Sound

This page is a work in progress.
This page may thus be incomplete. Its content may be changed in the near future.

Sound is never noticed by users, until it isn't there. It forms an integral part of the user feedback experience. For example, the beeps produced by Windows Explorer, KDE and other desktop environments when users click on the wrong thing. It is also becoming a far more dominant part of the gaming and multimedia experience; with many games and movies now featuring 5.1 channel soundtracks. As such, it is becoming more important for hobby operating systems to support sound playback, at the very least.

An example of an older-style PCI sound card

Contents

A Short History of PC Sound

The original sound device on PCs was the PC Speaker, available as far back as the IBM PC (introduced in 1981). In the late 1980s several manufacturers started producing add-in sound devices, notably Creative (the Game Blaster) and AdLib (the AdLib Music Synthesizer Card). Of these, Creative was far more successful, and the Sound Blaster 16 became a de-facto standard for more than half a decade. Almost all cards today still have some backwards compatibility with the Sound Blaster 16.

In 1997 Intel specified a new standard, AC97 (short for *Audio Codec 1997*), that virtually replaced the Sound Blaster standard by specifying higher quality sampling. In 2004 Intel produced yet another standard, this time the Intel High Definition Audio standard (codenamed *Azalia*), which specifies yet another improvement on previous standards' audio quality.

Sound cards and standards for the professional (and audiophile) markets have evolved separately from the mainstream cards. The important feature in these cards is not fancy features, but very low latency (<5ms) and high quality sampling (24 bit sampling at 96KHz not being uncommon). MIDI is an important part of professional audio today, and has been since the early 1990s. Cards can be from 2 channels through to 32 and more.

Programming for Sound

Programming sound support into your operating system is not simply a case of writing a driver for a fairly generic card (such as the Sound Blaster 16) and letting your userspace applications deal with it; it is a case of putting together an architecture that will support sound generation, transport (to and from different devices, applications and even computers), mixing, output, and control, all the while retaining low latencies and high quality. In this, there are several key problems:

- Your sound transport architecture should be lossless, that is, 32 bit samples should be carried at 32 bits. Carrying these samples at 16 bits (for example) will lead to sound degradation, and is to be avoided.
- The same transport mechanism should also provide support for varied sample rates. 44100Hz is the standard for CDs, but DVDs carry sound at a sample rate of 48000Hz. Treating these two as the same will lead to very strange noises being emitted from the speakers. In addition, these will need conversion to a common sample rate before being output.
- Two applications **will** need access to the sound devices simultaneously. For example, notifications from an Instant Messaging program as well as the user's MP3 player. The operating system has the job of mixing these sound streams together and producing an intelligible result.
- Today's audio is not stereo, it can be any number of channels. Simpler applications still use mono sound, while DVD and High definition movies have 5.1 and 7.1 channel sound tracks. Your operating system will need to be able to route sound to any of these channels, and decide where to place mono and stereo sounds in the larger channel set. In addition, MIDI audio also needs to be routed correctly to and from devices and applications.
- Sound, like gaming graphics, must be completed in real time. Skipping audio caused by random hard disk or CPU usage is *very* noticeable. Unfortunately, simply increasing buffer size is not always the best solution, as this causes people listening to freshly-recorded audio ('monitoring' in the industry speak) to suffer delays.

Some (but not all) sound cards will have features allowing some of this processing to be done in hardware. Simple mixing and sample processing are the most common things to have dedicated hardware.

Some applications, such as games, will try to use the maximum number of channels possible. Older versions of Windows allowed applications to have 'exclusive' control over the sound hardware for low latency sound with guaranteed channels. However, since Windows Vista all sound (including volume) is software mixed supporting an unlimited number of channels.

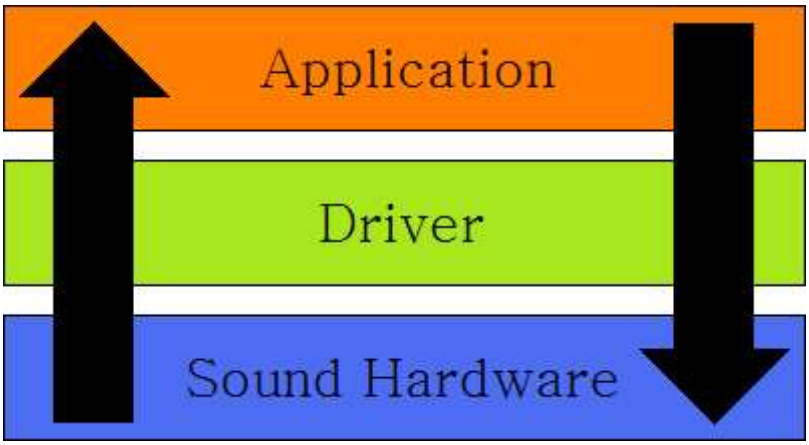
Software mixing all sounds has several advantages for very little CPU overhead:

- Easier to develop sound card drivers - all they must do is stream a buffer to the device rather than worry about channels, channel effects, channel volume, and what-not.
- Provides a consistent environment across hardware - a game will know it will not run out of channels and all channel effects (echo, reverb, etc) will be supported.

The Sound Architecture

This page is a work in progress.
This page may thus be incomplete. Its content may be changed in the near future.

The Simple Approach



This is the simplest of all architectures: a single application dealing with a single device through a single device driver. This would work well with a UNIX-based philosophy: read(), write() and ioctl() all have immediate and obvious effects. The application can read a number of samples from the sound device, process them (perhaps recording them to disk, adding a reverb, etc), and then write the resulting sound samples back out onto the device's speaker port.

The Ring/Hub Approach

A more complex example of an audio architecture is based around a ring concept. The ring is responsible for transport of audio signals around the system, and each module plugged on to the ring acts on the signal in some way.

Each application, input or output, mixer, filter, or plugin of some sort could be represented on the ring, and audio travels around being modified by some or all of the other ring modules. This has been successfully implemented in hardware on Creative's line of X-Fi sound cards; see here (http://www.pcper.com/article.php?aid=177&type=expert&pid=2) for more detail.

In a kernel this need not be implemented strictly as a ring: a hub architecture achieves roughly the same effect in a different manner.

ToDo: Describe additional methods, research and flesh out these ones more. See what is being used in today's kernels (information on how ALSA and co work is sketchy without reading the source).

Resources

- An overview of digital audio
- An overview of MIDI
- The ALSA project (http://www.alsa-project.org/) has documentation that may be useful in designing a sound system, as well as GPLed drivers for many different sound cards.