Grundlagen der

Elektronischen Datenverarbeitung

Verfasser Jupp Joachimski

Ein paar Worte vorweg:

Dieses Skriptum ist für all diejenigen gemacht, die ungern mit dem Auto fahren, ohne zu wissen, wie ein Motor funktioniert. Im Klartext: Sie müssen das nicht gelesen haben, um an dem Vortrag „Einführung in die Hard- und Software“ teilnehmen zu können. Sie können sogar von dem Vortrag profitieren, ohne dies hier gelesen zu haben. Nur ist es möglich, dass dann die eine oder andere Frage offen bleibt.

Das hier ist natürlich kein Einführungsbuch und kann weder ein solches ersetzen noch will es das. Durch diese Ausführungen sollen Sie die notwendigsten Informationen über die Ablaufzusammenhänge in der EDV erhalten. Ich kann aber hier – wie im Vortrag - ebenso wenig alle erheblichen Fragen darstellen wie ich die gesamte Menschheitsgeschichte auf 12 Seiten zusammenfassen könnte.

Was gehört eigentlich zur EDV?

Dumme Frage, werden Sie sagen, die Computer natürlich! Nicht nur, muss die Antwort lauten. Die Computer sind nur die eine Seite! Alle Arbeitsmittel der EDV zusammen nennt man

Betriebsmittel

und sie bestehen aus

Hardware und Software

Zentraleinheit Steuerwerk Rechenwerk Speicher	Peripherie Eingabewerk Ausgabewerk ggfs. zusätzliche Speichermedien wie z.B. Platte Floppy, Streamer	Systemsoftware (Betriebssystem)	Anwendersoftware z.B. Programmsprache In dieser geschrieben z.B.: Textprogramm Mit diesem gefertigt: Text

Das macht es verständlich, dass dieses Skriptum aus zwei Teilen besteht, von denen sich der erste mit der Hardware befasst, der zweite mit.. , na ja, sehen Sie selbst!

Sie sollten sich für dieses Skriptum etwas Zeit lassen und nicht die Vorstellung haben, Sie müssten im ersten Durchgang alles vollständig verstanden und behalten haben. Manche Probleme der EDV werden erst mit der Zeit verständlich – sie sickern.

Bei vielen Fragen werden Sie im Verlaufe der Zeit sehen, dass sie sich ganz von selbst lösen. Nichtsdestoweniger wage ich hier den Versuch, Ihnen die kostbare Zeit des Kurses für Ihre praktische Arbeit zu sparen und nehme es hin, wenn Sie nach dem ersten Durchlesen des Skriptums laut murren. Vielleicht unternehmen Sie nach dem Besuch des Kurses noch einmal einen Versuch, diese Seiten zu lesen. Sie werden merken, um wie vieles alles plötzlich leichter geht. Aber machen wir uns nichts vor: Geschenkt bekommen Sie das Verständnis für die EDV nicht. Sie müssen schon auch selbst etwas dazu tun.

A. Die Hardware

I. Rechen- und Steuerwerk

Was ist eigentlich ein Computer?

Ich bin mir sicher, Sie haben wenigstens einen schon in der Hand gehabt. Selbst diejenigen unter Ihnen, die sich bisher standhaft geweigert haben, mit Taschenrechner, Computer oder Smartphone umzugehen und daher 7 x 9 noch im Kopf ausrechnen können, werden kaum leugnen, irgendwann einmal einen Abakus - die handliche Rechenmaschine mit den 100 Kugeln - benutzt zu haben. Das ist die einfachste Form eines Computers - genauer gesagt einer digitale Rechenmaschine. Sie können mit dem Begriff "digital" nicht viel anfangen? Ich will gleich bei meinem Beispiel Abakus bleiben. Erinnern Sie sich noch an die Arbeitsweise dieses Geräts? Sie wollten damals 3 und 2 zusammenzählen und schoben dafür erst einmal drei Kugeln von links nach rechts, dann zwei weitere hinzu und zählten nun ab, wie viele insgesamt rechts waren.

Wir wollen diesen Vorgang einmal in seine Einzelschritte zerlegen:

- Eine Kugel nach rechts.
- Kontrolle: rechts ist eine Kugel, dann eine Leerstelle.
- Nächste Kugel nach rechts.
- Kontrolle: rechts sind zwei Kugeln, dann eine Leerstelle.
- Nächste Kugel nach rechts.
- Kontrolle: rechts sind drei Kugeln, dann eine Leerstelle. Drei Kugeln entsprechen dem ersten Wert.

.....

Sie können diesen Vorgang weiter bis zum Ende jeder Rechnung denken und werden eines feststellen: Immer lassen sich Ihre Rechenschritte so zerlegen, dass Sie diese auf die Frage reduzieren können, ob etwas ist (Kugel) oder nicht ist (Leerstelle). Das Ergebnis beziehen Sie aus der Häufigkeit der Einzelschritte. Wenn das nicht so wäre, gäbe es auch keinen Computer. Er benutzt natürlich keine Kugeln, sondern elektrische Ladung. Sie kann an einem bestimmten Punkt da sein (das ergibt den Wert 1) oder fehlen (das ergibt dann den Wert 0).

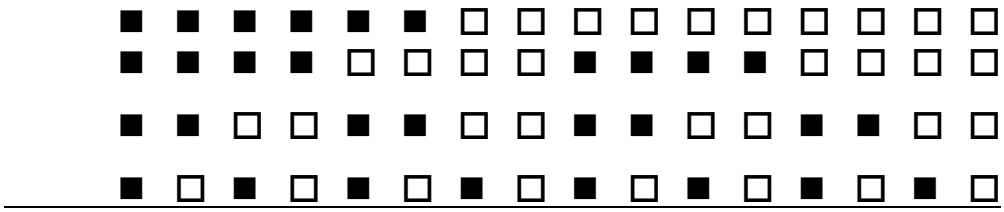
Diese Zerlegung komplexer Rechenvorgänge in einzelne Schritte nennt man Digitalisierung; erst sie ermöglicht einen Computer. So wie Sie rechnen und denken kann er nicht. Sie haben nämlich, vielleicht verbunden mit irgendwelchen Erinnerungen, Zahlenmuster im Kopf und rechnen mit ihnen. Wenn Sie 3 und 2 addieren wollen, erinnert sich Ihr Gehirn, dass das Ergebnis seit der 1.Volksschulklasse 5 war und es eigentlich keinen rechten Grund für eine andere Betrachtung gibt.

Technisch wird die digitale Datenbehandlung durch sog. „Flip-Flop"-Schaltungen realisiert. Ich könnte Ihnen natürlich jetzt den Schaltplan eines Flip-Flops zeigen und würde damit sicher die 10% der neugierigen Leser zufriedenstellen. Gleichzeitig würde ich aber die anderen 90% vergraulen - deswegen

verzichte ich darauf[1]. Es genügt hier zum Verständnis, wenn Sie wissen, dass es sich bei den heutigen Computern dabei um die Zusammenschaltung von zwei Transistoren handelt, die einen ganz besonderen Effekt zeitigt:

> Wird Spannung an die Schaltung gelegt, so ist an einem der beiden Ausgänge der Schaltung ebenfalls Spannung zu finden, am anderen jedoch nicht. Bei einem neuerlichen Spannungsstoß wechselt die Ausgangsspannung zum anderen Ausgang, der erste wird spannungslos. Wir haben damit so etwas wie ein "elektrisches Gedächtnis", das sich schon für eine einfache Anwendung nutzen lässt. Aber bitte beachten Sie auch: Das elektrische Gedächtnis ist nur solange in Funktion, wie die Stromversorgung läuft.

Nehmen wir einmal an, wir würden vier solcher Anordnungen gleichzeitig betreiben und Glühbirnen an jeweils einen Ausgang anschließen; den anderen einer jeden Schaltung legen wir still. Mit dieser Vorrichtung ließen sich folgende 16 Zustände zeigen, wobei hier "■" für eine dunkle, „□“ für eine helle Glühbirne steht:



Wert:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Sie könnten sich damit schon ein (kleines) Datenübermittlungssystem aufbauen: Stellen Sie sich vor, Sie würden jedem Wert einen Buchstaben zuordnen. Es würde zwar für ein Alphabet noch nicht ganz reichen, aber auf manche Buchstaben wie c, j, q, x und y sowie auf die Umlaute könnte man ja verzichten - zur Not erweitern wir einfach auf eine fünfte Reihe!. So könnten Sie über eine Lichttafel Nachrichten über einige Distanz übermitteln. Wenn Sie sich die Leuchtzustände aufschreiben, können Sie sogar diese Nachrichten ohne Decodierung weitergeben oder wiederholen.

Hier haben Sie gerade das Prinzip der Lochkartenrechnung erfunden. Wenn Sie sich statt der leuchtenden Glühbirnen die Löcher in einem Papierstreifen oder in einer Karte vorstellen und dann noch zwei Lochreihen zugeben, dann sind Sie beim Lochkartensystem. Es speichert die Information auf der Lochkarte in der einfachsten Weise nach dem Strickmuster „Loch - Nichtloch".

Zurück zur Arbeitsweise des Computers. Er baut sich aus der Zusammenfassung von Einzelinformationen (ja oder nein) ein Rechensystem auf. Intern rechnet allerdings der Computer also nicht mit dem uns geläufigen Dezimalsystem, das ja nur deswegen so verbreitet ist, weil die meisten Leute 10 Finger haben, sondern benutzt das sog. Binärsystem, welches mit zwei Ziffern auskommt. Der Computer kennt ja nur die beiden Informationszustände "ja" oder "nein". Ebenso arbeitet das Binärsystem, das nur die beiden Zahlen 1 und 2 kennt. Unsere gewohnten dezimalen Zahlen sehen etwas merkwürdig in diesem System aus:

[1] Wer die Erklärung nicht entbehren kann, möge sich bitte auf dem Dienstweg - dreifach handschriftlich im Original - an mich wenden.

00000000 = 0	00000100 = 4	01000000 = 64	11100000 = 224
00000001 = 1	00001000 = 8	10000000 = 128	11100001 = 225
00000010 = 2	00010000 = 16	10000001 = 129	11111110 = 254
00000011 = 3	00100000 = 32	11000000 = 192	11111111 = 255

Sie merken, zwei Ziffern reichen eigentlich zur Darstellung jeder Zahl! Da sich wiederum alle Rechenarten auf Subtraktion oder Addition zurückführen lassen und weil der elektrische Strom ein sehr flottes Wesen hat, kann der Computer mit dieser Methode gut und schnell rechnen. Wie?

Rechenoperationen

Wie rechnet nun der Computer mit den Zahlen? Binär natürlich! Wobei eines zu beachten ist: Jede Rechenoperation ist im Grunde genommen nur eine Vergleichsoperation. Machen wir ein Beispiel, indem wir 5 und 6 zusammenzählen:

5 bedeutet binär 0101
6 bedeutet binär 0110
Die Summe wäre 11.
11 bedeutet binär 1011

Betrachten Sie die letzten drei Ziffern des binären Wertes von 11! Fällt Ihnen etwas auf? Sie folgen im Vergleich zu den darüberstehenden Ziffern einer bestimmten Regel:

> Sind die Zahlen oben verschieden, so steht unten eine 1.
> Sind die Zahlen oben gleich, so steht unten eine 0.

Wenn wir einmal die Übertragskomplikation bei der ersten Ziffer außer Betracht lassen, können wir also die Addition auf einen Vergleich zwischen zwei Zahlen zurückführen. Der Übertrag (hier die erste Stelle) wird durch einen weiteren Vergleich bewirkt:

> Sind die Zahlen oben und unten gleich und haben sie den Wert 1, so wird die nächste Stelle links um 1 höher gesetzt.

So rechnet der Computer - und das ist gleichzeitig die Grundlage aller Arbeit. Jede andere Rechnung lässt sich wie jede andere Computerfunktion auf derartige Vergleichsoperationen zurückführen.

Halt, werden Sie rufen. Wie lässt sich dann Text wiedergeben? Nun, ich werde Ihnen zeigen, dass man sogar auf einer Rechenmaschine schreiben kann. Hier sehen Sie es:

```
----------oooooooooo
------oo--------oooooooo
-----o--o--------oooooooo
----o----o-------oooooooo
---o------o------oooooooo
--o--------o-----oooooooo
--o--------o-----oooooooo
--oooooo----------oooo
--o--------o-----oooooooo
--o--------o-----oooooooo
```

Zugegeben, schön sieht das **A** nicht aus, aber unser Abakus ist auch ein sehr grober Apparat. Übrigens, so wie hier erklärt, kommt ganz genau gesagt nur der Buchstabe auf dem Bildschirm oder im Drucker zustande: durch sinnvolle Anordnung von weißen Punkten (Kugel) oder schwarzen Punkten (Leerstelle). Die übrig bleibenden Kugeln rechts verschluckt der Rechner selbstverständlich. Auch stellt der Computer 25 Zeilen je 80 Zeichen dar. Worauf es aber dabei ankommt, ist, dass ein elektronischer Rechner Daten nur "digitalisiert" verarbeiten kann, d.h. er zerlegt jede Information in Einzelpunkte und speichert diese. Das Zerlegen nennt man auch "Digitalisieren".

Die „Von-Neumann-Prinzipien“

Über den Aufbau und die Funktionsweise der Computer wurden schon viele Betrachtungen angestellt. Immer noch gültig sind die von dem Informatiker von Neumann 1946 entwickelten Prinzipien:

1. *Jede Datenverarbeitungsanlage besteht aus 5 Funktionseinheiten:*
   Steuerwerk Rechenwerk Speicher (= Zentraleinheit)
   Eingabewerk Ausgabewerk (= Peripherie)
2. *Die Rechnerstruktur ist von der Problemstellung unabhängig.*
3. *Programme, Daten, Zwischen- und Endergebnisse werden im selben Speicher abgelegt.*
4. *Der Speicher ist in gleich große "Zellen" unterteilt (=Bytes), die fortlaufend nummeriert sind. Über diese Adressen kann der Zellinhalt abgerufen oder verändert werden.*
5. *Aufeinanderfolgende Befehle werden in aufeinanderfolgenden Speicherzellen abgelegt. Der nächste Befehl wird vom Steuerwerk durch Erhöhen der Befehlsadresse angesprochen.*
6. *Es gibt Sprungbefehle, die ein Abweichen von der gespeicherten Befehlsreihenfolge ermöglichen.*
7. *Der minimale Befehlsvorrat besteht aus arithmetischen oder logischen Befehlen, Transportkommandos und bedingten Sprüngen.*
8. *Daten werden binär codiert (0 oder 1)*

Der Mikroprozessor

Steuerwerk und Rechenwerk eines Computers sind üblicherweise in einer Baueinheit zusammengefasst: dem Mikroprozessor, auch CPU (**c**entral **p**rocessing **u**nit) genannt. Er stellt sozusagen eine Kombination von Herz und Kopf des Computers dar. War er in der "Steinzeit" der EDV noch ein manuell zusammengelötetes Werk von Röhren, Kondensatoren und Widerständen, so übernahmen nach und nach Transistoren die Aufgaben. Die Mikroprozessoren wurden immer kleiner und leistungsfähiger. Nur als Anhaltspunkt für deren Größe: Der Prozessor eines handelsüblichen Personalcomputers überdeckt lediglich eine Fläche von 2 x 6 cm bei etwa 0,5 cm Bauhöhe.

Ich will Ihnen einmal in einer Tabelle zeigen, wie der Weg vom "Röhrencomputer" zu den heutigen Superrechnern verlaufen ist:

Generation	Beginn	Transistorzahl[2]	technische Kennzeichen	Rechenzeit[3]
1	1944	--- -	Elektronenröhren	1s
2	1958	1	gelötete Halbleiterschaltkreise	0,1s
3	1965	50	teilintegrierte Schaltkreise	0,001s
4	1972	50000	hochintegrierte Schaltkreise	0,0001s
5	1980	>100000	höchstintegrierte Schaltkreise	< 0,000001s
6	1995	>10.000.000.000	„Mini“Chips	< 0,000000001s

Die Leistung eines Mikroprozessors wird in Flops (**fl**oating **o**perations **p**er **s**econd = Gleitkommaoperationen pro Sekunde) und in Ips (**i**nstructions **p**er **s**econd = Befehle pro Sekunde) gemessen. Es gibt dabei so etwas wie eine normierte Standardrechnung mit Gleitkommazahlen (=Zahlen ohne vorgegebene Nachkommastellenanzahl). Ein ganz erhebliches Kriterium der Prozessorleistung ist es aber auch, welche Speicherplatzgröße er ansprechen kann. Der Prozessor muss nämlich so ausgelegt sein,

---

[2] auf einem Rechenschaltkreis

[3] für eine Standardrechnung (Mittelwert aus 70% Addition und 30% Multiplikation). Wie der nicht ganz unbekannte Witz sagt: Hätten sich die Kraftfahrzeuge ebenso entwickelt, würde ein VW-Käfer heute 5 € kosten, 50.000 km/h schnell und 5 cm groß sein.

dass er von seiner mathematischen Leistung her den gesamten Speicherplatz abdecken kann. Um das richtig zu verstehen, müssen Sie aber noch einen anderen Begriff kennen:

Das Adressregister

Jede Eingabe in eine digitale Rechenmaschine führt zur Aufnahme der Information an einer bestimmten **Speicherstelle**. Der Rechner merkt sich dann, wo er die Information untergebracht hat und kann sie auf Bedarf abrufen. Im Kleinen kennen Sie das auch vom Taschenrechner: er kann eine Zahl oder ein Ergebnis in seinen Speicher übernehmen, bei Bedarf wieder hervorholen und damit weiterrechnen. Ein Computer hat nun nicht nur einen, sondern Zehntausende bis Milliarden solcher Speicherplätze.

Die Speicherpunkte werden dabei in den schon erwähnten miniaturisierten Halbleiterschaltungen angesiedelt. Zahlen und Buchstaben werden dort unter ihren binären Werten geführt. Bei jedem Tastendruck sendet die Tastatur des Computers an den Speicher eine bestimmte Zahlenfolge, die einer Ziffer oder einem Buchstaben zugeordnet ist. Kommt also z.B. der Impuls "65" herein, so sieht der Rechner in seinem Verzeichnis nach, findet dort unter "65" das "A" und bildet dieses am Bildschirm ab.

Um dem Computer diese Zahlen mundgerecht zu servieren, müssen sie erst in Binärwerte umgerechnet werden; dies besorgt unter anderem ein Schaltelement, welches das Herz des Computers bildet: der Mikroprozessor. Er steuert auch alle Rechenabläufe sowie die Ein- und Ausgabe der Daten in den oder aus dem Speicher.

Bei dem Weiterarbeiten mit dem Inhalt der Speicherplätze muss der Prozessor in jedem Einzelfall im sog. "Adressregister" nachfragen, wo die Information gespeichert ist. Handelt es sich um sehr komplexe Informationen, etwa um ein Maschinenspracheprogramm, so kommt der Benutzer nicht umhin, dem Rechner dabei zu helfen und von vorneherein die gewünschte Speicherstelle anzugeben. Da bei einem derartigen Vorgehen der Rechner nicht erst im Adressregister nachfragen muss, kann er auf diese Weise besonders schnell arbeiten. Das machen sich die sog. "Assemblersprachen" zunutze, deren Programme besonders schnell ablaufen, aber auch eine intime Kenntnis des Innenlebens der Computer beim Benutzer voraussetzen.

Die Bedeutung des Adressregisters wird an folgendem Vorgang verständlich:

Es erfolgt zunächst eine Eingabe (über das "Gedächtnis" des Computers oder über eine Eingabedatei):

> Speichere 1 an Speicherstelle 00000049
>
> Vermerke bei "Tastatur" : 1 = 49

Bei der Ausgabe geschieht folgendes:

> Die Tastatur meldet "1" an den Prozessor.
> Der Prozessor stellt fest: "1" ist an Speicherstelle 049
> Ausgegeben wird: 0011 0001.
> Der Bildschirmprozessor oder der Drucker wandeln um in "1"

Wie die Umwandlung geschieht, sehen Sie am besten am Beispiel der Zusammensetzung von Buchstaben in einem Abakus.

Leistungsfähigkeit:

Für die kleinste mögliche Information, die "0" oder "1" hat sich das englische Wort "**bit**" (**bi**nary **d**igit) eingebürgert. Ein normaler Computer ist nun aber in der Lage, auf dem Bildschirm 256 (=2^8) verschiedene Zeichen darzustellen. Im Binärcode benötigt man für die Definition dessen schon 8 Zahlen, die hintereinander geschrieben werden. Diese Informationsblöcke je 8 Zahlen nennt man **byte**. Unser Abakus kann etwa 12 byte darstellen, da er 100 bits fasst. Übrigens können Sie hier nicht einfach davon ausgehen, dass ein Kbyte 1000 byte sind: In der EDV wird mit Zweierpotenzen gerechnet. Demzufolge ist ein Kbyte gleich 1024 byte, ein Mbyte demnach 1024 Kbyte oder 1024 * 1024 = 1048576 byte usw.

Wir müssen bei Beurteilung der Leistungsfähigkeit eines Computers zwei Werte unterscheiden: Die **Geschwindigkeit** eines Rechners hängt zum Teil davon ab, wie viele Daten der Prozessor auf einmal handhaben kann. Waren die ersten Prozessoren schon froh, jeweils ein bit verkraften zu können, so ist die Verarbeitung von 32 bit zur gleichen Zeit jetzt schon Standard. Der Trend geht seit 1995 zu Prozessoren, die 64 bit oder 8 byte gleichzeitig (= parallel) verarbeiten können.

Die **Speichergröße** kennzeichnet nicht etwa die Leistungsfähigkeit des Prozessors, wenn sie auch mit dieser zu tun hat (er muss den Arbeitsspeicher ja bedienen können), sondern die tatsächliche Größe des Arbeitsspeichers (**RAM** = **R**andom **A**ccess **M**emory). Entscheidend ist im Verhältnis zwischen der Speichergröße und der Leistung des Prozessors immer der kleinere Wert: Kann der Prozessor nur ca. 640.000 Speicherstellen ansprechen, dann hilft es nichts, wenn tatsächlich ein größerer Speicher vorhanden ist.

Der oben genannte Speicher ist der sog. Arbeitsspeicher, der durch Vorgänge bei der Eingabe in seinem Inhalt verändert wird. Seinen schönen englischen Namen "**r**andom **a**ccess **m**emory" benutzt niemand; jeder hält sich an die Abkürzung "**RAM**". Daneben wird Ihnen noch gelegentlich das Kürzel "**ROM**" für "**r**ead **o**nly **m**emory" auffallen. Dies ist der unveränderbare Speicher eines Computers, dessen Inhalt vom Hersteller vorgegeben wird und dafür sorgt, dass sich nach dem Einschalten überhaupt etwas abspielt. Hier wird ein Teil des Betriebssystems eines Computers abgelegt. Nur über dieses kann der Benutzer in einen Dialog mit dem Rechner eintreten, ohne gezwungen zu sein, jede einzelne Anweisung in binärem Code einzugeben. Im ROM muss zumindest eine Anweisung für den Computer abgelegt sein, aus einem Speichermedium das Betriebssystem zu laden. Ein derartiges Minimalprogramm wird "**BIOS**" (**B**asic **I**nput and **O**utput **S**ystem) genannt.

B. Die Software

Ein Computer kann im Prinzip nur die Ja-Nein-Befehle (Ladung-Nichtladung) verstehen. Müsste man sich immer so mit ihm verständigen, so wäre er keine große Hilfe. Der Zugang zu seinen Vorteilen wäre einigen wenigen Programmierern vorbehalten und es wäre nicht recht klar, worin diese Vorteile überhaupt bestünden. Deswegen muss man dafür sorgen, dass man mit dem Computer in möglichst menschenverständlicher Sprache umgehen kann. Die Voraussetzungen dafür bringt er mit seiner Merkfähigkeit ja mit.

Jeder Zugang zum Computer oberhalb des Niveaus der Maschinensprache setzt demnach voraus, dass irgendwann einmal jemand dem Computer in Maschinensprache die für die Kommunikation notwendigen Schritte gezeigt hat. Deswegen ist die erste Ansprechebene diejenige der

I. Maschinensprache

Sicher ist die dem Rechner am meisten gemäße Sprache die allein aus Ja/Nein-Signalen bestehende. Da aber auch schon der Prozessor die Zahlen byteweise verstehen kann, benutzt man zur Eingabe der einfachsten und grundlegenden Informationen das sog. hexadezimale System (=16-Stellensystem), welches mit acht Ja-Nein-Kombinationen 256 (=28) Informationen mitteilen kann. Da man nun aber keine einzelnen Ziffern mit Werten über 10 kennt, muss man sich für 10 mit a, für 11 mit b usw. behelfen. So bedeuten in dem gebräuchlichen hexadezimalen System die Ziffern

01 = 1	0c = 12	11 = 17	50 = 80	a0 = 160	f0 = 240
02 = 2	0d = 13	1a = 26	60 = 96	b0 = 176	f1 = 245
09 = 9	0e = 14	20 = 32	70 = 112	c0 = 192	fa = 250
0a = 10	0f = 15	30 = 48	80 = 128	d0 = 208	fe = 255
0b = 11	10 = 16	40 = 64	90 = 144	e0 = 224	ff = 255

Die höchste zweistellige Zahl im Hexadezimalsystem ist ff (Dezimalwert 255). Mit 256 Werten kann man schon ein ganz komfortables Alphabet darstellen, das außer den Groß- und Kleinbuchstaben, den Satzzeichen und Zahlen auch noch nationale Sonderzeichen wie z.B. die deutschen (ä, ö, ü, ß) und Grafikzeichen umfasst. Direkte hexadezimale Eingaben in den Rechner macht nur der Programmierer bei der Eingabe der Basisinformationen, Sie brauchen sich damit nicht zu beschäftigen.

II. Die Programmiersprachen

stellen den Versuch dar, den Ausgangspunkt für lauffähige Programme weitgehend zu vereinheitlichen, d.h. einen Betrieb auf allen Computern ohne Rücksicht auf die Besonderheiten des jeweiligen Prozessors zu ermöglichen. Sie werden als Maschinenspracheprogramm von Diskette in das RAM eingeladen oder befinden sich bereits im ROM, wenn vom Zuschnitt des Computers her zu erwarten ist, dass er vorwiegend in einer Programmsprache betrieben wird. Gerade Heimcomputer melden sich meist beim Einschalten schon nicht mehr in ihrem Betriebssystem, sondern in der Programmsprache **BASIC** (**B**eginners **A**ll-purpose **S**ymbolic **I**nstruction **C**ode). Das ist übrigens nicht nur für solche Geräte typisch: Der superschnelle "Archimedes" z. B konnte sich BASIC als Betriebssystem leisten, weil er so schnell war, dass ihn eine Programmiersprache nicht wesentlich bremste.

Aus der Sicht des Informatikers sind die Programmsprachen nach der Art ihrer Kommunikation mit dem Rechner zu unterscheiden:

> In **Compilersprachen** geschriebene Programme müssen vor dem Start in ein Maschinenspracheprogramm übersetzt werden. Hierzu bedient man sich eines anderen maschinensprachlichen Programms, eben des Compilers. Das erfordert zwar Vorarbeit, macht aber die Programmausführung sehr schnell, da dann direkt der Prozessor angesprochen wird und der Rechner nicht gezwungen ist, Befehle während ihrer Ausführung in eine ihm gemäße Sprache zu übersetzen.
>
> Programmsprachen, die mit einem **Interpreter** (Übersetzer) arbeiten, erlauben einen direkten Programmstart nach Beendigung der Eingabe. Der hard- oder softwaremäßig im Rechner vorhandene Übersetzer wandelt während des Programmablaufs die dort genannten Befehle in Maschinensprache um. Das kann viel Zeit kosten und den Ablauf bremsen. Andererseits lassen sich Prozeduren (Programme), die in solchen Sprachen geschrieben werden, leichter testen.

Es sind auch Mischsysteme möglich, bei denen der Interpreter nur wenige Begriffe kennt und übersetzt; der Rest muss compiliert werden (z.B. bei der Programmsprache FORTH). Optimal sind allerdings sog. Programmentwicklungsumgebungen, die schon bei der Eingabe prüfen, ob Syntax und Logik des Programms stimmen (z.B. Quickbasic), ja eventuell sogar eine interpreterweise Ausführung des später zu kompilierenden Programms ermöglichen.

Eine weitere Unterscheidung wird danach getroffen, ob die Sprache besonders maschinennah oder besonders anwendernah konstruiert wurde. Maschinennahe Sprachen sind einfach strukturiert und beschränken sich auf sehr wenige Befehle. Andere Kommandos müssen aus diesen Grundbefehlen zusammengesetzt werden. Diese Sprachen ermöglichen aber sehr schnell ablaufende Programme. Demgegenüber sind anwendernahe Programme eher langsam, dafür aber recht komfortabel. Ein gutes Beispiel für eine anwendernahe Programmiersprache ist BASIC. Auch die sog. Abfragesprachen, die in Anwenderprogramme wie unser HIT integriert sind, können als besonders anwendernah gelten. Als Beispiel einer besonders maschinennahen Sprache sei "C" genannt.

Dies ergibt folgenden Überblick über die

Programmsprachengenerationen

1-3 **Prozedurale** (imperative) Sprachen, d.h.: Programmablauf wird durch die Reihenfolge der Anweisungen bestimmt.

1 reine Maschinensprache - binär codiert

2-5 symbolische (mnemonische) Sprachen

2 maschinennahe Sprache: Assembler

3-5 höhere (problemorientierte) Sprachen

3 höhere prozedurale Sprachen, z.B. C, BASIC, COBOL, FORTRAN, PASCAL, PL/1

4-5 **deskriptive** Sprachen

4 Programmierumgebungen, Programmgeneratoren, Abfragesprachen, Metasprachen (Datenbank- oder Textabfragesprachen)

5 Programmiersprachen der "künstlichen Intelligenz", z.B. LISP, SMALLTALK, PROLOG

Die Programmsprachen und ihre Eigenheiten alle aufzuführen hieße den Rahmen dieser Einführungsschrift zu sprengen. Wir können uns auch im EDV-Kurs wegen dessen vorgegebener Umfangsbeschränkung nicht damit befassen. Ich kann Ihnen lediglich einen kurzen Überblick über die typischen Leistungen einiger weniger, aber besonders bedeutsamer Programmsprachen geben:

Assembler Maschinencode in Abkürzungen (Mnemonics); Compiler notwendig; sehr schnell in der Ausführung. Keine Übertragbarkeit auf andere Systeme.

C maschinennahe Compilersprache, die mit sehr wenigen Grundbefehlen auskommt; diese werden in sog. "Funktionsmodulen" zusammengesetzt und aus einer "Bibliothek" abgerufen. Einwandfreie Übertragbarkeit auf andere Betriebssysteme, schnell.

BASIC (**B**eginners **a**ll purpose **S**ymbolic **I**nstruction **C**ode) ist eine Interpretersprache, in der Ursprungsversion kenntlich an den Zeilennummern, universell einsetzbar, langsam, leicht erlernbar, beschränkt auf 64 Kbyte. Es sind bereits Compilerversionen am Markt, die diese Nachteile vermeiden, aber: zahlreiche "Dialekte" hindern die Übertragbarkeit.

PASCAL aus Lehrsprache entwickelt, compilerorientiert; "strukturierte" Programmierung" = Aufteilung in einzelne Blöcke, die für sich selbständige Unterprogramme bilden. Standardisiert, daher gut auf andere Computer übertragbar.

PROLOG (**Pro**grammation en **Log**ique = Programmieren in Logik) Programmsprache mit "eingebauter Problemlösefähigkeit" Künstliche Intelligenz"), Compiler erforderlich, stellt anhand von "Regeln" Beziehungen zwischen "Objekten" her, z.B. Aufgabe:

Peter geht in das Kino
Hans geht mit Peter
Wohin geht Hans?
Antwort des Programms: *Hans geht in das Kino*

III. Das Betriebssystem

In diesem ist bei den meisten Rechnern eine direkte Ansprache möglich. Das Betriebssystem wird vom Hersteller entweder maschinensprachlich im Festspeicher (ROM) abgelegt und von dort beim Einschalten abgerufen oder von Diskette bzw. Festplatte geladen. Neben zahllosen hauseigenen Betriebssystemen sind sehr weit verbreitet die Systeme Windows und Unix (LINUX).
Was ist nun so ein Betriebssystem?

Häufig wird gesagt, das Betriebssystem sei die Schnittstelle zwischen Mensch und Maschine; da ist wohl etwas dran. Aber genauer! Ein Betriebssystem ist
- ein in einer maschinennahen Programmsprache geschriebenes
  - auf andere Rechnertypen nicht ohne weiteres übertragbares
    - Programm, das
      - die Verbindung zwischen Benutzer und Rechner ermöglicht,
      - Voraussetzung für die Lauffähigkeit von Anwendungen ist,
      - dem Rechner die Kontrolle der Peripheriegeräte erlaubt und
      - Voraussetzung für die Verwaltung von Datenbeständen ist.

Seine **Aufgaben** sind also u.a.:

a. Verwaltung der Dateiverzeichnisse
- Anlage
- Bewegung zwischen Dateiverzeichnissen
- Zugriffserlaubnisse regeln
- Verschieben und Kopieren
- Löschen

b. Verwaltung von Dateien
- Erstellen und Ändern
- Ausgabe auf Bildschirm und Drucker
- Speicherung
- Zugriffserlaubnisse verteilen
- Verschieben, Kopieren und Umbenennen
- Löschen

c. Steuerung der Peripheriegeräte

- Laufwerke
- Festplatte
- Drucker

d. Organisation der Benutzung nur bei Mehrplatzsystemen

- Eigentümerstellungen definieren
- Zugriffserlaubnisse vergeben
- Kommunikation ermöglichen

Das Betriebssystem bietet eine Reihe von Befehlen zur Bearbeitung von Dateien. Als Datei wird eine unter einem Namen geführte und als solche speicher- und ladbare Ansammlung von Daten verstanden. Das englische Wort "**file**" (Akte) kennzeichnet den Begriff sehr gut: Der Aktendeckel verbindet die einzelnen Schriftstücke. Solche Akten müssen erst erstellt (editiert) werden, wobei hierfür zahlreiche Editierhilfen zur Verfügung stehen, um die Dateien auf dem Bildschirm so einzugeben, wie es sich der Benutzer vorstellt. Anschließend lassen sie sich nach bestimmten Begriffen abfragen, ordnen, sortieren, ausdrucken, speichern und wieder laden. Für diese Vorgänge sind entweder direkte Befehle auszusprechen, die sofort auf den Rechner wirken oder Programmdateien anzulegen, die einen geordneten Ablauf der einzelnen Schritte vorschreiben und gesondert gestartet werden müssen.

Die Datei ist bei einfachen Betriebssystemen gleichzeitig die höchste Organisationsform, in der Daten gehalten werden können. Moderne komplexe Betriebssysteme wie MSDOS oder UNIX kennen allerdings noch weitergehende Strukturen:

Mehrere Dateien werden in einem Ordner gesammelt;
mehrere Ordner bilden ein Unterarchiv,
mehrere Unterarchive ein Oberarchiv.

Dabei sind die Bezeichnungen "Ordner", "Unterarchiv", "Oberarchiv" vollkommen willkürlich. Die Schachtelungstiefe ist nämlich praktisch unbegrenzt. So sind Oberoberarchive, Oberoberoberarchive usw. denkbar. Bei Betriebssystemen, die eine derartige Struktur aufweisen, spricht man von einem "hierarchischen" Aufbau.

Kann der Rechner eine Mehrzahl von Aufgaben gleichzeitig erledigen?

Eine dumme Frage vorweg: Wozu eigentlich?

Nun, stellen Sie sich einmal vor, Sie lassen durch Ihr Textprogramm einen langen Text ausdrucken. Ist es wirklich notwendig und sinnvoll, mit der nächsten Arbeit am Computer warten zu müssen, bis der Ausdruck erledigt ist? Oder: Ein umfangreiches Programm wird compiliert, was ordentlich dauert. Könnte das nicht im Hintergrund geschehen?

Die Frage der sog. „Multitaskingfähigkeit" wird überwiegend vom Betriebssystem entschieden. Es muss bei mehreren Aufgaben gleichzeitig den Prozessor so steuern können, dass er z.B. die ersten 200 Millisekunden einer Sekunde für die Aufgabe 1, die zweiten für Aufgabe 2 usw. aufwendet. Das macht durchaus Sinn, da der Computer so schnell arbeiten kann, dass er eigentlich die meiste Zeit arbeitslos herumsitzt und auf Eingaben seines Herrn und Meisters wartet. Die dabei verstreichende Zeit sollte man doch nutzen können.

Wenn ein Betriebssystem das steuern kann, muss es auch dafür sorgen, dass sich die verschiedenen Aufgaben gegenseitig nicht in die Quere kommen, es muss also eine Aufgabe vor der anderen schützen können. Multitaskingfähige Betriebssysteme sind z.B. OS2, Windows 95, UNIX sowie die Betriebssysteme der "Main Frames" (Großrechner).

Mehrplatz- oder Einzelplatzsystem?

In der Regel entscheidet weniger die Hardware als die Software über die Frage, wie viele Benutzer an einem Zentralrechner arbeiten können. Natürlich müssen für mehrere Benutzer auch mehrere Bildschirm- und Tastaturanschlüsse (Schnittstellen) vorhanden sein, sonst fehlen einfach die Voraussetzungen zur Verbindungsaufnahme. Aber schon die Leistungsfähigkeit des Zentralprozessors ist keine ausschlaggebende Voraussetzung für die Mehrplatzfähigkeit des Rechners. Natürlich wäre der Anschluss mehrerer Bildschirme an einen Heimcomputer ein Belastungstest für die Geduld der Benutzer, aber im Prinzip stünde nichts entgegen.

Das Betriebssystem muss bei mehreren Benutzern auch multitaskingfähig sein; das reicht aber allein noch nicht. Wenn ein Betriebssystem den Anforderungen, die nun einmal an Datensicherheit und Datenschutz zu stellen sind, genügen will, dann muss es auch verhindern können, dass ein Benutzer im Gebiet eines andern "wildern" und z.B. dessen Daten lesen, verändern oder gar löschen kann.

Echte Mehrplatzfähigkeit in einem einzelnen System zeigen neben UNIX nur die Betriebssysteme der "Main Frames". Windows-Rechner werden durch eine Software so vernetzt, dass mehrere Rechner gleichzeitig auf eine – auf dem Server laufende – Datenbank zugreifen können. Die Mehrplatzfähigkeit wird also durch die Software gewährleistet.

Benutzerfreundlichkeit

Die meisten Betriebssysteme stammen noch aus der Zeit, als harte Männer noch richtige harte Männer waren und man sich damit abfand, dass man zur Inbetriebnahme eines Computers eben ein absolut unverständliches Kauderwelsch beherrschen musste. In dem Maße, in dem sich der Computer weitere Bevölkerungskreise - speziell solche ohne Informatikstudium - erschloss, wurden die Softwareproduzenten nachdenklich im Hinblick auf die Benutzerfreundlichkeit. Plötzlich waren Programme auf dem Markt, die dem Anwender seine eigene Oberfläche spendierten, mit grafischen Hilfestellungen arbeiteten und vor allem die notwendigen Befehle als Auswahlpunkte sogenannter Menüs boten. Die meisten Betriebssysteme arbeiten jetzt mit grafischen Oberflächen, die eine Bedienung auf Kommandozeilenebene überflüssig machen. Da gibt es zwar immer noch diejenigen, die derartige Segnungen der Zivilisation genauso ablehnen wie der gestandene deutsche Autofahrer ein Automatikgetriebe, die an den Lagerfeuern der EDV-Prärie sitzen und von alten Zeiten schwärmen. Sie werden aber weniger …