Lab 2 Assignment

YingTa Lin (ytl287)

Saturday 19th September, 2015

Assignment 1:

(Nothing need to be submit on this assignment)

Assignment 2:

(Nothing need to be submit on this assignment)

Assignment 3:

1. Why does encoding (or digitizing) a sample into a 16-bit signed integer have a width of 2?
   The unit used to represent width is bytes, and the resolution of a 16 bit signed integer is equals to 2 bytes.
2. What is WIDTH when the PyAudio format is paInt8?
   The result of `print pyaudio.get_format_from_width(1,unsigned=False)` is 16 and the result of `print pyaudio.paInt8` is 16 as well.
   The WIDTH is for format paInt8 is 1 byte

Assignment 4:

1. What hexadecimal symbol string gives -1 as a signed 16-bit integer using unpack() command?
   By using `struct.pack('>h',-1)` command, I get `\xFF\xff` which is the hexadecimal symbol for 1 converted with 2's complement indicates negativity.
2. What hexadecimal symbol string gives 256 as a signed 16-bit integer using unpack() command?
   The same command `struct.pack('>h',256)` had used to get the result `\x01\x00`.
3. What hexadecimal symbol strings give the above numbers as signed 32-bit integers using unpack() command?
   The result is basically the same, but extra 16 bits added to the left:
   For -1, the command returns `\xFF \xFF \xFF \xFF` and for 256, it returns `\x00 \x00 \x01 \x00`.
4. Change the value of gain to a larger number (e.g. 400000, 800000 or even larger). What happens? What error do you get?
   The error is that short format requires SHRT_MIN <= number <= SHRT_MAX.

```
Traceback (most recent call last):
  File "filtering_paInt16_a.py", line 48, in <module>
    str_out = struct.pack('h', out)     # 'h' for 16 bits
struct.error: short format requires SHRT_MIN <= number <= SHRT_MAX
```

   The error occur because with larger gain, the maximum value of variable 'out" will exceed the boundary of short format. For higher gain, we can use signed 32 bits options with index 'i'.
5. In filtering_paInt16_a.py, how should you set the gain to ensure the peak amplitude

does exceed the maximum allowed value of 2^{15}-1 ?
out = y0 * gain
|out| <= 2^{15}-1 which means |y0 * gain| < = 2^{15}-1 and therefore gain should less or equal than ($(2^{15}-1)/|yo|$)
10173.8 is the maximum gain for this input x and filter system.

6. Modify filtering_paInt16_a.py to avoid run-time overflow errors even if gain is very high. Do this by inserting an if statement to verify if the sample value is in the allowed range; and if not, to set the sample value to its maximum (positive or negative) allowed value, before writing the sample value to the audio stream. Test your program by setting the gain to a high value. What effect does this have on the sound produced by the program?
   It sounds longer, but saturate at the the maximum amplitude.
   (Implemented with question 7 in python scripts: bound_samples.py and check_samples.py)
7. Implement the if statement by defining a python function with def. Put your function into its own module (file). Then import your module and use your function to modify the code example so that gain can be an arbitrary floating-point value without leading to a run-time error.
   See scripts: bound_samples.py and check_samples.py
8. Write a version of filtering_paInt16_a.py using 8 bits/sample. You may use either paInt8 or paUInt8 as the PyAudio format. Ensure run-time overflow errors do not occur.
   See filter_8bit_samples.py
9. Write a version of filtering_paInt16_a.py that applies the filter twice. The filter will be a fourth-order filter, but it should be implemented as two second-order filters in cascade.
   See python script two_sec_order_filter.py
10. Describe how to design two second-order filters (with same resonant frequency f1) so that the rise-time and decay-time of the impulse response can be specified? (The two filters will have different pole radii.) Given an example of the design in Matlab and its real-time implementation in Python/PyAudio