# 中期检查报告：音频播放

大作业-8：杨碧茹（组长），董文冲，闵安娜，林智鑫

## 一、已选题目调研

### xv6

xv6是一个类Unix操作系统，有现代操作系统的基础框架和功能。

### 本组选题

本组选题是在xv6基础上实现音频播放功能。主要涉及音频文件的读入和解码、与音频驱动（声卡）的交互、播放过程中的控制功能，以及诸如启动程序、错误处理等配套功能。

### 核心难点

整个项目的重难点有以下几个方面：

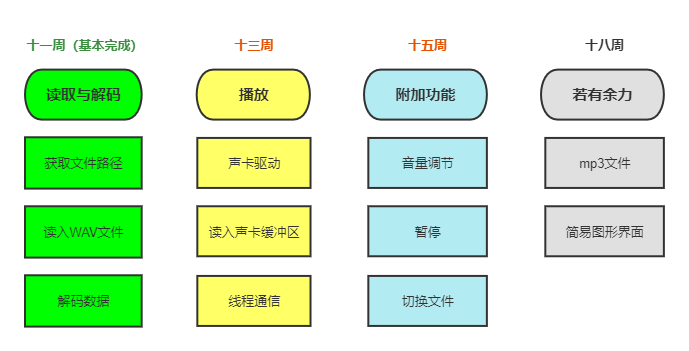
1. 读入指定的音频文件；
2. 对音频文件按照格式进行解码；
3. 将解码后的音频正确使用声卡和音频驱动播放；
4. 在播放过程中，播放线程之外另外开启控制线程，接收用户的暂停等指令。

详细的功能点、技术路线将在下文中详细描述。

## 二、预期功能目标

本项目有以下几个预期功能目标：

* 音频播放器可以从Shell中启动，并接收播放文件名、额外选项等参数列表；
* 音频播放器可以定位并读入指定的音频文件，并支持至少一种音频类型(wav)的解码；
* 音频播放器可以将解码后的音频文件正确地通过音频驱动和声卡播放；
* 在播放过程中，可以进行暂停/继续、停止、变速、切换音频等基础操作；
* 对预料之外的用户输入和错误有一定的处理能力。

具体**开发时间表**如下：  


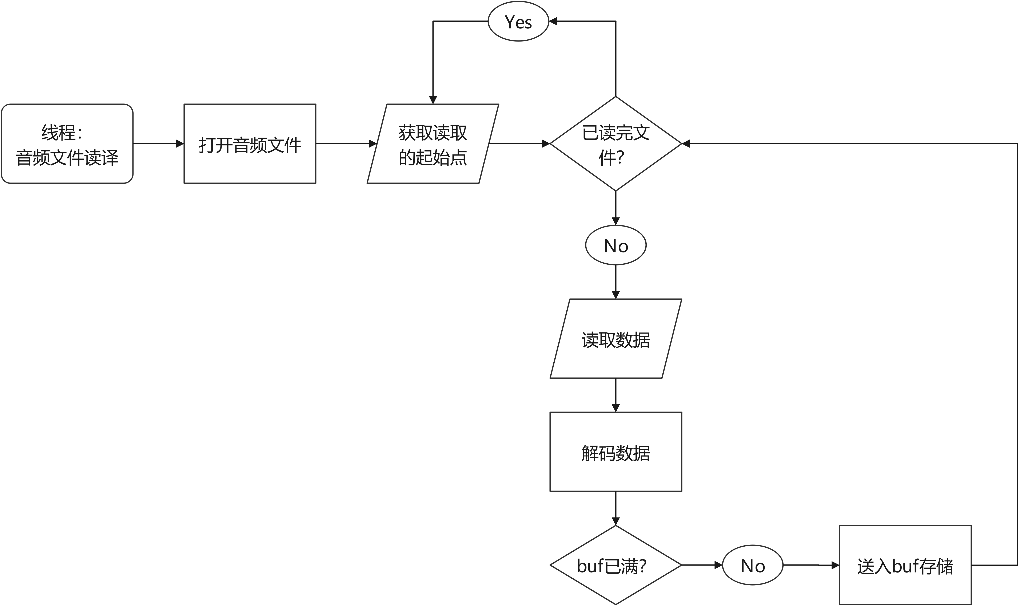
## 三、技术框架

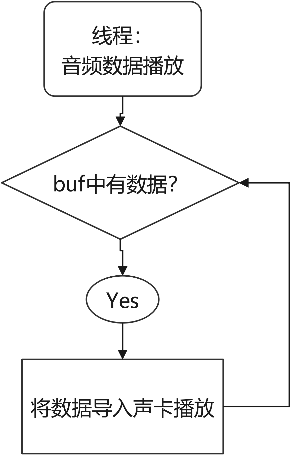
### 整体思路

使用3个线程实现需求。线程1负责读入音频文件并解码（音频文件读译）；线程2负责解码后的数据导入声卡播放（音频数据播放）；线程3负责接收外界指令，对线程1、2进行调控（控制器）。

### 流程图

## C:\Users\ybr19\Documents\100%_MOREHAB\Learning\第三学年\春季学期\操作系统\作业\大作业\image\中期报告\process0.png



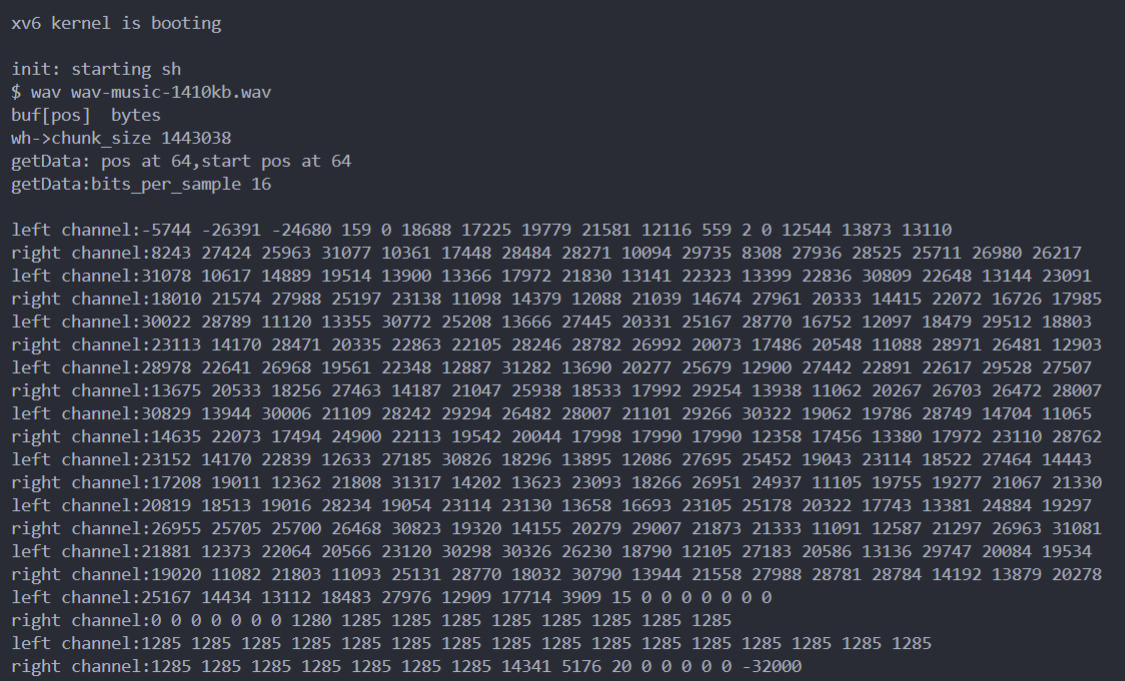


## 四、已实现功能目标

### WAV文件的读入

1. wav 文件以 RIFF 为基础，该格式以 Header、FormatChunk、DataChunk 三部分组成。
2. Header 首先四个字符是大端序 ChunkID，它总是 RIFF，指明其格式；之后是四字节小端序 ChunkSize，表示文件的总字节数 - 8，这个 magic number 是 ChunkID 与 ChunkSize 合占八字节，即 ChunkSize 表示其之后的所有字节的大小。之后是四字节大端序 Format，对于 wav 总是 WAVE。
3. FormatChunk 首先是四字节大端序 Subchunk1ID，其值总是 fmt，表示 FormatChunk 从此开始。在此后是四字节小端序 Subchunk1Size，表示 FormatChunk 总字节数 - 8，这个 8 的含义与 2 中相同。之后是二字节小端序 AudioFormat，对于 wav 总是1。NumChannels 二字节小端序，表示总声道个数。SampleRate 四字节小端序，表示每个通道上每秒包含多少帧。ByteRate 四字节小端序，大小等于 SampleRate *BlockAlign，表示每秒含多少字节。BlockAlign 二字节小端序，等于 NumChannels* BitsPerSample / 8，表示每帧的多通道总字节数。BitsPerSample 二字节小端序，表示每帧包含多少比特。
4. DataChunk 首先是四字节大端序 Subchunk2ID，其值总是 data，表示 DataChunk 从此开始。在此后是四字节小端序 Subchunk2Size，表示 data 段的总字节数，也就是 DataChunk 的大小 - 8。之后是小端序 data，大小为 Subchunk2Size，表示音频波形的帧数据，各声道按帧交叉排列。

代码打印输出解析数据（此处暂时用short打印，测试文件为双声道）如下：



### 修改xv6最大文件大小限制

#### 文件读写

xv6无法直接在ubuntu上读文件，要把文件读入fs.img（文件系统映像）。

#### 修改限制

WAV文件大小计算公式是采样频率(kHz) x 采样位数 x 声道数 x 时间(秒) / 8 = 文件大小(kb)，如果采用如下的参数： 采样率：8kHz 采样位数：16 声道数： 2 ，那么：一分钟WAV文件的大小 = 8 x 16 x 2 x 60 / 8 = 1920KB , 可近似成 2M 计算。

而目前，xv6文件被限制在140个扇区，即71,680字节。这个限制来自于一个xv6 inode包含12个“直接”块号("direct" block number)和一个“单间接”块号，这个块号指的是一个最多可以容纳128个块号的块，总数为12+128=140。这对于播放WAV文件是不够的。

xv6文件系统由inode组成，每个inode是单个未命名的文件。整个磁盘读写的最小单元为block (xv6为512字节)。 xv6文件系统采用位图块来管理磁盘中的块，每个块可以管理的大小为 BPB = BSIZE \* 8，若该标志为0，则块空闲，否则已经使用。其中整个磁盘分布如下图。

第0个block为启动区，第1个block为超级块 (也是根目录所在的块),接下来是连续分布的dinode，最后是连续分布的BPB(块位图)

假设块大小为BSIZE, 则

1. 每个块包含的dinode结构体的数量为: IPB = BSIZE/sizeof(dinode)。等价于第i个dinode所在的block为IBLOCK(i) = i/IPB+2
2. 第i个block所在的位图块为： BBLOCK(b, ninodes) = b/BPB+ninodes/IPB+3

默认的xv6直接分配的时候，文件尺寸的大小为NDIRECT·BSIZE，索引分配的时候，文件尺寸的大小为NINDIRECT·BSIZE ,其中NINDIRECT = BSIZE/sizeof(uint)

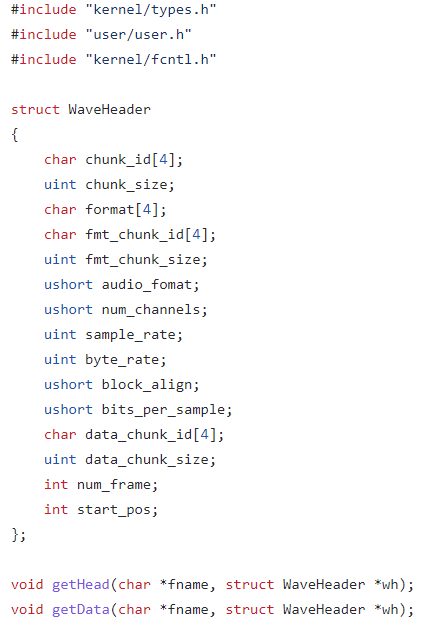
以下有3种方法可以更改xv6文件系统代码（以防后续有冲突，都进行尝试，目前没有冲突），使每个inode中支持“双间接”块("doubly-indirect" block)，其中包含128个单间接块地址，每个单间接块最多可以包含128个数据块地址。其结果是，一个文件将能够由最多16523个扇区(或大约8.5 mb)组成:11+128+128\*128=16523。

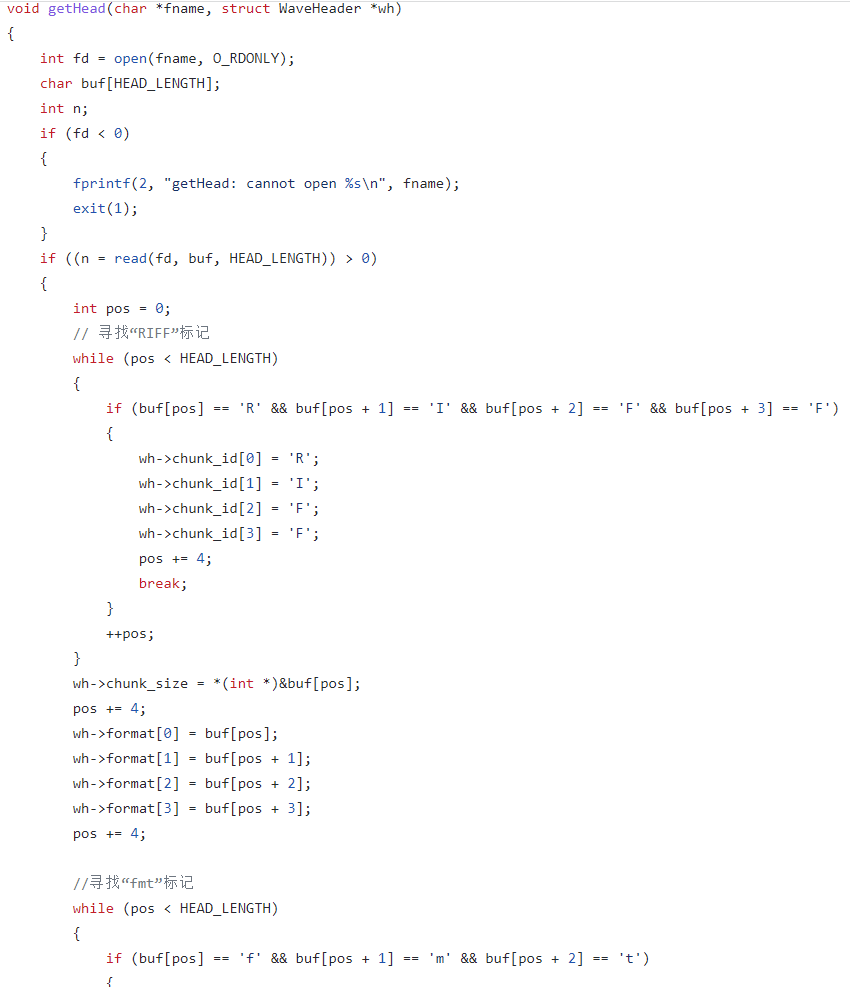
1. 最直接的是改变block size大小和直接分配/索引分配时的设定大小。
2. 或者改变直接分配的模式，把inode 结构体中uint addrs[NDIRECT+1]中所有的索引都指向一个块（即都变成INDIRECT模式，这时可以最多支持(512/4) *(512/4)* 512 = 8M。
3. 或者在一级索引节点后增加二级索引节点，改变文件大小限制所在扇区数量。修改bmap()，使它除了直接块和单间接块之外，还实现了双间接块，使每个inode中支持“双间接”块，其中包含128个单间接块地址，每个单间接块最多可以包含128个数据块地址。其结果是，一个文件将能够由最多16523个扇区(或大约8.5 mb)组成:11+128+128\*128=16523。

### 尚需修补的部分

1. 应当读取 BitsPerSample 比特（也就是 BitsPerSample / 8 字节大小，可以考虑开一个多个 short 的 union，或者干脆为了好写只允许它是特定的值）的数据作为一帧，每通道读取一帧之后统一送到某个地方，每秒重复这个过程 SampleRate 次。
2. 为了便于增加暂停/变速播放等功能，每个batch读取数据的大小应该设定新的函数来求。buf区大小和每次getData的大小参数，后续还需进行修改测试。
3. 耳机声效，暂停等功能应该在wav reader和导入声卡buf之间加。

### 代码截图





## 五、目前遇到的问题和困难

1. 难以对声卡做调试或测试。我们甚至不知道添加虚拟声卡后如果操作系统播放了声音会输出到电脑的实际声卡还是怎么样。
2. 缺乏与硬件打交道的具体知识。需要通过总线传输数据、直接与声卡寄存器交互，并且需要查阅大量相关手册。

## 六、人员分工

**调研与文档**：全组

**音频读入与解码**：董文冲、闵安娜

**声卡驱动与播放**：杨碧茹、林智鑫