Music Shield 用户手册

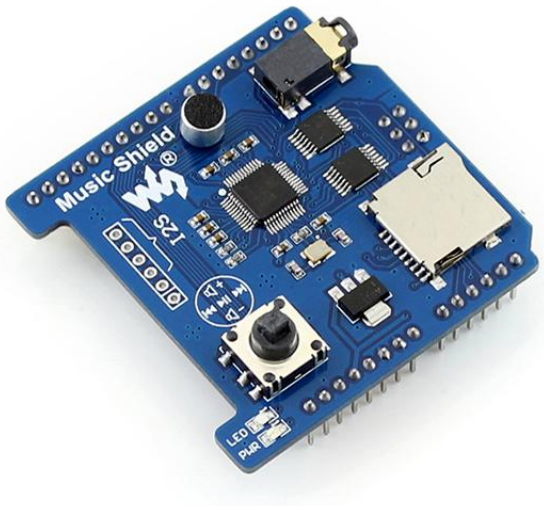
1. 产品概述

Music Shield 是基于 VS1053b 解码芯片的 Arduino 音频播放模块，带 TF 卡槽，支持常用音频文件格式。

产品特性如下：

- 基于 Arduino 标准接口设计，兼容 UNO、Leonardo、NUCLEO、XNUCLEO 开发板
- 支持的格式：MP3/AAC/WMA/WAV/MIDI 等
- 板载 TF 卡，可播放 TF 卡中的相关文件
- 板载 MIC 录音功能，带标准 3.5mm 四段耳机插座
- 板载 74VHC125 电平转换芯片，可兼容 3.3V 和 5V 单片机接口
- 易使用的多功能导航键，轻松控制音乐播放与音量调节
- 引出 I2S 和 MIDI 接口，方便功能扩展

注意：因传输速度和内存限制，UNO、Leonardo 等暂不支持 WAV 音频文件播放以及录音功能，建议使用 NUCLEO 或 Due 等。

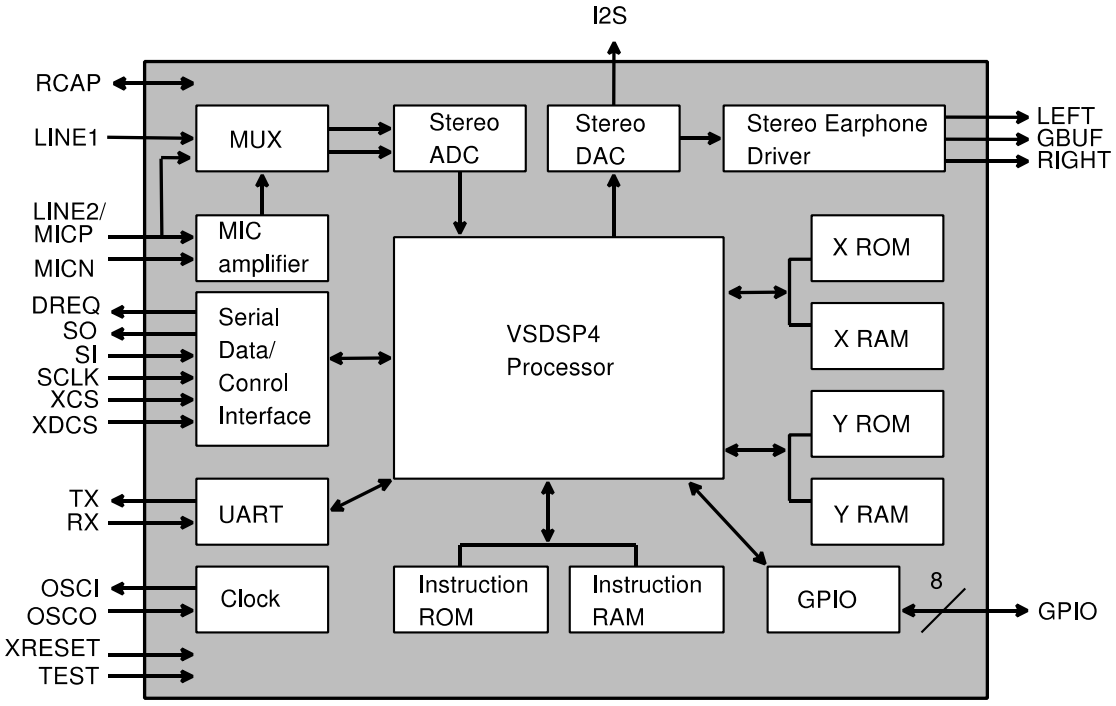


版本：1.1，日期：2015 年 12 月 7 日

2. 使用说明

2.1. VS1053B

Music Shield 采用芬兰 VLSI 公司的 VS1053 作为音频编解芯片，它内部结构如图所示：



控制端口：VS1053 通过 SPI 接口与单片机通信，通过 7 根信号线共同控制，分别如下：

XRSET	异步复位控制线，低电平有效。
XCS	片选输入，低电平有效
XDCS	数据片选，低电平有效。
SI	串行输入
SO	串行输出
SCK	串行时钟线
DREQ	数据请求，如果它为高电平，则可以接收最少 32 字节的 SDI 数据或者接收一条 SCI 命令。当流缓冲区太满和 SCI 命令正在执行的期间，DREQ 会转换到低电平，此时应该停止向 VS105 发送数据和新命令

音频输出：VS1053 可通过 I2S 接口或耳机接口输出音频，LEFT,RIGHT 为耳机左右声道，GBUF 为耳机公共缓冲器。Music shield 保留 I2S 输出接口（LOROUT，MCLK，SCLK，SDTA）。

录音输入：mic 差分输入 MICP/MICN；线路输入，LINE1/LINE2。Music shield 板载麦接到 mic 差分 MICP/MICN 输入，耳机麦接到 LINE2 输入。

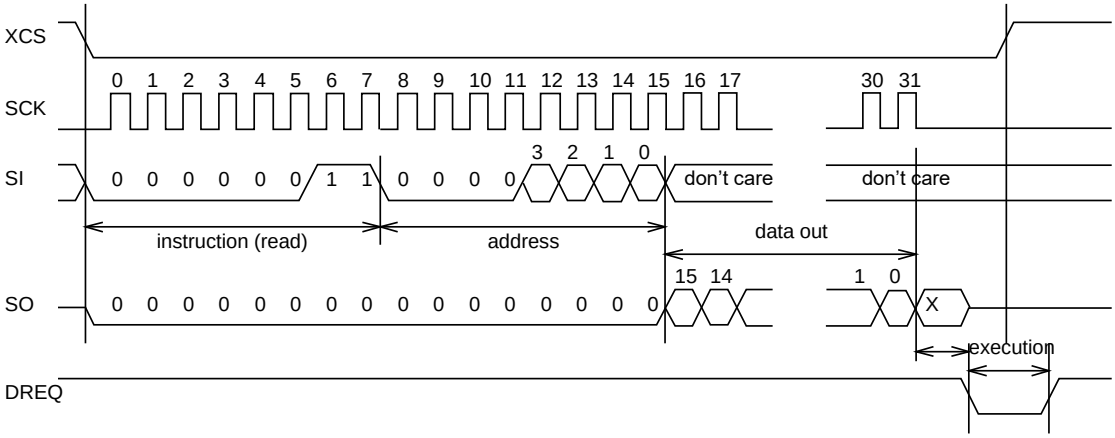
2.2. 串行命令接口的串行协议（SCI）

此串行命令接口 SCI 的串行总线协议包括：一个指令字节、一个地址字节和一个 16 位的数据字。每次读取操作或写入操作均可以访问一个寄存器。由于那些数据位是在 SCK 的上升沿读取的，所以用户只能在 SCK 的下降沿上更新数据。每个字节的 MSb 总是被首先发送。在整个传送操作的期间，XCS 必须要保持为低电平。

这些操作是通过 8 位指令码来指定的。所支持的指令是读取操作和写入操作，如下表所示：

指令		
名称	操作码	操作
READ	0b0000 0011	读取数据
WRITE	0b0000 0010	写入数据

SCI 读操作：



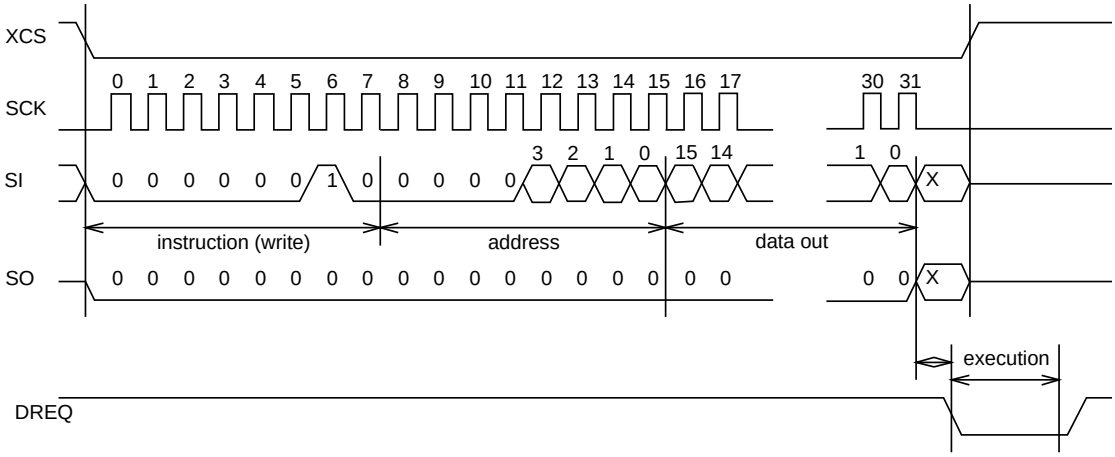
（参见 SV1053b 数据手册图 6）

VS1053b 使用上图时序对寄存器进行读取操作。首先，XCS 信号线被拉到低电平来片选此设备。随后，读取操作码(0x3)加上 8 位宽度的地址后，组成的 16 位字通过 SI 信号线发送到设备。在地址被读取之后，SI 信号线上发送的任何数据都将被芯片忽略。而被确认的地址中的十六位宽度数据将在 SO 信号线上移动输出。

XCS 信号应该在数据移动送出之后驱动到高电平。

DREQ 在读取操作期间会被芯片短暂的拉到低电平，这是非常短的时间，并不需要用户特别的留意。

SCI 写操作：



（参见 SV1053b 数据手册图 7）

写入 VS1053b 寄存器的操作要使用上图顺序。XCS 信号线先下拉到低电平表示选中该设备。将写操作码 (0x2) 加上 8 位的字地址通过 SI 信号线发送到 VS1053b。在这个数据字移位发送的最后一个时钟结束之后，XCS 应该上拉到高电平来结束这个写入顺序。

2.3. SCI 寄存器

VS1053 共有 16 个 SCI 寄存器，通过读写如下寄存器现实 VS1053 各种控制。如下表所示：

SCI 寄存器组，前缀 SCI_					
寄存器	类型	复位值	时间长度	简称	说明
0x0	rw	0x4800	80 CLKI	MODE	模式控制
0x1	rw	0x000C	80 CLKI	STATUS	VS1053b 的状态
0x2	rw	0	80 CLKI	BASS	内建的低音/高音控制
0x3	rw	0	1200 XTALI	CLOCKF	时钟频率加乘数
0x4	rw	0	100 CLKI	DECODE_TIME	解码时间长度（秒）
0x5	rw	0	450 CLKI	AUDATA	各种音频数据
0x6	rw	0	100 CLKI	WRAM	RAM 写/读
0x7	rw	0	80 CLKI	WRAMADDR	RAM 写/读的基址
0x8	r	0	80 CLKI	HDAT0	流的数据标头 0
0x9	r	0	210 CLKI	HDAT1	流的数据标头 1
0xA	rw	0	80 CLKI	AIADDR	应用程序的起始地址
0xB	rw	0	80 CLKI	VOL	音量控制

0xC	rw	0	80 CLKI	AICTRL0	应用程序控制寄存器 0
0xD	rw	0	80 CLKI	AICTRL1	应用程序控制寄存器 1
0xE	rw	0	80 CLKI	AICTRL2	应用程序控制寄存器 2
0xF	rw	0	80 CLKI	AICTRL3	应用程序控制寄存器 3

以下介绍几个主要的寄存器。（更多详情请参见 SV1053b 数据手册 8.7 节）

SCI_MODE 寄存器：该寄存器用于控制 VS1053 模式。

位元	名称	功能	值	说明
0	SM_DIFF	差分	0 1	正常的同相音频 左通道反相
1	SM_LAYER12	允许 MPEG layers I & II	0 1	不允许 允许
2	SM_RESET	软件复位	0 1	不用复位 复位
3	SM_CANCEL	取消当前的文件解码	0 1	不取消 取消
4	SM_EARSPEAKER	EarSpeaker 低设定	0 1	关闭 激活
5	SM_TESTS	允许 SDI 测试	0 1	不允许 允许
6	SM_STREAM	流模式	0 1	不是 是
7	SM_EARSPEAKER	EarSpeaker 高设定	0 1	关闭 激活
8	SM_DACT	DCLK 的有效边沿	0 1	上升沿 下降沿
9	SM_SDIORD	SDI 位顺序	0 1	MSb 在前 MSb 在后
10	SM_SDISHARE	共享 SPI 片选	0 1	不共享 共享
11	SM_SDINEW	VS1002 本地 SPI 模式	0 1	非本地模式 本地模式
12	SM_ADPCM	ADPCM 录音激活	0 1	不激活 激活
13	-	-	0 1	正确的 错误的
14	SM_LINE1	mic / 线路 1 选择	0 1	MICP LINE1

15	SM_CLK	输入时钟范围	0 1	12..13 MHz 24..26 MHz

这个寄存器，我们关注几个重要的位：

第 2 位 SM_RESET 为软件复位，

第 12 位 SM_ADPCM 为激活录音功能，

第 14 位 SM_LINE1 为选择 mic 或线路作为录音输入。

SCI_BASS 寄存器：该寄存器可设置 VS1053 高低音效。

名称	位域	说明
ST_AMPLITUDE	15:12	高音控制，步长为 1.5dB（-8..7, 0 关闭）
ST_FREQLIMIT	11:8	下限频率，步长为 1000Hz（1..15）
SB_AMPLITUDE	7:4	低音增强，步长为 1dB（0..15, 0 关闭）
SB_FREQLIMIT	3:0	频率上限，步长为 10Hz（2..15）

通过这个寄存器如上位的设置，可随意配置自己喜欢的高低音效。

SCI_CLOCKF 寄存器：这个寄存器用来设置时钟频率、倍频等相关信息，该寄存器各位描述如表所示：

SCI_CLOCKF bits		
名称	位域	说明
SC_MULT	15:13	时钟乘数
SC_ADD	12:11	允许的附加乘数
SC_FREQ	10:0	时钟频率

SCI_VOL 寄存器：该寄存器用于硬件音量控制。这个音量寄存器的高字节是控制左通道音量的，低字节是控制右通道音量的。对每个通道，数值在 0..254 的范围内设置可以实现在最大音量级别内的微调（步长 0.5 dB）。左通道数值是通过乘上 256 后再加到这个数值上的。所以，最大的音量是 0、无声是 0xFEFE。

SCI_HDAT1/SCI_HDAT0 寄存器：在激活了 IMA ADPCM 录音之后，寄存器 SCI_HDAT0 和 SCI_HDAT1 有了新的功能。 IMA ADPCM 的采样缓冲区是 1024 个 16 位字。缓冲区的填充状态可以通过 SCI_HDAT1 来读取。如果 SCI_HDAT1 大于 0，则你可以从 SCI_HDAT0 读取许多个 16 位的字。如果数据没能足够快的读取，则缓冲区会溢出并返回空的状态。

2.4. 播放音频文件

用 VS1053 为硬件音频解码器，播放音频文件非常简单。只需三步即可实现。

1) 复位 VS1053

启动时，硬件和软件复位，让 VS1053 状态回到原始状态，准备解码下一首歌曲。

2) 配置 VS1053 相关寄存器

需要配置的寄存器包括模式寄存器（SCI_MODE）,时钟寄存器（SCI_CLOCKF）,音调寄存器（BASS）,音量寄存器（VOL）等。

3) 发送音频数据

当配置完成后，就可以不断向 VS1053 发送音频数据。只要是 VS1053 支持的音频格式，VS1053 就能自动识别解码，进行播放，直到音频数据发送完。

（注：仅当 DREQ 为高电平，VS1053 才能接收数据）

2.5. 录音

录音和播放步骤都是差不多，先复位，然后设置相关寄存器。这里主要配置如下寄存器：

寄存器	位域	说明
SCI_MODE	2, 12, 14	开始 ADPCM 模式，选择：mic/线路 1
SCI_AICTRL0	15..0	采样率 8000..48000Hz（在录音启动时读取的）
SCI_AICTRL1	15..0	录音增益（1024 = 1×）或 0 是自动增益控制（AGC）
SCI_AICTRL2	15..0	自动增益放大器的最大值（1024 = 1×，65535 = 64×）
SCI_AICTRL3	1..0 2 15..3	0=联合立体声(共用 AGC), 1=双声道(各自的 AGC), 2=左通道, 3=右通道 0=IMA ADPCM 模式，1=线性 PCM 模式 保留，设置为 0

IMA ADPCM 录音模式是通过设置 SCI_MODE 中的 SM_RESET 和 SM_ADPCM 位来激活的。如果 SM_LINE1 被设置，则“线路输入”会替代“mic”的差分输入。

（Music Shield 板载麦被接到 mic 差分输入，耳机麦接到线路输入 2）。

在激活 ADPCM 录音之前，用户必须要将正确的值写入 SCI_AICTRL0 和 SCI_AICTRL3 中。这些值仅在录音被启动时才读取。SCI_AICTRL1 和 SCI_AICTRL2 可以随时更改，但在激活之前最好写入好的初始值。

设置好寄存器后就可以不断读取音频数据了。如果 SCI_HDAT1 大于 0，则你可以从 SCI_HDAT0 读取（SCI_HDAT1）同样多个 16 位的字。

注：这些录音数据不包括音频文件的 RIFF 标头，录音完毕要在数据之前加上标头才是一个真正的音频文件。

2.6. 实时 MIDI

如果在启动时 GPIO0 是低和 GPIO1 是高，则实时 MIDI 模式被激活。在此模式下 PLL 是设定为 4.0×，UART 被配置为 MIDI 的数据传输率 31250bps，并且实时 MIDI 数据是从 UART 和 SDI 读出的。这两个输入不应该同时使用。如果您使用的 SDI，首先要发送 0xff 然后发送 MIDI 数据字节。

（Music Shield 引出 UART 的 RX 管脚作为 MIDI 输入端口。）

注：如果 GPIO1 不为高，实时 MIDI 可以使用 SCI 加载一个小的补丁代码来开始。

3. 快速入门

3.1. 引脚说明

1) D11、D12、D13 默认为 SPI 接口的 MISO、MOSI、SCLK 总线。
2) A0、A1、A2、A3 分别接到 VS1053 的 XRESET、DREQ、XDCS、XCS。
3) D10,D9 分别为 SD 卡的片选 CS 和检测管脚 SD_Detect
4) D7,D6,D5,D4,D3 分别为功能按键的下（音量-），左（上一首），中（播放/暂停/录音），右（下一首），上（音量+）。
5) D8 为 LED 指示灯(播放/录音时闪烁)。

3.2. 操作与现象

下面章节以本公司两款不同类型的开发板为例，描述具体操作步骤及实验现象。

3.2.1. XNUCLEO-F103RB（主控芯片 STM32F103R）：

1. 音频播放实验

1) 将音频文件复制到 TF 卡的根目录下，并将 TF 卡插入 Music shield TF 卡座中。
2) 将开发板连接到电脑。
3) 编译下载 Demo 程序。

4) 打开串口调试助手，选择正确串口号并设置如下。

波特率：9600；数据位：8；停止位：1；校验位：None;控制流：None

5) 插入耳机可以听到播放的音乐，控制多功能按键上下调节音量和左右选择曲目，中间按键短按为播放或暂停，长按进入录音功能(程序选择耳机麦录音，如用板载 mic 录音需修改程序)，再短按停止录音。串口接收如下所示：

```
**********playing list**********
1 . test.wav
2 . test.aac
3 . test.flac
4 . test.mp3
5 . test.ogg
6 . test.wma

Playing test.wav ...
Recording......
Press play button to stop.
Recording end
```

2. MIDI(实时 MIDI 不需要 SD 卡)

1) 编译下载 Music Shield MIDI 程序。

2) 将开发板连接到电脑，打开串口调试助手，选择正确串口号并设置如下。

波特率：9600；数据位：8；停止位：1；校验位：None;控制流：None

3) 插入耳机可以听到一段美妙的 MIDI 音乐，串口接收如下所示：

```
Init vs10xx in MIDI format...
done
Fancy Midi Sounds
 Instrument: 30
N: 27
N: 28
N: 29
N: 30
N: 31
N: 32
N: 33
N: 34
N: 35
```

```
N: 36
N: 37
N: 38
N: 39
N: 40
```

3.2.2. Arduino UNO PLUS

1. 播放实验（因 UNO 受速度和内存限制，暂不支持高质量无损音乐播放和录音功能）

1) 将音频文件复制到 TF 卡的根目录下，并将 TF 卡插入 Music shield TF 卡座中。

2) 将示例程序解压至以下 Arduino 安装目录路径： ..\arduino\libraries 中。

3) 运行 Arduino 软件，点击 File --> Examples --> MusicPlayer --> MusicShield 打开程序，并编译下载。

4) 将开发板连接到电脑，打开串口调试助手，选择正确串口号并设置如下。
波特率：115200；数据位：8；停止位：1；校验位：None;控制流：None

5) 插入耳机可以听到播放的音乐，串口接收如附录所示，控制多功能按键上下调节音量和左右选择曲目，中间按键短按为播放或暂停。串口接收如下所示：

```
---- songs in TF card (root dir) ----
RECORD~1.WAV
TEST.ACC
TEST.MP3
TEST.OGG
TEST.WAV
TEST.WMA
Playing RECORD~1.WAV
[done!]
Playing TEST.WAV
```

2. MIDI(实时 MIDI 不需要 SD 卡)

1) 点击 File --> Examples --> MusicPlayer --> MusicShield 打开程序，并编译下载。

2) 将开发板连接到电脑，打开串口调试助手，选择正确串口号并设置如下。
波特率：115200；数据位：8；停止位：1；校验位：None;控制流：None

3) 插入耳机可以听到一段美妙的 MIDI 音乐，串口接收如下所示：

```
Init vs10xx in MIDI format...done
Fancy Midi Sounds
 Instrument: 30
N: 27
```

```
N: 28
N: 29
N: 30
N: 31
N: 32
N: 33
N: 34
N: 35
N: 36
N: 37
N: 38
N: 39
N: 40
```

4. 版本历史

版本	修改	日期
1.0	初始版本	2015 年 7 月 9 日