

Proseminar Linux 2002

Sound unter Linux

Johannes Tysiak
Betreuer: Jan Suchanek
Universität Ulm

Proseminar Linux 2002

Sound unter Linux

1. Geschichte	3
2. Sound unter Linux	3
3. Das Application Programming Interface	4
4. Sound Formate	4
5. Interfaces: Open Sound System	6
6. Interfaces: Advanced Linux Sound Architecture	7
7. OSS vs ALSA	8
8. Weitergehende Konzepte	9
8.1 Network Audio System	9
8.2 Analog Real-Time Synthesizer	9
8.3 Enlightened Sound Daemon	10
9. Professionelle Sound Software: Ardour	10
10. Zukunft	11
11. Appendix	12

1. Geschichte

Nachdem sich Linux immer mehr etablierte und eine immer größere Community hinter der Entwicklung stand, wurde es langsam Zeit für eine Unterstützung der Soundkarten. So gab es in der Version 1.0 des Kernel zum ersten Mal rudimentären Support für Sound Devices. Hannu Savolainen hatte ein Soundkonzept, welches zu diesem Zeitpunkt noch fest im monolithischen Kernel eingebaut war. Dies war der erste Ansatz des „Open Sound System“ (OSS), die API der Soundtreiber im Kernel 1.0 enthält dementsprechend auch schon Teile der späteren OSS API.

Aufgrund der Tatsache, dass viele Hersteller Daten zu Ihrer Soundkarte aus Angst vor Plagiaten nicht an die Open Source Community herausgaben, war OSS bald ins Stocken gekommen, was Support für neuere Karten betraf. So entschied sich Savolainen bald, die Entwicklung der freien Kernaltreiber abzugeben und fing an, mit seiner Firma 4Front an einer neuen Lösung zu arbeiten, die er OSS/Linux taufte (oft auch OSS/4Front genannt). Dies war ein kommerzielles Produkt, was es für ihn einfacher machte, an die entsprechenden Spezifikationen für neuere Karten zu kommen. Savolainen unterstützte jedoch auch weiterhin die Free Software Community, indem er einige seiner Treiber auch mit Quellcode veröffentlichte. Diese Treiber sind heute noch unter dem Namen OSS/Lite bekannt (oder auch OSS/Free).

Alan Cox (Entwickler bei Red Hat), Kernel Guru und Maintainer der 2.2.x Version des Kernels, wurde von Red Hat beauftragt, die Kernel Sound Treiber umzuschreiben und den Sound Support unter Linux zu verbessern. Dies resultierte in einem neuen Treiberset, welches komplett modular aufgebaut ist. Er extrahierte die gemeinsamen Teile des Treibercodes und entwickelte aus ihnen shared Module mit einer gemeinsamen API. Diese API ist fast identisch mit der von OSS/Linux, es fehlen jedoch einige Funktionen gegenüber der kommerziellen Version.

ALSA (Advanced Linux Sound Architecture) ist der neueste Ansatz, Soundtreiber für Linux zu entwickeln. Jaroslav Kysela sammelte einige Entwickler, um eine neue flexiblere API zu entwickeln, die vor allem die Potentiale seiner Lieblingskarte, der Gravis Ultrasound, ausschöpfen sollte. ALSA startete als ein alternativer Treiber für das Gravis Board, wurde jedoch später zu einer vollständigen Sound API, die eine Großzahl von Soundkarten, darunter auch viele neuere unterstützt.

ALSA soll ab dem Kernel 2.6.x zum Standard-Soundtreiber werden.

2. Sound unter Linux

Der Linux Kernel beinhaltet Gerätetreiber als Block oder Character Treiber. Ein Block Device beheimatet ein Filesystem, daher sind z.B. Treiber für IDE Festplatten oder Tape Drives Block Devices. Character Devices werden wie normale Dateien angesprochen, während das Character Device die eigentlichen System I/O Calls auf das gewünschte Gerät übernimmt. Treiber werden entweder in den Kernel kompiliert oder als Module dynamisch in den Kernel Space geladen. Wenn ein Modul in den Kernel Space geladen wurde sind seine Dienste genau wie jede andere Systemfunktion verfügbar. Anwendungen können diese Services benutzen, indem sie in die speziellen Dateien (hier Device Files) schreiben. Diese Device Files sind im `/dev` Verzeichnis zu finden, für Soundkarten sind insbesondere `/dev/dsp` und `/dev/audio` wichtig.

Gerätetreiber zu schreiben kann von simpel zu komplex variieren. Linux-Treiber für Soundkarten zu entwickeln ist eine relativ komplexe Aufgabe, da eine vollständige technische Dokumentation und Spezifikation der Sound Karte vorliegen muss. Glücklicherweise finden sich immer mehr

Hersteller mit einer offenen Einstellung zur Freigabe dieser Informationen. Der resultierende Code kann in `/usr/src/linux/drivers/sound` gefunden werden.

3. Das Application Programming Interface

Wie zuvor schon erwähnt greifen Anwendungen unter Linux nicht direkt auf die Hardware zu. Daher müssen Entwickler sich nicht damit beschäftigen, wie eine bestimmte Soundkarte angesprochen werden muss. Anstattdessen bietet ein „Application Programming Interface“ (API) dem Entwickler die Möglichkeit, ein hardwareunabhängiges Kommandoset zu benutzen, um das Gerät anzusprechen.

Die OSS/Free API (zu finden unter `/usr/src/linux/include/linux/soundcard.h`) ist das Standard Programming Interface für Kernel Sound Module. Die OSS/Linux API (im Standardpfad der OSS/Linux Installation unter `/usr/lib/oss/soundcard.h` zu finden) bietet eine verbesserte und erweiterte Version.

ALSA hat eine größere Reichweite an Operationen als die OSS/Free API und wird oftmals als der Nachfolger von OSS/Free als Standard-Kerneltreiber betrachtet. Dementsprechend ist ALSA auch schon im neuesten Developer Kernel der 2.5.x Reihe zu finden. Der ALSA Treiber kann zudem mit einem OSS/Free Emulationsmodus kompiliert werden, was ALSA somit komplett kompatibel zur existierenden Kernel Soundtreiber API macht.

ALSA erlaubt eine größere Kontrolle des Soundgeräts, bleibt jedoch gleichzeitig hardwareunabhängig, was zur Programmierung von allgemein benutzbarer Software ein wichtiges Kriterium ist. ALSA bietet eine Menge Neuerungen, die vor allem Audioentwicklern im professionellen Bereich zugute kommt. Als Beispiel seien hier Multitrack Hard Disk Recorder unter Linux genannt. Gleichzeitig wurde die Möglichkeit für PCM plugins und transparenten Netzwerksound geschaffen. Die ALSA API ist im Verzeichnis `alsa-lib-x.x.x/include` zu finden (`asound.h`).

4. Sound Formate

Zum besseren Verständnis der Welt des Sounds hier zunächst einmal ein paar grundsätzliche Worte zum Thema Audioformate.

MIDI:

MIDI steht für Musical Instrument Device Interface. MIDI Dateien haben gewöhnlicherweise die Endung `.mid`. Sie enthalten Sequenzing Informationen, d.h. Informationen, wann welches Instrument auf welche Weise gespielt werden soll. Da das eigentliche Erzeugen des Sounds hierbei von der Hardware übernommen wird, und MIDI keine digitalisierte Analogwelle ist, hängt die Qualität sehr stark von der verwendeten Hardware ab.

Modules:

Modules (im Sinne von PC Sound) sind digitale Musikdateien, die eine bestimmte Anzahl von Samples und Sequenzing Informationen haben, mit der sie dem Abspielgerät sagen, wann welches Sample (oder auch Instrument) auf welchem Track mit welcher Höhe gespielt werden soll, optional mit welchem Effekt, wie z.B. Vibrato.

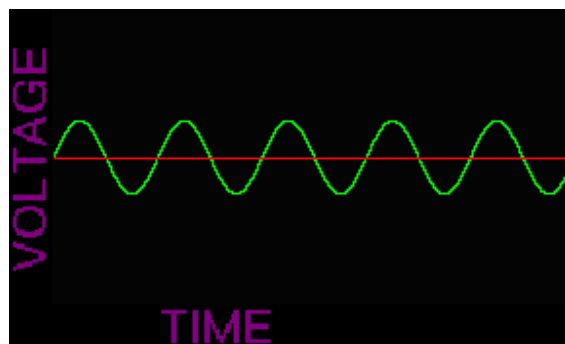
Ein Vorteil gegenüber MIDI ist, dass Modules jegliche Art von Sound (z.B. auch menschliche Stimmen) beinhalten können. Ein weiterer besteht darin, dass der Sound auf jeder Plattform gleich klingt, da die Samples im Modul sind. Ein großer Nachteil ist der dadurch resultierende Größenunterschied im Vergleich zu MIDI. Hinzu kommt, dass es für Modules keinen wirklichen Standard gibt. Modules entstammen ursprünglich dem Amiga.

Das meistbenutzte Format hat die Erweiterung .mod. Es gibt jedoch eine Menge andere Erweiterungen, abhängig vom entsprechenden Format.

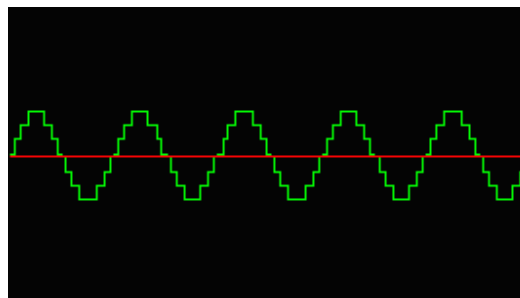
WAV:

WAV (oder auch RIFF WAV) ist der von Microsoft für Windows 3.11 eingeführte Standard um PCM (Pulse Code Modulation) encodete Audiosignale zu speichern. Ähnliche Formate sind AIFF, SND, oder AIFC, welche alle dem PCM Standard ähnliche Verfahren benutzen. PCM beschreibt Samples abhängig von ihrer Wellenform in einer bestimmten Zeitperiode.

Ein analoges Signal wird aufgenommen und digitalisiert durch AD-Wandler (Analog-Digital Wandler). Der digitale Wert (eine Lautstärke zu einer bestimmten Zeit), den der AD-Wandler übergibt kann weiter verarbeitet werden. Hier ein Bild einer perfekten Sinuswelle:



Das nächste Bild zeigt eine digitale Repräsentation dieser Welle:



Wie leicht zu erkennen ist, hängt die Qualität des Digitalen Audiosignals stark ab von der Zeit (Aufnahmerate) und Auflösung der Voltstärke. Das so erlangte digitale Signal kann zurück zu einer analogen Repräsentation umgewandelt werden durch einen entsprechenden DA-Wandler (Digital-Analog Wandler).

Andere Standards:

Zu Zeiten der digitalen Vernetzung müssen auch Audiodaten komprimiert werden. Um dies zu erreichen wurden Kompressionsverfahren wie z.B. MPEG, Real Audio, WMV (Windows Media

Player) oder auch Ogg Vorbis entwickelt. Auf diese soll an dieser Stelle jedoch nicht näher eingegangen werden.

5. Interfaces: Open Sound System

`/dev/mixer`

Mit diesem Interface kann eine Anwendung auf die eingebauten Mixer-Systeme der Soundkarte Zugriff nehmen. Ein Mixer macht es möglich, die Lautstärke verschiedener Quellen für Aufnahme und Wiedergabe zu regeln. Desweiteren wird dieses Device File dafür benutzt, Aufnahmequellen auszuwählen. Der OSS Treiber unterstützt mehrere Mixer in einem System. Die Mixer-Devices werden dementsprechend `/dev/mixer0`, `/dev/mixer1`, usw. genannt. `/dev/mixer` ist ein symbolischer Link auf einen dieser Mixer.

`/dev/sndstat`

Dieser Device File ist für Diagnosezwecke vorhanden. Er gibt direkt lesbare Rückmeldungen über die von OSS gefundenen Ports und Geräte. Durch das Kommando „`cat /dev/sndstat`“ können hilfreiche Informationen über die Treiberkonfiguration eingesehen werden.

`/dev/dsp` und `/dev/audio`

Dieses sind die wichtigsten Device Files für Soundanwendungen. Die auf dieses Device geschriebenen Daten werden mit dem DAC/PCM/DSP-Chip der Soundkarte abgespielt. Durch Lesen des Device Files werden Daten der derzeitigen Aufnahmequelle zurückgegeben.

Die Devices `/dev/audio` und `/dev/dsp` sind sehr ähnlich. Der Unterschied besteht in der Kodierung der PCM Samples. `/dev/audio` ist für die Kompatibilität zu Sun wichtig. Die Namensgebung ist ähnlich zu `/dev/mixer`.

Aufgrund des Aufbaus dieser Device Files können mittels `cat` Audiodaten direkt auf die Soundkarte geschrieben und somit ausgegeben werden.

Beispiele:

```
$ cat sample.voc > /dev/dsp      (Soundblaster-Files .voc)
$ cat sample.wav > /dev/dsp      (PCM-Files .wav)
$ cat sample.au > /dev/audio     (Sun-kompatible Files .au)
```

`/dev/sequenzer` und `/dev/music`

Mit `/dev/sequenzer` kann der Sequenzer auf der Soundkarte angesprochen werden. Dies wird z.B. bei Soundeffekten in Spielen benutzt. Hierbei wird Zugang zu den Synthesizer-Geräten auf der Soundkarte gewährt. Zusätzlich können mit diesem Device File externe Synthesizer angesprochen werden, die z.B. über MIDI-ports und den entsprechenden General MIDI Standard synchronisiert werden. `/dev/music` ist dem sehr ähnlich. Der Unterschied liegt darin, dass dieses Interface sowohl Synthesizer als auch MIDI-Geräte auf dieselbe Art anspricht, damit Software geschrieben werden kann, die vom Gerät selbst abstrahiert.

`/dev/midi`

Dieses Device File sendet MIDI Daten roh und direkt auf den MIDI-Bus der Soundkarte. Dies wird vor allem für MIDI SysEx Anwendungen benutzt.

`/dev/dmfm`

Ein direktes Interface zu den FM Synthesizern der Karte. Im FM Synthesizer werden z.B. MIDI-Informationen zu Klängen synthetisiert, indem die Klangwelle moduliert wird. Daraus resultiert naturgemäß bei weitem schlechterer Sound als bei professionellen Synthesizern.

`/dev/dmmidi`

Direkter Zugriff auf den MIDI-Bus der Karte, besonders für spezialisierte Anwendungen.

6. Interfaces: Advanced Linux Sound Architecture

`/proc/asound`

Dies ist ein Informationsinterface. Hier können verschiedene Abfragen zur Soundkarte und den Treibern gemacht werden.

Beispiel:

```
bash$ cat /proc/asound/cards
0 [card1          : SB16 - Sound Blaster 16
                          Sound Blaster 16 at 0x220, irq 5, dma 1&5
```

`/dev/snd/controlCX`

Über dieses Interface kann Zugriff auf bestimmte Parameter der Karte gewährleistet werden. Dies ist insbesondere ein Vorteil für neuere Soundkarten mit einer Vielzahl an eigenen Effekten / Befehlen.

`/dev/snd/pcmCXDX`

Dieses Device ist für die Ausgabe von rohen PCM Daten gedacht. Es ähnelt dem `/dev/dsp` der OSS-Treiber. Hierbei ist folgende Nummerierung vorgesehen: `/dev/pcmC0D0` ist das erste PCM-Device auf der ersten Karte, `/dev/pcmC0D1` dementsprechend das zweite PCM-Device auf der ersten Karte und so weiter. Diese Nummerierung gilt auch für die weiteren Device Files.

`/dev/snd/mixerCXDX`

Dieses Interface bietet Zugriff auf die Mixerfunktionen der Soundkarte (siehe auch `/dev/mixer` bei OSS).

`/dev/snd/midiCXDX`

Hiermit kann Zugriff auf den MIDI-Bus der Karte erlangt werden, ähnlich dem `/dev/midi` bei den OSS-Treibern.

`/dev/snd/seq`

Sequenzier-Interface: Hiermit kann der Sequenzer der Soundkarte angesprochen werden.

`/dev/snd/timer`

Dieses Interface benutzt den Timer der Soundkarte, was insbesondere bei neueren Soundkarten zur Synchronisation mehrerer PCM-Devices und der entsprechenden Sequenzer und MIDI-Devices wichtig ist.

7. OSS vs ALSA

Irgendwann kommt jeder Linux User mit dem Bestreben nach Klang zu der Frage der richtigen Treiber. Oftmals wird OSS gewählt, durch die Implementierung im aktuellen Kernel. Die Frage welcher Treiber welche Vorteile bietet bleibt oft unbeachtet.

OSS ist bis heute ein bewährtes System, basierend auf dem ersten Treiber für den Soundblaster. Die meiste Software unter Linux benutzt bis heute die OSS API, die in den OSS/Free bzw. OSS/Linux Treibern nativ eingebunden ist. Die meisten OSS Treiber bieten im Gegensatz zu ALSA eine bei weitem einfachere Installationsroutine. Viele Linux-Distributionen haben eine automatische Erkennung der Soundkarte und installieren daraufhin die entsprechenden OSS Treiber. Insbesondere sei hier auch die Bereitstellung der kommerziellen OSS/Linux Treiber in die SuSE Distribution erwähnt, als ALSA noch in den Kinderschuhen steckte.

Der Nachteil der OSS Treiber ist jedoch genau der, dass die API eben größtenteils nur eine Erweiterung der ursprünglichen API aus den Zeiten des Soundblasters ist. Dies kann heutzutage nicht mehr genügen, manche Soundkarten haben CPUs mit derselben Mächtigkeit wie PCs zum Zeitalter des Soundblasters. Dementsprechend wird die API vor allem in den OSS/Linux Treibern ständig erweitert, Befehle sind jedoch oftmals abhängig von der verwendeten Hardware.

ALSA Treiber haben zwar momentan noch teilweise einen höheren Installationsaufwand, bieten auf der anderen Seite aber eine weit ausgereifere API. ALSA reizt selbst die neu ins Projekt aufgenommenen Soundkarten völlig aus. Die API steht trotz der noch nicht existierenden Version 1.0 des Projektes fast vollständig. Den Nachteil der großen Verbreitung der OSS API beseitigt ALSA durch den OSS Emulationsmodus, in dem vollständige Rückwärtskompatibilität zu den OSS Treibern geboten wird.

Der größte Vorteil der ALSA Treiber liegt in ihrem modularen Aufbau. So ist ein Treiber keineswegs ein monolithisches Stück Code, sondern ein Konstrukt anderer Treiber. Das Einbeziehen neuer Hardware ist einfacher als bei OSS, da in großen Teilen fertiger Code benutzt werden kann, und nur noch auf hardwarespezifische Funktionen dieser Karte eingegangen werden muss.

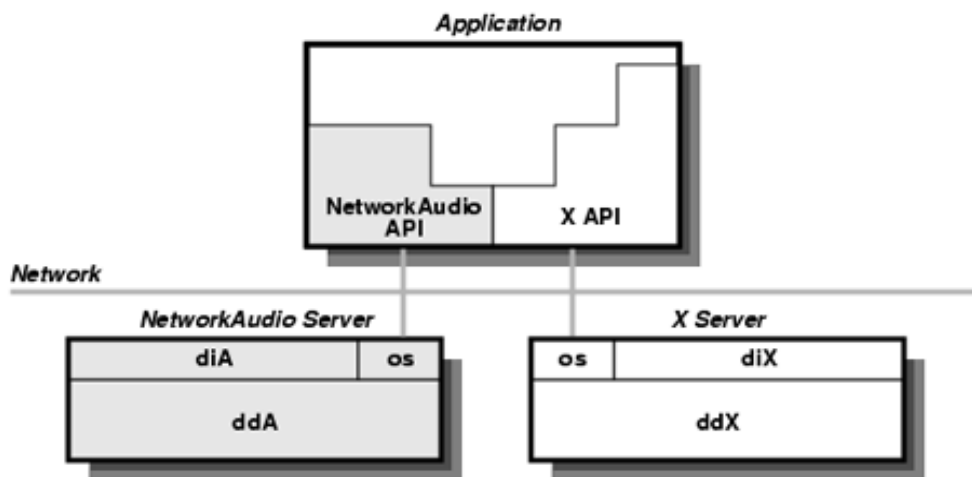
ALSA hat eine sogenannte „Soundcard Matrix“, in der der User seine Karte suchen kann. Die Chipsätze der Karte werden aufgelistet und die nötigen Treiber sofort angegeben. Anhand dieser Matrix ist auch leicht zu sehen, falls eine Soundkarte bisher noch nicht unterstützt wird, wie wahrscheinlich es ist, dass sie bald unterstützt wird. Gestützt wird diese Aussage auf das Layout der Karte und die verwendeten Hardwarebausteine. (ALSA Soundcard Matrix: <http://www.alsa-project.org/~goemon/>)

Im direkten Vergleich ist ALSA daher für die meisten Soundkarten die bessere Wahl, vor allem wenn die Karte bis an ihre Grenzen ausgereizt werden soll. Dies ist sicher auch der Grund, weswegen im Developer Kernel 2.5.x ALSA die Standard Sound Treiber sind.

8. Weitergehende Konzepte

8. 1. Network Audio System

Das „Network Audio System“ (NAS) wurde von Network Computing Devices (NCD) entwickelt, um Audiodaten über ein Netzwerk abspielen, aufnehmen und manipulieren zu können. Wie das X Window System benutzt NAS das Client / Server Modell, um Anwendungen von Soundkartentreibern abzukapseln.



Aus dem Whitepaper von NAS:

Network Audio bietet einen mächtigen, gleichzeitig aber simplen Ansatz, um Audiodaten zwischen Anwendungen und X Terminals, PCs und Workstations auszutauschen. Anwendungen bestimmen, wie mehrere Ein- und Ausgaben miteinander verstrickt werden, der Server routet die Daten dann automatisch zwischen den Komponenten und konvertiert gleichzeitig Sampleraten und Datenformate.

Sounddaten können auf dem Server zwischengespeichert werden und mehrere Male wieder benutzt werden, oder sie können direkt zum angeschlossenen Ausgabegerät gesendet werden. Anwendungen können dynamisch die Lautstärke ändern. Eingabegeräte, wie z.B. Mikrophone können benutzt werden um Audiodaten aufzunehmen. Anwendungen können diese Daten über ein Netzwerk zurücklesen, die Resultate auf dem Server für spätere Wiederverwendung speichern, oder sie an ein Ausgabegerät weiterleiten.

Mit NAS können X Entwickler die heutzutage für Anwendungen benötigte Soundunterstützung mit dem in Zukunft immer größer werdenden Bedarf an Flexibilität durch vernetzte Rechner verbinden.

8. 2. Analog Real-Time Synthesizer

Der Analog Real-Time Synthesizer (aRts) ist ein modulares System, um analogen Sound und Musik (insbesondere MIDI) auf einem digitalen Computer zu synthetisieren. Indem er Bausteine namens Module benutzt, kann der Benutzer sich komplexe Audioverarbeitungsprogramme bauen.

Diese Module bieten Funktionen wie z.B. Wellengeneratoren, Filter, Effekte, Mixer und Wiedergabe digitaler Audiodaten in verschiedenen Formaten. Der dazugehörige Sound Server (artsd) mixt Audiodaten von mehreren Quellen in Echtzeit, und erlaubt somit mehreren Anwendungen gleichzeitig, transparent die Soundhardware zu benutzen. aRts bietet zudem Netzwerkfähigkeit, um Sound über ein Netzwerk abzuspielen.

Als ein Kern der KDE 2 Desktop Umgebung bietet aRts die Basis für die KDE Multimedia Architektur und wird zukünftig weitere Medientypen wie z.B. Video unterstützen. Wie KDE läuft auch aRts auf einer Großzahl von Betriebssystemen, wie z.B. Linux und den BSD Varianten. Es kann jedoch auch unabhängig von KDE betrieben werden.

8. 3. Enlightened Sound Daemon

Der Enlightened Sound Daemon (ESD) ist ein aRts ähnlicher Ansatz, jedoch bei weitem weniger entwickelt. ESD bietet die Möglichkeit mehreren Anwendungen Zugriff auf ein Soundgerät zu gewähren. Er bietet wie aRts Netzwerktransparenz, d.h. es ist auch möglich, Sound über ein Netzwerk weiterzugeben.

ESD bietet zudem die Möglichkeit, bestimmte Soundevents zu definieren und diese für schnelleren Zugriff zu cachen und dann über eine dazugehörige ID abzurufen und abzuspielen.

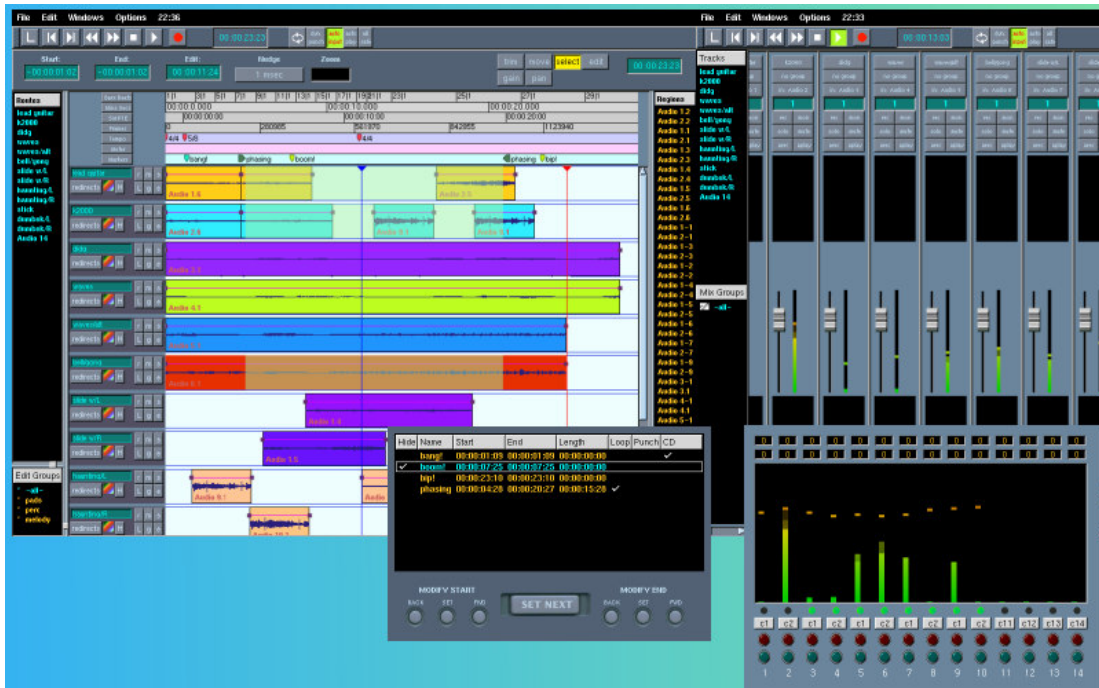
9. Professionelle Sound Software: Ardour

Um neben den vielen Applikationen die für normale Anwender existieren, welche Sound nebenbei als Effekt benutzen (Spiele, Videos u.v.a.), und reiner Abspielsoftware (mpg123, xmms, u.v.a.) ein Beispiel für professioneller Soundsoftware anzuführen sei hier „Ardour“ genannt.

Ardour ist als softwareseitiger Ersatz zu professioneller Aufnahme-Hardware wie Alesis ADAT oder Tascam MDM, also digitalen Harddiskrecordern und digitalen Mixern, gedacht. Auf Windows bzw. Mac gibt es ähnliche Software, wie z.B. ProTools, Nuendo, Sonar oder Logic Audio und nicht zu vergessen Cubase. Ardour plant, die Menge an Möglichkeiten der vorhandenen kommerziellen Software durch ein Open Source Projekt auf einem stabilen System zu reproduzieren.

Obwohl das Projekt momentan noch weit von einem Release Stadium entfernt ist, ist die Menge an Features jetzt schon beeindruckend. Nennenswert ist an dieser Stelle z.B. die Möglichkeit, Sound in 24bit bei einer Wiederholrate von 96kHz aufzunehmen, was selbst manche professionell eingesetzte Software wie Cubase erst seit kurzem kann.

Ardour profitiert vor allem durch die Verwendung der ALSA Treiber als Grundlage für ihre Software. Jedes unter ALSA benutzbare Soundgerät ist unter Ardour bis zu seinen physikalischen Grenzen zu benutzen. Insbesondere muss hier die professionelle Soundkarte RME Hammerfall genannt werden, für die unter ALSA Treiber zur Verfügung stehen. Diese Karte war eine der ersten Karten aus dem professionellen Bereich, bei der die Verwendung unter Linux möglich wurde.



Das graphische Interface der Software ist den existierenden professionellen Lösungen nachempfunden, und soll daher einen Umstieg auf die freie Software vereinfachen. Ardour wird ein großer Schritt in Richtung professioneller Sound unter Linux sein.

10. Zukunft

Die Zukunft für Linux Sound Treiber sieht durchaus nicht schlecht aus. Hersteller von Soundkarten entdecken Linux als stabiles System, dass durchaus für populäre Soundkarten auf einer Desktop-Lösung geeignet ist, während Hersteller professioneller Soundsysteme die Performance und die niedrige Latenzzeit beeindruckt, die durch einen simplen Kernelpatch zu erreichen ist.

Auf der anderen Seite gibt es auch den immer größer werdenden Bedarf an Spielen unter Linux. Um mit anderen Betriebssystemen auch in dieser Hinsicht konkurrenzfähig zu bleiben, darf natürlich auch die Unterstützung von Sound nicht fehlen. Kommerzielle Spieleanbieter (wie z.B. Loki Software) werden auch in den kommenden Jahren die Linuxcommunity vergrößern, diese neuen User werden mehr und besseren Support für ihre Soundkarten fordern.

Komischerweise beisst sich in dem Szenario, unter welchem Treiberentwickler unter Linux zu leiden haben, die Katze selbst in den Schwanz. Hersteller sehen sich nicht bereit, Linux Treiber oder ausführliche Dokumentationen und Spezifikationen vorzulegen, bis professionellere Software für Ihre Karten existiert. Auf der anderen Seite macht das Schreiben solcher Software ohne die entsprechende Hardware natürlich wenig Sinn. Diese Situation verbessert sich jedoch in letzter Zeit immer mehr durch einige wenige „Wellenbrecher“ wie z.B. die Software Ardour oder die RME Hammerfall.

Zusammenfassend lässt sich sagen, dass Sound Treiber unter Linux einer steten Entwicklung unterzogen sind, die seit dem ersten Erscheinen der Soundblaster Treiber im Jahre 1992 rapide zunimmt. Bald wird es Support für immer mehr Soundkarten, professionelle wie semiprofessionelle geben, die Installationsroutinen werden einfacher und im Softwarebereich wird sich einiges tun. Somit lässt sich abschliessend nur noch sagen: „Es klingt immer besser.“

11. Appendix

Quellen:

ALSA Project	http://www.alsa-project.org/
OSS/Linux	http://www.4front-tech.com/
OSS/Free	http://www.linux.org.uk/OSS/
NAS	http://radscan.com/nas.html
aRts	http://www.arts-project.org/
ESD	http://www.tux.org/~ricdude/EsounD.html
Ardour	http://ardour.sourceforge.net/