

单片机实现 MP3 播放的方法

有一个东西你一定听说过或用过，那就 MP3 播放器。MP3 播放器以其小巧的体积、强大的功能、优异的音质倍受人们的青睐。如果把它嵌入到我们的单片机系统中，实现音频输出，那么对系统的增色是不言而喻的。单独拿单片机来说，要解码 MP3 文件，是不可能的，因为从处理速度和资源各个方面都是不能满足要求的。所以要依赖于专用 MP3 解码芯片，而单片机要作的就是对其进行控制。这里我们围绕芬兰 VLSI 公司出品的 VS1003 来进行解 MP3 的实现方法。

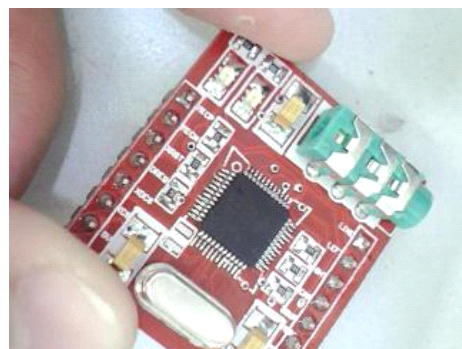
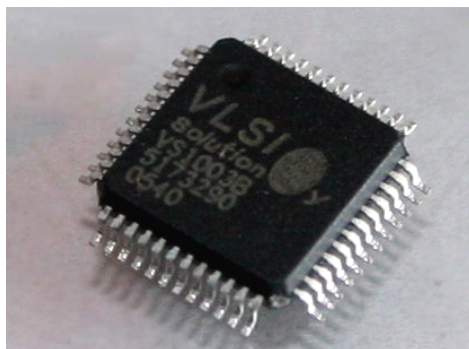
1、VS1003 芯片

1) 芯片简介

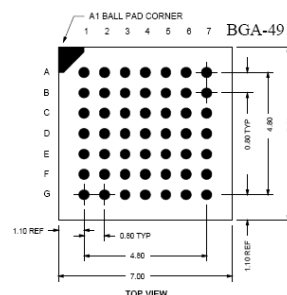
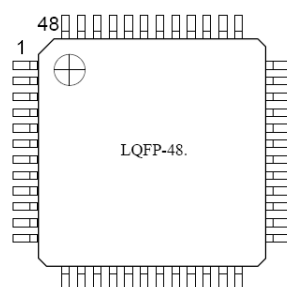
VS1003是由荷兰VLSI公司出品的一款单芯片的MP3/WMA/MIDI音频解码和ADPCM编码芯片，其拥有一个高性能低功耗的DSP处理器核VS_DSP，5K的指令RAM，0.5K的数据RAM，串行的控制和数据输入接口，4个通用IO口，一个UART口；同时片内带有一个可变采样率的ADC、一个立体声DAC以及音频耳机放大器。

VS1003通过一个串行接口来接收输入的比特流，它可以作为一个系统的从机。输入的比特流被解码，然后通过一个数字竟是控制器到达一个18位过采样多位 $\epsilon - \Delta$ DAC。通过串行总线控制解码器。除了基本的解码，在用户RAM中它还可以做其他特殊应用，例如DSP音效处理。

2) 芯片实物与 VS1003 MP3 模块

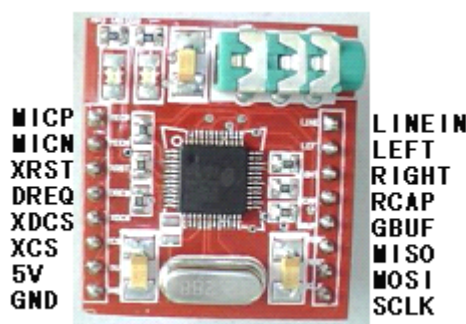


3) 芯片封装



以下的讲述都是针对于 LQFP-48 封装的。

4) MP3 模块接口



5) VS1003 特性

- 1.能解码 MPEG1 与 MPEG2 音频层 III (CBR+VBR+ABR) ;WMA 4.0/4.1/7/8/9
5~384kbps 所有流文件;WAV (PCM+IMA AD-PCM);产生 MIDI/SP-MIDI 文件。
- 2.对话筒输入或线路输入的音频信号进行 IMA ADPCMM 编码
- 3.支持 MP3 和 WAVV 流
- 4.高低音控制
- 5.单时钟 12~13MHz
- 6.内部 PLLL 锁相环时钟倍频器
- 7.低功耗
- 8.内含高性能片上立体声数模转换器,两声道间无相位差
- 9.内含能驱动 30 欧负载的耳机驱动器
- 10.模拟, 数字, I/O 单独供电
- 11.为用户代码和数据准备的 5.5KB 片上 RAM
- 12.串行的控制/数据接口
- 13.可被用作微处理器的从机
- 14.特殊应用的 SPI Flash 引导
- 15.借高度用途的 UART 接口
- 16.新功能可以通过软件和 4 GPIO 添加

6) VS1003 的引脚定义

管脚名称	LQFP-48	管脚类型	管脚功能
MICP	1	AI	同相差分话筒输入, 自偏压
MICN	2	AI	反相差分话筒输入, 自偏压
XRESET	3	DI	低电平有效, 异步复位端
DGND0	4	DGND	处理器核与 I/O 地

CVDD0	5	CPWR	处理器核电源
IOVDD0	6	IOPWR	I/O 电源
CVDD1	7	CPEW	处理器核电源
DREQ	8	DO	数据请求，输入总线
GPIO/DCLK	9	DIO	通用 I/O2 /串行数据总线时钟
GPIO3/SDATA	10	DIO	通用 I/O3 /串行数据总线数据
XDCS/BSYNC	13	DI	数据片选端/字节同步
IOVDD1	14	IOPWR	I/O 电源
VCO	15	DO	时钟压控振荡器VCO 输出
DGND1	16	DGND	处理器核与I/O 的地
XTALO	17	AO	晶振输出
XTALI	18	AI	晶振输入
IOVDD2	19	IOPWR	I/O 电源
DGND2	20	DGND	处理器核与I/O 地
DGND3	21	DGND	处理器核与I/O 地
DGND4	22	DGND	处理器核与I/O 地
XCS	23	DI	片选输入，低电平有效
CVDD2	24	CPWR	处理器核电源
RX	26	DI	UART接收口，不用时接IOVDD
TX	27	DO	UART发送口
SCLK	28	DI	串行总线的时钟
SI	29	DI	串行输入
SO	30	DO3	串行输出
CVDD3	31	CPWR	处理器核电源
TEST	32	DI	保留做测试，连接至IOVDD
GPIO0/SPIBOOT	33	DIO	通用I/O0 /SPIBOOT,使用100K 下拉电阻
GPIO1	34	DIO	通用I/O1
AGND0	37	APWR	模拟地，低噪声参考地
AVDD0	38	APWR	模拟电源
RIGHT	39	AO	右声道输出
AGND1	40	APWR	模拟地
AGND2	41	APWR	模拟地
GBUF	42	AO	公共地缓冲器
AVDD1	43	APWR	模拟电源
RCAP	44	AIO	基准滤波电容
AVDD2	45	APWR	模拟电源
LEFT	46	AO	左声道输出
AGND3	47	APWR	模拟地
LINE IN	48	AI	线路输入

7) VS1003 的功能寄存器

VS1003 共有 16 个 16 位的寄存器，地址分别为 0X0~0XF;除了模式寄存器 (MODE, 0X0) 和状态寄存器 (STATUS, 0X1) 在复位后的初始值分别为 0X800

和 0X3C 外，其余的寄存器在 VS1003 初始化后的值均为 0。下面将 VS1003 各寄存器逐一进行介绍。

1.MODE（地址：0X0 可读写）

bit0:SM_DIFF

SM_DIFF=0 正常音频相位

SM_DIFF=1 左声道反转

当 SM_DIFF 置位时，VS1003 将左声道反相输出，立体声输入将产生环绕效果，对于单声道输入将产生差分（反相）左/右声道信号。

bit1:SM_SETTOZERO

置零。

bit2:SM_RESET

SM_RESET=1，VS1003 软复位。软复位之后该位会自动清零。

bit3:SM_OUTOFWAV

SM_OUTOFWAV=1,停止 WAV 解码。

当你要中途停止 WAV、WMA 或者 MIDI 文件的解码时，置位 SM_OUTOFWAV，并向 VS1003 持续发送数据（对于 WAV 文件发送 0）直到将 SM_OUTOFWAV 清零;同时 SCI_HIDAT1 也将被清零。

bit4:SM_PDOWN

SM_PDOWN=1，软件省电模式，该模式不及硬件省电模式（可由 VS1003 的 XRESET 来激活）。

bit5:SM_TESTS

SM_TESTS=1，进入 SDI 测试模式。

bit6:SM_STREAM

SM_STREAM=1,使能 VS1003 的流模式。

bit7:SM_PLUSV

SM_PLUSV=1，MP3+V 解码使能。

bit8:SM_DACT

SM_DACT=0，SCLK 上升沿有效;SM_DACT=1，SCLK 下降沿有效。

bit9:SM_SDIORD

SM_SDIORD=0，SDI 总线字节数据 MSB 在前，即须先发送 MSB;

SM_SDIORD=1，SDI 总线字节数据 LSB 在前，即须先发送 LSB;

该位的设置不会影响 SCI 总线。

bit10:SM_SDISHARE

SM_SDISHARE=1,SDI 与 SCI 将共用一个片选信号（同时 SM_SDINEW=1），即将 XDCS 与 XCS 这两根信号线合为一条，能省去一个 IO 口。

bit11:SM_SDINEW

SM_SDINEW=1，VS1002 本地模式（新模式）。VS1003 在启动后默认进入该模式。（这里所说的模式指的是总线模式。）

bit12:SM_ADPCM

SM_ADPCM=1，ADPCM 录音使能。

同时置位 SM_ADPCM 和 SM_RESET 将使能 VS1003 的 IMA ADPCM 录音功能。

bit13:SM_ADPCM_HP

SM_ADPCM_HP=1，使能 ADPCM 高通滤波器。同时置位 SM_ADPCM_HP、SM_ADPCM 和 SM_RESET 将开启 ADPCM 录音用高通滤波器，对录音时的背景噪音有一定的抑制作用。

bit14:SM_LINE_IN

录音输入选择，SMLINE_IN=1,选择线入（line in）;SM_LINE_IN=0，选择麦克风输入（默认）。

2.SCI_STATUS(地址：0X1 可读写)

SCI_STATUS 为 VS1003 的状态寄存器，提供 VS1003 当前状态信息。

3.SCI_BASS(地址：0X2 可读写)

重音/高音设置寄存器。

VS1003 的内置的重音增强器 VSBE 是种高质量的重音增强 DSP 算法，能够最大限度的避免音频削波。当 SB_AMPLITUDE（bit:7~4）不为零时，重音增强器将使能。可以根据个人需要来设置 SB_AMPLITUDE。例如，SCI_BASS=0x00f6，即对 60Hz 以下的音频信号进行 15dB 的增强。当 ST_AMPLITUDE（bit:15~12）不为零时，高音增强将使能。例如，SCI_BASS=0x7a00，即 10kHz 以上的音频信号进行 10.5dB 的增强。

4.SCI_CLOCKF(地址: 0X3 可读写)

bit15~bit13:SC_MULT

时钟输入 XTALI 的倍频设置, 设置之后将启动 VS1003 内置的倍频器。

bit12~bit11:SC_ADD

用于在 WMA 流解码时给倍频器增加的额外的倍频值。

bit10~bit0:SC_FREQ

当 XTALI 输入的时钟不是 12.288M 时才需要设置该位段, 其默认值为 0, 即 VS1003 默认使用的是 12.228M 的输入时钟。

5.SCI_DECODE_TIME(地址: 0X4 可读写)

解码时间寄存器。当进行正确的解码时, 读取该寄存器可以获得当前的解码时长 (单位为秒)。可以更改该寄存器的值, 但是新值须要对该寄存器进行两次写操作。在每次软件复位或是 WAV(PCM、IMA ADPCM、WMA、MIDI)解码开始与结束时 SCI_DECODE_TIME 的值将清零。

6.SCI_AUDATA(地址: 0X5 可读写)

当进行正确的解码时, 该寄存器的值为当前的采样率 (bit:15~bit1) 和所使用的声道 (bit0)。采样率须为 2 的倍数; bit0=0, 单声道数据, bit0=1, 立体声数据。写该寄存器直接改变采样率。

7.SCI_WRAM(地址: 0X6 可读写)

读寄存器用来加载用户应用程序和数据到 VS1003 的指令的数据 RAM 中。起始地址在 SCI_WRAMADDR 中进行设置, 且必须先于读写 SCI_WRAM。对于 16 位的数据可以在进行一次 SCI_WRAM 的读写中完成; 而对 32 位的指令字来说则需要两次连续读写。字节顺序是大端模式, 即高字节在前, 低字节在后。在每一次完成全字读写后, 内部指针将自动增加。

8. SCI_WRAMADDR(地址: 0X7 可读写)

用于设置 RAM 读写的首地址。

9.SPI_HDAT0gng SPI_HDAT1(地址: 0X8 只读)

这两个寄存器用来存放所解码的音频文件的相关信息, 为只读寄存器。

当为 WAV 文件时, SPI_HDAT0=0X7761, SPI_HDAT1=0X7665;

当为 WMA 文件时, SPI_HDAT0 的值为解码速率 (字节/秒), 要转换为位率的话则将 SPI_HDAT0 的值乘 8 即可, SPI_HDAT1=0X574D;

当为 MIDI 文件时, SPI_HDAT0 的值可以参考 VLSI 的技术文档第 33 页, SPI_HDAT1=0X4D54;

当为 MP3 文件时, SPI_HDAT0 和 SPI_HDAT1 包含较为复杂的信息(来自于解压之后的 MP3 文件头), 包括当前正在解码的 MP3 文件的采样率、位率等, 具体请参考数据手册的第 33 页到第 34 页。复位后 SPI_HDAT0 和 SPI_HDAT1 将清零。

10.SCI_AIADDR(地址: 0XA 可读写)

用户应用程序的起始地址, 初始化先于 SCI_WRAMADDR 和 SCI_WRAM。如果没有使用任何用户应用程序, 则该寄存器不应进行初始化, 或是将其初始化为零。

11.SCI_VOL(地址: 0XB 可读写)

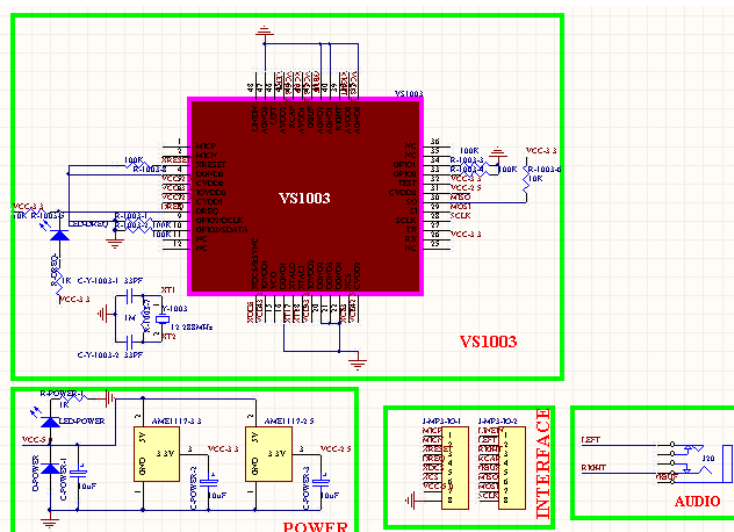
音量控制寄存器。高八位用于设置左声道, 低八位用于设置右声道。设置值为最大音量时的衰减倍数, 步进值为 0.5dB, 范围为 0 到 255。最大音量时的设置值为 0x0000, 而静音为 0xffff。例如, 左声道: -2.0dB, 右声道: -3.5dB, 则 $SCI_VOL=(4 \times 256)+7=0x0407$ 。硬件复位将使 SCI_VOL 清零(最大音量), 而软件复位将不改变音量设置值。

(设置静音(SCI_VOL=0xFFFF)将关闭模拟部分的供电。)

12.SCI_AICTRL[X](地址: 0XC~0XF 可读写)

用于访问用户应用程序。

8) VS1003 有应用电路



2、VS1003 的驱动方法

这里就来介绍单片机对 VS1003 的控制方法，最终实现 MP3 播放。

1) 准备工作

在对 VS1003 进行驱动之前，我们需要确保以下几点已经没问题，否则后面的工作都将是没有意义的。

1.VS1003 各部分的供电电压与输出电压值是不同的。

供电部分	最小电压	推荐电压	最大电压
AVDD（模拟部分）	2.5V	2.8V	3.6V
CVDD（数字部分，内核）	2.4V	2.5V	2.7V
IOVDD（I/O 电压）	CVDD-0.6V	2.8V	3.6V

2.VS1003 与单片机正确可靠连接。

VS1003 与单片机连接的引脚主要有 7 个，分别为 SO、SI、SCLK、/XCS、/XRESET、DREQ、/XDCS。只有保证它们与单片机正确可靠的连接，才能对 VS1003 进行有效的操作与控制。

2) 写命令操作

要控制 VS1003 首先要实现的就是写命令，这是控制是否成功的前提。关于通信接口部分，是一种同步串行接口方式（SPI 从机模式），它要求 SCLK 信号必须由外部电路产生，数据（SDATA）在 SCLK 的上升沿或下降沿时被写入。在笔者的实验中，采用的是软件模拟 SPI，读者也可以选用带有硬件 SPI 的单片机（如 STC12 系列、AVR 系列等），驱动效果会更好。写命令的过程如下：

- 1.等待 DREQ 为高（当 DREQ 为低时，说明芯片还没有就绪）
- 2.将 XCS（命令片选）拉低
- 3.写入 0x02
- 4.写入寄存器地址
- 5.分别写入数据的高字节与低字节
- 6.将 XCS 置高

实现代码如下：

```
void wr_commad(unsigned char addr,unsigned char hdat,unsigned char ldat )
{
    DREQ=1;
```



```

while(!DREQ);
XCS=0;
spi_write(0x02);
spi_write(addr);
spi_write(hdat);
spi_write(ldat);
XCS=1;
}

```

3) VS1003 的初始化

如其它芯片一样，初始化对于 VS1003 来说同样是极其重要的。初始化的过程大致是这样的：

- 1.硬件复位：接 XRESET 拉低
- 2.延时，将 XDCS、XCS、XRESET 置高
- 3.向 MODE 中写入 0X0804
- 4.等待 DREQ 为高
- 5.设置 VS1003 的时钟：SCI_CLOCKF=0x9800，3 倍频
- 6.设置 VS1003 的采样率：SPI_AUDATA=0xbb81，采样率 48k，立体声
- 7.设置重音：SPI_BASS=0x0055
- 8.设置音量：SCI_VOL=0x2020
- 9.这一步被很多人忽视，向 VS1003 发送 4 个字节的无效数据，用以启动 SPI 发送

实现代码如下：

```

void Mp3Reset(void)
{
XRESET=0;
delay(100);
XDCS=XCS=XRESET=1;
wr_commad(0x00,0x08,0x04);
delay(10);
DREQ=1;
}

```

```

while(!DREQ);
wr_commad(0x03,0x98,0x00);
delay(10);
wr_commad(0x05,0xbb,0x81);
delay(10);
wr_commad(0x02,0x00,0x55);
delay(10);
wr_commad(0x0b,VOL_VALUE,VOL_VALUE); // 音量
delay(10);
spi_write(0);
spi_write(0);
spi_write(0);
spi_write(0);
}

```

在进行了正确的初始化后，还要着重检查一下VS1003 的模拟部分是否正常
将 VS1003 的所有 DVDD、AVDD 管脚以及 XRESET、TEST（第 32 个引脚）接 +3.0V，然后测量 RCAP 引脚，它应该是 1.3V 左右，否则芯片模拟部分未正常工作。

4) 正弦测试

在上面的各种操作与检测没有问题后，就可以让 VS1003 放出声音了。可以利用 VS1003 自带的正弦测试对音频输出进行测试。要启动 VS1003 的正弦测试，需要向其写入正弦测试命令。这里提供启动正弦测试的流程，在真实的硬件运行通过，最终的效果是在耳机中听到单一频率的正弦音（频率可以通过程序来更改）。

具体流程如下：

- 1.进入 VS1003 的测试模式：SPI_MODE=0X0820
- 2.等待 DREQ 为高
- 3.将 XDACS 接低，而 XCS 要置高，选择 VS1003 的数据接口
- 4.向 VS1003 发送正弦测试命令：0X53 0XEF 0X6E 0X30 0X00 0X00 0X00 0X00

其中 0X30 为频率，用户可以修改为其它值

5.延时一段时间

6.退出正弦测试，发送命令： 0X45 0X78 0X69 0X74 0X00 0X00 0X00 0X00

7.延时一段时间

8.循环以上流程

实现代码如下：

```
void Sintest(unsigned char x)
{
    wr_commad(0x00,0x08,0x20);
    DREQ=1;
    while(!DREQ);
    XDCS=0;XCS=1;
    spi_write(0x53);
    spi_write(0xef);
    spi_write(0x6e);
    spi_write(x);
    spi_write(0);
    spi_write(0);
    spi_write(0);
    spi_write(0);
    spi_write(0);
    delay(5000);
    spi_write(0x45);
    spi_write(0x78);
    spi_write(0x69);
    spi_write(0x74);
    spi_write(0);
    spi_write(0);
    spi_write(0);
    spi_write(0);
    delay(5000);
    XDCS=1;
```

```
}
```

如果能够通过这一步，就说明你的 VS1003 已经做好了为你播放 MP3 的准备了。下面的工作 就是将 MP3 文件的数据有条不紊地发给 VS003,让它来为你完成 MP3 的解码和播放任务 。

5) MP3 文件数据写入

以上的对 VS1003 的初始化与测试都通过后，现在就可以给它发送 MP3 文件了。但是这时就又出现一个新的问题。MP3 文件通常是比较大的，小的也要 1M~2M，如果使单片机内部的 Flash Rom 的话，容量是远远不够的。需要有一种大容量的存储器来作为 MP3 文件的载体。在笔者的调试系统中采用了 SD 卡（256M）、U 盘（1G）与移动硬盘（40G）来存储 MP3 文件。关于 SD 卡与 U 盘的读写方法可以参看相关章节。这些大容量的存储设备通常也是按照扇区来进行读写的，但在实际的应用中更多的是结合 FAT32 等文件系统来进行文件读写。文件系统部分可以参照《FAT32 的存储机制及其在单片机中的实现》。

这里抛开存储介质不谈，只谈数据的写入方法。其实写入数据的方法十分简单。主要就是看 DREQ 信号，在 VS1003 的 FIFO 能够接受数据的时候输出高电平。每次可以写入 32 个字节的数据。而 DREQ 变低时，单片机就要停止数据的发送。

具体的写数据的方法如下：

- 1.将 XD_CS 拉低
- 2.等待 DREQ 为高
- 3.通过 SPI 写入数据
- 4.在文件没有结束前不断重复 2 与 3 操作
- 5.在所有的数据都发送完毕后，最后发送 2048 个无效字节，用以清除 VS1003 的数据缓冲区
- 6.将 XD_CS 置高

以下是笔者的程序中的写数据部分：

```
XD_CS=0;  
for(j=621;j<2783;j++)  
{  
for(k=0;k<8;k++)
```

```

{
    MMC_get_data_LBA(j,64,get);
    for(i=0;i<64;i++)
    {
        DREQ=1;
        while(!DREQ);
        spi_write(get[i]);
        //delay(60000);
    }
}
}
}
for(temp=0;temp<2048;temp++)
{
    DREQ=1;
    while(!DREQ);
    spi_write(0);
}
XDCS=1;

```

配合 CH375 读 U 盘模块，以 U 盘作为存储介质，加以相应的驱动程序，便可以
实现 U 盘 MP3。