目 录

一、芯片的硬件说明

1、芯片的供电

芯片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、芯片的指示灯[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、芯片的指示灯[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、音频出输说明

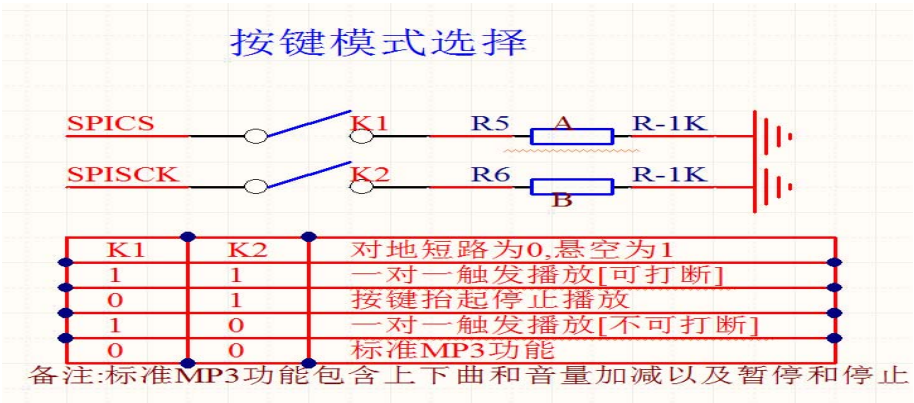
1、SPK1 和 SPK2 接到喇叭两瑞，可以不分正负极，注意：只允许接 2W 以下的喇叭

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、芯片调试说明

(1)、我们的芯片是默认插上 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)、调试芯片先从易到难，先从简到到复杂，按键保证正常后再调串口，串口调试手调试正常后，再调单片机，TF 卡没声音，先通过 USB 插电脑，看能不能读到 TF 卡盘符



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

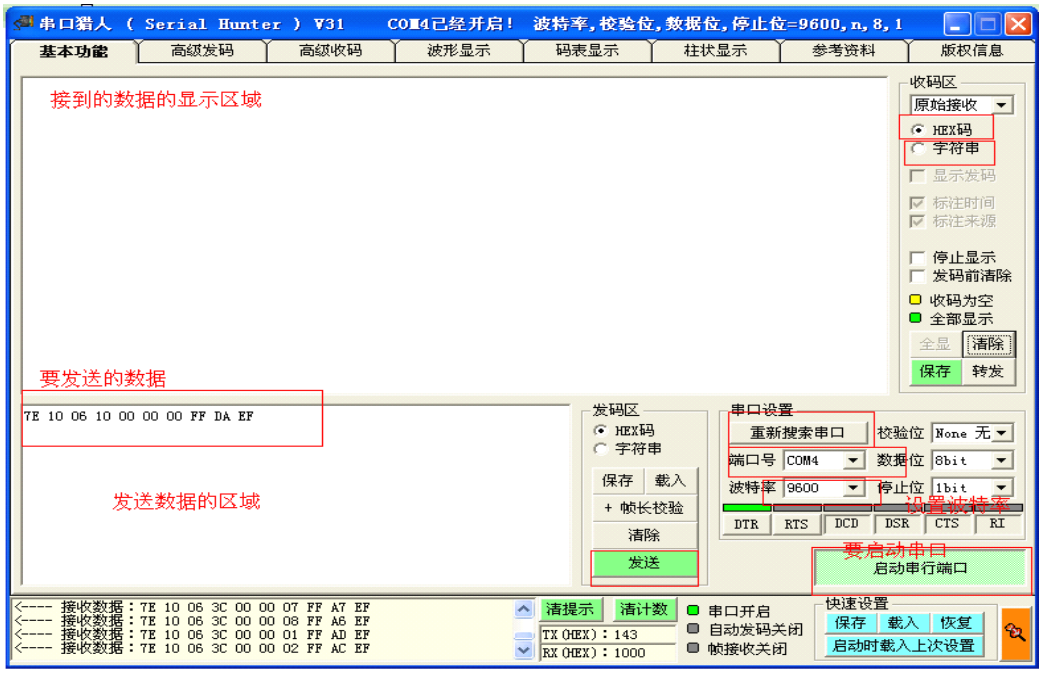
[下一首]	7E FF 06 01 00 00 00 FE FA EF	7E FF 06 01 00 00 00 EF	
[上一首]	7E FF 06 02 00 00 00 FE F9 EF	7E FF 06 02 00 00 00 EF	
[指定曲目]	7E FF 06 03 00 00 01 FE F7 EF	7E FF 06 03 00 00 01 EF	指定第一首播放
	7E FF 06 03 00 00 02 FE F6 EF	7E FF 06 03 00 00 02 EF	指定第二首
	7E FF 06 03 00 00 0A FE EE EF	7E FF 06 03 00 00 0A EF	指定第 10 首
音量加	7E FF 06 04 00 00 00 FE F7 EF	7E FF 06 04 00 00 00 EF	
音量减	7E FF 06 05 00 00 00 FE F6 EF	7E FF 06 05 00 00 00 EF	
[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
	7E FF 06 08 00 01 01 FE F1 EF	7E FF 06 08 00 01 01 EF	循环播放 FLASH 的 FOLDER1 的第一曲
	7E FF 06 08 00 02 01 FE F0 EF	7E FF 06 08 00 02 01 EF	循环播放 FLASH 的 FOLDER2 的第一曲
[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 UDISK
	7E FF 06 09 00 00 02 FE F0 EF	7E FF 06 09 00 00 02 EF	指定播放 TF
	7E FF 06 09 00 00 04 FE EE EF	7E FF 06 09 00 00 04 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	
[芯片复位]	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	
[指定文件夹文件名]	7E FF 06 0F 00 01 01 FE EA EF	7E FF 06 0F 00 01 01 EF	指定为"01"的文件夹，曲目为"001"
	7E FF 06 0F 00 01 02 FE E9 EF	7E FF 06 0F 00 01 02 EF	指定为"01"的文件夹，曲目为"002"

支持 1000 首	7E FF 06 14 00 10 FF FD D8 EF	7E FF 06 14 00 10 FF EF	指定为"01"的文件夹，曲目为"0255"
	7E FF 06 14 00 17 CF FE 01 EF	7E FF 06 14 00 17 CF EF	指定为"01"的文件夹，曲目为"1999"
	7E FF 06 14 00 C0 01 FE 26 EF	7E FF 06 14 00 C0 01 EF	指定为"12"的文件夹，曲目为"0001"
	7E FF 06 14 00 C0 FF FD 28 EF	7E FF 06 14 00 C0 FF EF	指定为"12"的文件夹，曲目为"0255"
	7E FF 06 14 00 C7 CF FD 51 EF	7E FF 06 14 00 C7 CF EF	指定为"12"的文件夹，曲目为"1999"
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
指定文件夹循环播放	7E FF 06 17 00 02 00 FE E2 EF	7E FF 06 17 00 02 00 EF	指定 02 文件夹循环播放
	7E FF 06 17 00 01 00 FE E3 EF	7E FF 06 17 00 01 00 EF	指定 01 文件夹循环播放
随机播放	7E FF 06 18 00 00 00 FE E3 EF	7E FF 06 18 00 00 00 EF	随机播放
单曲循环播放	7E FF 06 19 00 00 00 FE E2 EF	7E FF 06 19 00 00 00 EF	单曲循环播放开启
	7E FF 06 19 00 00 01 FE E1 EF	7E FF 06 19 00 00 01 EF	单曲循环播放关闭
DAC 的设置	7E FF 06 1A 00 00 00 FE E1 EF	7E FF 06 1A 00 00 00 EF	开 DAC
	7E FF 06 1A 00 00 01 FE E0 EF	7E FF 06 1A 00 00 01 EF	关 DAC
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

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	
U 盘总文件数	7E FF 06 47 00 00 00 FE B4 EF	7E FF 06 47 00 00 00 EF	当前设备的总文件数
TF 总文件数	7E FF 06 48 00 00 00 FE B3 EF	7E FF 06 48 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	
U 盘当前曲目	7E FF 06 4B 00 00 00 FE B0 EF	7E FF 06 4B 00 00 00 EF	当前播放的曲目
TF 当前曲目	7E FF 06 4C 00 00 00 FE AF EF	7E FF 06 4C 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	
查询文件夹曲目总数	7E FF 06 4E 00 00 01 FE AC EF	7E FF 06 4E 00 00 01 EF	查询 01 文件夹或者 FOLDER1 的总曲目数
查询 TF 或者 U 盘总文件夹数	7E FF 06 4F 00 00 00 FE AC EF	7E FF 06 4F 00 00 00 EF	只支持 TF 卡和 U 盘
当前的文件夹指针[FLASH]	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[支持 FLASH]

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：

```
/*****************************************************************************************************
  - 功能描述:  求和校验
  - 隶属模块:
  - 参数说明:
  - 返回说明:
  - 注:
                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);
}
```

0 = 24 + x 类比 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

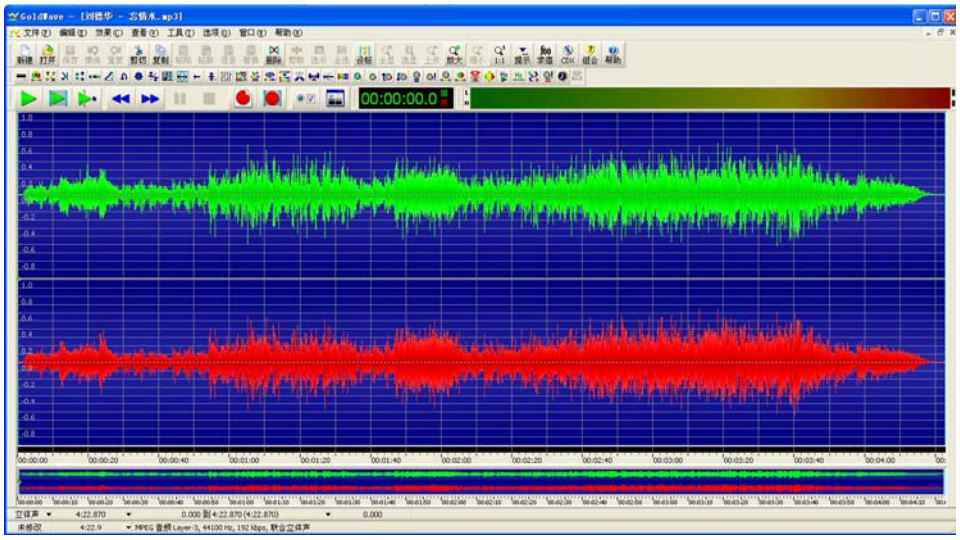
附表 1-1 YX5300-24SS FLASH 容量与音频时间长度对换表：(单位：S)

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、争对SPIFLASH 的小容量、稳定的特点，我们开发了YX5300-24SS。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



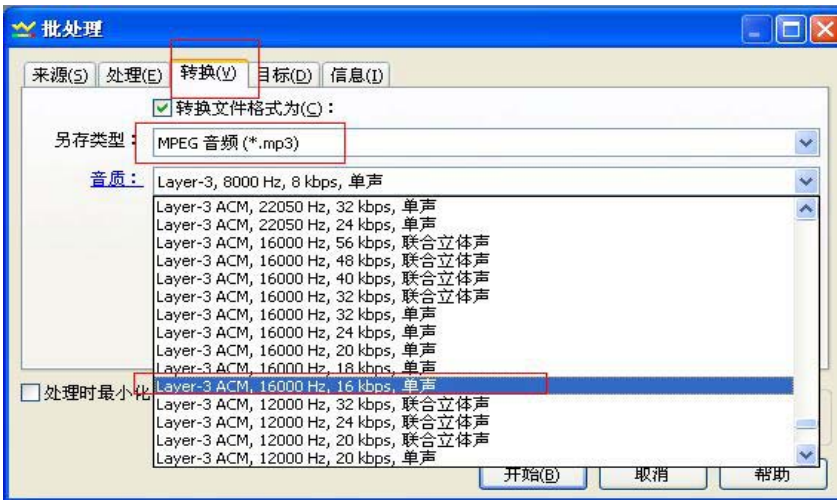
从上图的左下角，我们可以看到“世界第一等.MP3”的采样率高达44100HZ。比特率为256KBS。这个参数，说明当前的歌曲音质是相当好的，所以占用了4.5M 的空间。

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:

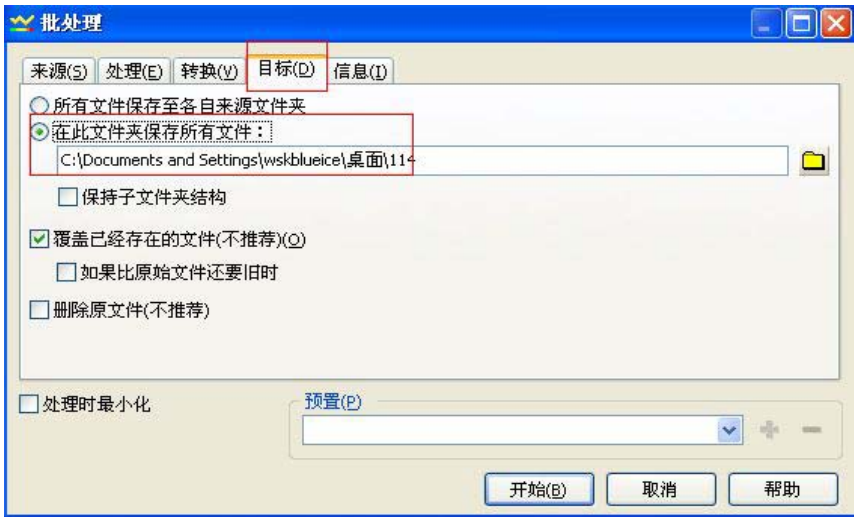
使用“GoldWave”这款软件。

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：

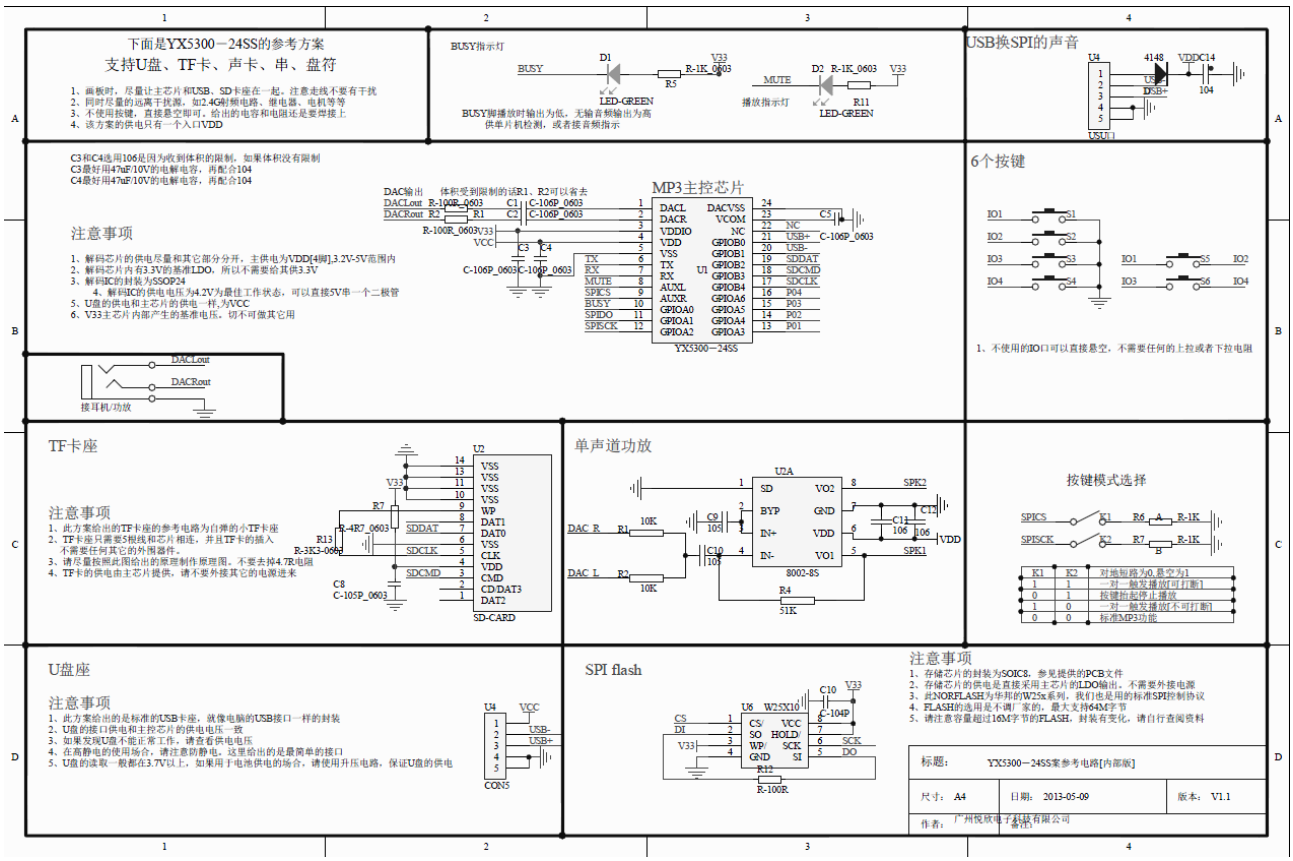
1、如果是wav 文件的话，也可以使用这个软件转换成MP3

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，电脑上面播放的效果，和我们芯片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、芯片的原理图


下面是YX5300－24SS的参考方案
支持U盘、TF卡、声卡、串、盘符
1、画板时，尽量让主芯片和USB、SD卡座在一起，注意走线不要有干扰
2、同时尽量的远离干扰源，如2.4G射频电路、继电器、电机等等
3、不使用按键，直接悬空即可，给出的电容和电阻还是要焊接上
4、该方案的供电只有一个入口VDD
BUSY指示灯
BUSY脚播放时输出为低，无输音频输出为高
供单片机检测，或者接音频指示
播放指示灯
USB换SPI的声音
C3和C4选用106是因为收到体积的限制，如果体积没有限制
C3最好用47uF/10V的电解电容，再配合104
C4最好用47uF/10V的电解电容，再配合104
注意事项
1、解码芯片的供电尽量和其它部分分开，主供电为VDD[4脚],3.2V-5V范围内
2、解码芯片内有3.3V的基准LDO，所以不需要给其供3.3V
3、解码IC的封装为SSOP24
4、解码IC的供电电压为4.2V为最佳工作状态，可以直接5V串一个二极管
5、U盘的供电和主芯片的供电一样,为VCC
6、V33主芯片内部产生的基准电压，切不可做其它用
DAC输出 体积受到限制的话R1、R2可以省去
MP3主控芯片
YX5300－24SS
6个按键
1、不使用的IO口可以直接悬空，不需要任何的上拉或者下拉电阻
接耳机/功放
TF卡座
注意事项
1、此方案给出的TF卡座的参考电路为自弹的小TF卡座
2、TF卡座只需要5根线和芯片相连，并且TF卡的插入
不需要任何其它的外围器件。
3、请尽量按照此图给出的原理制作原理图。不要去掉4.7R电阻
4、TF卡的供电由主芯片提供，请不要外接其它的电源进来
SD-CARD
单声道功放
8002-8S
按键模式选择
U盘座
注意事项
1、此方案给出的是标准的USB卡座，就像电脑的USB接口一样的封装
2、U盘的接口供电和主控芯片的供电电压一致
3、如果发现U盘不能正常工作，请查看供电电压
4、在高静电的使用场合，请注意防静电，这里给出的是最简单的接口
5、U盘的读取一般都在3.7V以上，如果用于电池供电的场合，请使用升压电路，保证U盘的供电
CON5
SPI flash
W25X10
注意事项
1、存储芯片的封装为SOIC8，参见提供的PCB文件
2、存储芯片的供电是直接采用主芯片的LDO输出，不需要外接电源
3、此NORFLASH为华邦的W25x系列，我们也是用的标准SPI控制协议
4、FLASH内选用是不满厂家的，最大支持64M字节
5、请注意容量超过16M字节的FLASH，封装有变化，请自行查阅资料
标题： YX5300－24SS案参考电路[内部版]
尺寸： A4
日期： 2013-05-09
版本： V1.1
作者： 广州悦欣电子科技有限公司

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) ;
}
```