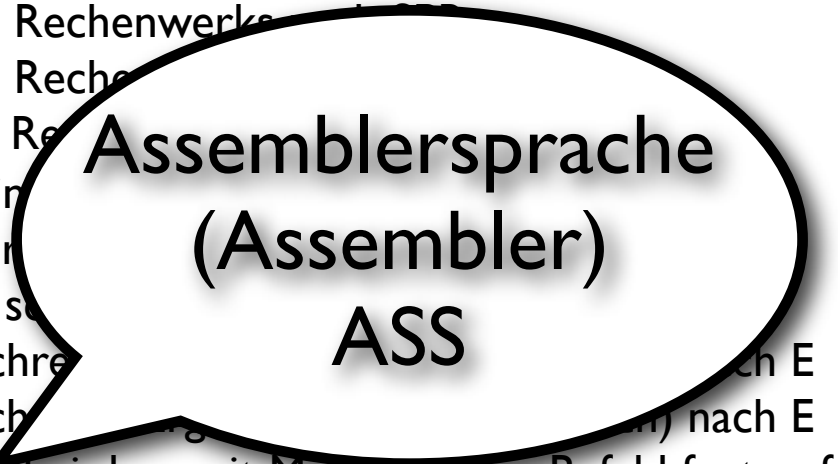


Vom Programm zum Computer
maschinennahe Programmierung

read	Speicherzelle, deren Adresse in SAR angegeben ist, nach SPR
write	Inhalt von SPR in Speicherzelle, deren Adresse in SAR angegeben ist
SPR→A	Inhalt des SPR in das Register A des Rechenwerks
A→SPR	Inhalt von Register A des Rechenwerks nach SPR
B→A	Inhalt von Register B des Rechenwerks in Register A
E→A	Inhalt von Register E des Rechenwerks in Register A
n→A	ganzzahlige Konstante n in Register A
add	Addiere A und B und schreibe Ergebnis nach E
sub	Subtrahiere B von A und schreibe Ergebnis nach E
less	Ermittle «A»<«B» und schreibe Ergebnis (wahr oder falsch) nach E
lesseq	Ermittle «A»≤«B» und schreibe Ergebnis (wahr oder falsch) nach E
iftruegoto M	Setze die Verarbeitung bei dem mit M markierten Befehl fort, sofern in E der Wert wahr steht, anderenfalls gehe zum nächsten Befehl
goto M	Setze die Verarbeitung bei der mit M markierten Anweisung fort
get	Eingabe eines Datums in das Register A
print	Drucke den Inhalt des Registers A
print "..."	Drucke den Text zwischen den Anführungszeichen
stop	Ende der Berechnung.

Vom Programm zum Computer
maschinennahe Programmierung

read	Speicherzelle, deren Adresse in SAR angegeben ist, nach SPR
write	Inhalt von SPR in Speicherzelle, deren Adresse in SAR angegeben ist
SPR→A	Inhalt des SPR in das Register A des Rechenwerks
A→SPR	Inhalt von Register A des Rechenwerks
B→A	Inhalt von Register B des Rechenwerks
E→A	Inhalt von Register E des Rechenwerks
n→A	ganzzahlige Konstante n in
add	Addiere A und B und schr
sub	Subtrahiere B von A und s
less	Ermittle «A»<«B» und schre
lesseq	Ermittle «A»≤«B» und sch
iftruegoto M	Setze die Verarbeitung bei dem mit M markierten Befehl fort, sofern in E der Wert wahr steht, anderenfalls gehe zum nächsten Befehl
goto M	Setze die Verarbeitung bei der mit M markierten Anweisung fort
get	Eingabe eines Datums in das Register A
print	Drucke den Inhalt des Registers A
print "..."	Drucke den Text zwischen den Anführungszeichen
stop	Ende der Berechnung.



Vom Programm zum Computer
Assembler und PRO

- Theoretische Informatik: Wenn der Speicher unbegrenzt wäre, dann gibt es zu jedem PRO-Programm ein semantisch äquivalentes ASS-Programm.

Vom Programm zum Computer
Beispiel

- Einlesen einer Zahl n und einer Zahlenfolge und Aufsummieren der Zahlen >n bzw. ≤n
- Speicherbelegungsplan:

Variable	Adresse der Speicherzelle
größer	10
kleiner	11
n	12
x	13

Vom Programm zum Computer
Assembler und PRO

```
10→A
A→SAR
0→A
A→SPR
write              größer←0
11→A
A→SAR
write              kleiner←0
12→A
A→SAR
get
A→SPR
write              get(n)
13→A
A→SAR
get
A→SPR
write              get(x)
```

```
schleife:0→B
         equal              x=0?
         iftruegoto ende
         12→B
         B→SAR
         read
         SPR→B
         greater            x>n?
         iftruegoto         sonst
         11→B
         B→SAR
         read
         SPR→B
         add
         E→SPR
         write         kleiner←kleiner+x
         goto next
```

Vom [illegible]puter

```
sonst:  10→B
        B→SAR
        read
        SPR→B
        add
        E→SPR
        write           größer←größer+x
next:   13→A
        A→SAR
        get
        A→SPR
        write           get(x)
        goto schleife
ende:   10→A
        A→SAR
        read
        SPR→A
        print           print(größer)
        11→A
        A→SAR
        read
        SPR→A
        print           print(kleiner)
        stop.
```

Vom Programm zum Computer
Ebenenmodell der Rechnerarchitektur

- Jedes PRO-Programm muß gemäß der skizzierten Vorschrift in ein semantisch äquivalentes Programm der Sprache ASS transformiert werden
- jeder Einzelschritt hat algorithmischen Charakter (nachprüfen!)
- Übersetzung leisten Übersetzerprogramme

Vom Programm zum Computer
Übersetzer

Definition:

Ein **Übersetzer** (oder **Compiler**) ist ein Programm, das Programme einer Programmiersprache in semantisch äquivalente Programme einer anderen Programmiersprache umwandelt.

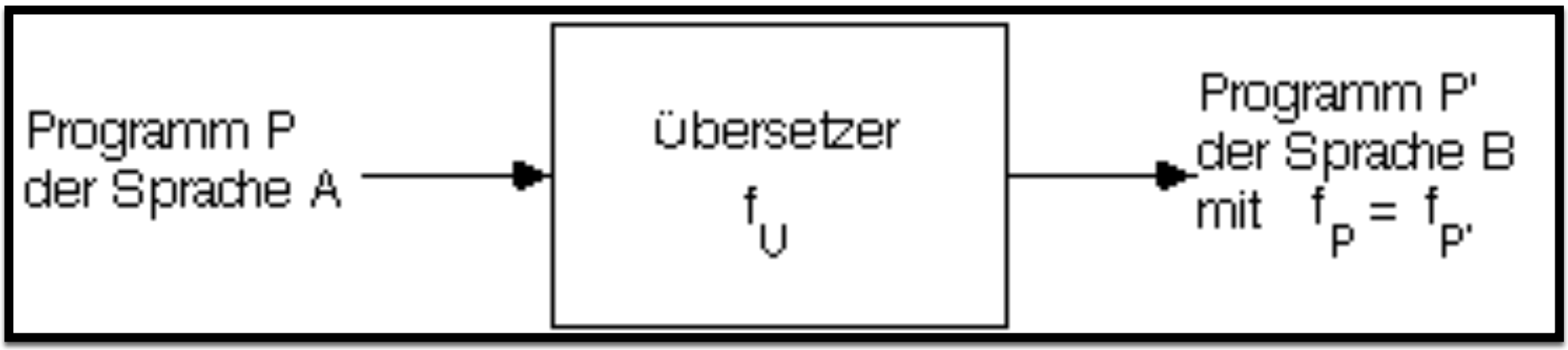
Vom Programm zum Computer
Übersetzer - berechnete Funktion

Übersetzer U realisieren Funktionen von der Sprache A in die Sprache B:

f_U: {P | P ist Programm der Sprache A}→

{P | P ist Programm der Sprache B} mit

$f_U(P)=P'$, so daß gilt $f_P=f_{P'}$.



Vom Programm zum Computer
Interpreter

Alternatives Vorgehen: Programm, das PRO-Programm P Anweisung für Anweisung sowie zugehörige Eingabedaten x einliest, *interpretiert* und sofort durch geeignete Befehlsfolge in ASS ausführt.

Die Ausgabe, die ASS-Programm liefert, entspricht der Ausgabe, die P mit Eingabe x liefern würde.

Definition:

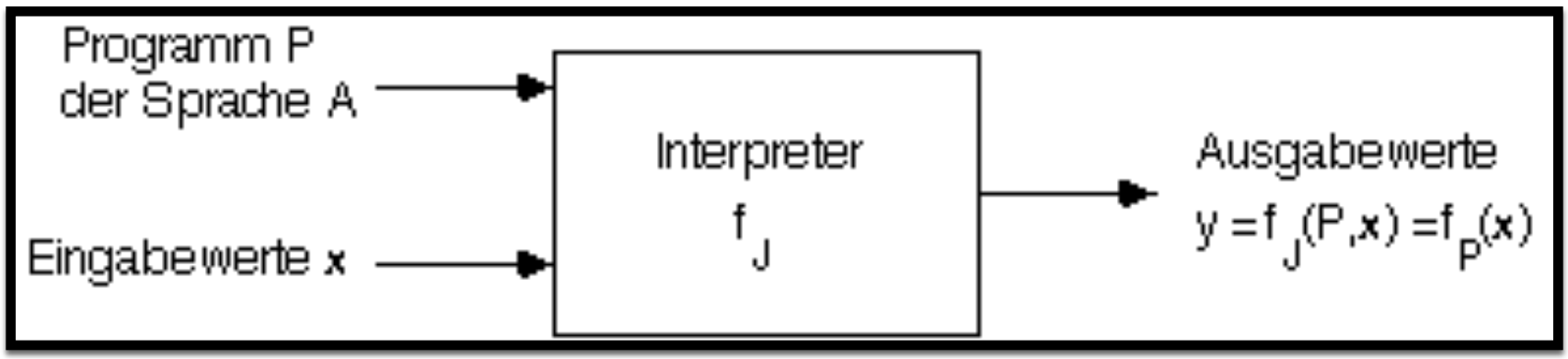
Ein Programm, das Programme einer anderen Programmiersprache einliest und sofort schrittweise ausführt, bezeichnet man als **Interpreter**.

Vom Programm zum Computer
Interpreter - berechnete Funktion

Interpreter J für die Sprache A berechnet eine Funktion:

f_J: {P | P ist Programm der Sprache A}
×{x | x ist mögliche Eingabe}→
{y | y ist mögliche Ausgabe} mit

$f_J(P,x)=f_P(x)=y$.



Vom Programm zum Computer

Beispiel - Interpreter

x←x+y ⇒

```
50→A
A→SAR
read
SPR→A
65→B
B→SAR
read
SPR→B
add
50→A
A→SAR
E→SPR
write.
```

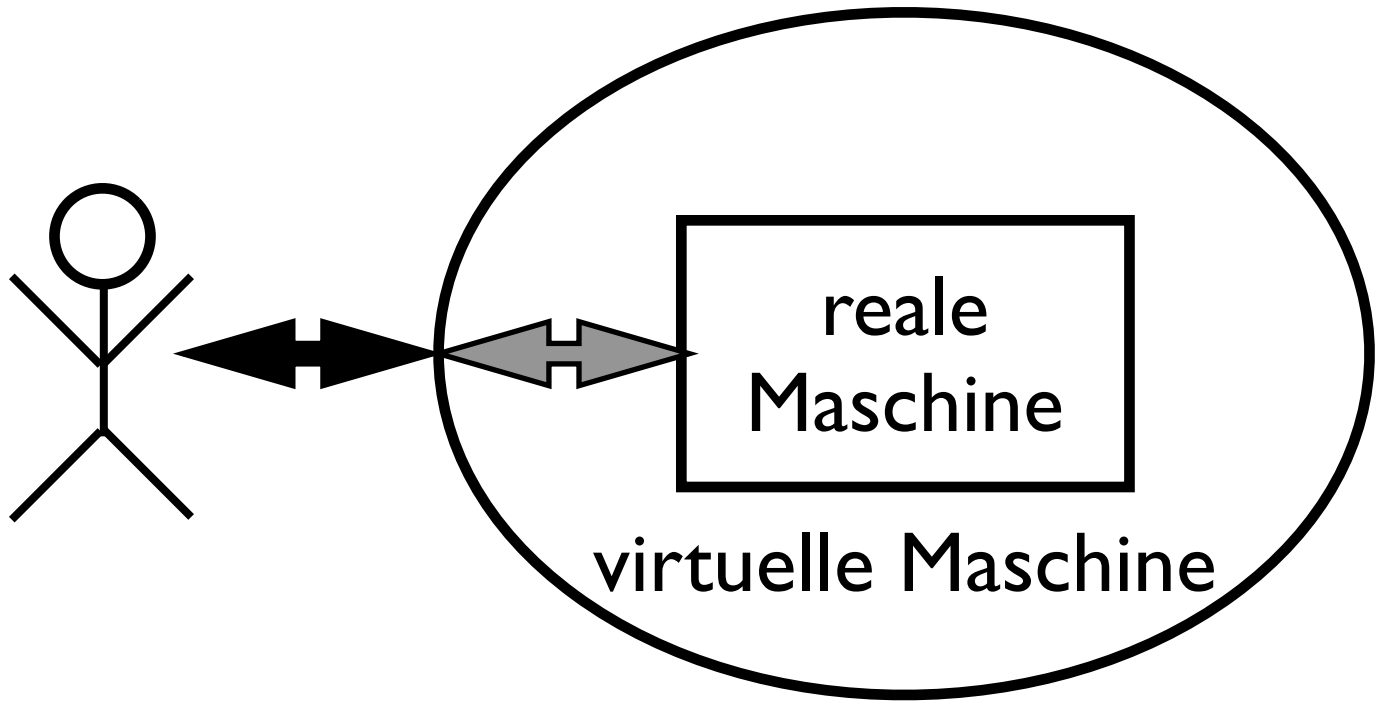
Adressen 50 für x und 65 für y
bereits reserviert bei Lesen
der PRO-Deklaration
def x,y: Zahl

Vom Programm zum Computer
virtuelle Maschine

- Eine gegebene Maschine C für ASS verhält sich vermöge eines geeigneten Übersetzers oder Interpreters wie eine Maschine für PRO.
- Übersetzer bzw. Interpreter machen aus der **realen** Maschine für die Sprache ASS eine **virtuelle** (d.h. gedachte) Maschine für PRO.

Vom Programm zum Computer

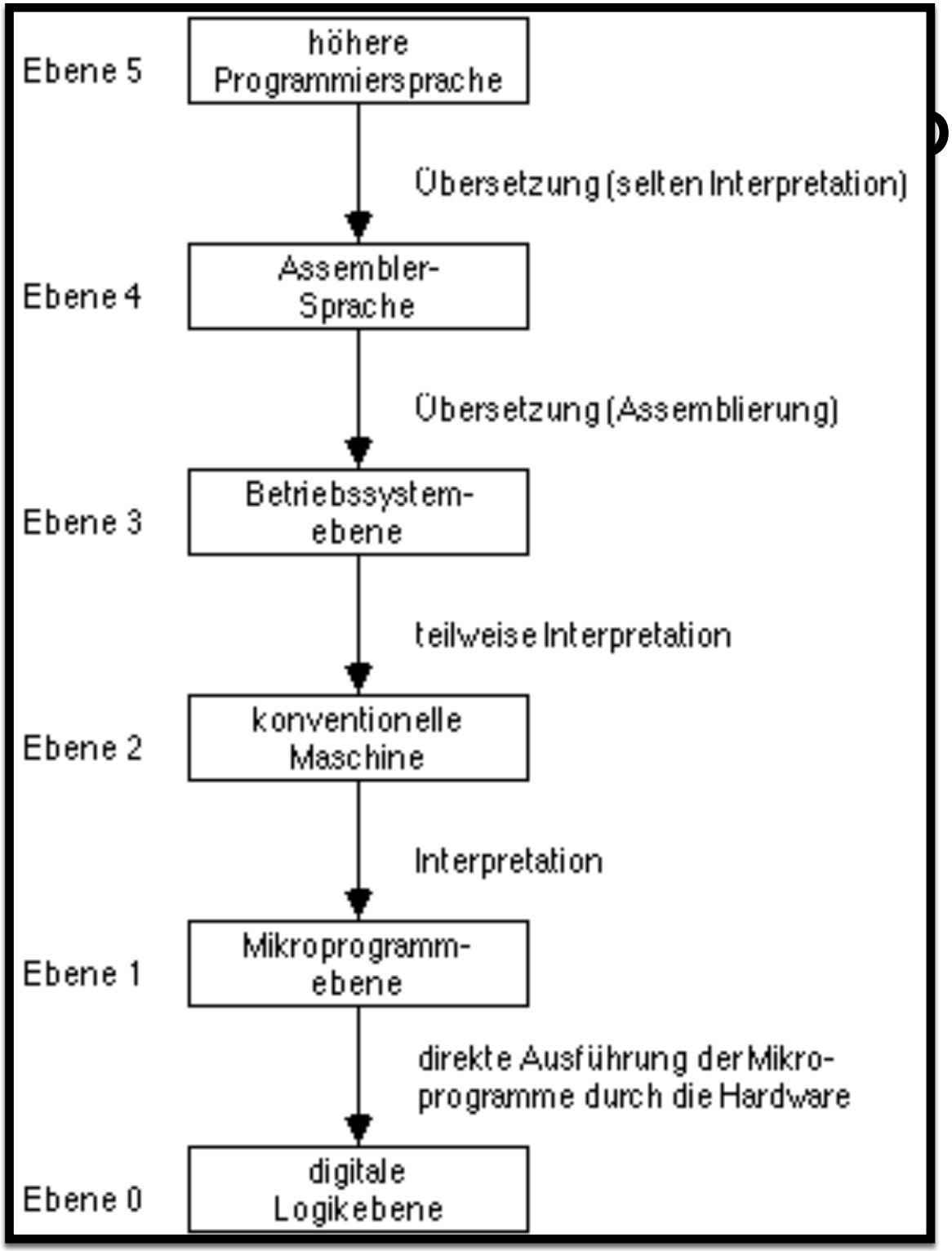
virtuelle Maschine



Vom Programm zum Computer
Ebenenmodell

- unser Vorgehen: PRO → Sprache ohne Konstruktoren → Sprache mit vereinfachten arithm. Ausdrücken → ASS
- in der Praxis: sechs Abstraktionsebenen (Ebenenmodell der Rechnerarchitektur)
- Je Ebene:
  - spezielle Form von *Daten*
  - spezielle Form von *Operationen*, die nach oben hin immer mächtiger werden.
  - *Übersetzer* oder *Interpreter*, der Programme einer Ebene auf semantisch äquivalente Programme der unmittelbar darunterliegenden Ebene abbildet.

Vom [illegible]buter


Ebene 5
höhere Programmiersprache
Übersetzung (selten Interpretation)
Ebene 4
Assembler-Sprache
Übersetzung (Assemblierung)
Ebene 3
Betriebssystem-ebene
teilweise Interpretation
Ebene 2
konventionelle Maschine
Interpretation
Ebene 1
Mikroprogramm-ebene
direkte Ausführung der Mikro-programme durch die Hardware
Ebene 0
digitale Logikebene

Vom Programm zum Computer
Ebene 0: digitale Logikebene

- Grundbaustein: **Gatter**, n Eingänge, m Ausgänge
- Daten: zwei unterschiedliche Spannungswerte 0 und 1 (**digital**)
- Gatter berechnen Funktionen

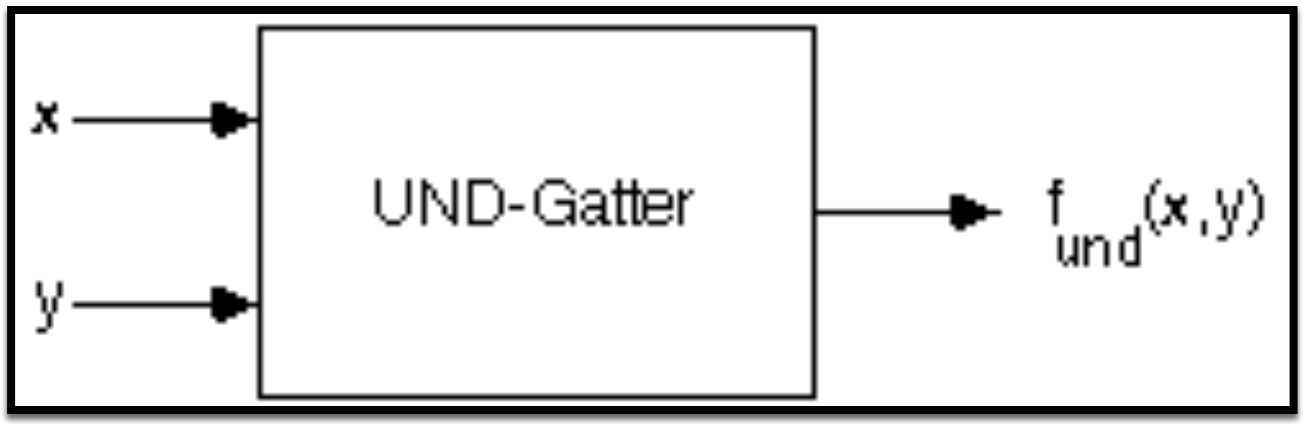
$$f: \{0,1\}^n \rightarrow \{0,1\}^m$$

Vom Programm zum Computer
UND-Gatter

- **UND-Gatter** realisiert UND-Funktion

$f_{und}: \{0,1\}^2 \rightarrow \{0,1\}$ mit

$$f_{und}(x,y) = \begin{cases} 0, \text{ falls } x=0 \text{ oder } y=0, \\ 1, \text{ falls } x=1 \textbf{ und } y=1. \end{cases}$$



Vom Programm zum Computer
Ebene 0: digitale Logikebene

- Zusammenschalten von Gattern ⇒

  komplizierte Funktionen
- unterhalb Ebene 0 weitere Ebenen → Elektrotechnik/Physik
- Ebene 0 hat kaum informatikbezogene Konzepte

Vom Programm zum Computer
Ebene 1: Mikroprogrammebene

- informatikbezogenes Konzept: Mikroprogramm
- Programm besteht aus Mikrobefehlen:
  - Transportiere 0-1-Information von einer Stelle des Computers zu einer anderen
  - Prüfe, ob eine Gatterleitung den Wert 1 besitzt, und führe ggf. eine Operation aus
  - Lege auf bestimmte Gatterleitung den Wert 0
- Befehlsvorrat: 20 oder viel mehr Mikrobefehle
- direkte Ausführung der Mikrobefehle durch Logikebene

Vom Programm zum Computer
Ebene 2: konventionelle Maschine

- Neuerung: Datenstrukturierung
  - Gruppierung von Gatterleitungen zu Registern/ Speicherzellen
  - Zusammenfassung von Mikrobefehlen zu neuen mächtigeren Befehlen
- ⇒ **Maschinensprache** (immer noch 0-1-Folgen)

Interpreter

- Maschinenprogramm → Mikroprogramm
- unterste Ebene, die als "gewöhnlicher" (System-) Programmierer zugänglich ist

Vom Programm zum Computer
Ebene 3: Betriebssystemebene

- wie Ebene 2 + weitere komfortable Sprachelemente
  - zur Speicherverwaltung
  - zur Verarbeitung mehrerer Programme gleichzeitig
  - zur Bearbeitung von Unterbrechungen
  - zur Kommunikation mit anderen Rechnern
  - zur Unterstützung der Ein- und Ausgabe
- immer noch 0-1-Folgen
- Interpretation

Vom Programm zum Computer
Ebene 4: Assemblerebene

- symbolische Bezeichnungen für Daten und Befehle
- 01100111 ⇒ MOVE R1,R3 oder R3←R1
- Assemblierer *übersetzt* Assembler-Programme in semantisch äquivalente Programme der Sprache von Ebene 3
- Assemblierer geschrieben in Sprache der Ebene 3
- ASS gehört zu Ebene 4
- vor 1960 hörte an dieser Stelle das Ebenenmodell auf

Vom Programm zum Computer
Ebene 5: höhere Programmiersprache

- **höhere** oder **problemorientierte** Programmiersprachen ↔ **niedere** oder

  maschinenorientierte Programmiersprachen (Sprachen der Ebenen 0 bis 4)

- *Übersetzer* oder *Interpreter* oder Mischung aus beidem in Programme der Ebene 3 oder 4

- zur Zeit ≥1000 höhere Programmiersprachen (Ada, ALGOL, BASIC, C, COBOL, Eiffel, Fortran, Java, Lisp, ML, Modula-2, Oberon, Pascal, Prolog)

Vom Programm zum Computer
Ebenenmodell

- weitere Ebenen: z.B. Ebene 6=Anwendungsprogramm
- Ebene 0: Hardware
- Ebene 2-5: Software
- Ebene 1: **Firmware**
  - bei Fertigung vom Hersteller in die elektronischen Bausteine eingebettet
  - prinzipiell Software, jedoch über einen längeren Zeitraum (oder immer) fest
  - nur mit Hilfsmitteln veränderbar (also quasi "hard").

Vom Programm zum Computer
Ebenenmodell

- Beachte: Einteilung in Hard-, Firm- und Softwareebenen ist *Modell*
  - Jede Anweisung, die durch Software ausgeführt wird, kann man auch in Hardware verlagern
  - Umgekehrt kann jede Operation, die durch Hardware realisiert ist, auch durch geeignete Software simuliert werden
- Designentscheidungen: Verarbeitungsgeschwindigkeit, Kosten, Erstellungszeit, Grad der Parallelisierung, Qualität der Bauelemente usw.

Vom Programm zum Computer
Ebenenmodell

- Beachte: Einteilung in Hard-, Firm- und Softwareebenen ist *Modell*
  - Jede Anweisung, die durch Software ausgeführt wird, kann man auch in Hardware verlagern
  - Umgekehrt kann jede Operation, die durch Hardware realisiert ist, auch durch geeignete Software simuliert werden
- Designentscheidungen: Verarbeitungsgeschwindigkeit, Kosten, Erstellungszeit, Grad der Parallelisierung, Qualität der Bauelemente usw.

Hardware und Software sind logisch äquivalent

Vom Programm zum Computer
Ebenenmodell - Schlußbemerkungen

- Computersysteme werden von verschiedenen Ebenen aus betrachtet
- Jede Ebene beschreibt das System auf einem anderen Abstraktionsniveau
- Unterscheidungsmerkmale: vorhandene Grundoperationen und -objekte
- Grundelemente jeder Ebene werden durch Kombination der Grundelemente der darunterliegenden Ebene gebildet