

# Amiga-FAQ

---

Häufig gestellte Fragen und Antworten

21 March 1994

## Jochen Wiedmann

Copyright © Jochen Wiedmann

Am Eisteich 9

72555 Metzingen (Deutschland)

Tel. 07123 / 14881

Internet: [wiedmann@mailserv.zdv.uni-tuebingen.de](mailto:wiedmann@mailserv.zdv.uni-tuebingen.de)

Dieses Dokument enthält einige häufig gestellte Fragen und versucht, Antworten zu geben. Seine Absicht ist es, neuen Benutzern zu helfen und die Zahl der News-Artikel zu reduzieren, die erfahrene Benutzer nicht mehr sehen wollen.

Es ist erlaubt, sowohl veränderte als auch unveränderte Kopien dieses Dokumentes herzustellen und zu verteilen, vorausgesetzt, daß dabei die Bestimmungen der "GNU General Public License" eingehalten werden und die Copyright-Notiz sowie diese Erlaubnis unverändert auf allen Kopien enthalten sind.

Der Autor gibt **absolut keine** Garantie, daß die hier gegebenen Antworten korrekt sind. Einige dieser Antworten sind von anderen Benutzern beigetragen worden und ich habe teilweise nicht einmal die Möglichkeit, auch nur die einfachsten Tests durchzuführen. Vorschläge, weitere Beiträge, neue Fragen und Antworten, Kritik und Beschimpfungen (oh, wie ich mein 'nil:' liebe :-)) sind aber sehr willkommen.

Bitte beachten Sie, daß viele Abschnitte bis jetzt noch fast völlig leer sind. Ich verstehe von den betreffenden Abschnitten nichts und habe nicht das Gefühl, als ob ich daran etwas ändern könnte. Damit sind Sie an der Reihe: Füllen Sie die Lücken und teilen Sie mir mit, was man hier noch aufnehmen kann!

---

# 1 Hardware

Dieses Kapitel enthält Fragen zur Amiga-Hardware.

## 1.1 Was sind 68EC020, 68EC030 und 68LC040?

Motorola, die Herstellerfirma der 680x0-Familie bietet auch gestutzte Versionen ihrer Prozessoren an. Diese sind etwas preisgünstiger als die Originale, weshalb Commodore auch den 68EC020 in den A1200 und den 68EC030 in den A4000/030 einbaut. Dafür können sie aber auch etwas weniger.

Der Unterschied zwischen 68020 und 68EC020 ist, daß letzterer nur 16MB Speicher adressieren kann. (Das ist der Grund, warum der A1200 nur maximal 10MB RAM haben kann.) In den meisten Fällen sollte man den Unterschied aber gar nicht bemerken.

Das ist beim 68EC030 anders: Viele Benutzer werden feststellen, daß der 68030 eine MMU (Memory management unit) hat und der 68EC030 nicht. (Beim 68020 gibt es noch die Möglichkeit, eine externe MMU, den 68851 nachzurüsten.) Es gibt einige wichtige Programme, die eine MMU brauchen, z.B. Enforcer (ein Hilfsprogramm zur Fehlersuche), GigaMem (ein Programm zur Emulation von virtuellem RAM) oder alle aktuellen Unix-Versionen. (siehe Abschnitt 6.1 [Unix], Seite 20) Wer eines dieser Programme verwenden will und keinen A4000 oder A3000 hat, braucht deshalb eine Prozessorkarte mit einem 68030 oder 68040.

Der 68LC040 ist ein 68040 ohne FPU. Siehe Abschnitt 1.2 [FPU], Seite 1.

## 1.2 Was ist ein mathematischer Coprozessor (FPU) ?

Die Prozessoren 68000 bis 68030 bieten ausschließlich Integer (= Ganzzahl) Arithmetik. Floating-Point (Fließkomma) Operationen werden über eine Befehlssequenz ausgeführt. Floating-Point Units (FPUs) bzw. mathematische Coprozessoren sind für diese Aufgaben optimiert.

Im wesentlichen kann man drei FPU-Typen unterscheiden: Den 68881, 68882 und die interne FPU des 68040. Aufgrund der Trennung von Arithmetikeinheit und Konvertierlogik (notwendig zur Konversion anliegender Zahlenformate in das prozessor eigene 80bit-Format) ist die 68882 FPU bis zu 1.5 mal schneller als die 68881. Die interne FPU des 68040 besitzt darüber hinaus eine dreistufige Pipeline, allerdings sind in ihr nicht alle Befehle der 68881/68882 realisiert. Die fehlenden (trigonometrischen) Befehle werden per Software emuliert (z.B. über die 68040.library).

Für viele Programme (Raytracing, DTP, Mathematik, TeX) existieren spezielle Coprozessorversionen. Je nach Auslastungsgrad kann damit eine Geschwindigkeitssteigerung bis zum Faktor 50 erreicht werden.

Michael Kaiser (kaiser@ira.uka.de)

## 2 Das Betriebssystem

Dieses Kapitel behandelt Fragen zum Betriebssystem, sowohl Kickstart als auch Workbench.

### 2.1 Kann ich eine andere als die eingebaute Kickstart benutzen?

Zunächst ein paar Worte zur Legalität der Benutzung einer anderen Kickstart: Es ist **nicht** erlaubt, Kopien von Kickstarts fremder Computer zu erstellen und auf einem anderen Computer zu benutzen! (Es ist sogar fraglich, ob man das auf eigenen Maschinen darf, wenn man mehr als einen Amiga hat.)

Natürlich ist es möglich und für gewisse Personen (z.B. Developer, siehe Abschnitt 3.4 [Developer], Seite 5) auch erlaubt. Es gibt zwei verschiedene Möglichkeiten, eine Hardware- und eine Softwarelösung. Erstere ist, eine Platine in den Computer einzubauen, die sowohl die eigene als auch eine andere Kickstart aufnehmen kann, zwischen denen man dann umschalten kann.

Die Softwarelösung benötigt ein Programm (einen sogenannten Softkicker) und die Kopie der anderen Kickstart. Der Softkicker alloziert RAM, lädt die Kickstart-Kopie in dieses RAM und führt einen Reset aus. Natürlich fehlt dann nach dem Reboot unter der neuen Kickstart etwas RAM: 256KB weniger für Kickstart 1.2 oder 1.3 und 512KB für die neueren Kickstarts.

Es gibt verschiedene Softkicker, von denen einige eine MMU benötigen (siehe Abschnitt 1.1 [MMU], Seite 1). Eine nahezu perfekte Lösung für die Kickstart 1.3 bietet das Programm kick13 (Aminet, Directory 'util/misc'), weil hier bereits die Kickstart integriert ist. (Mit Erlaubnis von Commodore!) Für neuere Kickstarts würde ich SKick 3.43 (Aminet, Directory 'os20/util') empfehlen, das keine MMU braucht und viele verschiedene Kickstarts unterstützt. Die Kopie des ROM zu erzeugen ist mit einem Programm wie dem Folgendem möglich:

```
#include <stdio.h>

#define kickorig 0xf80000 /* 0xfc0000 für Kick 1.2 und 1.3 */
#define kicklen 0x080000 /* 0x040000 für Kick 1.2 und 1.3 */

void main(int argc, char*argv[])
{ FILE *fh;

  if ((fh = fopen("kickstart.file", "w")) != NULL)
    { result = fwrite(kickorig, kicklen, 1, fh);
    }
  fclose(fh);
}
```

## 3 Programmierung

In diesem Kapitel finden vermutlich nur Programmierer Interessantes.

### 3.1 Was ist die beste Dokumentation für Programmierer?

Die beste verfügbare Dokumentation sind sicherlich die RKM's (ROM Kernel Manuals, die schwarzen) von Commodore. Sie werden von Addison-Wesley veröffentlicht.

The Amiga ROM Kernel Manual: Libraries, ISBN 0-201-56774-1  
 The Amiga ROM Kernel Manual: Devices, ISBN 0-201-56775-X  
 The Amiga ROM Kernel Manual: Includes and Autodocs, ISBN  
 0-201-56773-3  
 The Amiga Hardware Manual, ISBN 0-201-56776-8  
 The Amiga User Interface Style Guide, ISBN 0-201-57757-7

Vor allem die Libraries sind ein Muß. Weniger nützlich sind die Includes und Autodocs: Sie sind auf Diskette als Online-Hilfe sicher nützlicher. Siehe Abschnitt 3.3 [Include-Dateien], Seite 4.

AmigaDOS wird in diesen Büchern kaum behandelt. Die Autodocs geben einige Informationen, aber um tiefer einzusteigen braucht man das

The AmigaDOS Manual, 3rd Edition, ISBN 0-553-35403-5

ebenfalls von Commodore, das von Bantam Books herausgegeben wird.

Eine gute Wahl ist auch

The Amiga Guru Book

von Ralph Babel. Das Buch beginnt mit einem ca. 250-seitigen allgemeinen Überblick über verschiedenste Aspekte der Programmierung des Amiga. Für Anfänger dürften vor allem die Abschnitte über die Amiga-Datentypen sowie die Amiga-Includes und die amiga.lib interessant sein, aber auch Erfahrene finden hier mit Sicherheit noch Neues, was zum Teil nicht einmal in den RKM's enthalten ist. Den größten Teil des Buches nehmen aber 500 Seiten nur zu AmigaDOS ein. Dieser Teil ist meines Erachtens der wichtigste, weil das AmigaDOS-Manual der schlechteste Teil der offiziellen Dokumentation ist. Das Buch ist sehr dicht geschrieben und deshalb weniger leicht lesbar als die RKM's, aber ich empfehle es als Zusatz und vor allem anstelle des AmigaDOS-Manuals. (Allerdings ist das Guru-Buch kein Ersatz für die Libraries & Devices, die hier nicht behandelt werden.) Unglücklicherweise hat das Buch keine ISBN-Nummer und ist deshalb nur bei den folgenden Adressen erhältlich:

Buchhaus Gonski, Neumarkt 18a, 50667 Köln  
 Telefon: 0221/2090972, Fax: 0221/2090959

Buchhandlung Bouvier, Am Hof 32, 53113 Bonn  
 Telefon: 0228/7290169 Fax: 0228/7290178

Hirsch & Wolf OHG, Mittelstraße 33, 56564 Neuwied  
 Telefon: 02631/83990 Fax: 02631/839931

DTM-Computersysteme, Dreierherrenstein 6a, 65207 Wiesbaden  
 Telefon: 06127/4064 Fax: 06127/66276

Unlimited GmbH, Kehrstraße 23, 65207 Wiesbaden  
 Telefon: 06127/66555 Fax: 06127/66636

Periscope, Attn: Cody Lee, 1717 W Kirby Ave, Champaign, IL 61821, USA  
 Phone: +1 (217) 398 4237 Fax: +1 (217) 398 4238

Someware, 27 rue Gabriel Peri, 59186 Anor, France  
 Phone: +33 27596000 Fax: +33 27595206  
 E-Mail: didierj@swad.adsp.sub.org

### 3.2 Was ist CATS?

Dies ist eine Abteilung von Commodore West Chester, die früher Commodore Amiga Technical Support hieß und später in Commodore Application and Technical Support umbenannt wurde. Die Mitglieder arbeiten unabhängig von der Entwicklungsabteilung, aber eng mit ihr zusammen, und versuchen, Entwicklern außerhalb von Commodore beim Erstellen guter Amiga-Anwendungen zu helfen, sei das eine Hard- oder Software. Dazu hat CATS eine Menge an Informationen und Tools gesammelt, auf Floppy, CD oder Papier. Ein großer Teil dieses Materials ist auch der Allgemeinheit, d. h. für Nicht-Developer zugänglich. Aber bitte CATS nicht mit einer Hotline verwechseln!

Amerikaner können das Material von

CATS - Developer Applications  
 Commodore  
 1200 Wilson Drive  
 West Chester, PA. 19380

bekommen, der Distributor für ganz Europa ist die

Fa. Hirsch & Wolf  
 Mittelstr. 33  
 56564 Neuwied  
 Tel. 02631/83990

Dr. Peter Kittel, peterk@cbmger.de.so.commodore.com

### 3.3 Wo bekomme ich die Amiga-Include-Dateien?

Der einzige legale Weg, die Include-Dateien und die AutoDocs zu bekommen (und Sie sollten sie bekommen, sie sind *sehr* nützlich!), ist, Developer zu werden (siehe siehe Abschnitt 3.4 [Developer], Seite 5) oder das sogenannte NDU (Native developers update kit, auch als NDUK, NDK bekannt oder ADU für Amiga Developer Update) bei CATS zu kaufen. Siehe Abschnitt 3.2 [CATS], Seite 4. Es kostet etwa 50DM, was sicher ein fairer Preis ist. Dabei handelt es sich um ein 5-Disketten-Set, das neben den aktuellen Includes und AutoDocs auch Tools für Programmierer, z.B. Enforcer, Mungwall, Sushi und die debug.libg (zum Debuggen) sowie anderes, z.B. CatComp (siehe Abschnitt 3.8 [Lokalisierung], Seite 9) oder Report (für Bug-Reports oder Vorschläge an Commodore) enthält. Die aktuelle Version ist 3.1 und für Programmierer beinahe obligatorisch.

Wem die Includes genügen, der kann diese über ein Update seines Compilers (kommerzielle Compiler oder die registrierte Dice-Version) oder von den Fish-CD's bekommen. Siehe Abschnitt 8.5.2 [Fish-CD], Seite 28.

### 3.4 Wie werde ich Developer?

Um Developer zu werden, braucht man die ADSP-Antragsformulare (Amiga developer support program). Um diese zu bekommen, sollte man einen Brief an die lokale Commodore-Niederlassung schreiben und nach diesen Papieren fragen, in denen das weitere erklärt wird. In Deutschland ist die Adresse folgende:

Commodore  
Lyoner Straße 38  
60528 Frankfurt

Es gibt drei verschiedene Arten von Developern:

#### **Registered**

Hier bekommt man vor allem Zugang zum adsp-Kommunikationsnetz (eine Art Commodore-internes Usenet), über das man direkt mit anderen Entwicklern auch von Commodore selber über seine Probleme diskutieren kann. Die Jahresgebühr beträgt 150 DM, die einmalige Aufnahmegebühr 50 DM.

**Certified** Dies ist die interessanteste Klasse: Man bekommt von der meisten Systemsoftware die aktuellen Beta-Versionen (z. B. Kickstart und Workbench) sowie die dazugehörigen Includes und AutoDocs. Certifieds bekommen aber nicht jede Beta und in der Regel auch keine Hardware-Beta. Man bezahlt 400 DM pro Jahr dafür und eine einmalige Aufnahmegebühr von 100 DM.

#### **Commercial**

Kommerzielle Entwickler bekommen im wesentlichen dieselben Informationen wie die "Zertifizierten", allerdings kompletter, sprich auch Beta-Hardware, und etwas früher. Dies hat auch seinen Preis: 700DM pro Jahr plus einmalige Aufnahmegebühr von 100DM.

Die Preise und die angebotenen Möglichkeiten können von Land zu Land verschieden sein, auch gibt es meines Wissens nicht in allen Ländern den "Registered". Ein guter Tip ist es, eine Gruppe zu bilden und damit die Kosten zu reduzieren.

Alle Entwickler müssen das sogenannte NDA (Non disclosure agreement) unterzeichnen. Dies besagt, dass sie die erhaltenen Informationen bei Strafe nicht an Unbefugte weitergeben dürfen.

### 3.5 Was für Compiler (und Assembler) gibt es?

Es gibt viele Programmiersprachen auf dem Amiga, kommerzielle Compiler ebenso wie frei kopierbare. Ich möchte nur diejenigen aufzählen, die mir bekannt sind oder die ich aus einem anderem Grund für erwähnenswert halte.

**Assembler** Alle C-Compiler beinhalten einen Assembler. Frei kopierbar und zuverlässig sind A68K und PhxAss. (Aminet, Directory 'dev/asm' oder Fish-Disks 521 bzw. 906)

## C

**C++** Frei kopierbare C-Compiler sind der `gcc` (der sein eigenes Directory 'dev/gcc' auf dem Aminet hat) und die Probeversion (mit der man aber schon eine ganze Menge anfangen kann) von `Dice` (per FTP von `ftp.uni-paderborn.de`, Directory '/news/comp.binaries.amiga/volume91/languages' oder auf Fish disk 491). `Dice` ist allerdings Shareware, aber die Registrierung kostet nur 90DM. Der große Vorteil von `gcc` ist, daß man ihn auf der ganzen Welt und auf nahezu jedem Computer findet. Ein weiterer Vorteil ist, daß er sogar einen C++-Compiler enthält! Aber er ist langsam und benötigt 4MB RAM oder mehr. `Dice` dagegen ist sehr schnell. Beide Compiler machen einen vertrauenswürdigen Eindruck.

Kommerzielle C-Compiler sind `Aztec-C` und `SAS-C`. Ich kann `Aztec-C` leider nicht mehr empfehlen, denn der Compiler scheint nicht mehr weiterentwickelt zu werden. Was die kommerziellen Compiler auszeichnet, sind ihre hervorragenden Source-Level-Debugger, die den anderen fehlen. `Dice` wird vermutlich bald ebenfalls kommerziell werden und dann ähnliche Debugger beinhalten. `SAS-C` enthält sogar einen Crosscompiler, der C++ in C übersetzt und ebenfalls den Source-Level-Debugger unterstützt.

Comeau C++ ist ebenfalls ein Crosscompiler, was an und für sich kein Problem wäre. Aber Comeau C++ hat keinen integrierten C-Compiler, man braucht also zusätzlich `SAS-C`, `Aztec-C` oder `Dice`. Aber er unterstützt die neuesten Standards und läuft wie `gcc` auf vielen verschiedenen Systemen. In Deutschland wird auch Maxxon C++ angeboten, über das ich nichts sagen kann. Beide Compiler sind kommerziell. Comeaus Adresse ist:

Comeau computing  
91-34, 120th Street  
Richmond Hill, NY, 11418-3214  
USA

E-Mail: Greg Comeau, [comeau@bix.com](mailto:comeau@bix.com)

**Forth** JForth soll eine exzellente Forth-Version sein. Unter anderem enthält es objektorientierte Erweiterungen, ein volles Amiga-Interface und einen Anwendungsgenerator. Es ist erhältlich von:

Delta Research  
P.O. Box 151051  
San Rafael, CA 94915-1051

Phone: (415) 453-4320  
E-Mail: Phil Burk, [phil@ntg.com](mailto:phil@ntg.com)  
Mike Haas, [haas@starnine.com](mailto:haas@starnine.com)

**Fortran** (Seufz! Es gibt immer noch Leute, die es brauchen :-<) Frei kopierbar sind BCF (Fish disk 470) und `f2c`, der Fortran in C-Quelltext umwandelt. (Aminet, Directory '/dev/misc'). Ein kommerzieller Compiler ist von ABSoft erhältlich. Allerdings

sind dies alles nur Fortran-77-Compiler, es gibt keine Fortran-90-Compiler auf dem Amiga.

**Lisp** Frei kopierbare Lisp-Interpreter sind XLisp (Fish-Disk 181) und OakLisp (Fish-Disks 519 und 520) und CLISP ('/pub/lisp/clisp/binaries/amiga' at 'ma2s2.mathematik.uni-karlsruhe.de'). Auch Compiler gibt es: Gambit (Fish-Disks 764 und 765) sowie Scheme-to-C (Fish-Disks 556-558). Von Interesse ist vielleicht eine Mail-Liste: Senden Sie dazu eine Mail mit dem Wort 'Subscribe' an [amigalisp@contessa.phone.net](mailto:amigalisp@contessa.phone.net).

**Prolog** '/dev/lang/UNSWProlog.lha' und 'dev/lang/sbp3\_1e' auf dem Aminet sowie 'SBProlog' auf der Fish-Disk 141 und 'SBProlog' auf der Fish-Disk 145 sind frei kopierbare Prolog-Interpreter.

**Modula-2** M2Amiga wird in Europa und Benchmark Modula-2 in den USA angeboten. Beide sollen sehr gut sein und sowohl über gute Source-Level-Debugger als auch eine umfangreiche Bibliothek verfügen. Besonders M2Amiga wird sehr gut unterstützt durch eine deutsche Benutzergruppe (AMOK), die z.B. eine eigene PD-Serie anbieten.

M2Amiga ist bekommt man bei

A+L AG  
Daderiz 61  
2540 Grenchen  
Schweiz

Tel.: +41/65/52 03-11  
Fax: -79

und Benchmark Modula-2 ist erhältlich von:

Armadillo Computing  
5225 Marymount Drive  
Austin, Texas 78723  
USA

Phone/FAX: 512/926-0360.  
EMail: Jim Olinger, [jolinger@bix.com](mailto:jolinger@bix.com)

## **Oberon**

**Oberon-2** A+L vertreibt auch AmigaOberon, der sowohl Oberon als auch Oberon-2 beherrscht. Der Compiler kommt mit einer integrierten Entwicklungsumgebung (incl. freikonfigurierbarem Editor) und umfangreicher Modulbibliothek. Library Linker zum einfachen Erzeugen von AmigaOS shared libraries sowie run-time Source-Level-Debugger sind ebenfalls erhältlich.

Auch für diesen Compiler gibt es Unmengen von Modulen und Bsp.-Sourcen auf AMOK-Disks.

**Pascal** Es gibt einen PD-Compiler namens PCQ (Aminet, Directory 'dev/lang' oder Fish-Disk 511), der allerdings kein voller Pascal-Compiler ist und dem sehr wesentliche Dinge fehlen. P2C konvertiert Pascal in C und ist auf der Fish-Disk 341 zu finden. (Aminet: '/dev/misc/p2c120.lha') Ferner gibt es zwei kommerzielle Compiler namens HiSoft-Pascal (von der gleichnamigen Firma) und KickPascal von Maxxon.

HiSoft und P2c behaupten, kompatibel zu Turbo-Pascal 5.0 zu sein. HiSoft hat außerdem einen guten Source-Level-Debugger.

### 3.6 Warum funktioniert keine Esc-Sequenz?

Viele Drucker kommen mit einem Handbuch, das erklärt, welche ESC-Sequenzen welche Funktion auslösen. Aber wenn man diese Sequenzen dann einmal ausprobiert, passieren oft merkwürdige Sachen, nämlich entweder gar nichts oder etwas ganz anderes, als geplant war. Und das hat auch einen Grund, nämlich die Amiga-Druckertreiber. Diese Treiber sind so gebaut, daß sie nur einen bestimmten Satz ANSI-Esc-Sequenzen verstehen, nicht die (verschiedenen) druckereigenen, von den verschiedenen Druckerherstellern definierten. Der Sinn dabei ist, daß jede Anwendung auf dem Amiga nur diesen einen Standardsatz an Sequenzen verwendet und so nicht zu wissen braucht, welcher Drucker tatsächlich angeschlossen ist. Der Druckertreiber übersetzt dann diese Standardsequenzen in die druckereigenen. Eine Liste der verfügbaren ANSI-Esc-Sequenzen findet sich im aktuellen Workbench-Handbuch (oder in älteren AmigaDOS-Handbüchern). Wenn Du nun eine Steuersequenz an den Drucker schicken willst, die es nicht als ANSI-Sequenz gibt, hast Du zwei Möglichkeiten, dies doch zu erreichen:

1. Umgeh den Druckertreiber (der erfolglos versuchen würde, die Sequenz zu interpretieren oder zu übersetzen) und sende die Ausgabe **nur** für die Länge dieser Sequenz an 'PAR:' (bzw. 'SER:'). Dabei muß man die Druckerausgabekanäle umständlich oft umschalten, und man muß wissen, wo der Drucker angeschlossen ist ('PAR:' oder 'SER:').
2. Benutze eine spezielle ANSI-Sequenz, genau für diesen Fall gedacht:

'Esc [<n>"<x>'

wobei '<n>' die dezimal geschriebene Anzahl an Bytes in der Sequenz '<x>' ist, die gerade die spezielle Drucker-Sequenz enthält. Diese ANSI-Sequenz sagt dem Druckertreiber, daß er die nächsten '<n>' Bytes nicht interpretieren oder übersetzen soll.

Aber beide Methoden haben einen großen Nachteil, wenn sie in einem Anwendungsprogramm verwendet werden: Man verliert die Druckerunabhängigkeit! Solange man sich an die ANSI-Sequenzen hält, kann man jeden Drucker der Welt ansteuern, solange es einen Amiga-Druckertreiber für ihn gibt. Wenn man anfängt, druckereigene Sequenzen zu verwenden, ist das Programm an diesen einen Druckertyp gebunden und ist mit keinem anderen benutzbar (oder man müßte einige Dutzend neue Druckertreiber für dieses Programm erstellen).

Dr. Peter Kittel, peterk@cbmger.de.so.commodore.com

### 3.7 Kann ich AmigaBasic auf dem A1200 verwenden?

Letztens ging eine Kontroverse über AmigaBasic durch die Netze: Ich sagte, auf dem A1200 läuft es einigermaßen problemlos, jemand anders berichtete, daß es bei jedem kleinsten Fehler sofort komplett abstürzt, was ich nicht nachvollziehen konnte.

Jetzt kann ich es: Es liegt am Sound-Prefs-Editor. Wenn man in ihm die Sound-Ausgabe ganz abschaltet, kann man mit AmigaBasic arbeiten. Wenn hier ein Sound (z. B. Piepsen)

angewählt ist, kollidiert das mit dem Sound, den AmigaBasic offensichtlich zu Fuß und nicht ganz korrekt selber erzeugen will, bumm.

Abhilfe also:

1. Im Sound-Editor den Ton abstellen.
2. Auf einem A4000 (oder einem A1200 mit Fast-RAM-Erweiterung<sup>1</sup>) muß man zusätzlich NoFastMem aktivieren.
3. Möglichst SUBs vermeiden und stattdessen herkömmliche GOSUBs benutzen, dann ist die Kompatibilität zu neueren Prozessoren höher.

Dr. Peter Kittel, peterk@cbmger.de.so.commodore.co

### 3.8 Wie lokalisiere ich mein Programm?

Nehmen wir an, wir wollen ein `HelloLocalWorld.c` schreiben. Das letztendliche Programm sieht dann ungefähr so aus:

```
#include "HelloLocalWorld_Cat.h"
#include <exec/clib_protos.h>

struct Library *LocaleBase;

void main(int argc, char *argv[])
{
    /* Öffne die locale.library. (Kein Abbruch, wenn sie nicht
       da ist, weil dann einfach die eingebauten Strings verwendet
       werden. Aus diesem Grund auch keine Verwendung des
       AutoOpening, auch wenn es der Compiler beherrscht.
    */
    LocaleBase = OpenLibrary("locale.library", 38);
    OpenHelloLocalWorldCatalogs(NULL, NULL);

    printf(GetString(MSG_Hello));

    CloseHelloLocalWorldCatalog();
    if (LocaleBase) CloseLibrary(LocaleBase);
}
```

Die Funktion `GetString` prüft, ob die gewünschten Kataloge vorhanden sind und liefert einen Zeiger auf einen String, entweder den eingebauten oder den Katalogstring. (In unserem Fall den deutschen String.)

Der Hauptunterschied zum gewohnten `HelloWorld.c` ist also (abgesehen von der minimalen Initialisierung und dem Gegenstück am Programmende), Strings durch einen Funktionsaufruf zu ersetzen. Man braucht also eine Datei `'HelloLocalWorld_Cat.c'`, die die Funktionen `OpenHelloLocalWorld`, `GetString`, `CloseHelloLocalWorld_Cat.h` und die eingebauten Strings enthält (dies koennte ein Array sein, das unter anderem

---

<sup>1</sup> Nur bei einem zusaetzlichen Prozessor

```
array[MSG_Hello] = "Hello, local world.\n";
```

enthält) und ein Includefile 'HelloLocalWorld\_Cat.h', das die ID's wie MSG\_Hello definiert. Es ist nicht nötig, zu wissen, wie diese Dateien intern arbeiten, insbesondere benötigt man auch keine Kenntnis der `locale.library`!

Dazu gibt es verschiedene Kataloggeneratoren (im Folgenden KG), nämlich 'CatComp' (nur fuer Developer), KitCat, (nur deutsche Dokumentation, was hierzulande kein Problem ist), 'MakeCat', (das ich nicht kenne) und FlexCat (das ich empfehle, einerseits, weil es sehr flexibel im erzeugten Source ist und z.B. Lokalisierung unter 2.0 sowie beliebige Programmiersprachen unterstützt, selbst Amiga-E, Cluster, Pascal, ... und andererseits, weil es von mir ist ;-), die diese Dateien sowie die Kataloge erzeugen. (Der obige Quelltext könnte je nach KG leicht unterschiedlich aussehen.) Siehe Aminet, directory 'dev/misc'.

Wie funktionieren diese KGs? Zunächst erzeugt man eine sogenannte Katalogbeschreibung (Catalog description), die so aussehen könnte:

```
; Mit einem Semikolon beginnende Zeilen sind Kommentare
# language english
; die Sprache der eingebauten Strings
# version 0
; die Katalogversion (0 = beliebig)
MSG_Hello (1/15/30)
Hello, local world
```

Jeder String wird durch zwei Zeilen wie die letzten beiden definiert: MSG\_Hello ist die String-ID, (1/15/30) gibt den Wert der ID sowie die minimale und maximale Länge an. (Diese Argumente koennen auch weggelassen werden, in welchem Fall einfach die nächste freie ID verwendet wird.)

Nun schreiben wir das Programm. Sobald es fertig ist, wird mit dem KG eine sogenannte Kataloguebersetzung (eine für jede andere Sprache als die eingebaute) erzeugt, die so aussehen könnte:

```
; Mit einem Semikolon beginnende Zeilen sind Kommentare
## language deutsch
; the catalog language
## version $VER: Deutsch.catalog 1.0 (22.12.93)
; the catalog files version string
MSG_Hello

; Hello, local world
```

Beachten Sie die leere Zeile nach der String-ID! (Die Argumente von ## language und ## version wären vielleicht leer.) Hier müssten jetzt die deutschen Strings eingesetzt werden. Mit dem KG wird daraus dann der eigentliche Katalog erzeugt. (Beachten Sie auch, daß hier die Angaben über String-ID und Stringlänge fehlen: Sie werden aus der Katalogbeschreibung übernommen.

Wenn das Programm verändert wird (neue Strings, andere Längen) und die Katalogbeschreibung sich damit ebenfalls ändert, dann kann der KG analog benutzt werden, um auch die Kataloguebersetzung und damit den Katalog auf den neuesten Stand zu bringen.

### 3.9 Wie erhält man einen Zeiger auf das Fenster einer Konsole?

Die folgende Funktion liefert den Window-Zeiger eines CON-Fensters. Sie kann unter allen Versionen des Amiga-OS ausgeführt werden.

```

struct Window *getConWindowPtr(BPTR fh)
{
    struct Window *w;
    struct FileHandle *cfh;
    struct StandardPacket *sp;
    struct InfoData *id;
    struct MsgPort *mp;

    w = NULL;

    if((cfh = BADDR(fh))->fh_Type != NULL)
    {
        if(sp = AllocMem(sizeof(struct StandardPacket),
                        MEMF_PUBLIC | MEMF_CLEAR))
        {
            if(id = AllocMem(sizeof(struct InfoData),
                            MEMF_PUBLIC | MEMF_CLEAR))
            {
                if(mp = CreatePort(NULL, 0))
                {
                    sp->sp_Msg.mn_Node.ln_Name = (char *)&sp->sp_Pkt;
                    sp->sp_Pkt.dp_Link          = &sp->sp_Msg;
                    sp->sp_Pkt.dp_Port          = mp;
                    sp->sp_Pkt.dp_Type          = ACTION_DISK_INFO;
                    sp->sp_Pkt.dp_Arg1          = MKBADDR(id);

                    PutMsg(cfh->fh_Type, &sp->sp_Msg);
                    (void)WaitPort(mp);
                    (void)GetMsg(mp);

                    if(sp->sp_Pkt.dp_Res1)
                        w = (struct Window *)id->id_VolumeNode;

                    DeletePort(mp);
                }
                FreeMem(id, sizeof(struct InfoData));
            }
            FreeMem(sp, sizeof(struct StandardPacket));
        }
    }

    return w;
}

```

Anmerkungen:

Auf ein CON-Fenster direkt zuzugreifen kann Konflikte mit Aktionen des CON-Handlers hervorrufen. Seien Sie vorsichtig!

Um den Window-Zeiger einer CLI-Konsole zu erhalten, übergibt man ein durch `Open("**",`

MODE\_OLDFILE) gewonnenes FileHandle an obige Funktion.

Das Ergebnis der obigen Funktion kann sehr wohl NULL sein, etwa im Falle eines AUX-Handlers oder falls ein AUTO-CON-Handler sein Fenster nicht öffnen kann.

Schickt man ein ACTION\_DISK\_INFO-Paket an einen AUTO-CON-Handler (2.0+), so verliert dessen Fenster seine speziellen AUTO-Eigenschaften (es kann also nicht mehr durch das Betätigen des Close-Gadgets geschlossen werden), weil der in id\_VolumeNode gelieferte Window-Zeiger gültig bleiben muß.

Alles in allem: Verwenden Sie diese Funktion nicht. :-)

Weitere Informationen finden Sie auf den Seiten 273, 276, 435, 463, 485 und 629 im "Amiga Guru Book" (siehe Abschnitt 3.1 [Dokumentation], Seite 3).

Ralph Babel, rbabel@babylon.pfm-mainz.de

### 3.10 Was sind Pragmas?

Pragmas sind spezielle Anweisungen an den Präprozessor eines C-Compilers. Zwei Probleme entstehen bei der Verwendung von Pragmas:

1. Pragmas sind hochgradig compilerspezifisch. Nicht einmal die Amiga-Compiler haben dieselben Pragmas, selbst wenn damit das gleiche bewirkt wird.
2. Man kann sich nicht darauf verlassen, daß ein Compiler Pragmas ignoriert, die er nicht für steht. Dies gilt selbst dann, wenn man einen Ausdruck wie den folgenden verwendet:

```
#ifndef MY_COMPILER
#pragma DoAnything
#endif
```

Das letztere Problem läßt sich umgehen, indem man Pragmas wie folgt in eigene Include-Files setzt. (Das gleiche gilt übrigens auch für andere Präprozessor-Kommandos wie #asm (Aztec-C) oder #extern (C++).)

```
#ifndef MY_COMPUTER
#include <mypragmas.h>
#endif
```

Aber was machen Pragmas auf dem Amiga? Meistens werden sie verwendet, um dem Compiler mitzuteilen, wie gewisse Library-Funktionen aufgerufen werden. (Tatsächlich wird fast immer diese Verwendung gemeint, wenn Amiga-Besitzer über Pragmas sprechen.) Gewöhnliche C-Funktionen erwarten ihre Argumente auf dem Stack, Library-Funktionen dagegen in bestimmten Registern. Ferner erwarten sie den Library-Base-Pointer in Register a6. Betrachten wir eine Pragma-Anweisung von Aztec-C:

```
#pragma amicall(SysBase,0xd2,FreeMem(a1,d0))
```

Dies weist den Compiler an, das erste Argument in Register a1 und das zweite in d0 zu laden. Ferner wird der Inhalt der Variablen SysBase in Register a6 geladen. Maxon-Pragmas sehen genauso aus, Dice- und SAS-Pragmas sind allerdings etwas komplizierter:

```
#pragma libcall SysBase FreeMem d2 0902
```

Hier ist d2 (wie 0xd2 oben) der **Library-Vektor-Offset** (siehe nächstes Beispiel). Die letzte Ziffer ist die Zahl der Argumente, die davorstehende 0 ein Code für das Register mit dem Ergebnis und die davor stehenden Ziffern sind Codes für die Register mit den Argumenten in verkehrter Reihenfolge. (Die Codes bedeuten 0=d0, 1=d1, .., 8=a0, 9=a1, a=a2, ..)

Ein Kommando wie 'FreeMem(fib,sizeof(\*fib));' würde ein Compiler nun in folgenden Code übersetzen:

```
move.l  _fib,a1
move.l  260,d1      ; sizeof(struct FileInfoBlock)
move.l  _SysBase,a6
jsr     -0xd2(a6)   ; 0xd2 = _LVOFreeMem
```

FreeMem in dieser Art aufzurufen ist kürzer und schneller als zunächst die Argumente auf den Stack zu legen und dann eine Funktion \_FreeMem aufzurufen, die letzten Endes doch nur dasselbe tun und die Argumente vom Stack in dieselben Register laden würde.

Das Portierungsproblem der Pragmas umgeht man, indem man sie folgendermaßen in den eigenen Quelltext einbindet:

```
/* Lade den Funktionsprototyp. Dieser ist nicht vom */
/* verwendeten Compiler abhängig. */
#include <clib/exec_protos.h>

/* Pragmas sind vom Compiler abhängig, aber wenigstens */
/* die Namen der Dateien mit Pragmas sind relativ */
/* einheitlich. */
#ifdef AZTEC_C
#include <pragmas/exec_lib.h>
#endif
#if defined(__SASC) || defined(_DCC) || defined(__MAXON__)
#include <pragmas/exec_pragmas.h>
#endif
#ifdef __GNUC__
#include <inline/exec_lib.h>
#endif
```

Das obige Beispiel kann problemlos mit allen angegebenen Compilern verwendet werden und produziert optimalen Code.

Eine abschließende Frage bleibt allerdings: Wie bekommt man die Pragmas? Die meisten Compiler haben bereits fertige Pragmas im Lieferumfang. Allerdings hilft das nicht, wenn man z.B. eine neue Library benutzen möchte oder nur die Pragmas einer veralteten Version hat. In diesem Fall kann man die Pragmas selbst aus den sogenannten FD-Files erzeugen. Dazu haben die meisten Compiler ein Utility namens **fd2pragma**. (Das NDU hat z.B. ein Directory namens FD, in dem die FD-Files aller Libraries und Devices des OS enthalten sind. siehe Abschnitt 3.3 [Include-Dateien], Seite 4) Es gibt auch ein frei kopierbares fd2pragma, das Pragmas für Aztec, Dice, SAS und Maxon sowie LVO-Files für den Aztec-Assembler und eventuelle Tag-Versionen produziert. (Aminet, 'dev/misc/fd2pragma2\_0.1ha' oder auf den Fish-CD's)

### 3.11 Wo finde ich die Funktion xxx?

Zunächst sollte man sich versichern, daß die Funktion tatsächlich fehlt: Z.B. Floating-Point-Funktionen befinden sich in einer speziellen Link-Library, die erst mit der Option '-lm' eingebunden wird. Ferner kann es eine fehlende Variable sein: Wenn man z.B. ohne es zu bemerken eine Intuition-Funktion benutzt, dann wird der Linker über das Fehlen eines Symbols `IntuitionBase` klagen. In diesem Fall muß man also lediglich das Folgende irgendwo im globalen Teil seines Quelltextes einbauen:

```
struct Library *IntuitionBase;
```

(Und vergessen Sie nicht, die Library mit `OpenLibrary()` zu eröffnen und mit `CloseLibrary()` zu schließen! :-)

Allerdings könnte die Funktion natürlich tatsächlich fehlen. Wenn man zum Beispiel nur die Version 2.0 der `amiga.lib` hat, dann fehlen etwa die Locale-Funktionen oder die Memory-Pool-Funktionen obwohl sie prinzipiell verwendbar sind.<sup>2</sup> Die einfachste (und beste) Lösung ist, das sogenannte NDU zu kaufen. Siehe Abschnitt 3.3 [Include-Dateien], Seite 4. Wer nicht solange warten möchte, für den ist die Frage, welche Art von Funktion in seiner Link-Library fehlt:

Einfache Amiga-Library-Funktionen (z.B. 'exec/AllocPooled' oder 'locale/OpenCatalogA')■ kann man mit Pragmas aufrufen und braucht dann lediglich Informationen über die Aufrufkonventionen (Library-Base, Library-Vektor-Offset und Argumentregister)

Tag-Funktionen sind meistens einfach Zwischenfunktionen, die ihre Argumente auf dem Stack erwarten und dann die eigentliche Library-Funktion aufrufen. Wenn man z.B. die Funktion 'dos/AllocDosObject', die eine Konstante und einen Zeiger auf ein Array als Argumente erwartet, nach der obigen Methode konstruiert hat, dann hat man damit auch ihre Stack-Variante! Dazu erzeugt man einfach die folgende Funktion:

```
#include <clib/dos_protos.h>
#include <pragmas/dos_pragmas.h> /* Evtl. anderer Name */

void *AllocDosObjectTags(ULONG objtype, Tag tag1, ...)
{ return(AllocDosObject(objtype, (struct TagItem *) &tag1);
}
```

Mit `fd2pragma` kann das auch automatisch geschehen. Siehe Abschnitt 3.10 [Pragmas], Seite 12.

Einige Funktionen bleiben aber noch übrig: `Amiga.lib` enthält nämlich auch Funktionen, die für sich selbst interessant sind, z.B. die BOOPSI-Funktionen ('DoMethod', 'DoSuperMethod'), die Memory-Pool-Funktionen ('LibAllocPooled', 'LibCreatePool', die Ersatz für entsprechende 3.0-Funktionen sind) oder 'HookEntry', das sehr hilfreich bei der Programmierung von Hooks ist. Diese kann man nur durch entsprechende, disassemblierte und neu assemblierte oder kompilierte Versionen ersetzen. Im AmigaFAQ-Archiv sind einige dieser Funktionen bereits enthalten.

---

<sup>2</sup> Dieses Problem betrifft vor allem Besitzer von Aztec-C, das seither nicht weiter unterstützt wird und von Dice, der manchmal etwas unvollständig ist. Ich besitze beide ..

## 4 Anwendungen

Dieses Kapitel enthält Informationen zu verschiedenen Anwendungsbereichen.

### 4.1 Text-Editoren

Editoren sind Programme, die die Eingabe von unformatiertem Text ermöglichen. Dies ist meist Text, der anschließend durch den Computer verarbeitet wird. Programmierer benutzen etwa Editoren zur Eingabe der Quelltexte. Unter Unix beginnt Textverarbeitung meist mit einem Editor, dessen Ausgabe dann mit einer Textbearbeitungssprache (z.B. Abschnitt 4.4 [LaTeX], Seite 17) in gutaussehende Dokumente umgewandelt wird, da unter Unix Textverarbeitungsprogramme kaum verbreitet sind.

#### Kommerzielle Produkte

CygnusEd Professional und TurboText scheinen hier die Nase vorn zu haben. Auf den Fish-Disks gibt es Dutzende von Shareware-Editoren. Eine Demoversion von TurboText ist auf Fish-Disk 445 und eine allerdings sehr alte von CygnusEd auf Fish-Disk 95. Die Folgenden Editoren sind frei kopierbar.

**Emacs** Gnu Emacs (Aminet, Directory 'util/gnu') kommt von Unix und ist möglicherweise der König der Editoren. Er ist riesig (über 1MB), enthält eine unglaubliche Vielfalt von Funktionen (z.B. ein eigenes Fenstersystem und sogar ein Spiel :-)) und beliebig erweiterbar: Emacs enthält einen eigenen Lisp-Interpreter! Andererseits ist er dadurch schwer erlernbar und so beliebig konfigurierbar, daß man mit dem Emacs eines anderen oft nicht mehr umgehen kann...

**Vi** Einfacher, aber weniger konfigurierbar und nicht erweiterbar ist Vi. Er ist vor allem bei Unix-Benutzern beliebt, weil es ihn auf *jeder* Unix-Maschine gibt. Die Wahl zwischen beiden ist eine Sache der persönlichen Vorliebe, kann einen allerdings fürs Leben kennzeichnen. Vim und Stevie (Fish-Disk 591 bzw. 256 und Aminet, Directory 'util/edit') sind Vi-Versionen für den Amiga.

**DME** Viele Amiga-Programmierer mögen DME. Er ist schnell und voll konfigurierbar: Menüs und Tastenbelegung sind beliebig. Er ist auch leichter erlernbar als Emacs und Vi. Es gibt drei verschiedene Versionen: AmokEd, DME und XDME. Die Wahl ist Geschmacksache. (Oberon- und Modula-Programmierer bevorzugen häufig den AmokEd, weil der in Oberon geschrieben ist und die Fehlermeldungen von AmiOberon direkt anzeigen kann. C-Programmierer dagegen stehen öfter auf letzterem.) Quellen: Aminet, Directory 'util/edit', Fish-Disk 776 (XDME) und 749 (AmokEd), AMOK 90.

### 4.2 Was für Textverarbeitungsprogramme gibt es?

Textverarbeitungsprogramme benötigt man, um Briefe, Artikel oder ähnliches zu schreiben. Dies ist natürlich eine der wichtigsten Anwendungen eines Computers und eine für Sie geeignete

Wahl bestimmt häufig, wie gut Sie mit Ihrem Computer überhaupt zurechtkommen. Textverarbeitungsprogramme besitzen teilweise Fähigkeiten, die sie sogar für DTP (siehe Abschnitt 4.3 [DTP], Seite 16) geeignet machen. Für einfacher strukturierte Texte, etwa Quelltexte benützt man allerdings meist besser einen Editor. Siehe Abschnitt 4.1 [Editoren], Seite 15.

Man unterscheidet Wysiwyg-Programme (What you see is what you get) und Seitenbearbeitungssprachen. Wysiwyg-Programme sollten schnell, komfortabel und einfach erlernbar sein. Die meisten Anwender ziehen sie vor. Die Alternative ist ein funktioniert ähnlich wie Compiler: Man verarbeitet Text-Dateien mit einem Programm, das das Layout übernimmt und eine weitere Datei erzeugt, die dann mit auf dem Bildschirm angezeigt oder auf einem Drucker ausgegeben werden kann. LaTeX ist eines dieser Programme. Siehe Abschnitt 4.4 [TeX], Seite 17. Lout ist ein weiteres solches Programm, schneller und einfacher als LaTeX und mit vollständiger Dokumentation versehen, aber natürlich wesentlich weniger verbreitet. Lout erzeugt PostScript-Dateien. (Ob dies ein Vor- oder Nachteil ist, muß jeder selbst beurteilen. ;- ) Siehe Abschnitt 4.5 [PostScript], Seite 17. Beide Programme sind frei kopierbar.

Wysiwyg-Programme gibt es viele, allerdings alle kommerziell: Final Copy II, Wordworth, Word Perfect, AmiWrite, Beckertext II, Maxon Word und viele andere. Zur Wahl kann ich nur eines sagen: Lassen Sie sich viel Zeit bei der Auswahl.

### 4.3 Desktop Publishing

Diese Stärke dieser Programme ist die beliebige Anordnung und Gestaltung von Text- oder Grafikelementen. Bei Textverarbeitungen sind diese Fähigkeiten meist unterentwickelt. Umgekehrt fehlen DTP-Programmen häufig wichtige Funktionen zur Manipulation des Textes selbst. Allerdings kommen die besten Textverarbeitungen sicher nahe an den DTP Bereich (etwa Microsoft Word auf dem Mac oder PC) und umgekehrt ist Framemaker (UNIX, Mac, DOS) ein Beispiel eines DTP-Programms, das die wichtigsten Textverarbeitungsfunktionen enthält. Auf dem Amiga gibt es leider weder in der einen noch in der anderen Richtung hervorragende Beispiele, wenn auch die wichtigsten Textverarbeitungen schon einiges leisten. Wenn man nicht gerade regelmäßig kleinere Zeitungen herausbringt, dann wird in den meisten Fällen ein Textverarbeitungsprogramm genügen. Und wer umgekehrt komplexere Funktionen wie mathematische Formeln, umfangreiche Indexe und Referenztabellen benötigt, der wird womöglich auf eine Seitenbearbeitungssprache angewiesen sein. Siehe Abschnitt 4.2 [Textverarbeitungen], Seite 15.

Es gibt derzeit nur kommerzielle DTP-Systeme, vor allem ProPage und PageStream, die in den letzten Jahren wohl gegenseitig Hase und Igel miteinander gespielt haben. Derzeit scheint PageStream 3.0 vorne zu liegen. Beide Programme liegen im Preis um 300\$, in den USA gibt es wohl auch Studentenermäßigung (um 40%). Eine ausführlichere Beschreibung wäre willkommen.

### 4.4 Was ist TeX und wo bekomme ich es?

TeX ist eines der mächtigsten Textverarbeitungssysteme, die es gibt. Es kann praktisch beliebige mathematische Formeln ebenso darstellen wie komplexe Tabellen, Funktionen plotten (mit Hilfsprogrammen), Indexe oder Inhaltsverzeichnisse erzeugen und viele andere Dinge. Sein

größter Vorteil ist, daß es frei kopierbar ist (TeX selbst, nicht unbedingt die Drucker- und Bildschirmtreiber) und sich dadurch auf der ganzen Welt und auf praktisch jeder Computerfamilie verbreitet hat. Sein größter Nachteil ist, daß es sehr unhandlich und schwer zu erlernen ist (Es arbeitet ähnlich wie ein Compiler.) und keinerlei Wysiwyg-Fähigkeiten hat. Aber viele Leute mögen es. (Dieser Text ist übrigens mit TeX erstellt worden. 8-) Siehe Abschnitt 4.2 [Textverarbeitungen], Seite 15.

Es gibt im wesentlichen zwei Implementationen auf dem Amiga. Amiga-TeX von Thomas Rockicki und Radical Eye Software ist kommerziell und kostet etwa 200-300\$. Es soll wirklich exzellent sein und seine Besitzer schwören darauf.

Ich persönlich empfehle PasTeX, eine frei kopierbare Version. Es gibt immer wieder Leute, die über Installationsprobleme klagen (besonders, was das Laden und die automatische Erzeugung von Fonts angeht), aber ich kenne niemanden, der unzufrieden ist, nachdem es erst einmal installiert ist. (Ein Freund mit TeX-Kenntnissen kann enorm hilfreich sein.) Wenn man sich an die Vorgaben des Installationsskripts hält, sollte es eigentlich keine Probleme geben. Wesentlich ist allerdings, daß man alles nötige installiert und dazu braucht man:

- 5 Disketten mit dem eigentlichen TeX-Compiler
- 2 Disketten mit MetaFont
- Nichts weiter

Viele Leute fragen nach Fonts. Diese sind Teil des MetaFont-Paketes und werden automatisch erzeugt, wenn sie gebraucht werden, vorausgesetzt natürlich, daß TeX so wie in der Dokumentation beschrieben installiert ist. (Das Erzeugen der Fonts kann am Anfang natürlich dauern bis die wichtigsten erst mal da sind.) Bitte beachten Sie, daß die PasTeX-Disketten mit Zoom gepackt sind. (siehe Abschnitt 7.2 [Endungen], Seite 21) PasTeX bekommt man per FTP z.B. bei `ftp.uni-passau.de` im Directory `‘/pub/amiga/tex/PasTeX1.3’`.

## 4.5 Gibt es PostScript-Interpreter auf dem Amiga?

PostScript ist eine Programmiersprache für das Ausdrucken von ganzen Seiten. Apple hat PostScript populär gemacht, indem sie es in ihre Drucker eingebaut haben. Inzwischen gibt es viele Programme, die ihre Ausdrücke als PostScript-Quelltexte machen können. Bis vor kurzem brauchte man für PostScript einen relativ teuren Laserdrucker. Inzwischen gibt es aber auch Programme, die PostScript-Dateien auf anderen Druckern ausgeben können.

Einer der Vorteile von PostScript ist, daß es unabhängig von der gewählten Auflösung ist. Man kann also die Vorteile der möglichen Auflösung auf dem Drucker ausnutzen und trotzdem dieselbe Datei auf dem Bildschirm mit seiner meist schlechteren Auflösung darstellen.

Es gibt zwei frei kopierbare PostScript-Interpreter, Post und GhostScript. Post ist eine shared-Library mit Programmen zur Ausgabe auf Drucker und Bildschirm. Man kann damit beispielsweise in AmigaTeX PostScript-Dateien als Bilder in TeX-Dateien einbauen. Siehe Abschnitt 4.4 [TeX], Seite 17. GhostScript besteht ebenfalls aus zwei Programmen, GhostScript

selbst (der eigentliche Interpreter) und GhostView, der Benutzeroberfläche. Quellen: Aminet (Directories 'text/print' und 'text/dtp'), Fish Disk 669.

## 5 Grafik

Grafik ist eine der Stärken des Amiga. Warum hat es hier so wenig Fragen und Antworten?

### 5.1 Was heiß Chunky- und Planar-Display?

Einfach gesagt stehen die Bezeichnungen `chunky` und `planar` (Kürzel für `bitplanar`) für verschiedene Arten, graphische Daten im RAM des Computers zu speichern. Sie sind einfach zu verstehen, aber vielleicht etwas schwierig zu erklären.

Die Anzeige eines Computers besteht aus einem Netz von Pixeln. Jedes Pixel kann man sich als eine Zahl denken, die für die Farbnummer des Pixels steht. Hier ist zum Beispiel eine ganz einfache Anzeige mit 4 Farben:

```
00302132
```

Der Amiga speichert dies im `bitplanaren` Modus, d.h. es werden verschiedene sogenannte Bitplanes verwendet, in denen zu jedem Pixel genau ein Bit gehört. Für eine Zahl zwischen 0 und 3 brauchen wir 2 Bits, also auch zwei Bitplanes, die dann so aussehen:

```
00100110    Dies ist Bitplane 0
00101011    Dies ist Bitplane 1
-----
00302132    Nun addieren wir sie, wobei wir die zweite mit 2
              multiplizieren
```

Das ist also die gewünschte Grafik. Nun gäbe es aber natürlich auch eine andere Möglichkeit: Wir könnten die jeweils 2 Bits direkt hintereinander anordnen in sogenannten Chunks:

```
00 00 11 00 01 10 11 01 = 00302132
```

Dies ist das Prinzip des Chunky-Modus. Man kann im allgemeinen kaum sagen, daß eine dieser beiden Methoden besser oder schlechter ist. Allerdings haben natürlich beide ihre Vor- und Nachteile:

Zunächst hat vielleicht jeder schon einmal gesehen, daß auf dem Amiga beim Scollen von farbigen Text ein gewisses Flackern entsteht. Genauer gesagt ändern sich kurz die Farben. Was dabei passiert ist, daß der Computer Bitplanes verschiebt, gleichzeitig aber dieselben Daten für die Anzeige verwendet werden. Wenn etwa gerade Bitplane 0 verschoben wurde, aber Bitplane 1 noch nicht verschoben ist, so hätten wir kurzfristig im obigen Beispiel die folgende Anzeige:

```
01001100    Dies ist Bitplane 0 (nach links verschoben)
00101011    Dies ist Bitplane 1
-----
01203122    Nun addieren wir sie wieder
```

Sobald die zweite Bitplane ebenfalls verschoben ist, stimmt wieder alles, aber kurzfristig entsteht dabei eben jenes Flackern. Bei einer Chunky-Anzeige dagegen wäre eben nur ein Teil des Bildschirms schon verschoben und ein anderer Teil noch nicht.

Umgekehrt ist es im Chunky-Modus schlecht möglich, mit beliebiger Anzahl von Farben zu arbeiten: Da ein Byte 8 Bits hat, gehören etwa bei 4 Farben zu jedem Byte 4 Pixel. Man muß also stets erst berechnen an welcher Stelle des Bytes die Informationen zu einem bestimmten Pixel sitzen. Das ist aufwendig. Noch schlimmer wird die Sache bei 8 Farben: Da beginnen die Pixel noch nicht mal an der gleichen Stelle. Das ist sehr umständlich und rechenzeitaufwendig. In der Praxis gibt es daher Chunky-Anzeigen nur im 8-Bit-Modus (256 Anzeigen) und im 24-Bit-Modus (16 Millionen Farben). Es ist allerdings möglich, daß die Anwender da gar nicht so unglücklich darüber sind...

## 5.2 Was ist Doublebuffering?

Bei animierter Grafik entsteht das Problem, daß gleichzeitig die Daten verändert und angezeigt werden. Dabei kommt es dann unweigerlich zu einem gewissen Flackern. (siehe Abschnitt 5.1 [Chunky vs. Planar], Seite 18)

Die Lösung des Problems ist es, quasi zwei Bildschirme zu benutzen: Der eine wird immer angezeigt. Gleichzeitig wird auf dem anderen Bildschirm, der nicht angezeigt wird das neue Bild gezeichnet. Es gibt kein Flackern, da die Grafik-Hardware nur auf das RAM des ersten Bildschirms zugreift und der Prozessor nur auf das des zweiten. Dann wird umgeschaltet und der zweite Bildschirm angezeigt. Auf dem ersten kann jetzt gezeichnet werden.

## 5.3 Was für Monitore arbeiten am Amiga 1200 oder 4000?

Monitore kann man klassifizieren nach der horizontalen Frequenz, die sie für ihre Anzeige benötigen. Fernseher und Commodore's 1084 benötigen z.B. etwa 15 kHz, VGA und SVGA benötigen mindestens etwa 30 kHz. Multisync-Monitore können verschiedene Frequenzen darstellen.

Man kann also jeden dieser Monitore am A1200 verwenden, **aber**:

- Mit einem gewöhnlichen VGA/SVGA-Monitore kann man nur einige Anzeige-Modi (DblPal, DblNTSC und/oder Productivity, d.h. (320|640) x (256|512|1024) Pixel) verwenden. Dies ist großartig für die Workbench und die meisten ernsthaften Anwendungen, aber manche grafikorientierten Programme, vor allem Spiele laufen fast sicher nicht: Sie benutzen nämlich die Preferences nicht, übernehmen statt dessen einfach die Maschine und gehen von einem 15 kHz Monitor aus. Ferner kann man das Bootmenü nicht verwenden: Das arbeitet nämlich ebenfalls nur mit 15 kHz. Schließlich haben VGA-Monitore keine Lautsprecher und die VGA-artigen Modi unterstützen keine Genlocks.
- Das größte Problem eines 15 kHz-Monitors ist das ständige Flickern im Interlace-Modus. Ein kleiner Tip ist hier die Verwendung von NTSC anstelle von PAL. Dies erhöht die Refresh-Rate von 25 Hz auf 30 Hz, kostet allerdings einige darstellbare Zeilen. (Maximal

möglich sind 482.) Die NTSC- und PAL-Modi sind aber nicht so schlecht, wie die Leute denken: Bei einem Monitor mit viel Phosphor (der also lange nachglüht) ist PAL-Laced einigermaßen akzeptabel und gibt eine Auflösung von 1448x566 in SuperHiRes. Mehr ist auf keinem AGA-Amiga möglich.

- Die beste Lösung sind die Multisync-Monitore, vorausgesetzt sie unterstützen die Bereiche von 15-31 kHz Horizontal- und 50-72 Vertikalfrequenz. Die neuen 1940- und 1942-Monitore von Commodore sind nicht schlecht, allerdings etwas umständlich zu handhaben: Bei jedem Umschalten des Anzeige-Modus muß man nämlich auch die horizontale/vertikale Größe und den Offset von Hand einstellen.<sup>3</sup> Geeignet sind z.B. auch der Mitsubishi EUM 1491 oder der EIZO 9060S.

Für einen VGA/SVGA- oder Multisync-Monitor braucht man ein Kabel, das etwa 30 DM kostet.

## 6 Emulationen

Was? Der Amiga ist nicht gut genug? Sie wollen noch eine andere Maschine? Also gut, dann schauen sie hier...

### 6.1 Kann ich meinen Amiga unter Unix benutzen?

Es gibt derzeit drei Unix-Versionen für den Amiga. Alle benötigen mindestens einen 68030 wegen der MMU (siehe Abschnitt 1.1 [MMU], Seite 1), 68040-Unterstützung ist erst in Arbeit. Alle scheinen Probleme mit manchen Harddisk-Controllern zu haben, man sollte sich deshalb anhand der Dokumentation informieren, ob sie auf der eigenen Maschine überhaupt laufen. Unix benötigt natürlich eine ungeheure Menge von Ressourcen, 8Mb RAM und eine 150Mb-Partition für Unix dürften eher die Untergrenze sein.

1. Commodore hat in der Vergangenheit ein System-V-Unix angeboten. Es enthielt TCP/IP, X11 und andere Software und hat eigentlich einen guten Eindruck gemacht, aber war teuer und wird vor allem nicht mehr weiterentwickelt. Commodore-Unix benötigt einen Streamer, denn es wird auf solchen Bändern ausgeliefert.
2. Ein Linux-Port wird vorbereitet, allerdings gibt es derzeit nur den Kernel, das heißt die untersten Funktionen des Betriebssystems. Für Spezialisten könnte es interessant sein, kann aber wohl nicht empfohlen werden. Linux ist frei kopierbar. Quellen: [ftp.uni-paderborn.de](ftp://ftp.uni-paderborn.de/pub/amiga/linux), Directory '/pub/amiga/linux' oder [ftp.uni-erlangen.de](ftp://ftp.uni-erlangen.de/pub/LINUX/MIRROR.tsx-11/680x0), Directory '/pub/LINUX/MIRROR.tsx-11/680x0'.
3. NetBSD ist ebenfalls frei kopierbar. Wie Linux ist es noch nicht fertig, aber macht derzeit gute Fortschritte. Beispielsweise läuft wohl schon die meiste GNU-Software, vor

---

<sup>3</sup> Es gibt einen Patch, der dies auch ohne manuelle Eingriffe möglich macht. Siehe 'os30/util/Monitor30Patch.lha' auf dem Aminet.

allem Emacs und gcc. Es ist auf jeden Fall einen Blick wert. Quellen: `ftp.uni-paderborn.de`, Directory `‘/pub/amiga/netbsd’` oder `ftp.uni-erlangen.de`, Directory `‘/pub/amiga/unix/NetBSD’`.

## 6.2 Ist es möglich, den Amiga als X11-Terminal zu benutzen?

Es gibt zwei Softwarepakete, die das möglich machen:

GfxBase bietet ein kommerzielles Paket an. Es enthält verschiedene Window-Manager und Clients. Eine Demo-Version ist auf dem Aminet. (`‘gfx/x11/GfxBase-X11-Demo.lha’`)

Frei kopierbar ist DaggeX. Allerdings ist dieses möglicherweise noch nicht ganz fertig, zumindest wird die Versionsnummer mit 0.91 angegeben. Zu finden ist es ebenfalls auf dem Aminet. (`‘gfx/x11/DaggeX-0.91.lha’` und `‘gfx/x11/twm_930531.lha’`)

## 7 Verschiedenes

Dieses Kapitel enthält Dinge, die nicht in die anderen passen.

### 7.1 Gibt es eine Unix-Version von LhA.

Siehe Abschnitt 7.2 [Endungen], Seite 21.

### 7.2 Was sind Dateien, die mit ... enden?

Die meisten dieser Endungen sagen einem, daß die betreffende Datei komprimiert (gepackt) ist oder ein Archiv aus mehreren Dateien ist oder beides. (Einige Programme können sogar ganze Disketten archivieren.) Übliche Endungen und die dazugehörigen Programme sind:

- .sfx**            Gepackte Archive, die in ein ausführbares Programm eingebunden sind; dieses wird einfach aufgerufen, um sich selbst zu entpacken. (sfx = self extract)
- .lha**
- .lzh**            Gepackte Archive; empfohlen: LhA (`‘util/arc/LhA_e138.run’` auf Aminet oder Fish-Disk 715) oder Lx (`‘util/arc/lx100.lha’` auf Aminet), es gibt auch eine Unix-Version (`‘misc/unix/lha-1.00.tar.Z’`)
- .dms**            Mit DMS komprimierte Disketten; Quellen: Aminet (`‘util/arc/dms111.sfx’`) oder Fish-Disk 406
- .zom**            Mit Zoom komprimierte Disketten; Quellen: Aminet (`‘util/arc/Zoom_5.4.lha’` oder Fish-Disk 682); eine ältere Version findet man auf Fish-Disk 459, diese ist möglicherweise für das Entpacken von PasTeX nötig

<b>.zoo</b>	Komprimierte Archive; empfohlen: Zoo ('util/arc/zpp2-10.lzh' auf Aminet oder Fish-Disk 527)
<b>.Z</b>	
<b>.z</b>	
<b>.gz</b>	Komprimierte Dateien; empfohlen: Gzip ('util/pack/gzip124x.lha' auf dem Aminet). diese Dateien sind meist Unix-Dateien
<b>.tar</b>	Archive; empfohlen: tar ('util/arc/tar.lha' oder 'util/arc/gtar10.lha' auf Aminet oder Fish-Disk 445), ebenfalls meist Unix-Dateien. Man findet häufig .tar.Z oder .tar.gz.
<b>.arj</b>	Komprimierte Archive; empfohlen unarj ('util/arc/unarj-0.5.lha' auf Aminet)
<b>.zip</b>	Komprimierte Archive; empfohlen UnZip ('util/arc/unzip-5.1.lha' auf Aminet). Dies sind meist MS-Dos-Archive.

### 7.3 Gibt es ein Programm wie Stacker, um die Hard-Disk zu packen?

XFH ist eine gute Möglichkeit. Es arbeitet als Handler und benützt die XPK-Libraries, man kann also zwischen verschiedenen (und in Zukunft vielleicht noch weiteren) Komprimiermodi wählen. (NUKE ist eine gute Wahl. Der einzige Nachteil ist, daß die Größe von Dateien durch das verfügbare RAM beschränkt ist, unter 2MBytes RAM kann man Probleme bekommen.

Eine andere Möglichkeit ist EPU. Es ist Shareware und sollte dasselbe wie XFH bieten und außerdem ohne die Probleme mit der Dateigröße.

Quellen: Aminet, Directory 'util/pack' und Fish-Disk 754 (XFH) sowie 858 (EPU).

### 7.4 Wo bekomme ich Fish-Disk xxx?

Einige FTP-Server haben genügend Platz oder ein CD-Rom gemounted und haben alle Fish-Disks online verfügbar:

ftp.isca.uiowa.edu	(USA, directory '/amiga/fx/fxxx')
ftp.hawaii.edu	(USA, directory '/pub/amiga/fish')
ftp.dfv.rwth-aachen.de	(Germany, directory '/pub/amiga/fish')
ftp.uni-erlangen.de	(Germany, directory '/pub/amiga/pd')
ftp.funet.fi	(Finland, directory '/pub/amiga/fish')

Beachten Sie bitte, daß die CD-Rom's nicht immer gemounted sind. Siehe Abschnitt 8.3 [FTP], Seite 24.

Eine andere Moeglichkeit wäre, Ihren oertlichen PD-Händler zu fragen. :-)

## 7.5 Wie füllt man die Tintenkartuschen der HP Deskjet-Drucker nach?

Dies ist eigentlich keine Amiga-spezifische Frage, taucht aber in den Amiga-Newsgruppen so hartnäckig und regelmäßig auf, daß sie auch in dieser FAQ beantwortet wird.

Die Drucker der Deskjet-Serie von HP besitzen einen in den Druckkopf integrierten Tinten-Vorratsbehälter. Wenn dieser Behälter leer ist, muß normalerweise der komplette Druckkopf ausgetauscht werden. Doch es geht auch billiger: Man kann den Tintenbehälter mit etwas Übung problemlos mehrfach nachfüllen.

Man benötigt dazu eine Einwegspritze mit passender Nadel, etwas schwarze Füllfederhalter-Tinte (z.B. Pelikan 4001), sowie Isopropanol (a.k.a. Isopropylalkohol, erhältlich in jeder gutsortierten Apotheke).

Als erstes gibt man in das neu gekaufte Tintenfläschchen einige Tropfen Isopropanol (ca. 0,5-1,0 ml pro 30 ml Tinte). Für eine Füllung des Druckkopfes zieht man dann etwa 10-15 ml der Tintenmischung in die Spritze, sticht mit der Nadel in die Öffnung im oberen (grünen) Teil des Druckkopfes und spritzt dann die Tinte langsam und vorsichtig in den Druckkopf. Achten Sie darauf, daß Sie die Nadel nicht ganz "bis zum Anschlag" einstecken, während des Einspritzens muß die verdrängte Luft noch durch die Öffnung entweichen können, sonst tritt die Tinte eventuell durch die Düsen am unteren Ende des Druckkopfes aus.

Die Dosierung des Isopropanols ist etwas kitschig; zuviel fördert das Verlaufen der Tinte auf dem Papier und führt zu einem unsauberem Schriftbild, zuwenig führt eventuell zu verstopften Düsen am Druckkopf.

Es gibt mittlererweile von verschiedenen Herstellern auch sogenannte Nachfüllkits, die passende Spritzen und fertig vorbereitete Tintenmischungen enthalten. Preislich liegen diese "Fertiglösungen" etwa in der Mitte zwischen der Füllertinte und einem neuen Druckkopf.

Die neuen Spezialdruckköpfe mit doppeltem Volumen lassen sich übrigens angeblich nicht mehr nachfüllen - es wäre schön, wenn das jemand mal aus eigener Erfahrung bestätigen oder verneinen könnte...?

Jürgen Weinelt, jow@rz.uni-wuerzburg.de

## 8 Wo und wie bekomme ich Software?

Drei Fragen entstehen in diesem Zusammenhang: Was für Programme gibt es überhaupt, wo und wie bekomme ich sie und wie kann ich sie dann nach Hause bringen?

### 8.1 Dateien und Datenbanken zur frei kopierbaren Software

Natürlich muß man wissen, wo man welche Software überhaupt findet. Viele wichtige Dinge sind bereits angegeben worden, wie ich hoffe. Weitere Informationen liefern:

**AmigaSciUnixSchool**

ist eine Software-Liste im Ascii-Format. Sie wird monatlich in den Newsgroups `news.groups.comp.sys.amiga.applications`, `comp.unix.amiga`, und `news.answers` gepostet. (Aminet: `'text/doc/AmigaSciUnixSchool-4.01'`). Sie behandelt alles, was auch hier angegeben wurde und vieles mehr, z.B. GNU-Software, Libraries (Link-Libraries und shared Libraries), Shells, Unix-Kommandos, wissenschaftliche Software und vieles mehr.

**FishCon** sind die gesammelten Inhaltsverzeichnisse der Fish-Disketten im Ascii-Format. (Aminet: `'fish/doc/fishcon-???.lzh'`)

**FishXref** ist ein Kreuzreferenzverzeichnis der FishCon-Dateien, ebenfalls im Ascii-Format (`'fish/doc/fishxref-???.lzh'` auf Aminet)

**KingFisher**

Eine Fish-Disk-Datenbank, getrennt in Programm (Fish-Disk 863 oder Aminet, `'fish/doc/Kingfisher1_30.lha'`), und Datendatei `'fish/doc/KFData850.lha'`, die die Suche nach Namen oder Kontext ermöglichen.

## 8.2 Eine Sammlung von Testberichten

`comp.sys.amiga.reviews` ist eine moderierte Newsgruppe, in der ausschließlich Testberichte über Soft- und Hardware, Bücher und alles mögliche andere, den Amiga betreffendes veröffentlicht werden. Es ist immer eine gute Idee, hier nachzuschauen, wenn man an etwas Bestimmtem interessiert ist. Natürlich findet man in der eigentlichen Newsgruppe nur die jeweils neuesten Berichte, aber die älteren werden archiviert und sind per FTP erhältlich bei `math.uh.edu`, Directory `'/pub/Amiga/comp.sys.amiga.reviews'` oder auf den Fish-CD's.

## 8.3 Empfangen von Dateien von einem FTP-Server

Software zu laden ist einfach, wenn man Zugang zum Internet mit einem Programm namens FTP (File Transfer Program) hat. Unix-Computer haben häufig beides.

FTP erlaubt Zugriffe auf andere Maschinen und speichern und/oder Laden von Dateien. Natürlich braucht man eine Zugangsberechtigung auf der anderen Maschine, aber viele Maschinen erlauben Zugang für jedermann/frau, wenn man sich als Benutzer `ftp` oder `anonymous` anmeldet und als Paßwort die eigene Mailadresse angibt. Für Amiga-Besitzer sind die wichtigsten FTP-Server die Aminet-Server, die sich gegenseitig Dateien übertragen und so im wesentlichen dieselben Dateien anbieten. Aminet-Server sind

USA (MO)	<code>ftp.wustl.edu</code>	128.252.135.4
USA (CA)	<code>ftp.cdrom.com</code>	192.153.46.2
USA (TX)	<code>ftp.etsu.edu</code>	192.43.199.20
Scandinavia	<code>ftp.luth.se</code>	130.240.18.2
Germany	<code>ftp.uni-kl.de</code>	131.246.9.95
Germany	<code>ftp.uni-erlangen.de</code>	131.188.1.43
Germany	<code>ftp.cs.tu-berlin.de</code>	130.149.17.7
Germany	<code>ftp.uni-paderborn.de</code>	131.234.2.32

Germany	ftp.uni-oldenburg.de	134.106.40.9
Germany	ftp.coli.uni-sb.de	134.96.68.11
Switzerland	ftp.eunet.ch	146.228.10.16
Switzerland	litamiga.epfl.ch	128.178.151.32
UK	ftp.doc.ic.ac.uk	146.169.2.1

Alle diese Server haben ein Directory '/pub/aminet', wo man massig Software findet. Bitte benutzen Sie einen Server in Ihrer Nähe! Einige andere wichtige Server sind

ftp.funet.fi	(Finnland)
ftp.isca.uiowa.edu	(USA)
ftp.hawaii.edu	(USA)
ftp.cso.uiuc.edu	(USA)
ftp.dfv.rwth-aachen.de	(Deutschland)

Grind, Aachen und Erlangen haben z.B. die komplette Fish-Disk-Serie parat! Siehe Abschnitt 7.4 [Fish-Disk xxx], Seite 22.

Um sich mit einem Server in Verbindung zu setzen (z.B. ftp.uni-erlangen.de), gibt man ein:

```
ftp ftp.uni-erlangen.de
```

Der Server antwortet mit der Aufforderung, den Benutzernamen einzugeben. Ale Benutzernamen gibt man

```
ftp
```

ein. Nun wird man nach einem Paßwort gefragt. Hier sollte man seine Mailadresse (wenn man eine hat, sonst einfach ftp) eingeben.

Nun ist man mit dem Server verbunden und kann eine Reihe von Kommandos ausführen. Die wichtigsten sind:

**?** Gibt einen Hilfstext aus. Man kann auch **? Kommando** eingeben, um Hilfe zu einem bestimmten Kommando zu verlangen.

**bin** Informiert FTP, daß man binäre Dateien transportieren will. Es ist immer eine gute Idee, dies als allererstes Kommando einzugeben! Ohne dieses Kommando können empfangene Dateien verändert und damit nutzlos sein.

**get <Datei>**

Lädt die angegebene Datei vom Server. Auf den meisten Unix-Maschinen kann man auch 'get file.txt -' oder 'get file.txt |more' eingeben, um sich die angegebene Datei auf den Bildschirm ausgeben zu lassen. (Achtung: Hier darf **kein** Blank zwischen | und dem Wort more sein!)

**mget <pat>**

Lädt die angegebenen Dateien. Im Unterschied zu get dürfen hier auch Unix-Wildcards (\* oder ?) verwendet werden.

**put <file>**

**mput <pat>**

Wie get und mget, aber es werden Dateien *zum* Server geschoben. Dies ist meist nur in speziellen Directories mit Namen wie 'incoming' oder 'new' erlaubt. Man kann dort Dateien plazieren, die man auf dem Aminet frei zugänglich machen will.

**cd <dir>** Wie das übliche cd. Die Kommandos get, mget, put, mput, dir und ls beziehen sich auf das angegebene Directory.

**dir [<dir>]**

**ls [<dir>]** Wie 'list' und 'dir' auf dem Amiga. Beachten Sie allerdings, daß FTP-dir dem Amiga-list entspricht.

**bye** Verläßt das FTP-Programm.

Wenn man FTP das erste Mal benutzt hat, wird man feststellen, daß immer die gleichen Schritte ausgeführt werden:

1. Benutzernamen eingeben (meist ftp)
2. Paßwort eingeben (meist die Mailadresse)
3. bin eingeben
4. In ein bestimmtes Directory wechseln (meist '/pub/aminet/...')

Dies kann man automatisieren. Dazu braucht man eine Datei namens '.netrc'. in seinem Home-Directory. Diese muß unbedingt nur für Sie selbst lesbar sein, FTP akzeptiert sie sonst nicht! (Dies erreichen Sie unter Unix mit dem Kommando 'chmod go-rwx .netrc'.) Die .netrc-Datei enthält für eine Reihe von Servern je einen Eintrag, die durch Leerzeilen getrennt werden. Ein typischer Eintrag sieht etwa so aus:

```
machine ftp.uni-erlangen.de
login ftp
password <Ihre Mailadresse> oder <ftp>
macdef init
    bin
    cd pub/aminet
```

Auf einigen Computern ist auch der Servername 'default' erlaubt, der für alle andere Maschinen außer den angegebenen gilt.

## 8.4 Empfangen von Dateien von einem Mail-Server

Eine andere Möglichkeit, Dateien zu empfangen sind die Mail-Server. Dazu braucht man die Möglichkeit, an Internet-Adressen Mail zu verschicken und zu empfangen. Es funktioniert, indem man an den Server eine Mail schickt, indem man ihm sagt, was man haben möchte. Die Dateien werden dann ebenfalls als Mail geschickt, allerdings kodiert. Man braucht ein Programm namens `uudecode`, um sie zu dekodieren.

Die wichtigsten Mail-Server sind:

```

ftpmail@decwrl.dec.com
mailserver@nic.funet.fi
ftp-mailer@ftp.informatik.tu-muenchen.de
mrcserv@janus.mtroyal.ab.ca
mail-server@ftp.cs.tu-berlin.de
mail-server@rtfm.mit.edu

```

Die an einen Server zu schickende Mail darf eine Reihe von Kommandos enthalten. Die wichtigsten sind:

**Help**        Veranlaßt den Server, einem einen Hilfstext zu mailen, indem eine umfangreiche Anleitung enthalten ist.

**Limit <Anzahl>**

Gibt an, daß eine einzelne Mail höchstens <Anzahl> KByte lang sein darf. Größere Dateien werden in mehrere kleinere aufgeteilt, die als separate Mails verschickt werden. Beachten Sie, daß einzelne Mails durch die Deodierung und den Mailheader auch etwas länger sein können!

**Cwd <dir>**

Wie der cd-Befehl; das angegebene Directory wird von den Kommandos send und dir benutzt.

**Index**        liefert eine Liste von Dateien und/oder Directories, die der Server anbietet. Diese Liste kann **sehr** lang werden! (Berlin z.B. 1 MByte)

**Index <item>**

liefert eine Liste von Dateien, deren Namen <item> enthält.

**Dir [<dir>]**

liefert eine Liste von Dateien und Directories im angegebenen Directory

**Send <file1> <file2> ... <fileN>**

Liefert einem die angegebenen Dateien

**Begin**        Veranlaßt den Server, alle Zeilen oberhalb zu informieren.

**End**         Wie Begin, aber für die unten folgenden Zeilen. (Eine Signatur zum Beispiel!)

Eine typische Mail an einen Mail-Server sieht also so aus:

```

BEGIN
CD /pub/aminet/util/arc
SEND LhA_e138.run
END

```

## 8.5 Die Fish-Disketten

Eine sehr gute Quelle sind die Fish-Disketten. Man muß inzwischen Floppies und CD's unterscheiden.

### 8.5.1 Die Amiga-Library-Disks

Fred Fish hat in der Mitte der Achtziger begonnen, frei kopierbare Software zu sammeln. Es gibt inzwischen mehr als 900 Disketten und viele sehr gute Software darauf. Die meisten Amiga-Händler bieten sie an und die meisten Amiga-Zeitschriften enthalten Anbieter, die sie für ca. 3 DM pro Diskette mit der Post versenden.

Es gibt Dinge, die auf den Fish-Disketten, nicht aber auf dem Aminet verfügbar sind. Trotzdem ist es möglich, sie per FTP zu bekommen. Siehe Abschnitt 7.4 [Fish-Disk xxx], Seite 22.

Die Fish-Serie wird es in dieser Form nicht mehr lange geben. Beginnend mit Nummer 1000 will Fred Fish nur noch CD-Roms herausgeben. Es ist noch nicht ganz klar, wie diese aussehen werden und ob deren Software auch auf Disketten erhältlich sein wird. Siehe Abschnitt 8.5.2 [Fresh-Fish-Serie], Seite 28.

### 8.5.2 Die Fresh-Fish-Serie

Fred Fish bietet weiterhin frei kopierbare Software an, allerdings jetzt auf CD. Es gibt zwei verschiedene Arten:

1. Die monatlichen CDs werden in drei Teile unterteilt sein:
  1. Neues Material, sowohl auf den Disketten erscheinende als auch dort nicht erscheinende Software. Auf der ersten CD sind das etwa 84Mb.
  2. Nützliche, bereits installierte Software, die direkt von der CD benutzt werden können und damit Platz auf der Festplatte sparen. (GNU Emacs, Gnu C, GNU C++, Amiga E, PasTeX, AmigaGuide, Installer, 2.0 und 3.0-Includes, verschiedene Archivierprogramme, das AmiCDROM Filesystem, GNU und BSD-Utilities...) Dieser Teil macht etwa 150Mb auf der ersten Disk aus.
  3. Älteres Material, das bereits früher erschienen ist. (Ungefähr 404Mb auf der ersten CD, entsprechend Fish-Disk 600-910.)
2. Die zweite Sorte wird im wesentlichen mehr Software, dafür aber in gepacktem Format enthalten. (Diese CD's sind speziell für Mailboxen gedacht, die dadurch diese Software zum Downloaden anbieten.)

Ich empfehle vor allem die erste Sorte. Sie kosten etwa 20\$ plus 5\$ Versandkosten und können bei der folgenden Adresse bestellt werden:

Amiga Library Services  
610 N. Alma School Road, Suite 18  
Chandler, AZ 85224-3687  
U.S.A.

Phone/FAX: (602) 917-0917

Als beste Zahlungsweise haben sich angeblich Kreditkarten bewährt. Allerdings bin ich überfragt, welche von Fred akzeptiert werden.

## 8.6 Wie kann ich MS-Dos-Disketten lesen und schreiben?

Für Besitzer von Workbench 2.1 oder höher ist das kein Problem: Das Programm CrossDos ist da Teil der Workbench. Man muß lediglich 'pc0:' mounten, indem man die Datei 'Sys:Storage/DOSDrivers/pc0:' startet oder nach 'Devs:Storage/DOSDrivers' verschiebt. MD-Dos-Disketten in 'df0:' können nun ganz normal behandelt werden, indem man jeweils das Wort 'df0:' durch 'pc0:' ersetzt. Z.B. kann man das Directory mit 'dir pc0:' anzeigen.

Alle anderen benötigen ein Programm namens msh. (Aminet, Directory 'misc/emu' oder Fish-Disk 382) Nachdem man die Datei 'devs:MountList' wie in der Dokumentation angegeben verändert hat, muß man lediglich im CLI das Kommando 'mount msh:' eingeben und kann dann wie bei CrossDos damit verfahren, wobei man natürlich jeweils 'msh:' anstelle von 'pc0:' angeben muß.

## 8.7 Wie transportiere ich sehr große Dateien

Es gibt einige Archive, die zu groß sind, um auf eine Diskette passen. (Das gcc-Archiv ist z.B. 3,5 MByte groß.) Um diese zu transportieren, benötigt man ein Programm, das sie in kleinere Teile aufteilt, die dann auf verschiedenen Dateien transportiert werden. Ich empfehle Martin Schlodders Splitter. (Aminet, 'util/misc/splitter\_121.lha') Das Archiv enthaelt Binaries fuer den Amiga und MS-DOS und der Quelltext sollte ohne Probleme auf jedem Unix-Rechner zu compilieren sein.

## Das Amiga-FAQ-Archiv

Die Amiga-FAQ ist in verschiedenen Formaten erhältlich: Im Ascii-Format (wie sie in den Netzen geposted wird), im AmigaGuide-Format (wie sie auf einem Amiga wohl am praktischsten ist) und im dvi-Format zum Ausdrucken. Weiter gibt es einige Dinge, die nützlich oder interessant sein könnten, die aber nicht in den Text eingebunden werden konnten:

txt/amiga.history	Zur Geschichte des Amiga
txt/story.txt	Die Commodore-Story (oder: Die Tramiel-Story ;-)
txt/amiga.newsgroups	Übersicht über comp.sys.amiga.*
txt/amiga.sites	Liste von FTP-Servern
txt/AmigaOverview.tex	Übersicht über die Amiga-Soft- und Hardware
txt/Hardware.tips	Für Hardware-Bastler
txt/Nullmodem.txt	Anleitung zum Bau eines Nullmodems
src/JWSplit.c	Der Quelltext eines Dateisplitters
src/JWJoin.c	Das Gegenstück zu JWSplit.c
src/addtoc.c	Fügt ein Inhaltsverzeichnis zu mit texinfo erzeugten doc's bei (dieser Text verwendet es)

Um diese öffentlich zugänglich zu machen, habe ich mich dazu entschlossen, diese in einem Archiv zu sammeln. Es heißt 'AmigaFAQxxxxxxg.lha', wobei xxxxxx das Datum der letzten Version ist. Sie finden es per FTP im Aminet, Directory 'text/docs'.

## Beiträge zur Amiga-FAQ

Die FAQ kann nicht nützlich sein und nicht weiterentwickelt werden ohne Ihre Hilfe. Vorschläge, Beiträge, neue Fragen und Antworten, Kritik, alles ist willkommen.

Bitte beachten Sie, daß einige sehr wesentliche Themen bis jetzt unterrepräsentiert sind oder gänzlich fehlen: Nichts über Animation, Sound, Grafikkarten. (Alles über die *endgültige* Merlin-Software ...) Dies sind einige der Stärken des Amiga! Aber ich verstehe nichts davon :-)

Also nehmen Sie ihre Tastatur (Ihren Bleistift? Na gut, wenn's sein muß..) und schreiben Sie an:

Jochen Wiedmann  
Am Eisteich 9  
72555 Metzingen (Deutschland)  
Tel. 07123 / 14881

Internet: [wiedmann@mailserv.zdv.uni-tuebingen.de](mailto:wiedmann@mailserv.zdv.uni-tuebingen.de)

## Danksagungen

Meinen Dank an:

**Reinhard Spisser and Sebastiano Vigna**

für die Amiga-Version von TexInfo. Dieser Text wurde damit erstellt.

**The Free Software Foundation**

für die Originalversion von TexInfo und viele andere hervorragende Programme.

**Dylan McNamee**

für die Abschnitte über Editoren, Textverarbeitungen, DTP und PostScript.

**Joseph Luk**

für die Abschnitte über Chunky/Planar und Double-Buffering

**Urban Dominik Mueller**

für die FAQ über FTP- und Mail-Server

# Index

•	
.arj	21
.dms	21
.gz	21
.lha	21
.lzh	21
.netrc	24
.tar	21
.z	21
.Z	21
.zip	21
.zom	21
.zoo	21
<b>6</b>	
68EC020	1
68EC030	1
68LC040	1
<b>A</b>	
Amiga-FAQ-Archiv	29
Amiga-Library-Disks	28
AmigaBasic	8
AmigaSciSchool	23
Anonymous	24
Assembler	6
AutoDocs	4
<b>B</b>	
Beiträge	30
<b>C</b>	
C	6
C++	6
Catalog description	9
Catalog translation	9
CatComp	9
CATS	4
Chunky	18
Commodore, Frankfurt	5
Commodore, West Chester	4
Compiler	5
Console window	11
CrossDos	29
<b>D</b>	
DaggeX	21
Danksagungen	30
Dateiendungen	21
Desktop Publishing	16
Developer	5
Dokumentation	3
DoMethod	14
DoSuperMethod	14
Doublebuffering	19
Druckersteuerung	8
DTP	16
<b>E</b>	
Editoren	15
Emulationen	20
Endungen	21
Enforcer	1
Esc-Sequenzen	8
<b>F</b>	
FD-files	12
fd2pragma	12
Fehlende Funktionen	14
Fish-CD-Rom's	28
Fish-Disketten	27
Fish-Disks	22
Fish-Floppies	28
FishCon	23
FishXref	23
FlexCat	9
Forth	6
Fortran	6
FPU	1
Fresh Fish CD-Rom's	28
FTP-Servers	24
<b>G</b>	
Geschichte	29
GfxBase	21
GigaMem	1

Grafik ..... 18

## H

HD-Kompression ..... 22

Hirsch & Wolf ..... 4

HookEntry ..... 14

HP-Deskjet ..... 23

## I

Include-Dateien ..... 4

## K

Kartuschen ..... 23

Kataloge ..... 9

KingFisher ..... 23

KitCat ..... 9

Konsolenfenster ..... 11

## L

LibAllocPooled ..... 14

Linux ..... 20

Lisp ..... 7

locale.library ..... 9

Localisierung ..... 9

## M

Mail-server ..... 26

MakeCat ..... 9

Manuale ..... 3

MMU ..... 1

Modula-2 ..... 7

Monitore ..... 19

MS-Dos-Disketten ..... 29

Msh ..... 29

Multiscan ..... 19

## N

NDA ..... 5

NDK ..... 4

NDU ..... 4

NDUK ..... 4

NetBSD ..... 20

## O

Oberon ..... 7

## P

Packer ..... 21

Packer on Unix ..... 21

Pascal ..... 7

Planar ..... 18

PostScript ..... 17

Pragmas ..... 12

Prolog ..... 7

## R

RAM, virtuelles ..... 1

RKM's ..... 3

ROM Kernel Manuals ..... 3

## S

Splitten von Dateien ..... 29

Stacker ..... 22

## T

TeX ..... 17

Text-Editoren ..... 15

Textbearbeitungssprachen ..... 15

Textverarbeitung ..... 15

Tintenstrahldrucker ..... 23

## U

Unix ..... 20

Unix-LhA ..... 21

## V

VGA ..... 19

## W

Wysiwyg ..... 15

## X

X11 ..... 21

XFH ..... 22

XPk ..... 22

# Table of Contents

<b>1</b>	<b>Hardware</b> .....	<b>1</b>
1.1	Was sind 68EC020, 68EC030 und 68LC040? .....	1
1.2	Was ist ein mathematischer Coprozessor (FPU) ? .....	1
<b>2</b>	<b>Das Betriebssystem</b> .....	<b>2</b>
2.1	Kann ich eine andere als die eingebaute Kickstart benutzen? ....	2
<b>3</b>	<b>Programmierung</b> .....	<b>2</b>
3.1	Was ist die beste Dokumentation für Programmierer? .....	3
3.2	Was ist CATS? .....	4
3.3	Wo bekomme ich die Amiga-Include-Dateien? .....	4
3.4	Wie werde ich Developer? .....	5
3.5	Was für Compiler (und Assembler) gibt es? .....	5
3.6	Warum funktioniert keine Esc-Sequenz? .....	8
3.7	Kann ich AmigaBasic auf dem A1200 verwenden? .....	8
3.8	Wie lokalisiere ich mein Programm? .....	9
3.9	Wie erhält man einen Zeiger auf das Fenster einer Konsole? ....	11
3.10	Was sind Pragmas? .....	12
3.11	Wo finde ich die Funktion xxx? .....	14
<b>4</b>	<b>Anwendungen</b> .....	<b>15</b>
4.1	Text-Editoren .....	15
4.2	Was für Textverarbeitungsprogramme gibt es? .....	15
4.3	Desktop Publishing .....	16
4.4	Was ist TeX und wo bekomme ich es? .....	16
4.5	Gibt es PostScript-Interpreter auf dem Amiga? .....	17
<b>5</b>	<b>Grafik</b> .....	<b>18</b>
5.1	Was heißt Chunky- und Planar-Display? .....	18
5.2	Was ist Doublebuffering? .....	19
5.3	Was für Monitore arbeiten am Amiga 1200 oder 4000? .....	19
<b>6</b>	<b>Emulationen</b> .....	<b>20</b>
6.1	Kann ich meinen Amiga unter Unix benutzen? .....	20
6.2	Ist es möglich, den Amiga als X11-Terminal zu benutzen? .....	21
<b>7</b>	<b>Verschiedenes</b> .....	<b>21</b>
7.1	Gibt es eine Unix-Version von LhA. ....	21
7.2	Was sind Dateien, die mit ... enden? .....	21

7.3	Gibt es ein Programm wie Stacker, um die Hard-Disk zu packen?	22
7.4	Wo bekomme ich Fish-Disk xxx?	22
7.5	Wie füllt man die Tintenkartuschen der HPDeskjet-Drucker nach?	23
<b>8</b>	<b>Wo und wie bekomme ich Software?</b>	<b>23</b>
8.1	Dateien und Datenbanken zur frei kopierbaren Software	23
8.2	Eine Sammlung von Testberichten	24
8.3	Empfangen von Dateien von einem FTP-Server	24
8.4	Empfangen von Dateien von einem Mail-Server	26
8.5	Die Fish-Disketten	27
8.5.1	Die Amiga-Library-Disks	28
8.5.2	Die Fresh-Fish-Serie	28
8.6	Wie kann ich MS-Dos-Disketten lesen und schreiben?	29
8.7	Wie transportiere ich sehr große Dateien	29
	<b>Das Amiga-FAQ-Archiv</b>	<b>29</b>
	<b>Beiträge zur Amiga-FAQ</b>	<b>30</b>
	<b>Danksagungen</b>	<b>30</b>
	<b>Index</b>	<b>31</b>