ProNET

COLLABORATORS			
	TITLE : ProNET		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		December 8, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

ProNET

1.1 ProNET Docs

```
                    This is the documentation on
                              ProNET V2

                            A Network System

                (c) Copyright 1994-1995 by Michael Krause

    ++ ProNET Version 2 is (very-low-cost, non-crippled) SHAREWARE ++
```

1.2 ProNET Introduction

ProNET is a hardware/software combination that allows you to install a more or less simple network between two or even more Amigas, which means you can share all devices like hard disks, floppies and CD-ROM drives on all computers. You can use parallel cables and serial (null-modem) cables to connect your Amigas.

If you've used ParNet (by Matt Dillon) before, you might be used to this kind of software, but ProNET has got some advantages:

- installs a new AmigaDOS device for each target (no NET:)
- recognizes disk changes
- supports most of the V40 DosPackets
- 'cd's into network directories do work
- written in pure Assembler, thus short and a bit faster

Major improvements to Version 1 include:

- completely new pronet.device, modular interface driver concept
- no freezing while data transfer
- network restart
- transfer module for serial port
- reqtools-requesters now recognize ProNET devices
- lots of other bug fixes
- full source codes supplied
- completely new documentation

1.3 ProNET System Requirements

All software works on all systems with at least Kickstart 1.3 (or 1.2??) and any processor, except the Handler which requires OS 2.0.

The hardware, consisting of a simple cable, is connected to the parallel ports. You can use the cable you have used with ParNet -- If you don't have used ParNet before, here are the instructions:

1.4 ProNET Making a Parallel Cable

{ Copied from the original ParNet distribution }

The following connections need to be made on a DB25 cable:

You are making a cable that connects the two PARALLEL ports
of your Amiga together.

CABLE: Connect D7-D0,SEL,POUT, and BUSY across,

```
    Connect ACK (FLAG interrupt) to SEL locally:

  (2-9)   D7-D0   ------------    D7-D0
  (12)    POUT    ------------    POUT
  (11)    BUSY    ------------    BUSY       PARALLEL PORT
  (13)    SEL     --+------+--    SEL
  (10)    ACK     -/        \-    ACK
  (18-22) GND     ------------    GND  (18-22)
```

Double check the gender for the DB25 connectors you will need to connect to your Parallel Port.

A500/A2000, Parallel port is Female so you need a Male connector
A1000, Parallel port is Male so you need a Female connector.

The easiest thing to do is to buy a premade cable with all 25 lines passed and DB25 connectors on both ends (Double check the gender's that they match before you buy the cable!) And then rip it apart and cut-and-seal those wires which are not supposed to be connected. You also need to bridge ACK to SEL as per the diagram above ON BOTH ENDS OF THE CABLE, as shown above.

```
  ** NEVER PLUG IN AN UNMODIFIED CABLE BETWEEN THE TWO COMPUTERS!
  ** DOUBLE CHECK YOUR CABLE BEFORE INSTALLATION!
```

WARNING! INTERFERENCE WITH SERIAL-PORT.

The RI (Ring Indicate) line on the Amiga's serial port uses the SEL line to source a transitor. This interferes with the SEL line which, as you can see, is part of the network.

Be sure that

(a) either no serial cable is attached or that
(b) It doesn't connect RI or that
(c) your modem doesn't connect RI internally.

Note from the ProNET author: I didn't hear about any problems from people owning a modem and using ParNet at the same time - but that doesn't mean i take responsibility for any damages..

1.5 ProNET The AmigaDOS

This sections tries to briefly explain names like 'devices', 'handlers' and 'file systems'. If you find yourself familiar with these things you can skip this section and start installing. If you don't quite understand the install procedure, this section might enlighten you!

Because AmigaDOS can handle more than one data storage medium, each one has got its own name. The most known examples are 'DF0:' for the internal floppy disk drive and 'HD0:' for the boot partition of your hard disk. Every time you access a file on one of these media, you will state this name, provided the disk is not already the root of the current directory.

All these names describe a thing in AmigaDOS which is called the AmigaDOS Device. 'CD0:', 'HD7:', 'PRT:' and 'RAM:' all describe different AmigaDOS Devices, or simply Devices. As you have probably noticed about the example of 'RAM:', the AmigaDOS Device is only a symbol for a storage medium, since the Ram Disk is not really a part of the hardware!

Every Device features a control program, called Handler or FileSystem. These differ in that the FileSystem is used to collect a number of files on a storage medium, as opposed to a Handler, which is used with Devices that don't feature files, but merely push around data. An example of a Handler-controlled Device is 'PRT:'; or did you ever succeed in creating a directory on this Device?

Once a FileSystem gets a request to access a certain file, it hands some data over to a further level of abstraction, the Exec Device, which finally controls the hardware. Why this now? Let's take a hard disk and a floppy disk for example: Both are built in a very similar way -- they consist of a number of blocks of a fixed size organized in cylinders and tracks. Because of that these media are controlled by the same FileSystem -- it only has to be aware of the different size of the disks. The control of the read/write heads and drive motors which differs of course between hard disk and floppy disk is finally done by the Exec Device. In this way, the smallest amount of memory is consumed, because both media share a large block of code! Examples for such Exec Devices are 'trackdisk.device' and 'scsi.device'.
When dealing with Handlers instead, there sometimes are no underlying Exec Devices, or they are not controlling read/write heads and motors, but certain interfaces, such as the 'serial.device'.

To install a new AmigaDOS Device, one has to use the Mount command, which requires a so called MountList entry. There are two possibilities where to store this entry. The standard way previous to Kickstart 2.0 was to collect all entries in the file 'DEVS:MountList'. Alternatively, you can create a new file for each entry and put this file into the 'SYS:Storage/DosDrivers' directory. This file should then be called exactly like the Device which is described by it. Look into this directory, you'll find some stuff for PC0: there.

The 'Mount' command, called with the Device name to mount ('CD0:', 'HD3:' etc.) as an argument, will search all possible places for the entry. Additionally, the contents of 'DEVS:DosDrivers' are automatically mounted on system startup, which is not the case with the 'DEVS:MountList' and 'SYS:Storage/DOSDrivers'.

Please don't confuse the Exec Device and the AmigaDOS Device, though both are called just 'Device' in general!

1.6 ProNET Installation

- How to install ProNET -

Making a Parallel Cable The parallel connection cable

AmigaDOS	Always useful to know
pronet.device	The basic transfer routines
File System	The file network
Utilities	Some other tools

1.7 ProNET Installation - pronet.device

Copying the files

The pronet.device is the core of the whole ProNET package. It contains all routines for managing the data transfer between Amigas. It does NOT contain any transfer routines as was the case in version 1, because it turned out to be extremely hard to add new routines for other ports this way. This device should be copied into the 'DEVS:' directory of all participating machines.

The transfer routines are external code modules stored in the 'DEVS:ProNET' directory. You will have to copy all modules you want to use (because of their very small size, it will not hurt anyone to copy all of them). Please don't copy them directly into 'DEVS:', but keep them in their right directory!

Currently available transfer modules are 'internal-parallel' for the internal parallel port (really), as was the only possibility in ProNET V1, and 'serial' for any serial port which can be controlled via a 'serial.device' compatible Exec Device, that is 'BaudBandit.device', 'gvpserial.device' etc.

Creating the configuration file

Now create the file (or copy the example) 'DEVS:ProNET.config'. This file will contain configuration data for all applications using the pronet.device.

This file MUST contain at least one line of the form:

pronet-device-<unitnumber>: <drivername> <driverdata>

One line describes a so called Unit of the device, which is equivalent to a port accessable by one of the ProNET drivers. The more ports and drivers you have, the more Units you can create. You will be able to use every unit to connect to another machine, so if your central Amiga has a Multiport expansion with 3 serial ports, you will be able to connect to 5 (five!) external Amigas using the internal parallel and serial ports, as well as the Multiport card.

Every unit is identified by any positive number. The units are defined by lines of the type shown above. <drivername> is replaced by the name of the external driver module, and <driverdata> is data which is required by this driver.

Installing the 'internal-parallel' driver

First we will install the internal-parallel driver as Unit number zero. Of course you can use a different Unit number on another machine, it's your choice! You don't have to use this driver at all!
We create or change the already present line into:

```
pronet-device-0: internal-parallel 0 5
```

The number 5 is required by the driver module and describes the priority of the transfer process. If you don't know what I'm talking about, leave this number.

The number 0 is the so called machine number and is essential for a correct function of the network. Two machines connected via the internal-parallel driver need to have different numbers here. One machine requires a '0', the other one a '1'. If you give both sides the same number, you run the risk of some dangerous network deadlocks! (No, you don't destroy your hardware, the network just doesn't work the way you want it to)

This driver requires a ParNet compatible cable to be connected to the built-in parallel port.

Installing the 'serial' driver

We have to use a different Unit number here, let's say number one:

```
pronet-device-1: serial BaudBandit.device 0 38400
```

'BaudBandit.device' is the name of the 'serial.device' you want to use, '0' is the appropriate unit, which would describe the number of the serial port on multiport cards. Finally, 38400 is the baud rate, which should be equal on two computers connected together :) Try how high you can set it!

This driver requires a 7-wire/RS232 null-modem cable to be connected to the respective port.

1.8 ProNET Installation - File System

The procedure is here only described from the view of one main machine. It is ofcourse possible to do this crossed-over!

Copying the files

At first copy pronet-server into the 'C:' directory or somewhere else in the path of the remote machine, that is the machine from which you want to 'import' some AmigaDOS Devices into the main machine.

Put the pronet-handler into the 'L:' directory of the main machine.

Creating MountList entries

You must create a new MountList entry for every new device. We'll start with one device here. Let's assume your remote machine has a hard disk drive 'HD0:', which should also be accessable on the main machine. At first, we must create a device name under which this Device will be accessed. Since your main machine also features a hard disk 'HD0:', we use a different one, namely 'HD4:':

What we want to do now is to create an AmigaDOS Device on our main machine called 'HD4:', which is a copy of the hard disk 'HD0:' of the remote machine!

Now load your favourite text editor and open the 'DEVS:MountList'. Go to the end and insert the example ProNET MountList from this package there:

```
CD0:    Stacksize = 4096
    Priority  = 10
    GlobVec   = -1
    Unit    = 0
    Flags   = 0
    Device    = devs:pronet.device
    Surfaces  = 1
    BlocksPerTrack  = 1
    Reserved  = 0
    LowCyl    = 0
    HighCyl   = 0
    FileSystem  = l:pronet-handler
    Mount   = 1
#
```

It doesn't matter if you don't understand this stuff, important thing is to change the two items set in italics:

CD0: should be replaced by the name you intend to give the device, i.e. for our example HD4:.

The number in the 'Unit' line determines which port to use. Insert a number here which you defined in the configuration file some minutes ago, e.g. 0 for the internal parallel port!

Until now your Amiga still doesn't know exactly what device to access on the remote machine. You want HD0: to be accessed, and this is what you add to the 'DEVS:ProNET.config' file:

```
HD4: HD0:
```

This means that the ProNET Device 'HD4:' will access the device 'HD0:' on the server machine, whatever this machine is or where to find it.

Both parts belong indivisibly together, MountList and ProNET.config entry. If one of these things is missing, you can't install the network. Another mistake which was done by users from V1 quite frequently is that the same MountList was created on the server machine. This is not necessary or even disturbing, since the pronet-server is enough! You need to create MountList entries there if you want to import devices 'crossed-over', but in most configurations one machine is just a data server.

Note that you have to repeat these steps for every new device, e.g. 'RAM:' which you might call 'RAN:' on the main machine! You could even call it 'REMOTERAMDISK:', the name doesn't matter at all.

For further possibilities of configuration see chapter Configuration.

1.9 ProNET Installation - Utilities

pronet-run

Copy 'pronet-run' into C: of your main machine.

pronet-page

Copy 'pronet-page' into C: of your main machine.

pronet-talk

Copy 'pronet-talk' into C: of all participating machines. Then add the following line to ProNET.config, if it doesn't already exist:

pronet-talk: 1994

The number 1994 specifies the ProNET.device Port Number, which is totally uninteresting for the user. Very important is to keep this number equal on all machines!!

1.10 ProNET Starting the Network

Start 'pronet-server' on all machines that are file servers with the first argument being the ProNET Unit to serve for, that would be in our example '0' for the 'internal-parallel' and '1' for a 'serial' port. You must start the 'pronet-server' for every Unit again!
Then use the 'mount' command to install the network devices.

If your main machine crashes, you don't need to reboot the servers IF THERE WAS NO PRONET-SERVER RUNNING ON THAT MACHINE! It is completely sufficient to mount the devices again.

The current version of the 'internal-parallel' driver only works 100% stable while both machines are switched on. As soon as one machine gets switched off, the other one might hang or crash!

1.11 ProNET Example Configuration

This chapter is for people that didn't understand my documentation. I will show you here the configuration I'm using every day.

My main computer is an A1200. Another Amiga, an A500, is located under my table and an A570 CD-ROM Drive is attached to it. I use ProNET to import the CD-ROM into the A1200 via the parallel port.

First, the ProNET.config files:

------- >·8 --- >·8 --- >·8 --- >·8 --- >·8 --- >·8 --- >·8 ---

A1200:

```
pronet-device-0: internal-parallel 1 5

pronet-talk: 1994

DF2: df0:
CD0: cd0:
RAN: ram:
```

A500:

```
pronet-device-0: internal-parallel 0 5

pronet-talk: 1994
```

------- >·8 --- >·8 --- >·8 --- >·8 --- >·8 --- >·8 --- >·8 ---

Now the MountList I use on the A1200:

------- >·8 --- >·8 --- >·8 --- >·8 --- >·8 --- >·8 --- >·8 ---

```
DF2:  Stacksize   = 4096
   Priority = 10
   GlobVec     = -1
   Unit     = 0
   Flags    = 0
   Device      = devs:pronet.device
   Surfaces = 1
   BlocksPerTrack = 1
   Reserved = 0
   LowCyl      = 0
   HighCyl     = 0
   FileSystem  = l:pronet-handler
   Mount    = 1
#

CD0:  Stacksize   = 4096
   Priority = 10
   GlobVec     = -1
   Unit     = 0
   Flags    = 0
```

```
   Device      = devs:pronet.device
   Surfaces = 1
   BlocksPerTrack = 1
   Reserved = 0
   LowCyl      = 0
   HighCyl     = 0
   FileSystem  = l:pronet-handler
   Mount = 1
#

RAN:  Stacksize   = 4096
   Priority = 10
   GlobVec     = -1
   Unit     = 0
   Flags    = 0
   Device      = devs:pronet.device
   Surfaces = 1
   BlocksPerTrack = 1
   Reserved = 0
   LowCyl      = 0
   HighCyl     = 1
   FileSystem  = l:pronet-handler
   Mount    = 1
#

------- >·8 --- >·8 --- >·8 --- >·8 --- >·8 --- >·8 --- >·8 ---
```

In order to start the network, I switch on both computers, run the 'pronet-server' on the A500 (I can leave out the argument '0' because that's the default) and use 'mount cd0:' on the A1200. Additionally, I can import the remote RamDisk by using 'mount ran:' and the remote disk drive by using 'mount df2:'.

1.12 ProNET pronet-talk

pronet-talk is a program which two network users can use to chat. It opens a window on the Workbench screen and transmits all input immediately through a ProNET Unit of your choice. All data arriving at the pronet-talk port at that Unit is displayed in the same window.

Simply run 'pronet-talk' with an optional argument, the Unit number. If no argument is supplied, pronet-talk uses the default '0'. This is what one would do with the example configuration from above.

1.13 ProNET pronet-run

pronet-run enables you to execute shell commands on a remote machine without having to use its keyboard. You give pronet-run a command which is started on the remote machine as if you had typed it in there. Another argument is the Unit number to which the command should be sent. The pronet-server must run on the remote machine.

```
 Examples:

  pronet-run 0 "dir cd0:"
  pronet-run 1 "list >ram:filelist sys: all"

 Note that the command string is put in `"'s.
```

1.14 ProNET pronet-page

```
 pronet-page sends a message to a remote machine. The message will be
displayed in a small Alert window and must be replied by use of the left
mousebutton. The pronet-server must run on the remote machine.

 Example:

  pronet-page 7 Hey, Unit 7... are you still working??

 Note that the message is _not_ put in `"'s.
```

1.15 ProNET Configuration

```
 Here's the standard Mountlist entry for usage with the 'pronet-handler':

XXX:  Stacksize = 4096
  Priority  = 10
  GlobVec   = -1
  Unit    = X
  Flags   = 0
  Device    = devs:pronet.device
  Surfaces  = 1
  BlocksPerTrack  = 1
  Reserved  = 0
  LowCyl    = 0
  HighCyl   = 0
  FileSystem  = l:pronet-handler
  Mount   = 1
#

 There are 2 lines that may be interesting: At first, the 'Mount = 1'
line instructs the 'Mount' command to load the handler immediately, thus
installing the 'pronet.device', which results in waiting for the other
machine. If you want to mount all the network devices in your
startup-sequence (or the DEVS/DosDrivers directory) for later usage, you
have to change the '1' into a '0'. So when you intend to startup the
network you just make a 'dir cd0:' for example without having to mount it
at first !

 The HighCyl line: It can be possible that network-volumes have the same
name as a volume on the main computer, e.g. when putting together two
Amigas with hard disk, both having a partition called 'Workbench'! That
may cause you and AmigaDOS problems. To avoid this, you can change the 0
```

into a 1 in this line. Then ProNET inserts the Unitnumber at the beginning of every volume name inserted in the corresponding device.

1.16 ProNET Known Problems

The A570 CD-ROM Drive has problems recognizing disk changes. If you change the CD, just type a 'diskchange cd0:' either on that machine or over the network.

Large ACTION_WRITEs may lead to problems, since a corresponding amount of memory needs to be allocated on the server machine. This may occur when copying huge files via Workbench. Most DOpus like programs transfer data in small steps instead.

1.17 ProNET Update from V1 Notes

A lot has changed since Version 1. I want to help you upgrading here. Please note first that the concept of 'pronet.device' has changed completely. It would be best if you deleted everything and followed the install procedure as described above. Please don't confuse the name 'Unit'. What was called Unit in V1 is now the machine number, which only exists in the internal-parallel driver.

The MountList entries now feature a '1' in Surfaces and BlocksPerTrack. This is to let ReqTools recognize network devices.

Note that pronet-talk, pronet-run and pronet-page all need a Unit number specified now. Also, pronet-run requires the command string to be set in '"'s.

If your main machine crashes, you now don't need to reboot the servers. It is completely sufficient to mount the devices again. The servers will automatically notice that you have rebooted.

The final change is that I've delete the 'install' drawer. It is very important that users read the documentation, so that they understand what they do.

1.18 ProNET Registration

"Yes, another one of these Shareware programs..."

But I'm convinced of the fact that paying the Shareware fee won't kill you, because it is extremely low for a program of this power (and for any program), namely 10 DM, which is about 7 US$. I believe that a higher price is too much for someone who doesn't use his machine(s) 24 hours a day; a fact that applies to most people I know, me included.

If you really find my package worth 10DM, please send this amount of money or an equivalent amount in US$, £, SFr or FF (NO COINS please!) to:

Michael Krause
Mannesallee 24
D-21107 Hamburg
GERMANY

Of course you won't get a keyfile or something similar, because you already own a non-crippled version of ProNET V2!

1.19 ProNET Support

Due to the low registration fee and the costs of 'normal' mail, it's generally not possible for me to answer every question concerning ProNET by post mail. If you've got an Internet address, the chance of an answer will rapidly increase. Also try to find a solution to your problem in the FAQ section of this guide first.

New versions of ProNET will be immediately spread via Aminet, the Fred Fish CD-ROMS and BBSes. If you want to have a new version directly mailed to you on disk, the registration fee increases to 15 DM.

If you have to report a bug, send a letter to the address above or to this Internet address:

rawstyle@online.sh.sub.de

Bug reports should be as exact concerning the configuration as possible; ahem... it doesn't matter which mouse you have got...

By the way, it seems that the routing to my email address is very slow and sometimes it doesn't work at all, but if a mail reaches me, it will be generally answered on the same day!
Besides that, please don't send large files, because I have to pay for both incoming and outgoing mail (german mails are free)! Thanks.

1.20 ProNET Credits & Thanks

pronet.device,
pronet-server,
pronet-handler,
pronet-talk,
pronet-run,
pronet-page
and all related files by

Michael Krause

ParNet and the connection cable by

Matt Dillon

Beta Testing (no order) by

Christoph Dietz
Holger H.
Michael Schepers
Thomas Schwarz
Wolfgang Gutberlet
Nicola Soggia

[thank you very much!]

Many thanks to Nik Soggia <nsoggia@falcon.telnetwork.it> for finding a solution to the ReqTools problem from V1!

Besides all that I want to thank all people that sent letters containing more than just 'ProNET does not work' and all the ones encouraging me to continue working on this package. Version 2 would never have been released if I hadn't received so much mail from you!

1.21 ProNET History

Version 1 released on 01-Nov-94 (initial release) including:
- pronet.device 33.9
- pronet-handler 33.5
- pronet-server 33.5
- pronet-talk 33.0
- pronet-run 33.0
- pronet-page 33.0

Version 2 released on 24-Jun-95 including:
- pronet.device 35.4
- pronet-handler 34.2
- pronet-server 34.9
- pronet-talk 34.1
- pronet-run 34.1
- pronet-page 34.0

- If the main machine resets, there is no need to reset the server any more.
- The device doesn't freeze the machine.
- Priority of transfer process can be set now
- Disk change recognition improved again
- Completely new documentation
- READ-actions are now transferred splitted, and don't require as much RAM on the server as in V1.
- EXAMINE-Packets could cause memory loss on the server
- Completely new pronet.device, with modular driver concept
- "+"-Expansion >> Unitnumber
- C include file
- everything I forgot to write here
- Lots of bug fixes, thus a lot more stable

1.22 ProNET Future Development

I am planning to add these features to ProNET in the next days, weeks or months:

- Drivers for Multi-Port cards
- 'Netted' CON:/RAW: windows
- proserial.device for using the remote serial port(s)
- Graphical transfer statistics program
- SANA-II device interface
- Unmounting of the network

- ...Your suggestions!

1.23 ProNET Frequently Asked Questions

- Is it possible to stop the network in order to do some printing and then reconnect both Amigas via a parallel port switch?

  No, this is not possible, and it will not be in future, because the network relies on the parallel port registers not to be changed by the Operating System.

- Does there exist a 'pronet.device' for add-on parallel ports, like the Multiface card?

  There is no such device yet, as I can't find developer information on these cards on direct access basis. The 'parallel.device' substitutes don't help. Perhaps there's someone out there who knows how to do it!

- Is there a possibility to start a program on the remote machine and see its output in a CON-Window on the main machine?

  No, this is not possible as yet, but will perhaps be implemented in the future.

- Is it possible to use the serial port of the remote machine to e.g. connect a second modem or using a MIDI interface and a modem at the same time?

  This is a thing I'm working on currently (hope I will ever finish all the things I've begun programming...). Thanks to Giovanni Gigante for the great idea!

- Will the 'pronet-handler' ever run on Kickstart 1.3?

  No.

- Would a special 020/030/040 version of 'pronet.device' improve its

speed?

I don't think so -- but please, try changing the source code! :)

- I've discovered a bug: When I reset the main computer and mount all network devices again, I can continue working without problems. However, when I reset the machine on which pronet-server is running and start pronet-server again, the network is dead.

  This is no bug, because the pronet-server creates some structures for the main machine which are cleared if you reset the server machine! A conclusion from that is, that 'crossed-over' networks, where both machines are server and client, are not allowed to do a reset at all! This is an essential difference between ProNET and 'real' network packages, such as Envoy.

- Can pronet.device used as an interface driver for SANA-II compliant network software?

  No, this is not possible, since my device is not SANA-II compliant :) However, I intend to code a device stub so you could use all ProNET drivers with e.g. Envoy or AmiTCP.

- Why didn't I program my device SANA-II compliant??

  This is because I've never heard about this network standard until one month ago :(

- I find it disturbing that the shell I started 'pronet-server' from is completely unusable then.

  As is the case with _every_ CLI program, you can start it with
  run >nil: <nil: pronet-server
  to suppress all output and put it into the background.

- When I switch off one machine, the other one immediately stops working.

  I have not fixed this yet, because the remaining machine would be useless anyway without the second one.

Hmm... sorry that there were not too much questions I could answer with a simple 'Yes' :) ohh well but let me think about that again... ahhh here it is:

- Is ProNET better than ParNet?

  <Guess> :-)

1.24 ProNET Disclaimer

"I won't read this yet again..."

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDER AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

1.25 ProNET License

- This license applies to the product called "ProNET", a collection of programs for the Amiga computer, published by Michael Krause under the concepts of shareware, and the accompanying documentation. The terms "Program" and "Package" below, refer to this product. The licensee is addressed as "you".

- You may copy and distribute verbatim copies of the package as you receive it, in any medium, provided that you conspicuously and appropriately publish only the original, unmodified package, with all copyright notices and disclaimers of warranty intact and including all the accompanying documentation, example files and anything else that came with the original.

- Except when otherwise stated in this documentation, you may not copy and/or distribute this program without the accompanying documentation and other additional files that came with the original. You may not copy and/or distribute modified versions of this program.

- You may not copy, modify, sublicense, distribute or transfer the program except as expressly provided under this license. Any attempt otherwise to copy, modify, sublicense, distribute or transfer the program is void, and will automatically terminate your rights to use the program under this license. However, parties who have received copies, or rights to use copies, from you under this license will not have their licenses terminated so long as such parties remain in full compliance.

- By copying, distributing and/or using the program you indicate

your acceptance of this license to do so, and all its terms and conditions.

- Each time you redistribute the program, the recipient automatically receives a license from the original licensor to copy, distribute and/or use the program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein.

- You agree to cease distributing the program and data involved if requested to do so by the author.

1.26 ProNET Sourcecode

There were some people that urged me to include the sources in the ProNET package. And so I did, here they are. They were all written in Assembler. There are no Makefiles, because they are mostly single-file sourcecodes. They should be compilable by PhxAss, DevPac and similar assemblers.

I've included the source codes because many problems concerning AmigaDOS could be easily solved by having a look at them. I've spent a whole year on developing ProNET and I want to help programming beginners to gain knowledge about the Amiga's OS.

Commercial use of the sources is not intended. Besides that, I don't want anyone to change them and recompile a completely new ProNET distribution. This will be permitted if I should somewhen quit the Amiga scene and stop developing this package.

To cut it short: The sources were included as a programming tutorial.

1.27 ProNET Programmers

It is now extremely easy to add new drivers to ProNET. I realized that writing a completely new pronet.device was a bit too hard, so I introduced the modular driver concept in V2.

A ProNET driver is a simple executable which is copied into "DEVS:ProNET/". This executable will be loaded for every ProNET Unit which requires it; it doesn't have to be pure. The code is entered at the very first position, which must contain an initialization routine:

d0 contains the bytes "RST!" for identification, so that this driver can not be executed by accident.

d1 is a Flags register. It is _not_ mapped to the Flags argument of the OpenDevice call!! Currently, this should be ZERO; and if it contains another value, your driver should return an error.

a1 contains a pointer to the configuration string behind the driver ID, for example:

```
pronet-device-7: yourdriver bla this is data for your driver
```

a1 would then contain a pointer to the null-terminated string 'bla this is data for your driver'.

a0 finally points to an array which must be filled out by your routine:

```
   UBYTE ReadSignalBit
   UBYTE pad0
   APTR  ReadQuery
   APTR  Read
   APTR  Write
```

ReadSignalBit must contain a valid signal bit for the pronet.device task. The task will include this bit into its Wait() loop and call the function pointed to by 'ReadQuery', if this bit becomes set.

Your ReadQuery routine must analyze the incoming packet and return the following values:

- d0 length of the packet's main data
- d1 ProNET destination port
- d2 ProNET source port

If your routine comes to the conclusion that it was called by mistake and no incoming packet is pending, it must return NULL in d0! If this routine returns something useful, it's guaranteed that the routine pointed to by 'Read' will be called after that.

The Read routine will be called with a pointer to the place to put the packet data into in a0. It must copy the incoming packet there and return. You don't have to return an error code, because this routine should _always_ work!

Let's come to the Write routine now. It is called when the pronet.device wants your driver to transmit data to the other machine. The routine will be called with following values:

- a0 *data1
- d0.l length of data1 (guaranteed to be even)
- a1 *data2
- d1.l length of data2 (guaranteed to be even)
- d2 destination port
- d3 source port

Both chunks of data must be appended so that the other side thinks you've transmitted only one big chunk of data! The expression d0+d1 is always guaranteed to be lower than or equal to $4000. If d1 contains zero, only one chunk of data is transmitted.

If your routine succeeds in transmitting the data, you must return NULL in d0; if timeout occured or the line was busy or anything else which prevented you from transmitting, you must return a different value.

The initialization routine must preserve registers d2-d7/a2-a6 and return an error code in d0. If NULL is returned, the initialization succeeded, other values indicate an error.

Some last words: Please don't code something like a startup handshake,

because one side can always reset and try to reconnect to the other one. If you can't prevent yourself from doing this, add a note to your docs that reconnect is impossible!

1.28 ProNET Supported DosPackets

ProNET supports most new DosPackets introduced with V36 and V39: Perhaps you've sometimes seen one of these '2.0 Pkt ACT_PARENT_FH' requesters from ParNet - they are no longer annoying you! So this is a list of all packet types supported by ProNET:

Name	introduced with
ACTION_FINDINPUT	
ACTION_FINDUPDATE	V33
ACTION_FINDOUTPUT	
ACTION_END	
ACTION_READ	
ACTION_WRITE	
ACTION_SEEK	
ACTION_CURRENT_VOLUME	
ACTION_SET_FILE_SIZE	V36
ACTION_LOCK_RECORD	V36
ACTION_FREE_RECORD	V36
ACTION_LOCATE_OBJECT	
ACTION_FREE_LOCK	
ACTION_COPY_DIR	
ACTION_PARENT	
ACTION_SAME_LOCK	V36
ACTION_CREATE_DIR	
ACTION_CHANGE_MODE	V36
ACTION_FH_FROM_LOCK	V36
ACTION_COPY_DIR_FH	V36
ACTION_PARENT_FH	V36
ACTION_EXAMINE_OBJECT	
ACTION_EXAMINE_NEXT	
ACTION_EXAMINE_FH	V36
ACTION_DELETE_OBJECT	
ACTION_RENAME_OBJECT	
ACTION_MAKE_LINK	V36
ACTION_READ_LINK	V36
ACTION_SET_COMMENT	
ACTION_SET_DATE	
ACTION_SET_PROTECT	
ACTION_INFO	
ACTION_RENAME_DISK	
ACTION_INHIBIT	
ACTION_FORMAT	V36
ACTION_SERIALIZE_DISK	V39
ACTION_MORE_CACHE	
ACTION_WRITE_PROTECT	
ACTION_IS_FILESYSTEM	V36

```
ACTION_NIL
ACTION_FLUSH
ACTION_DISK_INFO

These packets are not yet supported:

ACTION_EXAMINE_ALL    V36
ACTION_EXAMINE_ALL_END    V39
ACTION_SET_OWNER    V39
ACTION_GET_DISK_FSSM
ACTION_FREE_DISK_FSSM
ACTION_ADD_NOTIFY   V36
ACTION_REMOVE_NOTIFY    V36
```