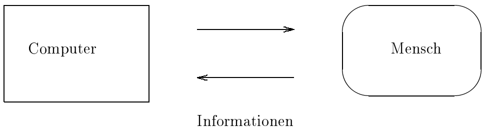
Kapitel 1

Rechnerarchitektur

Vom (mechanischen) Spezialgerät zum Universalrechner



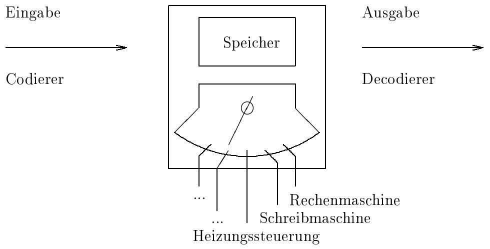
Im Gegensatz

- zur mechanischen Kurbel-Rechenmaschine,
- zur Schreibmaschine,
- zum Heizungsthermostat,
- zur Programmautomatik einer Waschmaschine

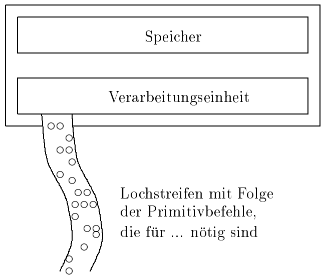
ist ein *Computer* ein universell einsetzbares Gerät zur Verarbeitung von Daten (Rechnen, Datenverarbeitung (DVA), Bereitstellung von Informationen). Diese Universalität wird dadurch erreicht, daß

- die Informationen einheitlich als binäre Daten (0/1-Informationen, Spannung da/nicht da) codiert werden (Ein-/Ausgabe),
- nur kurzzeitig verfügbare Daten in ihrer binären Codierung zwischengespeichert werden (Speicher)

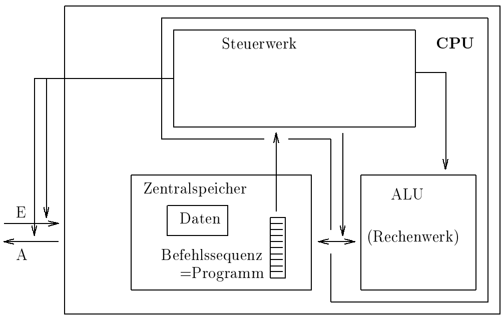
und eine *universelle* Binärdatenverarbeitungseinheit durch *Umschaltung* für fast alle denk- und lösbaren Informationsverarbeitungsaufgaben eingesetzt werden kann. Natürlich wird in der folgenden Abbildung keine volle Universalität er-



reicht; lediglich eine durch Schalterstellung definierbare *1 aus N* Spezialauswahl ist realisiert. Sieht man jedoch eine große Anzahl von die Daten nur gering ändernden *Spezialmaschinen* , sogenannte *Primitivbefehle* oder *Maschinenbefehle* vor, und baut aus *sequentieller* Aneinanderreihung solcher Befehle den gewünschten Effekt nach und nach auf, so ist die erstrebte Universalität erreicht. Erkennt



man nun schließlich, daß die benötigten *Sequenzen von Primitivbefehlen* auch nur eine spezielle Art von Information sind, die deshalb — wie die Ein- und Ausgabedaten — ebenfalls codiert im Speicher abgelegt werden können, so haben wir den prinzipiellen Aufbau eines Computers entwickelt:



Um beliebige Programme in den *Computer* hineinzubekommen und dann ablaufen lassen zu können, ist ein Verwaltungsprogramm, das *Betriebssystem* nötig, das den Computer erst benutzbar macht. Dies ist entweder in nichtflüchtigem Speicher (ROM = *read only memory*, Inhalt bleibt auch in ausgeschaltetem Zustand erhalten) fest eingebaut oder wird von einer Festplatte, einem Magnetband, ... mit Hilfe des im ROM fest eingebauten *Urladers* oder *Monitorprogramms* geladen.

Aufgabe 1.1 *Diskutieren Sie die Vor- und Nachteile eines Betriebssystems im ROM und eines von Platte geladenen Betriebssystems.*

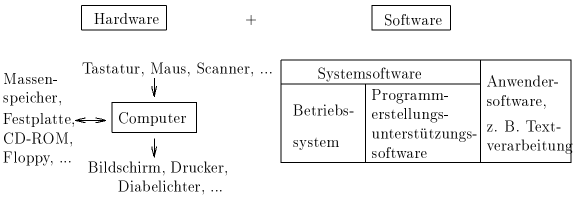


Abbildung 1.1: Computersystem

Tabelle 1.1: *Zeittafel zur Entwicklung der Computertechnik*

1833	Charles Babbage (1792-1871), Professor an der Universität Cambridge (Großbritannien), entwirft und baut einen programmgesteuerten mechanischen Rechenautomaten, die *Analytical engine*. Sie enthält ein 4-Spezies-Rechenwerk, 1000 Zahlenspeicher, Lochkartensteuerung und Ergebnisdruckwerk (nicht vollendet).
1941	Vorführung des ersten arbeitsfähigen programmgesteuerten Rechenautomaten ZUSE Z3 in Relaistechnik durch Konrad Zuse (geb. 1910)
1944	Inbetriebnahme des programmgesteuerten elektromechanischen Rechenautomaten Mark 1 von Howard H. Aiken (1900-1973).
1944/46	Formulierung der Prinzipien des *von-Neumann-Computers* (John von Neumann (1903-1975). Realisiert erstmals mit der EDVAC (1952/53)
1946	Inbetriebnahme des ENIAC, des ersten Computers mit Elektronenröhren durch John P. Eckert (geb. 1919) und John W. Mauchley (1907-1980). Beginn der Epoche der elektronischen Computer.
1951	Beginn der Serienproduktion elektronischer Computer mit der Anlage UNIVAC I der Firma Remington Rand. Beginn der 2. Computergeneration.
1955	Erster Computer mit Transistoren: TRADIC (Bell. Labor.)
um 1965	Computer-Familie IBM/360. Beginn der 3. Computergeneration. Der Begriff *Rechnerarchitektur* wird erstmals verwendet.
um 1965	Minirechner PDP-8 (Digital Equipment Corp.). Kleinere Rechner entstehen neben den Mainframes.
1969	Pilotprojekt Weitverkehrsrechnernetz ARPANET (USA) in Betrieb genommen.
1971	4-Bit-Mikroprozessoren i4004 der Firma INTEL Corp. Beginn der Mikroprozessorära. Rascher Übergang zu 8-Bit-Mikroprozessoren.
1975-1980	Personalcomputer auf Mikroprozessorbasis und die Software dafür werden zu Massenartikeln. Beginn der 4. Computergeneration.
um 1978	Erste Installationen lokaler Rechnernetze.
1978-1980	16-Bit-Mikroprozessoren kommen auf den Markt.
1979	Standardvorschlag der ISO »*Reference model of open system interconnections*« für Rechnernetze.
1981-1985	Personalcomputer mit 16-Bit-Mikroprozessoren werden marktbestimmend; insbesondere PC XT und AT von IBM und Kompatible dazu sowie das Betriebssystem MS-DOS der Firma Microsoft.
ab 1988	Personalcomputer mit 32-Bit-Mikroprozessoren kommen auf den Markt und lösen im Verlauf einiger Jahre die 16-Bit-Systeme ab.
ab 1996	Workstations mit 64-Bit-Mikroprozessoren

Quelle: D.Werner (Hrsg.): Taschenbuch der Informatik; Fachbuchverlag Leipzig, 1995

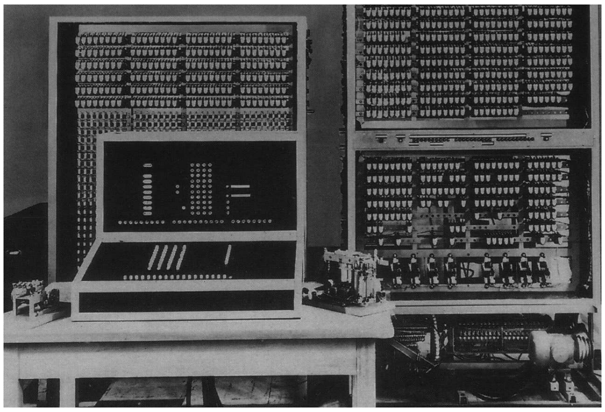
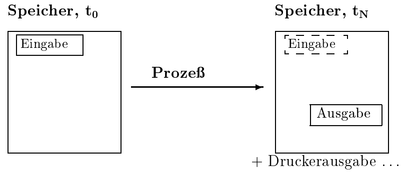
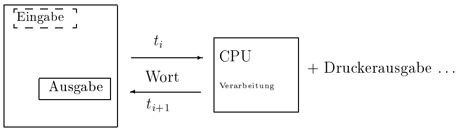
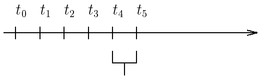


Abbildung 1.2: Der erste arbeitsfähige programmgesteuerte Rechenautomat, der ZUSE Z3 von Konrad Zuse



Die gewünschte Abbildung, die eine *Eingabe* in eine *Ausgabe* konvertiert, wird in Einzelschritten, Zeiteinheit für Zeiteinheit errechnet:





Machinenzykluszahl heute:
typisch

$$\frac{1}{100\,MHz} = \frac{1}{100 \cdot 10^6\,s^{-1}} = 10 \cdot 10^{-9}\,s = 10\,ns$$

1.1 Der von-Neumann-Rechner

Blockschaltbild eines Computers

Beim prinzipiellen Computeraufbau nach Abbildung 1.3 stellt sich die Frage, wann welche Systemeinheit welche Bus-Leitungen benutzen darf. Dies wird

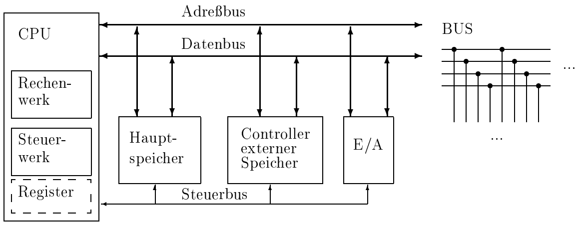


Abbildung 1.3: Universalrechner im Aufbau

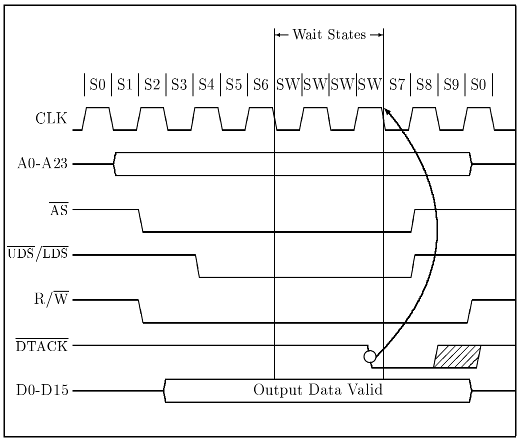


Abbildung 1.4: „circuit switched“ Datentransfer

durch den Controller (CPU oder DMA-Chip) mittels der *Handshake-Steuerbus-Leitungen* gemäß Abbildung 1.4 geregelt.

Aufgabe 1.2 *Diskutiere:* write *Daten von Datenbus D0-D15 in die Speicherzelle mit der auf A0-A23 anliegenden Adresse.*

Bemerkungen:

- Signalnamen:

CLK	=	clock
A0-A23	=	address bus
$\overline{AS}$	=	address strobe
$R/\overline{W}$	=	read/write
$\overline{UDS}, \overline{LDS}$	=	upper/lower data strobe
D0-D15	=	data bus
$\overline{DTACK}$	=	data transfer acknowledge

- strobe = Gültigkeitssignal für z.B. die Daten auf dem Adreßbus

Problem:

- *circuit switched*: Der Bus blockiert, bis die anstehende Operation ausgeführt ist: *wait states*

Alternative:

- *packet switched*: Jeder Datentransfer wird in kleine Pakete zerlegt. Diese werden nacheinander nichtblockierend auf den Bus geschickt. Dadurch kann eine weitere CPU (ein weiterer Thread) schon wieder Daten transferieren, obwohl die erste CPU ihren Transfer noch nicht abgeschlossen hat.

Im Inneren der CPU

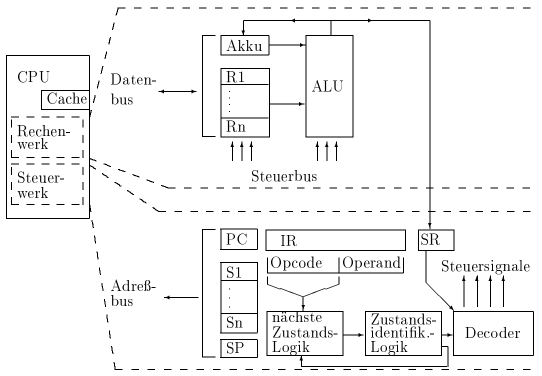


Abbildung 1.5: CPU

Programmausführung:

```
PC := StartingAddress;
RunBit := 1;
while RunBit = 1 do begin
  IR := memory(PC);   PC := PC + Befehlslänge(IR);
                                 häufig=1, fest
  InstrType := DetermineInstrType(IR) ;
  DataAddress := LocateData(IR,InstrType,PC);
  if DataAddress <> nil then
    data := memory(DataAddress);
  else
    data := non_valid;
  Execute(IR,InstrType,data,memory,registers,PC,RunBit);
end;
```

Ein von-Neumann-Rechner ist durch folgende Merkmale gekennzeichnet (zu 6.-8. vgl. auch Kapitel 1.2):

Prinzipien:

1. **Computerbestandteile:** CPU (= Rechenwerk und Steuerwerk), Speicher, Ein- und Ausgabewerk.
2. **Universalität:** Spezialisierung nur durch von außen eingegebenes Programm.
3. **Programme als Daten:** Eingabedaten, Zwischen- und Endergebnisse sowie Programme werden im selben Speicher abgelegt.
4. **Linearer Speicher:** Der Speicher ist in gleichgroße Zellen, die fortlaufend numeriert sind, eingeteilt.
5. **Sequentielle Ausführung:** Aufeinanderfolgende Befehle eines Programms werden in aufeinanderfolgenden Speicherzellen abgelegt. Beim Programmablauf wird im allgemeinen der Befehlszähler fortlaufend inkrementiert.
6. **Sprünge:** Sprungbefehle ermöglichen ein Durchbrechen der linearen Ausführungsreihenfolge.
7. **Zusätzliche Primitiva:** Datentransferbefehle, arithmetische Operationen, logische und Vergleichsoperationen, Graphik-/Multimediabefehle; indizierte, relative, ... Adressierung.
8. **Binäre Codierung:** Daten (Befehle, Adressen, Operanden) werden binär codiert. Geeignete Schaltwerke (Decodierer) sorgen für die richtige Entschlüsselung.

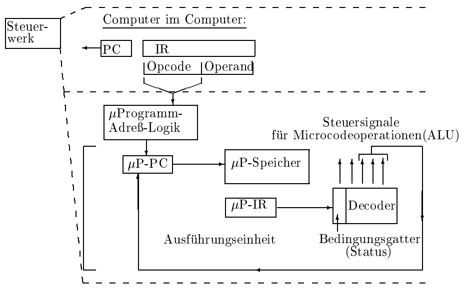


Abbildung 1.6: CISC = Microprogrammierte CPU

Das Konzept des *Zusammenbaus* komplexer Aufgaben aus einer Sequenz primitiver Aktionen war so erfolgreich, daß in Zeiten, als die Datentransferzeit zwischen Speicher und CPU im Vergleich zur Ausführungszeit eines Primitivbefehls sehr groß war (vor $\sim$ 1970), das Rechenwerk wiederum als (Mikro-)Computer aufgebaut wurde: Die Software reduziert den Schaltungsaufwand (=Anzahl der Gatter, Anzahl der Leitungen, ...) und „interpretiert" den Maschinencode der CPU (Computer im Computer). Heute wird für leistungsfähige PC's und Workstations das Rechenwerk wieder direkt in Hardware aufgebaut:

RISC (= reduced instruction set computer)-Designprinzip :

- Analysiere Anwendungen, um die Schlüsseloperationen zu finden.
- Reduziere unter allen Umständen die „Datenpfadzykluszeit" (Register $\rightarrow$ ALU $\rightarrow$ Register) für diese Operationen, also: *keine* Microcode-Interpretation (Maschinencode=Microcode).
- Jedes „neue" Feature ist nur zuzulassen, wenn es häufig benutzt wird und die Mehrzahl der existierenden anderen Feature nicht verlangsamt.

(Wiederhole diesen Design-Zyklus für die anderen CPU-Ressourcen: Cache, Speichermanagement, Gleitkomma-Koprozessoren, ...)