NAME

speaker, **spkr** - console speaker device driver

SYNOPSIS

device speaker
#include <dev/speaker/speaker.h>

DESCRIPTION

The speaker device driver allows applications to control the PC console speaker on an IBM-PC--compatible machine running FreeBSD.

Only one process may have this device open at any given time; open(2) and close(2) are used to lock and relinquish it. An attempt to open when another process has the device locked will return -1 with an EBUSY error indication. Writes to the device are interpreted as ‘play strings’ in a simple ASCII melody notation. An ioctl(2) request for tone generation at arbitrary frequencies is also supported.

Sound-generation does not monopolize the processor; in fact, the driver spends most of its time sleeping while the PC hardware is emitting tones. Other processes may emit beeps while the driver is running.

Applications may call ioctl(2) on a speaker file descriptor to control the speaker driver directly; definitions for the ioctl(2) interface are in <*dev/speaker/speaker.h*>. The tone_t structure used in these calls has two fields, specifying a frequency (in Hz) and a duration (in 1/100ths of a second). A frequency of zero is interpreted as a rest.

At present there are two such ioctl(2) calls. SPKRTONE accepts a pointer to a single tone structure as third argument and plays it. SPKRTUNE accepts a pointer to the first of an array of tone structures and plays them in continuous sequence; this array must be terminated by a final member with a zero duration.

The play-string language is modeled on the PLAY statement conventions of IBM Advanced BASIC 2.0. The MB, MF, and X primitives of PLAY are not useful in a timesharing environment and are omitted. The ‘octave-tracking’ feature and the slur mark are new.

There are 84 accessible notes numbered 1-84 in 7 octaves, each running from C to B, numbered 0-6; the scale is equal-tempered A440 and octave 3 starts with middle C. By default, the play function emits half-second notes with the last 1/16th second being ‘rest time’.

Play strings are interpreted left to right as a series of play command groups; letter case is ignored. Play command groups are as follows:

CDEFGAB Letters A through G cause the corresponding note to be played in the current octave. A note letter may optionally be followed by an "*accidental sign*", one of # + or -; the first two of these cause it to be sharped one half-tone, the last causes it to be flatted one half-tone. It may also be followed by a time value number and by sustain dots (see below). Time values are interpreted as for the L command below.

O **n** If **n** is numeric, this sets the current octave. **n** may also be one of L or N to enable or disable octave-tracking (it is disabled by default). When octave-tracking is on, interpretation of a pair of letter notes will change octaves if necessary in order to make the smallest possible jump between notes. Thus ‘‘olbc’’ will be played as ‘‘olb>c’’, and ‘‘olcb’’ as ‘‘olc<b’’. Octave locking is disabled for one letter note following >, < and O[0123456]. (The octave-locking feature is not supported in IBM BASIC.)

> Bump the current octave up one.

< Drop the current octave down one.

N **n** Play note **n**, **n** being 1 to 84 or 0 for a rest of current time value. May be followed by sustain dots.

L **n** Sets the current time value for notes. The default is L4, quarter or crotchet notes. The lowest possible value is 1; values up to 64 are accepted. L1 sets whole notes, L2 sets half notes, L4 sets quarter notes, etc.

P **n** Pause (rest), with **n** interpreted as for L **n**. May be followed by sustain dots. May also be written ~.

T **n** Sets the number of quarter notes per minute; default is 120. Musical names for common tempi are:

	Tempo	Beats Per Minute
very slow	Larghissimo	
	Largo	40-60
	Larghetto	60-66
	Grave	
	Lento	
	Adagio	66-76
slow	Adagietto	
	Andante	76-108
medium	Andantino	

	Moderato	108-120
fast	Allegretto	
	Allegro	120-168
	Vivace	
	Veloce	
	Presto	168-208
very fast	Prestissimo	

M[LNS] Set articulation. MN (N for normal) is the default; the last 1/8th of the note's value is rest time. You can set ML for legato (no rest space) or MS for staccato (1/4 rest space).

Notes (that is, CDEFGAB or N command character groups) may be followed by sustain dots. Each dot causes the note's value to be lengthened by one-half for each one. Thus, a note dotted once is held for 3/2 of its undotted value; dotted twice, it is held 9/4, and three times would give 27/8.

A note and its sustain dots may also be followed by a slur mark (underscore). This causes the normal micro-rest after the note to be filled in, slurring it to the next one. (The slur feature is not supported in IBM BASIC.)

Whitespace in play strings is simply skipped and may be used to separate melody sections.

FILES

/dev/speaker speaker device file

SEE ALSO

spkrtest(8)

HISTORY

The **speaker** device appeared in FreeBSD 1.0.

AUTHORS

Eric S. Raymond <*esr@snark.thyrsus.com*>, June 1990

PORTED BY

Andrew A. Chernov <*ache@astral.msk.su*>

BUGS

Due to roundoff in the pitch tables and slop in the tone-generation and timer hardware (neither of which was designed for precision), neither pitch accuracy nor timings will be mathematically exact. There is no volume control.

The action of two or more sustain dots does not reflect standard musical notation, in which each dot adds half the value of the previous dot modifier, not half the value of the note as modified. Thus, a note dotted once is held for 3/2 of its undotted value; dotted twice, it is held 7/4, and three times would give 15/8. The multiply-by-3/2 interpretation, however, is specified in the IBM BASIC manual and has been retained for compatibility.

In play strings which are very long (longer than your system’s physical I/O blocks) note suffixes or numbers may occasionally be parsed incorrectly due to crossing a block boundary.