Project 3: Sound driver for GeekOS

CMSC412 Spring 2013

Overview

The task is to build a sound driver for GeekOS. The intent of the assignment is to provide some exposure to devices and their interface. The virtualization of QEMU may mean that the emulated device is a bit less picky about details, and can allow some extra debugging options.

Success will be demonstrated by playing a wave file named "/c/startup.wav" on startup, which may be 8 or 16 bit, mono or stereo, any valid sampling rate, up to 200KB long. The sound should play completely and without static, hiccup, or repeat. The sampling rate and other properties are in the header of the wav file.

Implement the SB16 driver. There are others that QEMU supports, but the SB16 is reasonably simple.

Details on the SB16 and WAV are available in the references at the end of this handout.

Stuff not to do

Input is not required. (I don't believe QEMU supports sound input.) MIDI and FM synthesis, other typical features of the sound card, are not required.

I did not need to configure the mixer or volume controls, so I don't expect you will need to either.

You need not implement a device file or system call or any other means of allowing a user-space program to cause a sound to be played. Play a sound on startup.

There are test sound files in the sound directory.

Notes

Qemu will require the "-soundhw all" (or "-soundhw sb16") arguments to include sound support. The Qemu build includes sound by default, each invocation, not so much. The "make run" target in the build subdirectory of GeekOS should take care of this, but if you run a custom qemu, it's something to be aware of.

You may make reasonable assumptions about hardware parameters: Assume the defaults of port 0x220, DMA 1 and 5, IRQ 5, even to the point of hard-coding these values. I'm not looking for production-quality driver code; just enough to show that you understand the basics.

You may want to build qemu yourself to include debugging output messages from the sb16 driver. The log of operations can help double check which bytes went to which ports.

In order to record the output on the submit server, I will use

```
env QEMU_WAV_FREQUENCY=22050 QEMU_AUDIO_DRV=wav qemu
```

to request that qemu write audio to disk. (It is not terribly easy to stumble upon the documentation that describes this feature.) The submit server will likely be limited to checking the size of this file and that the contents are nonzero, and teaching staff will have to review to ensure that the audio output is of adequate quality.

My implementation uses about 400 lines total, 170 lines if counting semicolons. It relies on the soundblaster auto-init dma strategy for playing sounds. I believe this to be the one true way, since periodic events in GeekOS are likely very tricky.

It should be easier to appease Qemu's soundblaster emulator than it would be to satisfy a real card. Enjoy the simplicity.

Correct sound output depends on many details. Use revision control to keep track of your changes if you start resorting to adding and removing code that might work.

Functions

You should not need to program the DMA yourself: the `Setup_DMA` function should work.

`Reserve_DMA()` is useful.

`In_Byte()` and `Out_Byte()` are useful.

`Install_IRQ()` and `Enable_IRQ()` are useful.

Helpful documents

- A collection of generally useful hardware documents for an MIT class: `http://pdos.csail.mit.edu/6.828/2011/reference.html`
- SoundBlaster guide: `http://pdos.csail.mit.edu/6.828/2008/readings/hardware/SoundBlaster.pdf`
- SoundBlaster code tutorial: `http://www.justindeltener.com/sound-programming/sound-blaster-16-tutori` (was: `http://www.inversereality.org/files/mainpage.pdf`). Note that there's a "sound programming" menu with detail on individual topics.)
- Another: `http://homepages.cae.wisc.edu/~brodskye/sb16doc/sb16doc.html`
- Wave format: `https://ccrma.stanford.edu/courses/422/projects/WaveFormat/`
- General hardware: `http://wiki.osdev.org/Main_Page` (Although the text is not uniformly good, there are pointers to reference material.)

If you find more, you're welcome to post a comment on piazza, and required to cite your sources in sound.c.