README

COLLABORATORS			
	TITLE : README		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		July 19, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

README

1.1 README.guide

```
NSDPatch
********

   This document contains all the information needed to set up and use
NSDPatch.


  Copyrights & Distribution
  Background
  Overview
  Contents
  Installation
  Tutorial
  Support
  Common Problems, FAQ
  Big Disk Support
  To Do
  History

NOTE: On pre V39 (< Workbench 3.0) AmigaGuide you will see some funny
marks on the screen.  They are V39 formatting commands which make the
text look nicer under >=V39.  They do not reduce the amount of
information shown in this document.
```

1.2 README.guide/Copyrights & Distribution

```
Copyrights, Warranty, and Legalese
**********************************

   Copyright ©1996-99 by Heinz Wrobel

   WARRANTY: None whatsoever. Standard disclaimer applies.

   LEGALESE: FREELY REDISTRIBUTABLE, NOT PD! CHANGES NOT ALLOWED!
```

1.3 README.guide/Background

```
Why did NSDPatch become reality?
********************************

What is NSD?
============
```

NSD is referring to the New Style Device standard as originally documented on the Amiga Technologies Amiga Developer CD in the

```
Amiga_Dev_CD_v1.1:DevInfo/DeviceDevelopment
```

directory. It is intended to provide for clean device usage in the future. In the meantime, updates are available on current developer CDs. So please get a current one.

Updates to this standard are available via the Internet on 'ftp.amiga.de' or 'www.amiga.de' and it is probably a good idea to check for them once in a while.

If you want more information about NSD, you can also contact the author of this document at the address mentioned in Support.

```
A few lines of history
======================
```

NSD, the official method to identify capabilities of a given Exec device/unit combination, is not available for all the existing devices out there. So a bunch of people spent some time to implement a benign OS patch that puts NSD capabilities into basically any Amiga with at least OS 2.04 while not adversely affecting existing setups.

Some were very directly involved in thinking up and testing NSDPatch. Here is a list in no particular order:

- Olaf Barthel
- Bernhard Möllemann
- Michael van Elst
- Angela Schmidt
- Klaus Burkert
- Joanne Dow

```
Thank you.
```

To read what NSDPatch can do for you, continue with the Overview.

1.4 README.guide/Overview

```
Overview
********

What is NSDPatch?
=================

   NSDPatch will hook into the OS (V37+, 2.04+) and patch almost
arbitrary types of Exec devices to make them look like NSD compliant
devices. With a configuration file, you can set up NSD like behaviour
even for most devices that were not known to the authors of NSDPatch.
NSDPatch does not affect DOS devices like 'PAR:' in any way. NSDPatch
only handles Exec devices like 'serial.device'.

   NSDPatch also includes some support for trackdisk like devices to
emulate the NSD 64 bit access commands on top of old devices. With
NSDPatch and the AI V43 FFS, you can use partitions >4GB on disks >4GB
with controllers that don't know about NSD. Read on to find out how to
set this up.

   NSDPatch also includes RDB mount functionality to support special
configurations.

   NSDPatch also fixes some bugs with certain devices like the V40
mfm.device or incompatibilities of, e.g., a JAZ drive and V40
scsi.device.

   NSDPatch will allow you to turn on correct command rejection via
IOERR_NOCMD for devices that would crash otherwise.

   NSDPatch will optionally try to do some magic for broken SANA2
devices, too.

   NSDPatch makes it possible to install device/unit mappings to
virtually rename device units to support broken software that tests
only the device name for assumed functionality.

   NSDPatch may also be useful for developers of NSD using SW to
simulate different and possibly incomplete or buggy NSD environments
for testing.

What is NSDPatch not?
=====================

   NSDPatch is not intended to replace an NSD upgrade for old devices
forever. It is also not intended to provide for every single NSD bell
and whistle in the specs. It is intended to ease the migration path and
to give some basic NSD capabilities and convenient device related fixes
to those who otherwise couldn't have it at all.
```

1.5 README.guide/Contents

What is included?
****

This is a short list of the files that you got. For details on what to do with them, continue with the chapter Installation.

- 'NSDPatch'

  The main program.

- 'NSDPatch.cfg'

  The default config file. This can be edited to your needs. Please report additions that you make back to the author listed in Support to make inclusion in future versions possible

- 'NSDQuery'

  A useful tool to check Exec devices for their NSD capabilities or for their general 'IOERR_NOCMD' support.

- 'Install'

  A simple Installer script to add NSDPatch to your system. Sometimes it is easier to do it manually as described in Installation.

- 'NSDPatchInstall.rexx'

  An AREXX script needed to support the Installer script 'Install'.

- 'README.guide'

  This document.

1.6 README.guide/Installation

Installation
****

READ THIS!

There is an Installer script 'Install' included, that does a quick installation and setup of NSDPatch, including its configuration. If you go this way, you may still want to read below about what happens. If you don't want the automatic installation, you can read below about the details of a manual installation. The automatic installation works best with an unmodified 'Startup-Sequence'. If you have doubts here, you may want to install NSDPatch manually.

Whatever you do, it is really not that hard. You don't have to understand every bell and whistle of NSDPatch to install or use it. Typically, you just plug it in and go as described below.

Keep in mind that NSDPatch is a fairly low level OS patch. So before starting any kind of installation, save all your current work. You may have something strange in your system that may cause a crash.

1.7 README.guide/Basic Installation

Basic Installation
==========

1. Place a copy of the file 'NSDPatch.cfg' into 'DEVS:'.

2. Copy NSDPatch into 'SYS:C'.

3. Place these lines into your 'SYS:S/Startup-Sequence' immediately after 'SetPatch' is called:

```
IF EXISTS C:NSDPatch
  C:NSDPatch
ENDIF
```

   It is very important that these lines are added immediately after 'SetPatch'. That is why NSDPatch must be placed into 'SYS:S/Startup-Sequence' and not in 'User-Startup'.

   Alternatively, you can run the included Installer script 'Install' with Installer V43, which does a quick installation of NSDPatch, provided that your startup is reasonably unmodified.

   SPECIAL NOTE: If you map trackdisk.device units, you must start any 'noclick' system hack after SetPatch but before NSDPatch.

4. Reboot and watch the messages that are output by NSDPatch. These messages tell you what NSDPatch does. Devices may be listed that you don't have or use in those messages. This is not a problem. If you can't agree with the output in some way, edit the configuration as described in Custom Installation. Normally, you don't need to do anything here.

5. Assuming there were no problems, add the 'QUIET' option to the 'C:NSDPatch' line in your startup. This suppresses the messages on further bootups. Problems should always be reported to the address in the Support chapter.

6. If you intend to modify the default configuration, read Custom Installation.

7. That's it. This is the end of a successful installation. If oyu have any problems with it, check Trouble Shooting.

1.8 README.guide/Custom Installation

```
Custom Installation
===================

NOTE: This assumes that have done the Basic Installation successfully.
If not, please check Trouble Shooting before proceeding.

   First, take a look at the default configuration file 'NSDPatch.cfg'.
It describes in detail all the available configuration options and
contains all the entries for a basic OS 3.1 based system including
entries for devices known to be broken.  You don't need to understand
every single option for NSDPatch right now, but the general overview of
what is in that file may well help. Before adding patch line options,
always reread their respective descriptions.

   When you have looked over the configuration file and want to boldly
go where quite a few now have gone before, check the useful points
below.

   · You may have third party devices which don't support the
     'IOERR_NOCMD' error value correctly. This means that the driver
     will not reject unknown commands with 'IOERR_NOCMD', but either
     crash or do very strange things instead. An easycheck for this
     type of broken device is described in Trouble Shooting. In that
     case, use the 'IOERRNOCMD' option for the respective patch
     configuration line. If you want to boot from such a device, read
     the description of the 'RDBUNIT' option carefully. If you
     encounter such a device, please report it to the address mentioned
     in Support.

   · You may have a trackdisk like device supporting the third party
     TD64 command set. In that case, use the 'TD64' option for the
     respective patch configuration line. By using the 'TD64' option,
     the official NSD 64 bit commands will be rerouted to the already
     existing TD64 functionality instead of being emulated via
     'HD_SCSICMD'. Current Guru-ROM's and Phase 5 SCSI devices may
     support TD64 but not yet the corresponding NSD commands. No exact
     versions can be named at this time. If you know them, please
     report them to the address mentioned in Support.

   · NSDPatch can be invoked multiple times to install more patches.
     Installed patches cannot be changed without a reboot.

   · You can invoke NSDPatch with a config file that doesn't actually
     patch anything but only uses the 'ACTIVATE' and 'RDBUNIT' features
     of NSDPatch for late mount functionality.  You can also do this
     for single lines with the 'PATCHCONFIGLINE' command line option of
     NSDPatch. All of this is considered a positive side effect of the
     NSDPatch design.

   · You may have a SANA2 device that can't handle a NULL
```

'ios2_BufferManagement' pointer on startup. If you have one of those, you will get Enforcer hits with most likely an access to address 0 first on a check with NSDQuery when Enforcer is installed. In that case, specify the 'SANA2MAGIC' option in the configuration.

- Optionally, check the configuration file in 'SYS:DEVS' for any devices that you don't want to have patched and comment out these lines. Add entries for any special devices. Please report configuration lines for 3rd party devices to the address named in Support. Keep another copy of your changes in case of future automatic updates. Use NSDQuery to check if a device already supports NSD.

1.9 README.guide/Trouble Shooting

```
Trouble Shooting
================
```

NSDPatch is considered to be very stable and benign. Also, the default configuration should already cover most of the broken things that are out there. If you encounter problems, there may be some useful hints to solve them below.

If you find a solution or can't solve a problem with the hints given below, please report the issue to the address in Support.

If NSDQuery behaves strangely on any device, patched or not, please report the exact circumstances, including all configuration and version information for your Amiga to the address in Support.

NOTE: Never change too many things at once when trying to solve a problem. It typically only confuses things.

- It was a little white lie that NSDPatch must be started right after SetPatch for the first time. While it has a much better chance of catching everything at that point of time, you may well start it from the command line later during testing. When I am testing new patch lines or new features, I temporarily remove NSDpatch from 'C:' and start it manually after bootup to test things. For testing and trouble shooting of isolated events, this is permissible.

- To determine active broken devices, do a crash test after saving all your current work. Run 'NSDQuery CHECKALL BRIEF'. This checks all currently active devices fairly quickly for compatibility problems. If you have very broken Exec devices, the machine may crash after an 'OpenDevice()' has been printed in the shell window. Please follow the directions below to safely deal with this and report the bad device and the last output of NSDQuery to the author of NSDPatch. Thanks. NSDPatch will help you to work around this problem and the default configuration should cover pretty much all known broken devices.

- Try 'NSDQuery <whichever.device>' on patched devices to check if everything was successful. If you encounter problems check Trouble Shooting and Custom Installation again. Please report all problems to the author with the device names, unit numbers, and versions that cause the problems!

- If you are not sure that a patch was successful, use the 'NSDQuery' tool as described above to check on that device.

- In the unlikely case that NSDPatch crashes your machine on the first invocation, you may want to

  - Temporarily take out any other system hacks and patches one by one to find out if one or more of them clash with NSDPatch.

  - try NSDPatch with an empty configuration file to verify basic installation compatibility.

  - reduce the configuration file to a single line, e.g., for 'parallel.device' to make testing easier.

  - check your system for any virus

  If you have problems removing NSDPatch from your startup when something crashes right away, boot up without using the 'Startup-Sequence', and type in manually 'SetPatch' and 'delete C:NSDPatch' at the command line.

- If multiple calls to NSDQuery with changing OS memory usage while a device is in memory reveal that the reported 'io_Device' value changes for every invocation, NSDPatch will not be able to patch this device for any openers that opened the device before NSDPatch was installed. This is not a bug in NSDPatch.

- The NSDPatch 64 bit disk emulation capabilities or the 'FIXSCSIUPDATE' option may not work as expected on an IDE 'scsi.device' that has been patched by 'atapi.device'. This is not a bug in NSDPatch.

1.10 README.guide/New Patch Lines

```
Adding New Patch Lines
======================
```

1. The safe solution is emailing a copy of the device file in question to the author. He will generate a suitable patch line to be placed into 'NSDPatch.cfg'. The quick and dirty solution is outlined below.

2. First make sure that everything important is saved because the tests for what you need may crash the system. If possible, run Enforcer.

3. Try NSDQuery on the device. If your Amiga crashes or severly

misbehaves, you will need the 'IOERRNOCMD' option! If you get a report that the device is NSD, you don't need to patch it.

4. Check 'NSDPatch.cfg' for an existing similar line and make a copy, changing the name appropriately. If you have, e.g., 'magicscsi.device', you may want to use the line for 'scsi.device' as first approximation and put the name 'magicscsi.device' into the copied line. For, e.g., 'mynewaudio.device', you would take the line for 'audio.device' as example.

5. Put the new line with the device's name into 'NSDPatch.cfg'. If needed according to 3., add the 'IOERRNOCMD' option if it isn't already there.

6. Start NSDPatch and try NSDQuery again. Now, the device should respond as NSD device, based on the new config line.

7. Never change too many patch options at once. After changing a patch option, you must reboot for it to take effect. Read the descriptions in 'NSDPatch.cfg' before using options.

8. In case of problems, contact the address listed in Support.

9. If the patch line works well and solves your problems, please forward it to the address listed in Support.

1.11 README.guide/Tutorial

Tutorial
****

The basic patch options are pretty well explained in 'NSDPatch.cfg' with lots of examples. Also, there are examples for the RDB late mount functionality in that file. So a tutorial for NSDPatch at this point of time really remains To Be Defined.

1.12 README.guide/Support

Support
****

If you have any comments on NSDPatch, please send them to the address listed below. In case of problems, please include the output of 'SHOWCONFIG DEBUG' with your mail or email.

Heinz Wrobel
Karlstr.16
82131 Gauting
Germany

<heinz@hwg.muc.de>

1.13 README.guide/FAQ

```
Common problems, Frequently Asked Questions
*******************************************

Common Problems
===============

   Whatever you problems are, a general way to approach them is
outlined in Trouble Shooting in the Installation chapter.

Implementation
==============

   This may not be very interesting to most readers as it hints at some
technical details. NSDPatch hooks itself into the exec.library
OpenDevice() function. It will patch any opened device that should be
patched and "preprocess" requests sent to it this way. The user
software sees an NSD device then and the device itself will not have to
be reworked in strange device specific ways. Devices patched like this
can still be expunged and reloaded freely. The permanent NSDPatch
footprint in memory is about 5KB plus memory for configuration data for
each device to be patched. NSDPatch does not need to be started in the
background and it does not use any ugly SegList splitting hacks or
similar stunts. NSDPatch should also be compatible to all debugging
tools and all 680x0 CPU's.
```

1.14 README.guide/Big Disk Support

```
Support for drives and partitions exceeding 32 bit
**************************************************

   NSDPatch can also be used to help with partitions and disks >4GB by
emulating the needed 64 bit commands that are part of the NSD
specification. How would you set this up? Here is an example of how I
did it for a 6.4MB IDE drive. Read it completely first!  Familiarity
with your User's Manuals, the AmigaShell, and HDToolBox is assumed.

   · Install NSDPatch.

   · Hook up your >4GB disk to your computer and partition the first
     4GB as described in the user's manuals of your Amiga or Harddisk
     Controller. As long as this doesn't work, nobody will be able to
     help you with space >4GB. For the further discussion I will refer
     to the device name of your controller as 'scsi.device'. Fill in
     what it really is on your machine.

   · Making a backup of any data on this big drive is now a very good
```

idea.

- Put at least V43 FFS into the RDB with HDToolBox and then continue to set up partitions exceeding the 4GB range. Note that HDToolBox will display all sorts of strange or negative byte and size values for these partitions. This can be safely ignored as the partitioning will work anyway. Make sure that the DOS types set up actually match what you have set up when putting V43 FFS into the RDB. If they don't match, you get V40 FFS from ROM on these partitions which will abysmally fail and most likely trash data at least on your first partition. Always use at least V43 FFS because it knows how to do things right for large disks. You can use 'VERSION <device:>' to check the FFS version for a mounted partition.

- Use NSDQuery on your scsi.device unit with the large disk. It should report the four 'NSCMD_TD_xx64' commands among other things. If it doesn't, check your 'NSDPatch.cfg' if the entry for your scsi.device is available and not in comments and/or contact the author of NSDPatch.

- Reboot. When NSDPatch is started right after SetPatch, any partitions beyond the 4GB range will automagically be activated.

- Always, always use the 'QUICK' option when formatting a partition exceeding the 4GB range with the V40 'FORMAT' command. The V40 'FORMAT' command does not know how to do things right otherwise and will trash your disk!

1.15 README.guide/To Do

My TODO list
****

Currently, I really have no special TO DO list other than collecting more configuration lines for NSDPatch.cfg. Let me know if you have any special wishes.

1.16 README.guide/History

History
****

- 43.21
  - NSDPatch will now internally sort the patches always in the correct order to make sure that the most precise patch specification is always used first if applicable.

  - I disabled CONSOLEHACK functionality for now. This needs more work.

· 43.20
  - Took out AVOIDFORBID for console.device for now. This needs more research. Also, the majik console.device workaround code will now only activate if you also specify CONSOLEHACK. Until I know why this code is not reliable, especially on the Draco, it will not be easily user visible. Too dangerous. NSDPatch should be dead reliable and I try to keep it that way.

· 43.19
  - MOUNTANY is now obsolete. Late mount functionality has better error messages now and will always process all entries matching the pattern given via the ACTIVATE option.

  - Rearranged 3rd party device entries in NSDPatch.cfg for clarity.

  - The patch always needs less than 64 bytes stack now during OpenDevice(). This is not expected to change in the future.

  - AVOIDFORBID handling should be safer now. Semaphore single threading is done device specific to avoid dead locks.

  - The RDB code could stop inadvertently after checking one partition or file system without looking at any more. Should be fixed now.

  - The automagic big disk partition activation really activated about anything that walks. Should be fixed now.

  - Some text and NSDPatch.cfg improvements.

  - All new README.guide.

· 43.18
  - For any patched trackdisk like device, NSDPatch will now try to automagically activate any mounted but yet unactivated partitions. This is for better support of large partitions where you don’t have to run NSDPatch with an ACTIVATE option anymore then. To suppress this feature, the new SKIPMOUNT option is used. Also, there is a new heavy majik feature to single thread OpenDevice() calls without Forbid(). If this works as expected, e.g., console.device should no longer hog the system during opening large shell windows.

· 43.17
  - As it turns out, there are devices that fail to open, do not set io_Device and return with no error. NSDPatch "fixes" this now to avoid crashes and returns IOERR_OPENFAIL in that case.

· 43.16
  - Even more patch lines added. Now a new option INTBEGINIO is available to make NSDPatch really safe for devices where BeginIO() may be called from within interrupt code. This affects the patch lines for audio.device and timer.device. The RDB late mount functionality will no longer complain

about already existing DOS mounts if MOUNTANY has been specified. This makes using patterns easier.

· 43.15
  - The version/revision match code ignored general patch requests. This made it impossible to patch, e.g., an ISNSD device with FIXSCSIUPDATE.

· 43.14
  - Cleanup for FIXSCSIUPDATE option. Will ignore an error on internal handling now just like scsi.device. This keeps FORMAT happy.

· 43.13
  - Cleaned up some of the 64 bit emulation code. A few SCSI CDB's are set up better now. Added FIXSCSIUPDATE option. It will turn CMD_UPDATE for scsi.device into something that does no longer conflict with JAZ drive behaviour even though it should still work as expected for other devices. NSDPatch will use significantly less stack in its BeginIO patch for patched devices in the general case now.

· 43.12
  - NSDPatch no longer pops up requesters when activating DOS devices. Whenever you specify QUIET as option, NSDPatch will no longer complain about already mounted entries when using the RDB functionality. More updates to the config file. SINLEPATCHONLY is now the default option as it tends to be a lot more compatible to many debug tools commonly used with AmigaOS. To get the the old behaviour back where the device specific patch is reinstalled for each and every OpenDevice, you can use the new TRYMULTIPATCH option. Note that with other tools patching into the same vectors you may cause infinite loops when using this option!

· 43.11
  - Added MOUNTANY option for RDB support. This facilitates RDB mount magic by allowing access to any partition, not just partitions not marked as automount. It is meant for the advanced user. Added PATCHCONFIGLINE command line option. You can use NSDPatch with a single simple config line. It will not read in a config file then unless you also specify one. The default DEVS:NSDPatch.cfg will now only be read if no command line options have been used. Updated the configuration file a little.

· 43.10
  - Added SINGLEPATCHONLY option for device patch lines. With this option, a device will only patched for its initial open call. This fixes interaction with CMD, which will hang in an endless loop without SINGLEPATCHONLY. This again shows how dangerous the use of the exec.library SetFunction() call is.

· 43.9
  - Should work with >=V37 now.

  - Supports device mapping now. Implementation of this feature

has been funded by ATL, Inc.

· 43.8
- Reduced stack usage during OpenDevice() patch by about 32 bytes. The patch needs about 68 bytes now during OpenDevice().

- The SANA2MAGIC did not work very well. There was a problem in handling paths. So it often did not even get activated.

· 43.7
- There was a bug in the Version/Revision recognition. NSDPatch patched too much at times. This should not have hurt, though.

- Reduced processing time in OpenDevice() a little.

- Mixing different versions of NSDPatch should not cause any harm now.

- Added new PatchInfo option that tells you about NSDPatch activities. Note: This option is more of a debugging hack!

- More patch lines.

· 43.6
- Now supports LVO's that can't normally be SetFunction()'ed.

- Now supports version specific patches correctly. The override general patches.

- Now supports special SANA2MAGIC for callback handling of old devices.

- Now checks the default patch file DEVS:NSDPatch.cfg automatically, if none has been specified.

- The config file is no longer named v40.nsdpatchcfg. It now already has the default name and also has a C= version string.