THE UNIVERSITY OF
ALABAMA IN HUNTSVILLE

DEPARTMENT OF ENGINEERING

CPE 381 - FUNDAMENTALS OF SIGNALS AND SYSTEMS FOR COMPUTER ENGINEERS

Real-time Signal Processing

Author:
Joshua Estes

March 9, 2023

1 The Problem and The Solution

The problem we needed to solve in this phase of this project is the recording and processing of the audio file. For my solution, I used Audacity to record an introduction of myself, and then filled the rest of the remaining time, up to fourteen seconds, with music. After exporting the audio file out of Audacity, I had to read it into Matlab and overwrite the data, after seven seconds, with a sine wave. After writing to a new audio file, I had to write a program in C or C++ to read in the audio file and add noise to the file. For the C++ portion of the project, I used a struct to read in the header of the WAV file, and printed the data to stdout and the appropriate data to a summary file. For reading in the data, I read in two bytes of data at a time and operated on that data before writing it to a new file.

2 WAVE File Format

There are thirteen fields in the WAVE header. The first section of fields is the RIFF Header. The RIFF header contains thee fields, the `ChunkID`, the `ChunkSize`, and the `Format` field. The `ChunkID` field is a four byte field that contains the letters "RIFF" in ASCII from. The `ChunkSize` field is a four byte field that contains the size of the rest of the file, not including the bytes from the `ChunkId` field and the `ChunkSize` field. The third field in this chunk is the `Format` field, which contains the letters "WAVE" in big-endian format.

The second section contains eight fields that specify the format of the sound data. The first four byte field is the `Subchunk1ID`, which contains the letters "fmt " in big-endian form. The next field, the `Subchunk1Size`, is a four byte field that contains the size of the rest of the sub-chunk after this field. The `AudioFormat` field is a 2 byte field that indicates what kind of compression the data has undergone. A value of one would be PCM (Linear Quantization). The `NumChannles` field is a two byte field that indicates the number of channels in the file, where one is mono and two is stereo.

The next field in the format chunk is a four byte field that indicates the sample rate in the audio date, called `SampleRate`. The next four byte field is the `ByteRate` field which specifiers the byte rate of the data, or amount of bytes per second to be played when playing the audio. The next two fields is the `BlockAlign` field and the `BitsPerSample` field. The `BlockAlign` field is the amount of bytes contained in one sample, which includes sub-samples for each channel. The `BitsPerSample` field indicates the size of each sample in bits.

The next chunk of data is the data sub-chunk, which contains the size of the data and the actual sound data. The first field is the `Subchunk2ID`, which is a four byte field that contains the letters "data" in big-endian form. The second field is the `Subchunk2Size`, which contains the number of bytes in the data itself. The final field in this sub-chunk is the `Data` field, which is the actual audio data. Figure A in the appendix shows the format of the header which mimics how it would be formatted in the WAV file itself.

3 The Summary File

For my summary text file, there were thirteen fields. The first field contains my name and the second field is the name of the input WAV file. The next two fields are the amount of smaple in the file interpreted two different ways, the first is assuming that each channel is its own sample, and the second is assuming that both channels samples make one sample together. The next three fields are the sampling frequency, the number of bits per sample, and the length of the recording. The next four fields is the maximum value in each channel, before and after adding noise to the file. The last two fields are the execution time of the program in nanoseconds and in milliseconds.

4 Can This Program Run in Real Time?

Looking at the summary file, the execution time of the program is always around 1877 milliseconds. Since it takes 1.877 seconds for this program to process fourteen seconds of data, this program could operate in real time. At a higher sampling frequency, such as 384000 Hz, then the program would be overwhelmed and not operate in real time. A WAV file with a sampling frequency of 384000 Hz would take 64,548 milliseconds to process, on an Nvidia AGX Xavier.

Appendix

A WAV Header Struct

```
struct WAV_HEADER
{
        uint8_t RIFF[4];
        uint32_t ChunkSize;
        uint8_t WAVE[4];
        uint8_t fmt[4];
        uint32_t Subchunk1Size;
        uint16_t AudioFormat;
        uint16_t numChannels;
        uint32_t SamplesPerSec;
        uint32_t bytesPerSec;
        uint16_t blockAlign;
        uint16_t bitsPerSample;
        uint8_t Subchunk2ID[4];
        uint32_t Subchunk2Size;
};
```

B Output File from 384000 Hz on Nvidia AGX Xavier

```
Name:                                        Joshua Estes
File Name:                                   Test.wav
Number of samples in total:                  10752000
Number of samples:                           5376000
Sampling Frequency:                          384000
Number of bits per sample:                   16
Record length:                               14
Maximum value in channel 1 before noise:     32767
Maximum value in channel 2 before noise:     32767
Maximum value in channel 1 after noise:      32767
Maximum value in channel 2 after noise:      32767
Execution time in nanoseconds:               64548476597 nanoseconds
Execution time in milliseconds:              64548 milliseconds
```

C Output File from 11025 Hz on Nvidia AGX Xavier

```
Name:                                        Joshua Estes
File Name:                                   ESTES_J_mod.wav
Number of samples in total:                  308700
Number of samples:                           154350
Sampling Frequency:                          11025
Number of bits per sample:                   16
Record length:                               14
Maximum value in channel 1 before noise:     19167
Maximum value in channel 2 before noise:     19005
Maximum value in channel 1 after noise:      19122
Maximum value in channel 2 after noise:      19142
Execution time in nanoseconds:               1865117116 nanoseconds
Execution time in milliseconds:              1865 milliseconds
```

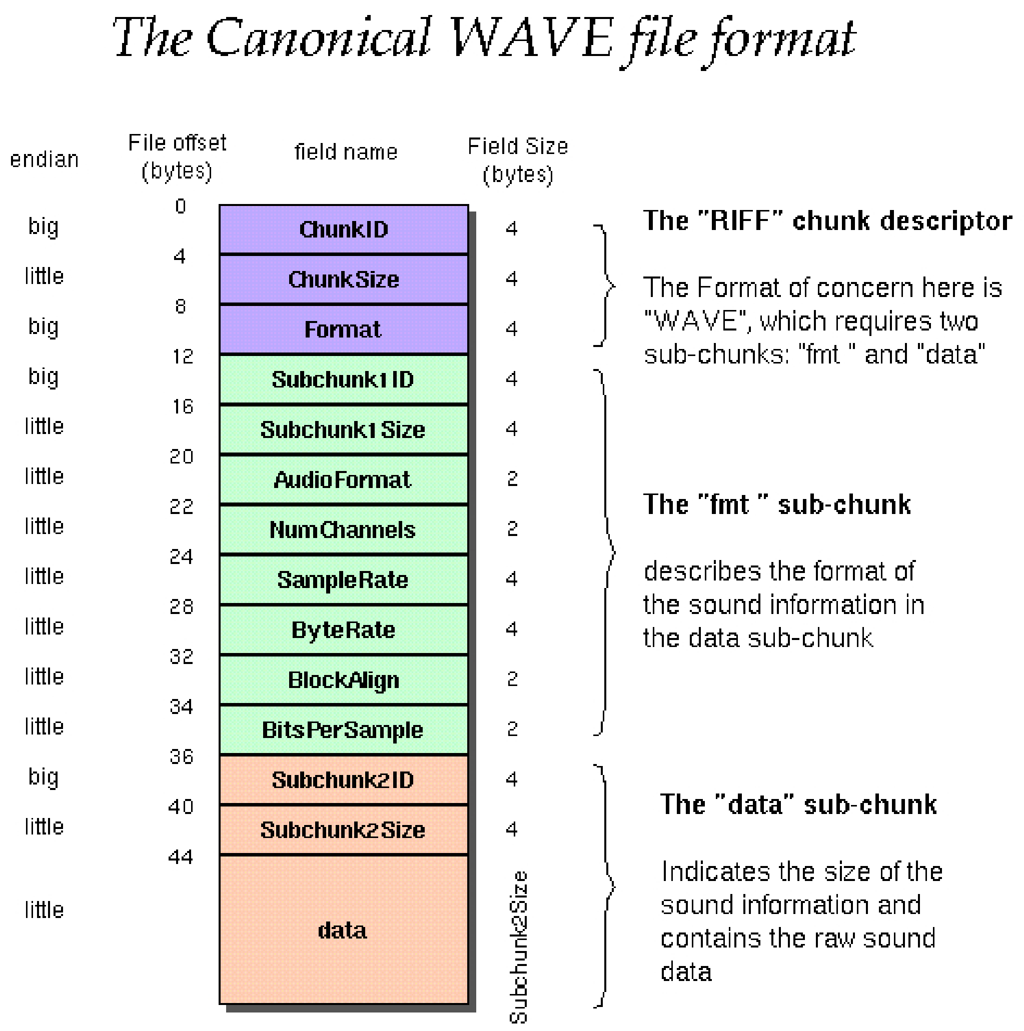

Figure 1: The Canonical WAVE File Format

Source: sapp.org