Курсовая работа. Работа с мультимедиа файлами WAV

Цель: Ознакомиться с типизированными файлами на примере формата wav. Отработать навык работы со структурами, методами обработки данных файла.

1. Методические указания. Структура файла WAV.

Для большинства wav файлов может быть использована следующая структура заголовка (для наглядности поля сгруппированы смещением от левого края):

```
struct WAVEHEADER
{
    char chunkId[4];
    unsigned long   chunkSize;
        char format[4];
            char subchunk1Id[4];
                  unsigned long   subchunk1Size;
                  unsigned short  wFormatTag;
                  unsigned short  nChannels;
                  unsigned long   SamplesPerSec;
                  unsigned long   AvgBytesPerSec;
                  unsigned short  blockAlign;
                  unsigned short  BitsPerSample;
            char subchunk2Id[4];
                  unsigned long   subchunk2Size;

  };
```

Поле **chunkId[4]** содержит информацию о формате файла для wav файла это **RIFF** (RIFF - Resource Interchange File Format, то есть контейнер для потоковый данных). По типу формата файла определяется смещение остальных полей и самих данных.

Поле **chunkSize** содержит размер в байтах оставшейся части файла, то есть за исключением полей **chunkId[4]** и **chunkSize**. Таким образом, общий размер файла равен chunkSize+8.

Поле **format[4]** указывает на конкретный формат потоковых данных, так как формат RIFF может быть использован для хранения данных avi или rmi. Для wav файла значение этого поля равно “WAVE”.

Поле **subchunk1Id[4]** представляет собой заголовок fmt-чанка, то есть той части заголовка, где описываются параметры wav файла.

В поле **subchunk1Size** содержится размер подструктуры subchunk1, то есть 16 байт для формата PCM (Pulse Code Modulation).

Поле **wFormatTag** определяет аудио формат значение 0 используется для неизвестного формата, значение 1 для формата PCM. На данный момент количество существующих форматов превышает 200.

Поле **nChannels** определяет количество каналов. Чаще всего это Mono(1) или Stereo(2).

Поле **SamplesPerSec** определяет частоту дискретизации в Гц, например 16000 Гц. Сэмпл – это блок звуковых данных включающий оба канала.

Поле **AvgBytesPerSec** или ByteRate определяет количество передаваемых байт в секунду воспроизведения.

Поле **blockAlign** содержит размер Сэмпла в байтах. Может принимать значения 1, 2, 4 и тд.

Поле **BitsPerSample** содержит размер сэмпла в битах.

Поле **subchunk2Id[4]** обозначает начало чанка данных, то есть непосредственно отсчетов звукового сигнала. Содержит символы “data”.

Поле **subchunk2Size** содержит размер области данных в байтах.
Количество сэмплов в файле можно определить следующим образом subchunk2Size*8/BitsPerSample.

Использование данной структуры корректно для формата PCM, но для других форматов может привести к ошибкам, так как структура подзаголовков может отличаться.

После заголовка WAVEHEADER следуют непосредственно чередующиеся отсчеты звукового сигнала:

Sample[0],Sample[1],Sample[2],Sample[3] и тд.

Для Mono звука Sample[0] содержит Channel0[0], так как в данном случае в сэмпле содержится один канал.

Для Stereo звука Sample[0] содержит ChannelLeft[0] и ChannelRight[0], так как в данном случае в сэмпле содержится два канала.

2. Задание к выполнению.

Написать программу на языке C/C++, осуществляющую действия в соответствии с заданием. Считать заголовок и данных исходного wav файла. Определить параметры звукового файла и вывести их на экран. В соответствии с вариантом задания определить параметры выходных звуковых файлов и вывести их на экран, создать выходной файл(ы), проверить работоспособность программы.

Варианты заданий:

1. Реализовать функцию разделяющую входной wav файл на каналы (на два выходных файла) с возможностью уменьшения амплитуд любого из каналов.

2. Реализовать функцию разделяющую входной wav файл на две части (на два выходных файла) по времени в зависимости от заданного значения времени.

3. Реализовать функцию, объединяющую два wav файла в один путем добавления отсчетов второго в конец первого, параметры BitRate, SamplesPerSec соответствуют первому файлу. Поля nChannels и blockAlign для входных файлов должны быть одинаковы.

4. Реализовать функцию объединения двух mono файлов в один stereo файл, где первый файл является левым каналом, а второй правым каналом выходного wav файла. Значения BitsPerSample, SamplesPerSec должны быть одинаковыми. При различной длине звуковых сигналов выходной файл должен иметь длину наименьшего.

5. Реализовать функцию изменения количества битов на отсчет.

Выбор варианта осуществляется по формуле $(N_{группы}+N_{вжурнале})\%5+1$

Комментарии.

Рекомендуется использовать функции:

```
fopen_s(&f,file_name,open_type);
```
Для работы с wav файлами используйте бинарные типа “wb” и “rb”

```
fread(&buf,sizeof(buf_type),count,f);
fwrite(&buf,sizeof(buf_type),count,f);
```
Считывание и запись данных в файл может производиться поэлементно, а также применительно ко всем данным в зависимости от значения count.

Если после создания нового wav файла, он не воспроизводится, то вероятнее всего ошибка в заголовке файла, изменение любых параметров звукового сигнала кроме амплитуд происходит в заголовке.

3. Содержание отчета

1. Титульный лист.
2. Цель работы.
3. Задание к лабораторной работе.
4. Листинг программы.
5. Результат работы программы (данные звуковых файлов)
6. Выводы по лабораторной работе.