NAME

sndstat - nvlist-based PCM audio device enumeration interface

SYNOPSIS

To compile the driver into the kernel, place the following lines in the kernel configuration file:

```
device sound
```

DESCRIPTION

The ioctl interface provided by */dev/sndstat* device allows callers to enumerate PCM audio devices available for use. In other words, it provides means to get the list of all audio devices available to the system.

IOCTLS

For ioctl calls that take an argument, the following structure is used:

```
struct sndstioc_nv_arg {
        size_t nbytes;
        void *buf;
};
```

Here is an example of an nvlist object with explanations of the common fields:

```
dsps (NVLIST ARRAY): 1
  from_user (BOOL): FALSE
  nameunit (STRING): [pcm0]
  devnode (STRING): [dsp0]
  desc (STRING): [Generic (0x8086) (Analog Line-out)]
  pchan (NUMBER): 1 (1) (0x1)
  rchan (NUMBER): 0 (0) (0x0)
  info_play (NVLIST):
    min_rate (NUMBER): 48000 (48000) (0xbb80)
    max_rate (NUMBER): 48000 (48000) (0xbb80)
    formats (NUMBER): 16 (16) (0x10)
    min_chn (NUMBER): 2 (2) (0x2)
    max_chn (NUMBER): 2 (2) (0x2)
  provider_info (NVLIST):
    unit (NUMBER): 0 (0) (0x0)
    bitperfect (BOOL): FALSE
    pvchan (NUMBER): 1 (1) (0x1)
```

rvchan (NUMBER): 0 (0) (0x0)
provider (STRING): [sound(4)]
,

from_user Whether the PCM audio device node is created by in-kernel audio subsystem or userspace providers.

nameunit The device identification in the form of subsystem plus a unit number.

devnode The PCM audio device node relative path in devfs.

desc The descripton of the PCM audio device.

pchan The number of playback channels supported by hardware. This can be 0 if this PCM audio device does not support playback at all.

rchan The number of recording channels supported by hardware. This can be 0 if this PCM audio device does not support recording at all.

info_play Supported configurations in playback direction. This exists only if this PCM audio device supports playback. There are a number of name/value pairs inside this field:

min_rate Minimum supported sampling rate.

max_rate
Maximum supported sampling rate.

formats Supported sample formats.

min_chn Minimum supported number of channels in channel layout

max_chn
Maximum supported number of channels in channel layout

info_rec Supported configurations in recording direction. This exists only if this PCM audio device supports recording. There are a number of name/value pairs inside this field:

min_rate Minimum supported sampling rate.

max_rate

Maximum supported sampling rate.

formats Supported sample formats.

min_chn Minimum supported number of channels in channel layout

max_chn
Maximum supported number of channels in channel layout

provider_info Provider-specific fields. This field may not exist if the PCM audio device is not provided by in-kernel interface. This field will not exist if the provider field is an empty string.

provider A string specifying the provider of the PCm audio device.

The following ioctls are provided for use:

SNDSTIOC_REFRESH_DEVS Drop any previously fetched PCM audio devices list snapshots. This ioctl takes no arguments.

SNDSTIOC_GET_DEVS Generate and/or return PCM audio devices list snapshots to callers. This ioctl takes a pointer to *struct sndstioc_nv_arg* as the first and the only argument. Callers need to provide a sufficiently large buffer to hold a serialized nvlist. If there is no existing PCM audio device list snapshot available in the internal structure of the opened sndstat. *fd*, a new PCM audio device list snapshot will be automatically generated. Callers have to set *nbytes* to either 0 or the size of buffer provided. In case *nbytes* is 0, the buffer size required to hold a serialized nvlist stream of current snapshot will be returned in *nbytes*, and *buf* will be ignored. Otherwise, if the buffer is not sufficiently large, the ioctl returns success, and *nbytes* will be set to 0. If the buffer provided is sufficiently large, *nbytes* will be set to the size of the serialized nvlist written to the provided buffer. Once a PCM audio device list snapshot is returned to user-space successfully, the snapshot stored in the subsystem's internal structure of the given *fd* will be freed.

SNDSTIOC_ADD_USER_DEVS Add a list of PCM audio devices provided by callers to */dev/sndstat* device. This ioctl takes a pointer to *struct sndstioc_nv_arg* as the first and the only argument. Callers have to provide a buffer holding a serialized nvlist. *nbytes* should be

set to the length in bytes of the serialized nvlist. *buf* should be pointed to a buffer storing the serialized nvlist. Userspace-backed PCM audio device nodes should be listed inside the serialized nvlist.

SNDSTIOC_FLUSH_USER_DEVS Flush any PCM audio devices previously added by callers. This ioctl takes no arguments.

FILES

/dev/sndstat

EXAMPLES

The following code enumerates all available PCM audio devices:

```
#include <sys/types.h>
#include <err.h>
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/nv.h>
#include <sys/sndstat.h>
#include <sysexits.h>
#include <unistd.h>

int
main()
{
        int fd;
        struct sndstioc_nv_arg arg;
        const nvlist_t * const *di;
        size_t i, nitems;
        nvlist_t *nvl;

        /* Open sndstat node in read-only first */
        fd = open("/dev/sndstat", O_RDONLY);

        if (ioctl(fd, SNDSTIOC_REFRESH_DEVS, NULL))
                err(1, "ioctl(fd, SNDSTIOC_REFRESH_DEVS, NULL)");

        /* Get the size of snapshot, when nbytes = 0 */
```

```
		arg.nbytes = 0;
		arg.buf = NULL;
		if (ioctl(fd, SNDSTIOC_GET_DEVS, &arg))
			err(1, "ioctl(fd, SNDSTIOC_GET_DEVS, &arg)");

		/* Get snapshot data */
		arg.buf = malloc(arg.nbytes);
		if (arg.buf == NULL)
			err(EX_OSERR, "malloc");
		if (ioctl(fd, SNDSTIOC_GET_DEVS, &arg))
			err(1, "ioctl(fd, SNDSTIOC_GET_DEVS, &arg)");

		/* Deserialize the nvlist stream */
		nvl = nvlist_unpack(arg.buf, arg.nbytes, 0);
		free(arg.buf);

		/* Get DSPs array */
		di = nvlist_get_nvlist_array(nvl, SNDST_DSPS, &nitems);
		for (i = 0; i < nitems; i++) {
			const char *nameunit, *devnode, *desc;

			/*
			 * Examine each device nvlist item
			 */

			nameunit = nvlist_get_string(di[i], SNDST_DSPS_NAMEUNIT);
			devnode = nvlist_get_string(di[i], SNDST_DSPS_DEVNODE);
			desc = nvlist_get_string(di[i], SNDST_DSPS_DESC);
			printf("Name unit: ‘%s‘, Device node: ‘%s‘, Description: ‘%s‘0,
			   nameunit, devnode, desc);
		}

		nvlist_destroy(nvl);
		return (0);
	}
```

SEE ALSO

sound(4), nv(9)

HISTORY

The nvlist-based ioctls support for **sndstat** device first appeared in FreeBSD 13.0.

AUTHORS

This manual page was written by Ka Ho Ng <*khng@FreeBSD.org*>.