Public-Domain-Software für den Atari ST
Anwenderprogramme

Robert Tolksdorf

Reproduktion der Papierausgabe von 1999

Bismarckstr. 18
14109 Berlin

Contents

Chapter 1

Einleitung

Der Atari ST ist ein idealer Computer für Hobbyanwender. Seine Leistung war bis zu seiner Einführung nicht oder nur zu unerschwinglichen Preisen erhältlich. Die grafische Umgebung GEM und die vielen verfügbaren Hochsprachen wie C, Pascal oder Modula-2 machen ihn jedoch auch zu einem ernsthaften Arbeitsmittel, das sowohl Programmierfreaks als auch Anwender fasziniert.

Die Atari-ST Besitzer entdeckten schnell das Public-Domain-Konzept, das bislang vor allem bei CP/M- und MS-DOS-Systemen verbreitet war. Public-Domain bedeutet, daß ein Programmierer auf eine Vermarktung eines Programms verzichtet und es der Öffentlichkeit zur Verfügung stellt. All diese Programme können völlig legal kopiert, benutzt und weitergegeben werden. Die Autoren sind sogar daran interessiert, daß ihr Programm eine weite Verbreitung findet.

Der Autor kann so seinen Namen bekanntmachen und Programmierfähigkeiten unter Beweis stellen. Für viele Autoren haben sich aus dieser Bekanntheit weitere Kontakte ergeben, die sich in der Zusammenarbeit mit Software-Häusern oder Verlagen in barer Münze auszahlen.

Für den Anwender sind Public-Domain-Programme die Möglichkeit Software zu benutzen, die semi-professionellen Ansprüchen durchaus gerecht werden kann. Der Hobbyist kann sich kaum eine komplette Software-Bibliothek für alle Anwendungen kaufen, denn auch wenn Software für den Atari ST preisgünstig ist, kommt man mit einer vollständigen Ausstattung mit Programmiersprachen, Textverarbeitung, Grafikprogrammen oder Kommunikationssoftware leicht auf vierstellige Beträge. Public-Domain Programme sind hingegen praktisch kostenlos und leicht verfügbar. Viele Autoren bieten für einen geringen Obulus einen Update-Service an, in dessen Rahmen sie eventuell verbesserte Programmversionen nachliefern.

Raubkopierer werden sich eine ähnliche Sammlung aus “professioneller”

Software mit zugegebenermaßen stärkeren Leistungen ebenfalls schnell und "preiswert" zulegen können. Doch gehen sie dabei ein hohes Risiko ein, denn falls man ihnen auf die Schliche kommt, müssen sie mit Gerichts- und Anwaltskosten sowie Schadensersätzen rechnen, die ebenfalls in die Tausender gehen.

Somit ist Public-Domain-Software ein Ausweg aus dem Dilemma, zwar einen preiswerten Rechner zu besitzen, die Software-Kosten, die eine vernünftige Anwendung ermöglichen, aber nicht tragen zu können. Dieses Buch ist ein Beitrag zur Verbreitung dieses Konzepts.

Es ist inzwischen ein leichtes, über 100 doppelseitige Disketten mit Public-Domain-Programmen für den Atari ST zu füllen. Da diese Menge von über 70 Megabyte an Programmen natürlich nicht mehr überschaubar ist (und täglich anwächst), wurden für dieses Buch die besten Programme aus den wichtigsten Anwendungskategorien ausgewählt, so daß der Leser sich ein leistungsstarkes System zusammenstellen kann. Die vorgestellten Programme decken alle Anwendungen, wie Textverarbeitung oder Grafik ab. Dateiverwaltung, Tabellenkalkulation und Kommunikation vervollständigen die Anwendungsbereiche.

Oft ergeben sich neben der Unübersichtlichkeit der verfügbaren Software für viele Probleme aus der Tatsache, daß Anleitungen für die Programme in Fremdsprachen verfaßt sind oder nicht beiliegen. Sie finden daher zu jedem vorgestellten Programm eine Art kleines Handbuch, das Ihnen die sinnvolle Arbeit mit Ihrer Public-Domain-Software ermöglicht.

Selbstverständlich kann Public-Domain-Software nicht alle Spezialprobleme lösen, wie z.B. das immer währende Thema Druckeranpassung. Fragen Sie doch in solchen Fällen einfach einen Freund oder achten Sie auf Disketten, die von Zeitschriften oder Public-Domain-Versendern angeboten werden. Man kann bei der Fülle der Programme für den Atari ST davon ausgehen, daß jedes wirkliche Problem schon gelöst ist. Falls nicht, erproben Sie doch einmal selber Ihre Programmierkenntnisse und geben Sie das Ergebnis als Public-Domain weiter – vielleicht können Sie so jemanden helfen, der vor den gleichen Schwierigkeiten steht.

Berlin, August 1988 Robert Tolksdorf

Chapter 2

Public-Domain, was ist das ?

Es gibt mehrere Begriffe, die im Allgemeinen unter *Public-Domain* zusammengefaßt werden. Public-Domain, Shareware, Partialware, Freeware oder Beggarware sind Klassifizierungen, die oft in Anleitungen zu "frei kopierbaren Programmen" auftauchen. Leider sind sie oft nicht genau genug erklärt.

Daher zunächst ein kleiner Ausflug in das Urheberrecht, wobei ich Sie natürlich nicht mit juristischen Details langweilen will (ich übergehe dabei länderspezifische Eigenheiten des Urheberrechts). Wenn ein Programmierer Software schreibt, entsteht dadurch eine eigene geistige Schöpfung, d.h. er verwendet nicht nur sein technischen Know-How, sondern läßt auch seine Kreativität spielen. Dies mag eine abstrakte Programmfunkion betreffen, aber auch die Gestaltung z.B. eines Icons. Dadurch, daß er etwas neues schafft, ist sein Werk urheberrechtlich geschützt.

Seine Rechte an einem Programm lassen sich aufteilen in die eigentlichen Urheberrechte und in die Verwertungsrechte. Die Urheberschaft besagt, daß ein Stück Software die Schöpfung eines bestimmten Programmierers ist. Dieses Recht ist eher ideeller Art und läßt sich nicht übertragen oder verkaufen.

Anders ist das bei den Verwertungsrechten. Aus ihnen ergeben sich im Besonderen die materiellen Vorteile, die durch die Verbreitung eines Programms entstehen. Bei allen frei kopierbaren Programmen gibt der Autor Teile dieser Rechte an die Öffentlichkeit, also an alle interessierten Computer-Benutzer kostenlos weiter.

Im Allgemeinen betrifft das zunächst das Recht zur Vervielfältigung und

Weitergabe, sprich das Kopieren von Dateien, Programmen oder ganzen Disketten nicht nur für persönliche Sicherungskopien. Die Autoren haben meistens ein Interesse daran, daß sich ihre Programme durch Austausch zwischen Computer-Hobbyisten verbreiten. Dadurch werden ihre Namen bei Anwendern bekannt und sie erhalten die persönlich Befriedigung, daß ihre Programme Anklang finden und von Interesse sind.

Ein weiteres Verwertungsrecht ist das Nutzungsrecht, also die Erlaubnis ein Programm zu benutzen. So banal das klingen mag, wenn Sie ein kommerzielles Programm erwerben, bezahlen Sie eigentlich nur für dieses Recht. Da Software keinen faßbaren Gegenstand darstellt, und das Drumherum, wie Handbücher oder Programm-Disketten sicherlich nicht den Programmpreis aufwiegt, ist die Nutzungslizenz der eigentliche Teil des Kaufvertrages. Sie schließen ihn – beim Barkauf meist ohne daß Sie es eigentlich merken – mit dem Händler oder mit dem Software-Haus.

An dieser Stelle unterscheiden sich *Public-Domain* und *Shareware*. Bei ersterem erteilt der Autor allen, die das Programm haben, automatisch und kostenlos die Nutzerlizenz. Public-Domain Programme sind also wirklich "freie" Programme, bei denen keine Kosten entstehen und bei denen die Programmierer alle Verwertungsrechte aufgeben. Freie Programme sind also ohne Rechtsverletzungen frei kopierbar.

Bei *Shareware* dagegen müssen Sie die Nutzerlizenz extra erwerben, indem Sie dem Autor einen bestimmten Geldbetrag zukommen lassen. Dessen Größe liegt meist bei DM 10 bis DM 50, bei amerikanischen Programmen bei ca. $10 bis $30. Als Gegenleistung erhalten Sie meistens eine Registrierung als ordnungsgemäßer Nutzer und einen Update-Service. Strenggenommen ist also die Benutzung von Shareware ohne Registrierung nicht erlaubt. Das Kopieren und Weitergeben ist jedoch uneingeschränkt möglich.

Der Begriff *Freeware* ist identisch mit Shareware. Jedoch ist dieser Ausdruck inzwischen geschützt (durch Andrew Fluegelmann, Autor eines Terminalprogramms für den IBM-PC) und darf deshalb nicht mehr verwendet werden.

Partialware – ein weitere Wortschöpfung, die von "Software" abgeleitet wurde – ist nicht so verbreitet. Dabei wird Public-Domain und Shareware kombiniert: "Im Umlauf" ist eine Programmversion als Public-Domain, die jedoch nicht alle Funktionen des eigentlichen Programms enthält. Durch eine Registrierung wie bei der Shareware erhalten Sie eine vollständige oder erweiterte Programmversion. Oftmals – z.B. bei Rechtschreibprüfprogrammen oder bei Astronomie-Software – erhalten Sie bei der Registrierung Daten-Files wie z.B. Wörterbücher oder Sternenkataloge, durch die das Arbeiten mit dem Programme erst richtig ermöglichen.

Der Ausdruck *Beggarware* ("Bettlerware") entstand angeblich bei der amerikanischen Public-Domain Sammlung für CP/M-Programme SIG/M,

die – etwas puristisch – Shareware und jeglich kommerzielle Absichten ablehnt. Obwohl Beggarware etwas abschätzig klingt, könnte man den Begriff auch auf Public-Domain Programme anwenden, bei denen der Autor eine kleine Spende erbittet, ohne gleich das Shareware-Konzept zu anzuwenden.

Den vielen Vorführ-Disketten zu kommerziell vertriebenen Programmen möchte ich den Namen *Demoware* geben. Man bekommt meistens in einer automatisch ablaufenden "Show" einige Fähigkeiten von käuflichen Programmen gezeigt und muß dann nicht mehr die Katze im Sack kaufen. Das Kopieren und Weitergeben solcher Demos ist in der Regel erlaubt, allerdings bringen sie für nicht-kaufinteressierte wenig.

Auch wenn Sie programmieren können, frei kopierbare Programmen sollten Sie nicht verändern. Es kann sogar sein, daß der Autor ausdrücklich Änderungen verbietet und den Source-Text nicht zur Verbreitung freigibt. Teilen Sie Anregungen lieber dem Autor direkt mit, so daß er sie in eine neue Version einbauen kann. Die oftmals beiliegenden Quell-Codes der Programm dienen meistens nur dazu, interessierten Programmierern Einblick in verwendete Tricks zu bieten. Falls Sie doch etwas verändert haben, vermerken Sie dies ausdrücklich als Kommentar in dem neuen Source-Text und beschreiben Sie die Wirkungsweise Ihrer Änderungen genau!

Was mag nun die Motivation sein, Programme auf diesem Weg in Umlauf zu bringen und auf Rechte zu verzichten ? In der Steinzeit der Hobby-Computerei waren die Rechnerbesitzer auf selbstgeschriebene Programme angewiesen, da es keine Software-Häuser gab, die sich für "Spielzeuge" interessierten.

Heutzutage kann es aber nur noch Idealismus sein, der uns die vielen Public-Domain Programme von teilweise erstaunlicher Qualität beschert. Viele auch noch so kleine Utilities werden immer noch für teures Geld angeboten, denken Sie nur an die vielen kommerziellen RAM-Disks für den Atari ST. Vielleicht ist Public-Domain ein Mittel, Software-Häuser zu einer vernünftigeren Produkt- und Preispolitik zu bewegen. Shareware soll sich in den USA für die Autoren oftmals auch finanziell auszahlen; das mag an der etwas anderen Mentalität unter der Computer-Freaks dort liegen.

Ich werde in diesem Buch nur noch von *Public-Domain* sprechen. Sie wissen jetzt, was sich hinter den Begriffen in Programmanleitungen verbirgt, und was es mit den Aufforderungen zu kleinen Spenden auf sich hat. Wenn Sie ein Programm wirklich nutzen können und für eine bestimmte Anwendung keine kommerzielle Software kaufen müssen, sollten Sie dem Autor Ihre Anerkennung auch in barer Münze zollen!

Chapter 3

Der Texteditor PROED

Umfragen haben ergeben, daß die Hauptanwendung von Computern im privaten Bereich eindeutig die Textverarbeitung ist. In den folgenden Kapiteln lernen Sie drei Public-Domain Programme kennen, mit denen Sie ein komplettes Textsystem zur Verfügung haben. Es handelt sich um einen Texteditor, einen leistungsstarken Textformatierer und ein Hilfsprogramm zum Ausdruck von Listings oder Korrekturen.

Um Texte auf einem Computer zu erzeugen brauchen Sie einen Texteditor. Auf dem Atari ST gibt es verschiedenen kommerzielle Systeme, die unter GEM laufen und auch Leistungen wie Spaltenverarbeitung und Grafikeinbindung bieten. Auf dem Public-Domain-Sektor sind Editoren meist mit einer "normalen" Umgebung, also als TOS-Applikationen, bei denen die Kommandoeingabe über Tastenkombinationen erfolgt. Gründe dafür sind die einfachere Programmgestaltung unter TOS und die einfache Anpassung von Editoren, die für andere Systeme ohne GEM geschrieben wurden.

Für deutschsprachige Benutzer sind diese angepaßten Programme leider mit dem Nachteil verbunden, daß Umlaute meist nur wenig oder überhaupt nicht unterstützt werden. Bei der Auswahl eines Editors für dieses Buch wurde daher vor allem darauf geachtet, daß er die Eingabe von Umlauten ermöglicht. Weiterhin ist ein Texteditor etwas anderes als ein Programmeditor, mit dem Quelltexte für Programmiersprachen geschrieben werden. So können Programmeditoren in der Regel keinen automatischen Zeilenumbruch durchführen, wodurch eine weitere Bedingung für die Auswahl eines Programms aufgestellt war.

Ich habe mich schließlich für das Sharware-Programm PROED von Jerry Cole entschieden, da es meine Ansprüche an einen Texteditor weitgehend erfüllt. Die Auswahl eines Editors ist zwar sehr stark von den

persönlichen Gewohnheiten bestimmt; trotzdem glaube ich, daß dieses Programm für alle Geschmäcker genug bietet.

Um Beschwerden von Germanisten vorzubeugen: Das Verb, das das Benutzen eines Editors beschreibt lautet korrekterweise "edieren". Nun hat sich aber allgemein die sprachlich falsche Schreibweise "editieren" eingebürgert, die ich auch verwenden werde. "Computerchinesisch" bringt eben oft seltsame Sprachschöpfungen hervor.

3.1 Programmstart

Nach dem Programmstart vom GEM-Desktop aus wird der Bildschirm in den Textmodus umgeschaltet und PROED fragt Sie in der untersten Bildschirmzeile nach dem Namen der zu editierenden Datei. Sie haben an dieser Stelle vier Möglichkeiten. Wenn Sie eine schon bestehende Textdatei bearbeiten wollen, geben Sie einfach deren Namen an. PROED lädt den Text und Sie können mit der Bearbeitung beginnen.

Falls Sie den Datei-Namen nicht mehr genau wissen, können Sie sich ein Inhaltsverzeichnis Ihrer Diskette oder Festplatte anzeigen lassen. Dazu geben Sie anstelle des Filenamens eine Maske ein. Eine Maske besteht aus festen Zeichen und den Platzhaltern `?` und `*`. PROED sucht nun alle Datei-Namen, auf die die Maske paßt und zeigt sie an.

Für den Platzhalter `?` paßt jedes einzelne Zeichen und für `*` jede Zeichenkette. Die Maske `KAP?.TXT` würde z.B. auf `KAP1.TXT` oder `KAPA.TXT` passen, nicht aber auf `KAPITEL.TXT` oder `KAP.TXT`. Bei einer Maske `KAP*.*` würden im Inhaltsverzeichnis `KAPITEL.DOC` oder `KAPA.TXT` stehen, nicht aber `KAP`. Die Platzhalter, die auch "Wildcards" genannt werden, können in einer Maske auch gemischt vorkommen.

Um einen neuen Text zu erstellen, drücken Sie einfach `Return`. Den Namen der Datei, in die PROED das Dokument schreiben soll, legen Sie dann beim Abspeichern fest. Alle Dateien sucht PROED im aktuellen Directory, also immer in dem Ordner, aus dem Sie den Editor gestartet haben. Wenn Sie auf Texte in anderen Ordnern oder auf anderen Laufwerken zugreifen wollen, müssen Sie den Pfadnamen entsprechend miteingeben.

Als letzte Aktion, die bei der Auswahl des Datei-Namens möglich ist, können Sie mit der `Esc`-Taste die Eingabe abbrechen. Damit verlassen Sie PROED und haben wieder den GEM-Desktop vor sich.

Diese Möglichkeiten haben Sie immer, wenn PROED nach einem Datei-Namen fragt. Sie können einen Namen eingeben, ein Inhaltsverzeichnis anzeigen oder mit `Esc` die Eingabe abbrechen. Mit der `Help`-Taste können Sie sich diese Optionen auch von PROED beschreiben lassen – allerdings in englischer Sprache.

Wenn Sie PROED mit der Endung `.TTP` auf Ihrer Diskette verwenden, können Sie beim Start vom Desktop den Namen des Textes, den Sie bearbeiten wollen, als Parameter übergeben.

Wenn PROED im Bearbeitungsmodus ist, finden Sie in der ersten Bildschirmzeile einige hilfreiche Angaben. Ganz links oben sehen Sie den Namen der gerade bearbeiteten Datei. Wenn Sie einen neuen Text erzeugen steht dort `New File`.

Die Cursor-Position finden Sie in der Mitte der ersten Zeile. Dort zeigt PROED die `Line.` die Zeile und bei `Col.` die Spalte an, in denen sich der Cursor befindet. Auf der rechten Seite der ersten Bildschirmzeile können bis zu sechs Angaben über eingestellte Modi stehen. Sie werden mit den Schalterfunktionen beeinflußt, die Sie in Kapitel 3.8 auf Seite 22 beschrieben finden.

Ab der zweiten Bildschirmzeile zeigt PROED den Text an. Das Ende der Datei wird durch die Zeile `--EOF--` markiert. In der untersten Zeile können Sie an Kommandos Parameter übergeben, wie Sie es schon bei der Eingabe des Datei-Namens kennengelernt haben.

3.2 Cursor-Bewegungen

Wenn der Text auf dem Bildschirm angezeigt wird, können Sie den Cursor auf vielfältige Weise bewegen. Die Pfeiltasten des Atari ST setzen ihn jeweils ein Zeichen nach links oder rechts bzw. eine Zeile nach oben oder unten. Für diese einfachen Bewegungen gibt es jeweils eine Alternative:

$\leftarrow$ = Control+B = Cursor links
$\rightarrow$ = Control+F = Cursor rechts
$\uparrow$ = Control+P = Cursor nach oben
$\downarrow$ = Control+N = Cursor nach unten

Mit "Control+B" ist gemeint, daß Sie gleichzeitig die `Control`-Taste und B drücken. Analog dazu bedeutet "Alternate+D" das Drücken von `Alternate` und D.

Falls Sie – vielleicht aus Versehen – die Maus berühren, werden Sie bemerken, daß sie ebenfalls Cursor-Bewegungen auslöst. Bei PROED ist die Maus so geschaltet, daß sie wie der Cursor-Block arbeitet; Maus nach links entspricht also der Taste $\leftarrow$. Die Bedeutung der Mausknöpfe werden Sie später kennenlernen.

Durch die Tastenkombinationen Alternate+B und Alternate+F läßt sich der Cursor jeweils um ein Wort nach links oder nach bewegen. Ein Wort bedeutet dabei, daß er auf das nächste Leerzeichen in der entsprechenden Richtung gesetzt wird.

Control+S und Control+E verschieben den Cursor an den Beginn bzw. an das Ende der aktuellen Zeile. Mit Ende ist nicht der rechte Bildschirmrand gemeint, sondern das erste Leerzeichen nach dem letzten Wort in der Zeile.

Die nächstgrößeren Cursor-Bewegungen in vertikaler Richtung verschieben den Zeiger um eine Seite: Alternate+P oder Funktionstaste 2 (F2) nach oben und Alternate+N oder F1 nach unten. Dabei wird jeweils um 20 Textzeilen gescrollt, ein Wert, der mit einem Kommando, das Sie gleich kennerlernen allerdings verändert werden kann.

Zum Anfang des Textes gelangen Sie durch Alternate+S oder Shift+F2 und zum Ende mit Alternate+E oder Shift+F1. Control+Z macht die aktuelle Zeile zur ersten Zeile auf dem Bildschirm, wobei der Cursor dabei allerdings nicht verschoben wird.

Die nächsten beiden Cursor-Kommandos werden nicht mit einer Kombination mit der Control- oder Alternate-Taste eingegeben. PROED kennt in einer zweiten Kommandoebene die "extended commands". Sie werden durch die Taste F10 oder Control+X eingeleitet und bestehen jeweils aus zwei Zeichen. Nach F10 erscheint in der untersten Bildschirmzeile `Command: __` und der Cursor wartet auf eine Eingabe. Ich werde die erweiterten Kommandos mit dem Zeichen ↪ kennzeichnen. Wenn Sie aus Versehen zu dieser Eingabe geraten, können Sie mit `Esc` wieder in den Eingabemodus zurückkehren.

Das erste erweiterte Kommando, das Sie kennenlernen ist ↪SL. Sie müssen also zunächst F10 oder Control+X drücken, dann die Buchstaben `SL` eingeben und mit `Return` bestätigen.

Mit dem Kommando können Sie eine bestimmte Zeile des bearbeiteten Textes direkt erreichen. Nach der Befehlseingabe erwartet PROED mit `Show Line Number: ____` die Eingabe einer Zeilennummer. Der Cursor springt dann an die eingegebene Zeile, was besonders praktisch ist, wenn Sie Fehler korrigieren wollen die Ihnen z.B. ein Textformatierer angezeigt hat.

Das zweite erweiterte Kommando, ↪MR legt die Anzahl der Zeilen fest, um die PROED bei einer Seitenbewegung – also bei Alternate+N und Alternate+P – den Cursor bewegen soll. In der letzten Bildschirmzeile müssen Sie bei `Modify Roll - # of Lines: ___` die entsprechende Zahl eingeben. Als Voreinstellung bewegt PROED den Cursor – und damit den Bildschirminhalt – 20 Zeilen; um auf 10 Zeilen umzustellen wäre ↪MR 10 notwendig.

Neben diesen Befehlen können Sie auch mit der `Tab`-Taste den Cursor zur nächsten Tabulatorposition bewegen. Dazu müssen allerdings Tabulatoren definiert sein.

Einen Tabulator definieren Sie, indem Sie Control+T drücken. PROED

setzt nun intern eine Marke in der Spalte, in der sich der Cursor befindet. In der ersten Bildschirmzeile, in der auch die aktuelle Cursor-Position angezeigt wird, erscheint ein `t` hinter der Spaltenangabe, wenn sich der Zeiger auf einer Tabulatorposition befindet. Durch erneute Eingabe von Control+T können Sie einen Tabulator wieder löschen.

3.3 Löschen

Für das Löschen von Textteilen gibt es bei PROED ebenfalls mehrere Kommandos. Die Eingabe von `Delete` oder Control+D löscht das Zeichen unter dem Cursor. Mit `Backspace` können Sie das Zeichen links vom Cursor löschen, wobei er zusätzlich um ein Zeichen nach links bewegt wird.

Das Wort, auf dem der Cursor momentan steht, läßt sich mit Alternate+D löschen. Steht er auf einem Leerzeichen, so löscht das Kommando alle Leerzeichen bis zum nächsten Wort. Control+K löscht alle Zeichen rechts vom Cursor bis zum Ende der Zeile. Die aktuelle Zeile wird mit Control+R komplett entfernt. Einen Löschbefehl für größere Textabschnitte finden Sie in Kapitel 3.6 auf Seite 20.

Mit der `Undo`-Taste ist es möglich, Löschoperationen rückgängig zu machen. Bei "kleinen", also zeichen- oder wortweisen Löschungen geht dies allerdings nur, wenn sich der Cursor noch in der gleichen Zeile befindet. PROED speichert immer nur das ursprüngliche Aussehen der aktuellen Zeile in einem internen Puffer. Komplett gelöschte Zeilen hingegen werden von PROED wieder an der Stelle in den Text aufgenommen, an der sie vor dem Löschen standen.

3.4 Marken

Um Textstellen schnell aufzufinden, können Sie diese markieren. Wenn Sie F5 drücken merkt sich PROED die Position des Cursors. Sie haben zwei Marken zur Verfügung, die Sie mit Alternate+L oder F3 anspringen können.

Da es zwei Markierungen gibt und nur ein Kommando zum Erreichen der Marken, springt PROED sie in der Reihenfolge der Eingabe an. Wenn Sie zum dritten Mal F5 benutzen, wird die erste Marke durch die jetzige Cursor-Position ersetzt.

Falls Sie den Text ändern, also Zeilen hinzufügen oder entfernen, so verschieben sich die Marken automatisch mit. PROED merkt sich also nicht etwa eine Zeilennummer, sondern tatsächlich eine Textposition.

3.5 Zeilenoperationen

Wenn Sie neuen Text einfügen wollen, können Sie Mit Control+U für Sie eine Leerzeile unter der aktuellen ein, um sich Platz für neuen Text zu schaffen. Dabei springt der Cursor automatisch an den Anfang dieser Zeile. Wenn Sie den Compose-Modus eingeschaltet haben (dazu mehr in Kapitel 3.8 auf Seite 22) wird auch mit der `Return`-Taste automatisch eine neue Zeile erzeugt. Wollen Sie eine Zeile über der Cursor-Position einfügen, müssen Sie Control+O benutzen.

Eine Zeile können Sie mit Control+Y an der Cursor-Position teilen. Der Zeilenteil rechts vom Cursor wird in eine neue Zeile darunter verschoben. Control+J leistet das Gegenteil, das Zusammenfügen zweier Zeilen. Das Kommando verbindet die Zeile über dem Cursor mit der aktuellen. Dabei dürfen beide Zeilen zusammen aber nicht mehr als 80 Zeichen umfassen, ansonsten erscheint die Meldung `Join Lines not performed....Line would exeed column 80` und der Befehl wird nicht ausgeführt.

Die Zeichen der Zeile über der Cursor-Position können Sie mit Control+C in die aktuelle kopieren. Dabei werden alle Zeichen ab der Cursor-Spalte bis zum Zeilenende übernommen.

Mit Control+A kann schließlich die aktuelle Zeile eingerückt werden. Dabei verschiebt PROED die Zeichen so, daß das erste Zeichen der Zeile an die aktuelle Cursor-Position kommt. Insgesamt darf die Zeile nicht länger als 80 Zeichen werden, falls doch, meldet PROED `Alignment not performed....Line would exeed column 80` und verweigert die Einrückung. Den gleichen Befehl können Sie auch innerhalb von Blöcken verwenden (siehe Kapitel 3.6).

3.6 Blockoperationen

Wenn Sie mit PROED mehrere Textzeilen oder Absätze z.B. verschieben oder löschen wollen, stehen Ihnen die Blockbefehle zur Verfügung. Dazu müssen Sie einen Textteil als Block markieren, auf den dann verschiedenen Kommandos angewandt werden können. Sie ersparen sich damit umständliche Kommandofolgen, z.B. ein hundertmaliges Zeilenlöschen.

Den Anfang und das Ende eines Blockes markieren Sie mit F4 oder Control+M. Ist noch kein Block benutzt, wird durch F4 Anfang und Ende auf die Zeile gelegt, in der sich der Cursor befindet. PROED zeigt alle Zeilen, die zu einem Block gehören invertiert an. Sie können nun einfach zum "anderen Ende" des Blocks bewegen, egal ob nach oben oder nach unten im Text. Eine erneute Eingabe von F4 setzt die Markierung. Wenn Sie die Maus verwenden wollen, können Sie alternativ mit dem linken Mausknopf

den Block markieren.

Ein erstes Block-Kommando ist Control+A. Es richtet alle Zeilen des Block nach der aktuellen Cursor-Position aus. Wenn der Cursor in Spalte 4 steht, und Sie das Kommando auslösen, wird der Block um 4 Zeichen eingerückt. Keine der Zeilen darf durch das Einrücken länger als 80 Zeichen werden.

Zum Verschieben einer markierten Textpassage an eine andere Stelle dient ↪MB. Da es sich um ein Kommando aus den "extended commands" handelt, müssen Sie es mit F10 oder Control+X einleiten. Das Kommando bewegt den vorher definierten Block an die momentane Cursor-Position. Anstelle von ↪MB können Sie auch den rechten Mausknopf verwenden, was schneller als die Kommandoeingabe geht.

Beim Verschieben wird der Block an seiner ursprünglichen Position entfernt. Mit ↪CB können Sie ihn auch kopieren, also eine Textpassage vervielfältigen. Die Kopie wird an der aktuellen Cursor-Position eingefügt.

Mit ↪DB haben Sie ein weiteres Löschkommando zur Verfügung. Es entfernt den definierten Block, der allerdings mit der `Undo`-Taste wiederhergestellt werden kann.

Nach einem Blockkommando werden die Markierungen entfernt. Sie können diese aber auch mit Shift+F4 entfernen, wenn Sie aus Versehen einen Block markiert haben. Gleichwertig dazu ist die Eingabe der Tastenkombination Alternate+M.

3.7 Suchen und Ersetzen

Mit Such- und Ersetzkommandos können Sie schnell bestimmte Textpassagen aufsuchen oder automatisch ein Wort im ganzen Text durche ein anderes ersetzen.

Mit ↪FS lösen Sie das Suchkommando aus. Der Editor fragt Sie unter `Search Value:` nach einer Zeichenkette, die er suchen soll. Bei dessen Eingabe können Sie auch das Zeichen `?` als Platzhalter verwenden. `E???ing` findet sowohl "Editing" als auch den "Ehering". Da es sein könnte, daß Sie auch das Zeichen `?` suchen, fragt PROED nach, ob es als Platzhalter verwendet werden soll (`Use ? character for wildcard?`). Normalerweise antworten sie hier `Y`, da Platzhalter geschickt eingesetzt eine praktische Sache sind. Das `?`-Wildcard haben Sie bereits bei der Eingabe von Datei-Namen kennengelernt.

PROED setzt den Cursor auf die Stelle, an der er die gesuchte Zeichenkette zum ersten Mal findet. Mit Control+L können Sie den Editor erneut nach der gleichen Zeichenkette suchen lassen.

↪RS ist der Ersetzungsbefehl. Dazu müssen Sie eingeben, welche Zeichenkette PROED suchen und durch welche es sie ersetzen soll. Auch hier kann `?` als Platzhalter verwendet werden. PROED kann sogar Platzhalter im Ersetzungs-String verarbeiten. Haben Sie einen Text "mega2 mega4" können Sie mit "RS", "mega?" und "MEGA ST?" als Ergebnis "MEGA ST2 MEGA ST4" erzeugen. Neben der obligatorischen Frage, ob `?` Platzhalter ist, läßt PROED Sie auch noch bestimmen, ob Sie jede Ersetzung einzeln bestätigen wollen (`Verify each occurrence? (Y,N)`). Bei längeren Texten ist das vielleicht sinnvoll, besonders wenn Sie Platzhalter benutzen.

Als kleinen Komfort bietet PROED auch noch das ↪CS-Kommando an, das zählt, wie oft die angegebene Zeichenkette im Text vorkommt. Die Eingabe läuft wie bei ↪FS ab; nach der Suche zeigt PROED die Anzahl der gefundenen Entsprechungen neben `Occurrances Found:` als Zahl an.

Alle diese Befehle wirken ab dem Cursor, bearbeiten also nur den Text "unter" der aktuellen Position. Wenn sich der Cursor in einem definierten Block befindet, wirken Such- und Tauschkommandos nur in diesem Textbereich.

3.8 Schalter

PROED besitzt verschiedene Schalter, über deren Einstellung Sie rechts in der ersten Bildschirmzeile informiert werden. Mit der `Insert`-Taste können Sie zwischen Einfüge- und Überschreibmodus umschalten. Bei Einfügen macht PROED für jedes neu eingegebene Zeichen Platz, beim Überschreiben ersetzt es den Buchstaben unter dem Cursor. Die `Backspace`-Taste verhält sich auch unterschiedlich: Im Einfügemodus löscht sie das Zeichen links vom Cursor; beim Überschreiben wird es lediglich durch ein Leerzeichen ersetzt. Die Anzeige `Ins` zeigt den Einfügemodus an.

Der Compose-Modus, den Sie mit F9 umschalten, legt das Verhalten der `Return`-Taste fest. Ist er eingeschaltet – auf dem Bildschirm wird `Com` angezeigt – erzeugt sie ebenso wie Control+U eine neue Zeile. Bei ausgeschaltetem Compose-Modus bewegt PROED bei Eingabe von `Return` den Cursor lediglich an den Anfang der nächsten Zeile.

Mit F8 wird der Einrückmodus aktiviert. Wenn am Anfang einer Zeile Leerzeichen stehen, rückt PROED bei eingeschaltetem Modus jede neue folgende Zeile um ebensoviele Leerzeichen ein. Es wird dann praktisch ein linker Rand beachtet. In der Anzeige wird der Modus durch `Indent` markiert.

Bei der Eingabe von Fließtext wollen Sie sicher nicht wie bei einer Schreibmaschine am Ende einer Zeile `Return` drücken. Vielmehr soll PROED automatisch eine neue Zeile beginnen, wenn das gerade eingegebene Wort

nicht mehr auf die aktuelle paßt. Dazu dient der Word-Wrap, den Sie mit F7 schalten können und der mit `Wrap` angezeigt wird.

Ich hatte Ihnen am Anfang dieses Kapitels über PROED beschrieben, daß die Eingabe von Umlauten eines der Kriterien für die Auswahl des Programms war. Nach einigem Experimentieren werden Sie festgestellt haben, daß Umlaute in PROED normalerweise ignoriert werden. Es gibt dafür einen speziellen Modus, den Sie mit Alternate+C einschalten müssen. Der Editor nimmt dann alle Zeichen an, die normalerweise als Sonder- oder Steuerzeichen gelten, also auch die Umlaute. Er zeigt das durch `Ctl` in der ersten Bildschirmzeile an.

Der Nachteil ist allerdings, daß Sie im Steuerzeichen-Modus auf alle Kommandos mit der Control-Taste verzichten müssen. Dafür können Sie aber längere Texte eingeben, ohne die Umlaute jedesmal mit einer Taste einleiten zu müssen. Leider ist dieses Problem typisch für Programme, die aus den USA stammen und trifft alle Benutzer, die spezielle Zeichen verwenden, denken Sie nur an die vielen Accents im Französischen.

Der letzte Modus, den PROED kennt, ist der Outline-Modus. "Outlining" oder "Folding" ist ein Konzept, das bei einigen Editoren angeboten wird und bei bestimmten Texten recht nützlich ist. Im Outline-Modus, der durch Alternate+O eingeschaltet und mit `Out` angezeigt wird, zeigt PROED nur die Zeilen an, die links von der aktuellen Cursor-Spalte Zeichen enthalten.

Stellen Sie sich einen Text vor, in dem alle Zeilen eines Kapitels außer der Überschrift um fünf Zeichen eingerückt sind. Die Abschnitte in den Kapiteln werden um zehn Zeichen eingerückt, wieder ohne die Überschriften. Mit dem Outline-Modus können Sie sich nun sehr einfach einen Überblick über die Struktur des Textes verschaffen. Steht der Cursor in der fünften Spalte werden nur noch die Kapitelüberschriften angezeigt. Steht er in der zehnten Spalte zeigt PROED den eigentlichen Inhalt der Abschnitte nicht an. Sie können so die Gliederung eines Textes auf einen Blick erkennen.

PROED erlaubt im Outline-Modus leider nur die Bewegung des Cursors und keinerlei Eingaben. Sie können damit allerding sehr schnell von einem Kapitel zum anderen gelangen, wenn Sie Ihren Text wie beschrieben aufbauen.

Bei einem Schwarz-Weiß-Bildschirm kann PROED mit Shift+F10 die Anzeige zwischen 24 und 49 Zeilen umgeschaltet werden. Bei 49 Zeilen wird ein kleinerer Zeichensatz verwendet, dafür haben Sie mehr Überblick. Den letzten Schalter kennen Sie schon: Control+X oder F10 um die erweiterten Kommandos zu erreichen.

3.9 Datei-Funktionen

Nachdem Sie eine Weile die schon vorgestellten Kommandos ausprobiert haben, wäre es an der Zeit, den Text abzuspeichern oder PROED zu verlassen und morgen weiter zu machen.

Wenn der Text nur zwischengespeichert werden soll, weil Sie Ihrem Atari ST nicht trauen, dann können Sie mit ↪SE das Dokument auf Diskette oder Festplatte schreiben. Dabei werden Sie nochmals nach dem Datei-Namen gefragt. Hatten Sie beim Start von PROED schon einen Namen angegeben, so erscheint er in der Eingabezeile und Sie brauchen nur noch `Return` zu drücken. War ein neuer Text in Bearbeitung, müssen Sie noch den gewünschten Namen zum Abspeichern festlegen. Nach dem ↪SE-Befehl können Sie mit der Gewißheit weiter-editieren, daß Ihr Text auch nach einem Rechnerabsturz nicht ganz verloren ist.

Soll die Bearbeitung eines Textes abgeschlossen werden, geben Sie ↪QS ein. PROED speichert das Dokument ab und fordert Sie anschließend wieder zur Eingabe des Namens einer weiteren Text-Datei auf. Wenn Sie keinen anderen Text mehr bearbeiten wollen, können Sie, wie schon beschrieben, mit `Esc` zum Desktop zurück gelangen.

↪QA ist eine Abart des ↪QS-Kommandos. Damit können Sie die Bearbeitung verlassen, ohne abzuspeichern. Ein Grund dafür wäre, daß Sie einen Text nur mit dem Editor anschauen wollten oder ihn z.B. mit dem ↪RS-Befehl völlig durcheinander gebracht haben.

Einen definierten Block können Sie auch einzeln abspeichern. Dazu dient ↪SB, wobei Sie noch einen Datei-Namen für den Textausschnitt eingeben müssen. Der Block wird nicht aus dem Text entfernt, lediglich die Blockmarkierungen werden gelöscht.

Falls der Datei-Name, den Sie für einen Block verwenden wollen, bereits vergeben ist, fragt PROED bei Ihnen nach: `File Exists: (A)ppend (R)eplace or (E)xit?`. Sie können nun auswählen, ob der Block an die schon bestehende Datei angehängt werden soll (`A`), ob er sie ersetzen soll (`R`) oder ob Sie vielleicht das Kommando abbrechen und einen anderen Datei-Namen probieren wollen (`E`).

Mit dem ↪SB können Sie sich z.B. eine Reihe von Textbausteinen erstellen. Mit dem Kommando ↪IF lesen Sie Dateien von Diskette oder Festplatte in den bearbeiteten Text ein. PROED fügt die neuen Zeilen ab der aktuellen Cursor-Position ein.

Wenn Sie während der Bearbeitung eines Textes einen zweiten ansehen oder verändern wollen, können Sie mit ↪XE eine weiteres Dokument öffnen. PROED fragt nach einem Datei-Namen und Sie können dann den zweiten Text so bearbeiten, wie Sie es gewohnt sind. Allerdings kehren Sie dann mit ↪QS oder ↪QA wieder zum ersten Text und nicht zum Desktop

zurück.

3.10 Drucken

PROED kann Dateien ausdrucken, während Sie an einem Text arbeiten. Der Befehl zum Starten des Druckens lautet ↪PF. PROED fragt Sie nach einem Datei-Namen und beginnt dann, den Text an den Drucker zu schicken, während Sie die Arbeit am geladenen Text fortführen können.

Falls Sie Papier nachlegen müssen oder aus anderen Gründen den Drucker anhalten wollen, geben Sie das ↪HP-Kommando. Mit ↪RP veranlassen Sie PROED die Ausgabe wieder aufzunehmen. Beendet wird der Ausdruck entweder durch das Ende der ausgewählten Datei oder das ↪KP-Kommando, das den Druckvorgang endgültig abbricht.

Wenn Sie einen Teil des gerade bearbeiteten Textes ausgeben wollen, müssen Sie ihn als Block markieren. Mit dem Befehl ↪PB veranlassen Sie PROED den Ausschnitt an den Drucker zu schicken. Dabei ist natürlich keine Bearbeitung möglich, denn Sie könnten ja im Block selber während des Druckens etwas ändern.

3.11 Voreinstellungen

PROED lädt einige Voreinstellungen beim Programmstart aus der Datei `PROED.PF`. Es gibt dabei sechs Einstellungen, die mit dem ↪PO-Befehl verändert werden können. PROED zeigt ein kleines "Fenster" an, in dem die Änderungen vorgenommen werden. Die Optionen sind (mögliche Antworten in eckigen Klammern):

`Generate Automatic Backup File [Yes/No]:`

Falls `Yes` ausgewählt wird, erzeugt PROED beim Abspeichern ein Kopie des ursprünglichen Textes mit der Extension `.BAK`, so daß Sie über diese Datei das Original noch im Zugriff haben.

`Maximum Memory Size to Allocate [nnn]:`

Hier können Sie angeben, wieviel Speicher PROED für einen Text reservieren soll. PROED akzeptiert Eingaben von 10 bis 999 Kilobyte. Normalerweise brauchen Sie an der Vorgabe 999 Kilobyte nichts zu ändern, es sei denn, Sie hätten gleichzeitig andere Programme laufen, die Speicherplatz benötigten.

`Compress Output Files with Tabs [Yes/No]:`

Im ASCII-Zeichensatz, den auch der Atari ST in erweiterter Form benutzt, ist ein Steuerzeichen (TAB) reserviert. Wenn Sie `Yes` auswählen, ersetzt PROED jeweils acht Leerzeichen durch TAB und spart somit etwas Speicherplatz auf Ihrer Diskette. Die Verwendung diese Option ist so lange problemlos, wie Sie die so komprimierten Dateien mit PROED weiterbearbeiten. Es kann aber auch andere Programme geben, die das Steuerzeichen TAB nicht richtig interpretieren können, so daß Sie normalerweise `No` auswählen sollten.

`Turn Cursor Off on Exit [Yes/No]:`

Diese Option betrifft ein GEM-spezifisches Problem, das Sie vielleicht schon beobachtet haben: Wenn ein Programm den Text-Modus einschaltet erscheint ein blinkender Cursor. Falls er nach Verlassen des Programms nicht abgeschaltet wird, blinkt der Cursor im GEM-Desktop unaufhörlich weiter, was recht störend wirkt. Wenn Sie also PROED vom Desktop aus starten wollen, sollten Sie `No` eingeben. Bei Verwendung einer Kommando-Shell (`COMMAND.PRG`) kann der Cursor dagegen eingeschaltet bleiben.

`Color for Foreground Text [nnn]:`

Hiermit können Sie die Farbe einstellen, in der die Textzeichen auf dem Bildschirm erscheinen. Dabei gibt nnn den Wert in oktaler Schreibweise an, mit dem die Farbregister des Atari ST geladen werden. Beim Monochrome-Bildschirm SM124 können Sie nur 000 für schwarze und 777 für weiße Schriftfarbe eingeben. Im Farbmodus gelten diese Werte auch; wenn Sie allerdings andere Farben verwenden wollen, müssen Sie sich mit der internen Farbdarstellung auskennen.

`Color for Background [nnn]:`

Analog zur letzten Option wird hiermit die Farbe für den Hintergrund angegeben. Wollen Sie den Text Schwarz auf Weiß, geben Sie oben 000 und hier 777 an, bei weiß/schwarzer Darstellung genau umgekehrt. Für andere Farbkombinationen müssen Sie sich wiederum mit den Internas des ST auseinandersetzen.

Nachdem Sie alle Optionen festgelegt haben, fragt PROED Sie nach einem Datei-Namen, unter dem dieses "Profile" gespeichert werden soll. Wenn die Änderungen beim nächsten Programmlauf sofort aktiv werden sollen, müssen Sie `PROED.PF` angeben.

↪SP ermöglicht es, die Einstellungen auch nachträglich zu speichern. ↪LP lädt eine Voreinstellung und läßt sie aktiv werden. Sie könnte z.B. verschiedene Profiles für den Farb- und Monochrome-Monitor abspeichern und diese nach Bedarf laden.

Sie können öfters verwendete Zeichenketten als Abkürzung definieren und dann zeitsparend eingeben. Zur Definition einer Abkürzung dient das Kommando ↪DK. In der unteren Bildschirmzeile fragt PROED mit `Enter Key to be Defined: _` nach dem Zeichen, das als Abkürzung Verwendung finden soll. Gültige Abkürzungen sind alle Zeichen außer Steuer- und Sonderzeichen, zu denen auch die Umlaute zählen. PROED unterscheidet bei den Kürzeln zwischen Groß- und Kleinschreibung.

Haben Sie hier z.B. den Buchstaben `a` als Abkürzung gewählt, fragt PROED mit `Enter the Value of <ESC>-a:` nach dem Text, für den die Abkürzung `a` stehen soll. Sie können nun bis zu fünfzig Zeichen eingeben oder einen schon vorhandenen Abkürzungstext verändern. Die Eingabe können Sie mit `Return` bestätigen oder mit `Esc` abbrechen. Danach erscheint wieder die Frage nach einem Kürzel, die Sie wiederum mit `Esc` abbrechen können. Alle definierten Abkürzung speichert PROED beim Kommando ↪SP mit den anderen Voreinstellungen ab.

Bei der Texteditierung können Sie den Abkürzungstext durch Eingabe von `Esc` und dem Kürzel abrufen. Die Abkürzung `a` wird also mit `Esc a` in den Text übernommen. Falls Sie allerdings mit Alternate+C den Modus zur Eingabe von Steuerzeichen und Umlauten eingeschaltet haben, erkennt PROED `Esc` nicht mehr als Einleitung eines Kürzels, sonders fügt die Taste als Steuerzeichen in den Text ein. Bei der Verwendung von Abkürzungen muß dieser Modus als ausgeschaltet bleiben.

Mit dem Kommando ↪SK können Sie sich von PROED eine Liste aller definierten Abkürzungen anzeigen lassen. Mit `Return` gelangen Sie wieder in der Textbearbeitung zurück.

3.12 Übersicht

Zum Nachschlagen sind hier nochmals alle PROED-Kommandos aufgelistet. Erweiterte Kommandos finden Sie mit dem ↪-Zeichen markiert. Sie können sich mit der `Help`-Taste auch von PROED Befehlsübersichten anzeigen lassen, allerdings nur in englischer Sprache.

Cursor-Bewegungen

←	Cursor links
Control+B	Cursor links
→	Cursor rechts
Control+F	Cursor rechts
↑	Cursor nach oben
Control+P	Cursor nach oben
↓	Cursor unten
Control+N	Cursor nach unten
Maus	entsprechend Bewegung
Alternate+B	Wort links
Alternate+F	Wort rechts
Control+S	An Zeilenanfang
Control+E	An Zeilenende
Alternate+P	Seite nach oben
F2	Seite nach oben
Alternate+N	Seite nach unten
F1	Seite nach unten
Alternate+S	An Textanfang
Shift+F2	An Textanfang
Alternate+E	An Textende
Shift+F1	An Textende
↪SL	Zeile anzeigen
Control+Z	Aktuelle Zeile zu erster Bildschirmzeile machen
↪MR	Seitengröße festlegen

Tabulatoren

Control+T	Tabulatorstopp setzen oder löschen
Tab	Nächsten Tabulatorstopp anspringen

Löschen

Delete	Zeichen unter Cursor löschen
Control+D	Zeichen unter Cursor löschen
Backspace	Zeichen links von Cursor löschen
Alternate+D	Wort oder Leerzeichen löschen
Control+K	Zeile ab Cursor löschen
Control+R	Zeile löschen
Undo	Löschung rückgängig machen

Marken

F5	Marke setzen
Alternate+L	An Marke springen
F3	An Marke springen

Zeilenoperationen

Control+U	Zeile unter aktueller einfügen
Return	Zeile unter aktueller einfügen (nur im Compose-Modus)
Control+O	Zeile über aktueller einfügen
Control+Y	Zeile an aktueller Cursor-Position aufteilen
Control+J	Zeile an vorherige anfügen
Control+C	Vorherige Zeile ab Cursor-Position kopieren
Control+A	Zeilen auf aktuelle Cursor-Position einrücken

Blockoperationen

Control+M	Blockanfang und -ende markieren
F4	Blockanfang und -ende markieren
Linker Mausknopf	Blockanfang und -ende markieren
Alternate+M	Blockmarkierungen löschen
Shift+F4	Blockmarkierungen löschen
↪DB	Block löschen
↪MB	Block an aktuelle Cursor-Position verschieben
Rechter Mausknopf	Block an aktuelle Cursor-Position verschieben
↪CB	Block an aktuelle Cursor-Position kopieren
Control+A	Zeilen des Blocks auf aktuelle Cursor-Position einrücken

Suchen und Ersetzen

↪FS	Zeichenkette suchen
Control+L	Weitersuchen
↪RS	Zeichenkette ersetzen
↪CS	Vorkommen einer Zeichenkette zählen

Schalter

Alternate+O	Outline-Modus
Alternate+C	Eingabe von Steuerzeichen und Umlauten
F7	Wortumbruch
F8	Automatisches Einrücken
F9	Compose-Modus
Shift+F10	24/49 Zeilen-Anzeige
Insert	Einfügemodus
Control+I	Einfügemodus
Control+X	Erweiterte Kommandos
F10	Erweiterte Kommandos

Datei-Funktionen

↪SE	Text speichern und weiterbearbeiten
↪QS	Text speichern und verlassen
↪QA	Verlassen ohne zu speichern
↪IF	Datei ab aktueller Cursor-Position einfügen
↪XE	Zweiten Text bearbeiten
↪SB	Block auf Massenspeicher schreiben

Drucken

↪PF	Text ausdrucken
↪HP	Drucken anhalten
↪RP	Drucken weiterführen
↪KP	Drucken abbrechen
↪PB	Block ausdrucken

Voreinstellungen

↪PO	Voreinstellungen setzen
↪SP	Voreinstellungen speichern
↪LP	Voreinstellungen laden
↪DK	Abkürzungen definieren
↪SK	Abkürzungen anzeigen
Esc	Abkürzung einleiten
Help	Hilfe anzeigen
↪Help	Hilfe für erweiterte Kommandos anzeigen

Chapter 4

Der Textformatierer PROFF

Das "Salz in der Suppe" der Textverarbeitung ist neben der einfachen Korrekturmöglichkeit die Flexibilität bei der Formatierung. Rechter Randausgleich, Spaltenformatierung, automatische Erzeugung von Inhaltsverzeichnissen sind nur einige der vielen Möglichkeiten. Textverarbeitung kann man in zwei Aufgaben aufteilen: die Texterfassung und -editierung auf dem Rechner und die Formatierung für den Drucker. PROED ist nur zur Texterfassung gedacht, ihm fehlt ein Formatierungsprogramm. Hier bildet das Public-Domain-Programm PROFF – geschrieben von Ozan S. Yigit und Steven Tress – eine sinnvolle Ergänzung für den, der nicht nur Programme, sondern auch Fließtexte schreibt. Das Programm basiert auf NRoff, einem Formatierer unter dem Betriebssystem UNIX, der dort zum Standardumfang der Utilities gehört.

PROFF erhält als Eingabe einen unformatierten Text mit eingestreuten Kommandos zum Formatsteuerung. Die Ausgabe ist der fertig formatierte Text, der direkt an den Drucker oder an eine Datei geschickt werden kann. Besondere Merkmale sind die Fähigkeit zum Erzeugen von Inhaltsverzeichnissen und die Programmierbarkeit durch Makros.

Einen Eingabetext für PROFF können Sie mit jedem beliebigen Texteditor erfassen, der ASCII-Files erzeugt, also z.B. mit PROED, aber nicht mit 1stWord im WP-Mode. Das Aussehen des Eingabetextes brauchen Sie nicht gestalten, da er vollständig neu formatiert wird. Auch der Zeilenumbruch wird natürlich von PROFF übernommen, so daß Sie jederzeit eine neue Zeile beginnen können, auch wenn ein Absatz noch nicht zuende ist. Das Absatzende müssen Sie durch eine Leerzeile kennzeichnen.

Jedes Kommando an PROFF beginnt auf einer neuen Zeile mit einem Punkt am Anfang. Darauf folgt der Name des Kommandos (meist gibt es auch mehrere Abkürzungen für ein Kommando) und eventuel Parameter. Da der Zeilenumbruch keine Rolle spielt, macht es nichts aus, daß Sie eine neue Zeile für ein Kommando beginnen müssen.

Das Aussehen der Ausgabe wird mit diesen Kommandos festgelegt. Es gibt Befehle zur Definition des Seitenformats, zur Gestaltung von Kopf- und Fußzeilen, aber auch zur Generierung eines Inhaltsverzeichnisses. Fortgeschrittene können über Makros sogar eigene Kommandos programmieren.

Neben normalen Textzeilen und Kommandozeilen können im Eingabetext auch noch Kommentare stehen, die von PROFF überlesen werden und auch nicht in der Ausgabe erscheinen. Sie beginnen am Zeilenanfang mit den Zeichen `.!` oder `.#`.

Bei der folgenden Beschreibung der Kommandos werden optionale Parameter, also solche, die auch weggelassen werden können, in eckigen Klammern dargestellt. Falls mehrere Alternativen für die Parameter möglich sind, werden sie durch | getrennt. Falls ein Kommando mehrere Namen hat, gelten die Parameterangaben für alle aufgeführten Namen. Benötigt ein Kommando einen Wert als Parameter, so PROFF erwartet immer ein ganze Zahl.

4.1 Seitenlayout

`.offset` [+|-][*n*]
`.po`

Definiert einen linken Rand von *n*, um den der gesamte Text eingerückt wird. Wird *n* mit einem Vorzeichen angegeben, so verrechnet PROFF diesen Wert mit dem gerade geltenden linken Rand. Fehlt *n*, nimmt PROFF 0 an.

`.rightmargin` [+|-][*n*]
`.rm`

Legt den rechten Rand auf *n* Zeichen. Bei Angabe eines Vorzeichens wird *n* mit dem aktuellen Wert verrechnet. Fehlt *n*, so gilt die Voreinstellung 65 Zeichen.

`.pagesize` [*n*]
`.ps`
`.pl`

Stellt die Gesamtzahl der Zeilen auf einer Seite auf *n* Zeilen ein. Bei fehlendem *n* werden 66 Zeilen pro Seite angenommen.

Zur Gestaltung einer Seite verwendet PROFF vier Ränder, mit denen der Raum zwischen und nach Kopf- und Fußzeilen festgelegt wird.

`.m1` [*n*]

Setzt Rand 1 auf *n* Zeilen. Dieser Rand gibt an, wieviele Zeilen vom Blattanfang bis einschließlich einer Kopfzeile gedruckt werden. Die Voreinstellung für das Kommando ohne *n* ist drei Zeilen.

`.m2` [*n*]

Hiermit wird der Rand 2 auf *n* Zeilen gesetzt. Rand 2 ist die Anzahl der Leerzeilen, die zwischen einer Kopfzeile und dem Beginn des eigentlichen Textes stehen. Ohne *n* werden zwei Zeilen eingestellt.

`.m3` [*n*]

Das Kommando legt den Rand 3 auf *n* Zeilen fest. Er bestimmt die Anzahl der Leerzeilen, die zwischen der letzten Textzeile und einer Fußzeile stehen. Wird *n* weggelassen benutzt PROFF den Wert zwei.

`.m4` [*n*]

Hiermit wird festgelegt, wieviele Zeilen PROFF einschließlich der Fußzeile bis zum unteren Ende des Papiers druckt. Voreinstellung sind drei Zeilen.

Die Zahl der tatsächlichen Textzeilen errechnet sich aus pagesize − m1 − m2 − m3 − m4.

4.2 Seitenformatierung

`.spacing` [*n*]
`.spc`
`.ls`

Legt den Zeilenabstand fest. Für einen doppelzeiligen Ausdruck (jeweils eine Leerzeile im Text) müssen Sie `.spacing 2` angeben. Die Voreinstellung ist der Wert eins.

`.page` [*n*]
`.pg`
`.bp`

Hiermit wird eine neue Seite begonnen. PROFF gibt die Fußzeile aus, macht einen Seitenvorschub und druckt dann auf einer neuen Seite weiter. Wird als Parameter *n* angegeben, so setzt das Kommando die aktuelle Seitenzahl auf *n*.

`.pagenumber roman|arabic`
`.pn`

PROFF kann Seitenzahlen normal (`arabic`) oder mit Römischen Ziffern (`roman`) als I, II, III, IV usw. ausgeben. Das Kommando weist PROFF entsprechend an. Die Worte `roman` oder `arabic` müssen durch ein Leerzeichen getrennt hinter dem Kommando stehen.

`.nopaging`
`.np`

Nach diesem Kommando ignoriert alle `.page` und `.pagesize` Kommandos, was für Testausdrucke nützlich ist. Für Korrekturen benötigen Sie ja keinen exakten Seitenumbruch, der nur zusätzlich Zeit kostet.

`.paging`
`.pa`

Schaltet die Beachtung von `.page` und `.pagesize` wieder ein.

`.testpage` [*n*]
`.tp`
`.need`
`.ne`

Überprüft, ob auf der gerade bearbeiteten Seite noch *n* Zeilen frei sind. Falls das nicht zutrifft, wird eine neue Seite angefangen. Fehlt *n*, nimmt PROFF den Wert 0 an womit das Kommando dann wie `.page` arbeitet.

`.header` *text*
`.he`

Definiert eine Kopfzeile. Für *text* müssen Sie die eigentliche Kopfzeile einsetzen. Sie besteht aus drei Teilen, die links, in der Mitte und rechts im Kopf erscheinen. *text* beginnt mit einem Sonderzeichen, daß dann als Trennzeichen zwischen den drei Teilen verwendet wird und am Ende von *text* normals stehen muß. Eine mögliche Kopfzeile wäre:

`.header |links|Mitte|rechts|`

In der Ausgabe erschiene dann:

```
links           Mitte         rechts

Hier steht der eigentliche Text, der
aus der Eingabe stammt.
```

Als Trennzeichen können alle Zeichen verwendet werden, die weder Ziffern noch Buchstaben sind. Die Zeichen # und % haben eine Sonderbedeutung und sind nicht als Trennzeichen zugelassen. Kommt in *text* das Zeichen # vor, wird es durch die Seitennummer ersetzt. Beim Zeichen % setzt PROFF Uhrzeit und Datum des Ausdrucks ein.

`.footer` *text*
`.fo`

Legt eine Fußzeile fest. *text* ist wie bei `.header` aufgebaut. Das Beispiel

```
.footer ||Seite #||
```

würde die Ausgabe

```
Die  letzte  eigentliche  Textzeile.
               Seite 1
```

ergeben.

`.oh` *text*

Definiert eine Kopfzeile, die nur auf Seiten mit ungerader Seitennummer erscheint (odd header).

`.eh` *text*

Legt die Kopfzeile für Seiten mit gerader Seitennummer fest (even header). Die Unterscheidung von Seiten mit gerade und ungeraden Nummern ist bei der Ausgabe von Dokumenten interessant, die zweiseitig gelesen werden sollen. Schauen Sie sich z.B. die Kopfzeilen in diesem Buch an. Um die Seitennummer immer außen zu setzen erscheint sie auf ungeraden Seiten rechts, auf geraden links.

`.of` *text*

Definiert die Fußzeile für ungerade Zeilen ein (odd footer).

`.ef` *text*

Definiert die Fußzeile für Seiten mit gerader Seitennummer ein (even footer).

4.3 Zeilenformatierung

```
.fill
.fi
.f
```

Schaltet die Zeilenfüllung ein. PROFF versucht dann, soviele Worte wie möglich in eine Zeile zu packen. Dieser Modus ist beim Start von PROFF voreingestellt. Der Zeilenumbruch im Eingabetext ist somit egal, da er für die Formatierung überlesen wird.

`.nofill`
`.nf`

Schaltet die Zeilenfüllung aus. Der Umbruch im Eingabetext wird auch für die Ausgabe verwendet. Dieser Modus ist günstig, wenn Sie z.B. Tabellen oder Listings in einem Text verwenden, da sie sicherlich nicht umformatiert werden sollen.

`.justify`
`.ju`
`.j`

Hiermit wird der rechte Randausgleich eingeschaltet. Alle Zeilen werden so formatiert, daß sie durch Leerstellen zwischen den Worten bis zum rechten Rand reichen (Blocksatz). Beim Randausgleich muß gleichzeitig die Zeilenfüllung eingeschaltet sein. Randausgleich ist beim Start von PROFF voreingestellt.

`.nojustify`
`.nf`

Mit diesem Kommando wird der Randausgleich ausgeschaltet. Dadurch wird Flattersatz erzeugtm bei dem die Zeilen unterschiedliche Länge haben können.

4.4 Absatzformatierung

`.break`
`.br`

Beginnt einen neuen Absatz: Für die laufende Zeile wird kein Randausgleich mehr durchgeführt und der nachfolgende Text beginnt auf einer neuen Zeile.

`.leftmargin` [+|-][*n*]
`.lm`
`.in`

Führt ein `.break` aus und rückt die nachfolgenden Zeilen um n Zeichen ein. Bei einem Vorzeichen wird *n* mit der aktuellen Einrückung verrechnet. Eine sinnvolle Anwendung des Befehl ist ein längeres Zitat oder eine Aufzählung mit Ober- und Unterbegriffen.

`.left` [+|-][*n*]
`.ti`
`.i`

Ist identisch zu `.leftmargin`, nur bezieht sich die Einrückung nur auf die nächste Zeile. So können Sie einen zusätzlichen Einzug am Anfang eines Absatzes erreichen.

`.spaceto` [-][*n*]
`.st`

Formatiert die nächsten Zeilen so, daß sie ab Zeile *n* auf der aktuellen Seite ausgedruckt werden. Falls ein negatives Vorzeichen angegeben wird, druckt PROFF ab der Zeile *n*, vom Ende der Seite an gezählt.

`.skip` [*n*]
`.sp`
`.s`

Führt ein `.break` aus und überspringt *n* Zeilen. *n* wird dabei mit dem Wert multipliziert, der für das `.spacing` Kommando gilt.

`.center` [*n*|`on`|`off`]
`.ce`

Zentriert die folgenden *n* Zeilen. `.center on` schaltet die Zentrierung für alle folgenden Zeilen ein; `.center off` schaltet sie wieder aus.

`.underline` [*n*|`on`|`off`|`all`|`words`]
`.ul`

Schaltet die Unterstreichung für die nächsten *n* Zeilen aus dem Eingabetext ein. Mit dem Parametern `on` und `off` läßt sich die Unterstreichung für einen Textblock setzen.

Die Parameteroptionen `all` und `word` definieren, ob PROFF alle Zeichen unterstreichen soll, oder nur ganze Wörter. Bei der Option `words` werden Leerzeichen nicht unterstrichen.

`all`: Ein Satz mit Leerzeichen
`words`: Ein Satz mit Leerzeichen

`.bold` [*n*|`on`|`off`]
`.bd`

Schaltet Fettschrift für die nächsten *n* Zeilen oder für einen Textblock ein und aus.

`.disablebolding`

`.dbo`
`.db`

Nach diesem Kommando ignoriert PROFF den `.bold` Befehl. Da die Fettschrift durch doppeltes Ausdrucken eines Zeichens erreicht wird, kann dadurch der Ausdruck schneller werden, was vor allem bei Probeausdrucken zu empfehlen ist.

`.enablebolding`
`.ebo`
`.eb`

Ab diesem Befehl wird das `.bold` Kommando wieder wie gewohnt verarbeitet.

4.5 Automatisches Inhaltsverzeichnis

PROFF kann automatisch ein Inhaltsverzeichnis Ihres Textes erzeugen, also den Kapitelüberschriften die Seitenzahlen zuordnen. Dazu sind Befehle zum Übernehmen einer Zeile in das Verzeichnis und zum Ausgeben des Inhalts notwendig.

`.contline` [*n text*]
`.cl`

Hiermit wird eine Zeile als Kapitelüberschrift markiert. Sie erscheint sowohl im Ausdruck als auch im Inhaltsverzeichnis, dort mit Seitenangabe. *n* gibt dabei die Gliederungsstufe an und ist eine Zahl (von Null an gezählt). Sie wirkt sich beim Ausdruck des Inhaltsverzeichnisses als Einrückung aus, bewirkt aber keine Kapitelnummerierung. *text* ist die eigentliche Überschrift. Wenn beide Parameter fehlen, wird einfach eine Leerzeile in das Inhaltsverzeichnis übernommen, was der Übersichtlichkeit dient.

`.printcont` [*n*]
`.pc`

Wenn PROFF auf diesen Befehl trifft, gibt es das Inhaltsverzeichnis aus, das es aus den bisherigen `.contline` Kommandos zusammengestellt hat. *n* definiert, wie groß die Einrückung für jede Gliederungsstufe sein soll. Fehlt *n*, wird drei als Voreinstellung angenommen.

Dazu ein kleines Beispiel. Der Text, für den das Inhaltsverzeichnis erstellt werden soll, sieht folgendermaßen aus:

```
.contline 0 Hauptkapitel 1
Dieses ist das erste Hauptkapitel
```

```
.contline 1 Unterkapitel 1
Ein erstes Unterkapitel ...

.contline 1 Unterkapitel 2
auf das ein zweites folgt.

.contline 2 Unter-Unterkapitel 1
Es hat ein Unterkapitel

.! Leerzeile ins Inhaltsverzeichnis
.cl

.contline 0 Zweites Hauptkapitel
Zu guter letzt noch ein Hauptkapitel

.! Inhaltsverzeichnis ausdrucken
.skip 4
.center 1
Inhaltsverzeichnis

.pc
.! Ende
```

Das Inhaltsverzeichnis erscheint am Ende der Ausgabe so:

```
                    Inhaltsverzeichnis

Hauptkapitel 1.....................................    0
   Unterkapitel 1..................................    1
   Unterkapitel 2..................................    1
      Unter-Unterkapitel 1.........................    1

Zweites Hauptkapitel...............................    1
```

Ich werde Ihnen in Kapitel 4.7 auf Seite 46 selbstdefinierte Kommandos vorstellen, mit denen auch die Kapitelnummerierung automatisch von PROFF erzeugt wird.

4.6 Variable

PROFF ist in der Lage, einen Satz an Variablen zu halten, die entweder Text oder Zahlenwerte enthalten können. Sie können damit Ihre Ausgabe

variabel halten oder die Makroprogrammierung ausnutzen (s.u.).

`.set` *var* [*definition*]
`.vs`

Hiermit weisen Sie der Textvariablen *var* einen Wert zu. *var* ist der Name der Variablen, muß mit einem Buchstaben anfangen und darf nur alphanumerische Zeichen enthalten. Wenn Sie in dem Namen ein Leerzeichen verwenden, muß *var* in Anführungsstrichen stehen. Wenn Sie dann ein Anführungstrich im Namen verwenden wollen, wird dieser doppelt geschrieben. Erlaubte Namen sind also:

```
Robert
"Atari ST"
"eine ""PROFF"" Variable"
```

aber nicht:

```
1001
Atari ST
"Ein "-Zeichen"
```

definition ist der Text, den Sie der Variable zuweisen. Falls er fehlt, zeigt PROFF den Namen der Variablen auf dem Bildschirm an und wartet auf eine Eingabe des Textes durch den Benutzer.

Eine Variable können Sie nach ihrer Definition im Text verwenden. PROFF erkennt eine Variable im Text durch das Zeichen `$` und ihrem Namen. Der Formatierer setzt dann ihren Inhalt ein, so als würde er direkt im Eingabetext stehen. Falls Sie das `$`-Zeichen in Ihrem Text brauchen, setzen Sie das Fluchtsymbol "_" davor (durch `.echar` umdefinierbar).

Damit PROFF das Ende des Variablennamens erkennen kann, muß er durch eine Leerstelle beendet werden. Falls der folgende Text aber in der Ausgabe direkt auf den Variableninhalt folgen soll, können Sie den Namen der Variablen in geschweifte Klammern setzen ({}). PROFF erkennt dann an der schließenden Klammer das Ende des Namens und kann so weiteren Text direkt an den Inhalt der Variablen anhängen.
Dazu ein paar Beispielzeilen:

```
.set version 1
.set revision "2. Überarbeitung"
Dokument PROFF, ${version}. Version -- $revision
```

In die Ausgabe schreibt PROFF dann:

```
Dokument PROFF, 1. Version -- 2. Überarbeitung
```

`.get` *var prompt*
`.vg`

Wenn beim `.set` Befehl die Definition fehlt, fragt PROFF unter Angabe des Variablennamens den Benutzer nach einer Eingabe. Beim `.get` Kommando erscheint auf dem Bildschirm zusätzlich der Text, der in *prompt* steht. Das ermöglicht eine interaktive Eingabe des Inhalts von var möglich, bei der dem Benutzer eine Frage gestellt werden kann.

`.nr a-z` [+|-][*n*]

Neben den Textvariablen hat PROFF auch numerische Variablen, die Register. Sie werden mit den Buchstaben `a` bis `z` bezeichnet und können mit dem `.nr` Befehl gesetzt und verrechnet werden. Wird kein Operator angegeben, setzt PROFF das bezeichnete Register auf den Wert *n*.

Zum Rechnen dienen `+` und `-`. Bei ersterem wird der Wert des Registers um *n* erhöht; beim Minuszeichen um *n* erniedrigt. Für *n* sind immer ganze Zahlen zugelassen.

Der Inhalt eines Registers kann an beliebiger Stelle eingesetzt werden. Dazu wird er durch `@n`*z* gekennzeichnet, wobei *z* ein Buchstabe von a bis z ist. Wenn Sie das @-Zeichen im Ausgabetext brauchen, setzen Sie das Fluchtsymbol, den Unterstrich _ davor (`_@`).

Nehmen wir an, Sie schreiben einen Text, in dem mehrere Argumente mit 1. 2. usw. durchgezählt werden sollen. Die Nummerierung soll PROFF übernehmen. Unter Verwendung des Registers n sähe die Eingabe dann so aus:

```
.nr m 1
@nm. PROFF ist ein gutes Formatierprogramm

.nr m +1
@nm. PROFF ist Public-Domain

.nr m +1
@nm. Auch die Source von PROFF ist erhältlich
```

Die Ausgabe erscheint als:

```
1. PROFF ist ein gutes Formatierprogramm

2. PROFF ist Public-Domain

3. Auch die Source von PROFF ist erhältlich
```

Der Vorteil gegenüber dem direkten Einfügen der Zahlen ist die einfache Änderung der Aufzählung. Wenn Sie ein weiteres Argument einfügen wollen, brauchen Sie nicht die Ziffern in Ihrem Text zu ändern, da die Zählung automatisch geschieht.

4.7 Makros

Richtig programmierbar wird PROFF durch die Makros. Mit ihnen können Sie eigene Kommandos definieren, die dann wie die schon eingebauten Funktionen benutzt werden können. Es ist möglich, sehr komplexe Makros unter Benutzung der Variablen zu schreiben, mit denen sich auch neue Formatier-Funktionen, wie Schlagworterstellung oder volle Ausnutzung der Druckerfunktionen programmieren lassen.
Eine Makrodefinition beginnt mit

`.define` *Makroname*
`.de`

und wird mit

`.en`

beendet. Alles was zwischen diesen beiden PROFF Befehlen steht, wird später bei einem Makroaufruf mit *.Makroname* in den Text an der Stelle eingefügt, an der der Aufruf steht. Ein Makro kann aus normalem Text, PROFF-Kommandos oder erneuten Makroaufrufen bestehen. Eine einfache Anwendung von Makros ist die Definition von Kürzeln:

```
.define Adr
Hans Müller
Himmelsweg 3
9344 Nirgendwo
.en
```

Beim Aufruf durch

```
Meine Adresse:
.Adr
```

gibt PROFF die vollständige Adresse aus – eine Möglichkeit, die Variablen nicht bieten.

Jedem Makro können bis zu neun Parameter mitgegeben werden. Dabei wird jedes Wort, das nach dem Makroname steht, als Parameter aufgefaßt. Parameter können auch durch Kommata getrennt angegeben werden. In

der Makrodefinition können Sie diese durch das Zeichen `$` und einer Ziffer von 1 bis 9 ansprechen. PROFF setzt an diese Stellen die Parameterwerte. Folgen beim Makro-Aufruf keine Parameter, sein `$1` bis `$9` leer.

Dazu ein Beispiel, das mit einem Schlag die Zentrierung und Unterstreichung ein- oder ausgeschaltet:

```
.define ceul
.center $1
.underline $1
.en
```

Als Parameter beim Aufruf kommen dann die Worte `on` und `off` sowie eine Zahl in Frage. Er wird dann bei den Kommandoaufrufen eingesetzt. PROFF verarbeitet bei einem Aufruf `.cuel on` die Zeilen

```
.center on
.underline on
```

Ein Aufruf ohne Parameter (`.cuel`) wird bei der Verarbeitung von `.center` und `.underline` einen Fehler hervorrufen, da `$1` leer ist.
Durch Makros wird auch eine elegantere Ansteuerung von Sonderfunktionen Ihres Druckers möglich. Sie müssen keine `.write`-Kommandos mehr im Text benutzen, sondern definieren nur einmal Makros wie die folgenden:

```
.define elite
.write ^["M"
.en
.define pica
.write ^["P"
.en
```

Nun reicht ein Aufruf von `.elite`, um in diesem Fall einen EPSON-Drucker auf Elite-Schrift umzustellen und Sie müssen sich keine Code-Sequenzen mehr merken. Wenn Sie später einen anderen Drucker benutzen, der andere Steuerzeichen erkennt, müssen Sie nur noch die entsprechenden Makros zu ändern. Der Vorteil dabei ist, daß Sie den eigentlichen Text so belassen können, wie er ist.

Besonders einfach wird die Benutzung von Makros, wenn Sie sich eine Bibliothek, also eine Sammlung von Makros anlegen, die Sie öfters benutzen. Dann brauchen Sie nur noch diese Bibliothek, in der sonst kein weiterer Text steht am Anfang Ihres Textes mit `.include` einzulesen. Sie können auch beim Aufruf von PROFF einen Bibliothek angeben, die automatisch eingelesen wird.

Es ist auch möglich, die eingebauten Funktionen umzudefinieren. Dabei können Sie praktisch alles verändern, was allerdings nicht immer sehr sinnvoll ist. Jedoch sollten Sie nie ein Makro mit dem Namen `en` definieren, da PROFF dann logischerweise nicht mehr das Ende eines Makros erkennen kann, und somit nicht terminiert.

Abschließend noch die versprochenen Makros zur wirklich automatischen Erzeugung eines Inhaltsverzeichnisses. An ihm können Sie den Umgang mit Parametern und Variablen üben. Beim `.contline` Kommando wird lediglich eine Zeile in eine Tabelle übernommen und ihr eine Seitennummer zugewiesen, die dann im Inhaltsverzeichnis erscheint.

Die folgenden Makros erzeugen auch automatisch die Kapitelnummer. Es sind dabei drei Gliederungsstufen möglich, wobei für jede ein Makros definiert wird. Es gibt Hauptkapitel, Unterkapitel und Abschnitte, die jeweils mit einer Zahl gegliedert werden. Kapitel 3.2.1 wäre also der erste Abschnitt im zweiten Unterkapitel des dritten Hauptkapitels. Was müssen nun die Makros leisten?

Jedesmal, wenn ein neues Kapitel einer Gliederungsstufe beginnt, muß die Kapitelnummer weitergezählt werden und alle Zählungen auf feineren Gliederungsstufen zurückgesetzt werden. Die Makros müssen die Kapitelnummer erzeugen, die Zeile ins Inhaltsverzeichnis übernehmen und schließlich ausgeben.

Bei drei Gliederungsstufen benötigen die Makros also drei Zähler, wofür die Register verwendet werden, hier `r`, `s` und `t`. Bei einem neuen Hauptkapitel wird der Zähler `r`, der die Nummer der ersten Gliederungsstufe enthält um eins erhöht und die Zähler `s` und `t` auf Null gesetzt.

`@nr.` erzeugt die Nummerierung des Hauptkapitels. Diese wird zusammen mit der eigentlichen Überschrift ins Inhaltsverzeichnis per `.contline` auf der obersten Ebene übernommen. Da jedes Wort als ein Parameter gilt, müssen die Parameter 1 bis 9 angegeben werden. Dies hat allerdings den Nachteil, daß die Kapitelüberschriften nicht aus mehr als neun Worten bestehen dürfen.

Schließlich fügt das Makro `.section` mit `.skip 3` noch drei Leerzeilen nach der Überschrift in die Ausgabe ein. Bei den weiteren Gliederungsstufen (Makros `.subsection` und `.subsubsection`) werden lediglich weniger Zähler zurückgesetzt und die Konstruktion der Kapitelnummer ist etwas länger.

Da z.B. `subsubsection` etwas unhandlich ist, werden noch die Makros `.sc`, `.ssc` und `.sssc` definiert, die lediglich Abkürzungen sind. Sie reichen einfach die Parameter an die anderen weiter. Die Makrodefinitionen sehen wie folgt aus:

```
.! Makros für automatische Kapitelnummerierung und Inhaltverzeichnis-
.! erzeugung mit drei Gliederungsstufen.
```

```
.! Benutzt die Register r,s und t
.!
.! Hauptkapitelüberschrift, benutzt Register r,s und t
.define section
.! Hauptkapitelzähler erhöhen und die anderen auf 0 setzen
.nr r +1
.nr s 0
.nr t 0
.! ins Inhaltsverzeichnis übernehmen
.contline 0 @nr. $1 $2 $3 $4 $5 $6 $7 $8 $9
.break
.! ausgeben
@nr. $1 $2 $3 $4 $5 $6 $7 $8 $9
.skip 3
.en
.!
.! Unterkapitelüberschrift
.define subsection
.nr s +1
.nr t 0
.contline 1 @nr.@ns $1 $2 $3 $4 $5 $6 $7 $8 $9
.break
@nr.@ns $1 $2 $3 $4 $5 $6 $7 $8 $9
.skip 2
.en
.!
.! Abschnittüberschrift
.define subsubsection
.nr t +1
.contline 2 @nr.@ns.@nt $1 $2 $3 $4 $5 $6 $7 $8 $9
.break
@nr.@ns.@nt $1 $2 $3 $4 $5 $6 $7 $8 $9
.skip
.en
.!
.! Abkürzung für .section
.define sc
.section $1,$2,$3,$4,$5,$6,$7,$8,$9
.en
.!
.! Abkürzung für .subsection
.define ssc
```

```
.subsection $1,$2,$3,$4,$5,$6,$7,$8,$9
.en
.!
.! Abkürzung für .subsubsection
.define sssc
.subsubsection $1,$2,$3,$4,$5,$6,$7,$8,$9
.en
```

Der nun folgende Text ist eine Spielbeispiel für die Anwendung der Makros:

```
.! Der eigentliche Text
.section Hauptkapitel 1
Dieses ist das erste Oberkapitel

.subsection Unterkapitel 1
Ein erstes Unterkapitel ...
.ssc Unterkapitel 2
auf das ein zweites folgt.

.subsubsection Abschnitt 1
Es hat ein unterkapitel

.ssc Unterkapitel 3
Das dritte Unterkapitel

.sc Zweites Oberkapitel
Zu guterletzt noch ein Oberkapitel

.sssc Ein kleiner Abschnitt
mit einem "nullten" Abschnitt

.! Inhaltsverzeichnis ausdrucken
.skip 4
.center 1
Inhaltsverzeichnis

.pc
.! Ende
```

Er ergibt die folgende Ausgabe mit Inhaltsverzeichnis:

```
1. Hauptkapitel 1
```

```
Dieses ist das erste Oberkapitel

1.1 Unterkapitel 1

Ein erstes Unterkapitel ...

1.2 Unterkapitel 2

auf das ein zweites folgt.

1.2.1 Abschnitt 1

Es hat ein Unterkapitel

1.3 Unterkapitel 3
Das dritte Unterkapitel

2. Zweites Oberkapitel

Zu guterletzt noch ein Oberkapitel

2.0.1 Ein kleiner Abschnitt

mit einem "nullten" Abschnitt


                    Inhaltsverzeichnis

1. Hauptkapitel 1       ..................................... 1
   1.1 Unterkapitel 1       ................................ 1
   1.2 Unterkapitel 2       ................................ 1
      1.2.1 Abschnitt 1       .............................. 1
   1.3 Unterkapitel 3       ................................ 1
2. Zweites Oberkapitel       ............................... 1
      2.0.1 Ein kleiner Abschnitt      ..................... 1
```

Die Makros könnten Sie natürlich noch um weitere Kommandos erweitern. So könnten Ihre Hauptkapitel immer in Fettschrift und unterstrichen erscheinen, die Unterkapitel unterstrichen und die Abschnitte fett. Dies alles ist durch wenige zusätzliche Befehle möglich. Probieren Sie es zur Übung einmal aus!

Damit PROFF schnell und speicherplatzsparend arbeiten kann (selbst beim Atari ST ist der Speicher begrenzt), sollten Sie einige Punkte beachten. PROFF muß sich alle definierten Makros merken, weshalb Sie keine überflüssigen Makros definieren sollten. Vermeiden Sie auch lange Kommentare in den Definitionen, da sie bei der Speicherung nicht entfernt werden. Außerdem verlangsamen sie die Ausgabe. Die verwendeten Kommandos sollten möglichst in ihren Kurzfassungen benutzt werden. Anfangs sind die Langformen vielleicht eingängiger – bei einiger Übung erlernen Sie aber auch die Abkürzungen leicht.

4.8 Restliche Kommandos

`.cchar` [*char*]
`.cc`

PROFF benutzt zur Kennzeichnung von Kommandos das Punkt-Zeichen. Mit diesem Befehl können Sie dieses Fluchtsymbol zu *char* umdefinieren, falls Sie unbedingt einen Punkt am Anfang einer Zeile benötigen. Falls der Parameter fehlt, wird wieder der Punkt benutzt. Wenn Sie diesen Text mit PROFF formatieren wollten, müßten Sie `.cchar` verwenden, um die vielen Zeilen mit PROFF-Kommandos ausgeben zu können[1].

`.echar` [*char*]
`.ec`

Zur Markierung von Variablen und Registern werden spezielle Symbole verwendet, die allerdings durch Voranstellen von "_" auch ausgedruckt werden können. Nach einem `.echar` Kommando wird anstelle des Unterstrichs das angegebene Zeichen als Fluchtsymbol verwendet. Bei fehlendem Parameter gilt wieder das "_"-Zeichen.

`.lex` *kommando* [*neu*]
`.lx`

Hiermit können auch die Kommandonamen selber umdefiniert werden. PROFF führt eine Liste aller erlaubten Befehle; in dieser wird der ursprüngliche Name *kommando* entfernt und durch *neu* ersetzt. Dabei werden Makros nicht verändert und werden ungültig, falls sie den geänderten Kommandonamen enthalten. Seien Sie also vorsichtig, wenn Kommandos umdefiniert werden sollen, die Sie in Makros verwenden. Falls *neu* fehlt, "vergißt" PROFF das Kommando vollständig.

[1] Dieses Buch wurde allerdings mit TEX formatiert, einem System, das ähnlich, nur erheblich leistungsfähiger ist.

`.save`
`.sv`

Hiermit werden die momentanen Einstellungen, wie die Ränder intern auf einem Stapelspeicher[2] gesichert. PROFF speichert die Werte, die durch die folgenden Kommandos beeinflußt werden:

PROFF-Kommandos		Beeinflußter Wert
`.leftmargin`		Einrückung
`.rightmargin`		Rechter Rand
`.offset`		Linker Rand
`.spacing`		Zeilensprung
`.pagesize`		Zeilen pro Seite
`.m1`		Rand m1
`.m2`		Rand m2
`.m3`		Rand m3
`.m4`		Rand m4
`.cchar`		Fluchtsymbol für Kommandos
`.echar`		Fluchtsymbol für Sonderzeichen
`.pagenumber`		Art der Seitennummern
`.fill`	`.nofill`	Zeilenfüllung
`.justify`	`.nojustify`	Zeilenausgleich
`.paging`	`.nopaging`	Erkennung von `.page`
`.enablebolding`	`.disablebolding`	Erkennung von `.bold`
`.autoparagraph`	`.noautoparagraph`	Autoparagraph Einstellung

Diese Werte können mit

`.restore`
`.rs`

von diesem internen Stapel geholt werden. Die beiden Kommandos (die Sie immer paarweise verwenden sollten) sind bei wechselnden Seitenformaten praktisch. Stellen Sie sich vor, Sie geben Ihren Text in einem bestimmten Seitenformat aus. Nun sollen drei Seiten mit einem anderen Layout gedruckt werden. Ohne die beiden Kommandos müßten Sie alle Werte nach diesen Seiten wieder mit mehreren Befehlen auf ihren ursprünglichen Inhalt setzen. So brauchen Sie nur vor den drei Sonderseiten `.save` einzugeben, danach `.restore` und der Text läuft im alten Layout weiter.

`.require` [*filename*]

[2] Einen Stapelspeicher können Sie sich wie einen Kartenstapel vorstellen. Daten werden immer obenauf gelegt. Beim Zurückholen wird die oberste Karte abgehoben, so daß das zuletzt gespeicherte Datum als erstes gelesen wird. Bei Computern spricht man von einem "Stack".

`.include`
`.source`
`.so`

Hiermit können Sie mehrere Texte, die in verschiedenen Dateien stehen zu einem Ausdruck zusammenführen. PROFF führt die Ausgabe mit der Datei *filename* weiter und fährt dann im ursprünglichen Text fort. Der eingefügte Text kann ebenfalls `.require` Kommandos enthalten – die Tiefe solcher Schachtelungen ist in der vorliegenden Version auf acht Dateien begrenzt. Neben der Verkettung von Texten kann der Befehl auch zum Einlesen einer Makro-Sammlung verwendet werden. Bei der Angabe des Datei-Namens müssen Sie auf die Ordnerstruktur von GEMDOS achten! Der Datei-Name kann in Anführungsstrichen stehen, muß aber nicht.

`.noautoparagraph`
`.nap`
`.na`

PROFF erkennt einen neuen Absatz daran, daß die Eingabezeile mit einem Leerzeichen oder einem Tab beginnt. Das Programm beginnt dann eine neue Zeile (entsprechend dem Kommando `.skip`) und rückt diese um fünf Leerstellen ein. `.noautoparagraph` schaltet diese Erkennung ab, was nützlich ist, wenn Sie Listings in Ihrem Text verwenden, die natürlich nicht umformatiert werden sollten.

`.autoparagraph`
`.ap`

Die automatische Erkennung von neuen Absätzen wird wieder eingeschaltet.

`.write` *string*
`.wr`

Wenn Sie an Ihren Drucker spezielle Steuerzeichen schicken wollen, z.B. um Schriftarten anzusteuern, können Sie diese mit dem `.write` Kommando an die Ausgabe schicken, ohne die Formatierung zu beeinflussen. In *string* können neben normalen alphanumerischen Zeichen Zahlen und Controlcodes stehen. Zahlen werden dezimal notiert und können von 1 bis 255 gehen. Controlcodes werden mit dem Zeichen `^` eingeleitet, gefolgt von einem alphanumerischen Zeichen. Dabei entspricht das `a` und `A` dem Zeichen 0, `b` und `B` der 1 bis hin zu `_` für das Zeichen 31. Das Steuerzeichen ESC, das alle Drucker benötigen muß als `^[` in *string* auftauchen. Welche Codes Ihr Drucker kennt, müssen Sie dem Druckerhandbuch entnehmen.
Ein Beispiel für `.write` haben Sie schon in Kapitel 4.7 auf Seite 45 kennengelernt.

4.9 Aufruf von PROFF

Da PROFF von "großen" Betriebssystemen wie UNIX oder dem Großrechnersystem VMS auf den Atari ST übertragen wurde, hat es die für solche Systeme übliche Fähigkeit, beim Aufruf Parameter zu übernehmen. Falls Sie diese Möglichkeit ausnutzen wollen, müssen Sie entweder das Programm zum Start vom Desktop aus in `PROFF.TTP` umbenennen oder eine Kommando-Shell wie COMMAND oder GULAM verwenden. Der PROFF-Aufruf sieht dann so aus:

`proff` [+n1] [-n2] [-v] [-s] [-pon] [-ifn] input [output]

Dabei bedeuten die Parameter, die alle bis auf den Namen der Eingabedatei auch wegfallen können:

`[+`*n1*`]` PROFF beginnt den Ausdruck erst ab Seite *n1*.

`[-`*n2*`]` PROFF druckt nur bis zur Seite *n2* aus.

`[-v]` PROFF zeigt zusätzliche Informationen über den ausgedruckten Text und die interene Speicherverwaltung auf dem Bildschirm an.

`[-s]` PROFF hält den Ausdruck vor jeder neuen Seite an. Nützlich, wenn Sie Einzelblattzufuhr an Ihrem Drucker benutzen.

`[-po`*n3*`]` Entspricht einem `.po` *n3* Kommando am Anfang des Textes.

`[-`*ifile*`]` Entspricht einem `.include` *ifile* Kommando am Anfang des Textes. Sie können diesen Parameter auch mehrmals wiederholen, so daß mehrere Dateien eingefügt werden.

input Der Name der auszugebenden Textdatei

[*output*] Der Name der Ausgabedatei. Falls er fehlt, wird der formatierte Text einfach auf dem Bildschirm ausgegeben. Wenn Sie die Ausgabe direkt an den Drucker schicken wollen, geben Sie `PRN:` an (`PRN:` bezeichnet im Datei-System des Atari ST den Drucker).

Bei einem Aufruf

```
proff.ttp +2 -5 -v -po8 -imymacs.prf mytext.prf PRN:
```

verarbeitet PROFF die Seiten 2 bis 5 und benutzt dabei einen linken Rand von acht Zeichen. Die Datei `MYMACS.PRF` (eventuell mit einer Makrobibliothek) wird vor der Formatierung des eigentlichen Textes `MYTEXT.PRF` eingelesen und die Ausgabe wird nach PRN: geschickt, was unter GEMDOS den Drucker bezeichnet. Nach dem Programmlauf erscheint auf dem Bildschirm eine Statistik mit Angaben zum Programmlauf.

4.10 Übersicht

`.offset` [+|-][*n*] `.po`	Linken Rand auf *n* Zeichen festlegen, oder um *n* Zeichen verkleinern oder vergrößern.
`.rightmargin` [+|-][*n*] `.rm`	Rechten Rand auf *n* Zeichen festlegen oder um *n* Zeichen verkleinern oder vergrößern.

`.pagesize` [*n*]
`.ps`
`.pl`
	Seitenlänge auf *n* Zeilen festlegen.

`.m1` [*n*]	Rand vom Blattanfang bis zur Kopfzeile auf *n* Zeilen festlegen.
`.m2` [*n*]	Anzahl der Leerzeilen zwischen der Kopfzeile und dem eigentlichen Text auf *n* Zeilen festlegen.
`.m3` [*n*]	Anzahl der Leerzeilen zwischen dem Text und der Fußzeile auf *n* Zeilen festlegen.
`.m4` [*n*]	Rand von der Fußzeile bis zum Blattende auf *n* Zeilen festlegen.

`.spacing` [*n*]
`.spc`
`.ls`
	Zeilenabstand auf *n* Zeilen festlegen.

`.page` [*n*]
`.pg`
`.bp`
	Neue Seite beginnen. Falls *n* vorhanden, Seitenzahl auf *n* einstellen.

`.pagenumber roman|arabic` `.pn`	Seitennummerierung auf römische (`roman`) oder arabische (`arabic`) Ziffern einstellen.
`.nopaging` `.np`	`.page` und `.pagesize` ignorieren.
`.paging` `.pa`	`.page` und `.pagesize` beachten.

`.testpage` [*n*]
`.tp`
`.need`
`.ne`
	Prüfen, ob noch *n* Zeilen frei. Falls nicht, neue Seite beginnen.

`.header` *text* `.he`	Kopfzeile definieren.
`.footer` *text* `.fo`	Fußzeile definieren.
`.oh` *text*	Kopfzeile für Seiten mit ungerader Seitennummer definieren.
`.eh` *text*	Kopfzeile für Seiten mit gerader Seitennummer definieren.
`.of` *text*	Fußzeile für Seiten mit ungerader Seitennummer definieren.
`.ef` *text*	Fußzeile für Seiten mit gerader Seitennummer definieren.

`.fill`
`.fi`
`.f`
	Zeilenfüllung einschalten.

`.nofill` `.nf`	Zeilenfüllung ausschalten.

`.justify`
`.ju`
`.j`
	Rechten Randausgleich einschalten.

`.nojustify` `.nf`	Rechten Randausgleich ausschalten.
`.break` `.br`	Neuen Absatz beginnen.

`.leftmargin` [+|-][*n*]
`.lm`
`.in`
	Nachfolgende Zeilen einrücken oder Einrückung vergrößern oder verkleinern.

`.left` [+|-][*n*]
`.ti`
`.i`
	Nächste Zeile einrücken oder die Einrückung vergrößern oder verkleiner.

`.spaceto` [-][*n*] `.st`	Text auf Zeile *n* ausgeben.

`.skip` [*n*]
`.sp`
`.s`
	n Zeilen überspringen.

`.center` [*n*|`on`|`off`] `.ce`	Zentrierung schalten. Entweder die nächsten *n* Zeilen zentrieren oder Zentrierung ein- oder ausschalten.
`.underline` [*n*|`on`|`off`|`all`|`words`] `.ul`	Unterstreichen schalten. Entweder die nächsten *n* Zeilen unterstreichen oder Unterstreichen ein- oder ausschalten.
`.bold` [*n*|`on`|`off`] `.bd`	Fettschrift schalten. Entweder die nächsten *n* Zeilen fett drucken oder Fettschrift ein- oder ausschalten.

`.disablebolding`
`.dbo`
`.db`
	`.bold` ignorieren.

`.enablebolding`
`.ebo`
`.eb`
	`.bold` beachten.

`.contline` [*n text*] `.cl`	*text* in Inhaltsverzeichnis übernehmen.
`.printcont` [*n*] `.pc`	Inhaltsverzeichnis ausgeben mit einer Einrückung von *n* Zeichen je Gliederungsstufe.
`.set` *var* [*definition*] `.vs`	Textvariable *var* auf den Inhalt *definition* setzen oder vom Benutzer eingeben lassen.
`.get` *var prompt* `.vg`	Textvariable *var* mit Meldung *prompt* einlesen.
`.nr a-z` [`+`|`-`][*n*]	Numerische Variable setzen oder verrechnen.
`.define` *Makroname* `.de`	Makro-Definition beginnen.
`.en`	Makro-Definition beenden.
`.cchar` [*char*] `.cc`	Fluchtsymbol für Kommandos umdefinieren.

`.echar` [*char*] `.ec`	Fluchtsymbol für Variablen umdefinieren.
`.lex` *kommando* [*neu*]	Kommandonamen *kommando* in *neu* umdefinieren.

`:lx`
`:save`
`.sv`
	Interne Einstellungen sichern.

`.restore` `.rs`	Interne Einstellungen wiederherstellen.

`.require` [*filename*]
`.include`
`.source`
`.so`
	Textdatei *filename* einlesen.

`.noautoparagraph`
`.nap`
`.na`
	Absatzerkennung ausschalten.

`.autoparagraph` `.ap`	Absatzerkennung einschalten.
`.write` *string* `.wr`	Steuerzeichen ausgeben.

Chapter 5

Nicelist – Listing-Formatierer

Zum Ausdruck von Listings benutzt man normalerweise die Ausdruck-Routine des Desktops. Allerdings wird eine solche Datei dann nur unformatiert an den Drucker geschickt. Das Public-Domain-Programm NICELIST von Harrie F. A. de Leeuw erlaubt eine komfortable Ausgabe und bringt Übersicht in Ihre Listings.

Das GEM-gesteuerte Programm erlaubt verschiedene Schriftarten und Schriftgrößen. Damit Programmblöcke, z.B. Prozeduren nicht durch ein Seitenwechsel zerrissen werden, kann es angewiesen werden, sofort auf einer neuen Seite zu beginnen, falls nicht mehr genügend Zeilen frei sind. Automatisch erzeugt NICELIST eine Kopfzeile, in der der Datei-Name, Ausdruckdatum und -zeit sowie eine Seitennummer enthalten sind.

Nach dem Programmstart erscheint eine Dialogbox, wie Sie sie sicher auch von anderen GEM-Programmen her kennen.

In dieser Dialogbox geben Sie ein, welches Textfile ausgedruckt werden soll. Unter `Filename` wird der Datei-Name eingetragen, und bei `Directory` der Pfadname, d.h. der Ordner, in dem der Text zu finden ist. Mit den sechs `Drive`-Buttons legen Sie fest, auf welchem Laufwerk sich die Datei befindet.

Bei der Angabe des Datei-Namens können Sie auch die sogenannten Wildcards benutzen. Soll NICELIST z.B. alle Dateien ausdrucken, deren Name mit "B" beginnt, so müssen Sie `B*` als Datei-Namen angeben. Verwenden Sie das Zeichen `?` in der Namensangabe, so werden alle Dateien verarbeitet, die an der Stelle ein beliebiges Zeichen haben, z.B. würden mit `B?.DOC` die Dateien `B1.DOC`, `BA.DOC`, nicht aber `BAD.DOC` gedruckt.

Mit den drei Buttons unten rechts auf dem Bildschirm können Sie Programmaktionen auslösen. `Quit` verläßt das Programm; mit `Info` erhalten Sie eine kurze englischsprachige Anleitung zu NICELIST und `Start` schließlich veranlaßt das Programm, mit dem Ausdruck zu beginnen. Wenn Sie die Taste `Return` drücken, wird automatisch der `Start`-Button angewählt.

Daraufhin erscheint ein Ausgabefenster, in dem NICELIST Meldungen und Kommentare ausgibt. Sie erzeugt ein besonderes Kommando, das Sie später kennenlernen werden. In einer Alertbox meldet NICELIST die Beendigung des Ausdruck. Wenn die Angaben, die Sie zuvor über den Datei-Namen gemacht hatten, falsch waren können Fehlermeldungen erscheinen. Gab es keine Datei, auf die Ihre Angabe paßte, so meldet das Programm `No files selected!`. Gab es das angegebene Directory nicht, dann erscheint `Directory not found!`.

Die oben genannten Formatierungen werden durch spezielle Anweisungen gesteuert, die Sie in Ihr Listing einbetten. Diese Kommandos werden nicht mit ausgedruckt; ebenso lösen sie keinen Zeilenumbruch aus, so daß beispielsweise der mehrfache Wechsel der Schriftart selbst in einer Zeile möglich ist.

Jedes Kommando an NICELIST wird in ein Klammernpaar eingebettet. Klammerpaare sind dabei die Zeichenfolgen `(*` und `*)` sowie `/*` und `*/`. Als Programmierer erkennen Sie sofort, daß diese Zeichen in Pascal, Modula-2 und C Kommentare umschließen. Ein Compiler wird die Zeichen in den Kommentaren ignorieren, so daß die Formatierungsanweisungen die Programmentwicklung nicht stören.

Das Zeichen `%` leitet jedes Kommando ein. Das Kommando selber besteht aus einem oder mehreren Zeichen, wobei auf einige Befehle noch eine Zahl folgen kann. NICELIST ignoriert alle weiteren Zeichen innerhalb der Klammerpaare. Bedingung zur Erkennung eines Kommandos ist, daß das erste Zeichen des Klammerpaares – also `(` oder `/` – am Anfang einer Zeile im Eingabetext steht.
Mögliche Formatieranweisungen wären also

```
(* Überlies mich %F und mich auch *)
```

oder

```
/* %Normale Schrift */
```

Die erste Anweisung erzeugt auf dem Drucker einen Seitenvorschub; die zweite schaltet auf Normalschrift zurück. Damit wären wir bei den eigentlichen Kommandos, die sich in drei Gruppen einteilen lassen.

5.1 Die Kommandos

Für die allgemeine Formatsteuerung stehen drei Kommandos zur Verfügung:

`(* %F *)` oder `/* %F */`

erzeugt einen Seitenvorschub auf dem Drucker (`F` steht für "Formfeed"). Praktisch, wenn man ein Listing grob gliedern will: Für logisch zusammengehörige Prozeduren oder Funktionen wird eine neue Seite begonnen.

`(* %F`*xx*` *)` oder `/* %F`*xx*` */`

ist eine "intelligentere" Version des ersten Kommandos, wobei Sie anstelle von *xx* eine Zahl in dezimaler Schreibweise einfügen können. NICELIST überprüft, ob noch entsprechend viele Zeilen auf der gerade im Druck befindlichen Seite frei sind und erzeugt, falls dies nicht zutrifft, einen Seitenvorschub. Wenn Sie also wollen, daß ein Programmblock nicht durch einen Seitenwechsel unterbrochen wird, so zählen Sie ab, wieviele Zeilen notwendig sind und setzen die Formatieranweisung vor den Beginn des Blocks. Angenommen Sie haben eine Funktion programmiert, die im Listing 25 Zeilen einnimmt, sollte Ihr Programmtext wie folgt aussehen:

```
(* %F25 *)
Procedure ab ...
```

Das dritte Kommando zur allgemeinen Formatsteuerung gibt Ihnen Kontrolle über die oben angesprochene Kopfzeile mit Datei-Namen, Zeit und Datum sowie Seitennummer:

`(* %H+ *)` oder `/* %H+ */`
`(* %H- *)` oder `/* %H- */`

Mit dem "+"-Zeichen nach dem `H`-Kommando (`H` steht für "Heading", dtsch. "Kopfzeile") weisen Sie NICELIST an, auf jeder Seite die Kopfzeile mitzudrucken. Entsprechend schaltet das "-"-Zeichen diese aus.

Die Anzahl der Zeilen pro Seite steuert eine spezielle Datei, mit der die Druckeranpassung vorgenommen wird. Ihr Aussehen finden Sie in Kapitel 5.2 auf Seite 64 beschrieben.

Drucker sind heute in der Lage, Schrift in verschiedenen Variationen, wie Fett- oder Breitschrift zu Papier zu bringen. NICELIST besitzt Kommandos, mit denen einzelne Zeilen oder Zeichen hervorgehoben werden können. Die verschiedenen Stile sind in Ihrem Druckerhandbuch beschrieben.

`(* %E *)` oder `/* %E */`

schaltet die Breitschrift Ihres Druckers ein (`E` steht für "Enlarged print"). Normalerweise haben dann 40 Zeichen in einer Zeile Platz.

`(* %C *)` oder `/* %C */`

dagegen stellt eine Schmalschrift (`C` wie "Condensed print") ein. Ihr Drucker kann dann in einer Zeile 96 Zeichen ausgeben.

`(* %M *)` oder `/* %M */`

Hiermit wird der Fettdruck ("eMphasized print") eingeschaltet. Der Matrixdrucker gibt dann jedes Zeichen zweimal aus, nur beim zweiten Durchlauf etwas horizontal versetzt, wodurch sie dicker erscheinen.
Einen ähnlichen Effekt erzeugt

`(* %D *)` oder `/* %D */`

`D` steht dabei für "Double print". Jedes Zeichen wird wiederum zweimal gedruckt, nur diesmal vertikal versetzt. Es ergibt sich eine ähnlich Hervorhebung.

`(* %U *)` oder `/* %U */`

schaltet eine Unterstreichung ein, wodurch die Zeichen ebenfalls optisch hervorgehoben werden.

`(* %SU *)` oder `/* %SU */`

weist NICELIST an, die folgenden Zeichen hochgestellt auszugeben ("SUperscript"). Der Gegenpart dazu ist

`(* %SB *)` oder `/* %SB */`

was eine Tiefstellung auslöst ("SuBscript"). In beiden Fällen werden die Zeichen in halber Höhe dargestellt. Diese Tatsache nutzt der nächste Befehl aus:

`(* %T *)` oder `/* %T */`

Er kombiniert einige der obigen Varianten: Der Drucker wird auf die Schmalschrift eingestellt, gleichzeitig benutzt er die Tiefstellung. Zusätzlich wird ein Zeilenvorschub von 6/72 Zoll definiert, was genau die Hälfte des Normalen ist (`T` steht für "Tiny mode"). Da tiefgestellte Zeichen kleiner sind, können in dieser Schriftart 144 Zeilen pro Seite ausgegeben werden, also das Doppelte wie in normaler Schrift. Zur Einstellung des Zeilenabstandes gibt es auch noch das folgende Kommando:

`(* %SP`*xx*` *)` oder `/* %SP`*xx*` */`

Für *xx* müssen Sie eine Dezimalzahl eingefügt, womit der Zeilenabstand auf xx/72 Zoll definiert wird. Der normaler Zeilenabstand bei Matrixdruckern beträgt 12/72 Zoll. Wenn Sie ein Kommando `(* %SP11 *)` benutzen,

können Sie ca. sechs Zeilen mehr auf einer Seite unterbringen, ohne daß sich die Zeilen überlappen.
Die Anweisung

`(* %N *)` oder `/* %N */`

setzt alle vorherigen Schriftstile zurück und schaltet auf Normalschrift. Damit ist die zweite Kommandogruppe vollständig; es folgen noch zwei weitere Anweisungen von NICELIST.

`(* %B *)` oder `/* %B */`

schickt das BEL-Zeichen an den Drucker, was ein Klingelzeichen am Drucker auslöst. Der Befehl ist als letzte Zeile in Ihrem Listing nützlich, denn hat der Printer das Listing ausgegeben, erstattet er Ihnen so eine akustische Meldung, daß der Druck beendet ist.

`(* %R ... *)` oder `/* %R ... */`

ist das letzte Kommando von NICELIST. Wird diese Anweisung erkannt, gibt das Programm alles, das auf das `%R`-Kommando (`R` steht für "Remark", also "Kommentar") in der Listingzeile folgt auf dem Bildschirm aus. Sie können somit bei längeren Programmtexten Stand des Ausdrucks auf dem Bildschirm dokumentieren.

Damit diese Ansammlung von Befehlen nicht unillustriert bleibt, noch ein kleines Beispiel. Geben Sie mit einem Editor folgenden kleinen Text mit NICELIST-Befehlen ein:

```
(* %Remark: Testfile für NiceList *)
PROCEDURE
(*%M*)Test
(*%N*);
(* %R Prozedur Test wird ausgedruckt *)(*%T*)
VAR
(*%U *)a:INTEGER;
(*  %N     *)(*%T*)
BEGIN
  a:=a+1;
END;
(*%N*)
BEGIN
  Test;
END.
(*%R Fertig !!! *)
(*%B*)
```

Starten Sie nun NICELIST und lassen Sie den Text ausdrucken. Wenn Ihr Drucker richtig installiert ist (s.u.) erhalten Sie ein optisch aufgepepptes Listing, einige Meldungen auf dem Bildschirm und ein akustisches Signal beim Ende des Druckens.

Die Anweisungen bewirken, daß NICELIST alle Prozedurnamen hervorgehoben ausdruckt, die Variablendeklarationen unterstreicht und schließlich den eigentlichen Prozedurtext in der kleinen Tiny-Schrift druckt. Das Hauptprogramm erscheint wieder in normaler Schrift und mit `%R` meldet NICELIST den Fortgang des Ausdrucks.

Wie Sie sehen, ist es möglich, innerhalb einer Zeile die Schriftart zu wechseln. Allerdings muß jedes NICELIST-Kommando in einer neuen stehen, so daß der eigentliche Text leider an Lesbarkeit verliert. Die eingefügten `%R`-Befehle sind für die Bildschirmausgaben verantwortlich. Das abschließende `%B`-Kommando löst ein Klingelzeichen auf dem Drucker aus, das akustisch das Ende des Ausdrucks meldet.

5.2 Druckeranpassung

Damit NICELIST die verschiedenen Schriftarten überhaupt verwenden kann, muß das Programm die Steuercodes Ihres Druckers kennen. Da es dafür keinen wirklichen Standard gibt, wird eine Druckerinstallation benötigt. NICELIST benutzt dazu ein Textfile mit dem Namen `NLSETUP.PRT`, das beim Programmstart eingelesen wird, und in dem sich die Steuercodes als Klartext befinden.

Jede Zeile in dieser Datei besteht aus einem Kommandonamen, den dazugehörigen Steuercodes in dezimaler Schreibweise durch Kommatas getrennt und eventuell noch einem Kommentar. Zeilen, die mit dem /-Zeichen beginnen gelten komplett als Kommentarzeilen. Kommandoname, Steuercodes und Kommentar müssen durch mindestens ein Leerzeichen getrennt sein.
Die Kommandonamen lauten:

`SetDouble`	Doppelschrift ein
`SetEnlarge`	Breitschrift ein
`SetEmphasize`	Hervorgehobener Druck ein
`SetCondense`	Engschrift ein
`SetUnderline`	Unterstreichung ein
`SetSubscript`	Tiefgestellte Schrift ein
`SetSuperscript`	Hochgestellte Schrift ein

`ResetDouble`	Doppelschrift aus
`ResetEnlarge`	Breitschrift aus
`ResetEmphasize`	Hervorgehobene Schrift aus
`ResetCondense`	Engschrift aus
`ResetUnderline`	Unterstreichung aus
`ResetSubscript`	Tiefgestellte Schrift aus
`ResetSuperscript`	Hochgestellte Schrift aus
`SetSpacing`	Setzt den Zeilenvorschub des Druckers auf x/72 Zoll, wobei der Wert x nach dem Kommando an den Drucker gesandt wird.
`DotsPerLine`	Anzahl der Punkte, um die ein Zeilenvorschub das Papier bewegt
`NrOfLinesPerPage`	Anzahl der Zeilen auf einer Seite

Wenn Ihr Drucker für die Unterstreichung die Codes 27,45 und 1 verwendet, so muß in Ihrem Installationsfile die Zeile

```
SetUnderline   27,45,1   Kommentar: Unterstreichung ein
```

stehen. Dem Programm liegt eine Installationsdatei für einen EPSON-Drucker bei. Falls Sie einen anderen benutzen, ist es am einfachsten, wenn Sie diese Datei mit einem Texteditor bearbeiten und Ihre Steuercodes einsetzen.

5.3 Kommandoübersicht

Kommando	bewirkt
`%F`	Seitenvorschub
`%F`*xx*	Seitenvorschub, wenn weniger als `xx` Zeilen frei
`%H+`	Kopfzeile an
`%H-`	Kopfzeile aus
`%E`	Breitschrift ein
`%C`	Schmalschrift ein
`%M`	Fettdruck ein
`%D`	Doppeldruck ein
`%U`	Unterstreichen ein
`%SU`	Hochstellen ein
`%SB`	Tiefstellen ein
`%T`	Kleinschrift ein
`%SP`*xx*	Zeilenabstand auf *xx* Punkt einstellen
`%N`	Normalschrift ein
`%B`	Klingelzeichen auslösen
`%R ...`	Kommentar

Chapter 6

Datei-Verwaltung mit SBASE

Auf dem Public-Domain Sektor gibt es für den Atari ST sehr viel Software, mit der Sie Adressen, Videokassetten oder Disketteninhalte verwalten können. Jedes dieser Programme ist eine spezialisierte Datei-Verwaltung und nicht an andere als die vorgegebenen Anwendungen anpaßbar. Richtige Datei-Verwaltungen mit frei definierbaren Feldern, guten Suchmöglichkeiten und variablen Ausgabeformaten gibt es leider sehr wenige. Der Grund dafür ist klar: Eine Programm für eine Adressenkartei ist schnell in Basic eingetippt; ein richtiges Datei-Verwaltungsprogramm[1] ist komplexer und erfordert Programmierpraxis.

Ein solches System ist das Shareware-Programm SBASE von Mark Overmars aus den Niederlanden. Es bietet freie Datei-Gestaltung, ein elegantes Suchkonzept, Sortiermöglichkeiten und schließlich einen sehr starken Reportgenerator. Dateien können maximal bis zur Größe des verfügbaren Hauptspeichers anwachsen, was für kleine bis mittlere Anwendungen kein Hindernis ist, zumal das Programm dann schneller arbeitet als eine diskettenorientierte Dateiverwaltung.

[1] Eine Datenbank ist in der Lage, mehrere Dateien miteinander zu verknüpfen und Zusammenhänge zwischen Einträgen in unterschiedlichen Dateien herzustellen. Die meisten Programme für Home-Computer sind Datei-Verwaltungen, da ihnen diese Fähigkeiten völlig fehlen.

6.1 Datei-Definition

Nach dem Start von SBASE können Sie mit der normalen File-Select-Box eine Datei zum Bearbeiten auswählen. Beim ersten Programmstart werden Sie noch keine Datei haben, also muß zunächst das Aussehen einer neuen definiert werden. Dazu wählen Sie einfach den gewünschten Namen aus. Da SBASE keine Datei dieses Namens findet, fragt es in einer Alert-Box nach, ob eine neue angelegt werden soll, was Sie mit dem `Create`-Button bestätigen.

Nun können Sie für jedes der bis zu 16 Felder in einem Datei-Eintrag einen Namen, eine maximale Länge und einen Feldtyp festlegen. Der Name kann bis zu acht Zeichen lang sein und erscheint später bei der Dateneingabe in dieser Form auf dem Bildschirm. Jedes Feld kann bis zu 40 Zeichen aufnehmen. Die gewünschte Maximallänge tragen Sie unter `Length` ein. Für den Feldtyp bestehen drei Möglichkeiten:

`text` Hierbei können beliebige Zeichen in das Feld eingegeben werden. Beim Sortieren wird die alphabetische Reihenfolge verwendet.

`Number` Sie können Zahlen eingeben, wobei SBASE die Eingabe allerdings nicht überprüft. Dafür wird beim Sortieren für Zahlen die numerische Reihenfolge verwendet, so daß z.B. die 10 nicht vor die 2 einsortiert wird, wie das bei alphabetischer Ordnung der Fall wäre.

`Unique` Bei diesen Feldern fügt SBASE automatisch eine Zahl ein, die in der Datei einmalig vorkommt. Vereinsmitglieder würden so z.B. eine eindeutige Mitgliedsnummer bekommen, oder Ihre Bücher erhielten eine eindeutige Signatur. Die Länge eines solchen Feldes ist auf vier Zeichen festgelegt.

Nach der Definition eines Feldes können Sie mit `Next` das nächste Feld angehen oder mit `Ready` die Definition beenden. Sie haben nun die Struktur Ihrer Datei festgelegt und können sie mit Einträgen füllen. Ab jetzt verhält sich SBASE so, als wenn Sie eine schon vorhandene Datei editieren wollten.

Übrigens: Das erste, was Sie vor dem Einrichten einer neuen Datei tun sollten, ist den Rechner auszuschalten. Planen Sie auf Papier das Aussehen der Datei und der erforderlichen Felder. Der Grund für diese Sorgfalt ist, daß es mit SBASE schwierig (wenn auch nicht unmöglich) ist, eine Datei-Definition nachträglich zu ändern. Nach der Eingabe der Definition sollten Sie sie zunächst noch einmal auf Fehler überprüfen. Falls Sie welche entdecken, beginnen Sie die Definition neu. Haben Sie schon einige Daten eingegeben und stellen dann fest, daß Sie ein zusätzliches Feld benötigen, müssen Sie auf einem Umweg über ASCII-Dateien alle Daten auf Diskette

schreiben, dann eine neue Datei definieren und die alten Daten wieder einlesen. Diesen Vorgang finden Sie weiter unten zwar beschrieben, aber durch sorgfältige Planung ersparen Sie sich die Arbeit!

Definieren Sie nun eine kleine Beispieldatei, die Sie `TESTBASE.DAT` nennen sollten und an der wir im Folgenden einige der Möglichkeiten von SBASE durchspielen werden. Ein Eintrag in der Datei besteht aus acht Feldern mit den folgenden Eigenschaften:

Feldname	Feldlänge	Feldtyp	*Feldnummer*
Vorname	10	Text	1
Nachname	15	Text	2
Geschlecht	1	Text	3
Straße	20	Text	4
Nr.	3	Number	5
PLZ	4	Number	6
Ort	25	Text	7
ID	1	Unique	8

Wie Sie sehen, handelt es sich um eine Adreß-Datei. Die Feldnummer ist keine Eingabe an SBASE. Sie ist hier nur zur Orientierung aufgeführt und bekommt erst wieder beim Report-Generator Bedeutung.

6.2 Dateneingabe

Nach dem Laden einer Datei oder der Neudefinition erscheint auf dem Bildschirm das Arbeitsfeld von SBASE. Links oben wird der Name der gerade bearbeiteten Datei angezeigt, darunter genau ein Eintrag dargestellt. In dessen linker Spalte sind die Feldnamen zu sehen; in seiner rechten finden die eigentlichen Daten Platz. Freie Zeichen werden als Unterstriche dargestellt. Rechts vom Eintragsfeld sehen Sie einen Rollbalken, wie Sie ihn auch von den normalen GEM-Fenstern her kennen. Er dient zum Bewegen in einer Datei. Die rechte Hälfte des Arbeitsfeldes wird von den Funktions-Buttons eingenommen, über denen sich einige Statusangaben befinden.

Sie können nun den angezeigten Eintrag mit Daten füllen oder schon vorhandene ändern. Der Cursor, der durch einen senkrechten Strich dargestellt wird, befindet sich anfangs im ersten Feld und kann mit den Pfeiltasten in dem Feld bzw. zwischen den Feldern bewegt werden. Zum Löschen können Sie die Tasten `Backspace` zum Entfernen des Zeichens links vom Cursor, `Delete` für das Zeichen unter dem Cursor und `Esc` zum Löschen eines gesamten Feldes verwenden. Dieses Verhalten kennen Sie eventuell

auch von anderen GEM-Formularen oder -Dialogboxen, denn SBASE benutzt natürlich die gleichen GEM-Funktionen.

Wollen Sie nun weitere Einträge ausfüllen, müssen Sie zunächst einen neuen leeren Eintrag erzeugen. Dazu haben Sie zwei Möglichkeiten, die über die Buttons `Insert` und `Append` ausgelöst werden. `Insert` erzeugt einen neuen Eintrag vor dem gerade bearbeiteten und `Append` einen danach. Wenn Sie immer `Append` benutzen, befinden sich die Einträge in der Eingabereihenfolge in der Datei.

Dieses Vorgehen ähnelt stark dem Verfahren bei Karteikästen. Sie nehmen eine neue Karteikarte, füllen diese aus und stecken Sie in den Karteikasten. SBASE steckt sie immer vor oder nach die gerade bearbeitet Karte.

6.3 In den Daten blättern

Wenn Sie mehrere Einträge in einer Datei haben, brauchen Sie eine Aktion, mit der Sie sie durchsehen können. Dabei kommt der Rollbalken ins Spiel. Das Anklicken der Felder ⇑ und ⇓ zeigt den vorherigen bzw. den folgenden Datensatz an. Ähnlich wie bei GEM-Fenstern können Sie aber auch den Rollbalken selber mit der Maus verschieben und bewegen so die Anzeige in der Datei. Balken ganz nach oben zeigt den ersten Datensatz an, Balken ganz nach unten den letzten. Setzen Sie ihn mittenhinein, erscheint der Datensatz, dessen Position in der Datei der Stellung des Rollbalkens entspricht.

Die Größe des Balkens ergibt sich übrigens aus der Anzahl der vorhandenen Datensätze. Dieser Wert wird rechts oben als `Total` angezeigt. Daneben finden Sie die Angabe `Number`, bei der die Nummer des gerade angezeigten Datensatzes steht.

Kommen wir zurück zu unserer kleinen Adreß-Datei. Nachdem der Definition ihrere Struktur können nun Sie Daten eingeben. Benutzen Sie die beschriebenen Kommandos um folgende Datensätze zu erfassen:

```
Vorname Nachname G. Straße              Nr. PLZ  Ort             ID

Minni   Maus     w  Gemstraße           2   1632 Atarihausen     1
Moni    Monitor  w  Anzeigeweg         124 6910 Bildheim        2
Karl    Keyboard m  Tastenstraße       13  9120 Qwertystadt     3
Franz   Floppy   m  Schlappscheibenweg 72  3140 Diskettenburg   4
Gisela  Gem      w  Romstraße          92  3103 Datenbus        5
Martin  Menu     m  Klappstraße        1   2312 Dropdownstadt   6
Paula   Pixel    w  Linienstraße       400 1240 Monitorburg     7
```

```
Ivan    Icon     m  Dialogweg          12  1422 Resourcenhausen 8
```

Sie werden bei der Eingabe übrigens beobachten, daß im Feld `ID` immer schon eine Nummer vorgegeben wird (sie ist hier nur der Vollständigkeit halber abgedruckt; Sie brauchen sie nicht einzugeben). Sie haben diesen Effekt erreicht, indem Sie `ID` als `Unique`-Feld definiert hatten. Spielen Sie nun etwas mit dem Rollbalken herum und schauen Sie alle Datensätze nochmal durch.

6.4 Einträge löschen

Der `Delete`-Knopf löscht den gerade angezeigten Datensatz. Aus Sicherheitsgründen müssen Sie den Button zweimal anklicken (Doppelklick wie beim Öffnen eines Laufwerks auf dem Desktop), damit keine Daten aus Versehen verloren gehen. Dagegen gibt es noch eine zweite Sicherung: SBASE löscht einen Eintrag nicht sofort, sondern markiert ihn nur als gelöscht. Mit `Undel` können Sie daher einen gelöschten Datensatz wieder in die Datei aufnehmen. Geben Sie das Kommando mehrmals, so werden entsprechend viele vorher gelöschte Einträge wieder aktivert. Wirklich gelöscht werden die Daten erst, wenn Sie das Programm verlassen und damit die Datei auf Diskette oder Festplatte schreiben.

6.5 Einträge auswählen und sortieren

Der nächste Button auf dem Bildschirm heißt `Select`. Er dient dazu, einen Teil der Einträge auszuwählen. Diese Menge von Daten wird Auswahl genannt. Beim Programmstart haben Sie keine Auswahl getroffen oder genauer gesagt, alle Einträge sind ausgewählt. Sie können das an der Anzeige `Level` erkennen. Hier steht eine Null, d.h. es sind alle Einträge ausgewählt. Ihre Anzahl steht neben `Selected`.

Wenn Sie nun das `Select`-Kommando geben, erscheint eine Dialogbox, in der Sie eine Bedingung festlegen, die alle Einträge der neuen Auswahl erfüllen müssen. Dazu wählen Sie ein Feld, das die Bedingung erfüllen soll, die Art der Bedingung und einen Wert (Zeichenkette oder Zahl) für die Bedingung aus. Mögliche Bedingungen sind:

`<` Der Inhalt des Feldes muß kleiner dem angegebenen Wert sein. Bei Textfeldern heißt das, er muß im Alphabet vor der angegebenen Zeichenkette stehen (das heißt insbesondere, daß die 10 vor der 2 kommt!).

`<=`	Der Inhalt des Feldes muß kleiner oder gleich dem angegebenen Wert sein.
`=`	Der Feldinhalt muß gleich dem Wert sein.
`>=`	Der Feldinhalt muß größer oder gleich der Angabe sein. Bei Textfeldern muß er also im Alphabet nach dem Wert kommen.
`>`	Der Inhalt des Feldes muß größer der Eingabe sein.
`!=`	Der Feldinhalt muß ungleich dem Wert sein.
`Incl.`	Die angegebene Zeichenkette muß in dem Feldinhalt vorkommen. Bei Eingabe von "er" würden Felder mit "Robert" oder "hergeben" ausgewählt, nicht aber "Auswahl".
`NOT Incl`	Das Gegenteil von Incl. "Auswahl" würde ausgewählt, nicht aber "Robert" und "hergeben".

Haben Sie die Bedingung festgelegt und auf `OK` geklickt, wählt SBASE die entsprechenden Einträge aus, was an den Anzeigen für `Selected` und `Level` erkennbar wird. Letztere deutet schon an, daß Auswahlen geschachtelt werden können. Zunächst wird die Auswahl unter allen Einträgen getroffen. Eine erneute Auswahl bezieht sich dann nur noch auf die vorher ausgewählten Einträge. Diese Schachtelung, deren Tiefe `Level` anzeigt, kann bis über 32000 Stufen haben ...

Aus unserer kleinen Beispieldatei wollen wir alle weiblichen Personen, deren Vorname mit dem Buchstaben "M" beginnt herausfinden. Die erste Bedingung ist also "Feld 'Geschl.' ist gleich 'm'". In der Dialogbox, die auf das Anklicken von `Select` erscheint müssen Sie also den Feldnamen "Geschl.", die Operation "=" und als Parameter "w" auswählen bzw. eingeben. Es kommen vier Einträge in die Auswahl:

```
Vorname Nachname G. Straße       Nr. PLZ  Ort         ID

Minni   Maus     w  Gemstraße    2   1632 Atarihausen 1
Moni    Monitor  w  Anzeigeweg   124 6910 Bildheim    2
Gisela  Gem      w  Romstraße    92  3103 Datenbus    5
Paula   Pixel    w  Linienstraße 400 1240 Monitorburg 7
```

Um nun noch die zweite Bedingung festzulegen, wird `Select` nochmal aufgerufen und in der Dialogbox "Feld 'Vorname' enthält 'M"' festgelegt. Die entsprechende Operation ist "incl". Die zwei ausgewählten Einträge entsprechen den zwei Bedingungen:

```
Vorname Nachname G. Straße     Nr. PLZ  Ort         ID
```

```
Minni   Maus     w  Gemstraße  2   1632 Atarihausen 1
Moni    Monitor  w  Anzeigeweg 124 6910 Bildheim    2
```

Um nun aber wieder Zugriff auf die gesamte Datei zu erhalten, müssen die Auswahlen aufgehoben werden. Dazu dient `Back`, das ein Level der Auswahl "zurückgeht". In dem Beispiel kommen dann wieder alle Einträge mit "Geschl." gleich "m" zum Vorschein; nach einer Wiederholung des Kommandos sind alle Auswahlkriterien aufgehoben. Wählen Sie nun alle weiblichen Einträge aus, die im Postleitzahlengebiet 1000 wohnen. (Kleine Hilfe: Die zweite Auswahl muß in zwei Schritten vorgenommen werden: PLZ kleiner 2000 und PLZ größer 999.)

Ein weiteres Kommando von SBASE ist `Sort`. Mit ihm wird die Datei nach einem Feld sortiert. In einer Dialogbox legen Sie fest, nach welchem Feld und in welcher Reihenfolge geordnet wird. `Increasing` (aufsteigend) ist die normale Reihenfolge, bei welcher der "kleinste" Eintrag an den Anfang und der "größte" an das Ende der Datei sortiert wird. "Klein" bedeutet bei Textfeldern am Anfang des Alphabets. Mit `Decreasing` (fallend) wird genau andersherum sortiert.

In unserer Beispieldatei sollen die Einträge so sortiert werden, daß zunächst die weiblichen Mitglieder unserer Adreß-Datei und dann die männlichen eingeordnet sind. Beide Gruppen sollen dann jeweils alphabetisch sortiert sein. In einem ersten Sortiervorgang werden alle Einträge nach dem Feld "Nachname" aufsteigend sortiert. Da nun alle weiblichen Mitglieder unserer Datei (Feld "Geschl." gleich "w") vor den männlichen (Feld "Geschl." gleich "m") kommen sollen, wird das Feld "Geschl." fallend sortiert:

```
Vorname Nachname G. Straße              Nr. PLZ  Ort             ID

Gisela  Gem      w  Romstraße           92  3103 Datenbus        5
Minni   Maus     w  Gemstraße           2   1632 Atarihausen     1
Moni    Monitor  w  Anzeigeweg          124 6910 Bildheim        2
Paula   Pixel    w  Linienstraße        400 1240 Monitorburg     7
Franz   Floppy   m  Schlappscheibenweg 72  3140 Diskettenburg   4
Ivan    Icon     m  Dialogweg           12  1422 Resourcenhausen 8
Karl    Keyboard m  Tastenstraße        13  9120 Qwertystadt     3
Martin  Menu     m  Klappstraße         1   2312 Dropdownstadt   6
```

6.6 Datenausgabe

Um Ihre Datensätze, genauer gesagt, die Datensätze der aktuellen Auswahl zu Papier zu bringen, müssen Sie das `Report`-Kommando benutzen. Es fragt Sie nach dem Namen eines Reports und dem Ziel der Ausgabe. Sie

kann an den Drucker direkt oder in eine Datei, deren Namen Sie zusätzlichangegeben müssen, geschickt werden.

In einem Report stehen Angaben darüber, wie SBASE die Datensätze ausgeben soll. Dabei haben Sie vielfältige Möglichkeiten z.B. zur Druckersteuerung oder zur Verwendung von Schriftarten. Einen Report erzeugen Sie mit einem Texteditor im ASCII-Format[2], z.B. 1stWord, bei dem Sie allerdings den WP-Modus abschaltet müssen.

SBASE nimmt bei der Ausgabe den Report und den ersten Datensatz, mischt beide zusammen und fährt dann mit dem zweiten fort, bis alle Einträge durchgearbeitet wurden. In dem Report erkennt das Programm bestimmte Kommandos und fügt an ihrer Stelle z.B. die Einträge aus dem gerade bearbeiteten Datensatz ein. Alle anderen Zeichen im Report werden unverändert ausgegeben.

Jedes Kommando beginnt mit dem Zeichen \ (Backslash) und wird mit ihm beendet. Ein solches Zeichen zur Kennzeichnung von Befehlen nennt man Fluchtsymbol. Das Kommando wird jeweils mit einem Buchstaben eingeleitet (auf Groß- und Kleinschreibung achten!) und kann ein oder mehrere Parametern haben.

Das erste Kommando dient zum Einsetzen eines Feldeintrages:

`\f`$\star$`\`

Anstelle von $\star$ tragen Sie die Nummer des gewünschten Feldes ein. In einem Report, der mit der Beispieldatei verwendet werden soll, ergibt `\f2\` den Inhalt des Feldes "Nachname" des jeweiligen Datensatzes. Das Kommando hat noch eine zweite Option:

`\f`$\star$`:`$\#$`\`

Damit wird ebenfalls der Inhalt der Feldes $\star$ eingefügt, nur wird seine Länge auf $\#$ Zeichen mit Leerstellen aufgefüllt. Einträge, die länger als $\#$ sind, werden leider nicht abgeschnitten, sondern in voller Länge ausgegeben. Ist $\#$ gleich 0, so wird als Ausgabelänge der maximale Wert angenommen, der in der Definition der Datei festgelegt wurde.
Ähnlich diesem Kommando können Sie mit

`\n`$\star$`\`

den Namen des Feldes $\star$ in die Ausgabe einfügen. Die Option `\n`$\star$`:`$\#$`\` ist dabei ebenso möglich und hat die gleiche Bedeutung wie bei `\f`$\star$`:`$\#$`\`.
Die Nummer des gerade bearbeiteten Datensatzes kann auch in die Ausgabe übernommen werden.

`\`x`\`

[2] Also ohne Formatkommandos oder Angaben über Schriftarten

Hiermit gibt SBASE diese Zahl aus. Auch hier können Sie mit \x:#\ bestimmen, daß dafür # Zeichen verwendet werden.
Um die Datensätze aus unserer Adressenkartei auszugeben, wäre folgender Report sinnvoll:

```
Datensatz \x\:

\n1\: \f1\
\n2\: \f2\
\n3\: \f3\
\n4\: \f4\
\n5\: \f5\
\n6\: \f6\
\n7\: \f7\
\n8\: \f8\
```

Die Ausgabe sieht dann so aus:

```
Datensatz 1:

Vorname: Minni
Nachname: Maus
Geschl.: w
Straße: Gemstraße
Nr.: 2
PLZ: 1632
Ort: Atarihausen
ID: 1
```

Wenn Sie einen Teil des Reports nur an bestimmten Stellen ausgeben wollen, haben Sie dafür einige Kommandopaare zur Verfügung:

`\(B\ und \)B\`
Alles was zwischen diesen Kommandos im Report steht, wird nur bei der Verarbeitung des ersten Datensatzes berücksichtigt. Damit können Sie z.B. Überschriften für Ihre Reports definieren.

In unserem Beispielreport könnte die folgende Zeile eingefügt werden:

```
\(B\Daten aus der Testdatei \)B\

Datensatz \x\:

⋮
```

Den ersten Datei-Satz würde SBASE dann so ausgegeben:

```
Daten aus der Testdatei

Datensatz 1:

⋮

\(E\ und \)E\
```

Dies sind die Gegenstücke zu den letzten beiden Kommandos. Alles, was zwischen ihnen steht, wird nur bei der Verarbeitung des letzten Eintrags in der Datei berücksichtigt.
Eine Anwendung in dem Beispiel:

```
⋮

\n8\: \f8\

\(E\ Das war alles ! \)E\
```

Entsprechend die Ausgabe von SBASE:

```
⋮

Ort: Resourcenhausen
ID: 8

 Das war alles !
```

`\(A⋆\` und `\)A\`

Der zwischen diesen Kommandos stehende Teil des Reports wird nur bei den Einträgen bearbeitet, deren Nummer durch $\star$ teilbar ist, also bei 4 nur beim 4., 8., 12. Datensatz usw. Neben der Möglichkeit, Zwischenüberschriften zu erzeugen, können Sie damit z.B. dafür sorgen, daß Datensätze nicht durch das Seitenende geteilt ausgegeben werden. Wenn z.B. genau dreieinhalb Einträge auf eine Seite passen, müßten Sie nach jedem dritten einen Seitenvorschub ausgeben lassen.

`\(I⋆\` und `\)I\`

Der durch dieses Kommandopaar geklammerte Reportteil wird nur dann ausgegeben, wenn das Feld Nummer $\star$ in der Datei definiert ist. Sie können damit Reports erstellen, die sich auf alle möglichen Dateien anwenden lassen.
Ein solcher Report, der alle Felder samt Namen ausgibt, sieht folgendermaßen aus:

```
\(I1\\N1\: \F1\\)I\
\(I2\\N2\: \F2\\)I\
\(I3\\N3\: \F3\\)I\
\(I4\\N4\: \F4\\)I\
\(I5\\N5\: \F5\\)I\
\(I6\\N6\: \F6\\)I\
\(I7\\N7\: \F7\\)I\
\(I8\\N8\: \F8\\)I\
\(I9\\N9\: \F9\\)I\
\(I10\\N10\: \F10\\)I\
\(I11\\N11\: \F11\\)I\
\(I12\\N12\: \F12\\)I\
\(I13\\N13\: \F13\\)I\
\(I14\\N14\: \F14\\)I\
\(I15\\N15\: \F15\\)I\
\(I16\\N16\: \F16\\)I\
```

In den Reports können viele Möglichkeiten Ihres Druckers ausgenutzt werden. Für die Steuerung der Schriftarten dient das Kommando

```
\s⋆\
```

Hier müssen Sie für von $\star$ einen Buchstaben einsetzen, der angibt, welche Schriftart verwendet werden soll:

`B`	Breitschrift einschalten
`b`	Breitschrift ausschalten
`C`	Komprimierte (condensed) Schrift einschalten
`c`	Komprimierte Schrift ausschalten
`D`	Doppelte Größe einschalten
`d`	Doppelte Größe ausschalten
`E`	Elite-Schrift (12 Zeichen pro Zoll) einschalten
`e`	Elite-Schrift ausschalten
`I`	Kursivschrift (italic) einschalten
`i`	Kursivschrift ausschalten
`L`	Große Schrift einschalten
`l`	Große Schrift ausschalten
`N`	Schönschrift (Near-Letter-Quality) einschalten
`n`	Schönschrift ausschalten
`P`	Proportionalschrift einschalten
`p`	Proportionalschrift ausschalten
`Q`	Vierfache (quadruple) Schriftgröße einschalten
`q`	Vierfache Schriftgröße ausschalten
`S`	Kleinschrift (small) einschalten
`s`	Kleinschrift ausschalten
`U`	Unterstreichen einschalten
`u`	Unterstreichen ausschalten

Sie sehen, daß die Ausdrucke sehr flexibel gestaltet werden können. Welche Schriftarten Ihr Drucker unterstützt hängt von dem Fabrikat ab. Die Anpassung von SBASE an Ihren Drucker finden Sie in Kapitel 6.7 auf Seite 79 beschreiben.

`\p\`

Dieses Kommando erzeugt einen Seitenvorschub auf dem Drucker.
Die Ausgabe unserer Adressenkartei können Sie damit sehr leicht optisch verändern. Um die Nummer des Datensatzes in Kursivschrift auszugeben und jeweils nur einen Satz pro Seite zu drucken, müßten Sie den obigen

Report folgendermaßen ändern:

```
Datensatz \I\\x\\i\:

\n1\: \f1\
\n2\: \f2\
\n3\: \f3\
\n4\: \f4\
\n5\: \f5\
\n6\: \f6\
\n7\: \f7\
\n8\: \f8\
\p\
```

`\c⋆\`

Das Zeichen mit dem ASCII-Code ⋆ wird an den Drucker geschickt. Nur falls Sie besondere Fähigkeiten des Printers ausnutzen, oder Zeichen, die sich in Ihrem Editor nicht eingeben lassen, verwenden wollen, brauchen Sie dieses Kommando.

Das Fluchtsymbol \ kennzeichnet die Kommandos in einem Report. Daher kann es nicht mehr ausgegeben werden. Um \ zu drucken gibt es das Kommando.

`\\`

das genau das Fluchtsymbol an die Ausgabe schickt. Falls Sie in Ihrem Report dieses Zeichen oft verwenden wollen, so ist es einfacher, das Fluchtsymbol umzudefinieren. Dazu dient

`\e⋆`

Ab sofort wird das Zeichen, das Sie anstelle von ⋆ angeben, als Fluchtsymbol verwendet.

6.7 Druckeranpassung

Zur Anpassung an den Drucker verwendet SBASE eine Datei namens `SBASE.PRN`. Dabei handelt es sich um eine einfache ASCII-Datei, in der die Codes definiert sind, die für eine bestimmte Schriftart an den Drucker geschickt werden müssen.

Jede Zeile in SBASE.PRN beginnt mit einem Buchstaben, der die Schriftart angibt. Er ist identisch mit denen des `\s\`-Kommandos. Durch ein Leerzeichen getrennt folgt dann die Anzahl der Codes, die an den Drucker

geschickt werden und schließlich die Steuerzeichen selbst, wiederum durch Leerzeichen voneinander getrennt. Am Schluß der Zeile kann mit * beginnend noch ein Kommentar stehen.

Die Codes müssen Sie Ihrem Druckerhandbuch entnehmen. Wenn Sie sich mit Ihrem Printer gut auskennen, können Sie das \s\-Kommando sogar noch erweitern. Nach dem s folgt ja ein Zeichen, das die Schriftart identifiziert. Wenn Sie in SBASE.PRN zusätzliche Zeichen samt Codes definieren, so können Sie diese auch mit dem \s\-Kommando ausgeben. Natürlich können auch die schon vordefinierten Schriftarten anders verwendet werden, z.B. wenn Ihr Drucker keine vierfach großen Buchstaben ausgeben kann.

6.8 More

Hinter dem Button More verstecken sich einige weitere Kommandos von SBASE. Beim Anklicken erscheint eine zweite Dialogbox, in der sich entsprechende Funktions-Buttons befinden. Einige davon sind nicht anwählbar – sie sind in der aktuellen Version noch nicht implementiert und werden deshalb hier auch nicht besprochen.

Mit Import Data können Sie Daten, die nicht mit SBASE erfaßt wurden in Ihre Datei aufnehmen. Dazu müssen sie in einem bestimmten Format vorliegen. Jede Zeile stellt einen Eintrag dar, wobei die einzelnen Feldinhalte durch das Fluchtsymbol \ getrennt sind. Beim Einlesen werden die Daten entsprechend ihrer Reihenfolge in die Felder eingelesen, die Sie in Ihrer Datei definiert haben. Bei Feldern, die den Typ Unique haben, sollten Sie keine Daten einlesen, da diese Felder ansonsten nicht mehr einmalig in Ihrer Datei sind.

Wenn Sie alle Einträge, die Sie vorher mit Select ausgewählt haben, auf einen Schlag löschen wollen, benutzen Sie Delete Selection. Zur Sicherheit werden Sie noch einmal gefragt, ob Sie diese Daten wirklich löschen wollen.

Mit Undelete All werden alle Einträge, die vorher gelöscht wurden wieder in die Datei aufgenommen. Das Kommando ist nützlich, wenn Sie sich völlig vertan oder mit Delete Selection eine größere Datenmenge versehentlich gelöscht haben.

Change Selection ermöglicht es Ihnen, bei allen Einträgen der gerade aktuellen Auswahl ein Feld auf einen bestimmten Wert zu setzen. Nehmen wir z.B. an, bei einem Verein erhöht sich der Beitrag für alle Mitglieder über 18 Jahren von 50 auf 60 DM. Anstatt nun alle Einträge einzeln zu verändern, wählen Sie die Einträge mit Alter größer 18 aus und setzen dann das Feld Beitrag für diese Datensätze auf 60.

Das Kommando `Save Selection` sichert die gerade aktuelle Auswahl als eine eigene Datei. Diese kann dann wie gewohnt mit SBASE verarbeitet werden. Momentan gibt es dafür wenig Anwendungen. In einer weiteren Version von SBASE soll es jedoch möglich sein, verschiedene Dateien zu mischen oder zu vereinigen. (Dann könnte man auch fast von einer Datenbank sprechen).

6.9 Umdefinition des Datei-Aufbaus

Wie schon angesprochen, ist die Änderung der Struktur einer Datei etwas umständlich. Sie müssen dazu alle Einträge mit einem Report ausgeben, eine neue Datei einrichten und dann die Einträge wieder einlesen.

Der Report muß so gestaltet sein, daß er später mit dem "Import"-Kommando gelesen werden kann. Das heißt, alle Felder eines Datensatzes müssen in einer Zeile stehen und durch das \-Zeichen getrennt sein.

Für neu hinzukommende Felder müssen Sie eine leere Ausgabe erzeugen. Konkret heißt das, ein zusätzliches "\" zu erzeugen. Felder, die entfallen sollen, werden nicht ausgegeben.

In der Adreß-Datei, die bisher als Beispiel diente, soll das Feld "Geschlecht" wegfallen und zusätzlich vor der Ortsangabe ein Eintrag für das Land enthalten sein. Die Ausgabe der Datensätze wird von folgendem Report übernommen:

```
\f1\\\\f2\\\\f4\\\\f5\\\\\\f6\\\\f7\
```

Er gibt alle Felder bis auf das dritte und achte aus. Durch \\ wird das \ in der Ausgabe erzeugt. Das letzte Feld wird nicht mit ausgegeben, da es von Typ `unique` ist und sein Inhalt von SBASE erzeugt wird.

Nachdem Sie den Report in eine Datei gelenkt haben, stehen die Einträge als ASCII-Datei im richtigen Format auf der Diskette:

```
Minni\Maus\Gemstraße\2\\1632\Atarihausen
Moni\Monitor\Anzeigeweg\124\\6910\Bildheim
Karl\Keyboard\Tastenstraße\13\\9120\Qwertystadt
Franz\Floppy\Schlappscheibenweg\72\\3140\Diskettenburg
Gisela\Gem\Romstraße\92\\3103\Datenbus
Martin\Menu\Klappstraße\1\\2312\Dropdownstadt
Paula\Pixel\Linienstraße\400\\1240\Monitorburg
Ivan\Icon\Dialogweg\12\\1422\Resourcenhausen
```

Zwischen der Straßennummer und der Postleitzahl befindet sich jeweils leeres Feld, das in das neu zu definierende "Land"-Feld eingelesen wird.

Sie müssen jetzt noch die Neudefinition vornehmen, und `Import` die Daten einlesen. Damit ist die Datei-Struktur geändert.

Sie müssen dieses Vorgehen jeweils den Änderungen anpassen, die Sie vornehmen wollen. Achten Sie besonders darauf, nur die Felder, die übernommen werden sollen, auszugeben und für neue Felder eine leere Ausgabe zu erzeugen.

6.10 Speicherplatzbedarf

Da SBASE alle Daten im RAM hält, ist die Anzahl der Einträge begrenzt. Allerdings läßt sich einfach berechnen, wieviele Datensätze in Ihre Datei hineinpassen. Dazu müssen Sie für jedes Feld in Ihrer Datei-Definition dessen maximale Länge plus fünf Bytes nehmen. Für jeden vorhandenen Eintrag werden zusätzlich zehn Bytes benötigt. In der normalen Version von SBASE sind 500000 Bytes für Daten reserviert (bei der Spezial-Version `SBASE520` für alte 520 ST-Rechner mit nur einem halben Megabyte sind dies 100000).
Bei unserem Beispiel der Adreß-Datei bedeutet das:

Summe der maximalen Feldlängen	82 Bytes
Zusätzlicher Bedarf für 8 Felder	40 Bytes
Zusätzlicher Bedarf pro Datensatz	10 Bytes
	132 Bytes

Mit 500000 verfügbaren Bytes hätten also 3787 Adressen in der Datei Platz.

6.11 Fehlermeldungen

SBASE hat nur sehr wenige Fehlermeldungen. Die erste tritt auf, wenn die Datei `SBASE.RSC` nicht gefunden wurde: `No Resource file available`. Kopieren Sie dann die Datei in den Ordner, von dem aus Sie SBASE starten.

Wenn beim Programmstart zuwenig Speicher zur Verfügung steht, meldet SBASE `Not enough memory`. Ihnen bleibt dann nichts anderes übrig, als eine eventuell vorhandene RAM-Disk oder speicherfressende Accessories zu entfernen.

Falls während der Dateneingabe der Speicher ausgeht, erscheint `Modula 2 Runtime error #9`. Ihre Daten sind dann verloren. Sie können diese Situation umgehen, indem Sie vorher wie oben gezeigt ausrechnen, wieviele Datensätze maximal eingegeben werden können. Falls Sie mehr brauchen, müssen Sie eben weniger Felder verwenden oder deren Länge vermindern.

Chapter 7

Grafik mit Simple-Draw

Der Atari ST ist mit seiner hervorragenden Bildschirmauflösung und Farbdarstellung für Grafik ideal geeignet. In diesem Kapitel lernen Sie ein Malprogramm kennen, das sowohl für den Schwarz-Weiß-Bildschirm als auch für Farbdarstellung geeignet ist. Malen mit der Maus ist keine leichte Angelegenheit, wenn man künstlerisch tätig werden will. Der Einsatz zusätzlicher Hardware, wie Tablett, Lichtgriffel oder Scanner kann die Qualität erheblich steigern. Für das Erstellen von Diagrammen oder Schemazeichnungen reicht die Maus aber völlig aus.

Das Shareware-Programm Simple-Draw von Pim Coenradie aus den Niederlanden läuft in einer GEM-Umgebung und ist vollständig mit der Maus zu bedienen. Die Bilddateien sind kompatibel mit dem Doodle- und dem Neochrome-Format, in denen die meisten Bilder auf dem Atari ST gespeichert werden. Für die Weiterverarbeitung der Zeichnungen in einer Textverarbeitung kann ein anderes Format notwendig sein. Das Programm WordPlus z.B. enthält ein Accessory namens `SNAPSHOT`, mit dem ein Bildschirmausschnitt im 1st WordPlus-Format gespeichert werden kann. Es erlaubt Ihnen mit Simple-Draw erstellte Grafiken in Texte einzubinden.

7.1 Mausbenutzung

Die Maus dient zur Auswahl in den Menüs und Dialogboxen. Beim Zeichnen zeigen Sie mit ihr auf eine Bildschirmkoordinate und bestätigen mit einem der beiden Mausknöpfe. Wenn eine Zeichenoperation von Simple-Draw zwei Koordinaten benötigt, also z.B. Anfangs- und Endpunkt einer Linie, setzen Sie das erste Koordinatenpaar mit dem linken, das zweite mit dem rechten Mausknopf. Falls ein Kommando auch noch einen dritten

Bezugspunkt benötigt, ist wieder der linker Mausknopf an der Reihe.

Nach dem Programmstart und Bestätigung einer Anfangsmeldung erscheint auf dem Bildschirm ein Begrüßungsbild. Es wird jedesmal automatisch geladen, was auf Dauer unpraktisch sein kann. Um dies zu umgehen, müssen Sie nur ein leeres Bild erzeugen und auf Diskette unter dem Namen `TITEL.DAT` speichern.

Die erste Bildschirmzeile wird von der Menüleiste eingenommen, die zur Auswahl der Simple-Draw Kommandos dient. Ich stelle Ihnen die Menüs der Reihe nach vor – wie Sie sehen werden ist die Bedienung des Programms ein Kinderspiel.

7.2 Das `File`-Menü

Alle Einträge im `File`-Menü beziehen sich auf das momentan dargestellte Bild. Der erste Menüpunkt `Laden` dient zum Einlesen eines Bildes. Sie brauchen dazu nur in einer File-Select-Box, die Sie auch von anderen GEM-Programmen her kennen, den Namen auswählen. Da das aktuelle Bild dabei natürlich überschrieben wird, fragt Simple-Draw nach, ob Sie das auch wirklich wollen. Falls ja, müssen Sie den Knopf `Ja` anklicken.

Das Gegenstück dazu ist der nächste Punkt, `Save as`. Er dient zum Speichern des gerade bearbeiteten Bildes. Wieder müssen Sie in einer File-Select-Box den Datei-Namen ausgewählen, wobei Sie mit der File-Extension das Dateiformat bestimmen.

Mit `.SIM` oder `.DOO` wird es im Doodle-Format auf Diskette geschrieben. Dabei handelt es sich um das Bit-Muster im Bildspeicher mit 32 000 Bytes. Für Farbbilder ist das Degas-Format besser, da die Farbauswahl mitgespeichert wird. Verwenden Sie `.PI2` für Bilder in der mittleren Ausflösung mit vier Farben und `.PI3` bei der niedrigen Ausflösung mit 16 verschiedenen Farben. Diese Dateien belegen 32 034 bzw. 32 128 Bytes auf der Diskette.

Bei Speichern kann die Fehlermeldung `Sorry, deze disk is vol` erscheinen. Auf Ihrer Diskette ist dann nicht mehr genug Platz, um das Bild zu schreiben. Verwenden Sie eine andere Diskette oder löschen Sie nicht mehr benötigte Dateien, wozu Sie allerdings Simple-Draw verlassen müssen.

Das nächste Kommando werden Sie als erstes verwenden, denn es dient zum Löschen des Bildes im Speicher. Wählen Sie `Nieuw` aus und bestätigen mit dem `OK`-Knopf, daß Sie das Bild wirklich löschen wollen. Es ist dann für immer verloren, es sein denn, Sie hattes es bereits abgespeichert.

Auf diesem Weg können Sie auch ein leeres Begrüßungsbild erstellen. Sie müssen zuerst `Nieuw` wählen und dann einfach mit `Save as` unter dem Namen `TITLE.DAT` speichern.

Mit `Set Drive` können Sie festlegen, auf welchen Laufwerk Simple-Draw die Bilderdateien halten soll. Es erscheint eine Auswahlbox mit den Laufwerksbuchstaben `A:`, `B:` (beide für Diskettenlaufwerke) und `C:` (für eine RAM-Disk oder die Festplatte). Auf weitere Massenspeicher können Sie zugreifen, indem Sie in den File-Select-Boxen den Pfadnamen über der Datei-Liste entsprechend ändern.

Das Kommando `Stoppen` im `File`-Menü beendet Simple-Draw. Damit Sie nicht aus Versehen das gerade bearbeitete Bild verlieren, müssen Sie dieses Kommando bestätigen.

7.3 Das `Funkie`-Menü

Im Funktionsmenü wählen Sie die Zeichenoperationen aus. `Wisser` löscht kleine Flächen Ihres Bildes. Die Größe dieses Radiergummis kann unter dem Punkt `Grootte` im `Settings`-Menü eingestellt werden.

`Punk` wählt den Freihand-Zeichenmodus aus. An der Stelle, auf die der Mauspfeil zeigt, erscheint beim Drücken des linken Mausknopfes ein Punkt. Wenn Sie den Knopf gedrückt lassen, können Sie so nach Herzenslust herummalen. `Lijn` zieht eine Linie zwischen zwei Punkten. Die Anfangskoordinaten wählen Sie mit dem linken, den Endpunkt mit dem rechten Mausknopf aus. Die Dicke der Linie ist im `Settings`-Menü wählbar.

Mit `K-lijn` können Sie einen Linienzug zeichnen. Der Startpunkt des Linienzuges wird mit dem linken Mausknopf festgelegt. Die weiteren Koordinaten, an denen eine Linie aufhört und eine weitere beginnt, setzen Sie dann mit rechten Mausknopf. Wollen Sie einen neuen Linienzug beginnen, müssen Sie erneut links drücken.

Unter `Kwast` verbirgt sich ein Pinsel, dessen Größe wieder unter `Grootte` einzustellen ist. Er zeichnet ein Quadrat mit einem der Füllmuster, die ebenfalls eingestellt werden können. Wenn Sie den Mausknopf gedrückt lassen und auf dem Bildschirm hin- und herfahren können Sie einen räumlichen Effekt erreichen, für den die Schnelligkeit der Bewegung entscheidend ist.

`Waaier` läßt Sie Strahlen von einem Punkt aus ziehen. Sie legen diesen mit dem linken Mausknopf fest und können dann mit dem rechten jeweils eine Linie vom vorher ausgewählten Startpunkt aus ziehen. Halten Sie auch hier einmal den rechten Mausknopf gedrückt und fahren auf dem Bildschirm herum. So lassen sich ebenfalls interessante Effekte erzielen.

Nun zu den etwas größeren geometrischen Objekten. Mit `Box` ziehen Sie ein Rechteck auf. Die linke obere Ecke wird mit dem linken, die Ecke rechts unten mit dem rechten Mausknopf festgelegt. Das gleiche Ergebnis erreichen Sie mit `Round Box`, nur sind hierbei die Ecken des Rechtecks abgerundet. Beide Kommandos lasssen sich gut für Diagramme verwenden.

`Frame` funktioniert wie `Box`, allerdings wird die Rechteckfläche mit dem gerade gewählten Muster gefüllt, das Sie unter `Fillsoort` auswählen können. Kreise werden mit `Cirkel` gezeichnet. Den Mittelpunkt wählen Sie mit dem linken Mausknopf aus und die Größe des Kreises mit dem rechten. Sie zeigen dabei auf einen Punkt, durch den der Kreis gehen soll, so daß sein Radius dem Abstand zwischen den beiden gewählten Koordinaten entspricht.

`Tekst` dient zum Beschriften. Nach dem Drücken des linken Mausknopfs erscheint ein kleiner Cursor und Sie können Text eintippen und mit der `Backspace`-Taste löschen. `Return` schließt die Textzeile ab, worauf sie mit der Maus auf die endgültige Position geschoben werden kann. Der linke Mausknopf übernimmt den Text in das Bild an der momentanen Stelle. Das Aussehen der Buchstaben und ihre Größe können Sie im `STIJL`-Menü einstellen.

Die letzte Zeichenfunktion ist `Vullen`. Die Fläche, in die Sie mit dem Mauspfeil zeigen wird nach Drücken des linken Knopfes mit dem aktuellen Füllmuster ausgemalt. Die Fläche muß rundherum von einer Linie begrenzt sein. Ist das nicht der Fall, läuft die "Farbe" über das ganze Bild und macht Ihre Arbeit zunichte. Wenn Sie sich nicht ganz sicher sind, ob eine Fläche auch wirklich komplett begrenzt ist, sollten Sie das Bild vorher abspeichern, damit es nach einem "Unfall" zurückgeholt werden kann.

7.4 Das `Stijl`-Menü

In diesem Menü können Sie das Aussehen und die Größe des mit dem `Tekst`-Kommando erzeugten Textes einstellen.

Die ersten sechs Menüeinträge legen die Schriftart fest. `Normaal` ist die Grundschrift. Mit dieser Auswahl werden alle anderen Einstellungen gelöscht. Die weiteren Menüpunkte schalten jeweils ein Schriftattribut hinzu:

Menüpunkt	Schriftart
`Vet`	Fettschrift
`Licht`	Hellschrift
`Schuin`	Kursivschrift
`Onderstreept`	Unterstreichung
`Outlined`	Konturschrift

Wenn Sie fette und kursive Schrift verwenden wollen, müssen Sie `Vet` und `Schuin` auswählen. Falls Sie dann auf normale unterstrichene Buchstaben wechseln wollen, müssen zunächst die anderen Einstellungen mit `Normaal`

gelöscht und dann Onderstreept gewählt werden. Die Häkchen neben den Menüpunkten zeigen an, welche Schriftattribute gerade eingeschaltet sind.

Die Schriftgröße können Sie von 4 bis 26 Punkten in mehreren Schritten einstellen. Mit Punkten sind dabei keineswegs Bildschirmpixel gemeint, vielmehr handelt es sich um ein drucktechnisches Größenmaß. Die 4-Punkt-Schrift entspricht bei einem S/W-Bildschirm der kleinen Systemschrift, wie sie unter den Desktop-Icons zu finden ist. Hinter 13 Punkt verbirgt sich der normale Systemfont, mit dem auf z.B. die Menüzeile dargestellt wird.

Nach einer Auswahl im Stijl-Menü verhält sich das Programm so, als wenn Teckst benutzt wäre – Sie können also sofort mit der Textpositionierung und -eingabe beginnen.

7.5 Das Settings-Menü

Im Settings-Menü legen Sie Größe und Aussehen für Linien und den Pinsel fest. Die ersten vier Menüpunkte dienen zur Festlegung verschiedener Linienarten. Eine Auswahl von Recht erzeugt normale, durchgezogene Linien. Mit Gestreept werden gestrichelte Linien gezeichnet, wobei die Leerräume so groß wie die Striche sind. Bei Gestrippelt werden sie länger, so daß die Linie z.B. für Hilfslinien in Schemazeichnungen geeignet ist. Bei der Einstellung Morse malt Simple-Draw eine Punkt-Strich-Punkt Linie.

Die Stärke der Linien können Sie in den Menüeinträgen 1 mm bis 12 mm festlegen. Dabei hat Pim Coenradie anscheinend nicht richtig nachgemessen – die tatsächlich Dicke der Striche ist ungefähr halb so groß wie ausgewählt. Am besten probieren Sie einmal alle Linienstärken durch, dann finden Sie immer sofort das Gewünschte.

Die Einstellungen wirken auf mehrere Zeichenoperationen. Beeinflußt werden Linien, Rechtecke und Kreise.

Unter Grootte können Sie die Größe von Radiergummi und Pinsel festlegen. Es erscheint eine Alertbox, in der die Auswahl zwischen den Größen 2, 4 und 10 besteht. Radiergummi und Pinsel haben danach die Ausmaße 5×5, 13×13 bzw 21×21 Pixel.

7.6 Das Fills-Menü

Füllmuster werden beim Waaier-, Frame- und Vullen-Befehl benutzt. Eines der 36 eingebauten Füllmuster und Schraffuren oder ein selbstdefiniertes Muster können Sie mit Fill-keuze wählen. In einer Dialogbox wird das momentan angewählte angezeigt und mit den Buttons < und > schalten Sie

auf das vorherige oder nächste um. Nach Bestätigung mit `Klaar` befindet sich Simple-Draw im `Vullen`-Modus.

Das genannte selbstdefinierbare Muster können Sie mit `Fill-editor` bearbeiten. Die Menüleiste wechselt und auf dem Bildschirm erscheint das 16×16-Raster des Musters. Darin können Sie einen Punkt mit der linken Maustaste setzen, mit der rechten wird er gelöscht. Auf der linken Bildschirmseite zeigt Simple-Draw an, wie das Muster in Originalgröße aussieht.

Im `File`-Menü kann mit `Laden` ein Füllmuster von Diskette eingelesen werden. Bei Simple-Draw sind in dem Ordner `Fill` schon 22 weitere Muster vorhanden, die teilweise recht interessante Effekte erzielen. Wenn Sie selber einen guten Entwurf gemacht haben, können Sie ihn mit `Save as` abspeichern. Für den Datei-Namen, den Sie in der File-Select-Box festlegen, müssen Sie die Endung `.FIL` benutzen.

`Stoppen` schließlich beendet den Muster-Editor und Sie haben wieder das gewohnte Bild vor sich und können jetzt das neue Muster benutzen.

Im `Funktie` finden Sie einige komfortable Funktionen zum Gestalten des Musters. `Edit` bringt Sie in den Editier-Modus zurück, bei dem ein Fadenkreuz als Mauszeiger erscheint und Sie Punkte setzen und löschen können.

`Toggle` "dreht" alle Punkt um. Aus weißen Punkten werden schwarze und umgekehrt. Falls Sie ein Muster aus dem geladenen Bild übernehmen wollen, können Sie das mit `Pak`. Die Anzeige schaltet auf das bearbeitete Bild um und Sie können mit einem kleinen Rechteck einen Ausschnitt in das Muster kopieren. Dazu bestätigen Sie die Übernahmen mit der linken Maustaste.

Mit `Flip` stellen Sie das Muster durch Spiegeln an der senkrechten Achse auf den Kopf. In horizontaler Richtung arbeitet `Invert`: Linke und rechte Seite des Musters werden vertauscht.

Am Beginn eines Musterentwurfs können Sie mit `Vullen` und `Wis` die gesamte Fläche schwarz bzw. weiß einfärben. Je nachdem, ob Ihr Muster eher dunkel oder hell sein soll, ersparen Sie sich damit eine Menge Arbeit.

7.7 Das `Etc.`-Menü

Unter diesem letzten Menü verbergen sich noch einige interessante Funktionen. Mit `Block laden` können Sie einen Ausschnitt aus einem anderen Bild in das aktuelle einfügen. Nachdem Sie in einer File-Select-Box seinen Datei-Namen ausgewählt haben "kleben" Sie ihn mit dem linken Mausknopf in Ihr Bild.

Das Gegenstück dazu ist `Block saven` zum Abspeichern. Mit der linken Maustaste müssen Sie die linke obere Ecke des Ausschnitts festlegen, mit

der rechten die rechts untere. Ein solcher Block kann dann in andere Bilder übernommen werden. Sie könnten sich so eine Bibliothek von viel verwendeten Symbolen oder Zeichnungen anlegen.

`Copieer` kopiert einen Ausschnitt an eine andere Stelle im aktuellen Bild. Die Festlegung des Bildteils geschieht wie gerade beschrieben. Das Kopieren an die Stelle, auf die der Mauspfeil zeigt, lösen Sie mit dem linken Mausknopf aus. Simple-Draw kopiert den Ausschnitt so, daß seine linke obere Ecke an die Position des Mauszeigers kommt.

Ähnlich arbeitet `Verplaats`, nur wird der Ausschnitt dabei verschoben. An seiner ursprünglichen Position verbleibt ein weißes Rechteck. Nützlich, wenn Sie z.B. ein Diagramm auf dem Bild verschieben wollen; bei richtigen Bildern ist er nicht so praktisch.

Für das Löschen größerer Flächen brauchen Sie sich nicht den kleinen Radiergummi unter `Wisser` abzumühen. Mit `Wis Block` legen Sie wie bei den letzten Befehlen einen Ausschnitt fest, nur wird er nicht kopiert oder verschoben, sondern gelöscht.

Bei Diagrammen ist es nützlich, ein Hilfgitter zu haben, an dem Sie Linien und Rechtecke orientieren können. Mit `Rooster` läßt sich ein solches Raster über die Zeichnung legen. In einer Alert-Box haben Sie die Auswahl zwischen einem 10-, 20- und 30-Pixel Abstand zwischen den Gitterpunkten. Leider läßt sich das Raster nicht wieder entfernen, auch kann Simple-Draw Linien oder Rechtecke nicht automatisch an die Gitterpunkte ziehen werden. Trotzdem ist dieses Programmfeature eine kleine Hilfe.

Mit `Set colors` können Sie die Farbeinstellung für das Bild verändern. Bei einem Schwarz-Weiß Monitor bewirkt das Kommando lediglich ein Invertieren des Bildschirms, beim Farbmodus erscheint eine Dialogbox. In ihr können Sie mit dem linken Mausknopf die Farben für Linien, Rechtecke, Kreisen, Füllmustern und Text verändern. Klicken Sie einfach in den entsprechenden Farbbalken und Simple-Draw schaltet eine Farbe weiter.

Für Detailarbeiten haben Sie mit `Loep` eine Lupe zur Verfügung. Sie müssen den zu vergrößernden Bereich in einem kleinen Rechteck mit der linken Maustaste festlegen. Simple-Draw zeigt nun ein Raster an, in dem die Bildpunkte vergrößert dargestellt werden. Mit der linken Maustaste setzen oder löschen dort einzelne Pixel. Im oberen Bildschirmteil haben Sie die Auswahl zwischen mehreren Pixelfarben, wobei für Schwarz-Weiß Monitore nur die ersten beiden interessant sind, in der mittleren Auflösung können Sie vier Farben verwenden. Rechts oben zeigt Simple-Draw den Ausschnitt in Originalgröße und die Veränderungen an. Der Lupen-Modus wird mit dem Button `OK` verlassen.

Das letzte Kommando von Simple-Draw ist `Hardcopy`. Nach einer Bestätigung wird das Bild ausgedruckt, wozu das Programm die eingebaute Hardcopy-Routine des Atari benutzt. Falls Sie eine Anpassungen z.B. für

24-Nadel-Drucker oder Laserprinter geladen haben, kann sie problemlos verwendet werden.

Sie können alle wichtigen Funktionen mit dem Programm ausführen, die zum Zeichnen von Bildern, Diagrammen und Schemazeichnungen notwendig sind. Wenn Sie weitere Features einbauen wollen, haben Sie in der Datei `SIM_DRAW.BAS` den Sourcetext in GfA-Basic zur Verfügung. Allerdings handelt es sich dabei um die Quelle einer älteren Version, die noch nicht alle Funktionen enthält. Ach ja: Die Programmversion und Anschrift des Autors können Sie im `Desk`-Menü unter `Over Simple Draw` erfahren.

Chapter 8

Tabellenkalkulation mit STCALC

Tabellenkalkulation ist eine der faszinierendsten Anwendungen auf Microcomputern. Kommerzielle Calc-Programme (im Englischen Spreadsheet genannt) sind zum Teil vollständige eigene Programmiersysteme und leider auch sehr teuer. STCALC ist ein einfaches Spreadsheet auf Public-Domain-Basis.

Mit einer Tabellenkalkulation haben Sie ein Rechenblatt vor sich, das in Zellen aufgeteilt ist. In jeder Zelle können Werte, Formeln oder Zeichenketten stehen. Aufgabe des Programms ist es, die Zellen auszuwerten und die Ergebnisse der Formeln anzuzeigen. Sie haben so die Möglichkeit komplizierte Berechnungen und mathematische Modelle einfach darzustellen. Die Anwendungen reichen von simplen Mehrwertsteuerrechnungen bis hin zu kompletten Buchführungen. STCALC bietet 7500 Zellen und arithmetische Basisoperationen sowie eine Summenfunktion.

8.1 Cursor-Bewegungen

STCALC zeigt Ihnen nach dem Programmstart ein leeres Arbeitsblatt, auf dem sich der Zellencursor links oben befindet. Eingaben beziehen sich immer auf die Zelle, in der dieser Cursor steht. Sie können ihn mit den Pfeiltasten durch das Rechenblatt bewegen, wobei eine Kombination der Cursor-Tasten mit der Shift-Taste größere Bewegungen erlaubt: In der Vertikalen wird jeweils um 20 Zeilen nach oben bzw. unten gescrollt; in der Horizontalen wird der Ausschnitt des Rechenblatts um soviele Spalten verschoben, wie momentan auf dem Bildschirm sichtbar sind.

Bei STCALC müssen Sie jede Aktion durch ein Kommando einleiten. Immer wenn in der untersten Bildschirmzeile `COMMAND:` erscheint, können Sie einen Kommandobuchstaben eingeben. Der erste Befehl bietet eine weitere Möglichkeit, den Zellencursor zu bewegen. Nach der Eingabe von `G` oder `>`, fragt STCALC nach der gewünschten Zeile (`Row`) und Spalte (`Column`) und setzt den Zellencursor entsprechend der Eingabe. Da der Bildschirmaufbau leider nicht der schnellste ist, arbeitet das Goto-Kommando schneller als das Blättern im Spreadsheet mit den Pfeiltasten.

8.2 Zellenbezeichnungen

Bevor ich zu den anderen Kommandos komme, müssen Sie zunächst wissen, wie man eine Zelle bezeichnet. Eine Zelle wird durch die Angabe ihrer Zeile und Spalte beschrieben, wobei das Rechenblatt senk- und waagerecht von 1 an durchnummeriert ist. Die Zeilenangabe, die vor der Spaltenangabe kommen muß, wird durch ein `r` eingeleitet. Darauf folgt mit `c` beginnend die Spaltennummer. So wäre mit `r15c3` die Zelle in Zeile 15 und Spalte 3 gemeint.

Weiterhin sind relative Zellenbeschreibungen möglich. Dabei müssen Sie jeweils ein Wert angeben, der zu der Zelle, in der sich eine Formel befindet, addiert oder substrahiert wird. Diese Werte müssen in eckige Klammern eingefaßt werden. Steht in der Zelle r10c5 die Angabe `r[5]c[-3]`, ist damit der Wert der Zelle r15c2 gemeint.

8.3 Formeln

Mathematische Modelle bestehen aus Formeln. Für die Berechnung einer Mehrwertsteuer wäre die Formel "multipliziere Nettowert mit 1,14" nötig. In einer Tabellenkalkulation werden die Formeln in Zellen eingetragen. Die in ihnen verwendeten Werte stehen ebenfalls in Zellen. Also müßte in der Zelle, in der ein Bruttopreis errechnet wird, die Formel "Nettowert*1.14" stehen, wobei Sie anstatt des Namens "Nettowert" die Bezeichnung der Zelle, in der der Wert zu finden ist, einsetzen müssen.

Einen Wert oder eine Formel können Sie mit dem Value-Befehl in die aktuelle Zelle eintragen. Nachdem Sie `V` gedrückt haben, verlangt das STCALC eine Eingabe in der untersten Bildschirmzeile. Bei der Eingabe von Kostanten steht Ihnen ein Wertebereich von 10E-38 bis 10E+38 zur Verfügung. STCALC rechnet mit 7-stelliger Genauigkeit und interpretiert grundsätzlich alle Zahlen und Formeln als Fließkommazahlen interpretiert werden.

Für die Formeln stehen Ihnen die Grundrechenarten +, -, * und / zur Verfügung. Bei der Formelauswertung beachtet STCALC die Punkt-vor-Strich Regel. Wollen Sie die Formel anders auswerten, können Sie Teile des Ausdrucks klammern. Die Sonderfunktion `@sum` ermöglicht es, Teile des Rechenblatts aufzusummieren. Dazu wird der Funktion eine Bereichsdefinition als Argument gegeben.

Dieser Bereich kann eine Spalte, Zeile oder eine Fläche von Zellen sein. Sie müssen eine Anfangs- und Endzelle durch das `:`-Zeichen getrennt angeben. Soll die dritte Spalte bis zur 20. Zeile summiert werden, lautet die Funktion `@sum(r1c3:r20c3)`. Wollen Sie hingegen die Summe aller Felder in den Zeilen 5 bis 7 und den Spalten 4 bis 6, so schreiben Sie `@sum(r5c4:r7c6)`. Bei den Zellendefinitionen sind natürlich auch relative Angaben erlaubt.

Falls sich bei der Berechnung einer Formel Fehler ergeben sollten, z.B. Division durch Null, Ansprechen einer Zelle, die außerhalb des Rechenblatts liegt oder keinen Wert enthält, so erzeugt STCALC keine Fehlermeldung; vielmehr nimmt das Programm den Wert Null als Ergebnis an. Dieses Verfahren ist keineswegs ein Lapsus, sondern erlaubt auch einige Tricks. Wollen Sie z.B. erfahren, ob in der Zelle r5c7 ein Wert steht, so können Sie die Formel `r5c7/r5c7` verwenden. Ist sie leer, so würde eine Division durch Null entstehen, und STCALC errechnet als Ergebnis 0. Enthält sie einen Wert ungleich Null, so ergibt sich eine 1.

Auf diese Art kann man z.B. eine Durchschnittsberechnung durchführen. Angenommen, Sie benötigen den durchschnittlichen Wert der Spalte 5. Sie müssen also die Summe der Werte in der Spalte durch die Anzahl der vorhandenen Werte teilen. Die Summenbildung kann durch die `@sum`-Funktion problemlos geschehen. Um die Anzahl der vorhandenen Werte zu erhalten, schreiben Sie in alle Zellen der Spalte 6 jeweils die Formel `r[0]c[-1]/r[0]c[-1]`, und überprüfen so, ob in der jeweils direkt links daneben liegenden Zelle ein Wert steht. Nun brauchen Sie nur noch die Summe über die Spalte 6 bilden und erhalten die Anzahl der vorhandenen Werte für die Durchschnittsbildung. Sie sehen, daß viele in kommerziellen Programmen vorhandene Standardfunktionen einfach nachgebildet werden können.

Jedesmal, wenn Sie einen Wert eingegeben haben, berechnet STCALC das Arbeitsblatt neu, und zwar beginnend ab der aktuellen Cursor-Position in allen Zellen nach links und nach unten. Da die automatische Neuberechnung abgeschaltet werden kann läßt sie sich auch manuell mit dem Kommando ! auslösen.

8.4 Weitere Kommandos

Der Label-Befehl, dient zur die Eingabe einer Zeichenkette und wird durch die `L`-Taste aufgerufen. Sie können dann am unteren Bildschirmrand eine Zeichenkette eingeben, die in die aktuelle Zelle übernommen wird. Da STCALC eine sehr primitive Eingaberoutine benutzt, die leider keine Leerzeichen akzeptiert, müssen Sie stattdessen andere Zeichen verwenden, wobei sich z.B. der Unterstrich _ anbietet. Tippen Sie ein Leerzeichen, so wird die Eingabe abgebrochen und das Programm befindet sich wieder im Kommandomodus.

Wollen Sie einen Wert oder eine Formel z.B. in einer Spalte in alle Zeilen schreiben, so ersparen Sie sich mit dem Copy-Befehl (Taste `C`) viel Schreibarbeit. STCALC fragt nach, ob Sie den Inhalt der aktuellen Zelle nach rechts (`R`) oder unten (`D`) kopieren wollen. Sie müssen dann noch angeben, bis zu welcher Zeile oder Spalte Sie übertragen wollen und schon wird der Zelleninhalt vervielfacht. Als Beispiel soll in der Spalte 1 der Wert 50 in den Zeilen 10 bis 20 erscheinen. Gehen Sie mit dem Zellencursor nach r1c10 und geben Sie den Wert 50 ein. Rufen Sie nun das Copy-Kommando auf und antworten Sie `D` für nach unten kopieren und 20 für die unterste Zeile.

Auf gleiche Weise funktioniert der Lösche-Befehl (Taste `B` für Blank). Sie müssen ebenfalls die Löschrichtung und den Löschbereich eingeben. STCALC nimmt nun aus den angegebenen Zellen alle Eintragungen heraus. Um die Eingaben nach dem obigen Beispiel wieder zu entfernen, müssen Sie die gleichen Angaben machen, nur heißt der Befehl diesmal `B`.

Falls das ganze Spreadsheet gelöscht werden soll, kann der `N`-Befehl ("New" für "Neu") benutzt werden. Da diese Aktion eventuell unabsichtlich aufgerufen werden könnte, fragt STCALC sicherheitshalber in einer Alert-Box nach, ob es auch wirklich löschen soll.

Für das Speichern, Laden und Ausdrucken eines Arbeitsblatts ist der `F`-Befehl (File) zuständig. Es erscheint dann eine weitere Auswahl, nämlich Save für sichern, Load für laden und Print für ausdrucken eines Arbeitsblattes. Bei den beiden ersten Optionen (Tasten `S` und `L`) werden Sie nach einem Datei-Namen für den Spreadsheet-Inhalt gefragt. STCALC verwendet für seine Dateien die Endung `.STC`, die Sie nicht mit eingeben brauchen.

Beim Ausdruck können Sie auch Teile des Rechenblatts an den Drucker schicken. Sie werden nacheinander nach den Eckzellen des jeweiligen Ausschnitts gefragt, d.h. nach der obersten Zeile und Spalte, bei denen der Ausdruck beginnen soll, sowie nach der untersten Zeile und der Endspalte. Soll das ganze Arbeitsblatt gedruckt werden, so müssen Sie den Ausschnitt entsprechend groß angeben.

Wollen Sie nachträglich eine Spalte oder Zeile in das Spreadsheet einfügen,

müssen Sie den Insert-Befehl `I` (wie "Einfügen") benutzen. STCALC fragt nach, ob Zeilen oder Spalten hinzukommen sollen (`Rows or Columns`). Nach der Auswahl mit `R` oder `C` müssen Sie bestimmen, vor welcher Zeile oder Spalte eingefügt werden soll und schließlich wieviele Zeilen oder Spalten STCALC neu einrichten soll. Nach Ihren Angaben werden die Spalten rechts bzw. die Zeilen unter den neuen verschoben und Sie können neue Formeln eingeben. Dabei ist zu beachten, daß die Angaben in schon vorhandenen Formeln nicht umberechnet werden. Dazu ein Beispiel, das Sie am Bildschirm ausprobieren sollten:

Angenommen, in r3c1 steht der Wert 43.21, in r1c1 die Formel `r[+2]c[0]*1.14` und in r1c2 `r3c1*0.95`. Nun wollen Sie nach Zeile eins eine weitere einfügen. STCALC verschiebt also alle Zeilen ab der zweiten Zeile um eins nach unten und damit insbesondere die 43.21. Die beiden Formeln liefern nun falsche Ergebnisse: In der Zelle r1c1 stimmt die relative Angabe nicht, da er nun ja +3 lauten müßte. Auch in r1c2 ist nun eine "falsche" Formel vorhanden, weil der Wert nun in r4c1 steht. Sie müssen also vor dem Einfügen diese Probleme berücksichtigen !

Das Gegenstück zum `I`-Befehl ist Delete (`D`). Sie müssen die gleichen Angaben machen, nur werden jetzt die angegebenen Spalten oder Zeilen gelöscht und alle anderen entsprechend verschoben. Auch hierbei ergeben sich dabei die oben beschriebenen Probleme mit Formeln.

STCALC beinhaltet einige Voreinstellungen, die Sie mit dem Set-Kommando verändern können. Nach dem Drücken von `S` erscheint eine kleine Menüzeile. Mit `C` für Column-Width läßt sich festlegen, wie breit das Programm einzelne Spalten darstellt. Sie werden gefragt, auf welchen Spaltenbereich sich der neue Wert beziehen soll; wobei STCALC sich zunächst nach der Anfangs- und dann nach der Endspalte erkundigt. Nach der Eingabe der gewünschten Spaltenbreite wird der Bildschirm neu geschrieben. Beim Programmstart ist übrigens automatisch ein Wert von 10 Zeichen eingestellt.

Mit der Option `D` legen Sie fest, wieviele Stellen nach dem Komma bei Zahlen angezeigt werden sollen (zwei Nachkommastellen ist die Standardeinstellung). Die Abfrage für einen Spaltenbereich läuft wie oben beschrieben ab. Bedenken Sie, daß die Werte auch in die Spaltenbreite passen müssen. Zu den Nachkommastellen kommen noch die Vorkommastellen, der Dezimalpunkt und eine Stelle für ein eventuell negatives Vorzeichen. Kann STCALC einen Wert nicht darstellen, zeigt es dies durch `*`-Zeichen in der Zelle an.

Mit `R` für "Recalc" wird definiert, ob nach der Eingabe eines Wertes automatisch das Arbeitsblatt ab dem Zellencursor neu berechnet wird. Die aktuelle Auswahl "Automatisch" (beim Programmstart vorgegeben) oder "Manuell" (also durch das !-Kommando) wird rechts oben am Bildschirm neben der Anzeige des noch freien Speicherplatzes vermerkt. Wenn Sie

Tabellen eingeben und Berechnungen erst dann sinnvoll sind, wenn alle Werte bekannt sind, sollten Sie aus Zeitgründen auf manuell schalten.

Die letzte Option in dem kleinen Menü legt fest, in welcher Reihenfolge das Spreadsheet neu berechnet werden soll (`O`). Bei der Auswahl `R` wird zeilenweise vorgegangen, d.h. zunächst werden alle Zellen der ersten Zeile (r1c1...r1c50) und dann die der nächsten (r2c1...r2c50) ausgewertet. Beim spaltenweisen Vorgehen wird erst r1c1 bis r150c1 und dann r1c2 bis r150c2 errechnet. Jenachdem, wie Sie Ihre Formeln auf dem Rechenblatt plazieren, kann dies zu unterschiedlichen Ergebnissen führen. Am rechten oberen Bildschirm befindet sich eine kleine Anzeige, in der das `!`-Zeichen für spalten- und das `--`Zeichen für zeilenweise Auswertung steht, wobei ersteres beim Programmstart eingestellt wird.

Zur optischen Gestaltung der Anzeige steht der Title-Befehl (`T`) zur Verfügung. Er hat die Aufgabe, einen Teil der Zellen auf dem Bildschirm "einzufrieren", so daß sie trotz Scrollen noch sichtbar sind. Sie können somit z.B. Titelzeilen für Spalten ständig anzeigen. Angenommen, die ersten beiden Zeilen enthalten solche Titel (die mit dem L-Kommando eingegeben wurden). Sie müssen nun den Cursor in die zweite Zeile - die die letzte ist, die eingefroren werden soll - bewegen. Nach Eingabe von `T` erscheint eine Menüzeile. Wählen Sie dort `H` für horizontal aus. STCALC friert nun die ersten beiden Zeilen ein und setzt den Cursor in die dritte. Wenn Sie nach unten scrollen, bleiben die Titelzeilen sichtbar.

Durch Auswahl von `V` für vertikal können Sie Spalten einfrieren und mit `B` für beides sowohl Titel- als auch Randzellen. Wenn die Anzeige sich wieder normal verhalten soll, lassen sich mit `N` für "none" (keiner) diese Einstellungen wieder aufheben. Alle Voreinstellungen, also alle Definitionen durch das Set- und das Title-Kommando, werden von STCALC nicht gespeichert. Wenn Sie ein Arbeitsblatt laden, gelten (leider) wieder die Voreinstellungen.

Alle Kommandoeingabe und -nachfragen können mit der `Esc`-Taste abgebrochen werden. Falls Sie eine Auswahl treffen, die nicht vorgesehen ist, erscheint jeweils eine kleine Fehlermeldung und Sie befinden sich wieder im Kommandomodus. Verlassen wird STCALC durch `Q` für Quit (Verlassen). In einer Alertbox erscheint noch eine kleine Nachfrage, ob Sie das Programm auch wirklich beenden wollen. Dies ist wieder eine Sicherheitsmaßnahme gegen unbeabsichtigte Auswahl des `Q`-Kommandos, bei dem

natürlich das aktuelle Spreadsheet verloren geht.

8.5 Übersicht

Zum Nachschlagen sind hier alle STCALC-Kommandos tabellarisch aufgelistet:

Kommando	bewirkt
`G`	Zelle direkt anspringen
`>`	Zelle direkt anspringen
`V`	Formel in Zelle eintragen
`L`	Zeichenkette in Zelle eintragen
`!`	Rechenblatt neuberechnen
`C`	Bereich kopieren
`B`	Bereich löschen
`N`	Rechenblatt löschen
`I`	Zeilen oder Spalten einfügen
`D`	Zeilen oder Spalten entfernen
`F`	Datei-Funktionen
	`L` Rechenblatt laden
	`S` Rechenblatt abspeichern
	`P` Rechenblatt ausdrucken
`S`	Optionen setzen
	`C` Spaltenbreite festlegen
	`D` Anzahl der Nachkommastellen festlegen
	`R` Automatische oder manuelle Neuberechnung festlegen
	`O` Richtung bei der Neuberechnung festlegen
`T`	Titelzeile einfrieren
	`H` Zeilen einfrieren
	`V` Spalten einfrieren
	`N` Einstellungen aufheben
`Q`	STCALC verlassen

Chapter 9

Kommunikation mit UniTerm

Telekommunikation ist eine Computer-Anwendung, die in den nächsten Jahren immer größere Bedeutung bekommen wird. Mit Datenfernübertragung (abgekürzt "DFÜ") können Sie mit anderen Computern kommunizieren und somit an Mailboxen teilnehmen oder Datenbankangebote nutzen.

Eine Mailbox ist eine Mischung aus elektronischem Briefkasten und computerisiertem Diskussionsforum. Auf dem Mailbox-Rechner läuft ein Programm, das diese Dienste anbietet. Sie rufen über ein Modem den Rechner an, das dort angeschlossene Modem nimmt den Anruf an und startet das Mailboxprogramm. Sie können dann mit verschiedenen Kommandos Informationen abrufen, Briefe verschicken oder sich in den Mitteilungsbrettern äußern.

Die meisten Mailboxen, die es momentan gibt, wurden von Computer-Freaks für Computer-Freaks eingerichtet. Logischerweise sind daher die Hauptthemen der Umgang mit Kleinrechnern, mit Software oder allgemeine Statements zum gemeinsamen Hobby. In guten Mailboxen finden Sie ungeheuer viele Informationen und können direkt nachfragen, falls etwas unklar sein sollte.

Kommerzielle Mailboxen bieten zumeist Datenbanken an. Gegen einen Benutzungsbeitrag haben Sie Zugriff auf riesige Informationsmengen.

In den USA ist DFÜ eine völlig alltägliche Sache, die unter Computer-Freaks sehr verbreitet ist. In der Bundesrepublik behindert das Postmonopol die Ausbreitung dieser Technik, da alle benutzten Geräte zugelassen sein müssen. Das bedeutet insbesondere, daß die Einrichtung einer Mailbox teuer wird, da ausländische Billigproduzenten praktisch keine Chance

auf eine Zulassungsnummer haben. Die Bundespost setzt hierfür Kriterien an, die technisch zumeist unsinnig sind, und mit Befürchtungen über Störungen im Telefonnetz begründet werden.

Viele Betreiber von Mailboxen umgehen daher die Bestimmungen und schließen einfach ein Import-Modem an ihren Telefonanschluß an, was technisch kein Problem ist und auch noch keine einzige Leitung beschädigt hat. Dabei gehen sie allerdings ein großes Risiko ein, denn die Entdeckung einer solchen illegalen Mailbox kann Kosten für Bußgelder, Rechtsanwaltskosten und Beschlagnahme der Hardware nach sich ziehen.

Völlig legal ist es, wenn Sie an Ihren Atari ST eine postgeprüften Akustikkoppler – also mit FTZ- bzw. ZZF-Nummer – mit 300 Baud[1] anschließen und sich auf Datenreisen begeben. Es gibt inzwischen auch Koppler, die mit 1200 Baud arbeiten können; die postzugelassenen haben allerdings einen horrenden Preis.

Zum Zeitpunkt, da dieses Manuskript entsteht, läuft die politische Diskussion über die Aufgabe des Endgerätemonopols der Post. Es könnte sein, daß in Zukunft jeder Telefonteilnehmer an seine Steckdose anschließen kann, was er will. Wenn die Post dabei keine restriktiven Zulassungsbeschränkungen einführt, wird DFÜ legal auch mit technisch anspruchsvolleren Mitteln preiswert möglich sein.

Ein Anschluß mit Akustikkoppler und die Telefonnummer einer Mailbox nutzen Ihnen allerdings noch wenig, da Ihre Tastatureingaben ja an die richtige Schnittstelle geleitet werden müssen. Sie benötigen ein Terminalprogramm, und das beste, das es als Public-Domain gibt, werden Sie in diesem Kapitel kennenlernen: UniTerm.

Das Programm UniTerm von Simon Poole bietet ungeheuer viele Funktionen und sollte sich mit allen denkbaren Mailboxen oder Großrechnern benutzen lassen. Es emuliert mehrere weitverbreitete Terminals einschließlich einiger Tektronix Grafik-Terminals. Den Begriff "Terminalemulation" kennen Sie vom `EMULATOR`-Accessory der Atari-Systemdiskette, das einen VT-52-Emulator beinhaltet.

Bei einer Emulation verhält sich ein Rechner oder eine Software so wie ein Produkt eines anderen Herstellers. Mit dem VT-52-Emulator arbeitet der Atari ST so, als wenn er ein Terminal der Firma DEC mit der Bezeichnung VT-52 wäre. Das wichtige dabei sind die Steuerzeichen, die ein Terminal versteht – Sie müssen sich das als eine Konvention wie bei Druckern vorstellen. So, wie es Druckeranpassungen für die Steuercodes der EPSON-Familie gibt, ist eine VT-52 Emulation eine Bildschirmanpassung für diese Terminals.

[1] Baud ist die Maßzahl für die Geschwindigkeit der Datenübertragung. Sie gibt an, wieviele Bits pro Sekunde gesendet oder empfangen werden können. Bei 300 Baud werden somit etwas über 30 Zeichen in der Sekunde übertragen.

9.1 Terminal- und Kommandomodus

UniTerm kennt zwei Modi: Es kann sich in der normalen Terminalemulation befinden, in der Sie direkt mit einer Mailbox arbeiten oder im Kommandomodus, in dem Sie Befehle direkt an UniTerm geben.

Im Terminalmodus verhält sich UniTerm wie das emulierte Terminal: Der Bildschirm befindet sich im Zeichenmodus, wie bei TOS-Programmen und an seinem unteren Rand gibt es eine Statuszeile. Alle Tastatureingaben werden an die serielle Schnittstelle weitergeleitet und ankommende Zeichen erscheinen auf dem Bildschirm. Terminals sind in der Regel eher langweilige Geräte, eine GEM-Oberfläche paßt nicht in dieses Konzept.

Im Kommandomodus haben Sie UniTerm als GEM-Programm vor sich, mit Menüs und einem gewohnten Desktop. Alle Auswahlen werden mit der Maus vorgenommen.

Um den Unterschied nochmals klarzumachen: Im Terminalmodus haben Sie quasi den Bildschirm des angerufenen Rechners vor sich und benutzen dessen Oberfläche. Alle eingegeben Kommandos wirken auf das dort laufende Mailboxprogramm. Im Kommandomodus zeigt sich UniTerm und Sie nehmen alle Einstellungen vor, die das Verhalten von Uniterm auf dem Atari beeinflussen.

9.2 Der Kommandomodus

Beim Programmstart befindet sich UniTerm im Terminalmodus. Bei der ersten Sitzung werden Sie wahrscheinlich die Voreinstellungen an Ihren Geschmack und den angerufenen Rechner anpassen wollen. Um in Kommandomodus zu kommen, müssen Sie die `Help`-Taste drücken. Der Bildschirm wird umgeschaltet und Sie haben die Menüleiste vor sich.

Das `Desk`-Menü

Im `Desk`-Menü können Sie sich unter `About UniTerm ...` über den Autor und die aktuelle Version des Programms informieren. UniTerm wird sehr oft verbessert, so daß in kurzen Abständen neue Versionen erscheinen. Die hier besprochene Version ist 2.0b – wenn Sie eine neuere Kopie haben, müssen Sie Angaben über eventuelle Änderungen in der englischsprachigen Dokumentation suchen.

Das `File`-Menü

Im `File`-Menü können Sie Voreinstellungen für UniTerm laden oder sichern und einige Betriebssystemfunktionen ausführen.

Mit `Load Setup` laden Sie eine Datei mit Voreinstellungen. In einer File-Select-Box können Sie den Namen auswählen. Wenn Sie für verschiedene Mailboxen oder Großrechner sehr unterschiedliche Einstellungen benutzen, ist es praktisch, sie in jeweils einer Datei zu sichern und dann nach Bedarf einzuladen. Sie ersparen sich so einige Arbeit.

`Save Setup` speichert die momentan gültigen Einstellungen. Den Namen der Datei können Sie wieder per Maus auswählen oder über die Tastatur eingeben. Als Datei-Kennung sollten Sie `.SET` verwenden, dann verlieren Sie nicht den Überblick über Ihre Dateien. Die Datei mit dem Namen `UNITERM.SET` wird beim Programmstart automatisch geladen, also sollten Sie die am häufigsten benutzten Einstellungen unter diesem Name Datei ablegen.

Falls Sie ein selbstwählendes Modem benutzen, kann UniTerm auch selbständig eine Mailbox anrufen. Die dazu notwendigen Nummern lassen sich mit `Load Numbers` und `Save Numbers` laden und speichern.

Die nächsten vier Menüpunkte stellen einige Desktop-Operationen für Dateien zur Verfügung. Mit `Show Space` zeigt UniTerm den noch freien Platz auf dem Massenspeicher an. Wenn Sie z.B. eine sehr lange Datei von einem Großrechner kopieren wollen, ist es sinnvoll vorher zu überprüfen, ob Sie noch genug Platz zum Abspeichern haben oder vielleicht die Diskette wechseln sollten.

Der Ordner, in den UniTerm Dateien ablegt, setzen Sie mit `Set Path` fest. Falls Sie eine Datei nicht mehr benötigen oder unbedingt Platz auf dem Massenspeicher brauchen, können Sie mit `Delete File` eine Datei löschen. In einer File-Select-Box wählen Sie dazu den Filenamen aus.

Wenn Sie während einer Sitzung ein anderes Programm auf dem Atari laufen lassen wollen, oder auf ein Freizeichen warten müssen, brauchen Sie UniTerm nicht zu verlassen. Mit `Run Program` können Sie ein anderes Programm starten. Dabei bleibt UniTerm im Speicher und belegt natürlich RAM. Wenn das Programm beendet wird, erscheint nicht der GEM-Desktop, sondern UniTerm und Sie können weiter arbeiten. Es gibt allerdings schlecht programmierte Software, die in dieser Situation nicht korrekt arbeitet.

`Quit` verläßt nicht UniTerm, sondern nur den Kommandomodus, so daß sich das Programm wieder im Terminalmodus befindet. Das Gleiche erreichen Sie mit einem Klick der linken Maustaste oder durch Drücken von `Alternate+Insert`.

Das `Transfer`-Menü

Im `Transfer`-Menü legen Sie das Übertragungsprotokoll für Dateien fest. Falls Sie sich mit DFÜ noch nicht auskennen, bedarf dieser Begriff natürlich

einer Erklärung.

Wenn Sie über den Terminalemulator mit einem anderen Rechner kommunizieren, geschieht dies zeichenweise mit dem normalen ASCII[2]-Zeichensatz. Sie tippen Kommandos wie bei einer Kommandoshell oder einem BASIC-Computer aus der Zeit vor dem Atari ST ein.

Nun können Sie aber auch Daten zwischen Rechnern austauschen. Dabei kann es sich nicht nur um lesbare ASCII-Zeichen handeln, sondern auch um Bilderdateien, Programme und alles andere, was Sie auch sonst an Dateien kennen. Da es sich nicht um Texte handelt, kann man diese Aufgabe nicht mit einer Terminalemulation lösen. Was man braucht ist eine Datei-Übertragung.

Falls ein Textzeichen bei Ihnen falsch ankommt, können Sie aus dem restlichen Wort oder Text rekonstruieren, welcher Buchstaben eigentlich gemeint war. Bei einem Programm hingegen kann der kleinste Fehler die größten Schaden anrichten – im besten Fall funktioniert es nicht. Es ist also bei der Datei-Übertragung sinnvoll, die korrekte Übertragung zu überprüfen. Dies geschieht mit einem Protokoll.

Bei einem Protokoll ist genau festgelegt, wie Sender und Empfänger einer Datei zusammenspielen. Ich will Ihnen das an einem simplen Protokoll darstellen, das aber nur das Prinzip schildern soll.

Im Beispielprotokoll RT sei festgelegt, daß eine Datei in Blöcke von jeweils 128 Bytes aufgeteilt wird, die die Rechner einzeln übertragen. Zum Beginn der Übertragung schickt der Sender ein Zeichen, auf das der Empfänger mit einer Bestätigung antworten muß, um zu erklären, daß er zum Empfang bereit ist. Dieser erste Teil ist die Verbindungsaufnahme. Beide Teilnehmer sind jetzt übertragungsbereit.

Der Sender schickt in RT nun eine Zahl, die angibt, wieviele Blöcke gesendet werden sollen. Der Empfänger muß das wiederum bestätigen und stellt sich auf eine entsprechend große Datei ein.

Nach dieser Bestätigung beginnt der Sender mit der Blockübertragung. Vor jedem Block schickt er eine Ankündigung, daß die Übertragung jetzt beginnt. Sie wird vom Empfänger bestätigt und ein Block von 128 Bytes wird geschickt. Danach erwartet der Sender wieder eine Bestätigung.

Beim Empfänger liegt jetzt ein Block, aber ist er auch korrekt übertragen worden? Um das festzustellen wird eine Prüfsumme übermittelt, der Sender rechnet nach und bestätigt entweder oder fordert den Block nochmals an.

[2] ASCII = American Standard Code for Information Interchange

Auf diese Weise werden alle Blöcke übertragen. Wenn Bestätigungen des Empfänger ausbleiben, wiederholt der Sender die Anforderung einer Bestätigung. Ebenso wiederholt er eventuell einige Blöcke oder das Senden einer Prüfsumme. Auf diese Art werden also alle übertragenen Daten bestätigt und die Blöcke überprüft, so daß alles, was geschickt, wurde entweder korrekt ankommt oder abgelehnt wird[3].

Das war also unser Beispielprotokoll RT. Wie Sie sehen, sind Protokolle teilweise aufwendig, können aber sehr sicher gemacht werden. UniTerm bietet Ihnen im `Transfer`-Menü die Auswahl unter vier verschiedenen Protokollen an. `ASCII` ist dabei eigentlich kein Protokoll, es handelt sich um eine reine Zeichenübertragung, bei der allerdings eine Zeichenwandlung definiert werden kann.

`XMODEM` ist eine Public-Domain-Entwicklung, `YMODEM` eine Erweiterung davon, mit der mehrere Dateien nacheinander übertragen werden können. Etwas älter und auch mit Public-Domain-Ursprung ist `Kermit`. Die vielfältigen Optionen zu diesen Protokollen können Sie im `Settings`-Menü einstellen – ich werde sie Ihnen dort vorstellen.

Welches Protokoll Sie wählen hängt von dem angerufenen Rechner ab. Da sie untereinander überhaupt nicht kompatibel sind, müssen Sie sich vor einer Übertragung über das Protokoll der "Gegenseite" und die notwendigen Optionen informieren. Geschwindigkeitsunterschiede zwischen den Protokollen können eigentlich nur durch die jeweilige Implementierung entstehen, so daß ich Ihnen kein Protokoll besonders empfehlen möchte. Ich persönlich benutze meistens XMODEM.

Das `Settings`-Menü

Im `Settings`-Menü legen Sie die Parameter für die serielle Schnittstelle, die Terminalemulation, das Übertragungsprotokoll und die Speicherverwaltung von UniTerm fest.

Nach Auswahl von `RS232` erscheint eine Dialogbox, in der Sie die serielle Schnittstelle des Atari ST einstellen. Der erste Parameter ist die Baudrate, also die Übertragungsgeschwindigkeit. Hier können Sie von 300 bis 19200 Baud wählen; Akustikkoppler arbeiten in der Regel mit 300, Modems mit 1200 bis 2400 Baud. Die "gesplittete" Übertragungsrate 1200/75 Baud, wie sie das BTX-System verwendet, können Sie nicht einstellen, der Atari braucht dafür ein spezielles Zusatzprogramm.

`Flowcontrol` legt fest, wie die Zeichenübertragung synchronisiert wird. Wenn mehr Zeichen ankommen, als UniTerm ausgeben kann, gibt es einen

[3] Preisfrage: Funktioniert die Übertragung mit RT korrekt, wenn eine Bestätigung falsch übertragen wird oder sind dazu weitere Aktionen notwendig?

Stau im Eingabepuffer, so daß der Sender kurzzeitig gestoppt werden muß, bis die Zeichen verarbeitet sind. Bei `Xon/Xoff` benutzt UniTerm dazu die Steuerzeichen Control+Q und Control+S; bei `RTS/CTS` werden Leitungen an der Schnittstelle verwendet. Bei `None` können Zeichen verloren gehen.

Mit `Parity`, `Databits` und `Stopbits` wird die Übertragung der Bytes geregelt. Da es sich ja um eine serielle Schnittstelle handelt, kann ein Byte nicht am Stück geschickt werden, dazu wären acht Leitungen, also eine Parallel-Schnittstelle notwendig. Die Daten werden vielmehr bitweise verschickt. Um Anfang und Ende eines Bytes erkennen zu können, werden Stopbits verwendet. Parity ist eine Art Prüfsumme wie oben beschrieben, und Databits gibt an, wie lang ein Byte sein soll.
Die Einstellungen für das Prüfbit sind:

`None` Kein Prüfbit

`Even` Das Prüfbit ist gesetzt, wenn eine gerade Anzahl von Bits im Byte gesetzt ist.

`Odd` Das Prüfbit ist gesetzt, wenn eine ungerade Anzahl von Bits im Byte gesetzt ist.

Für die Bytelänge:

`8` Es werden acht Bits pro Byte verwendet.

`7` Es werden sieben Bits pro Byte verwendet.

Und für das Stopbit:

`1` Ein Stopbit kennzeichnet das Ende eines Bytes.

`2` Zwei Stopbits kennzeichnen das Ende eines Bytes.

Schließlich können Sie in der Dialogbox noch das Verhalten des Terminals einstellen. Dabei gibt es drei Möglichkeiten:

`Full` Ihre Eingaben werden vom angerufenen Computer als Echo zurckgeschickt.

`Echo` UniTerm übernimmt das Echo und gibt jede Eingabe auf dem Bildschirm aus.

`Local` Ihre Eingaben werden nur auf dem Bildschirm ausgegeben und nicht an den angerufenen Rechner geschickt.

Mit `Terminal 1` können Sie Einstellungen an der Terminalemulation vornehmen. Es erscheint eine Dialogbox, in der die wichtigsten Parameter gesetzt werden.

Sie haben die Auswahl zwischen fünf verschiedenen Emulationen. `VT 200`, `VT 102`, `VT 100` und `VT52` sind Terminals der Firma DEC, von denen Ihnen das VT52 schon durch das Accessory `Emulator` sein dürfte. `4014` emuliert ein Terminal Tektronics 4014 im "alpha mode"; bei `DCM` haben Sie das gleiche Gerät im "Display Control Mode 3" vor sich.

Wenn Sie eine private Mailbox benutzen, werden Sie kaum Terminalsteuerzeichen brauchen; hierfür ist z.B. `VT102` als Einstellung gut geeignet. Falls Sie aber mit einem Großrechner arbeiten wollen, muß das Terminal richtig eingestellt werden, da viele Steuersequenzen verwendet werden. Welche Emulation Sie dann mit welchen Parametern verwenden, müssen Sie bei dem entsprechenden Rechenzentrum in Erfahrung bringen. Die von UniTerm angebotenen Terminals sind so verbreitet, daß es kein Problem sein sollte eine optimale Anpassung zu finden.

Die meisten Terminals haben Funktionstasten. UniTerm verwendet die F-Tasten des ST für eigene Kommandos, so daß der Zehnerblock auf der Tastatur dafür herhalten muß. Mit der Einstellung `Appl.` werden für jede Taste entsprechende Sequenzen geschickt; mit `numeric` gelten nur die vier Tasten `(`, `)`, `/` und `*` als Funktionstasten.

Die Cursor-Tasten können ebenfalls verschiedene Steuerzeichen erzeugen. Mit `ANSI Set` und `ANSI Reset` müssen Sie sie an die Vorgaben des angerufenen Rechners anpassen.

In früheren Computer-Zeiten wurde der ASCII-Code mit nur sieben Bits benutzt. Mit `7 Bits` und `8 Bits` können Sie einstellen, ob UniTerm das achte Bit automatisch löschen soll oder ein ganzes Byte pro Zeichen für die Kommunikation verwenden darf.

`Jump` und `Smooth` stellt ein, ob der Bildschirm ruckartig oder gleitend gescrollt werden soll. Bei `Jump` verschiebt UniTerm den kompletten Bildschirm um eine ganze Zeile; bei `Smooth` schiebt sich der Bildschirminhalt sanft nach oben.

Für die Übertragung von Zeilenenden, also das Drücken von `Return` gibt es zwei Konventionen. Bei der ersten wird nur das Steuerzeichen CR für "Carriage Return" (Wagenrücklauf) gesendet. Es gibt aber auch Rechner, die danach noch ein LF für "Line Feed" (Zeilenvorschub) benötigen. Mit `CR->CR` und `CR->CRLF` stellen Sie UniTerm entsprechend ein.

Die `Wrap`-Einstellung beeinflußt den Zeilenumbruch. Wenn der angerufene Rechner nach dem achtzigsten Zeichen kein extra Steuerzeichen schickt, kann UniTerm selbständig einen neue Zeile anfangen, falls Sie `Auto` gewählt haben. Bei `None` schreibt UniTerm einfach in der gleichen Zeile weiter. Die ankommenden Zeichen werden dann am rechten Bildschirmrand übereinander ausgegeben, so daß sich die `Auto`-Einstellung empfiehlt.

Das Aussehen des Cursors können Sie unterschiedlich gestalten. Mit `Blink` wird er auf blinkend eingestellt, bei `Blink off` bleibt er ruhig.

`Underline` erzeugt als Cursor einen Unterstrich (_), `Block` einen rechteckigen Block.

Mit `Background` stellen Sie die Bildschirmfarbe ein. Bei `Light` zeigt UniTerm die Zeichen schwarz auf weiß, bei `Dark` sind die Buchstaben weiß und der Hintergrund schwarz.

Der letzte Parameter, den Sie in der Dialogbox auswählen können, ist das Verhalten der `Delete`-Taste. Bei der Einstellung `Delete` sendet sie das Steuerzeichen DEL, bei `Backspace` ein BS. Die Taste `Backspace` schickt den jeweils anderen Steuercode.

Durch Auswahl des Menüpunktes `Terminal 2` erscheint wieder eine Dialogbox, in der Sie weitere Terminalparameter einstellen. Unter `Answerback` legen Sie eine Zeichenkette fest, die bei Benutzung eines Modems, das selbst wählen kann, als Antwort geschickt wird.

Haben Sie einen Drucker angeschlossen, teilen Sie das UniTerm durch Auswahl von `Connected` mit. Anderenfalls sollten Sie `No Printer` einstellen. Soll UniTerm nach Protokolldrucken einen Seitenvorschub auf dem Drucker erzeugen, muß `FF` gewählt sein. Bei `Nothing` wird kein Steuerzeichen geschickt.

UniTerm besitzt einen kleinen eingebauten Zeileneditor, "Single Line Editor" genannt, den Sie unter `SLE` ein- (`Enabled`) oder ausschalten (`Disabled`) können. Seine Benutzung erkläre ich Ihnen in Kapitel 9.3 auf Seite 118. Den Namen einer Makrodatei, die beim Programmstart automatisch ausgeführt wird, können Sie unter `Auto executed macro` einstellen. Die Beschreibung der Makro-Befehle finden Sie in Kapitel 9.4 ab Seite 121.

Terminals kennen wie eine Schreibmaschine Tabulatoren. Das Steuerzeichen Control+I oder TAB setzt den Cursor einen Tabulatorstop weiter. Mit `Tabs` können Sie diese Stops in UniTerm setzen.

Es erscheint eine Dialogbox mit drei Zeilen, in denen Sie die Tabstops für die Spalten 1 bis 60, 61 bis 120 und 121 bis 132 einstellen können. Bei einem invertierten Zeichen ist an dieser Stelle ein Stop eingestellt. Mit der Maus schalten Sie die Stops durch Anklicken um.

In der Dialogbox, die nach Auswahl von `Graphics` erscheint, nehmen Sie die Einstellungen für die Grafikterminal-Emulation vor. UniTerm kann sich – wie schon beschrieben – wie einige Modelle des Herstellers Tektronix verhalten, so daß Sie bei Bedarf sogar Grafikprogramm auf einem Großrechner laufen lassen können.

Die erste Auswahlmöglichkeit erlaubt es, die Erkennung von Tektronix-Steuerzeichen abzustellen. Bei `Disabled` werden diese Codes ignoriert, bei `Enabled` bewirken sie ein Einschalten des Grafikmodus. Sie werden sich vielleicht fragen, wozu das gut sein soll. Nun, sicher haben Sie auch schon einmal eine "schlechte" Telefonverbindung erwischt. Die dann auftretenden Störgeräusche verändern eventuell Informationen, so daß fälschlicherweise

ein Code zum Umschalten in den Grafik-Modus ankommen kann, obwohl Sie mit einem reinen Textrechner kommunizieren. Und dieses Umschalten ist sehr lästig, denn UniTerm löscht dabei den Bildschirm und interpretiert ankommende Steuerzeichen als Grafikkommandos. Bei abgeschaltetem Tektronix-Modus erscheint zwar auch "Datenmüll" auf dem Schirm, aber Sie bleiben im Textmodus.

`GIN Termination String` und `Status Termination String` sind Zeichenketten, die nach einem GIN- bzw. Status-Kommando geschickt werden. Diese sehr speziellen Einstellungen müssen Sie nach Rücksprache mit dem angerufenen Rechenzentrum vornehmen.

Da der Bildschirm des Atari ST kleiner als die der emulierten Grafikterminals ist, können Sie Grafiken verkleinern lassen. Bei Auswahl von `True` werden Linien etc. am Bildschirmrand einfach abgeschnitten; bei `Fill Screen` paßt UniTerm die Größen an.

Eine letzte – wieder sehr spezielle – Einstellung für den Grafikmodus ist die Angabe, ob Sie bei der Emulation des Tektronix 4010 Terminals das Steuerzeichen DEL verwenden können (`Accept`) oder nicht (`Ignore`).

Unter dem Menüpunkt `Transfer` stellen Sie die Parameter für das gerade gewählte Übertragungsprotokoll ein. Um die Einstellungen für das Kermit-Protokoll vorzunehmen, müssen Sie also zunächst im `Transfer`-Menü `Kermit` einstellen und dann dem `Transfer`-Punkt im `Settings`-Menü wählen. Jedes Protokoll hat sehr unterschiedliche Optionen, die Sie so einstellen müssen, wie sie der angerufene Rechner verlangt. Die Protokolle selber sind Standards, die Einstellungen variieren jedoch.

Für den Transfer von ASCII-Dateien können Sie Übersetzungstabellen erstellen. Ankommende oder abgeschickte Zeichen werden dabei automatisch in andere gewandelt. Diese Option ist besonders nützlich, wenn der angerufene Rechner die Umlaute anders verwenden als UniTerm.

Es gibt getrennte Tabellen für empfangene und verschickte Zeichen, die Sie in der Dialogbox einstellen können. Für die eingehenden Zeichen müssen Sie den Bereich `Translation on input`, für gesendete `Translation on output` verwenden.

Den ASCII-Code des zu wandelnden Zeichens legen Sie mit drei Reglern fest. Jede der drei Stellen, die auf dem Bildschirm angezeigt werden, kann mit den Buttons ⇑ und ⇓ um eins erhöht bzw. verringert werden. Neben dieser Anzeige geben Sie den Code ein, in den das Zeichen übersetzt wird.

Dazu ein kleines Beispiel: Auf 8-Bit Rechnern wurde ein ASCII-Code verwendet, bei dem die Umlaute parallel zu einigen Sonderzeichen lagen. So entsprach das Zeichen `[` dem Umlaut `Ä`. Wenn Sie nun mit einer Mailbox arbeiten, die diesem alten Code folgt, können Sie die Übersetzungstabellen sinnvoll einsetzen.

Sie müssen dabei für `input` und `output` den Code 91 mit den Reglern

einstellen – er entspricht dem Zeichen [. Als Übersetzung tippen Sie 142 ein, was der Code für das Ä auf dem Atari ST ist. Jedesmal, wenn UniTerm das Zeichen mit dem ASCII-Code 91 erhält, wird es nach 142 gewandelt und die Umlaute erscheinen korrekt in Ihren Texten.

Ob UniTerm diese Übersetzungstabellen verwenden soll, legen Sie unter `Send` und `Receive` fest. Bei `Translated` übersetzt UniTerm die Codes anhand der Tabellen; bei `Raw` bleiben alle Zeichen unverändert.

Eine weitere Übersetzungsfunktion betrifft die Behandlung des Zeilenendes. Jedesmal, wenn Sie die Taste `Return` drücken, erzeugt UniTerm die Steuerzeichen CR und LF zur Kennzeichnung des Zeilenendes. Diese Konvention kann aber auf anderen Rechnern anders aussehen. Unter `Translate EOL to` legen Sie fest, auf welche Art UniTerm die Steuerzeichen CR LF wandeln soll. Sie haben die Auswahl zwischen `CR` für ein einzelnes CR, `LF` für ein einzelnes LF, `CRLF` ohne Unwandlung und schließlich `_CR` für Leerzeichen plus CR.

Unter `Start of file transfer` können Sie eine Zeichenkette eintragen, die zum Beginn einer ASCII-Übertragung gesendet wird. Als Gegenstück dazu wird der String, der bei `End of file transfer` eingetragen ist, am Ende des Transfers gesendet.

`Method` legt fest, ob die Übertragung mit dem angerufenen Rechner synchronisiert werden soll. Bei `Paced by Echo` wartet UniTerm darauf, daß ein abgeschicktes Zeichen als Echo identisch bestätigt wird und schickt dann das nächste. Bei `Normal` wird keine Rücksicht auf die Antwort der Mailbox genommen.

Einige Mailboxen sind nicht so schnell wie UniTerm, wenn es um den Empfang von Zeichen geht. Schließlich müssen die Eingaben abgespeichert und verschiedene Verwaltungstätigkeiten übernommen werden. Sie können daher die Ausgabegeschwindigkeit von UniTerm verzögern. Unter `Delay time` geben Sie eine Zeit in Millisekunden ein, die UniTerm nach jedem Zeichen warten soll. Wenn Sie das Zeitverhalten einer Mailbox noch nicht kennen, ist es sinnvoll, hier zunächst cirka 50 Millisekunden einzutragen und dann experimentell zu versuchen, die Verzögerung so niedrig wie möglich einstellen. Kommen einige Ihrer Eingabezeichen nicht korrekt bei der Mailbox an, ist der Wert zu niedrig.

Für die Protokolle XMODEM und YMODEM werden die gleichen Voreinstellungen verwendet. Die wichtigsten davon legen die Größe der Übertragenen Blöcke und die Art der Prüfsumme[4] fest.

Die Blockgröße kann 128 oder 1024 Bytes betragen. Sie müssen entsprechend der Vorgaben des angerufenen Rechners die Buttons auswählen. Wenn Ihre

[4]Sie erinnern sich an RT? Genau, es handelte sich bei dem Beispielprotokoll um eine Vereinfachung von XMODEM.

Mailbox beide Größen unterstützt, kann `1024` bei störungsfreien Leitungen schneller sein, da weniger Verwaltungsinformationen und Protokollaktionen übertragen werden müssen. Wenn Sie eine fehleranfällige Verbindung erwischt haben, sind `128` besser, da hierbei schadhafte Blöcke früher erkannt werden können.

Für die Prüfsumme stehen `ChkSum` für eine einfach Aufaddierung der Daten und `CRC` für den "Cyclic-Redundancy-Check" bereit. Für 128-er Blöcke wird besser `ChkSum` verwendet; bei Blöcken mit einem Kilobyte ist `CRC` sinnvoller.

`Accept ASCII-NUL` legt fest, ob das Steuerzeichen NUL (ASCII-Code 0) beim Transfer erlaubt ist. Bei reinen ASCII-Dateien kann bei Verwendung von XMODEM dieses Zeichen zur Synchronisation verwendet werden. Bei "normaler" Übertragung von Dateien, also auch von Programmen, sollten Sie hier `YES` auswählen.

`Timeout after` bestimmt, wie lange UniTerm beim XMODEM-Protokoll auf ein Zeichen warten soll. Wird diese Zeit überschritten, nimmt UniTerm an, daß die Verbindung unterbrochen wurde und stopt die Übertragung. Sie müssen diesen Wert mindestens so groß einstellen, daß der angerufene Rechner Zeit hat, die zu übertragenden Daten zu laden oder abzuspeichern.

Eine Übertragung, bei der ständig Fehler auftreten, kann UniTerm automatisch abbrechen. Alle Fehler werden mitgezählt und Sie können unter `Maximum number of errors` einstellen, wann UniTerm die Übertragung abbrechen soll. Bei "lauten" Telefonleitungen mit vielen Störungen kann es sinnvoll sein, eine Mailbox in der Hoffnung auf eine bessere Verbindung erneut anzurufen. Es kann aber auch sein, daß der angerufene Rechner Fehler macht, oder Sie einen Bedienungsfehler der Mailbox begangen haben. Um zu vermeiden, daß UniTerm endlos auf eine Übertragung wartet, wird die maximale Fehlerzahl daher begrenzt.

Bei der Einstellung der Parameter für eine Übertragung mit dem Kermitprotokoll sind einige Optionen ähnlich. `Timeout after` legt wiederum fest, wie lange UniTerm maximal auf ein Zeichen warten soll und `Maximum number of retries` enthält die maximale Anzahl der Wiederholungen, falls ein Fehler bei der übertragung auftritt.

`Number of padding characters` besagt, wieviele von dem Zeichen, das Sie unter `Padding charakter` festlegen, vor der Übertragung eines Pakets gesendet werden sollen. Die Größe eines Pakets[5] steht in `Packet size`. Kermit unterscheidet zwischen "kleinen" und "großen" Paketen. Kleine können bis zu 94 Bytes lang sein, wenn Sie – und der angerufene Rechner – große verwenden, können diese bis zu 2048 Bytes lang sein. Das Zeichen zur Markierung eines Paketanfangs tragen Sie unter `Start of packet` ein.

[5] Kermit verwendet den Begriff "Paket" anstelle des "Blocks" bei XMODEM.

Wenn sich Kermit nicht im Binary-Modus befindet, den Sie bei der Übertragung einschalten können, werden nur sieben Bits eines Zeichens übertragen. Zudem gelten die ASCII-Steuerzeichen mit einem Wert unter 32 nicht als Daten. Um sie dennoch übertragen zu können benutzt Kermit "Quoting Chars", die Sie sich als Anführungszeichen vorstellen können, mit denen 8-Bit-Daten und Steuerzeichen eingeschlossen werden.

Unter `Quote character` tragen Sie das Anführungszeichen für Steuerzeichen ein; unter `8 bit quote character` das für 8-Bit-Daten. Das sogenannte Prefixing, mit dem das achte Bit bei einer 7-Bit-Übertragung fest auf einen Wert voreingestellt ist, wird mit dem unter `Repeat prefix character` eingetragenen Zeichen eingeleitet.

Alle diese Einstellungen hängen natürlich vom angerufenen Rechner ab. Sie sollten sie nur dann verändern, wenn Sie genaue Unterlagen über das Kermit auf der "Gegenseite" haben. Die Dokumentation zu Kermit selber sollten Sie bei Spezialfällen ebenfalls genau studieren. Sie hat allerdings einen solchen Umfang, daß ich es an dieser Stelle mit einem Hinweis darauf belassen muß.

Die restlichen Buttons steuern die noch offenen Einstellungen. `IBM mode` besagt, daß UniTerm auf das Steuerzeichen XOn warten soll, bevor es ein Paket abschickt. Um diese Option nutzen zu können, müssen Sie bei der Einstellung der seriellen Schnittstelle das XOn/XOff-Protokoll ausschalten.

Für die Prüfsumme haben Sie die Auswahl zwischen einer acht (`Chk1`) oder sechzehn Bit langen (`Chk2`) Checksum. Eine dritte Möglichkeit ist wieder der "Cyclic-Redundancy-Check" `CRC`.

Bei `Incomplete Files` legen Sie fest, wie sich UniTerm verhält, wenn unvollständige Dateien empfangen werden, also einige Pakete fehlen. Mit `Keep` speichert UniTerm diese Dateien eben nicht komplett ab, und mit `Discard` werden sie verworfen.

Bitte denken Sie daran, daß alle Einstellungen bei den Protokollen von den Möglichkeiten des angerufenen Rechners abhängen. Sie müssen dessen Optionen kennen und sich danach richten. Wenn Sie sie ändern wollen, müssen Sie das auch bei der "Gegenseite" tun, sonst kann die Übertragung nicht funktionieren. UniTerm sollte sich aber an alle denkbaren Protokoll-Optionen anpassen lassen.

Mit `Buffers` können Sie die interne Speicheraufteilung von UniTerm verändern. Das Programm benutzt mehrere Puffer, deren Größe jeweils mit den Buttons `+` und `-` eingestellt wird.

Die Puffer sind:

- Die Größe des Transfer-Puffers, in dem bei der Datei-Übertragung gepuffert wird. Erst wenn dieser Speicherbereich voll ist, schreibt UniTerm die

Daten auf den Massenspeicher. Je größer dieser Wert ist, umso weniger Zugriffe benötigt UniTerm, was die Übertragung etwas schneller macht.

- Der Clipboard-Puffer, in dem UniTerm die mit der Maus definierten Ausschnitte (siehe Kapitel 9.3 auf Seite 119) ablegt.
- Die Größe des RS232-Puffers, also die Anzahl der Bytes, die an der seriellen Schnittstelle gesammelt werden. Wenn UniTerm nicht im Terminalmodus ist, kommen alle empfangenen Zeichen in diesen Puffer, bis sie verarbeitet werden können. Um diesen Wert zu verändern, müssen Sie allerdings die gesamten Voreinstellungen sichern, UniTerm verlassen und erneut starten.
- System-Puffer. Für eigene Daten braucht UniTerm einen Speicherbereich, der mindestens 5 KiloByte groß sein muß.
- Die Größe des History-Puffers. Wenn UniTerm Ihre Aktionen mitschreibt werden sie hier gespeichert. Die Größe dieses Puffers ist der Rest des verfügbaren Speichers, der nach Abzug der anderen Puffer übrig bleibt.

Das `Other`-Menü

In diesem Menü finden Sie noch zwei Arbeitserleichterungen: den Dialer und die Funktionstasten.

Mit `Dialer` erscheint eine Dialogbox, in der Sie bis zu zehn Telefonnummern festlegen können, die UniTerm bei Verwendung eines Modems selbsttätig wählen kann.

Sie haben für jeden Eintrag drei Felder: Das Namensfeld, den Namen einer Makro-Datei, die nach dem Anruf ausgeführt werden soll und schließlich die Telefonnummer. Die Makro-Datei erlaubt es Ihnen, automatisch nach dem Anruf Befehle abzusetzen oder Dateien zu schicken und vieles mehr. Zu Makrodateien erfahren Sie in Kapitel 9.4 ab Seite 121 alle Details.

`Prefix` und `Suffix` sind Zeichenketten, die vor und nach der Telefonnummer ausgegeben werden. Sie enthalten die entsprechenden Modem-Kommandos zum Wählen, die Sie natürlich an Ihr Gerät anpassen müssen.

Bei einer erfolgreichen Einrichtung einer Verbindung gibt UniTerm die Meldung, die unter `Connect msg.` auf dem Bildschirm aus. Beim Beenden der Verbindung wird die Zeichenkette unter `Hangup` an das Modem geschickt. Als letzter Parameter läßt sich noch einstellen, wie oft UniTerm das Wählen wiederholen soll. Sie tragen diese Zahl unter `Retries` ein.

Den Aufruf des Dialers finden Sie später beim Terminalmodus beschrieben. Hier zunächst zum zweiten Menüpunkt `Edit functionkeys`.

Sie können die Funktionstasten mit zwanzig Zeichenketten programmieren, die dann automatisch gesendet werden. Bestes Beispiel wäre hier der Einlog-Vorgang, bei dem Sie Namen und Passwort angeben müssen. Die entsprechenden Zeichen legen Sie auf eine Funktionstaste und brauchen somit nur noch eine einzige Taste zu drücken und werden sich nicht mehr verschreiben. In den Zeichenketten können auch Makrokommandos vorkommen.

Nach der Menüauswahl erscheint eine Dialogbox mit den zwanzig Zeichenketten. Sie werden den Funktionstasten (`F1` bis `F10`) und den Funktiostasten in Verbindung mit der `Shift`-Taste (⇑`F1` bis ⇑`F10`) zugeordnet.

Tastenkommandos

Falls Sie bei der Arbeit mit UniTerm im Kommandomodus nicht ständig zwischen Tastatur und Maus wechseln wollen, können fast alle Menüpunkte auch über Tasten aufgerufen werden:

Taste	Menü	Menüpunkt
I	Desk	About UniTerm ...
L	File	Load Setup ...
S		Save Setup ...
F		Show Space
P		Set Path
D		Delete File
R		Run Program
Q		Quit
V	Settings	RS232
1		Terminal 1
2		Terminal 2
A		File Transfer (Für ASCII-Transfer)
X		File Transfer (Für XMODEM-Transfer)
Y		File Transfer (Für YMODEM-Transfer)
K		File Transfer (Für Kermit-Transfer)
G		Graphics
T		Tabs
B		Buffers
C	Other	Dialer
E		Edit Function Keys

9.3 Der Terminalmodus

Im Terminalmodus verhält sich UniTerm so wie das emulierte Terminal. Für den Bildschirm sind 24 Zeilen mit 80 Zeichen reserviert. Alle gedrückten Tasten werden an die serielle Schnittstelle geschickt und alle ankommende Zeichen auf dem Bildschirm ausgegeben. Die letzte Bildschirmzeile dient zu Statusangaben. Sie finden hier die Nummer der Programmversion und weitere Anzeigen über den Arbeitsmodus, die Meta-Einstellung (siehe Kapitel 9.3 auf Seite 118) und den Zustand der CapsLock-Taste.

Am Ende der Zeile befinden sich die Status-LEDs. Reale Terminals haben meistens eine Reihe von Anzeigen über den Terminalzustand, die mit Leuchtdioden dargestellt werden. UniTerm emuliert die LEDs durch einzelne Zeichen in der Statuszeile. Die emulierten LEDs sind von `1` bis `4` durchnummeriert und treten nur dann in Aktion, wenn der angerufene Rechner entsprechende Kommandos schickt. Daneben gibt es fünf Anzeigen, die teilweise auch auf realen Terminals zu finden sind, teilweise UniTerm-Anzeigen darstellen. Die sechs LEDs sind:

`P/-` Ist der Protokolldruck eingeschaltet, wird `P` angezeigt.

`D/-` Beim Anliegen eines Signals an der seriellen Schnittstelle erscheint `D`. Es ist dann eine korrekte Verbindung mit dem angerufenen Rechner vorhanden.

`B/-` Wenn UniTerm ein Break-Signal sendet, erscheint `B`.

`L/-` Wenn durch ein Steuerkommando die Tastatur ausgeschaltet wurde – was bei einigen Terminals möglich ist –, zeigt UniTerm `L` an.

`C/-` Ist das Mitschreiben auf Diskette eingeschaltet, wird `C` angezeigt.

`I/R` Ist auf dem angerufenen Rechner ein Zeileneditor vorhanden und Sie fügen Zeichen in die Zeile ein, erscheint `I` für "Insert", wenn der Rechner das entsprechende Steuerzeichen sendet.

Drei weitere Anzeigen – neben der Angabe über die Programmversion – finden sich in der Statuszeile. `Caps` und `Meta` zeigen an, ob der Metamodus eingeschaltet ist und ob die Shift-Verriegelung (`CapsLock`-Taste) gedrückt wurde. Daneben zeigt UniTerm an. in welchem Kommunikationsmodus es sich befindet. Entsprechend Ihrer Auswahl bei den Terminalparametern erscheint `Online` für den `Full`-Modus, `Echo` und `Local` für die anderen möglichen Einstellungen.

Den Zehnerblock verwendet UniTerm entsprechend Ihren Einstellung als Funktionstasten, die Cursor-Tasten erzeugen die eingestellten Steuersequenzen. Mit den eigentlichen Funktionstasten werden die vorprogrammierten Zeichenketten abgerufen. Die Kommandos für Uniterm im Terminalmodus sind Kombinationen mit der `Alternate`-Taste. UniTerm hat

eine Reihe von vorprogrammierten Befehlen, die jedoch mit dem Makro-Kommando `%REASSIGN` (siehe Kapitel 9.4 auf Seite 127) erweitert werden können.

Die ersten drei Kommandos steuern das Aussehen des Bildschirms. Alternate+`F2` schaltet die Anzeige zwischen 24 und 49 Zeilen um. Bei 49 Zeilen verwendet UniTerm einen kleineren Zeichensatz. Mit Alternate+`F10` legen Sie die Anzeigebreite fest. Der Befehl schaltet zwischen achtzig und 132 Zeichen pro Zeile um. Alternate+`Y` schließlich hält die Bildschirmausgabe an. Dieses Kommando ist nicht zu verwechseln mit den Steuerzeichen Control+S und Control+Q, deren Verwendung vom angerufenen Rechner abhängt.

Mit einigen Befehlen ist die Steuerung der Terminalemulation möglich. Alternate+`F1` schaltet die Tektronix-Emulation an. Der Bildschirm wechselt dabei in eine Grafikdarstellung. UniTerm löscht ihn und setzt alle Parameter für die Emulation zurück. Ein Umschalten in die Grafik-Emulation ohne Rücksetzen ist mit Alternate+`F5` möglich.

Den bei der Tektronix-Emulation vorhandenen Zoom-Modus können Sie mit Alternate+`F9` einschalten. Es erscheint der Mauscursor und Sie können durch Aufziehen eines Rechtecks den Bereich des Bildes festlegen, den UniTerm vergrößern soll. Beim Drücken der Maustaste definieren Sie die rechte obere Ecke dieses Bereichs fest und legen dann bei gedrückter Maustaste die linke untere Ecke fest.

Im Zoommodus können Sie neue Vergrößerungen festlegen oder mit `Backspace` eine Vergrößerungsstufe zurückgehen. Die Cursor-Tasten verschieben den Ausschnitt um jeweils ein Drittel der Bildschirmgröße. Mit `Return` können Sie in die Originalgröße zurückschalten und `Undo` schließlich verläßt den Zoommodus.

Zurück zum Textmodus kommen Sie mit Alternate+`F6`. UniTerm benutzt dann wieder die VT100/102-Emulation. Dies ist besonders nützlich, wenn Sie eine störanfällige Leitung benutzen und UniTerm auf falsche Steuerzeichen hin den Grafikmodus einschaltet. Falls Sie UniTerm wieder in den Zustand, der bei Programmstart vorlag, zurückversetzen wollen, müssen Sie Alternate+`F7` benutzen. UniTerm lädt dann alle Voreinstellungen neu von Diskette oder Festplatte.

Den Bildschirminhalt können Sie wie auf dem ST üblich mit Alternate+`Help` auf dem Drucker ausgeben. Falls Sie dabei keine Grafikhardcopy wünschen, kann UniTerm mit Alternate+`F4` die Zeichen auch als solche an den Drucker schicken. Die Ausgabe geht dabei natürlich schneller. Wenn Sie den Bildschirminhalt mit dem Malprogramm DEGAS bearbeiten wollen – zum Beispiel bei Bildern, die mit der Tektronix-Emulation gesendet wurden – können Sie ihn mit Alternate+`P` im entsprechenden Format auf Diskette schreiben, wozu Sie noch den Datei-Namen in einer File-Select-Box

auswählen.

Der Protokolldruck kann mit Alternate+F8 ein- und ausgeschaltet werden. UniTerm gibt dann jedes Zeichen, das auf dem Bildschirm erscheint automatisch auch an den Drucker weiter. In der Statuszeile wechselt die Anzeige der P-LED entsprechend.

Das Mitspeichern aller Zeichen schalten Sie mit Alternate+I ein und aus. UniTerm legt dann alle Zeichen im History-Buffer ab. In einer Dialogbox haben Sie die Auswahl zwischen Start (Start) oder Beenden (Stop) des Mitschreibens und Abbrechen des Befehls (Cancel).

Mit Alternate+X können Sie den Puffer auf Diskette oder Festplatte schreiben und haben so ein Protokoll Ihrer DFÜ-Sitzung als Textdatei. Den Namen der Datei legen Sie in einer File-Select-Box fest. Bei der Arbeit mit UniTerm können Sie sich mit Alternate+O den History-Puffer anschauen. UniTerm "spielt" dann alle Zeichen nochmals auf dem Bildschirm ab. Ähnlich dazu kann UniTerm mit Alternate+R einen Text von Diskette so abspielen, als wenn alle Zeichen über das Telefon kommen würden. Die Datei müssen Sie wieder in einer File-Select-Box auswählen.

Wenn Sie das GDOS-System benutzen, läßt sich der History-Puffer mit Alternate+F auf eine GDOS-Station – damit ist hauptsächlich ein Drucker oder ein Metafile gemeint – ausgeben. Sie können ein Metafile mit Zeichenprogrammen wie EASYDRAW weiterverwenden. In einer Alertbox fragt UniTerm Sie nach der Ausgabestation Printer oder Meta File. Mit Cancel können Sie den Befehl abbrechen. Falls Sie GDOS nicht installiert haben kann der Versuch einer Ausgabe mit dem Absturz des Rechners enden.

Zum Empfang von reinen ASCII-Texten müssen Sie Alternate+C benutzen. UniTerm schreibt dann bis zum erneuten Drücken dieser Befehlstaste alle Zeichen in eine Datei. In einer Alertbox werden Sie von UniTerm gefragt, ob Sie das Mitschreiben beginnen (Start) oder beenden (Stop) wollen. Beim Starten müssen Sie dann in einer File-Select-Box den Namen der Datei auswählen, in die UniTerm schreiben soll. Falls Sie versehentlich Start wählen, wenn UniTerm schon auf ASCII-Empfang eingestellt ist, gehen alle bis dahin empfangenen Zeichen verloren und die Protokolldatei hat die Filelänge Null.

Für den Datei-Transfer mit Protokollen müssen Sie Alternate+T benutzen. Je nach gewähltem Protokoll verhält sich UniTerm anders. Beim ASCII-Protokoll werden Sie in einer File-Select-Box nach dem Namen der zu sendenden Datei gefragt. Ein Empfang von ASCII-Text ist nur mit Alternate+C oder Alternate+I möglich.

Bei der XMODEM- und YMODEM-Einstellung haben Sie in einer Dialogbox die Auswahl zwischen Empfangen (Receive) und Senden einer Datei (Send). Dabei müssen Sie jeweils in einer File-Select-Box einen Datei-

Namen festlegen. Während des Transfers zeigt UniTerm in einer Box Informationen über den Verlauf der Übertragung an.

Der Button `Utilities` bei der Auswahl gibt Ihnen Zugriff auf drei Hilfsfunktionen, die Sie auch im `File`-Menü im Kommandomodus finden. In einer weiteren Dialogbox können Sie wählen zwischen `Disk Space` zur Anzeige des noch freien Platz auf dem Massenspeicher, `Delete File` zum Löschen einer Datei und `Set Path` zum Setzen des Zugriffspfades für UniTerm. Mit `Quit` erscheint wieder die vorherige Dialogbox, die Sie über `Terminal` verlassen können.

Bei der `Kermit`-Einstellung erscheint eine etwas größerer Dialogbox. Die wichtigsten Buttons sind dabei `Send` und `Receive` für das Senden und Empfangen einer Datei, deren Namen Sie wieder in einer File-Select-Box auswählen. Der Button `Binary Mode` schaltet Kermit in einen Binär-Modus, mit dem Sie Programme übertragen müssen, da bei ihm alle acht Bits der Zeichen für Daten verwendet werden können. Mit `Utilities` haben Sie auch hier die Hilfsfunktionen zur Verfügung. `Return to terminal emulation` bringt Sie wieder in den Terminalmodus zurück.

Die restlichen Buttons, die unter `Server Commands` zusammengefaßt sind, brauchen Sie nur, wenn der angerufene Rechner ein vollständiges Kermit-Protokoll beherrscht. Er arbeitet dann als ein von Ihnen steuerbares Datei-System, mit dem Sie Zugriff auf Directories, Programme und Dateien haben. Sie können so z.B. Programme starten oder Dateien löschen. Diese umfangreichen Kommandos finden Sie in einem ausführlichen Kermit-Handbuch oder in einer speziellen Anleitung des angerufenen Großrechners oder Rechenzentrum beschrieben. Für Hobby-Mailboxen werden Sie die Kommandos nur sehr selten brauchen.

Den Transfer mit XMODEM, YMODEM und Kermit können Sie jeweils mit Control+C abbrechen. UniTerm schickt dann die in dem Protokoll festgelegten Steuerzeichen, um auch dem angerufenen Rechner mitzuteilen, daß der Transfer abgebrochen werden soll.

Wenn Sie ein Modem benutzen, können Sie mit Alternate+`0` bis Alternate+`9` die in der Telefonliste festgelegten Nummern wählen lassen. Automatisches Auflegen und Beenden der Verbindung lösen Sie mit Alternate+`H` aus. UniTerm benutzt dazu das mit der Telefonliste eingegebene Modem-Kommando. Den Antwort-String, der in den Terminalparametern unter `Answerback String` festgelegt wird, schicken Sie mit Alternate+`A` ab.

Ein kurzes Break-Signal geben Sie mit Alternate+`B` aus. Dabei wird die DTR-Leitung an der seriellen Schnittstelle nicht ausgeschaltet. Mit Alternate+`L` schickt UniTerm ein langes Break-Signal und schaltet dabei DTR aus.

Das Verlassen von UniTerm ist mit der `Undo`-Taste möglich. Sie werden

dann sicherheitshalber noch gefragt, ob Sie das Programm auch wirklich verlassen wollen, was mit dem `Yes`-Button bestätigt werden muß. Das Kommando Alternate+`Undo` beendet UniTerm ohne Rückfrage.

Schließlich kennt UniTerm im Terminalmodus noch die Tasten `Help` zum Umschalten in den Kommandosmodus, `Insert` für den Single-Line-Editor und Alternate+`CapsLock` zum Umschalten des Meta-Modus.

Der Meta-Modus

Bei den Einstellungen für das emulierte Terminal kam die Sprache schon einmal auf ASCII-Codes mit sieben und acht Bits. Die normalen Tasten beim Atari ST erzeugen ASCII-Zeichen, deren Wert unter 128 liegt, bei denen also das achte Bit immer ausgeschaltet ist. Da aber in den letzten Jahren für ASCII-Zeichen meist acht Bit verwendet werden, kann es sein, daß Sie ein solches Zeichen erzeugen, sprich eingeben wollen.

UniTerm kennt dafür den "Meta-Modus". Er wird durch die Tastenkombination `Alternate+CapsLock` eingeschaltet und bewirkt, daß bei allen Zeichen, die an die serielle Schnittstelle geschickt werden, das achte Bit gesetzt ist.

Die Taste "A" ergibt normalerweise den ASCII-Wert 65. Wenn Sie den Meta-Modus eingeschaltet haben, gibt UniTerm bei Eingabe von Alternate+A das Zeichen 193 aus[6]. Bei eingeschaltetem Meta-Modus erscheint in der Statuszeile die Anzeige `Meta`. Ob Sie diese Option benötigen, hängt vom angerufenen Rechner ab. Üblicherweise verwenden Großrechner und Mailboxen einen "normalen" ASCII-Code, bei dem das achte Bit eigentlich keine Rolle spielt[7], aber Ausnahmen bestätigen eben die Regel.

Der Zeileneditor SLE

Manche Großrechner oder Mailboxen mit wenig Komfort bieten nicht sehr viele Möglichkeiten, Zeilen zu editieren. UniTerm hat daher einen eingebauten Editor, mit dem Sie die letzten zwanzig Zeilen bearbeiten können. Er wird mit der `Insert`-Taste aktiviert. Die unterste Bildschirmzeile zeigt dann eine der zwanzig Editorzeilen an. Jede gedrückte Taste wird an der momentanen Cursor-Position eingefügt. Löschen können Sie entweder mit

[6] Im binären Zahlensystem hat eine 1 beim achten Bit einen Wert von 128. Das erste Bit hat den Wert $2^0 = 1$, dementsprechend ist eine 1 an der achten Stelle gleich $2^7 = 128$. Das binäre Zahlensystem ist auf Computern ein grundlegendes Konzept.

[7] Bei Großrechnern ist auch der EBCDIC-Code verbreitet, ein von IBM geschaffener Standard. Bei solchen Rechnern ist aber in der Regel eine Konvertierung zum ASCII-Code möglich. Bei der Zeichencodierung treffen hier zwei Welten aufeinander: Die Großrecher – auch "Jumbos" genannt – und die Mikros oder PC's. UniTerm's Meta-Modus ist ein kleiner Ausweg aus diesem Dilemma.

`Delete` das Zeichen unter dem Cursor oder mit `Backspace` das Zeichen links davon.

Der Cursor kann mit ← und → nach links und rechts in der Zeile bewegt werden. Mit ↑ und ↓ erscheint die letzte oder nächste Zeile aus dem Editier-Puffer.

Verlassen können Sie den SLE aus zwei Arten. Mit `Return` wird die aktuelle Zeile über die serielle Schnittstelle abgeschickt und Sie befinden sich wieder im Terminalmodus. `Insert` beendet ebenfalls, allerdings ohne die Zeile abzuschicken.

Mausbenutzung im Terminalmodus

Auch wenn der Terminalmodus hauptsächlich mit der Tastatur benutzt wird, kommt bei UniTerm auch hier die Maus zum Einsatz. Wenn Sie die linke Maustaste drücken erscheint ein Textcursor als Mauszeiger und Sie können ihn mit der Maus auf eine Bildschirmposition verschieben. Drücken Sie nun die linke Maustaste erneut, schickt UniTerm Steuerzeichen an den angerufenen Rechner, um den Cursor auf diese Position zu setzen.

Die rechte Maustaste löst das "Popup"-Menü aus, in dem Sie mit der Maus aus zwanzig möglichen Kommandos auswählen können. Bewegen Sie dazu einfach die Maus, bis der gewünschte Befehl invertiert dargestellt ist. Mit einem Druck auf die linke Taste lösen Sie ihn aus. Um das Popup-Menü zu verlassen, müssen Sie den Mauszeiger außerhalb des Menüs positionieren und dann drücken. Sie können das Menü Ihren eigenen Wünschen anpassen. Den dazu nötigen Makro-Befehl `%POPUP` finden Sie in Kapitel 9.4 auf Seite 127 beschrieben.

Als letzte Mausaktion können Sie ein "Clipboard" benutzen. Dabei definieren Sie bei gedrückter linker Maustaste einen Bildschirmausschnitt, auf den dann drei Kommandos angewandt werden können. `Cut` legt ihn im Clipboard-Puffer ab; mit `Add` wird der Ausschnitt an den Puffer angefügt. Auf `Send` schickt UniTerm den Ausschnitt über die serielle Schnittstelle ab.

Den Inhalt des Clipboard-Puffers können Sie mit der Funktion `SaveClip` aus dem Popup-Menü auf Diskette oder Festplatte schreiben. Dazu müssen Sie in einer File-Select-Box einen Datei-Namen auswählen.

Terminalkommandos im Überblick

Hier nochmals alle Kommandos für den Terminalmodus im Überblick:

Taste	Kommando
Alternate+`F1`	Tektronix-Emulation einschalten

Alternate+F2	Zwischen 24 und 29 Zeilen umschalten
Alternate+F3	History auf GDOS-Station schreiben
Alternate+F4	Bildschirminhalt als Text ausdrucken
Alternate+F5	Auf Tektronix-Grafikschirm umschalten
Alternate+F6	VT100/102-Emulation einschalten
Alternate+F7	Terminal zurücksetzen
Alternate+F8	Protokolldruck ein- und ausschalten
Alternate+F9	Zoommodus einschalten
Alternate+F10	Zwischen 80 und 132 Zeichen pro Zeile umschalten
Alternate+A	Antwort-String senden
Alternate+B	Kurzes Break-Signal senden
Alternate+C	Diskettenprotokoll ein- und ausschalten
Alternate+H	Telefon auflegen

Taste	Kommando
Alternate+I	History ein- und ausschalten
Alternate+L	Langes Break-Signal senden
Alternate+O	History anschauen
Alternate+P	Bildschirminhalt im DEGAS-Format abspeichern
Alternate+R	Protokolldatei abspielen
Alternate+T	Datei-Transfer starten
Alternate+X	History abspeichern
Alternate+Y	Bildschirmausgabe anhalten
Alternate+0...1	Telefonnummern automatisch wählen
Undo	UniTerm mit Rückfrage beenden
Alternate+Undo	UniTerm ohne Rückfrage beenden
Alternate+CapsLock	Metamodus umschalten
Alternate+Help	Bildschirmhardcopy drucken
Insert	Single-Line-Editor starten
Help	Kommandomodus einschalten
F1...F10	Funktionstasten-Makros ausführen bzw. Text senden
⇑F1...⇑F10	Funktionstasten-Makros ausführen bzw. Text senden
linke Maustaste	Cursor setzen oder Ausschnitt für Clipboard definieren
rechte Maustaste	Popup-Menü starten

9.4 Die Makrosprache

Die eingebaute Makrosprache erlaubt eine gewisse Programmierung innerhalb von UniTerm. Sie können sehr viele Arbeitsabläufe damit vollständig automatisieren, z.B. Anwählen einer Mailbox, Einloggen, Mailboxkommandos absetzen, Dateien lesen und schließlich ausloggen.

Makrokommandos können in eigenen Makrodateien abgelegt werden, die beim Start von UniTerm oder beim Anwählen einer Mailbox automatisch ausgeführt werden. Sie sind aber auch in den programmierbaren Funktionstasten erlaubt. Alle Zeichen, die kein Makrokommando sind, werden als Ausgabe an die serielle Schnittstelle geschickt.

Jedes Makrokommando beginnt mit dem Zeichen `%`. Es besteht aus einem Kommandonamen und eventuell Parametern, die in runden Klammern anzugeben und durch Kommatas zu trennen sind.

Es gibt drei Parametertypen. Normale Zeichenketten müssen von einfachen Anführungszeichen (') begrenzt sein. Ich werde sie im folgenden mit *str* bezeichnen. Zahlen können im Bereich von 0 bis 32767 liegen – Sie finden sie als *i* gekennzeichnet. Der dritte Parametertyp sind Wahrheitswerte, bei denen Sie entweder `TRUE` für wahr oder `FALSE` für falsch angeben müssen. Für diesen Typ verwende ich *b*.

UniTerm hat vier Variablen, die Sie für Stringparameter einsetzen können und die durch verschiedene Makrokommandos verändert werden. Das sind:

`$PATH` Enthält den aktuellen Pfadnamen. Er wird durch die Auswahl in File-Select-Boxen oder das `%PATH`-Kommando verändert und ist anfangs auf den Ordner eingestellt, aus dem UniTerm gestartet wurde.

`$FILENAME` Enthält den zuletzt ausgewählten Datei-Namen. Er wird durch die Auswahl in einer File-Select-Box verändert.

`$CURRENT` Enthält den Pfadnamen, den GEMDOS momentan verwendet.

`$TEMP` Wird bei verschiedenen Makro-Kommandos verwendet.

Nun zu den Kommandos:

`%ABORT(`*str*`)`

Beenden der gerade verarbeiteten Makrodatei. Die Variable `$TEMP` wird auf *str* gesetzt. Da Makros geschachtelt werden können, lassen sich so Ergebnisse übergeben.

`%CONCAT(`*str1*`,`*str2*`)`

Hängt die Zeichenkette *str1* an *str2*. Das Ergebnis steht in `$TEMP`.

`%JUMP(`*i*`)`

Die Makrodatei wird ab der Zeile *i* weiterverarbeitet. Sie können damit Schleifen in den Makros bauen. Es gibt allerdings keinen Mechanismus, der Endlosschleifen abfängt.

`%LOADSETUP(`*str*`)`

Lädt die Setup-Datei mit dem Namen *str* vom Massenspeicher. Entspricht dem `Load Setup`-Befehl im `File`-Menü.

`%MAKRO(`*str*`)`

Führt die Makrodatei mit dem Namen *str* aus.

`%ONKEY()`

Ist bei Ausführung des Kommandos eine Taste gedrückt, so wird der Rest der Zeile ausgeführt. Wenn keine Eingabe vorliegt, arbeitet UniTerm bei der nächsten Zeile der Makrodatei weiter. Sie können damit Abbruchbedingungen für Schleifen programmieren. Soll eine Schleife abgebrochen werden, dann müßte ein entsprechendes Programm so aussehen:

```
%ECHO('Beginn der Schleife')
%ONKEY()%ECHO('unterbrochen...')%JUMP(4)
%ECHO('Schleife geht weiter')%JUMP(1)
%ECHO('Schleife beendet')
```

Die Schleife liegt in den Zeilen 1 bis 3 und wird durch das `%JUMP(1)` wiederholt. Nur bei einem Tastendruck führt UniTerm den Rest der zweiten Zeile aus und trifft auf das Kommando `%JUMP(4)`, womit die Schleife beendet wird. Wenn Sie in einer Schleife keine solche Abfrage vornehmen, erzeugen Sie Endlosschleifen, und das Makro wird niemals beendet.

`%SUSPEND()`

Gibt die Mitteilung `Press any key...` auf dem Bildschirm aus und wartet auf einen Tastendruck.

`%PATH(`*str*`)`

Setzt des GEMDOS-Pfad für Datei-Zugriff auf *str*. Entspricht dem Kommando `Set Path...` aus dem `File`-Menü. Die Variable `$PATH` können Sie so direkt setzen.

`%WAIT(`*i*`)`

Wartet *i* Zehntelsekunden. Um die Ausführung für zwei Sekunden zu unterbrechen, müßten Sie `%WAIT(20)` programmieren.

`%ASSERT()`

Schaltet die Leitung DTR an der seriellen Schnittstelle ein.

`%BREAK(`*i*`,`*b*`)`

Schickt ein Break-Signal für *i* Millisekunden über die serielle Schnittstelle und teilt somit dem angerufenen Rechner eine Unterbrechung mit. Wenn für *b* `TRUE` übergeben wird, schaltet UniTerm die DTR-Leitung für diese Zeit aus.

`%DROP()`

Schaltet die DTR-Leitung an der seriellen Schnittstelle aus.

`%ECHO(`*str*`)`

Gibt die Zeichenkette *str* auf dem Bildschirm aus. Die Zeichen werden nicht an die serielle Schnittstelle geschickt. Im obigen Beispiel haben Sie gesehen, wie man so Mitteilungen über die Makroaktionen ausgeben kann.

`%ECHO(`*i*`)`

Gibt das Zeichen mit dem ASCII-Wert *i* auf dem Bildschirm aus. Sie können so auch Steuerzeichen in Bildschirmmitteilungen verwenden. Beim VT-52 Terminal ist die Steuersequenz zum Löschen des Bildschirms ESC-E. Eine Begrüßungsmeldung auf einem leeren Bildschirm müßten Sie dann so programmieren:

```
%ECHO(27)%ECHO('E')%ECHO('Willkommen beim Hackermakro !')
```

`%GET(`*str*`,`*i*`)`

Wartet auf die Eingabe der Zeichenkette *str*. *i* gibt an, wieviele Sekunden UniTerm warten soll. Wenn der Benutzer die richtige Eingabe macht, wird der Rest der Zeile verarbeitet. Sie können auf die Weise wieder Schleifenbedingungen programmieren.

`%SEND(`*str*`)`

Die Zeichenkette *str* wird über die serielle Schnittstelle ausgegeben.

`%SEND(`*i*`)`

Das Zeichen mit dem ASCII-Wert *i* wird an der seriellen Schnittstelle ausgegeben. Diese Funktion ist besonders wichtig, wenn Sie Mailboxkommandos absetzen wollen. Normalerweise wird jeder Befehl durch ein `Return` abgeschlossen, was den ASCII-Wert 13 (Steuerzeichen CR) abschickt. Ein solches Kommando müßte also so programmiert werden:

```
%SEND('read xmodem file testfile.dat')%SEND(13)
```

Das Mailboxkommando `read xmodem file testfile.dat` wird dann korrekt mit einem Return ausgelöst. Gerade bei der Programmierung der Funktionstasten, auf die Sie wahrscheinlich oft benutzte Mailboxbefehle legen werden, ist es so möglich, auch mehrere Kommandos mit einem Makro abzusetzen.

`%RUN(`*str1*`,`*str2*`)`

Führt das Programm mit dem Namen *str1* aus. *str2* wird dabei als Kommandozeile übergeben, die einige Programme verwenden können. Handelt es sich um ein `.TTP`-Programm und ist *str2* leer, fragt UniTerm wie der Desktop in einer Dialogbox nach der Kommandozeile.

`%INPUT(`*str*`)`

Erwartet eine Eingabe vom Benutzer. Dabei wird *str* als Mitteilung auf dem Bildschirm ausgegeben. Die Eingabe findet in einer Dialogbox statt und befindet sich nach dem Kommando in der `$TEMP`-Variablen. Der Rest der Makrozeile wird nur dann ausgeführt, wenn der Benutzer die Eingabe mit dem `OK`-Button bestätigt hat.

`%INLINE(`*b*`)`

Arbeitet ähnlich `%INPUT`, nur wird eine Zeile ganz normal im Terminalmodus eingegeben. Die Eingabe wird mit `Return` beendet und kann maximal achtzig Zeichen lang sein. Das Ergebnis steht danach in `$TEMP`. Wenn für *b* `TRUE` eingesetzt ist, werden alle Eingaben auf dem Bildschirm als Echo ausgegeben.

`%FILESELECTOR(`*str1*`,`*str2*`,`*str3*`)`

Bewirkt die Anzeige einer File-Select-Box und läßt den Benutzer einen Datei-Namen auswählen. *str1* und *str2* sind dabei der voreingestellte Pfad und Datei-Name. *str3* wird als Mitteilung über der File-Select-Box angezeigt. Der ausgewählte Datei-Name steht nach dem Kommando in `$FILENAME`; der ausgewählte Pfad in `$PATH`.

Ebenso wie bei `%ONKEY`, `%GET` und `%INPUT` führt UniTerm den Rest der Zeile nur dann aus, wenn die File-Select-Box mit `OK` verlassen wurde.

`%UNICOMMAND(`*i*`)`

Fast jedes Kommando von UniTerm kann mit diesem Befehl in einem Makro ausgelöst werden. Jeder UniTerm-Aktion ist eine Funktionsnummer zugeordnet, die Sie für *i* eingesetzen müssen. Die Funktionen und ihre Kennzahlen sind:

Funktion	Nummer	Tastenkombination im Terminalmodus
ResetTek	1	Alternate+F1
VDIOutput	2	Alternate+F3
PrintTextScreen	3	Alternate+F4
TekMode	4	Alternate+F5
TextMode	5	Alternate+F6
Reset	6	Alternate+F7
AutoPrint	7	Alternate+F8
Zoom	8	Alternate+F9
132ColumnToggle	9	Alternate+F10
ScrollLock	10	Alternate+Y
49LineToggle	11	Alternate+F2
SendAnswerBack	12	Alternate+A
ShortBreak	13	Alternate+B
DropDTR	14	
LongBreak	15	Alternate+L
SaveHistory	16	Alternate+X
ControlHistory	17	Alternate+O
Switch	18	
ControlCapture	19	Alternate+C
PlayBack	20	Alternate+R
SendFile	21	Alternate+T
DegasSave	22	Alternate+P
ReplayHistory	23	
Hangup	24	

Funktion	Nummer	Tastenkombination im Terminalmodus
Dial1	25	Alternate+1
Dial2	26	Alternate+2
Dial3	27	Alternate+3
Dial4	28	Alternate+4
Dial5	29	Alternate+5
Dial6	30	Alternate+6
Dial7	31	Alternate+7
Dial1	32	Alternate+8
Dial9	33	Alternate+9
Dial10	34	Alternate+0

SetPath	35	
DelFile	36	
DiskSpace	37	
F1	38	F1-Makro
F2	39	F2-Makro
F3	40	F3-Makro
F4	42	F4-Makro
F5	42	F5-Makro
F6	43	F6-Makro
F7	44	F7-Makro
F8	45	F8-Makro
F9	46	F9-Makro
F10	47	F10-Makro
SF1	48	⇑F1-Makro
SF2	49	⇑F2-Makro
SF3	50	⇑F3-Makro
SF4	51	⇑F4-Makro
SF5	52	⇑F5-Makro
SF6	53	⇑F6-Makro
SF7	54	⇑F7-Makro
SF8	55	⇑F8-Makro
SF9	56	⇑F9-Makro
SF10	57	⇑F10-Makro
Utilities	58	
ToggleMeta	59	
Help	60	Help

Nehmen wir an, eine Makrodatei soll den Bildschirm auf 49 Zeilen umschalten, die erste Mailbox aus der Telefonliste anrufen und alle Ausgaben auf dem Drucker protokollieren.
Die entsprechenden Kommandos wären:

```
%UNICOMMAND(11)
%UNICOMMAND(7)
%UNICOMMAND(25)
```

Damit läßt sich die Arbeit mit UniTerm weitestgehend automatisieren. Es sind so Mailbox-Sitzungen denkbar, bei denen Sie keine einzige Taste drücken müssen und per Makrodatei UniTerm konfigurieren, Datenbanken anrufen, dort automatische Datei-Transfers auslösen und schließlich die

Verbindung korrekt beenden. Lediglich für das Verlassen von UniTerm ist kein Kommando vorhanden.

`%REASSIGN(`*i1*`,`*i2*`)`

Falls Ihnen die Zuordnung von Kommandos im Terminalmodus und den Tastenkombinationen nicht gefällt oder Sie eine andere Zuordnung gewohnt sind, lassen sich die Befehle auf andere Tasten legen.

i1 bezeichnet die Taste, die zusammen mit Alternate gedrückt werden muß, um einen Befehl auszulösen. *i2* ist die Kennung der Funktion nach obiger Tabelle.

Zur Bezeichnung der Taste verwendet UniTerm die "Scan-Codes", die nicht dem ASCII-Wert der Taste entsprechen, sondern die Position auf der Tastatur angeben. Den benötigten Wert müssen Sie sich nach der Abbildung auswählen:

Esc	1	2	3	4	5	6	7	8	9	0	ß	'	#	Backspace
	2	3	4	5	6	7	8	9	10	11	12	13	41	

Tab	Q	W	E	R	T	Z	U	I	O	P	Ü	+		Delete
	16	17	18	19	20	21	22	23	24	25	26	27		

Control	A	S	D	F	G	H	J	K	L	Ö	Ä	Return	~
	30	31	32	33	34	35	36	37	38	39	40		43

Shift	<	Y	X	C	V	B	N	M	,	.	-	Shift
	96	44	45	46	47	48	49	50	51	52	53	

Eine mögliche Programmzeile wäre:

```
%REASSIGN(32,37)
```

Damit stellt UniTerm die Funktion "DiskSpace" auf Alternate+D auch im Terminalmodus zur Verfügung. 32 ist der Scancode der Taste "D" und 37 die Funktionsnummer von "DiskSpace". Auf diese Weise können Sie zusätzliche Befehle per Tastendruck benutzen, die in der normalen UniTerm-Einstellung nicht vorgesehen sind.

`%POPUP(`*i1*`,`*i2*`,`*str*`)`

Das Popup-Menü, das im Terminalmodus nach Drücken der rechten Maustaste erscheint, können Sie ebenfalls programmieren. Es bietet für zwanzig Einträge Platz, deren Positionen von 1 bis 20 durchnummeriert sind. Diese Position müssen Sie in *i1* einsetzen; das durch die Auswahl ausgelöst Kommando kommt in *i2*. In *str* geben Sie den Text an, der im Popup-Menü erscheinen soll. Die Makrozeile

```
%POPUP(1,7,'Drucke')
```

macht Ihnen die Steuerung des Protokolldruckers auf der ersten Position im Popup-Menü unter dem Eintrag `Drucke` zugänglich.

`%DIAL(`*i*`)`

Wählt die Telefonnummer *i* aus der Telefonliste.

`%LOADTEL(`*str*`)`

Lädt die Telefonliste, die sich unter dem Namen *str* auf dem Massenspeicher befindet. Sie muß vorher mit dem `Save Numbers`-Befehl aus dem `File`-Menü angelegt worden sein.

`%KERMIT(`*str1*`,`*b*`,`*str2*`)`

Startet einen Kermit-Datei-Transfer. *str2* enthält einen Datei-Namen, der je nach Transferart auch die Wildcards `?` und `*` enthalten kann. *str1* bestimmt den Übertragungsmodus:

`SEND` Dateien senden.

`REC` Eine oder mehrere Dateien empfangen.

`GET` Eine oder mehrere Dateien empfangen, wobei sich der angerufene Rechner im Server-Modus befindet.

b bestimmt die Art der übertragenen Daten. Bei `TRUE` werden alle acht Bits pro Byte verwendet; bei `FALSE` ist die Übertragung auf ASCII-Text mit sieben Bits eingestellt.

`%XMODEM(`*str1*`,`*str2*`)`

Löst einen XMODEM-Transfer aus. *str1* bestimmt die Übertragungsrichtung mit `SEND` zum Senden und `REC` zum Empfangen einer Datei. Deren Namen müssen Sie bei *str2* einsetzen.

`%YMODEM(`*str1*`,`*str2*`)`

Startet eine Übertragung nach dem YMODEM-Protokoll. Die Parameter haben dabei die gleiche Bedeutung wie beim `%XMODEM`-Befehl.

Alle Makrobefehle lassen sich auch abkürzen. Dabei müssen Sie allerdings soviele Buchstaben eingeben, daß das Kommando eindeutig zu identifizieren ist. Für das Kommando `%SEND` ist die Abkürzung `%S` ausreichend. Bei `%LOAD` kann UniTerm nicht entscheiden, ob `%LOADTEL` oder `%LOADSETUP` gemeint ist.

Die Abkürzungen haben den Vorteil, daß weniger Platz benötigt wird und UniTerm die Kommandos schneller ausführen kann. Sie sollten also die kurzen Versionen der Befehle sooft wie möglich verwenden.

Chapter 10

Datei-Kompression mit ARC

In diesem Kapitel werden Sie ein Programm kennenlernen, das Sie bei Public-Domain Software unbedingt brauchen werden: Das Komprimierungs- und Verschlüsselungsprogramm ARC.

Public-Domain Software nimmt oft viel Platz ein. Programme sind teilweise schon über 100 KByte groß; für die Dokumentationen sind 50 KByte keine Seltenheit. Wenn Sie sich überlegen, auf welchen Wegen Public-Domain Programme verbreitet werden, bedeutet das, daß immer weniger Software auf eine Diskette paßt, oder eine Übertragung mit Modem Stunden dauern kann.

Die Mathematik und Informatik haben mehrere Methoden entwickelt, Daten zu packen, ohne daß Informationen verloren gehen. Die Komprimierung beruht darauf, daß die meisten Daten Redundanzen enthalten, also mehr Informationseinheiten belegen, als eigentlich nötig. Denken Sie nur an deutsche Texte. Zwei der am häufigsten vorkommenden Wörter sind z.B. "der" oder "die". Im Computer belegen sie jeweils drei Bytes. Würden wir z.B. "D" als "der" und "d" als "die" lesen, hätten wir schon eine Menge Platz gespart. Bei Programmen gibt es ähnliche oft verwendete Sequenzen.

Eine menschliche Eigenart ist es, viele dieser Redundanzen zu verwenden. Um Informationen nun platzsparend zu speichern, braucht man ein Programm, das die angesprochenen Muster sucht und sie durch kürzere ersetzt. Damit die Informationen wieder nutzbar werden, muß es ein anderes Programm geben, das diesen Vorgang wieder rückgängig macht. Und zwar so, daß wieder die ursprünglichen Daten entstehen.

Das Public-Domain-Programm ARC von Tom Henderson übernimmt

genau diese Aufgabe. Wenn Dateien übertragen oder kopiert werden sollen, packt man sie auf eine kleinere Größe zusammen und spart so Zeit bei der Übertragung. Zur Nutzbarmachung der Dateien muß das vorher entstandene Archiv wieder "ausgepackt" werden.

ARC hat sich zu einem Standard entwickelt. Es gibt kompatible Implementationen für MS-DOS, Amiga und den Atari ST. ARC kann Dateien auf bis zu 40% ihrer ursprünglichen Größe reduzieren. Beim Packen wird ein Archiv (mit der Kennung .ARC) angelegt, das mit verschiedenen Kommandos erweitert, gelesen, entpackt und auch verschlüsselt werden kann. Wenn Sie viel mit Public-Domain Programmen umgehen wollen, ist der Umgang mit ARC unabdingbar.

10.1 Aufruf von ARC

Sie können ARC vom Desktop aus als `.TTP` Programm oder mit einer beliebigen Kommando-Shell starten. Der Aufruf hat die folgende Form:

```
arc.ttp {amufdxerplvtc}[bswnh][g<password>] <archiv> [{filename}]
```

In der ersten Gruppe in geschweiften Klammern stehen die ARC bekannten Kommandos. An sie können Optionen (die Buchstaben in eckigen Klammern) angehängt werden, wobei bei Verschlüsselung noch das Password hinzukommt. Dann folgt der Name des Archivs, das erzeugt oder bearbeitet werden soll. Falls Sie keine Datei-Kennung angeben, fügt ARC automatisch `.ARC` hinzu. Schließlich folgt eine Liste von Datei-Namen oder eine Datei-Maske.

Als Beispiel dient ein Directory namens `PROFF` (es handelt sich um den Textformatierer), dessen Files gepackt werden sollen:

```
 Directory Listing of PROFF\*.*

-   1531  04.01.88  02:17  MAKROS.PRF
-  34048  20.12.87  18:02  PROFF.PRG
-  47799  20.12.87  18:02  PROFFMAN.PRF
-   3840  20.12.87  18:02  PROFFSYM.NEW
-   1536  20.12.87  18:02  READ.ME
-    799  20.12.87  18:02  README
-   2586  20.12.87  18:03  SEE.ME
```

Die folgenden Beispielaufrufe von ARC sind bei der Benutzung einer Kommando-Shell gültig. Falls Sie die Beispiele vom Desktop aus anwenden wollen, sollten Sie den Buchstaben `h` an die Kommandokürzel anhängen. Diese Option bewirkt, daß ARC nach Ausführung auf einen Tastendruck wartet, damit

die Ausgaben nicht sofort vom Desktop überschrieben werden. Auch dürfen Sie den Programmnamen "arc" nicht als ersten Parameter in der Dialogbox des Desktops übergeben.

Als Erstes soll ein neues Archiv namens `PROFF.ARC` erzeugt werden. Dazu müssen Sie den `a`-Befehl (für "add") verwenden. Der entsprechende Aufruf lautet:

```
arc a proff proff\*.*
```

Er besagt, daß ARC alle Dateien, auf die die Maske `PROFF\*.*` paßt, in das Archiv `PROFF.ARC` packen soll. Da es noch nicht existiert, erzeugt ARC es neu. Beim Programmlauf gibt ARC folgende Meldungen auf dem Bildschirm aus:

```
Creating new archive: PROFF.ARC

Adding file:   MAKROS.PRF     analyzing, crunching, done.
Adding file:   PROFF.PRG      analyzing, crunching, done.
Adding file:   PROFFMAN.PRF   analyzing, crunching, done.
Adding file:   PROFFSYM.NEW   analyzing, crunching, done.
Adding file:   READ.ME        analyzing, crunching, done.
Adding file:   README         analyzing, crunching, done.
Adding file:   SEE.ME         analyzing, crunching, done.
```

Nun sind alle Files aus dem Directory gepackt. Sie können das mit dem `l`-Befehl überprüfen und alle Files aus dem Archiv auflisten:

```
arc l proff *.*
```

Die Maske `*.*` können Sie hierbei auch weglassen. ARC zeigt Ihnen allerdings nur die Namen der gepackten Dateien an. Interessanter ist das ausführliche Inhaltsverzeichnis, das mit dem `v`-Kommando ausgegeben wird.

```
arc v proff
```

ergibt folgende Bildschirmausgabe:

```
Name          Length   Storage    SF   Size now  Date      Time   CRC
============ ======== ========  ====  ========  ========= ====== ====
MAKROS.PRF       1531 Crunched   44%       861   4 Jan 88 2:17a  4D62
PROFF.PRG       34048 Crunched   30%     24110  20 Dec 87 6:02p  A3E7
PROFFMAN.PRF    47799 Crunched   52%     23347  20 Dec 87 6:02p  6F45
PROFFSYM.NEW     3840 Crunched   49%      1968  20 Dec 87 6:02p  68DF
READ.ME          1536 Crunched   35%      1008  20 Dec 87 6:02p  AC06
README            799 Crunched   30%       562  20 Dec 87 6:02p  F387
SEE.ME           2586 Crunched   39%      1603  20 Dec 87 6:03p  0867
```

```
        ==== ========             ==== ========
Total      7    92139             42%    53459
```

ARC gibt nun auch Informationen über die ursprüngliche Größe der Dateien, ihre Speicherungsart, die Platzeinsparung und die jetzige Größe im Archiv an. Schließlich erfahren Sie noch das Erstellungsdatum der Dateien und eine Prüfsumme, die allerdings für Sie von wenig Nutzen ist.

Kommen wir zurück zur Archivierung von Dateien: Beim `a`-Befehl werden die Files gepackt und in das Archiv gespeichert, bleiben aber noch in ihrer ursprünglichen Form bestehen. Falls Sie das nicht wollen, können Sie den `m`-Befehl ("move") benutzen, der die zu packenden Dateien nach der Archivierung löscht:

```
arc m proff proff\*.*
```

Wenn Sie ein Archiv zum platzsparenden Backup Ihrer Dateien verwenden, kann ARC geänderte Files mit dem `u`-Kommando ("update") automatisch übernehmen:

```
arc u proff proff\*.*
```

Alle Dateien aus PROFF, die nach der letzten Archivierung geändert wurden, übernimmt ARC in der aktuellen Version in das Archiv. Neue Dateien werden zusätzlich aufgenommen. Falls wirklich nur geänderte Dateien gesichert werden sollen, kann ARC mit dem Kommando `f` neu hinzugekommene Files ignorieren.

Das "auspacken" der Dateien aus einem Archiv geschieht mit dem `e`-Kommando ("Extract"). Anstelle von `e` kann auch `x` verwendet werden. Die angegebene Datei-Maske gibt nun an, welche Dateien aus dem Archiv geholt werden sollen. Im Beispiel sieht das so aus:

```
arc e proff *.*
```

ARC produziert beim Auspacken die folgende Bildschirmausgabe:

```
Extracting file: MAKROS.PRF
Extracting file: PROFF.PRG
Extracting file: PROFFMAN.PRF
Extracting file: PROFFSYM.NEW
Extracting file: READ.ME
Extracting file: README
Extracting file: SEE.ME
```

Bei einer Maske `*.PRF` würde ARC nur die Files `MAKROS.PRF` und `PROFFMAN.PRF` entpacken.

Es ist auch möglich, eine (Text-) Datei aus einem Archiv auf dem Bildschirm anzuzeigen:

```
arc p proff read.me
```

Ebenso können Sie ein Programm direkt aus einem Archiv starten:

```
arc r proff proff.prg proffman.prf
```

führt das Programm `PROFF.PRG` direkt aus. Dabei wird der Parameter `proffman.prf` an das Programm weitergegeben. Eine kleine Einschränkung hat dieser Befehl bei GEM-Programmen, die ein Resource-File benutzen: Der Name dieser Datei muß bis auf die Endung `.RSC` mit dem des Programms identisch sein.

Die nächsten drei Kommandos dienen zur Bearbeitung des Archivs. Löschen können Sie eine Datei aus dem Archiv mit `d`:

```
arc d proff read*.*
```

ARC löscht die Files `README` und `READ.ME` aus dem Archiv. Sie können nicht wiederhergestellt werden und sind verloren. Seien Sie daher bei den Namensmasken vorsichtig mit `*` und `?`.

Bei der Übertragung von Archiven oder beim Kopieren können immer Fehler auftreten. Es gibt daher den `t`-Befehl, der ein Archiv überprüft. Er testet die korrekte Struktur des Inhaltsverzeichnis und der einzelnen Einträge sowie deren Prüfsummen.

Da ARC verschiedene Packmethoden beherrscht und immer die günstigste auswählt, kann es passieren, daß Sie ein Archiv kopieren, das noch mit einer älteren Version von ARC erstellt wurde. Der `c`-Befehl überprüft alle Dateien und packt sie mit einer anderen Methode, falls das die Platzeinsparung erhöht. In der Regel brauchen Sie diesen Befehl aber nicht.

10.2 Optionen

An den Befehlsbuchstaben können Sie eine oder mehrere Optionen anhängen. Diese werden ebenfalls durch einen Buchstaben repräsentiert.

`b` Falls ein Archiv verändert wird, also Dateien neu hinzukommen, erneuert oder gelöscht werden, bleibt die alte Archiv-Version mit der Kennung `.BAK` auf der Diskette oder Platte erhalten.

`s` Die Dateien werden in ihrer ursprünglichen Form in das Archiv übernommen und nicht gepackt.

`w` ARC gibt keine Warnungen aus. Normalerweise fragt ARC z.B. dann bei Ihnen nach, wenn Sie eine Datei mit `a` einem Archiv hinzufügen wollen, das bereits einen Eintrag unter demselber Namen enthält.

`n` ARC gibt gar keine Meldungen aus.

`h` Nach Ausführung des Kommandos wartet ARC auf die Eingabe eines beliebigen Tastendrucks. Diese Option sollten Sie immer verwenden, wenn Sie ARC vom Desktop aus benutzen. Andernfalls werden alle Meldungen von ARC sofort wieder von den Fenstern überschrieben, was z.B. eine sinnvolle Benutzung des `l`-Befehls unmöglich macht.

Als Beispiel soll das PROFF-Directory gepackt (`a`-Kommando) und die alte Version des Archivs erhalten bleiben (Option `b`). ARC soll dabei keine Warnungen ausgeben (Option `w`) und nach der Abarbeitung des Kommandos auf einen Tastendruck warten (`h`-Option). Die Kommandozeile lautet dafür:

```
arc abwh proff proff\*.*
```

Daneben gibt es noch eine weitere Option zur Verschlüsselung von Dateien. Dabei muß als letzter Buchstabe ein `g` an die Kommando- und Optionenauswahl gehängt werden. Direkt darauf folgt ein Schlüssel, mit dem ARC die Dateien unlesbar macht. Ein Beispiel:

```
arc agrobert proff proff\mymacs.prf
```

Damit wird die Datei `MYMAC.PRF` wie üblich in das Archiv `PROFF` gepackt. Nur wird sie dabei noch mit dem Codewort "robert" verschlüsselt. Ein weiterer Zugriff auf die gepackte Datei ist nur unter Angabe des geheimen Schlüssels möglich. Die Dateien werden zwar bei dem `l`-Kommando angezeigt allerdings kann niemand, der das Kennwort nicht kennt auf sie zugreifen. Zum Entpacken wäre also

```
arc egrobert proff mymacs.prf
```

notwendig. Es gibt keinen Weg, das Schlüsselwort aus dem Archiv auszulesen oder es zu "knacken", da es nicht mitgespeichert wird. Wenn Sie also diese Option für Ihre Dateien verwenden und das Kennwort vergessen, sind die Daten für Sie verloren. (Außer, Sie finden das Codewort durch Zufall heraus. Die Zeit, die Sie dafür brauchen, ist aber – wie bei allen etwas intelligenteren Verschlüsselungsmethoden – wahrscheinlich sehr hoch.)

Die Komprimierungsroutinen von ARC arbeiten sehr gut. In der englischsprachigen Originalanleitung werden folgende Durchschnittswerte angegeben:

Datei-Art	Einsparung
Textdateien	45–55%
Programmdateien	25–30%
Bilddateien	45–55%

Wie Sie sehen, lohnt sich ein ARC-Lauf – und die hervorragenden Einsparungen sind der Grund dafür, daß sich das Programm als Standard durchsetzen konnte.

Natürlich ist ARC nicht blitzschnell. Da viele Zugriffe auf den Massenspeicher notwendig sind (ARC erzeugt ein Zwischenfile, das hinterher gelöscht wird), wird der Programmlauf durch Verwendung eines Diskettenlaufwerks gebremst. Es lohnt sich durchaus, ARC und das Archiv in eine RAM-Disk oder auf die Festplatte zu kopieren und dort zu bearbeiten.

10.3 Speichermethoden

Ich will Ihnen nun noch einen kleinen Überblick über die verschiedenen von ARC benutzten Packungsmethoden geben. Allerdings ist die genaue Arbeitsweise der jeweiligen Algorithmen zu kompliziert, als daß hier Platz genug für eine Erläuterung wäre.

Die einfachste Speicherungsmethode von ARC können Sie mit der `s`-Option erzwingen: Hier wird überhaupt nicht gepackt, sondern die Datei in ihrer ursprünglichen Form in das Archiv kopiert. Beim `V`-Kommando erscheint als Speicherungsform einer Datei `--`.

Bei der ersten richtigen Komprimierung werden einfach Zeichen, die mehrmals aufeinanderfolgen gepackt zu dem Zeichen und einer Angabe, wie oft es vorkommt. Aus "aaaaaa" würde – grob gesagt – "a6". Im Archiv-Inhaltsverzeichnis wird die Methode als `Packed` vermerkt.

Beim Huffman-Packen wird für Zeichen, die am meisten in der Datei vorkommen ein (kürzerer) Bitvektor erzeugt, so daß das am häufigsten vorkommende nur noch ein Bit anstatt acht belegt. Für Informatiker: Es wird ein Baum erzeugt, der nach der Häufigkeit der Zeichen so geordnet ist, daß der Weg zum häufigsten Zeichen am kürzesten ist. Beim `V`-Befehl gibt ARC `Squeezed` an.

Die dritte Speicherungsmethode wird Lempel-Zev-Komprimierung genannt. Hierbei werden bestimmte Muster kürzer dargestellt, wobei z.B. "ABC" ein Muster wie "XYZ" hat. "B" ist der auf "A" im Alphabet folgende Buchstabe, und nach diesem Muster folgt "Y" auf "X". Wird dieses Muster mit der Kennzahl 1 bezeichnet, speichert ARC im Archiv nur noch ab, daß das Muster 1 beginnend mit "A" oder "X" in der ursprünglichen Datei stand. Im Inhaltsverzeichnis erscheint bei einem solchen Archiv-Eintrag `crunched`.

Bei der letzten Methode wechseln dieses Muster während des Packens, weshalb sie Dynamische Lempel-Zev Komprimierung heißt. Beim `V`-Kommando erscheint als Speicherungsart ein `Crunched` (großgeschrieben !). Diese Methode scheint die effizienteste zu sein: ARC verwendet sie meistens.

Aber diese Informationen nützen Ihnen nur dann etwas, wenn Sie sich mit Komprimierungsverfahren beschäftigt haben. Darum zurück zur praktischen Anwendung.

10.4 Komfort mit ARCSHELL

Wenn Ihnen die Eingabe der Befehlszeile oder die Benutzung einer Kommandoshell zu lästig ist, können Sie auch das Programm ARCSHELL verwenden, bei dem Sie nur noch die gewünschte Aktion und die Files mit der Maus auswählen müssen.

Nach dem Programmstart werden Sie in einer ersten File-Select-Box gefragt, wo sich Ihr ARC-Programm befindet. Danach erscheint eine große Dialogbox, mit der Sie ARCSHELL steuern. Sie ist in zwei Hälften geteilt. Die linke dient zur Auswahl der ARC-Kommandos, wobei immer das weiß unterlegte als selektiert gilt. Die Buttons lösen folgende Aktionen aus:

Button	ARC-Aufruf
`ADD to archive`	Ausführen des A-Kommandos
`MOVE to archive`	Ausführen des M-Kommandos
`UPDATE in archive`	Ausführen des U-Kommandos
`FRESHEN in archive`	Ausführen des F-Kommandos
`DELETE from archive`	Ausführen des D-Kommandos
`EXTRACT from archive`	Ausführen des E- bzw. X-Kommandos
`RUN from archive`	Ausführen des R-Kommandos
`COPY to StdOut`	Ausführen des C-Kommandos
`List in archive`	Ausführen des L-Kommandos
`VERBOSE LIST`	Ausführen des V-Kommandos

In der rechten Hälfte können Sie die Optionen setzen. Die ausgewählten (erscheinen weiß unterlegt) werden mit ihrem Buchstaben beim ARC-Aufruf an das Kommando angehängt. Die Buttons entsprechen folgenden Optionen:

Button	ARC Option
`Retain Backup`	B-Option
`Suppress Compression`	S-Option
`SuppressWarning Messages`	W-Option
`Suppress Notes`	N-Option
`Hold Screen`	H-Option
`Encrypt / Decrypt`	G-Option

Auslösen können Sie den ARC-Aufruf mit dem `Do It`-Button (`Exit` beendet ARCSHELL). ARCSHELL verlangt dann noch die fehlenden Informationen von Ihnen. In einer ersten File-Select-Box müssen Sie das Archiv auswählen, in einer zweiten geben Sie den gewünschten Filenamen ein, wobei Sie auch die Wildcards `*` und `?` verwenden können. Die Auswahl von `ABBRUCH` in den File-Select-Boxen beendet ARCSHELL übrigens. Hatten Sie Encrypt/Decrypt ausgewählt folgt nun noch die Frage nach dem Schlüssel.

Dann setzt ARCSHELL einen Aufruf von ARC an GEMDOS ab und übergibt die Kommandozeile, die es entsprechend Ihren Angaben zusammensetzt. ARC führt das Kommando aus und es erscheint wieder die Dialogbox. Gerade wenn Sie mehrere Archive bearbeiten wollen, oder sehr lange Pfadnamen benutzen, ist dieses Programm bei der Arbeit nützlich, da die Auswahlen mit der Maus sicherlich weniger (tipp-) fehleranfällig sind.