

SD 卡音乐播放例程

1 实验简介

在其他实验中我们已经完成了单独的 SD 卡读写和音频模块的录音播放，本实验通过搜索 SD 卡中 WAV 音乐文件，然后送到音频模块播放，完成一个简单音乐播放器的功能。

2 实验原理

本实验一个关键是在没有文件系统的情况下，搜索 SD 卡每个扇区的内容，匹配出 WAV 文件，这里有一个假设：假设每一个文件都是从一个扇区第一个字节开始而且文件是连续存储的，经过大量实验，发现 FAT32 文件格式中的文件确实如此。

WAV 文件格式

大部分的文件都有文件头，WAV 也不例外，我们通过分析 SD 卡一个扇区的前几个字节，判断这个文件是否为 WAV 文件。

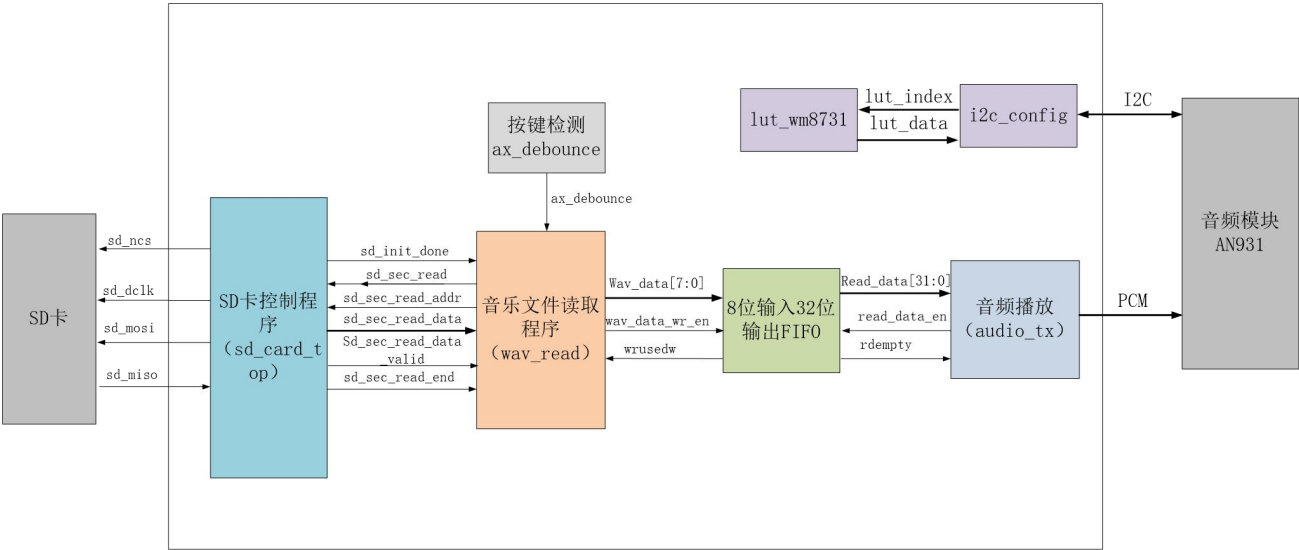
WAV 文件作为多媒体中使用的声波文件格式之一，它是以 RIFF 格式为标准的。RIFF 是英文 Resource Interchange File Format 的缩写，每个 WAV 文件的头四个字节便是“RIFF”，所以本实验就简单的以每个扇区的前 4 个字节是否为“RIFF”判断该文件是否为 WAV 文件，紧接着 4 个字节表示文件的大小，这样我们就可以确定要读取的数据量。WAV 文件头大小是 88 字节，在播放时要把前 88 个字节的文件头去掉。

OFFSET	LENGTH	VALUE	DESCRIPTION
0	4 bytes	'RIFF'	The file format ID.
4	4 bytes		Length of the file minus (-) 8 bytes.
8	4 bytes	'WAVE'	The data format ID.
12	4 bytes	'fmt '	The chunk ID.
16	4 bytes	32	Length of the chunk excluding the 8 bytes for the ID and length.
20	4 bytes		The codec ID.
24	4 bytes		The number of channels.
28	8 bytes		Samples per second.
36	8 bytes		Average bytes per second.
44	4 bytes		Block alignment.
48	4 bytes		Bits per sample.
52	4 bytes	'data'	The chunk ID.
56	4 bytes		Length of the data (chunk size minus (-) 8 bytes.
60	4 bytes	'fact'	The chunk ID.
	4 bytes	8	Chunk size minus (-) 8 bytes.
	8 bytes		Sample length.

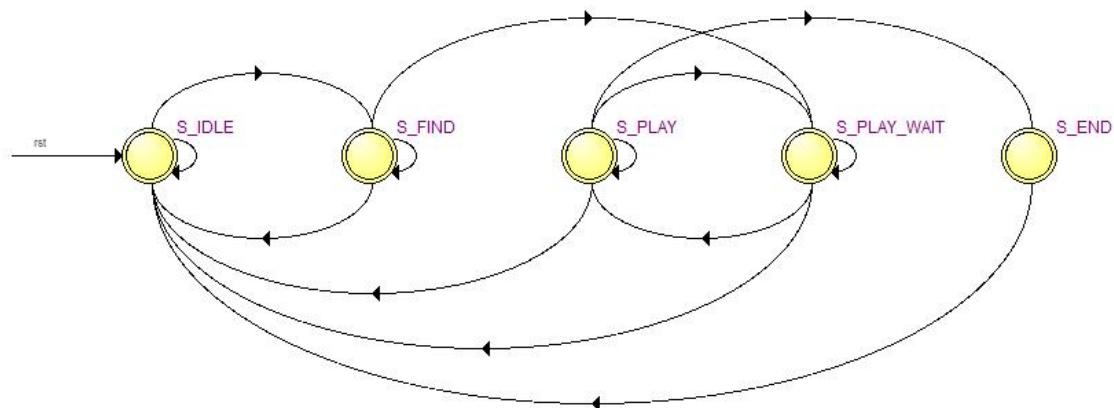
wav 文件头格式

3 程序设计

例程中通过 wav_read 搜寻 SD 卡中的 WAV 文件，并将 WAV 文件读取出来写入 FIFO，最后再从 FIFO 中将数据输出并写入音频模块。



由于在前面的例程中已经讲过其他模块的原理及使用这里不再赘述，仅说明 wav_read 模块。wav_read 模块完成了从 SD 卡中搜索 wav 文件，读取 wav 文件，状态机如下，如果有搜索请求（按键按下），则进入搜索状态“S_FIND”，如果搜索到 WAV 文件，进入“S_PLAY_WAIT”状态，检查 FIFO 的剩余空间是否够 512 个字节，如果有 512 个字节空间，就进入“S_PLAY”状态读取 wav 的一个扇区内容。



wav_read 模块状态机

在程序设计中，搜寻 SD 卡里图片的文件头是按每 8 个扇区地址（4K 字节）搜寻一次的形式操作，所以在一次搜寻结束后，下一次搜寻时需要将地址左移 3 位对齐以保证下一次搜寻的地址和图片在 SD 卡中存储的地址一致。

```

209         else
210         begin
211             //search every 8 sectors (4K)
212             sd_sec_read_addr <= sd_sec_read_addr + 32'd8;
213         end
214     end
215     else
216     begin
217         sd_sec_read <= 1'b1;
  
```

说明：因为 SD 卡格式化的时候按最小单元 4K 格式化，音乐文件的头文件一般都是在 SD 卡的 4K 分区的起始位置开始放置。

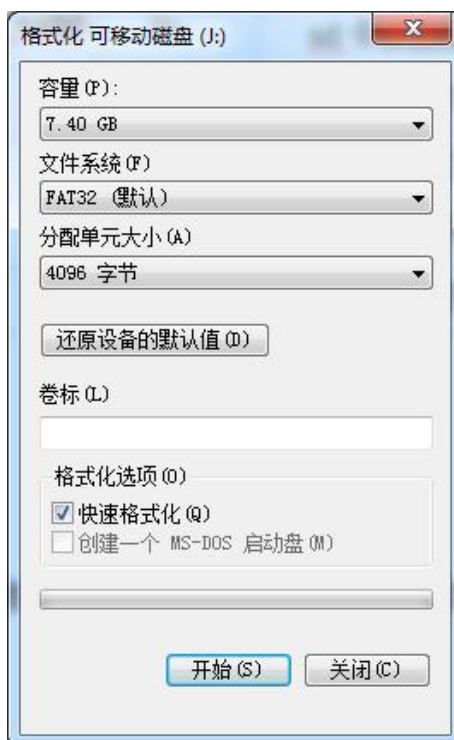
wav_read 模块的管脚说明：

信号名称	方向	说明
clk	in	时钟输入
rst	in	异步复位输入，高复位
ready	out	空闲状态指示
find	in	搜索播放请求
sd_init_done	in	sd 卡初始化完成
state_code	out	状态码 0, 表示 sd 还在初始化 1, sd 卡初始化完成，等待按键按下

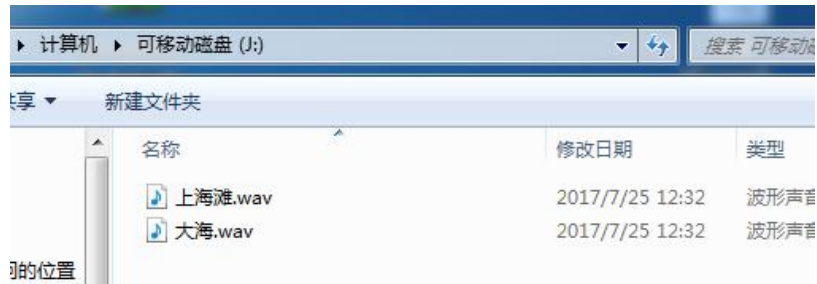
		2, 正在搜索 WAV 文件
		3, 找到 WAV 文件, 正在播放
sd_sec_read	out	sd 卡读请求
sd_sec_read_addr	out	sd 卡读请求扇区地址
sd_sec_read_data	in	sd 卡读到的数据
sd_sec_read_data_valid	in	sd 卡读数据有效
sd_sec_read_end	in	sd 卡读请求完成
fifo_wr_cnt	in	fifo 写端口已经使用量
wav_data_wr_en	out	wav 文件写使能
wav_data	out	wav 文件的音频数据

4 实验现象

- (1) 格式化 sd 卡 (fat32 格式), sd 卡必须是 2.0 以上的版本 (容量大于 4G)



- (2) 我们已经在 demo 的目录下为大家准备了一下的 2 个 wav 文件, 把 wav 格式文件放到 sd 卡中, 注意: **这里的 wav 文件是立体声双声道, 48K 采样率, 16bit 的音频文件。**



- (3) 将准备好的sd卡注入开发板的sd卡槽（sd卡不能带电插拔），插好音频模块，上电，下载实验程序，等待LED1灭、LED2、LED3、LED4亮（4'b0001）时，按下KEY2，这个时候等待LED1亮、LED2灭、LED3、LED4亮（4'b0010），表示正在搜索wav文件，如果找到LED1、LED2灭、LED3、LED4亮，这时候就可以听到音乐了。