

Einstieg in die Shell (Folge 3)

Goddies vom System

von Hannes Rügheimer und
Christian Spanik

Amiga-DOS organisiert den Datentransfer zwischen Computer und Massenspeichermedien bzw. Geräten, die Daten bekommen oder welche abgeben. Nicht immer ist das beteiligte »Gerät« so einfach erkennbar wie beim Diskettenlaufwerk oder Drucker. Da gibt es z.B. noch zwei ultraschnelle Speicherlaufwerke oder ein Gerät, das Texte vorliest. Was halten Sie von einer Erweiterung, die Daten von einem Anwendungsprogramm zum nächsten schickt? Sie brauchen für diese und weitere Geräte keinen Pfennig mehr zu zahlen, denn sie gehören Ihnen schon.

Der Trick ist einfach. Für jedes an den Computer angeschlossene Peripheriegerät braucht man (Treiber-)Software, um die Hardware zu steuern bzw. die Daten für die Ausgabe entsprechend aufzubereiten. Und Software ist ja flexibel. Man könnte z.B. einen Gerätetreiber entwickeln, der als Buchstaben kodierte Notenfolgen so in Steuersignale für ein Diskettenlaufwerk umwandelt, daß der darin befindliche Motor die entsprechenden Tonfrequenzen erzeugt und das Laufwerk damit das Lied abspielt.

■ Zurück zur Praxis: Die RAM-Disk gibt es beim Amiga schon lange. Die dazugehörige Software simuliert ein Diskettenlaufwerk im Arbeitsspeicher des Computers. Dadurch ergeben sich kurze Speicher- und Ladezeiten, weil lediglich Daten im Arbeitsspeicher kopiert werden und kein bißchen Mechanik diesen Vorgang verzögert. Der entscheidende Nachteil der RAM-Disk: Nach dem Ausschalten des Computers, sogar schon nach einem Reset ist der Inhalt unwiederbringlich verloren. Deshalb ist eine RAM-Disk kaum der geeignete Aufenthaltsort für wichtige, ungesicherte Daten. Bestenfalls temporäre Dateien und Kopien von Daten, die an anderer Stelle sicher gespeichert sind – nämlich auf Diskette, Festplatte oder Wechselplatte – sollten Sie in der RAM-Disk ablegen.

Sie kennen jetzt die grundlegende Arbeitsweise der Shell. Zum Desert servieren wir Ihnen ein paar besonders schmackhafte Kleinigkeiten. Unsere Devise: Zeit sparen, Speicher sparen, Geld sparen.

Auf dem Amiga gibt es zwei verschiedene RAM-Diskettenlaufwerke. Neben der normalen mit der Bezeichnung RAM, die automatisch immer zur Verfügung steht, läßt sich eine resetfeste Variante »RAD« nachträglich installieren. Wo liegt der Unterschied zwischen beiden?

Im Gegensatz zur Standard-RAM-Disk überleben die in der RAD abgelegten Dateien zumindest einen Reset bzw. Neustart (über <Ctrl Amiga_links Amiga_rechts>) des Computers.

unterschiedlich. Betrachten wir zunächst mal die 2.04: Öffnen Sie ein Shell-Fenster und geben Sie

mount rad:
ein. (Der Doppelpunkt dient wie immer zur Kennzeichnung von Geräte- bzw. Datenträgerbezeichnungen und ist nicht Teil des Namens.)

Anschließend können Sie mit

dir rad:
prüfen, ob es das neue Laufwerk nun wirklich gibt. Weil es bisher noch keinen Inhalt hat, gibt DIR allerdings nichts aus. Dafür er-

1,5 MByte freiem Arbeitsspeicher so nicht angelegt werden. Es ist zwar bei geringem Speicherausbau ohnehin nicht ratsam, mit der resetfesten RAM-Disk zu arbeiten, aber zumindest für experimentelle Zwecke zeigen wir Ihnen, wie Sie die Größe der RAD anpassen können. Sollten Sie sie bereits mit MOUNT eingebunden haben, wirken sich die folgenden Änderungen erst nach einem Aus- und Wiedereinschalten des Computers sowie einem erneuten MOUNT der RAD aus.

Anwender von OS 2.04 geben im Shell-Fenster ein:

```
ed devs:Mountlist
```

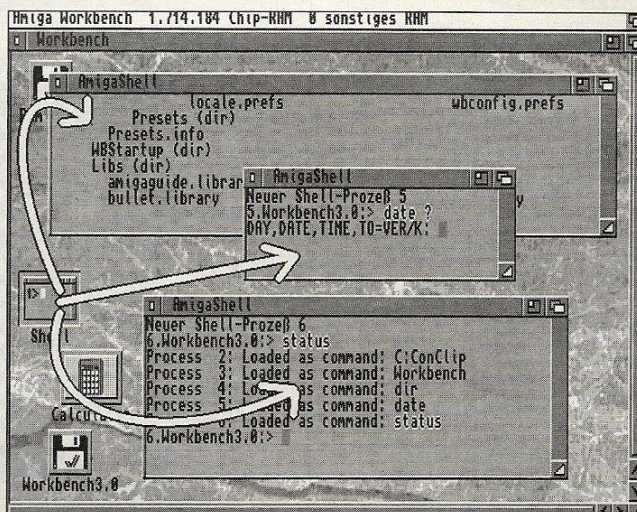
Im Fenster des Editors erscheint eine Textdatei namens »Mountlist«. Vermutlich gegen Ende finden Sie dort einen Eintrag, der etwa so aussieht:

```
RAD: Device = ramdrive.device
Unit = 0
Flags = 0
Surfaces = 2
BlocksPerTrack = 11
Reserved = 2
Interleave = 0
LowCyl = 0 ; HighCyl = 79
Buffers = 5
BufMemType = 1
#
```

Die meisten der hier gemachten Angaben sind nur für das Betriebssystem nützlich. Zu erkennen ist allerdings, daß AmigaDOS auch die resetfeste RAM-Disk wie eine richtige Diskette organisiert: Da ist von Oberflächen (surfaces) die Rede, von Spuren (cylinders) und Blöcken pro Spur (blocks per track).

Für die Größeneinstellung der RAD sollten Sie sich auf »HighCyl« beschränken. Voreingestellt sind 80 Spuren (von 0 bis 79). Wenn Sie die resetfeste RAM-Disk verkleinern wollen, brauchen Sie nur den Wert hinter »HighCyl« zu reduzieren. Eine Spur belegt im Arbeitsspeicher etwa 11 KByte. Ein paar Beispiele möglicher Konfiguration:

HighCyl	Größe der RAD
11	100 KByte
18	200 KByte
46	500 KByte
93	1 MByte



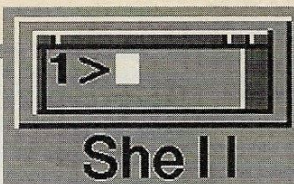
Dreierlei: Unterschiedliche Aktivitäten geschehen gleichzeitig in mehreren Shell-Fenstern – das ist Multitasking

Um die Verwaltung der RAD zu erleichtern, muß dafür ein fester Speicherbereich reserviert werden. Die RAM-Disk dagegen – und das ist einer ihrer Vorteile – verwaltet den Speicher dynamisch, belegt also nicht mehr Platz als für den Inhalt benötigt wird. Weil das bei der RAD nicht vorgesehen ist, wird diese resetfeste Variante auch nicht automatisch vom System bereitgestellt. Sie muß extra eingebunden (Amiga-Slang: geMOUNTet) werden.

Je nach Systemsoftware Version 2.04 oder 2.1 geschieht dies

scheint auf der Workbench ein Diskettenpiktogramm »RAM_0«. Sie können die neue RAM-Disk wahlweise über den Gerätenamen RAD oder den Datenträgernamen RAM_0 ansprechen. Es ist sogar möglich, mit RENAME neue Bezeichnungen für die RAD zu vergeben.

Wenn Sie einen Amiga mit 1 MByte oder noch weniger Speicher besitzen, gibt es allerdings Schwierigkeiten, denn die Standardgröße der RAD beträgt wie bei einer richtigen Diskette 880 KByte, kann also bei weniger als



Wie Sie sehen, ist auch eine Vergrößerung der RAD über HighCyl möglich. Nach der Änderung speichern Sie die Datei Mountlist wieder mit der Funktion »Save« aus dem Menü »Project«. Wer die RAD bisher noch nicht erfolgreich installieren konnte, sollte nun erneut die Anweisung

mount rad:

versuchen.

Etwas anders sieht das Ganze aus, wenn Sie schon mit der Workbench 2.1 arbeiten. Dort haben die Systementwickler nämlich das Konzept für die Einbindung der Gerätetreiber geändert. Die Mountlist gibt es nicht mehr.

Ab OS 2.1 gibt es keine Mountlist mehr

Dafür befinden sich in der Unterschublade »Storage/DOSDrivers« einige Piktogramme, die Sie in die Schublade »Devs« verschieben müssen, um den dazugehörigen Gerätetreiber beim nächsten Systemstart zu aktivieren. Tatsächlich stecken hinter den Geräte-Piktogrammen weitgehend die bisherigen Einträge der Mountlist. Wenn Sie also unter Version 2.1 die Größe der residenten RAD ändern wollen, müssen Sie entsprechend im Shell-Fenster eingeben:

ed devs:dosdrivers/RAD

Findet Amiga-DOS die Datei RAD dabei nicht, liegt sie vermutlich noch in der Schublade Storage. Geben Sie dann ein:

ed sys:storage/dosdrivers/RAD

Die im Editierfenster zu sehenden Zeilen sind zwar anders formatiert, aber es handelt sich um dieselben Informationen wie in der Mountlist. Auch dort sollten Sie für Größenänderungen nur den HighCyl-Wert modifizieren. Speichern Sie auch diese Änderung wie gehabt ab. Falls Sie die RAD zu diesem Zeitpunkt noch nicht eingebunden haben, tun Sie dies durch zweifaches Anklicken des Piktogramms RAD.

Die Probe aufs Exempel: Schieben Sie ein paar beliebige Dateien ins Fenster der RAD. Lösen Sie anschließend mit <Ctrl Amiga_links Amiga_rechts> einen Reset aus. Nach dem Systemstart steht das Laufwerk RAD wie zuvor mit den Dateien, die wir gerade hineingelegt haben, zur Verfügung.

Wenn sich beim Reset keine Startdiskette im Laufwerk 0 befindet und keine System-Partition auf der Festplatte vorhanden ist, versucht das System allerdings von der RAD zu booten. Für Amiga-Besitzer mit viel Speicher, aber ohne Festplatte kann das eine sinnvolle Alternative zum Diskettenstart sein. Das Laufwerk RAD muß dann wie eine Startdiskette eingerichtet werden. Im Normalfall machen Sie davon aber nur selten Gebrauch.

Eine eingebundene RAD werden Sie mit dem DOS-Befehl REMRAD (REMove RAD) wieder los. Er hebt allerdings nur die resetteste Verankerung unseres imaginären Laufwerks auf – erst nach einem weiteren Reset ist die RAD samt Inhalt verschwunden und damit der belegte Speicher wieder frei.

■ Amiga-DOS hat noch mehr dieser nützlichen Gerätetreiber auf Lager. Die PIPE z.B. realisiert eine Art Pipeline innerhalb des Amiga, über die Daten direkt, also ohne Umweg über irgendein Speichermedium, von einer Anwendung zur nächsten fließen.

Für unsere Experimente muß das Gerät PIPE natürlich aktiv sein: Die Anweisung

mount pipe:

erledigt das für Sie. Anwender der Workbench 2.1 klicken zweimal auf das entsprechende Piktogramm in DOSDrivers.

Stellen Sie sich vor: Sie haben mit Ihrem Videodigitizer ein Bild aufgenommen, und möchten dies nun mit einem Malprogramm bearbeiten. Der übliche Weg wäre eine Zwischenspeicherung in der RAM-Disk, um es von dort mit dem Malprogramm wieder zu laden.

Statt RAM geben Sie jetzt sowohl beim Speichern als auch Laden die PIPE als Gerätenamen ein (Beispiel: pipe:DigiBild). Und siehe da: ohne langfristig Platz in der RAM-Disk zu beanspruchen, wandert das Bild direkt über die Pipeline von der ersten zur zweiten Anwendung.

Falls es Ihnen an Speicherplatz oder Anwendungsprogrammen fehlt, können Sie das Ganze auch etwas einfacher testen: Öffnen Sie zwei Shell-Fenster. Geben Sie im ersten Fenster

COPY * TO Pipe:Test

und im zweiten

COPY Pipe:Test TO *

ein.

Aktivieren Sie wieder das erste Fenster und tippen Sie dort ein

paar beliebige Zeilen Text ein. Schließen Sie die Eingabe mit der Tastenkombination [CTRL] ab, dem Amiga-DOS-Kennzeichen für »Dateiende«. Prompt erscheinen alle eingegebenen Zeichen wie von Geisterhand im zweiten Shell-Fenster.

Der »Dateiname« (Test im Beispiel) sorgt übrigens dafür, daß auch mehrere Datenströme gleichzeitig über die Pipeline fließen können, ohne daß dabei etwas durcheinanderkommt.

■ Für besondere Effekte sorgt unser drittes Gerät. Geben Sie

mount speak:

im Shell-Fenster ein. Anschließend können Sie – vorausgesetzt, am Amiga ist ein Tonausgabemedium angeschlossen – die Wirkung dieses Treibers gleich ausprobieren:

copy s:startup-sequence to speak:

Amiga-DOS liest den Inhalt der Datei »Startup-Sequence« vor.

Sollten Sie unter Workbench 2.1 kein Piktogramm »Speak« finden, erzeugen Sie am besten selbst eins. Dazu braucht bloß das Piktogramm Pipe kopiert, in Speak umbenannt, und dann die dahinter steckende Datei mit dem ED editiert zu werden:

ed devs:dosdrivers/Speak

Ändern Sie die Datei so ab:

Handler = L:Speak-Handler
Stacksize = 6000
Priority = 5
GlobVec = -1

Nach dem Speichern können Sie das so entstandene Piktogramm wie jeden anderen »DOS-Driver« der 2.1 durch Doppelklick starten oder beim Systemstart automatisch einbinden lassen.

■ Das nächste Thema hat etwas mit den Skripten zu tun, die wir in der letzten Folge vorgestellt haben. Um sie aufzurufen, haben wir bisher immer den Befehl EXECUTE mit angegeben. Beispiel:

execute ram:meinSkriptresident

Die mitgelieferten Skripte PCD, DPAT und SPAT hingegen lassen sich offensichtlich auch ohne EXECUTE starten. Vielleicht haben Sie das mit Ihren Skripten auch versucht und eine Fehlermeldung à la »File is not executable« bekommen. Das ist jetzt vorbei. Der Trick bei der Sache:

Es gibt eine besondere Kennzeichnung, die eine Datei als Skript ausweist. Findet Amiga-DOS diese Markierung, startet es EXECUTE selbst und übergibt

die angegebene Datei zur Ausführung der DOS Befehle.

Insgesamt verwaltet DOS acht unterschiedliche Merkmale (protection bits), die Sie ein- oder ausschalten: Darf die Datei gelöscht, gelesen, beschrieben, ausgeführt werden? Soll sie versteckt werden (diese Eigenschaft wird auch unter Workbench 2.1 noch immer halberzig unterstützt), wurde sie seit dem letzten Backup verändert, oder handelt es sich um ein Skript? Ist der Programmcode in der Datei wiederholt ausführbar (reentrant)? Letzteres ist wichtig für den Befehl RESIDENT (darauf kommen wir noch zurück).

Mit dem DOS-Befehl PROTECT aktivieren bzw. deaktivieren Sie die Merkmale:

protect test +s

z.B. kennzeichnet eine Datei als Skript (s). Mit einem Minuszeichen vor dem Parameter wie bei

protect test -s

löschen Sie das Merkmal wieder. Nach

protect test -d

kann die Datei bis auf Widerruf mit »protect test + s« nicht mehr

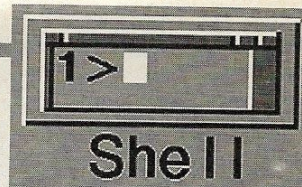
Kursübersicht

Die Dokumentation der Modelle Amiga 500, 600 und 2000 mit OS 2.0 enthält nur eine knappe Beschreibung der Shell, der textorientierten Bedienerschnittstelle des Computers. Viele Systemfunktionen sind aber nur über die Shell erreichbar. Der sichere Umgang damit erleichtert jedem Anwender die Installation neuer Hard- und Software. Dieser dreiteilige Kurs informiert Sie über wichtige Konzepte sowohl der Shell als auch des Betriebssystems.

Folge 1: Über »Execute Command« zur Shell; grundlegende Bedienung der Shell; Programme starten; Übergabeparameter nutzen; Zeichensatz ändern mit SETFONT; die Befehle CD, DIR, LIST und DELETE; Mülleimer löschen; logische Datenränder; der Befehl ASSIGN; Verzeichnishierarchien.

Folge 2: Suchpfade einrichten mit PATH; Skript- bzw. Batch-Programmierung; Modifikation von Startup-Sequence, User-Startup und Shell-Startup; die Skriptdateien PCD, SPAT und DPAT.

Folge 3: Goddies der Shell; die nichtflüchtige RAM-Disk RAD; andere Geräte (PIPE, SPEAK, PAR, PRT); der Befehl MOUNT; Schutzbits; Hard- und Softlinks mit MAKELINK; Tipparbeit sparen mit CONCLIP; die Befehle STATUS, ALIAS und RESIDENT.



mit dem DOS-Befehl DELETE gelöscht werden.

Der DOS-Befehl LIST zeigt Ihnen übrigens, welche Merkmale einer Datei gesetzt sind. Eine Angabe im Shell-Fenster wie

```
PCD      248      -s--rwe...
```

bedeutet etwa, das die Datei PCD eine Skriptdatei ist, und gelesen (readable), überschrieben (writable), ausgeführt (executable) und gelöscht (deletable) werden kann. »-s-rwe« könnte LIST als Kennzeichenfolge einer löschgeschützten Datei ausgeben.

■ Zu den besonderen Goodies von Amiga-DOS gehört ein Verfahren, das bislang noch weitgehend unbekannt ist: Links (Verbindungen). Damit können Sie auf Ihren Speichermedien viel Platz sparen.

Dort befinden sich (hoffentlich) alle Dateien bzw. Programme ordentlich nach Sinn und Zweck strukturiert in verschiedenen Verzeichnissen bzw. Schubladen. Oft benötigte Programme oder Daten möchten Sie aber vielleicht, ohne Abstieg in eine weit verzweigte Verzeichnishierarchie, griffbereit haben. Kein Problem: Mit »Leave Out« aus dem Workbench-Menü »Icons« auf die Workbench legen und dort anklicken.

Manchmal ist es jedoch besser, wenn sich alle Programme, die für eine Anwendung gebraucht werden, in einem Verzeichnis befinden. So gehören z.B. in die Schublade DTP das Layoutprogramm, ein Malprogramm und vielleicht noch ein Zeichenprogramm. Anstatt eine Datei nun komplett zu kopieren und damit ein zweites Mal auf einem Datenträger anzulegen, wird

mit MAKELINK lediglich ein Verweis darauf erzeugt. Dieser Verweis belegt gerade 512 Byte – von außen sieht er jedoch aus wie eine Kopie der Ursprungsdatei. Greift ein Programm oder DOS selbst auf so einen Dummy zu, bekommt es klammheimlich die Originaldatei. Aus Sicht des Betriebssystems weisen beide Verzeichniseinträge auf dieselbe Datei des Massenspeichers.

Das probieren wir gleich aus. Am besten nehmen Sie dafür eine leere Arbeitsdiskette oder legen ein neues Verzeichnis auf Festplatte an. Kopieren Sie dann eine beliebige Datei dort hinein – z. B. den DOS-Befehl INFO aus dem Systemverzeichnis c.

```
copy c:\info ""
```

Die Angabe "" als Ziel der Kopieraktion bewirkt übrigens, daß COPY die Kopie unter ihren ursprünglichen Namen im aktuellen Verzeichnis anlegt.

Mit dem gerade kopierten INFO prüfen wir, wieviel Platz noch auf dem für unser Experiment verwendeten Datenträger ist. Rufen Sie den Befehl auf und notieren Sie sich die Anzahl belegter Blocks (Spalte »Used«). Dann geben Sie ein:

```
makelink DummyInfo Info
```

Das war schon alles – MAKELINK hat unter dem Namen »DummyInfo« einen Eintrag für die Datei »Info« erzeugt. Mit den Befehlen LIST und DIR können Sie wie gewohnt nachschauen, was dabei herausgekommen ist. Dabei wird Ihnen in beiden Fällen eine Besonderheit auffallen: Beim DIR-Befehl erscheint in der Inhaltsliste hinter der Datei Dummy-

Info das Kürzel [hl]. Es steht für Hardlink und weist darauf hin, daß es sich bei dieser Datei um einen von MAKELINK erzeugten Dummy handelt. Nach LIST dürfte Ihnen auffallen, daß Dummy Info genauso groß ist wie das Original. Auf den ersten Blick sieht es so aus, als wäre DummyInfo eine Kopie von Info. In Wirklichkeit zeigt dieses Ergebnis aber, daß der Dummy-Eintrag bei jeder Gelegenheit so erscheint, als wäre er das Original.

Rufen Sie DummyInfo (und damit INFO) auf. Die Angabe benutzt Blocks in der Spalte Used für das entsprechende Laufwerk hat sich nur um 1 erhöht. Auf unserem Datenträger kann sich unmöglich eine identische Kopie von INFO befinden – der Trick hat funktioniert.

Nun kann die Dummy-Datei an eine beliebige Stelle desselben Datenträgers bzw. derselben Partition umziehen, und damit haben wir die gewünschte zweite Ausgabe des Programms.

Den MAKELINK-Befehl können Sie problemlos auch für Dateien mit der Endung »info«, und damit für die Piktogrammdateien verwenden. Beachten Sie aber, daß Sie neben der Dummy-Ausgabe der Piktogrammdatei immer ein Dummy der dazugehörigen Datei anlegen müssen. Beispiel:

```
cd dh1:grafik/DeluxePaintIII
makelink dh1:dtg/DPaint DPaint
makelink dh1:dtg/DPaint.info DPaint .info
```

Im übrigen können Sie auch Dummy-Einträge für ganze Verzeichnisse anlegen. In diesem Fall ist am Ende der MAKELINK-Anweisung allerdings der Zusatz FORCE anzugeben:

```
makelink dh1:dtg/NochEinDPaint dh1:
grafik/DeluxePaintIII force
```

Wenn Sie sich danach ins Verzeichnis NochEinDPaint begeben, landen Sie automatisch im Verzeichnis DeluxePaintIII.

Benötigen Sie einen Dummy-Eintrag nicht mehr, entfernen Sie ihn wie gewohnt mit DELETE. Amiga-DOS löscht selbstverständlich nur den Dummy und nicht etwa das Original. Würde hingegen die Originaldatei selbst gelöscht, wäre die Dummy-Datei ja sozusagen verwaist. Aus diesem Grund verschwindet die Datei nicht wirklich, sondern das zuletzt angelegte Dummy wird selbst zum Original. Mit anderen Worten: So lange es noch einen Verweis auf eine Datei gibt, kann diese nicht endgültig gelöscht

werden, wird der entsprechende Speicherplatz nicht freigegeben.

Eine weitere Anwendung von MAKELINK dürfte vor allem für die DTP-Anwender am Amiga interessant sein: Sowohl Professional Page als auch das System verwenden die Agfa Intellifonts. ProPage erwartet die Dateien mit der Umrissbeschreibung auf dem logischen Datenträger CGFonts mit der Endung »lib« und im Systemverzeichnis »fonts_bullet_outlines« hängt ein »type« dran. Ansonsten enthalten die Dateien mit der jeweiligen Schrift die gleichen Daten.

Am besten kopieren Sie das komplette Systemverzeichnis fonts auf die Partition mit dem DTP-Programm. In die User-Startup gehört dann der Befehl

```
assign fonts: Work:fonts
```

(Ersetzen Sie »Work« durch den Namen Ihrer Partition.)

Dann legen Sie mit MAKELINK in das Unterverzeichnis »bullet_outlines« einen Verweis auf jede ProPage-Schrift an, die Sie vom System und damit über Anwendungsprogramme wie Deluxe Paint nutzen möchten:

```
cd platte:fonts/bullet_outlines
makelink Caslon540.type cgfonts:caslon540.lib
```

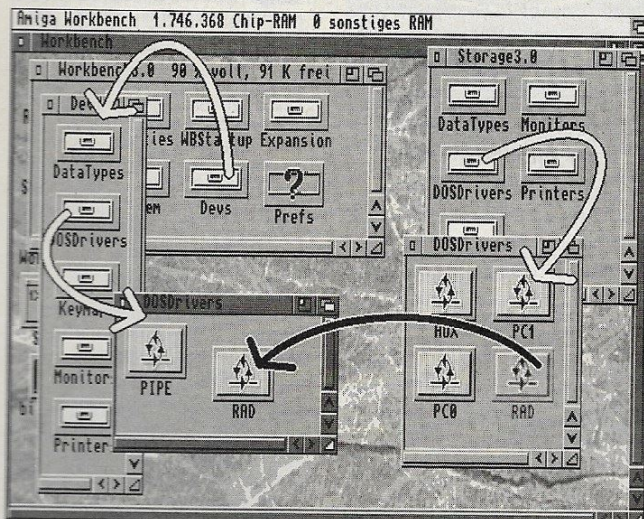
Danach brauchen Sie nur noch Fountain aufrufen, um die teilweise Installation der neuen »Systemschriften« zu beenden.

Zweimal denselben Font? Das ist zuviel

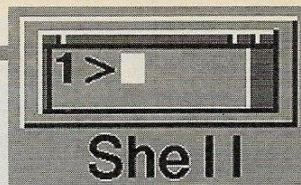
Was ist eigentlich ein »Hardlink«? Daneben soll es ja auch noch einen »Softlink« geben. Da wir in keiner Dokumentation eine plausible Erklärung des praktischen Unterschieds fanden, und MAKELINK z.Zt. sowieso nur Hardlinks erzeugen kann, können wir diese Frage leider nur unbefriedigend beantworten: »Weiche (symbolische) Verbindungen sind Verweise über mehrere Datenträger bzw. Partitionen.«

Aus dieser prägnanten Erklärung der Systemdokumentation leiten wir eins ab: Hardlinks verbinden nur Dateien, die sich auf demselben Datenträger bzw. auf derselben Festplatten-Partition befinden.

■ Der Amiga ist ein Multitasking-Computer. Das bedeutet, daß mehrere Tasks (bzw. Prozesse), also mehrere Programme



DOS-Treiber: Piktogramme in »DosDrivers« ablegen – beim nächsten Systemstart werden sie automatisch eingebunden



quasi gleichzeitig ablaufen können – und das gilt natürlich auch für die Software, die ein Shell-Fenster betreibt. Schon vorhin bei unserem Experiment mit der Pipe haben wir mit zwei Fenstern gearbeitet. Wissenswert ist allerdings, daß Amiga-DOS auch einen Befehl besitzt, der Sie über die Aktivitäten aller Shell-Prozesse informiert. Öffnen Sie doch mal drei Shell-Fenster. Geben Sie danach im ersten Fenster ein:

```
dir sys: opt a
```

Und im zweiten:

```
date ?
```

Und schließlich im dritten der Fenster:

```
status
```

Je nachdem, welche Nummern die von Ihnen geöffneten Shell-Fenster haben, und welche Programme beim Ablauf der Startup-Sequence im Hintergrund aufgerufen wurden, sollte die Ausgabe von STATUS etwa so aussehen:

```
Process 1: Loaded as command: DIR
Process 2: Loaded as command: ConCl ip
Process 3: Loaded as command: Workb
ench
Process 4: Loaded as command: DATE
Process 5: Loaded as command: STATU S
```

STATUS zeigt, mit welchen Befehlen bzw. Programmen sich die einzelnen Shell-Prozesse gerade beschäftigen. Die Nummern hinter Process entsprechen denen der Shell-Fenster. Konsequenter meldet STATUS für das aktive Shell-Fenster sich selbst.

Übrigens: Wenn Ihnen die Ausführung eines Programms in einem anderen Shell-Fenster zu lange dauert oder der Prozeß sich irgendwie festgebissen hat, können Sie ihm aus einem anderen Shell-Fenster ein Unterbrechungs-Signal (Break-Code) senden. Um z.B. von Shell-Prozeß 5 den DIR-Befehl von Prozeß 1 abzubrechen, geben Sie dort ein:

```
BREAK 1
```

Diese Aktion hat denselben Effekt wie die Tastenkombination <Ctrl C> bei aktiviertem Shell-Fenster 1. Viele Programme brechen daraufhin ab. Wenn eins nicht auf <Ctrl C> reagiert, ist auch der DOS-Befehl BREAK machtlös.

Auffällig ist in der STATUS-Liste noch etwas anderes: Da laufen offensichtlich einige Shell-Prozesse, die Sie selbst gar nicht gestartet haben, und von denen auch keine Shell-Fenster zu sehen sind. Das sind Programme, die beim Ablauf der Startup-Sequence gestartet wurden, und

nun im Hintergrund arbeiten. WORKBENCH ist das Programm, das hinter der Workbench-Oberfläche steckt und sie damit realisiert bzw. betreibt.

CONCLIP ist dafür zuständig, daß Sie in Shell-Fenstern beliebige Textpassagen mit der Maus kennzeichnen, mit <Amiga_links c> in einen Pufferspeicher übertragen und mit <Amiga_links v> an der aktuellen Cursor-Position einfügen können – eine ideale Hilfe, um komplizierte und lange Dateinamen aus Verzeichnislisten zu übernehmen, und sich damit die entsprechende Tipparbeit bei der Eingabe zu sparen. Komfort wird ja spätestens seit der Systemsoftware-Version 2.x auch für den Umgang mit der Shell großgeschrieben.

■ Am Schluß wollen wir uns noch die DOS-Befehle ALIAS und RESIDENT anschauen. Damit wären wir bei unserem kleinen Rundgang durch die Möglichkeiten der Shell schon fast wieder

Tipparbeit verbunden. Die Shell setzt für Abkürzungen, die Sie ALIAS definiert haben, automatisch die lange Fassung ein.

Sie können bei ALIAS-Definitionen auch Platz für eventuelle Parameter lassen. Nach Ausführung von

```
alias makeskript protect [] +s
```

z.B. brauchen Sie z.B. nur

```
makeskript Dateiname
```

eingeben, um die angegebene Datei als Skript zu kennzeichnen.

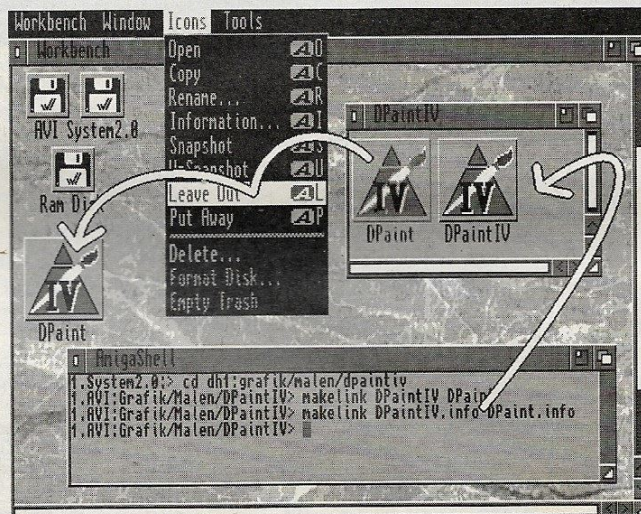
Insbesondere Diskettenanwendungen können ein Lied davon singen: Für häufig benutzte Befehle muß Amiga-DOS immer wieder auf die Systemdiskette zugreifen. Das kostet dann Zeit und Nerven wegen der erforderlichen Diskettenwechsel. Aus diesem Grund sind seit Workbench 2.04 eine Reihe wichtiger Amiga-DOS-Befehle fest in die Systemsoftware eingebaut (resident) und somit jederzeit ohne Diskettenzugriff ab-

dings Speicherverschwendung, denn für die Ausführung muß das Programm aus dem Speicher (der RAM-Disk) geladen und als ausführbares wieder im Speicher abgelegt werden. Es existiert damit praktisch zweimal im Speicher und belegt entsprechend Platz.

RESIDENT verankert ein Programm ausführbereit im Speicher und erspart so das Laden vor sowie Entfernen nach der Ausführung. Dafür muß allerdings eins gewährleistet sein: Wenn das Programm irgendwelche Daten manipuliert, muß bei jedem Aufruf die gleiche Ausgangslage vorliegen. Man kann nämlich Daten so in der Programmdatei unterzubringen, daß diese schon nach dem Laden ohne Ausführung einer einzigen Programm-anweisung bestimmte Werte haben. Wenn die dann im Verlauf der ersten Ausführung geändert werden, ist bei der zweiten eine andere Ausgangslage da.

Programme, die diese Voraussetzung erfüllen, nennt man »re-entrant« (wiedereintrittsfähig). Ein besonderes Protection-Bit, das sogenannte Pure-Bit, gibt Auskunft darüber, ob der Programmcode eines Befehls oder Programms diese Eigenschaft aufweist. Ist das Pure-Bit (zu erkennen am p in der Verzeichnisliste) nicht gesetzt, weigert sich RESIDENT, das angegebene Programm resident zu machen. Allerdings können Sie den DOS-Befehl mit sanfter Gewalt dazu zwingen: Mit der Option FORCE verankert RESIDENT auch Programme, die nicht »pure« sind. Allerdings gehen Sie damit das Risiko ein, daß es bei mehrfachem Aufruf dieses Programms zu Schwierigkeiten bis hin zu einem Systemabsturz kommen kann. Verwenden Sie beim Befehl RESIDENT also nur Programmdateien mit gesetztem Pure-Bit. Machen Sie von dieser Regel nur dann eine Ausnahme, wenn Sie wirklich sicher sind, daß daraus keine Probleme erwachsen.

Hiermit sind wir endgültig in die Tiefen fortgeschrittener Funktionen und Hintergründe der Shell bzw. des Betriebssystems vorgedrungen. Nach drei Folgen »Shell für Einsteiger« können Sie jetzt wichtige organisatorische Arbeiten in der Shell erledigen, viele Aspekte Ihres Amiga besser und flexibler beeinflussen, und sich dabei Ihren Aufenthalt dort so komfortabel wie möglich gestalten. Viel Spaß dabei. pa



Der andere Name ist wichtig, denn »Leave Out« entfernt nur das Piktogramm, nicht aber die Dateien im Verzeichnis

am Ausgangspunkt: dem Umgang mit Befehlen und Shell-Fenstern.

ALIAS haben wir bereits in der zweiten Folge kennengelernt. Der Befehl tritt häufig in der »Shell-Startup« auf. Unabhängig davon können Sie sich auch direkt im Shell-Fenster ein paar Abkürzungen zurechtlegen. Wenn Sie beispielsweise nicht immer »ED S:Startup-Sequence« eintippen wollen, dann legen Sie am besten folgende Abkürzung fest:

```
alias edstart ed s:Startup-Sequence
```

Zukünftige Aufrufe zur Änderung dieser Datei mit dem unhandlichen Namen sind dann wesentlich schneller und mit weniger

rufbar. Um welche Befehle es sich dabei handelt, können Sie leicht herausfinden: Geben Sie in einem Shell-Fenster RESIDENT ein. Alle Befehle, die in der nun erscheinenden Liste mit INTERNAL gekennzeichnet sind, sind von vornherein resident.

Wenn Sie zusätzlich bestimmte Befehle häufiger benötigen, sollte sie diese nachträglich resident machen. Auch das macht RESIDENT. Beispiel:

```
resident c:dir add
```

Die bisherige Praxis (vor OS 2.0) war, solche Befehle in der RAM-Disk unterzubringen und von dort aufzurufen. Das ist aller-