

目 录

一、模块的硬件说明2

二、模块接线图3

三、测试方法7

附录一：SPI-FLASH 容量与音频长度对照表8

三、码率的转换方式9

四、模块的原理图12

四、程序范例13

一、模块的硬件说明

1、模块的供电

模块最理想的工作电压为 4.2V。所以用户如果是采用 5V 的电源供电的，建议串接一个二极管

2、模块的指示灯[上电状态]

工作状态	下载模式	播放语音	暂停	睡眠
状态	快闪	慢闪	常亮	灭

备注：正常工作状态指示灯

3、模块的指示灯[工作状态]

工作状态	一对一可打断	抬起停止	一对一不可打断	标准MP3功能
状态	常亮2S	快闪2S[100ms取反]	中慢闪2S[200ms取反]	慢闪2S[500ms取反]

备注：此时指示灯，只在上电初始化的时候，指示2秒

4、音频输出说明

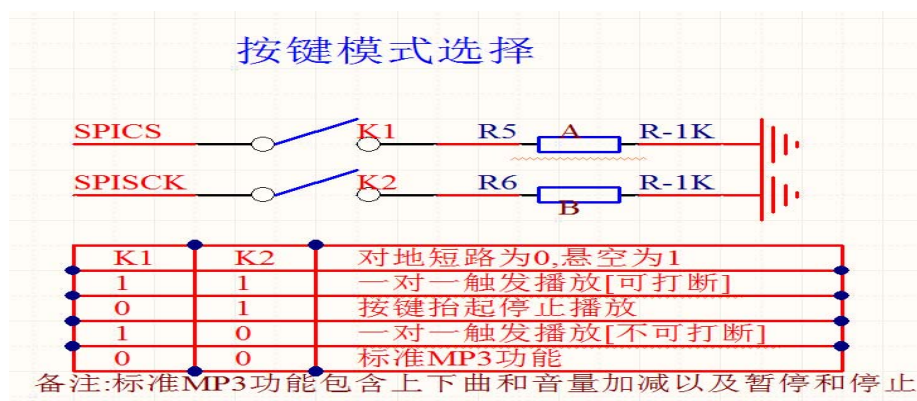
1、SPK1 和 SPK2 接到喇叭两端，可以不分正负极，注意：只允许接 2W 以下的喇叭

2、接功放或者耳机，直接接 DAC-R 和 DAC-L 两端，注意：共地（电源地）

5、模块调试说明

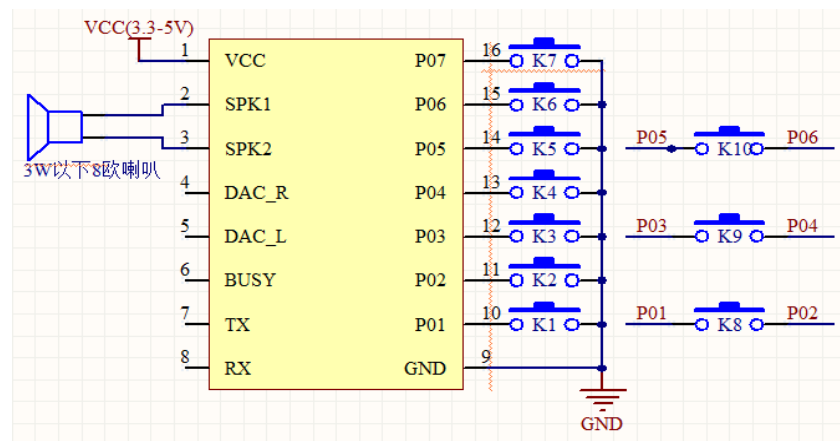
(1)、我们的模块是默认插上 USB 线，就进入下载模式。当用户更新完语音之后，触发任何一个 IO 口，即可退出下载模式，恢复正常的工作状态

(2)、模块发货前，都已经下载测试过声音的，客户拿到模块前，先不急着下载声音，先通过接小喇叭或者接功放或耳机的方式，插 USB（可当供电），把触发口对地短路触发，有声音发出后再换声音，注意，更新声音浏览打开完成后，再一次性更新，因为每一次更新模块内存都会擦除干净后烧写

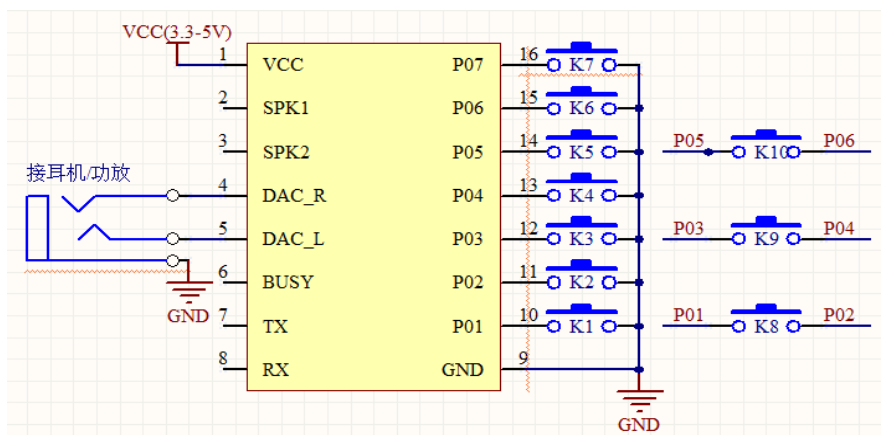


二、模块接线图

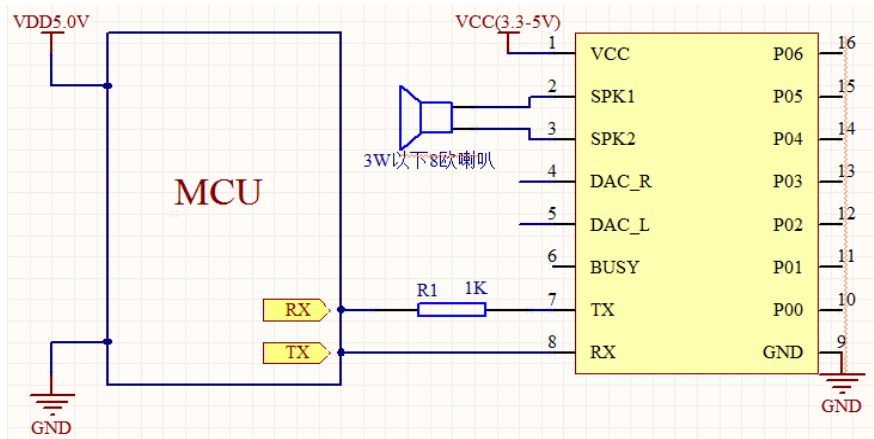
按键模式—接小喇叭



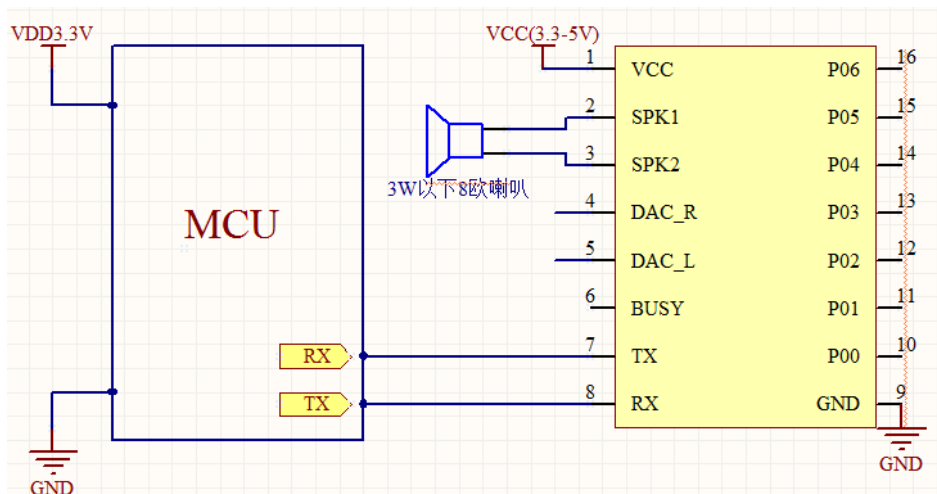
按键模式—接功放或耳机



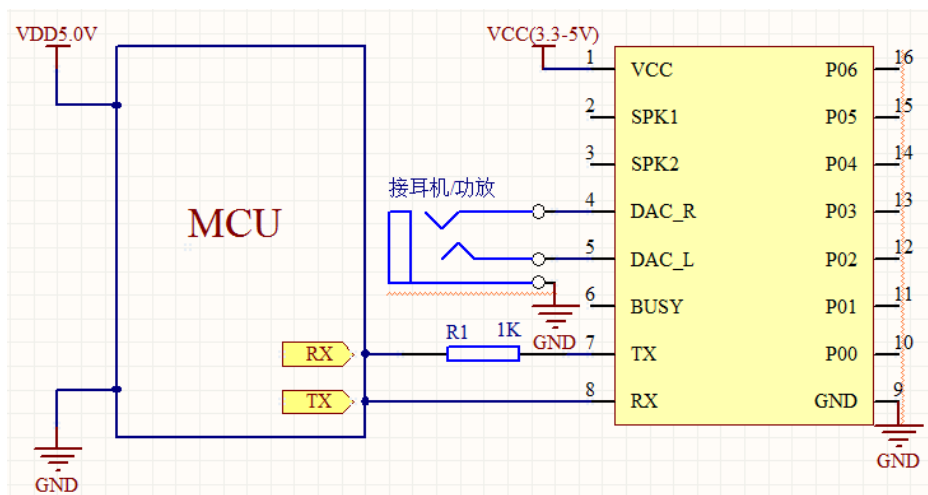
单片机 5V—接小喇叭



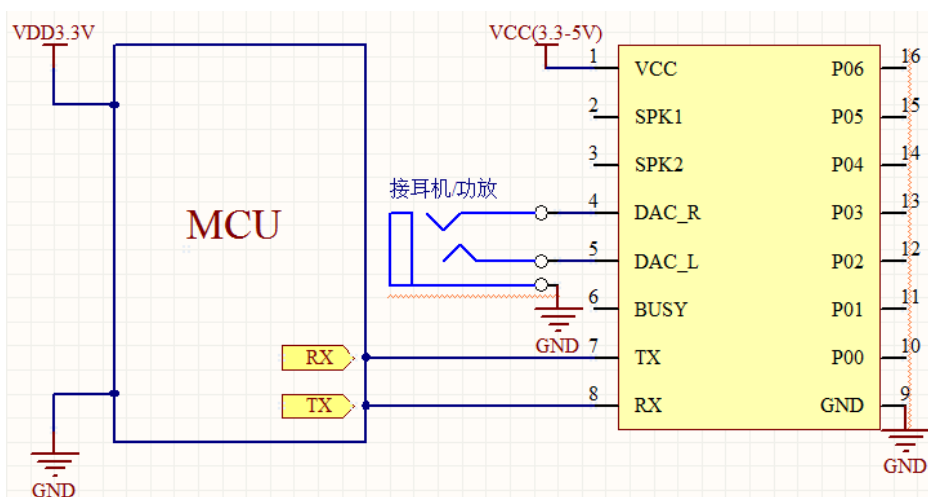
单片机 3.3V—接小喇叭



单片机 5V—耳机或功放



单片机 5V—耳机或功放

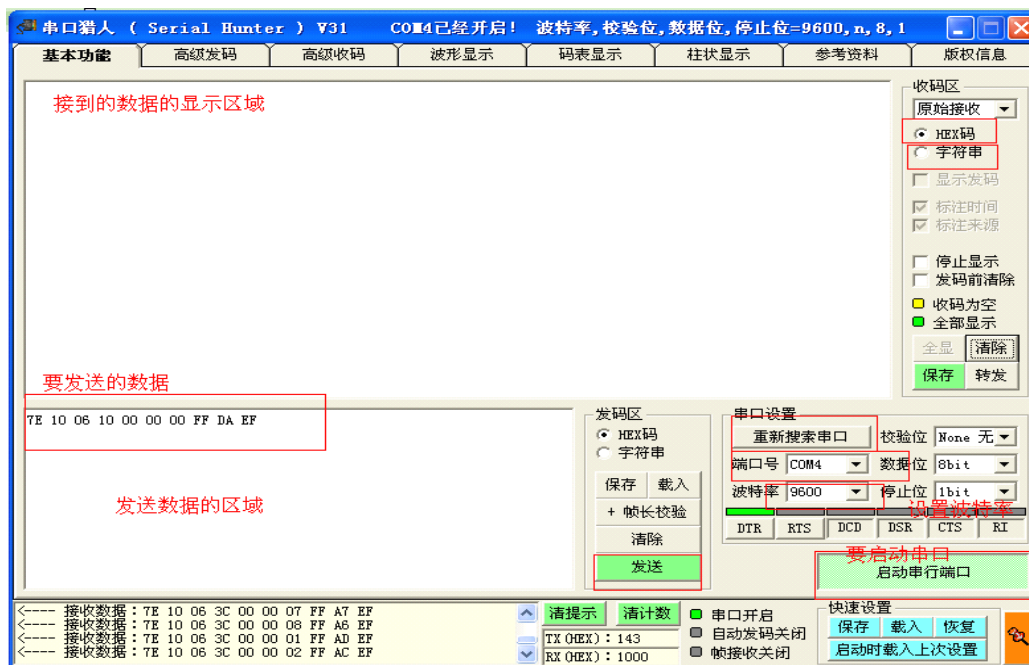


串口调试助手进行测试	发送的命令[带校验]	发送的命令[不带校验]	备注
[下一首]	7E FF 06 01 00 00 00 FE FA EF	7E FF 06 01 00 00 00 EF	
[上一首]	7E FF 06 02 00 00 00 FE F9 EF	7E FF 06 02 00 00 00 EF	
[指定曲目]	7E FF 06 03 00 00 01 FE F7 EF	7E FF 06 03 00 00 01 EF	指定第一首播放
	7E FF 06 03 00 00 02 FE F6 EF	7E FF 06 03 00 00 02 EF	指定第二首
	7E FF 06 03 00 00 0A FE EE EF	7E FF 06 03 00 00 0A EF	指定第 10 首
[指定音量]	7E FF 06 06 00 00 1E FE D7 EF	7E FF 06 06 00 00 1E EF	指定音量为 30 级
[指定 EQ]	7E FF 06 07 00 00 01 FE F3 EF	7E FF 06 07 00 00 01 EF	保留
[循环播放曲目]	7E FF 06 08 00 00 01 FE F2 EF	7E FF 06 08 00 00 01 EF	循环播放第一首
	7E FF 06 08 00 00 02 FE F1 EF	7E FF 06 08 00 00 02 EF	循环播放第二首
	7E FF 06 08 00 00 0A FE E9 EF	7E FF 06 08 00 00 0A EF	循环播放第十首
[指定播放设备]	7E FF 06 09 00 00 01 FE F1 EF	7E FF 06 09 00 00 01 EF	指定播放 FLASH
[进入睡眠模式]	7E FF 06 0A 00 00 00 FE F1 EF	7E FF 06 0A 00 00 00 EF	
[唤醒睡眠]	7E FF 06 0B 00 00 00 FE F0 EF	7E FF 06 0B 00 00 00 EF	
[模块复位]	7E FF 06 0C 00 00 00 FE EF EF	7E FF 06 0C 00 00 00 EF	
[播放]	7E FF 06 0D 00 00 00 FE EE EF	7E FF 06 0D 00 00 00 EF	
[暂停]	7E FF 06 0E 00 00 00 FE ED EF	7E FF 06 0E 00 00 00 EF	
停止播放	7E FF 06 16 00 00 00 FE E5 EF	7E FF 06 16 00 00 00 EF	停止软件解码
组合播放	7E FF 09 21 00 05 01 02 03 04 FE C8 EF	7E FF 09 21 00 05 01 02 03 04 EF	组合播放 5、1、2、3、4
组合播放	7E FF 0C 21 00 05 01 02 03 04 06 07 08 FE B0 EF	7E FF 0C 21 00 05 01 02 03 04 06 07 08 EF	组合播放 5、1、2、3、4、 6、7、8
带音量播放	7E FF 06 22 00 1E 01 FE BA EF	7E FF 06 22 00 1E 01 EF	30 级音量播放第 1 曲
	7E FF 06 22 00 0F 01 FE C9 EF	7E FF 06 22 00 0F 01 EF	15 级音量播放第 1 曲
	7E FF 06 22 00 0F 02 FE C8 EF	7E FF 06 22 00 0F 02 EF	15 级音量播放第 2 曲
带指定文件夹曲目播放	7E FF 06 30 00 01 01 FE C9 EF	7E FF 06 30 00 01 01 EF	FOLDER1 的第 1 曲
	7E FF 06 30 00 02 01 FE C8 EF	7E FF 06 30 00 02 01 EF	FOLDER2 的第 1 曲
	7E FF 06 30 00 03 01 FE C7 EF	7E FF 06 30 00 03 01 EF	FOLDER3 的第 1 曲

	7E FF 06 30 00 03 03 FE C5 EF	7E FF 06 30 00 03 03 EF	FOLDER3 的第 3 曲
指定文件夹曲目循环播	7E FF 06 31 00 03 03 FE C4 EF	7E FF 06 31 00 03 03 EF	FOLDER3 第 3 曲循环播放
	7E FF 06 31 00 01 03 FE C6 EF	7E FF 06 31 00 01 03 EF	FOLDER1 的第 3 曲循环播 放
指定文件夹循环播放	7E FF 06 32 00 03 01 FE C5 EF	7E FF 06 32 00 03 01 EF	FOLDER3 的第 1 曲开始循 环播放
查询当前状态	7E FF 06 42 00 00 00 FE B9 EF	7E FF 06 42 00 00 00 EF	
[查询音量]	7E FF 06 43 00 00 00 FE B8 EF	7E FF 06 43 00 00 00 EF	
[查询当前 EQ]	7E FF 06 44 00 00 00 FE B7 EF	7E FF 06 44 00 00 00 EF	
当前播放模式	7E FF 06 47 00 00 00 FE B4 EF	7E FF 06 47 00 00 00 EF	
FLASH 总文件数	7E FF 06 49 00 00 00 FE B2 EF	7E FF 06 49 00 00 00 EF	
FLASH 当前文件夹曲目 指针	7E FF 06 4D 00 00 00 FE AE EF	7E FF 06 4D 00 00 00 EF	
当前在线的文件夹	7E FF 06 60 00 00 00 FE 9B EF	7E FF 06 60 00 00 00 EF	查询在线的文件夹
当前的文件夹指针	7E FF 06 61 00 00 00 FE 9A EF	7E FF 06 61 00 00 00 EF	查询当前播放的文件夹
查指定文件夹曲目总数	7E FF 06 62 00 00 01 FE 98 EF	7E FF 06 62 00 00 01 EF	查询 FOLDER1 文件总数

三、测试方法

1、串口软件的操作

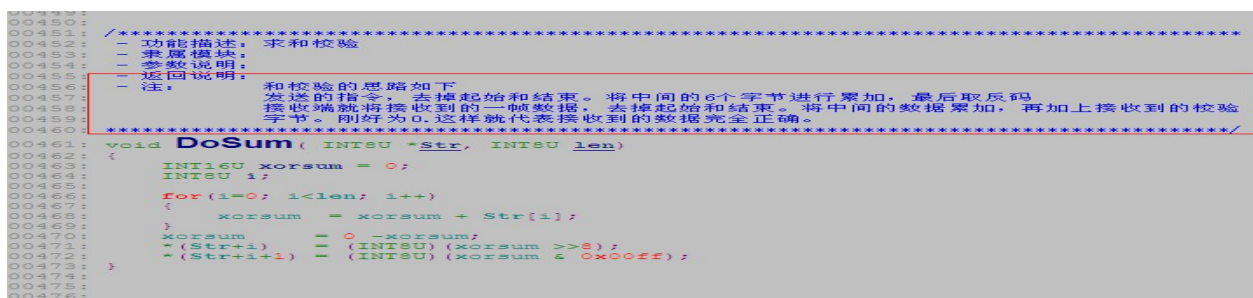


(1)、首先安装资料里的“串口猎人”软件，打开软件，首先要搜索串口，找到指定的端口之后，指定“波特率”，我们的模块默认的波特率为9600，最后就是“启动串行端口”，这样软件就配置好了。这里有两个概念需要明确一下第一是“HEX码”，我们默认是这个，这个是用来显示数据的。所以必须设置这里第二是“字符串”，这个是用来显示打印字符的，我们这里用不到。

(3)、软件配置OK之后，将需要的指令复制到发送区域，即可。具体的指令请参照模块的数据手册

(4)、如果模块的数据手册没有的测试指令的话，请自行计算，尤其需要注意的是校验和这两个字节如何计算不对的话，模块是不接受指令的。

校验码的计算方式：



$$0 = 24 + x \quad \text{类比} \quad 0000\ 0000\ (0) = 0010\ 0100\ (24) + 1101\ 1011\ (DB+1)$$

附录一：SPI-FLASH 容量与音频长度对照表

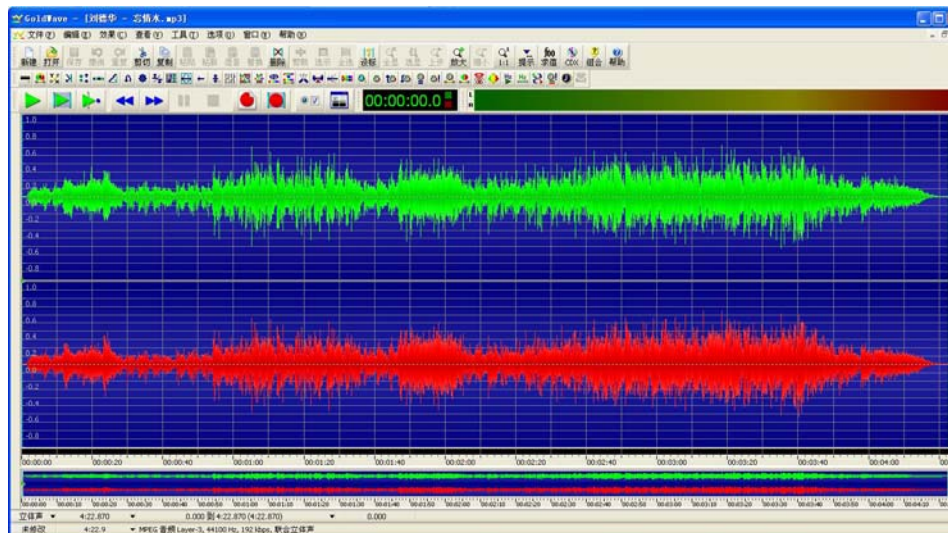
附表 1-1 MP3-FLASH-16p 模块 FLASH 容量与音频时间长度对换表：(单位：S)

容量 码率	4Mbits	8Mbits	16MBit	32MBit	64MBit	
16Kbps	252	505	1011	2022	4045	
24Kbps	163	327	654	1309	2618	
32Kbps	113	226	453	906	1812	
64Kbps	59	119	239	477	955	
96Kbps	41	81	162	325	651	
128Kbps	31	61	123	246	493	
160Kbps	24	49	97	194	389	
192Kbps	20	40	81	161	323	
256Kbps	15	30	60	120	241	
320Kbps	11	23	47	95	191	

注：MP3 文件大小决定于码率，与采样率无关。语音播报建议使用 16Kbps~64Kbps，音乐播放建议使用 32Kbps~96Kbps。

三、码率的转换方式

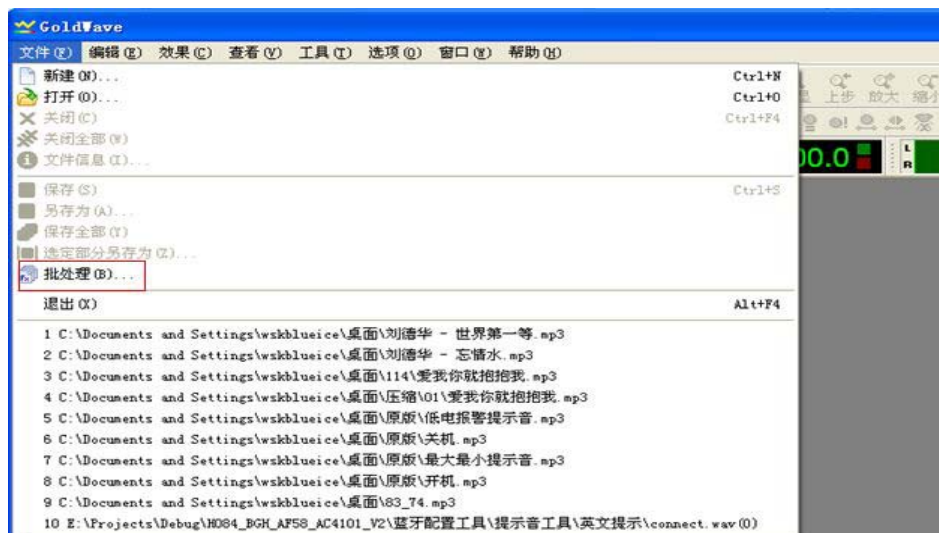
1、针对SPIFLASH 的小容量、稳定的特点，我们开发了MP3 – FLASH-16P 模块。直接通过手机的Microusb 线更新语音，但是对于常见的MP3 文件，大都是4M 字节左右，使用SPIFLASH，空间就显得很吃力了。但是我们一般作为语音播报和提示场合，根本就不需要很高的采样率



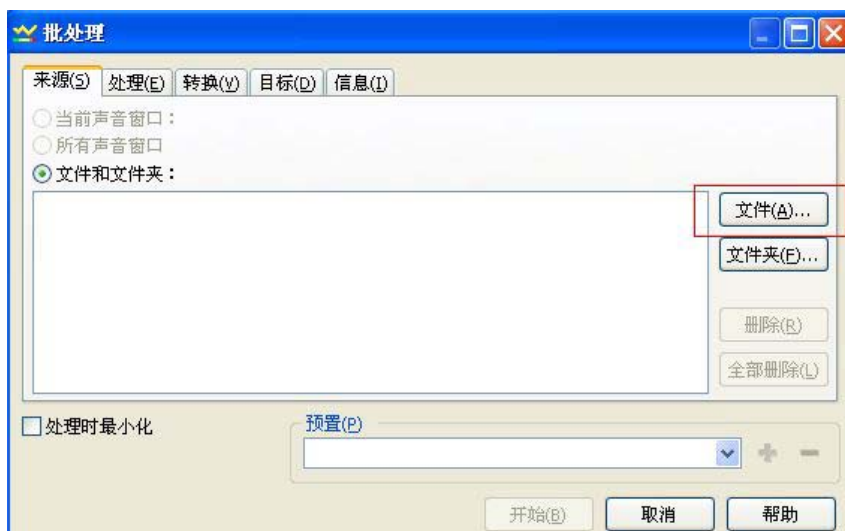
从上图的左下角，我们可以看到“世界第一等.MP3”的采样率高达44100HZ。比特率为256KBS。这个参数，说明当前的歌曲音质是相当好的，所以占用了4.5M 的空间。

2、但是实际我们根本不需要这么高的音质，这时就可以压缩了。如下：

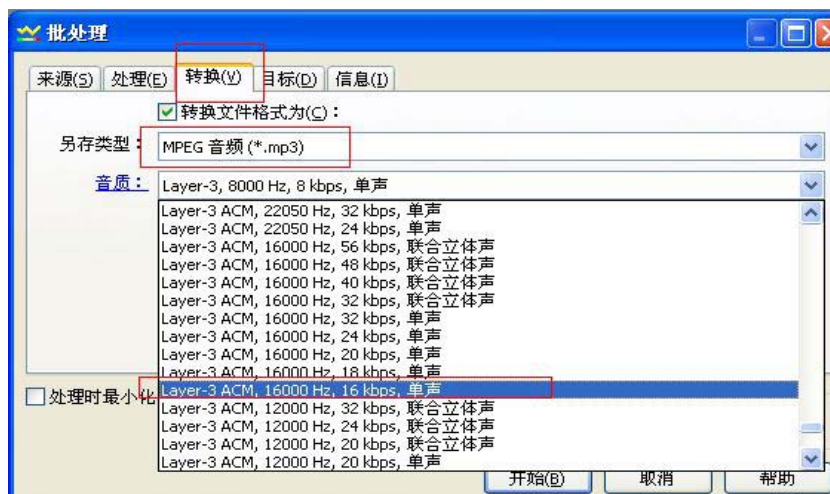
使用“GoldWave”这款软件。



点击批处理，



添加文件



选择“转换”，设置采样率为16000KHZ，比特率为16KBS。



再指定一下转换后存放文件的路径即可

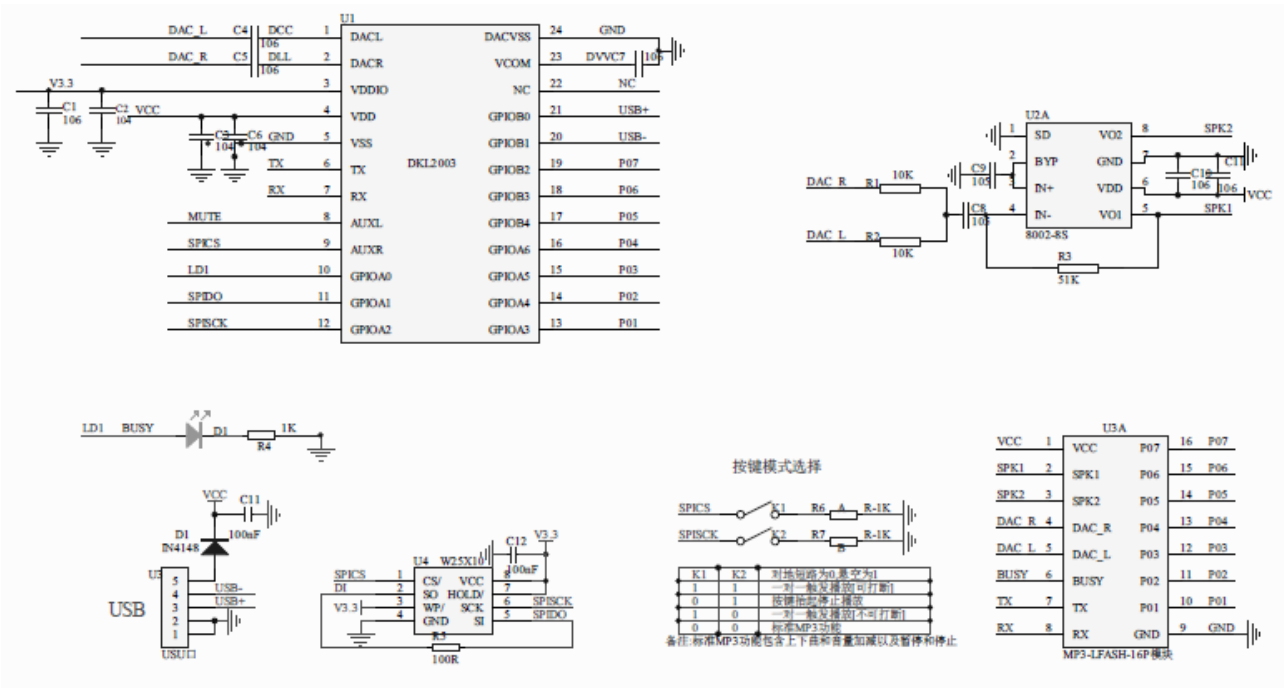


压缩之后，4.5M 的文件变成 507K 了。步骤就是这样

备注：

- 1、如果是wav 文件的话，也可以使用这个软件转换成MP3
- 2、转换之后的效果，用户可以直接先在电脑上面试听一下效果，电脑上播放的效果，和我们模块播放的效果是一致的
- 3、如果觉得音质不好，可以适当的增加采样率和比特率这两个参数。可以自己尝试一下
- 4、如果需要修改音源的音量，以及剪裁音源，都可以使用这款软件

四、模块的原理图



四、程序范例

程序范例：串口指定播放

```

/*****
- 实现功能：实现芯片上电分别指定播放第一曲和第二曲，基本的程序供用户测试
- 日期      ：2013-05-06
- 运行环境：STC   晶振：11.0592M   波特率:9600
- 备注      ：在普中科技的 51 开发板上调试 OK --- STC89C516RD+
1、该测试程序必须是模块或者芯片方案中有设备在线，譬如 U 盘、TF 卡、FLASH
*****/

#include "REG52.h"

#define COMM_BAUD_RATE  9600      //串口波特率
#define OSC_FREQ        11059200 //运行晶振：11.05926MHZ
static INT8U Send_buf[10] = {0};

void Delay_Ms(INT32U z)
{
    INT32U x=0 , y=0;
    for(x=110 ; x>0 ;x--)
        for(y=z; y>0;y-- );
}

/*****
- 功能描述：  串口 1 初始化
- 注：        设置为 9600 波特率
*****/

void Serial_init(void)
{
    TMOD = 0x20;           // 设置 T1 为波特率发生器
    SCON = 0x50;           // 0101,0000 8 位数据位，无奇偶校验
    PCON = 0x00;           //PCON=0;
    TH1=256-(OSC_FREQ/COMM_BAUD_RATE/32/12);//设置为 9600 波特率
    TL1=256-(OSC_FREQ/COMM_BAUD_RATE/32/12);
    TR1    = 1;           //定时器 1 打开
    REN    = 1;           //串口 1 接收使能
    ES     = 1;           //串口 1 中断使能
}

void Uart_PutByte(INT8U ch)
{
    SBUF = ch;
    while(!TI){;}
    TI = 0;
}

/*****
- 功能描述：  串口向外发送命令[包括控制和查询]
- 参数说明：  CMD:表示控制指令，请查阅指令表，还包括查询的相关指令
               feedback:是否需要应答[0:不需要应答，1:需要应答]
               data:传送的参数
*****/
```

```
void SendCmd(INT8U len)
```

```
{
    INT8U i = 0 ;
    Uart_PutByte(0x7E); //起始
    for(i=0; i<len; i++)//数据
    {
        Uart_PutByte(Send_buf[i]) ;
    }
    Uart_PutByte(0xEF) ;//结束
}
```

```
/******
```

- 功能描述：求和校验
- 和校验的思路如下：

发送的指令，去掉起始和结束。将中间的 6 个字节进行累加，最后取反码。接收端就将接收到的一帧数据，去掉起始和结束。将中间的数据累加，再加上接收到的校验字节。刚好为 0.这样就代表接收到的数据完全正确。

```
*****/
```

```
void DoSum( INT8U *Str, INT8U len)
```

```
{
    INT16U xorsum = 0;
    INT8U i;
    for(i=0; i<len; i++)
    {
        xorsum  = xorsum + Str[i];
    }
    xorsum      = 0 -xorsum;
    *(Str+i)    = (INT8U)(xorsum >>8);
    *(Str+i+1) = (INT8U)(xorsum & 0x00ff);
}
```

```
void Uart_SendCMD(INT8U CMD ,INT8U feedback , INT16U dat)
```

```
{
    Send_buf[0] = 0xff;    //保留字节
    Send_buf[1] = 0x06;    //长度
    Send_buf[2] = CMD;     //控制指令
    Send_buf[3] = feedback; //是否需要反馈
    Send_buf[4] = (INT8U)(dat >> 8); //datah
    Send_buf[5] = (INT8U)(dat);    //datal
    DoSum(&Send_buf[0],6);        //校验
    SendCmd(8);                   //发送此帧数据
}
```

```
void main()
```

```
{
    Serial_init() ;//串口寄存器的初始化设置
    Uart_SendCMD(0x03 , 0 , 0x01) ;//播放第一首
    Delay_Ms(1000) ;//延时大概 6S
    Uart_SendCMD(0x03 , 0 , 0x02) ;//播放第二首
    Delay_Ms(1000) ;//延时大概 6S
    Uart_SendCMD(0x03 , 0 , 0x04) ;//播放第四首
    while(1) ;
}
```