



Boomer

The New OpenSolaris Audio System

Garrett D'Amore
Staff Engineer
Sun Microsystems, Inc.

Boomer is...

- Modern audio subsystem for OpenSolaris
 - Surround sound (5.1 and 7.1)
 - Higher resolution (24-bit now, 32-bit later)
 - Higher sample rates (up to 192 kHz)
- Kernel software (mostly)
 - Framework (Mixer, format conversion, etc.)
 - Drivers (audiohd, audiopci, etc.)

Boomer is...

- Totally Open Source (nothing in usr/closed)
- Compatible
 - Legacy Sun API (audio(7I))
 - New Open Sound System APIs (dsp(7I))
 - Full support for legacy devices
- Performant
- Scalable
 - Sun Ray can have hundreds of DTUs

Boomer is *not*....

- MIDI
 - No wavetable synthesizer
 - No support for hardware MIDI devices
- A porting layer
 - Drivers use native OS services,
 - Porting takes effort
- A recording studio production facility
- A DSP facility
 - Latencies may be too high

Sun Solaris Audio Before Boomer

- Complex (very) mixer framework
 - STREAMS driven
 - Lots of dead code (multi-stream codecs, etc.)
 - Twisty maze designed to infuriate mere mortal driver developers
- No support for modern features
 - 16-bit 48kHz stereo only
 - Limited port selections
 - Not extensible

4Front Open Sound System: the Good

- Free and commercial
 - CDDL, BSD, and GPL versions
- Very portable
 - Ported to Solaris, Linux, and FreeBSD
 - Popular API (native for FreeBSD)
- Rich device support
 - Lots and lots of devices
 - Extensive features
 - Extensible mixer API

4Front OSS: the Bad

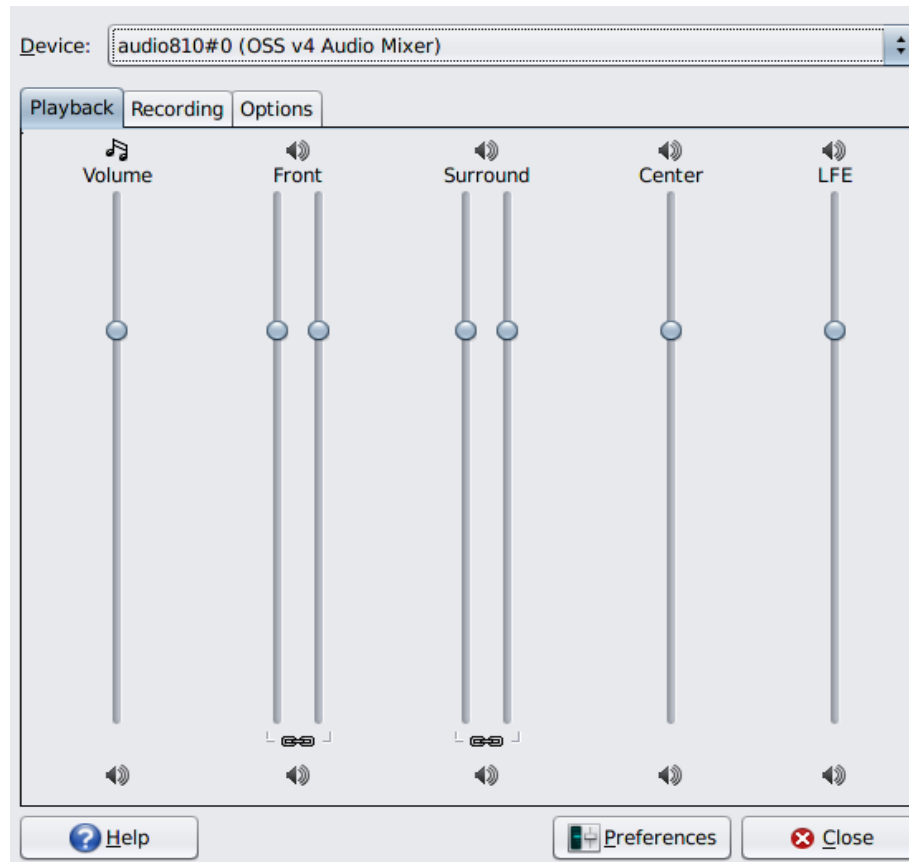
- Poor native support
 - Least common denominator porting layer
 - No support for MSI, suspend/resume, quiesce
 - Poor Sun audio(7I) emulation
- Scalability limitations
 - Not scalable to Sun Ray
 - Too many divides on hot code paths
- Too much work required from drivers
- ~~Separate vmix and remux drivers~~

Boomer: Best of Both

- Cherry picked code from Sun and 4Front
 - Hand-optimized version of GRC3 sample rate conversion from OSS
 - Hand optimized format conversion from OSS
 - Volume limiting code tuned from OSS
 - Legacy higher fidelity u-Law and A-Law conversions from Solaris
 - Drivers from both Sun and 4Front

○ Most of everything else written from scratch

Gratuitous Gnome Screenshot



Sun audio(7I) API: STREAMS

- Must support read(2), write(2)
 - “Preferred” way to move audio data
 - Special zero length write(2) semantic
- Must support putmsg(2), getmsg(2)
- Special ioctl(2) support
 - I_SETSIG used to enable SIGPOLL
 - Notification comes up /dev/audioctl file(s)
 - SIGPOLL via M_SETSIG STREAMS message
 - Other STREAMS semantics

Sun audio(7I): AUDIO_SETINFO

- Single structure for all audio configuration
 - Often incorrectly used (*realplayer*)
 - Read-modify-write breakages
- Not extensible
 - Single bit mask for all ports
 - Only 16-bit stereo PCM supported
 - No way to express “advanced features”
 - Jack retasking
 - SPDIF configuration

Sun audio(71): Mixer confusion

- /dev/audio vs. /dev/audioctl
 - One /dev/audio for write, one for read
 - One or more /dev/audioctl for device
- Mixer can have many audio nodes open
 - How do we know which audioctls correspond to which audios?
 - Answer: undocumented order of open(2)
- Some properties global, some local
 - ~~Port vs. Volume vs. Sample Rate~~

Sun audio(71): Boomer

- One “virtual” dev per real dev per process
 - One file recording, one playing back
 - Soft volume per application
 - No access to h/w settings (use OSS)
 - Monophonic gain control
- Nuke legacy mixer(71) & audio_support(71)
 - Used only be *sdtaudiocontrol* (EOF)
 - ~~Not supported on Sun Ray anyway~~

OSS API: /dev/dsp

- Originally designed for Linux, also used on FreeBSD
- Character device interface
 - read, write, poll, ioctl
 - Many ioctls
 - Optional mmap support (not in Boomer yet)
- “Fragment” based
 - Unit of latency in bytes
 - ~~Frequently (formerly had to be) power-of-~~

OSS: /dev/dsp concerns

- Not all commands equally supported
 - Read man page (dsp(7I)) for details
- Some devices use non-power-of-two fragment sizes
- Apps rely on device to “block”
 - Controls latency
 - Can't use deep buffers
- New OSSv4 commands not widely used

OSS: Enumeration API

- /dev/sndstat
 - Use read(2) to get human readable enumeration of devices
 - Use ioctls for programmatic interfaces
 - Some apps just parse text instead (ugh!)
 - /dev/mixer and /dev/sndstat are aliases
- Boomer separates this from mixer API
 - 4Front conflates them
 - Use real mixer API on real device node
 - Addresses security concerns

/dev/sndstat example

```
gdamore@tabasco{1}$ cat /dev/sndstat  
SunOS Audio Framework
```

Audio Devices:

```
0: audio810#0 NVIDIA AC'97, CK804 (DUPLEX)  
1: audiocmi#0 C-Media PCI Audio, CM8738-033 (DUPLEX)
```

Mixers:

```
0: audio810#0 NVIDIA AC'97, CK804  
    AC'97 codec: Avance Logic ALC655 (6 channels)  
1: audiocmi#0 C-Media PCI Audio, CM8738-033
```

OSS: Mixer API

- Device control
 - Volume
 - Port configuration
 - Hardware settings
- Free Form ASCII
 - Sort of... some names are “predefined”
 - Arbitrarily extensible
- Typed values
- Only used by “special” applications

OSS Mixer API: Cross Device Security

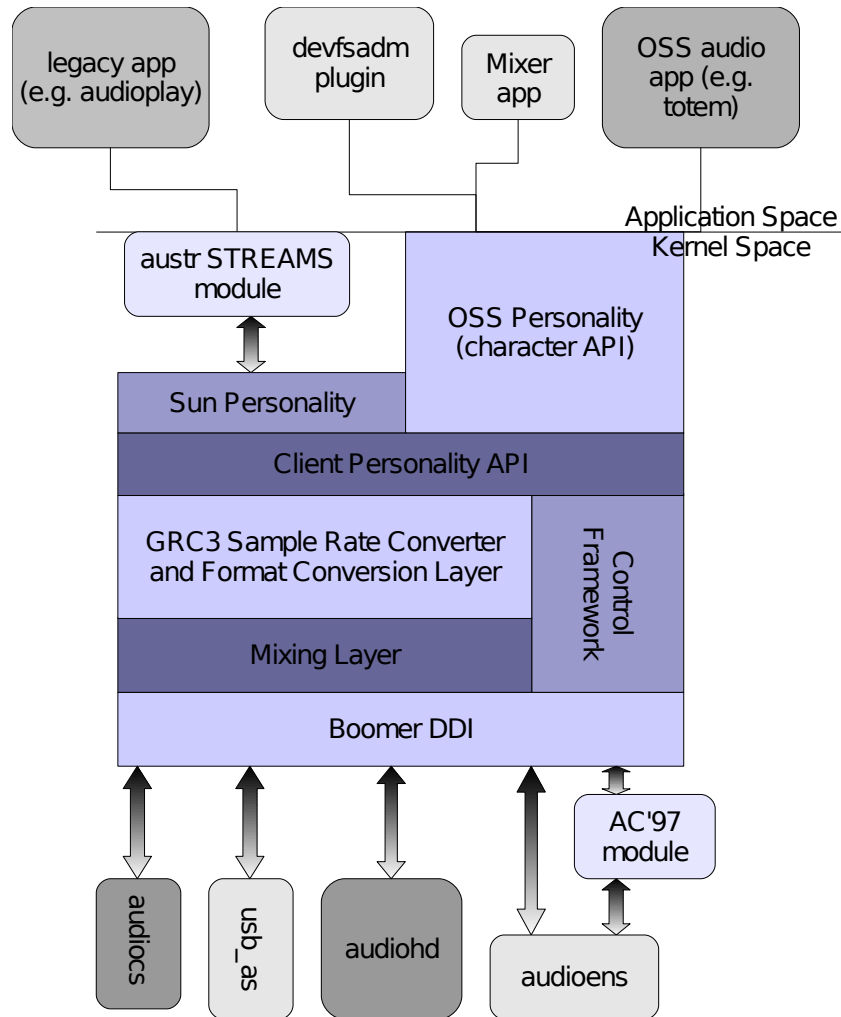
- 4Front uses a single `/dev/mixer` node
 - Identify actual devices by index
 - Single access guard on all devices (at `open(2)`)
- Boomer
 - Separate enumeration and device operations
 - via `/dev/mixer` or `/dev/sndstat`
 - Device operations permitted on device node



● e.g. `/dev/mixer4`,

`/dev/sound/audioens:0mixer`

Boomer Architecture



Boomer Audio Core

- Single kernel module
 - drv/audio
 - Ignore the austr behind the curtain
- API implementation
 - OSS, legacy Sun, client “personality” glue
- Sample rate & format conversion
- Mixing layer
- Control framework
- Underlying device driver interface

Boomer Data Quantities

- Fragments, frames, and samples, oh my!
 - One sample, usually 16-bits (2 bytes)
 - Can be 8, 16, or 24 bits in Boomer (32 later)
 - Frame consists of one or more samples
 - E.g. 7.1 surround sound has 8 samples per frame
 - Frame terminology is a Boomer “invention”
 - “Sample rate” (e.g. 48 kHz) is really “frame rate”
 - Fragment = chunk of frames processed

Client Personality Interface

- Internal “glue layer” between core and API
 - Narrow interface, not public (“audio_client.h”)
 - Shields API implementation from core details
- Designed to support present and future
 - Sun audio(7I)
 - OSS

Audio Buffer (FIFOs)

- Circular queue of fragments
 - Modeled after hardware DMA structures
 - Two per open file - one read, one write
 - Size is a whole number of fragments
- Consume and produce counters
 - 64-bit non-wrapping counters
 - Counts in frames
 - Updates usually in fragments, but not always
 - Index starts as modulo, but may be

austr STREAMS module

- STREAMS shim module
 - Not needed with push of PSARC 2009/380
 - Will be removed in a future update
- Uses LDI to provide reference counting
- Data path uses direct function calls for performance
- Actual implementation lives in audio core

Sample Rate Converter

- GRC3 licensed from 4Front
 - Constrained from 4Front implementation
 - Removed variable divides from hot code path
 - Replace with *fast* bit mask operations
- Operates on 24-bit data
 - Will be enhanced for 32-bit data later
 - Fixed point math
- Much faster than original Solaris or 4Front

Format Conversion

- Internally operates on 24-bit native-endian linear PCM
- ulaw and aLaw “decompression” from Sun
- Hand optimized conversions (other formats)
- Only simple channel conversions
 - Mono to stereo, “naive” down-mixing
 - Could be enhanced in the future

Mixing Layer

- What is “mixing”?
 - Multiple apps playing audio simultaneously
 - Solves “exclusive” audio device problem
- Challenges
 - Different formats, sample rates, etc.
 - Limits on hardware gain (distortion prevention)
 - Per application volume levels
 - Performance considerations

Boomer Mixing

- Uses 24-bit linear PCM (native) for now
- Per channel soft volume
- Master soft volume
 - for hw without gain control, also gain limiter
- Gain limiter licensed from 4Front
 - Effective and efficient
- Performant and scalable
 - Hundreds of applications playing audio

○ Always on – no vmix

Multichannel “remux”

- Devices present multich audio differently
 - Creative : pairs of stereo data
 - ICH, AudioHD: simple interleaving
 - Tadpole SPARCLE: reversed stereo (don't ask)
- Remux solves this elegantly
 - Boomer does this natively, with no extra cost
 - Can be expanded to support up- down-

Audio Device Driver Interface (DDI)

- Simplicity
 - Drivers not concerned with STREAMS
 - Drivers not concerned with character details
 - Typical AC'97 driver in ~ 1000 lines of C
- Easy to port from other OS'
 - Similar to 4Front , FreeBSD, Linux etc.
 - Takes about a week (or less) to do a driver port
 - For typical AC'97 h/w

Audio Control Framework

- Programmatic API for drivers to export control objects and operations
 - Some other frameworks/OS' call this “mixer API”
 - E.g. output port selection, surround gain settings, jack retasking, microphone boost, etc.
- Represented as free form name-value pairs
 - Some names “well-known” or stock values

AC'97

- Intel audio standard
 - provides registers, protocol from “audio chip” to codec, but not host device details
 - Like “mii”
- Boomer ac97 module simplifies drivers a lot
 - Just provide register read/write routines
 - Boomer provides control framework hooks

Future Directions

- Sun Ray Audio
- Virtualized audio
- lx_audio
- mmap support
- Digital audio, compressed audio
- Support for 32-bit audio
- Driver improvements
- More drivers
- Commit to the DDI

Sun Ray Audio

- Presently a separate shim
 - No mixing support
- Challenges
 - Scalability
 - Interface separation (no \$AUDIODEV)
 - Needs its own custom driver
 - Fixed sample rate may be problematic with low bandwidth networks

Virtualized audio

- Better support for hotplug
 - Default links may or may not be appropriate -
 - USB webcam shouldn't change *output*
 - USB headphones -- maybe -- needs preferences
- Secured zones (TX)
 - Today requires admins to start/stop apps to change audio perms
 - Some zones should be allowed to play, but not record

mmap support

- Easy to implement -1 day or so
- Testing a challenge
- Benefits
 - lower overhead for some audio
 - lower latencies
 - enabler for Jack “glitch free”
- Issues
 - 4Front discourages mmap due to restrictions
 - In OSS accesses device FIFO directly
 - In Boomer accesses client FIFO, no

Digital compressed audio

○ Several versions

- Dolby Digital (AC3)
- DTS Surround

○ Pass-thru

- lots of devices can do it
- requires “exclusive” use of device (no mixing)
- only useful for some apps (DVD players)

○ Live encoding

- allows mixing, access to single cable surround

More Drivers

- Creative devices
 - Audigy -- too many variants
 - X-Fi
- CMI devices
- VIA (“envy24”)
- Other less common ones
 - Yamaha
 - CS4280
 - etc.

Q&A

Project page:
<http://www.opensolaris.org/os/project/opensound/>