Amiga-FAQ

Ignaz Kellerer

Copyright © Ignaz Kellerer
Georg-Habel-Str. 11
81241 Munich (Germany)
Tel. (+49) 089 / 885147
Internet: kellerer@informatik.tu-muenchen.de

This document lists some frequently asked questions and tries to give answers. Its intention is to help new users and to reduce the amount of news that most experienced users don't like to read anymore.

The author gives **absolutely no** warranty that the answers given here are correct. Many of them were contributed by other users and I cannot even make marginal checks. If you think that something should be changed, please tell me. Suggestions, contributions, new answers, critics, flames (oh, how I like this 'nil:' :-) are rather welcome.

Please notice that there are many questions that aren't answered yet, even more: Whole sections that remain empty! I feel that I cannot give satisfying answers. So it's your turn: Fill the gaps and tell me what i should include into this document!

1 CPU, Custom chips, RAM and other stuff

This chapter contains questions concerning the Amiga-Hardware.

1.1 What are the 68EC020 and the 68EC030?

Motorola, the company producing the 680x0 family offers crippled versions of their processors. They are a little bit cheaper than the originals, that's why Commodore decided to build the 68EC020 into the A1200 and the 68EC030 into the A4000/030.

The difference between the 68020 and the 68EC020 is that the latter can address just 16Mb of memory. That's why the A1200 cannot have more that 10 Mb RAM. In most cases you will not notice the difference.

This is not the case for the 68EC030: Many owners will notice that the 68030 has an MMU and the 68EC030 doesn't have. There are some important programs depending on an MMU, for example Enforcer (a debugging utility), GigaMem (a program to emulate virtual memory) or all current Unix versions (see Section 6.1 [Unix], page 32). Other Amigas than the A4000 and the A3000 need an additional processor card to run these.

Finally the 68LC040 is a 68040 without FPU. See Section 1.2 [FPU], page 1.

1.2 What's an FPU?

The first 680x0 processors (upto 68030) could process integers only. Floating point operations had to be emulated by the software. An FPU is a chip (or part of a chip) that can process floating point operations, a mathematical coprocessor.

One separates three FPU types on the Amiga: The 68881, 68882 and the 68040's internal FPU. The 68882 is up to 1.5 times faster than the 68881, because it is splitted in two parts: A conversion unit (the FPU's are using an 80 bit format internally) and the arithmetic unit. The 68040's internal FPU adds a pipeline, but misses the trigonometric instructions of the others. These are still emulated by the software, 68040.library for example.

Special programs (Raytracing, DTP, Mathematics, TeX) are offered in a special coprocessor version which are up to 50 times faster than the original versions.

Michael Kaiser (kaiser@ira.uka.de)

1.3 Can I use a 3.5" HD in my A1200?

Many people would prefer to use a 3.5" instead of the smaller 2.5" drives as the former are much cheaper. This is possible, but you need a special cable to connect the drive to your IDE controller. Additionally you should put some isolating material between the drive and the main board. Some people report termic problems but I did not notice anything.

Some dealers offer a set which contains the cable, isolating material and for about 20$. See the magazines for more information.

Thomas Schuh (thomas@stepout.tynet.sub.org)

2 The Operating System

This chapter handles questions concerning the operating system, Kickstart as well as the Workbench.

2.1 Can I use another Kickstart than the builtin?

First let's drop some words on the Kickstart's Copyright: This belongs to Commodore, you **must not** use Kickstarts, without the right to do it! Especially it isn't allowed to make an image of anyone else's Kickstart and run this on your own Amiga. (I even doubt that it is allowed to do this on your own Amigas, if you have more than one.)

But of course it is possible and allowed for some people, developers for example. There are two different ways, a hardware solution and a software solution. The former is to buy a card which can hold two or more Kickstart ROMS and allows to select between when the System is booting.

The software solution needs a program (softkicker) and an image of the ROM. The softkicker allocates RAM, loads the ROM image into the allocated memory and reboots. Of course you have less RAM after the Reboot: 256Kb when running Kickstart 1.2 or 1.3 and 512Kb for Kickstart 2.0 or higher.

There are different Softkickers, some of them needing a MMU (see Section 1.1 [MMU], page 1). I recommend SKick 3.43 (Aminet, directory 'util/boot') because it doesn't need a MMU and supports many different Kickstarts. It is rather easy to create the ROM image using the following program:

```
#include <stdio.h>

#define kickorig 0xf80000   /*  0xfc0000 for Kick 1.2 und 1.3   */
#define kicklen  0x080000   /*  0x040000 for 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);
}
```

2.2 What is the Amiga equivalent of . (Current directory)?

The AmigaDOS equivalent for the . representing the current directory on Unix and certain non-reentrant interrupt handlers is the empty string, which is written as `""`.

Example:

```
COPY S:Startup-Sequence ""
```

copies your Startup-Sequence to the current directory.

There is some FD software available to patch a Unix-like interpretation of . and .. into the AmigaDOS, including, but not limited to, Martin Scott's UnixDirs. (Aminet, '`os20/util/UnixDirsII.lha`' or Fish-disk 837)

Arno Eigenwillig (arno@yaps.dinoco.de)

2.3 The PIPE: queue-handler

In the AmigaOS release 2.04, a new handler for passing data between programs was introduced. This handler is called '`L:Queue-Handler`', better known as the '`PIPE:`' device.

'`PIPE:`' implements a true `pipe` familiar from UNIX systems. Its function is to pass the output of one program to another program as input. You can chain several programs together using multiple pipes.

The advantages of using pipes are:

1. No need for a lot of RAM to keep the temporary files, or, no need to slow things down by storing temporary files on the HD.
2. Faster operation since the intermediate data doesn't need to be stored anywhere between programs.

The '`PIPE:`' device differs from the UNIX pipes in two important respects:

1. It is a device, so the input and output of the programs doesn't need to be a redirected standard output, although that is the general case. You can also use 'PIPE:' instead of files, with two restrictions; you can not read a directory from 'PIPE:', and the 'PIPE:' 'file' can not be 'seeked' in.

2. It does not support a flush operation. If all of the data written in a 'PIPE:' device is not read out, it will stay there, buffered. Thus you must always 'empty' a pipe before closing it.

3. Because of the above, if unread data exceeds the size of the 'PIPE:' internal buffer, the program writing to the pipe will 'block' (that is, suspend operation) until the data is read. See below on how to manually flush a pipe.

The PIPE: handler name syntax is 'PIPE:name/bufsize/bufnum', where 'name' is the name for the pipe channel. Using names you can have several simultaneous pipe operations. The optional arguments bufsize and bufnum specify the size and number of the buffers used by 'PIPE:'. Normally, you would just use 'PIPE:name'.

The default buffer size of 'PIPE:' is 4096 bytes, and the number of buffers (ie. channels) is unlimited.

Osma Ahvenlampi (Osma.Ahvenlampi@hut.fi)

2.3.1 Using PIPE: in a standard AmigaShell environment

First, make sure 'PIPE:' is mounted. You can do this by giving the command

```
1> Mount PIPE:
```

in a Shell window. On an AmigaOS 2.1+ system, 'PIPE:' is normally mounted on startup by placing the mount entry file 'PIPE' in the drawer 'DEVS:DosDrivers'.

In an AmigaShell window, give the commands:

```
1> Run List SYS: >PIPE:Listoutput
1> More <PIPE:Listoutput
```

[1]

These two commands list the contents of the 'SYS:' volume to a pipe and then type it in the Shell window. You could also use:

```
1> Run List SYS: NOHEAD >PIPE:Listoutput
1> Run Sort PIPE:Listoutput PIPE:Sortedoutput
1> More <PIPE:Sortedoutput
```

---

[1] Note that '1>' is the prompt AmigaShell gives when it is waiting for a command. The number varies, and the prompt can be customised to look completely different. It is not a part of the command.

Notice that you need to 'Run' all the programs except for the last one. This is so that you can get them all running simultaneously. You could also use several Shell windows, or any other way of running all the programs at the same time. Also this might not work if 'More' isn't the standard Commodore AmigaOS 2.04+ More found in 'SYS:Utilities'.

In case you need to manually flush a pipe, for example because the program that was reading from it aborts prematurely, it can be done with the command:

```
1> Type PIPE:name TO NIL:
```

2.3.2 The Pipe command

As you can see, this is a somewhat difficult way of using pipes, needing a lot of typing especially compared to the UNIX pipes. There is a solution for that.

Andy Finkel, a former Commodore software engineer wrote a command called `Pipe` (not to be confused with 'PIPE:') that simplifies the use of the 'PIPE:' device. Unfortunately, this command did not make it into the 2.04 release. However, Commodore has given the permission to distribute this, and a few other commands, freely. (Sources: Fish disk 637, Aminet, 'util/cli/finkelshelltools.lha')

For some reason Commodore did not include these commands in the later OS releases, although they work fine even with OS 3.1 (V40). Thus all users are required to get them from the above sources if they wish to use them.

The Pipe command usage is simple. The commands to be run with their inputs and outputs piped together are given as arguments to the command, separated by the | (vertical bar) character:

```
1> Pipe List SYS: | More
     or
1> Pipe List: SYS: NOHEAD | Sort IN: OUT: | More
```

Notice how Sort needs 'IN:' and 'OUT:'. These devices don't actually exist, but the Pipe command simulates them with the PIPE:. This is necessary because C:Sort can not use the 'standard IO', but insists on files.

The separator character | can be configured to be something else with the local variable _pchar.

2.3.3 Pipe command support in AmigaShell

There is a little known and quite useful undocumented feature built in the 2.04+ AmigaShell, that is, AmigaShell knows about the `Pipe` command. By setting the local variable _pchar, you can use pipes without even entering the Pipe command. AmigaShell will prepend it on the command line automatically if it sees the pipe separator command specified by _pchar on the command line.

Thus, after the command:

```
1> Set _pchar "|"
```

the above examples can be used like this:

```
1> List SYS: | More
      or
1> List SYS: NOHEAD | Sort IN: OUT: | More
```

Those familiar with the UNIX pipes will notice how the usage is very similar to the UNIX pipe usage.

2.3.4 Quick usage guide

1. Get the ShellTools archive from the Fish disk 673, or from Aminet, '`util/cli/hacks204.lha`'.
2. Install the commands in this archive in your Shell search path, preferably C:.
3. Put the command '`Set _pchar |`' in your '`S:Shell-Startup`' file.
4. Restart your AmigaShell so that it reads the '`S:Shell-Startup`'.
5. Try '`List SYS: | More`'
6. Read the ShellTools documentation and experiment.

2.3.5 Related things

The variable _mchar is used to separate one command from another in a shell line. So, after the command:

```
1> Set _mchar ";"
```

you can put more shell commands into the same line, separated from each other by ;.

2.3.6 Troubleshooting

Q: When I try the command '`List SYS: | More`' I get an error

                  `PIPE: Unknown command`

          I have '`PIPE:`' mounted, what's wrong?

---

2 The quotes around the vertical bar are important, to avoid AmigaShell interpreting the command as an attempt to use PIPE:, if _pchar is already set (for example if the Shell is started from within another one).

A: AmigaShell is referring to the command `Pipe` in the error, not the '`PIPE:`' device. You haven't installed the commands in the ShellTools archive.

Q: I get a requester that says

> `Please insert volume PIPE: in any drive`

when I try any of the command examples.

A: '`PIPE:`' is not mounted. Try '`Mount PIPE:`'. If that doesn't work, refer to your AmigaDOS manual or a friend that knows how to mount devices.

Q: When I try any of the examples the More window pops up but nothing is shown/I get an error afterwards/More pops up a file requester.

A: You are not using the standard AmigaOS `More` command. Either use another name or install a command that supports pipes. Possibilities are the C= More, Less, or Most for example.

Q: None of these happen but it still doesn't work as you described.

A: Are you using OS 2.04 later? Are you using the standard AmigaShell? Try with all extras disabled.

3 Programming

This chapter handles problems arising for programmers and compiler-users only.

3.1 What documentation do I need as an Amiga programmer?

The best information available are the RKMs (ROM Kernel Manuals), 3rd edition, by Commodore, published by Addison-Wesley:

```
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
```

Especially the RKM: Libraries is a must. The RKM: Includes and Autodocs isn't that much worth: Better get the same stuff on disk instead to have it online. See Section 3.3 [Includes], page 9.

AmigaDOS isn't included in these books. The Autodocs give much information, but to go deeper you probably need

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

published by Bantam Books.

A better choice for developers is

    The Amiga Guru Book

by Ralph Babel. The book starts with a survey on different aspects of programming the Amiga. (About 250 pages) Useful for beginners are the sections on the Amiga's data types (not to be confused with the 3.x DataTypes used by MultiView, for example), the Includes and the amiga.lib. But even experienced programmers will find useful things here that are missing in the RKMs. But the largest part are about 500 pages on AmigaDOS and, as I think, the most important, because AmigaDOS is the worst officially documented part of the OS. The book is rather concise and hence not as easy to read as the RKMs, but I recommend it as an addition and instead of the AmigaDOS manual. (Not, however, a replacement for the Libraries and Devices, which aren't covered here.) Unfortunately the book has no ISBN and is available in special stores only. But there are some mail order companies which offer it for about 50$ and which accept credit cards:

    Hirsch & Wolf OHG; Mittelstrasse 33; D-56564 Neuwied; Germany
    Voice: +49 (2631) 8399-0; Fax: +49 (2631) 8399-31
    E-Mail: <hhhirsch@carla.adsp.sub.org> (fax preferred)
    (Eurocard/Mastercard/VISA)

    Periscope; Attn: Cody Lee; 1717 W Kirby Ave; Champaign, IL 61821, USA
    Voice: +1 (217) 398 4237; Fax: +1 (217) 398 4238
    E-Mail: <periscope@cei.com>

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

See Section 8.10 [FAQs], page 42.

3.2 What is CATS?

This is a department at Commodore West Chester which was formerly named `Commodore Amiga Technical Support` and was later renamed `Commodore Application and Technical Support`. These are people that work independently of Engineering, but close together with them, and try to help developers outside of Commodore to create nice Amiga applications, software or hardware. To achieve this, CATS has gathered a lot of informations and tools, on floppy, CD, or paper. Much of this material is also available to the general public.[3] But don't mix this up with some sort of Hotline for everyone!

For Americans the address to get this material is

----

[3] Which means: For Non-Developers.

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

for all Europeans it's a company in Germany:

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

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

3.3 Where do I get the Amiga includes?

The only legal way to get the includes and autodocs (and you *should* get them, they are *very* useful!) is to become a developer (see Section 3.4 [Developer], page 9) or to buy the `NDU` (Native developers update kit, also known as `NDUK` or `NDK`), which is offered by CATS. They cost about 30$ plus shipping and this seems to me to be a fair price. See Section 3.2 [CATS], page 8. The current version is 3.1.

If you need only the includes, you could get them as well with a compiler (commercial compilers only) or by getting the Fresh Fish CD. See Section 8.6.2 [Fish CD], page 40.

3.4 How do I become a developer?

You need the `ADSP` (Amiga Developer Support Program) documents. To get this write a letter to your local Commodore branch asking for these documents. Everything else should be explained there. The german address (for the US address see Section 3.2 [CATS], page 8) is

```
        Commodore
        Lyoner Strasse 38
        60528 Frankfurt
```

There are three different developer versions:

Registered
          developers get access to the CBMNET (kind of a Commodore-internal Usenet), which makes it possible to discuss problems with other developers, Commodore engineers included. Registered developers pay about US$80 per year, plus initial US$30 initially.

Certified developers seem to me the most interesting claass: They have the possibility copy the most beta versions of the system software (Kickstart and Workbench) and the respective Includes and AutoDocs. (Not all beta versions and especially no beta hardware.) You pay about US$250 per year plus US$50 initially for this.

Commercial

> developers finally have in most details the same as certified developers, but may expect
> to get more beta versions and sooner, included beta hardware (This has **not** always
> been so in the past.), hencethey pay about US$400 plus US$50 initially.

The above describes the prices and the situation in Germany and might be different elsewhere.
Especially not all Commodore branches offer the registered status. A hint is to build a group of
users and become a developer group, so the costs are reduced.

All developers have to sign non-disclosure agreements (`NDA`) which specify that we may not dis-
cuss information outside of authorized areas even to other developers until given explicit permission
by Commodore.

3.5 What compilers (assemblers) are there?

There is a lot of programming languages on the Amiga, commercial as well as freely distributable.
I will enumerate only those that I know or which seem it worth to me otherwise.

Assembler All C-compilers have an Assembler included. Freely distributable are A68K and PhxAss
(directory '`dev/asm`' on Aminet or Fish disks 521 and 906)

C

C++ Freely distributable C-compilers are `gcc` (which has its own directory '`dev/gcc`' on
Aminet) and the evaluation version of `Dice` (for example per FTP from `ftp.uni-
paderborn.de`, directory '`/news/comp.binaries.amiga/volume91/languages`' or on
Fish disk 491). The advantage of gcc is that you find gcc versions all over the world
and on all computer systems. Another advantage is that C++ is included into gcc! But
it is slow and needs 4Mb of RAM or more. See Section 3.13 [The GNU C compiler],
page 20. See Section 8.9 [Mailing lists], page 41.

Commercial C compilers are `Aztec-C`, `Dice` and `SAS/C`. `Aztec-C` doesn't seem to get
further development any more. It should be remarked that the commercial compilers
have especially wonderful debugging utilities (Source level debuggers!) that the others
are missing.

SAS/C has announced to drop support of the Amiga-Compiler, too, due to the Amiga's
bad situation. However, the compiler is still available and up-to-date and has a cross-
compiler included, which translates C++ to C and supports the source level debugger
too. Considering the price of just 99$ for students and updates from recent versions or
other compilers it is still the most recommendable of the three. In europe the compiler
is available from

```
SAS Institute GmbH
PO Box 10 53 40
69043 Heidelberg
Germany

Phone: (49)6221-4160
EMail: eurdoc2@vm.sas.com
```

Dice is the cheapest and fastest commercial compiler. The greatest disadvantage of Dice (compared to the other commercial compilers) is the so-called Source-Line-Debugger: This means that you see the current line of source and can execute the program step by step, like with the other debuggers. On the other hand you can examine memory only, not the variables. To get information about DICE, send email to info@oic.COM. An automatic system will return complete details, including upgrade prices.

Comeau C++ is a crosscompiler like SAS/C++. That wouldn't be a problem, but Comeau C++ doesn't have a C compiler included. You need SAS/C, Aztec-C or Dice additionally. But it is AT&T cfront 3.0 compliant and supports exceptions. And like gcc it runs on many platforms. Maxxon C++ is offered in Germany. I cannot say anything on it. Both compilers are commercial. Comeau's address is:

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

EMail: Greg Comeau, comeau@bix.com
```

Forth   JForth is said to be an excellent Amiga port of Forth. Among its advantages are object oriented extensions, full Amiga interface and an application generator. It is available from:

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

Phone: (415) 453-4320
EMail: Phil Burk, phil@ntg.com
       Mike Haas, haas@starnine.com
```

Fortran   (Sigh! Still people who need it :-<) Freely distributable are BCF (Fish disk 470) and f2c, a Fortran to C converter (Aminet, directory '`/dev/misc`'). A commercial compiler is offered from ABSoft. All these are Fortran 77 compilers, I don't know any Fortran 90 compiler on the Amiga.

Lisp   Freely distributable Lisp interpreters are XLisp (Fish disk 181) and OakLisp (Fish disks 519 and 520) and CLISP ('`/pub/lisp/clisp/binaries/amiga`' at the server '`ftp ma2s2.mathematik.uni-karlsruhe.de`'). Lisp compilers are Gambit (Fish 764 and 765) and Scheme-to-C (Fish disks 556, 557 and 558). See Section 8.9 [Mailing lists], page 41.

Prolog   '/dev/lang/UNSWProlog.lha' and 'dev/lang/sbp3_1e.lha' on Aminet as well as
                 'SBProlog' on Fish disk 141 and 'SBProlog' on Fish disk 145 are freely distributable
                 Prolog interpreters.

Modula-2   M2Amiga is offered in Europe, Benchmark Modula-2 in the U.S. Both are said to be
                 very good, have a powerful source-level-debugger, a large library. Especially M2Amiga
                 has great support by a german user-group (AMOK) which for example offers own PD
                 disks. See Section 8.9 [Mailing lists], page 41.

                 M2Amiga is offered by:

                         A+L AG
                         Daderiz 61
                         2540 Grenchen
                         Schweiz

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

                 Benchmark Modula-2 is available from:

                         Armadillo Computing
                         5225 Marymount Drive
                         Austin, Texas 78723
                         USA

                         Phone/FAX: 512/926-0360.
                         EMail: Jim Olinger, jolinger@bix.com

Oberon

Oberon-2   Two compilers (both Oberon-2) are available: AmigaOberon (commercial) is offered by
                 the same company as M2Amiga. It is integrated into a full developers environment and
                 has a large library of modules. Library linker and source level debugger are available.

                 Oberon-A is a Freeware compiler. (Source: Aminet, 'dev/obero' directory) However,
                 this is a beta release, especially the module library is limited. See Section 8.9 [Mailing
                 lists], page 41.

                 The AMOK user group supports AmiOberon as well as M2Amiga.

Pascal   There is a PD-compiler called PCQ (Directory 'dev/lang' on Aminet or Fish disk 511).
                 It doesn't support all of Pascal and major features are missing. P2C, a pascal to C
                 converter is on disk 341. (Aminet: '/dev/misc/p2c120.lha') Additionally there are
                 two commercial compilers called HiSoft Pascal and KickPascal. HiSoft Pascal and P2C
                 claim to be compatible to Turbo Pascal up to 5.0. HiSoft has a source level debugger
                 included.

3.6 Those never working Esc sequences!

Many printers come with a manual that explains which `ESC` sequence causes which action on the printer. But there happen weird things when you try to send these sequences to your printer, either it does nothing, or it does something completely different. There is a reason, the Amiga printer drivers. These drivers are made in a way that they only understand a certain set of `ANSI Esc sequences`, not the special ones defined (differently) by the various printer manufacturers. The purpose is that every application on the Amiga just uses this one standard set of control sequences and this way doesn't need to know which printer is actually connected. The printer driver then translates these standard sequences into the special sequences a certain printer understands. A list of the available ANSI Esc sequences is found in the current Workbench manuals (or older AmigaDOS manuals). Now if you want to issue a control sequence to the printer that's not available as an ANSI command, you have two possibilities to achieve this:

1. Bypass the printer driver (that would unsuccessfully try to interpret the sequence) and send your output **only** during this sequence to '`PAR:`' (or '`SER:`', respectively). For this you have to close and open printer output channels very often which is rather tedious, and you have to know where ('`PAR:`' or '`SER:`') your printer is connected.

2. Use a special ANSI sequence, made exactly for this case:
   > '`Esc[<n>"<x>`'

   where '`<n>`' is the decimally typed number of bytes in the string '`<x>`', which actually contains your special printer sequence. This ANSI sequence tells the printer driver to not interpret or translate the next '`<n>`' bytes.

But both methods have one big disadvantage when used in an application program: You lose the printer independency! If you stick to ANSI sequen- ces, you can output to any printer on earth, as long as there is an Amiga printer driver for it. If you start to use special control sequences, your program will be tied to this single printer model and will not be usefull for any other (or you would have to provide some dozen new printer drivers for your application).

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

3.7 Is it possible to use AmigaBasic on the A1200?

We heard conflicting reports about AmigaBasic on the A1200: While I said that you can work sufficiently with it, others said that this isn't possible, as AmigaBasic crashes on the slightest little error. I couldn't reproduce this.

Now I can. It depends on the setting in the Sound Prefs editor. When you activate a sound there, this conflicts with sound that AmigaBasic tries to produce by hand and obviously not quite the correct way itself.

Easy solution: To work with AmigaBasic on the A1200, just

1. Switch off sound output in the Sound prefs editor.

2. On the A4000 (as well as on an A1200 with Fast Mem expansion[4]) you additionally need to run NoFastMem.

3. Better avoid SUBs and use conventional GOSUBs instead, then the compatibility with newer processors will be higher.

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

3.8 How do I localize my program?

Suggest, you want to write a `HelloLocalWorld.c`. Your final program will look like this:

```
#include "HelloLocalWorld_Cat.h"
#include <clib/exec_protos.h>

struct Library *LocaleBase;

void main(int argc, char *argv[])

{
    /* Open the locale.library. No kill, if not successfull.
       (Just use the builtin catalog strings instead.) Note, that
       we open locale.library here, even if our compiler supports
       AutoOpening.
    */
    LocaleBase = OpenLibrary("locale.library", 38);
    OpenHelloLocalWorldCatalogs(NULL, NULL);

    printf(GetString(MSG_Hello));

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

The routine GetString checks, if the wished catalogs are available and returns a pointer to either the builtin string or the catalog string. (In my case the german string.)

You see, the main difference besides the minor opening and closing stuff (OpenLibrary, Open-HelloLocalWorldCatalogs, ...) is to replace strings with a function call. Hence we need a file 'HelloLocalWorld_Cat.c', which holds OpenHelloLocalWorld, GetString, CloseHelloLocalWorld and the builtin strings (this could be an array, where

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

---

[4] If you have a turbo board

is defined) and an include file 'HelloLocalWorld_Cat.h', which defines the message ID's like MSG_Hello. You don't need to know, how these files work internally, especially you don't need to know locale.library!

There are some catalog generators (in what follows: CGs) available ('CatComp', for devlopers only, KitCat, german docs only, 'MakeCat', which I don't know and FlexCat, which I recommend, because it is most flexible in the generated source and supports catalogs on 2.0 and any language, even Amiga-E, Cluster, Pascal, ... and besides that: I'm the author ;-) are tools, that create HelloLocalWorld_Cat.h, HelloLocalWorld_Cat.c and the real catalogs for you. (The above code might differ slightly between the different CGs.) (See Aminet, directory 'dev/misc'.)

Of course they need to know how to use them. First create a so-called catalog-description file. This could look like this:

```
; Lines beginning with a semicolon are comment lines.
# language english
; the language of the builtin strings
# version 0
; the catalog version (0 = any)
MSG_Hello (1/15/30)
Hello, local world
```

Any string is defined by a line like the last two lines above: MSG_Hello is the message-ID, (1/15/30) says, that the value of MSG_Hello should be 1 (you may omit this, in which case just the next free number is used) and the string must not be shorter than 15 characters or longer than 30 characters. (These may be omitted too.)

Now write your program. Once you are ready, use the CGs to create a so-called catalog translation file. (One for any language different than the builtin.) In my case (german) this could look like this:

```
; Lines beginning with a semicolon are comment lines.
## language deutsch
; the catalog language (german)
## version $VER: Deutsch.catalog 1.0 (22.12.93)
; the catalog files version string
MSG_Hello

; Hello, local world
```

Note the empty line after the message ID. (The arguments of ## language and ## version would be missing as well.) You have to fill in the german strings here. Again using the CGs you create a catalog file from this. Additionally note, that no informations on the strings ID or length are behind MSG_Hello. They are taken from the catalog description file.

Once you change the program (adding strings, changing the string length) you change the catalog description as well, use the CGs in the same way to update the catalog translation and hence the catalogs.

3.9 How to obtain a pointer to a console's window

The following function returns the window pointer of a CON window. It can be executed safely under all versions of the Amiga's OS.

```
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;
}
```

Notes:

Accessing a console's window directly may interfere with operations performed by the CON handler. Be careful!

To obtain the window pointer of a CLI's console, pass the FileHandle returned by Open("*", MODE_OLDFILE) to the above function.

The result of the above function may well be NULL, e.g. in case of an AUX handler or if an AUTO CON handler is unable to open its window.

Sending an ACTION_DISK_INFO packet to an AUTO CON handler (2.0+) causes its window to lose its special AUTO properties (i.e. it can no longer be closed at any time by clicking on its Close gadget), as the window pointer returned in id_VolumeNode must remain valid from now on.

All in all: Don't use this function. :-)

For more information, please refer to pages 273, 276, 435, 463, 485, and 629 in "The Amiga Guru Book" (see Section 3.1 [Manuals], page 7).

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

3.10 What are pragmas?

Pragmas are special compiler commmands which control certain features of a C-compiler. Two problems arise when using pragmas:

1. Pragmas are compiler specific. You cannot expect that one compiler will understand pragmas of another compiler, even if both run on the Amiga.

2. You are not guaranteed that a compiler ignores pragmas, that he doesn't understand. Even more: It might not help, to use something like this:

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

A workaround is to put pragmas in a special include file (BTW, the same holds true for statements like #asm (Aztec-C) or #extern (C++) and replace the above with

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

But what do pragmas on the Amiga? The most common usage (not the only, but most meant when talking about pragmas) is to tell the compiler about how to call library functions: Simple C functions expect their arguments on the stack, but library functions want their arguments in

special processor registers and additionally the `library base` in register a6. Lets have a look at a pragma command of the Aztec-Compiler.

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

This tells the compiler to put the first argument of FreeMem in register a1, the second in register d0 and the value of the variable SysBase in register a6. Maxons pragmas look the same, Dice pragmas and SAS pragmas look a bit more complicated;

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

Here d2 is (like 0xd2 above) the `library vector offset` (see below), the digits 09 are codes for the argument register in reversed order (Register codes are 0=d0, 1=d1, .., 8=a0, 9=a1, a=a2, ..), the following 0 is the result's registerä(always d0))and the final digit 2 is the number of arguments.

A command '`FreeMem(fib,sizeof(*fib);`' could produce the following code, if the compiler has seen a pragma statement like above:

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

Calling FreeMem in that way is shorter and faster than pushing the arguments on the stack, calling a function _FreeMem which would do just the same like the above code by pulling the arguments from the stack.

The best way to use pragmas is to include statements like the following in your program:

```
/* Get the prototype for the function; note, that this is */
/* compiler independent.  */
#include <clib/exec_protos.h>

/* Get the pragma; compiler dependent, but most pragmas */
/* are in files with the same name.
#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.h>
#endif
```

The above example can be compiled on all these compilers and produce the best code.

A final question arises: How to get the pragmas? Most compilers have them included. However, sometimes you want to produce pragmas for yourself, for example if you are using new libraries or new versions with additional functions. In that case you can produce them from the so-called `FD`

files which should be a part of the developer docs of the library. (The NDU has a directory FD which contains FD files for all libraries and devices of the OS. see Section 3.3 [Includes], page 9) Most compilers have a utility with the name 'fd2pragma' or similar included which can do that for you. A freely distributable version which can produce pragmas for Aztec, Dice, SAS and Maxon as well as LVO files for assembler and stub routines for the tag versions is available on Aminet ('dev/misc/fd2pragma2_0.lha' and on the Fish CDs.

For pragmas under gcc see Section 3.13.5 [Inline Headers], page 22.

3.11 My Compiler/Linker is complaining about missing symbol xxx.

First be sure, that the function is really missing: For example floating point functions are in a special link library and you need a linker option like '-lm' to include it into your program. Another possibility would be that you are using a library function and didn't notice it. This might lead to a missing library base, 'IntuitionBase' for example. In that case just put something like

```
struct Library *IntuitionBase;
```

somewhere in the global part of your program. (Don't forget to call OpenLibrary() and CloseLibrary! :-)

However, you could as well use a function which really isn't present in your library at all. If you have, for example, an amiga.lib from 2.0 you would hardly find the locale functions or the pool memory functions.[5] Best solution is to get the NDU (see Section 3.3 [Includes], page 9), but you probably don't want to wait for it. In that case you have to find what kind of function you are missing.

Simple library functions (Examples: 'exec/AllocPooled', 'locale/OpenCatalogA') can be called with pragmas. However, you need informations on the name of the library base and where to put the arguments. See Section 3.10 [Pragmas], page 17.

Tag functions are mostly just stub functions which call library functions. If you have, for example, 'dos/AllocDosObject' which expects a constant and a pointer to an array of tags, you have the varargs version 'AllocDosObjectTags' which expects tags on the stack as well! Just create the following function:

```
#include <clib/dos_protos.h>
#include <pragmas/dos_pragmas.h> /*  Probably wrong name */

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

---

[5] This problem arises most frequently for owners of Aztec which is no longer supported and owners of Dice, which has sometimes rather incomplete libraries. I own both ...

Some functions still remain: Amiga.lib has some functions which are really doing valuable things and not just call a library: The BOOPSI functions (‘`DoMethod`’, ‘`DoSuperMethod`’) the memory pool functions (‘`LibAllocPooled`’, ‘`LibCreatePool`’, which are replacements of 3.0 functions). The only way to replace these is to get equivalents. The AmigaFAQ archive contains some of them (DoMethod, DoSuperMethod and HookEntry) in the ‘`programmer`’ directory as well as the most common pragma files and some examples of varargs functions. See [Amiga-FAQ Archive], page 42.

3.12 Where do I find the function xxx?

If you are not sure about the name of the appropriate function for a certain job or in which library to find it, you can search in these places:

In the Autodocs for each library you find at the very beginning a contents table listing all functions available in this library. They are sorted alphabetically, just like in the rest of the Autodocs. In the body part of the Autodocs you find complete explanations and specifications for each library function. See Section 3.3 [Includes], page 9.

The .FD files also offer a very compact overview over all functions of the diverse libraries, complete with a short information about the function arguments. If you already know roughly what you search for and e.g. only need the precise order of the arguments, you can find all needed information here. See Section 3.10 [Pragmas], page 17.

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

3.13 The GNU C compiler: general information and installation

This chapter contains general and installation information about the amiga-port of the GNU C compiler gcc.

3.13.1 Current Version

The current version of gcc, further refered to as <gcc-current>, is 2.5.8. Work is in progress for version 2.6 which will incldude a new version of the popular ixemul.library, the Unix system-calls emulation shared library, and a brand new C library avoiding gcc compiled programs to use ixemul.library.

3.13.2 Requirements

Any Amiga (ranging from A1000 upto A4000/40) will run amigados-gnu utilities. A minimum of 4MB memory is needed in order to compile small/medium projects. More memory will be needed

for large projects, such as recompiling gcc itself, or programming C++. Gigamem is known to work with GCC so *maybe* less memory will work. But in this case, an MMU equipped Amiga (A3000,A4000/40) is necessary. See Section 1.1 [68EC0xx], page 1.

A full installation, including C++/ObjC related files, inline headers, cbm-headers, (see Section 3.3 [Includes], page 9) roughly requires about 20 MB of HD space.

Gcc will work under KS 1.2/1.3, but the full functionality is only available under KS 2.x+. A fast CPU (eg. 68030@25MHz or better) is also of help.

3.13.3 Authors

Gcc and related GNU-software has been ported to the Amiga by the following folks: See Section 3.13.8 [How to get help], page 24.

```
Gcc v2.2.2 port:   Markus Wild
Gcc v2.3.3 port:   Markus Wild
Gcc v2.4.5 port:   Philippe Brand, Lars Hecking, Fred Fish
Gcc v2.5.0 and up: Philippe Brand, Fred Fish, Leonard Norrgard

Ixemul.library:    Markus Wild, Leonard Norrgard, R. Luebbert
Libnix:     Matthias Fleischer, Gunther Nikl
        Gerlib:            Gerhard Mueller
```

3.13.4 Sources for Gcc

All GCC sources & binaries are available on:

1. Aminet sites (wuarchive.wustl.edu and mirrors such as ftp.luth.se) in /pub/aminet/dev/gcc
2. Ramses The Amiga Flying BBS:
   ```
       +33-1-60037015  HST Dual v32 terbo 4800-21600
           +33-1-60037713  SupraFax v32bis    4800-14400
           +33-1-60037716  Tornado v22bis     1200-2400
   ```
   in Topic 'Development', Area 'Gcc' (are 156).

GNU source code is available on:

1. the same FTP site you've taken the binary distribution from
2. gnu.prep.ai.mit.edu (18.71.0.38) in '/pub/gnu'
3. Ramses The Amiga Flying BBS in Topic 'AmigaUnix/Unix/Linux/NetBSD', Area 'Gnu Source Code'

These archives should contain everything necessary to get you going, they don't include sources for ixemul.library, which are available on Aminet sites in '/pub/aminet/dev/gcc/ixemsrc3947.tar.gz'

(as of this writing, work is in progress for the v40 version of ixemul.library, sources will be available on an Aminet site near you).

As stated by Richard Stallman of the FSF:

"The GPL says that any distribution of binaries must contain either the source code or a written offer to supply source code (see the GPL for details of what is required)."

3.13.5 Inline Headers

The inline headers for gcc can be created from the original CBM fd files. See Section 3.3 [Includes], page 9. See Section 3.10 [Pragmas], page 17. They can be created as follows:

```
CLI> Assign INCLUDE: GCC:os-include
CLI> Assign FD: INCLUDE:fd
CLI> Makedir INCLUDE:inline
CLI> cd USR:bin/geninline
CLI> gen31
```

This will create all inline-headers in 'GCC:os-include/inline'. If you have 2.0 headers, use gen20 instead, if you have 3.0, use gen30. OS3.1 (rev 40.13) inline headers are included with <gcc-current>. See Section 3.13.1 [Current Version], page 20.

NOTE: perl scripts do not handle correctly AmigaDOS include files, which seems to mean they are somewhat broken. This needs some voluntary work ...

There is also another way of generating inline headers, using fd2inline program:

```
CLI> fd2inline <fd_file> <proto_file>
```

3.13.6 Amiga Libraries

Amiga-style libraries, such as amiga.lib, cannot be read by gcc directly. They need to be converted into a gcc-readable format using hunk2gcc, the AmigaDOS object converter made by Markus Wild.

To achieve this, simply grab a copy of latest amiga.lib (from Commodore Development Kit, see Section 3.3 [Includes], page 9) and make a new directory where you want your converted object files to go, cd into it, and enter

```
hunk2gcc amiga.lib [..further libs if you like..]
```

This generates an a.out object file for every program unit present in the hunk file (in this case, from amiga.lib).

As the final step convert all those files into an a.out style library by issuing:

```
    ar qc libamiga.a obj.*
    ranlib libamiga.a
```

The ranlib run builds a symbol table in the archive, and makes accesses to the library much faster.

NOTE: As long as you make no AmigaDOs specific calls, you can create a dummy library using:

```
    cat "int dummy;" >dummy.c
    gcc -c dummy.c
    ar crv libamiga.a dummy.o
    mv libamiga.a gcc:lib
```

3.13.7 Installation

IMPORTANT NOTE FROM Philippe Brand:

As this is FAQ, most often asked question about gcc-2.5.8 is: "I can't compile C++ programs, what's up ?"

In fact, I've messed up gcc-2.5.8 distribution. Instead of 2.5.8 versions of C++ and Objective-C compiler passes, 2.5.7 versions were included. You've got here 2 options.

- rename lib/gcc-lib/amigados/2.5.7 to 2.5.8 - get gcc258ud.lha on Aminet, which contains proper 2.5.8 versions.

From Aminet get also gcc258ud1.lha which contains some fixed libraries, headers etc...

And DON'T forget to grab gcc258-readme.lha also from Aminet.

1. If this is your first installation of GCC:
   ```
       CLI> cd Disk_with_lot_space:
       CLI> lha x gcc258-x.lha
   ```
   (where x stands for 1 through 4, 5 & 6 if you want also to have sources)

   Now from the Workbench, invoke GCC-Install. This is what people wanted, e.g. an Installer based configuration script for GCC.

   Thus you only have to unpack archives and the installer script will do the rest, creating environment variables, adding assigns to user-startup, copy library and handler.

   BTW:

   Installer and Installer project icon (c) Copyright 1991-93 Commodore-Amiga, Inc. All Rights Reserved. Reproduced and distributed under license from Commodore.

2. If you "upgrade" your gcc environment from v2.x.x, well just erase gcc directory then go back to step 1. Too many stuff have changed. Just make sure you've backuped all what you've added to the GCC tree before.

Note that you MUST use ixemul.library revision 45 and NOT revision 47, as the latter doesn't check for ENV:

You need to have a 50.000 stack size in order to compile with GCC. This should be enough for most projects. Note than while recompiling gcc with itself it has takeen more than 245KB stack. Stack can grow due to source complexity. Don't be afraid of it.

To use ar and/or ranlib, you should have a much larger stack, between 200.000 and 300.000, depending on library size.

3.13.8 How to get help

The current amiga-gcc-port maintainer is available:

```
Philippe BRAND
Fidonet: Ramses The Amiga Flying BBS 2:320/104.21
Email:   phb@colombo.telesys-innov.fr (ONLY for personnal email).
Ftp:     colombo.telesys-innov.fr:/pub/amigados-gnu
         or /pub/incoming/uploads for uploads.
```

There's also an amiga-gcc mailing list running in Finland. See Section 8.9 [Mailing lists], page 41. Philippe Brand will forward all questions to that list, but you'd be better off using it directly ;)

4 Applications

This chapter offers informations about major Applications.

4.1 Text Editors

Text Editors are programs allowing to enter and edit unformatted text. Generally, this means text that is meant to be manipulated by machine, rather than human. Programmers use these to enter the text for compilers. Since UNIX machines don't typically have word processors, most text processing starts with a text editor, then is filtered through a page layout system (Section 4.4 [LaTeX], page 26, for example) to produce attractive paper results.

Commercial Products

CygnusEd Professional and TurboText seem to be the main contenders in the professional realm. The Fred Fish disks contain dozens of other shareware text editors. A demo version of TurboText is on Fish disk 445. A very old demo of CygnusEd is on Fish disk 95 (testament to its lasting-power). In the following some freely distributable editors will be discussed.

Emacs   Gnu Emacs (the "G" is not silent) comes from Unix and is probably the king of editors
– it's huge (about 1 Megabyte), feature-packed (it does windows and even contains a
game!) and extensible (if you know lisp you can write new emacs functions and bind
them to any key combination). On the other hand, it may be too huge, its feature-
ladenness is imposing, and its extensibility often means you can't use someone else's
emacs configuration. Source: Aminet (directory '`util/gnu`').

Vi   The leaner, less configurable, non-extensible cousin to gnu emacs is vi (pronounced
"vee eye"). Unix people like vi especially because you find it on *any* Unix machine.
What you choose is personal preference, and will mark you for life. Vim is a good vi
for the amiga, and is on Fish disk 591 or in the '`util/gnu`' directory of Aminet.

DME   Many Amiga programmers like DME. It's fast, fully configurable; menus may be created
and any key may be mapped. It's much easier to learn DME than than Emacs or Vi.
There are three different versions: AmokEd, DME and XDME. It's a matter of opinion
which you prefer. (Oberon and Modula programmers like AmokEd because it's written
in Oberon and supports AmigaOberon error messages, C programmers like DME or
XDME.) Sources: Aminet (directory '`util/edit`'), Fish disk 776 (XDME) and 749
(AmokEd), AMOK 90.

4.2 What word processors are there?

A word processor is the typical application for writing notes, letters or reports on a computer.
Unless you prepare newsletters on a weekly basis, your word procesor is probably your workhorse
program. Thus choosing one you are comfortable with determines how comfortable you are with
your computer. Word processors can offer a variety of features, and many can approach the
sophistication required for Desktop Publishing (see Section 4.3 [DTP], page 26) but no one uses
them for programming, for which text editors are more suited.

One distinguishes between Wysiwyg programs (What you see is what you get) and page layout
languages. Wysiwyg programs should be fast, comfortable and easy to use. Most people prefer
them. The alternative is an approach that works similar to compilers. You feed text files to a a
program that produces the layout which may be previewed on screen or printed. LaTeX takes this
approach. See Section 4.4 [TeX], page 26. Lout is another such system which seems smaller, easier
to learn and has full documentation included, but it is nonstandard. Lout produces Postscript
output. (I don't know if this is an advantage or disadvantage. ;-) See Section 4.5 [Postscript],
page 27. Both programs are freely distributable.

There are a lot of wysiwyg programs, but only commercial products: FinalWriter, Final Copy
II, Wordworth, Word Perfect, AmiWrite, Beckertext II, Maxon Word and many others. I don't
dare to recommend any. All I can say is: Give yourself time to make a selection.

4.3 Desktop Publishing

These programs offer features lacking in word processors, usually tailored to flexible arrangement of text, but often don't provide all of the text manipulation that a good word processor provides. The best desktop publishing programs strive to provide the features of both, just as the best word processors strive to provide the features of desktop publishing programs. Microsoft Word (Mac, PC) is a good example of a word processing program that offers many page layout feature. Framemaker (UNIX, Macintosh, DOS, etc.) is an example of a desktop publishing system that offers most needed word processing functions. As yet, no Amiga program has bridged the gap, though the main word processors are coming close. (On the other hand, even many sophisticated programs don't support typesetting mathematics, tables, producing bibliographies, indexes, or cross-references. The page layout languages do, and programs like Frame are improving their support of such features.) Unless you need to prepare fancy newsletters or promotional literature, a word processing program is probably enough. See Section 4.2 [Word Processors], page 25.

There are not yet any freely distributable wysiwyg desktop publishing systems. Commercial products are ProPage and PageStream. They have been playing leapfrog for the past few years. It appears that PageStream 3.0 is about to leap ahead. A more detailed description of these products and their differences is welcome. Both programs' list prices are $299. Student discounts are available (approx 40% discount.)

4.4 What is TeX and where can I get it?

TeX is a very powerful wordprocessing system. It can display mathematical formulas or complex tables as well as function graphs, creates indices, contents and many other things. Its greatest advantage is that it is freely distributable (TeX, not the previewers and the printer drivers!) and that you find TeX all over the world on every computer family. Its greatest disadvantage is that it isn't very handy (works similar to a compiler) and it isn't wysiwyg. But many people like it. (BTW: This document is written using TeX. 8-) See Section 4.2 [Word Processors], page 25.

There are two major implementations on the Amiga. The first one, Amiga-TeX, from Thomas Rockicki and Radical Eye software is commercial. It is said to be excellent and his owners seem to be very satisfied. But it costs at least 200$.

I recommend PasTeX, a freely distributable version. People seem to have problems installing PasTeX, especially the font loading and generation (It's a quite complex program.) but I did not here anyone upset once it was installed. (A friend with knowledge of TeX helps immensely.) A few words should be said what you need:

- 5 disks containing the TeX-compiler itself
- 2 disks containing MetaFont
- Nothing more

Many people ask for fonts. They are included in the MetaFont-package and can get compiled by you. All you need to do is setting up your TeX-system in the right way which is described in the documentation. Please note that the PasTeX disks are compressed using the program Zoom. (see Section 7.2 [Endings], page 33) Sources: FTP at `ftp.uni-passau.de`, directory '`/pub/amiga/tex/PasTeX1.3`' and the Fish CDs.

4.5 Are there any Postscript interpreters?

PostScript is a programming language designed to be used to describe printing on pages. Apple helped make PostScript popular by selling printers with built in PostScript interpreters. Many programs have evolved to produce PostScript programs as their output, making PostScript the lingua franca of printing. Until recently, in order to print a PostScript file, you had to have a relatively expensive laser printer. The development that changed this was the software PostScript interpreter. These programs allow your computer to interpret PostScript programs, and produce the matrix of dots to send to your normal graphics printer.

One of the benefits of PostScript is that it is resolution independent. What this means is that it can support the highest resolution of your device – and that you can reasonably preview PostScript on a low resolution screen.

There are two free PostScript interpreters for the Amiga. Post and Ghostscript. Post comes as an Amiga shared library along with front ends for previewing to the screen and printing. This structure allows others to write programs that can show PostScript images on screen. In fact, AmigaTeX uses Post's library to support incorporation of PostScript into documents. Ghostscript similarly comes in two programs, but not as a shared library. Ghostscript is the rendering engine, and Ghostview is the front end. Sources: Aminet (directorys '`text/print`' and '`text/dtp`'), Fish disk 669

5 How about Graphics?

Graphics is one of the major strengths of the Amiga. Why don't we have more answers here? :-(

5.1 What are chunky and planar displays?

Simply put, the terms `chunky` and `planar` (short for `bitplanar`) refer to different ways of storing graphics information in a computer's memory. They are rather easy to understand, as far as things go, but incredibly difficult to explain:

Computer images are arranged as a grid of pixels, each of which can be thought of as a number representing the color number of the pixel, sort of like a paint-by-numbers scheme. For example, here's a simplified example image, in four colors:

```
00302132
```

The Amiga stores this image in a `bitplane` mode. That is, it is represented by several planes of bits (binary digits, 1s or 0s). This is a four-color image, so each color number could be represented by two bits. Therefore there are two bitplanes:

```
00100110    Here's bitplane 0
00101011    And here's bitplane 1
--------    Now, let's add them up, binary style:
00302132
```

Which is the final image. If the image was in two dimensions, it would truly be composed of bit planes. However, I'd need three dimensions to show multiple bitplanes overlayed, and therefore for simplicity we're working in one dimension (which is all we need).

Now, there's another way of storing this image. How about if we localize the bit data in little chunks?

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

This is the principle of the `chunky` pixel mode.

Both methods of image storage are perfectly logical, and no one can say that one is better than the other. However, there are certain technical aspects which cause certain advantages and disadvantages.

First, if you've seen colored text scroll on your Amiga, you know there is a bit of "flicker" that arises. Specifically, what happens is that while the text is scrolling, its color temporarily changes to something completely different. What's happening is that the computer's moving several bitplanes of data while the raster (monitor electron gun) is sweeping across the screen. What that means is that, if the raster catches the data while it's being moved, you can end up with some bitplanes being moved and some not. What if we filled bitplane 1 in the example above with 0s? Instantly all the 3s become 1s, and the 2s become 0s! This is what causes "flicker" when certain colors are scrolled. By contrast, if a chunky pixel display is caught while scrolling, all we see is a partially-scrolled image; the colors are preserved (since their units are the small ones).

That's a disadvantage to planar pixels, but what about chunky pixels? Well, recall that a computer organizes information in terms of 8 bit bytes. These groups are static; you cannot decide to all of a sudden organize data in terms of three bytes or something! Therefore, when using chunky pixels, things get complicated if we decide to use a nonconvenient number of bits per pixel. In practice, the 8-bit (256-color) mode, and 24-bit (16 million color) modes are the most common candidates for chunky pixel displays.

Finally, certain effects can be accomplished with the different systems. Bitplanar mode is particularly useful for things like shadows (where an extra bitplane is set with 1s instead of 0s), and chunky mode is great for perspective and "mapping" (since the data for each pixel is localized in a single "chunk"). The latter advantage makes chunky pixel mode really great for games, and is what made Wolfenstein 3-D possible.

We all know that Amigas use the bitplane system for storing images. However, the Macintosh and PC(VGA) both use chunky pixel modes. While we can optimize our RAM usage with "bizarre" modes like 8- and 128-color, they gain the advantages of non-flicker scrolling, and the programming simplicity of just writing a byte where you want the pixel to go.

The difference between the two modes becomes problematic in things like emulation. EMPLANT has a "chunky to planar" routine which it uses to convert a Macintosh display into an Amiga one. "Chunky to planar" routines are also useful for getting chunky-inclined things to run on Amigas (see TMAPDemo, rotdemo). On a side note, there was some confusion as to what EMPLANT used the MMU for with regard to chunky to planar. The MMU itself is incapable of performing the algorithm for the conversion; rather, it is used to detect what portions of the display memory are updated from the Mac side, and therefore the processor is saved from having to perform the chunky to planar conversion for the entire display.

I sincerely hope that helped clear up most of the mystery concerning the terms "Chunky" and "Planar"!

(Joseph Luk, jluc@eis.calstate.edu)

5.2 What is doublebuffering?

Don't be ashamed if you haven't a clue as to what this is. It's rather simple, really. Imagine you wanted to animate something by drawing out each frame. Now imagine you had only one piece of paper. Even if you drew pretty fast, it would be difficult to see what the animation looks like because you'd have to erase the frame you just drew, in order to draw the next one!

What if you had two pieces of paper? Why, then, you could see the difference between two adjacent frames. First you'd draw your first frame on the first piece of paper, then you'd draw the second frame on the other piece. Then you'd erase the picture on the first piece of paper, and draw in the third frame. Then you'd erase the picture on the second piece of paper, and draw in the fourth frame, and so forth.

This is the principle of doublebuffering. The computer shows you the picture it just drew, then turns around and draws the next one. It then swaps the picture in front of you with the one it just drew, and repeats the process. The result is a smooth animation, because you never need to see the computer draw; all you see is each finished product.

(Joseph Luk, jluc@eis.calstate.edu)

5.3 What monitors will work on my Amiga 1200/4000?

Monitors can be classified after the horizontal scan frequency they require. TVs, as well as C='s 1084 monitor, need frequencies around 15 kHz. VGA/SVGA need approx. 30 kHz. Multisync monitors can take many frequencies.

In short: You can use any monitor you want with an A1200. BUT:

– If you use a regular VGA/SVGA monitor, you can only use a few display modes (like DblPAL, DblNTSC and/or Productivity). I.e. (320|640) x (256|512|1024) for DblPAL. This is great for Workbench and all "serious" utilities (DTP etc), but don't expect any games to work... they don't use your preferences, just take over the machine and assume a 15 kHz monitor. Also, you cannot utilize the "Early Startup Control" screen (you know, disable cache, and that stuff), which also requires a 15 kHz monitor. Furthermore, VGA monitors don't have speakers. And the VGA-type modes don't support Genlocks. But for a lot of "serious" work, a VGA monitor is quite adequate.

– You already know what happens with a 15 kHz monitor; the flickering in Interlace mode. A small tip: Try to use NTSC instead of PAL. This increases the refresh rate from 25 Hz to 30 Hz, at the expense of lower vertical resolution (482 lines maximum). The NTSC and PAL modes aren't as bad as many people think. If your monitor has a lot of phosphorous (long afterglow), PAL Laced can be quite OK, and it gives you a resolution of 1448x566 in SuperHiRes. That's the highest resolution currently supported on AGA Amigas, in _any_ display mode.

– A Multisync gives you the best of both worlds. The new 1940 and 1942 monitors from C= are quite OK, although rather cumbersome to use... The h/v size and offset must be set manually each time you switch display mode.[6]

For a VGA/SVGA or Multisync monitor, you'd need a little shiny box which gives you the standard 15-pin "D" connector. It costs around $15.

There are other alternatives... like the "AmiVGA" box ($50 or so), which I think is a cheapo version of the Flicker Fixer. (But it's really a shame to use this on an AGA Amiga.)

Also, you can get a VGA monitor, and hook up your TV to the composite or RF port on the Amiga - one monitor for games, one for serious stuff.

(Per Espen Hagen, per.e.hagen@ffi.no)

---

[6] A patch is available on Aminet for Kickstart 3.0 to do this without manual actions in the file 'os30/util/Monitor30Patch.lha'.

5.4 How do I switch between PAL and NTSC?

PAL and NTSC are two different video standards, the former being European, and the latter being American. PAL has a slightly taller screen (256 lines non-interlaced, non-overscanned) as opposed to NTSC (200 lines), so if you see the bottom portion of a program's screen getting cut off on your American machine, chances are the program was written for PAL, and is running on your shorter NTSC screen. PAL and NTSC differences are somewhat less important to European users; since their machines default to PAL, running an NTSC program is no more than a minor annoyance having the screen only appear in the top portion of the display.

Therefore, for us NTSC folks, switching into PAL mode becomes important to avoid loss of some picture on Euro Demos, etc.

First, the most common misconception about switching between PAL and NTSC is that you need a Multiscan or special monitor for such a purpose. Not so! Just about any monitor can handle the minor signal difference between PAL and NTSC (50Hz vertical refresh versus 60Hz). The 108x, 19xx, and 20xx series, and even most TVs, can display both PAL and NTSC. Sometimes it is necessary to perform minor tweaking of vertical hold and/or v. size on your monitor to achieve full display, though this is trivial.

What IS needed to switch between PAL and NTSC in software, is a "Fatter" (1MB) Agnus or better. If your system has more than 512K of CHIP RAM (use the avail command to find out) you have this chip. If you do not, ou can still construct a hardware switch (see below).

The following directions assume you're an NTSC user who wants to switch into PAL mode, but the procedure for going from PAL to NTSC is much the same.

The most common use of switching into PAL is for self-booting games, demos, etc. The best way to accomplish this on pre-3.0 systems is to use Chris Hames' Degrader program (most recent version: 1.30). Once you have procured this program, the switch is as simple as selecting "50Hz", and "50Hz System", then installing the program's ROMTag (little program that runs at boot-time) by pressing the "Survive Reset(s)" button.

AmigaDOS 3.0 added a PAL/NTSC switch feature to its Boot Menu, and all you need to do in order to access this is hold down both mouse buttons as the computer boots. Select Display Options, Display Mode/PAL, and then Boot. If the program still fails to go into PAL mode (Zool is one I've found), you may wish to get Degrader anyway and try that.

Software PAL/NTSC switching is easy and painless. Some people have gone even further by installing hardware PAL/NTSC switches. This results in a system which even the most nasty hardware-banging programs can't bring to its original configuration (if it has been switched). The procedure is simple – most Amigas have jumpers already – but beyond the scope of this document.

Therefore, if your screen is too short or cut off, seek the different screen mode. You'll "see new horizons", literally!

(Joseph Luk, jluc@eis.calstate.edu)

[7]

6 Emulators

What? The Amiga isn't good enough? You really want it to be another machine? Well, look here...

6.1 Can I run Unix on my Amiga?

Actually there are three Unix versions on the Amiga. All of them need at least 68030 (see Section 1.1 [MMU], page 1), probably a 68040 in the near future and seem to have problems with many hard-drive-controllers. You should have a good look into the documentation before installing it. Unix needs much resources, say at least 10Mb RAM and a 150Mb Unix-partition on the hard-drive.

1. Commodore offered a System V Unix in the past. It contained TCP/IP, X11 and other software and seemed to make a good job, but it was expensive and after all Commodore has dropped developing it. Commodore-Unix needs a streamer because it is distributed on tapes.

2. A Linux port is prepared. But actually there is not very much than the kernel. Specialists might like to use it, but it cannot be recommended for now. Linux is freely distributable. Sources: `ftp.tsx-11.mit.edu` or `ftp.uni-erlangen.de`, both directory '/pub/Linux/MIRROR.tsx-11/680x0'. Or `ftp.tu-muenchen.de`, directory '/pub/comp/os/linux/680x0'.

3. NetBSD is freely distributable too. Like Linux it isn't ready for now, but it seems to make big steps. Most GNU software is said to run, especially emacs and gcc. I think it's worth to have a look on it. Sources: `ftp.uni-erlangen.de`, Directory '/pub/amiga/unix/NetBSD-regensburg'. See Section 8.6.2 [Fish CD], page 40.

---

[7] PAL/NTSC switching with a MultiSync monitor is possible on 2.0+ and easily accomplished by moving the PAL monitor type to (or NTSC, as applicable, I have them both in) the Devs/Monitors drawer (from the Storage/Monitors drawer), reboot, then go into the Prefs drawer and select the ScreenMode program. You will see at least two PAL modes available (PAL and PAL Interlaced). Simply select one of the (usually non-interlaced for demos) and select "Use". All windows on the Workbench will close, the video mode will change, and then the windows will reopen – that simple!

Allen J. Newton, anewton@alturia.abq.nm.us

6.2 Is it possible to use the Amiga as X11 terminal?

Yes, it is. There are two different packages available:

GfxBase offers a commercial version which is said to be excellent, but expensive. The distribution includes different window managers andclients. A demo version is on Aminet. ('`gfx/x11/GfxBase-X11-Demo.lha`')

DaggeX is freely distributable and probably not finished yet. (It calls itself version 0.91.) Source: Aminet, '`gfx/x11/DaggeX-0.91.lha`' and '`gfx/x11/twm_930531.lha`'.

6.3 Is there a way to start MS-Dos programs?

PC-Task is a software only IBM-PC emulator for all Amiga computers. The current release is 2.03 which includes emulation of 8086 with MDA/CGA/EGA/VGA, serial and parallel ports, mouse, two harddrives and two floppies. A demonstration version can be obtained from aminet sites: '`/pub/aminet/misc/emu/PCTaskDemo203a.lzh`'. Chris Hames (bytey@melbourne.dialix.oz.au) (pctask@quasar.dialix.au)

7 Miscellaneous

This last chapter contains some questions that don't fit in the chapters above.

7.1 Is there any unix version of LhA?

See Section 7.2 [Endings], page 33.

7.2 What are files ending with ...?

Most endings on FTP sites or Fish disks tell you that the file is compressed and/or is an archive containing more than one file. Some programs even archive whole disks. Frequently found endings and programs to handle the related files are:

.sfx   Compressed archives which are included in an executable program: Just call the program and it will extract itself (sfx = self extract)

.lha

.lzh   Compressed archives; recommended: LhA ('`util/arc/LhA_e138.run`' on Aminet or Fish disk 715) or Lx ('`util/arc/lx100.lha`' on Aminet), Unix version available ('`misc/unix/lha-1.00.tar.Z`')

.dms   Disks compressed using DMS (‘`util/arc/dms111.sfx`’ on Aminet or Fish disk 406)

.zom   Disks compressed using Zoom (‘`util/arc/Zoom_5.4.lha`’ on Aminet, Fish disk 682);
              an older version which you probably need for uncompressing PasTeX is found on Fish
              disk 459.

.zoo   Compressed archive; recommended: Zoo (‘`util/arc/zpp2-10.lzh`’ on Aminet or Fish
              disk 527)

.Z

.z

.gz   Compressed files; recommended gzip (‘`util/pack/gzip124x.lha`’ on Aminet), note
              that this are Unix files in most cases

.tar   Acrhive; recommended: tar (‘`util/arc/tar.lha`’ or ‘`util/arc/gtar10.lha`’ on
              Aminet or Fish disk 445), note that tar is a Unix archiver and you often find soething
              like `.tar.Z`.

.arj   Compressed archive; recommended unarj (‘`util/arc/unarj-0.5.lha`’ on Aminet)

.zip   Compressed archive; recommended UnZip (‘`util/arc/unzip-5.1.lha`’ on Aminet),
              note that this are MS-Dos archives in most cases

7.3 Is there a Stacker-like utility to pack my hard drive?

XFH does a good job. It operates as a handler and uses the XPK-libraries, so you have different
compression modes (NUKE is a good choice) and possibly even more in the future. The only
disadvantage is, that the size of a file is limited by RAM: Don't use it with less than 2MB of RAM.

XPKDisk by Olaf 'Rhialto' Seibert is another stacker program that takes advantage of the
variety of XPK libraries available. Unlike other harddisk compressors it does not compress the files,
but creates a pseudo partition and stores the whole tracks as compressed files on your harddisk.
Its major advantage is that it does not limit the maximum file size, because it works similar to
trackdisk.device and therefore does not need too much temporary storage. But be careful: Never
try to optimize an xpkdisk partition using ReOrg. It will trash your virtual partition.

Another possibility is EPU. It's shareware and should offer the same as XFH, but without
limiting the file size.

Sources: Aminet, directory ‘`util/pack`’, Fish disks 754 (XFH) and 858 (EPU)

7.4 Where do I get Fish disk xxx?

Those FTP servers have that much space (or a CD-Rom) to have all fish disks available online:

```
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.funet.fi            (Finland, directory '/pub/amiga/fish')
```

Note that the CD-Rom's are not always mounted. See Section 8.3 [FTP], page 36.

Another possibility would be to ask your local dealer. :-)

7.5 What is MUI and where do I find it?

MUI is a set of shared libraries which are used to build a comfortable GUI (`Graphical user interface`). The general idea of MUI is that the programmer determines only the logical structure of the GUI and the look (Font, Size, Windows on Workbench, Public Screen, own Screen, ...) is determined by the user only. From the programmers point of view using MUI is simpler than `gadtools.library` but much more powerful. On the other hand MUI interfaces are slower than GadTools-Interfaces, especially on old 68000 machines.

MUI is found in two archives, one for developers only and one for users. Source: Aminet, '`dev/misc`'.

8 Where and how do I get software and other informations?

Three questions arise in this context: Which programs can be found, where are they and how to get them and transfer them home?

8.1 Files and databases on freely distributable software

Of course you need to know where you find things. Many good choices are listed in this paper, as I hope. Other sources are:

AmigaSciSchool

is a list of software and where you find it in Ascii format. It is posted monthly to the newsgroups `comp.sys.amiga.applications`, `comp.unix.amiga`, and `news.answers`. Additionally you will find it on Aminet sites ('`text/doc/AmigaSciSchool-4.01`'). It handles everything listed here and many more, for example GNU software, libraries (linked and shared), shells, Unix commands, educational and scientific software and much more.

FishCon   are lists of the Fish disk contents. ('`fish/doc/fishcon-???.lzh`' on Aminet)

FishXref   is a cross reference list of the Fish contents. (‘`fish/doc/fishxref-???.lzh`’ on
                Aminet) FishXref and Fishcon are in Ascii format.

KingFisher

      A Fish disk database, (‘`fish/doc/Kingfisher1_30.lha`’, which is the program and
      ‘`fish/doc/KFData850.lha`’ ,which contains the data, on Aminet or Fish disk 863)
      allows search by name and context. See Section 8.6 [Fish], page 40.

8.2 A collection of tests

`Comp.sys.amiga.reviews` is a moderated newsgroup where tests of soft- and hardware, books
and anything else relevant to the Amiga are posted. Its always a good idea to check this news-
group for a review, if you are interested in something special. Of course you always find only
the latest reviews, but the older postings are available per FTP from `math.uh.edu`, Directory
‘`/pub/Amiga/comp.sys.amiga.reviews`’ or on the Fish CDs.

8.3 Getting files from a FTP server

Things are easy for those who have access to the Internet and a program called FTP (File
Transfer Program). Nearly all Unix computers have it, but not all of them allow the use of FTP.

FTP allows you to gain access to some other machines and store and/or retrieve files. Normally
one needs an acoount on the remote machine to use it, bat a number of machines have a setup that
allows everybody to log in as the user `ftp` or `anonymous`, so anybody may get files from them. The
most important of this servers are the Aminet servers, which mirror each other and hence should
have the same files. They are the best choice if you are looking for Amiga software. Aminet hosts
are

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

All these mirrors have a directory ‘`/pub/aminet`’, where you will find much stuff. Please use a
mirror close to you! Some other important hosts are

```
ftp.funet.fi              (Finland)
ftp.isca.uiowa.edu        (USA)
ftp.hawaii.edu            (USA)
ftp.cso.uiuc.edu          (USA)
ftp.dfv.rwth-aachen.de    (Germany)
```

Note that grind, aachen and erlangen have the full collection of Fish disks available! See Section 7.4 [Fish disk xxx], page 34.

To connect to a special host (ftp.uni-erlangen.de for example), you should type

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

The host answers by requesting your login. You should type

```
ftp
```

No you are asked for a password. Please type your Email address here, if you have one. If not, use the password ftp.

Now you're inside the host. There is a number of commands you may execute here. The most important are:

?   Prints the help text of the FTP command. Additionally you may type **? command** to get information on a special command.

bin   Tells the FTP program that you whish to transfer binary files. It is always a good choice to type bin as the very first command! Files you load without the bin command can be corrupt.

get <file>   Loads the given file from the host. On most Unix machines you can type something like 'get file.txt -' or 'get file.txt |more' to show a text on the screen. Note that there **must not** be any blank between the | and the word more!

mget <pat>
             Loads the given files. pat may contain Unix style like wildcards.

put <file>
mput <pat>
             Like get and mget, but transfer files from you to the remote host. This is in most cases not allowed, except for a special directory called 'incoming'. You can place files here which you want to make public.

cd <dir>   Like the usual cd command. The commands get, mget, put, mput, dir and ls refer to the current working directory.

dir [<dir>]

ls [<dir>] Like 'list' and 'dir' on the Amiga. Note that the FTP-dir corresponds to the Amiga-list!

bye   Leaves the FTP program.

When you have used FTP for the first times you will notice, that you always begin with executing the same steps:

1. Type the login (ftp in most cases)

2. Type the password (your mail address in most cases)

3. Enter the bin command

4. Change the current working directory ('`/pub/aminet`' for example)

This may get executed automatically. What you need is a file called '`.netrc`' in your home directory. Note that it needs to be protected against others! The FTP program doesn't use it, if it can be read by anything else than you. (Protection is set using the command '`chmod go-rwx .netrc`'.) The .netrc file contains some entries for your most favourite FTP sites, each separated by empy lines. A typicel entry may look like this:

```
machine ftp.uni-erlangen.de
login ftp
password <your mail address> or <ftp>
macdef init
    bin
    cd pub/aminet
```

Note that on some machines it is possible to use the machine name '`default`' which meets all machines not listed in .netrc.

8.4 Getting files from a Mail server

Another way to get files is to use a mail server. This assumes that you can send mail to Internet addresses and get mail from Internet hosts. It works by sending a mail to the server specifying some commands, for example send commands for the files you want. Important mail servers are

```
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
```

The most important commands are:

Help   Tells the server that you wish to get an Ascii file containing detailed explanation how to use the server.

Limit <number>

Specifys that you wish to get not more than <number> Kbytes per mail. Larger files get splitted into small pieces of at most <number> Kbytes which are sent as separate mails each. Note that the mails may get larger because of overhead.

Cwd <dir>

Sets the current working directory to <dir>. This directory is used by the commands send and dir.

Index   will return a list of files and/or directories that the server offers. Note that this may be **very** large!

Index <item>

returns a list of files containing <item> in their names.

Dir [<dir>]

returns a list of the files and directories in the given directory.

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

Tells the server to send the given files to you.

Begin   Tells the server to ignore all lines above this command.

End   Like Begin, but specifies to ignore the lines below. (A signature for example!)

A typical session would be to send the following mail to the mail server:

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

8.5 Getting files from a mailbox

You also can get files from one of the many BBS's (`Bulletin Board System`). Most of these are driven by private users, so there is nearly everywhere a mailbox which doesn't cause a too expensive telephone bill.

I want to publish a list of all Amiga mailboxes here. So please send to me a list of all mailboxes which are related to the Amiga - in any way. I need all telephone numbers of the boxes and their name. Tell me the nation where the box resides and - if possible - some more info (Does the box have Aminet or the Fish PD or FRAS? Is the box especially for Amiga users?)

Thanks, Ignaz Kellerer

8.6 The Fish PD series

A very good source is the Fish PD series. One distinguishes between the floppy disks and the CD-Roms.

8.6.1 The Amiga Library disks

Fred Fish has started in the middle eighties to collect freely distributable software on floppy disks. There are 1000 disks and very much good stuff on it. Most Amiga dealers sell them and most magazines contain addresses of people mailing them to you for about 3$ per disk or less. Fred Fish has terminated offering software on floppy disks. Instead, there is a dealer who offers the new software from the CD-Roms on floppy disks.

There are some things which can be found on the Fish disks, but not on Aminet. However, you can get them with FTP. See Section 7.4 [Fish disk xxx], page 34.

8.6.2 The Fresh Fish CD-Roms

Fred Fish is going on to offer freely distributable software. But now he collects it on CD-Rom's. He releases two different kinds of CD's:

1. Monthly released disks are divided into roughly three sections:
    1. New material, which includes the material from the new unreleased floppy disks as well as material which does not appear in the floppy distribution, about 84Mb on the first disk.
    2. Useful utilities that can be used directly off the CD-ROM if desired, thus freeing up the corresponding amount of hard disk space (GNU Emacs, Gnu C, GNU C++, Amiga E, PasTeX, AmigaGuide, Installer, 2.0 and 3.0-Includes, different archivers, tape drivers, the AmiCDROM filesystem and many other GNU and BSD tools, . . .), about 150Mb on the first disk.
    3. Older material from previous released floppy disks or CD-ROM's, about 404 Mb on the first disk. (Fish disk 600-910)
2. Disks containing the latest software as well as recent software in packed format only. (These are intended to be used for example in BBS's.)

I recommend especially the first kind of Fish CD's. They cost about 20$ plus 3$ for shipping (5$ outside USA/Canada/Mexico) and are available from

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

Phone/FAX: (602) 917-0917
```

8.7 How do I Read and write MS-Dos disks?

No problem for owners of Workbench 2.1 or higher: The program CrossDos is part of the Workbench. All you have to do is mounting the device '`pc0:`' by putting it into the drawer '`Devs:DOSDrivers`' or by double-clicking the icon in '`Sys:Storage/DOSDrivers`'. Ms-Dos disks in drive '`df0:`' can now be handled in the usual manner replacing the word '`df0:`' by '`pc0:`'. For example the directory can be shown with the command `dir pc0:`.

People still running Workbench 2.0 or lower need a program called `MSH`. You will find this on the Aminet (directory misc/emu) and on Fish disk 382. See Chapter 8 [Sources], page 35. After editing the file '`devs:MountList`' as described in the documentation you have to say `Mount msh:` in the CLI and can now do the same as above replacing the word '`pc0:`' with '`msh:`'.

8.8 How do I split large files?

There are some archives which are too large to fit on one disk. To transfer them on disks you need to split them into smaller pieces and transfer each part on a separate disk. I recommend Martin Schlodder's `Splitter`. (Aminet, '`util/misc/splitter_121.lha`'. The archive contains binaries for MS-DOS and should be compilable without problems on any Unix system.

8.9 Discussing things

A mailing list is a server that allows to discuss special things via Email. The server holds a list of people that are interested in the respective topic. For example, the gcc mailing list will discuss bugs, new features and other problems of gcc. (see Section 3.5 [Compilers], page 10) If a mail is sent to the server from someone he will forward this mail to all the other people on the list.

You will be added to the list by sending a mail to the server which will usually contain the word '`Subscribe`' or something similar. Once you are no more interested in the mailing list you can unsubscribe.

Unforunately the servers have a different syntax of subscribing and unsubscribing. The best way to find how to subscribe is to send a mail with the word '`Help`' to the server. He will reply a mail which should explain how to handle this special server.

Some mailing lists are:

```
Topic           Server

Amok            listserv@amokle.stgt.sub.org
Dice            dice-request@castrov.cuc.ab.ca           or
                dice-request@hactar.hanse.de (Germany)
gcc             listserv@lists.funet.fi
```

```
Lisp            amigalisp@contessa.phone.net
Mui             mui-request@taloa.unice.fr
Oberon-A        oberon-a-request@wossname.apana.org.au
```

8.10 Other FAQ's

This is not the only FAQ, of course. Lots of FAQ's are available either by being posted to newsgroups or by access via FTP.

Amiga related books FAQ

> This is a list of books for the Amiga, including short discussions, prices and sources. Newsgroups: comp.sys.amiga.misc, comp.sys.amiga.introduction, comp.sys.amiga.programmer (monthly) Ftp: rtfm.mit.edu, pub/usenet/comp.sys.amiga.misc. Maintainer: Marc Atkins, atkin@cs.umass.edu

AmiTCP/IP FAQ

> This is for users of AmiTCP/IP, a set of programs which allows to include an Amiga into a TCP/IP network. (Most well known nets, Internet for example use TCP/IP.) Newsgroups: comp.sys.amiga.misc, comp.sys.amiga.datacomm, comp.sys.amiga.networking (biweekly) Ftp: rtfm.mit.edu, pub/usenet/comp.sys.amiga.networking Maintainer: Neil J. McRae (atcpfaq@domino.demon.co.uk)

Amiga Networking FAQ

> Unlike the AmiTCP/IP FAQ this one wants to cover all aspects of networking, including TCP/IP and Envoy. Newsgroups: comp.sys.amiga.datacomm, comp.sys.amiga.hardware Ftp: rtfm.mit.edu, pub/usenet/comp.sys.amiga.networking Maintainer: Richard Norman (norman@afas.msfc.nasa.gov)

Point Manager FAQ

> Networking seems to be quite a problem: This FAQ os for Point Manager, a FidoNet-client (so-called points). Newsgroups: comp.sys.amiga.datacomm Ftp: rtfm.mit.edu, pub/usenet/comp.sys.amiga.datacomm Maintainer: Eric Krieger (pm_faq@quasar.hacktic.nl)

All about FTP

> Explains the usage of the file transfer program FTP. See Section 8.3 [FTP], page 36. Newsgroups: comp.sys.amiga.misc (monthly) Ftp: Aminet, info/start Betreuer: Urban Dominik Mueller (umueller@amiga.icu.net.ch)

The Amiga-FAQ archive

The Amiga-FAQ is available in different formats: Ascii format (which is posted to the nets) AmigaGuide format (which is the adequate format on the Amiga) and in dvi format (to be printed). Additionally there is some stuff, that might be useful or interesting, but could not be included into the Amiga-FAQ:

```
    txt/amiga.history       On the Amiga's history
    txt/story.txt           The Commodore story (or: the Tramiel
                            story ;-)
    txt/amiga.newsgroups    Overview on comp.sys.amiga.*
    txt/amiga.sites         List of FTP sites
    txt/AmigaOverview.tex   A short overview on the Amiga-Soft- and
                            Hardware
    txt/Hardware.tips       For those people who can't live without solder
    src/JWSplit.c           The source of a file splitter
    src/JWJoin.c            The opponent to JWSplit
    src/addtoc.c            Utility to add a toc to texinfo-created docs
                            (this document uses it)
```

I decided to collect these in the Amiga-FAQ archive. It is called AmigaFAQxxxxxx.lha (where xxxxxx is the date of the last release) and can be found on Aminet, directory 'docs/misc'.

Contributions

This FAQ can neither get useful nor hit further development without your help. Suggestions, contributions, new answers, critics, anything is rather welcome.

Please note, that very major subjects are absolutely missing yet: Nothing about sound, nothing on graphic cards, no Animation. These are some of the Amiga's best points! But I don't know them . . . :-(

So grab your keyboard (Your pencil? Well, if there's no other way. . .) and send mail to:

```
        Ignaz Kellerer
        Georg-Habel-Str. 11
  81241 Munich (Germany)
        Tel. (+49) 089 / 885147

        Internet: kellerer@informatik.tu-muenchen.de
```

Credits

My thanks go to:

Reinhard Spisser and Sebastiano Vigna
>   for the Amiga version of texinfo. This is written with it.

The Free Software Foundation
>   for the original version of texinfo and many other excellent programs.

Dylan McNamee

> for contributing the sections on Editors, Word Processors, DTP and Postscript and some wording fixes.

Joseph Luk

> for help in the section on chunky/planar, double buffering and PAL/NTSC

Urban Dominik Mueller

> for the FAQ on FTP and Mail-servers.

Lars Hecking (lhecking@nmrc.ucc.ie)
Philippe Brand (phb@colombo.telesys-innov.fr)

> for the complete gcc part

Jochen Wiedmann (zrawi01@decap2.zdv.uni-tuebingen.de)

> for the composition and posting of the Amiga FAQ until July 1994.

Index

Table of Contents