ComS 229 Project 1 (Part 3)

Tools for digital sound

September 30, 2013

0 Introduction

This is a "change" document. In other words, rather than writing another entire spec, this document focuses on what should be changed relative to parts 1 and 2 of the project.

1 The programs

Write or update following programs, in ANSI C. As usual, this describes the *minimum* required functionality; you are free to implement extra if you like.

1.0 `sndinfo`

No changes, but be ready for "stress tests".

1.1 `sndconv`

No changes, but be ready for "stress tests".

1.2 `sndcat`

No changes, but be ready for "stress tests".

1.3 `sndcut`

No changes, but be ready for "stress tests".

1.4 `sndshow`

No changes, but be ready for "stress tests".

1.5 `sndedit`

This is an audio file editor based on the `ncurses` library[1]. The executable takes a single argument:

```
$ sndedit path/to/file
```

[1] See for example `http://www.tldp.org/HOWTO/html_single/NCURSES-Programming-HOWTO/`

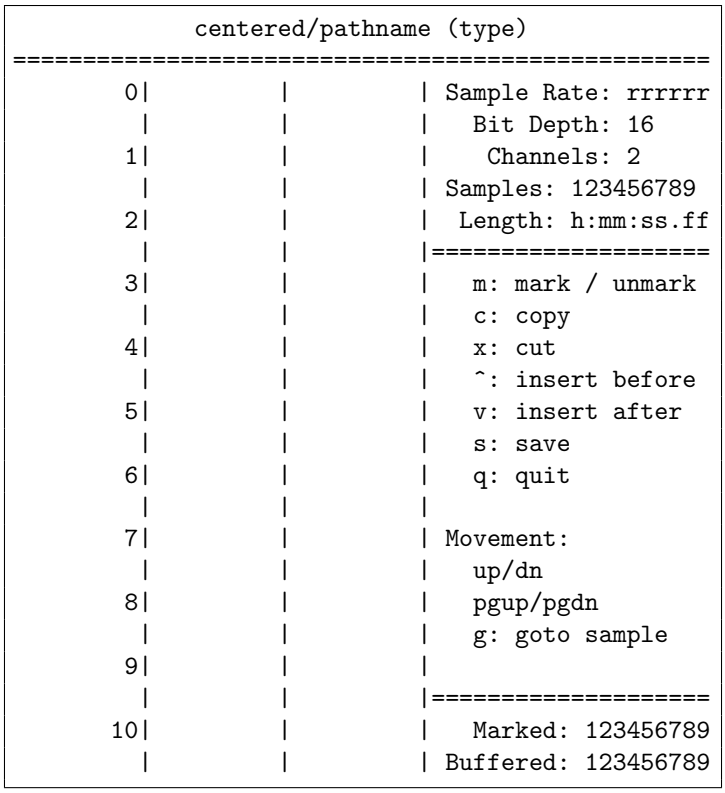

Figure 1: Screen map of `sndedit` in a 50 by 24 terminal.

where the argument is the pathname of an audio file (either AIFF or CS229 format) to be "edited". If the audio file cannot be opened, or is not a valid file, or some other error condition occurs, `sndedit` should print an appropriate message to standard error, and terminate. Otherwise, `sndedit` should display an interactive screen (using `ncurses`) very similar to the one shown in Figure 1. The top 2 lines are fixed as shown, with the first line used to display the pathname and format of the file being edited. The right–most 20 columns are used for the sound information and menu. The left–most 9 columns are used for the sample numbers. The remaining columns are used to display sound data, in a similar manner as `sndshow`.

Most of the time, the cursor should be somewhere in the center "`|`" column, and should move up and down when the "arrow keys" are pressed. The left portion of the screen (with the sound data) should scroll up and down when the cursor reaches the edge of the window, and whenever the page–up and page–down keys are pressed. Additionally, the following keystrokes should be supported, and should not be case sensitive:

'g' (goto): somewhere in the menu window, prompt the user for a sample number, and then restore the window. If a legal sample numer is entered, scroll the sound data window such that the cursor is on the specified sample.

'm' (mark): If no mark is set, then use the current sample number as the marked sample; otherwise, clear the marked sample. When a sample is marked, all samples between the marked sample and the cursor should be displayed in "reverse video". These are the samples that are "selected" for the copy and cut operations, below. If there is a marked sample, then the bottom–right corner of the display should show which sample number is marked; otherwise, that line should not appear in the display.

'c' (copy): If no samples are marked, then this menu item should not be available (not displayed and the keypress does nothing). Otherwise, all marked samples are copied into a buffer. If

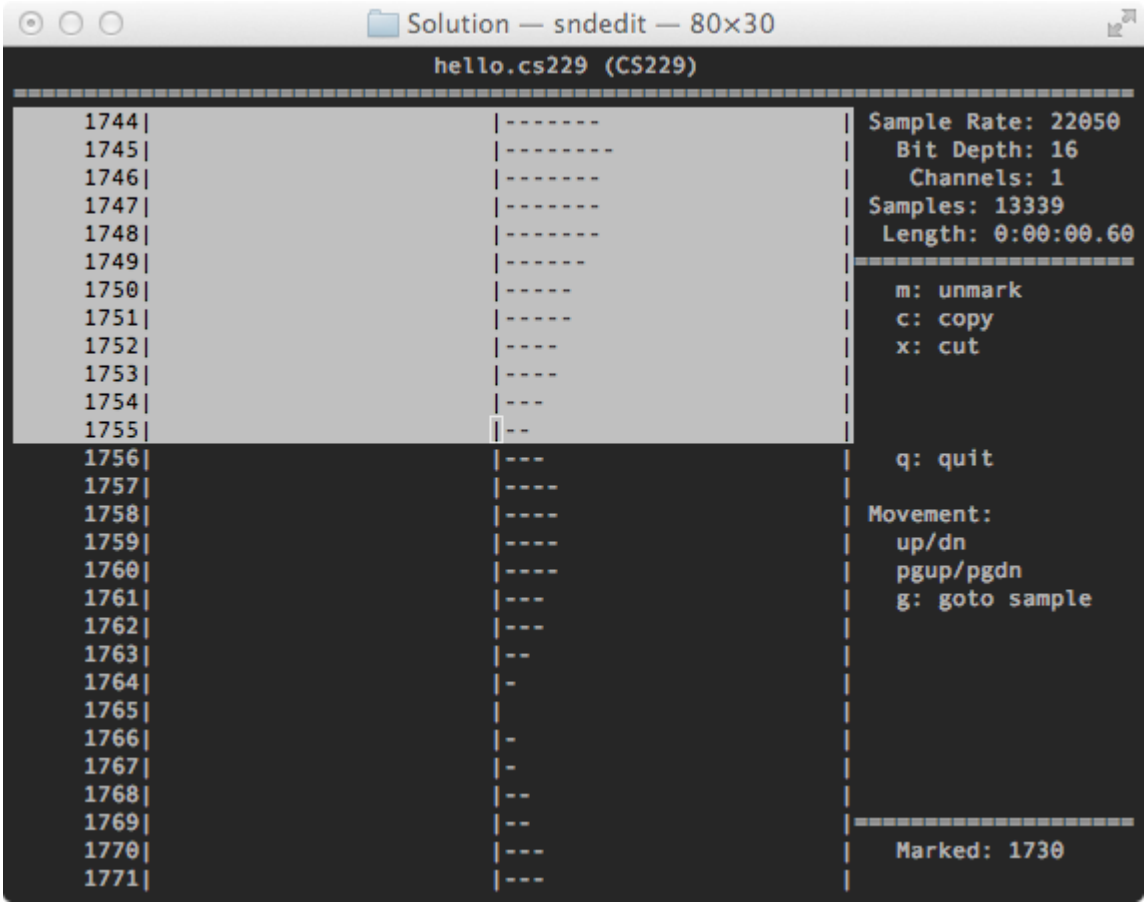

Figure 2: Sample screenshot for `sndedit`.

one or more samples are stored in the buffer, then the bottom–right corner of the display should show the current number of samples in the buffer.

'x' (cut): Just like copy, except the selected samples are also removed from the sound data.

'^' (paste): If no samples are in the buffer, then this menu item should not be available. Otherwise, copy the contents of the buffer into the sample data, right before wherever the cursor is currently positioned.

'v' (paste): Just like the other paste keystroke, except the buffer is copied right after wherever the cursor is currently positioned.

's' (save): If the sound data has not been modified, then this menu item should not be available. Otherwise, this causes any changes to the sound data to be saved back to the original file.

'q' (quit): Revert the screen back to normal, and quit.

Note that the upper right of the display, which shows the information about the sound data, should be updated whenever the sound data is modified. Figure 2 shows an actual screenshot. Note in the figure that sample number 1730 is marked, and the cursor is currently on sample number 1755.

2 Limits

- You may assume that the terminal window will not be resized while `sndedit` is running.
- You may require the terminal window to contain at least 40 columns and at least 24 rows for `sndedit` to run.
- You may assume the sample rate will be less than 1,000,000.
- You may assume the number of samples will be less than 1,000,000,000.

3 Submitting your work

You should turn in a gzipped tarball containing your source code, makefile, and a `README` file that documents your work. The tarball should be uploaded in Blackboard.

Your executables will be tested on `pyrite.cs.iastate.edu`. You should **test early, and test often on `pyrite`**. We will use some scripts to check your code; this means you should not change the name of the executables or the order in which your programs prompt for input.

4 Grading

The following distribution of points will be used for this part of the project.

sndinfo : 25 points

sndconv : 25 points

sndcat : 50 points

sndcut : 50 points

sndshow : 50 points

sndedit : 300 points

makefile : 50 points

As before, typing "`make`" should build all of your executables, and "`make clean`" should remove the executables and any object files.

Documentation & style : 100 points

Based on the `README` file and *all source code*.

Usability : 100 points

How usable (helpful error messages, etc.) are your programs. This includes user friendliness.

Total part 3 : 750 points