基于步进电机的MIDI播放器

谢作如 陈益漳 浙江省温州中学

DF的创客社区是学生们常常去的地方，在那里总能看到一些稀奇古怪的项目。这次学生看到的是一个用步进电机播放MIDI音乐的帖子，几个步进电机居然能播放出各种音符，实在很酷。根据帖子的介绍和提供的代码，我们很快发现其使用方式较老，并且需要Linux环境，操作不方便，兼容性也不好。于是，温州中学创客空间也买了相关的设备，研究了一种新的MIDI播放器方案。

原理分析

首先需要了解一些原理。步进电机是将电脉冲信号转变为角位移或线位移的开环控制电动机。它通过控制脉冲个数可以控制步进电机的角位移量，从而达到准确定位的目的；同时通过控制脉冲频率可以控制电机转动的速度和加速度。因为速度和角度可控，步进电机常用在3D打印机、激光雕刻机上。

步进电机运转时会发出噪音，而这噪音具有一定的规律，其音高与当前电机转速有直接关联。如果给步进电机合适频率的脉冲，步进电机就能够按照一定的音高发出我们需要的声音。经过反复实验和参考资料，直接以标准A（440Hz）的频率发送脉冲，并按照半音频率之间相差 $\sqrt[12]{2}$ 的规律（也就是升一个八度频率翻一倍），就可以发出对应的音。这一方式也适用于蜂鸣器等可以发出声音的元件。

接下来需要了解点MIDI知识。MIDI即乐器数字接口，是编曲界使用最广泛的数字音乐标准格式。MIDI的记谱方式和五线谱、简谱有所不同，标准A在MIDI中是A4，每升八度（12个半音）就会加一（如A1升八度变成A2），具体如表1所示。

一个MIDI信号可以有多个字节的内容，但第一字节的内容格式是固定的。第一字节称为状态字节，表示对MIDI设备当前状态的设定。状态字节的指令高位（前四位）一定会大于或等于8（0b1000），而低位（后四位）表示控制的轨道（0x00~0x0F）。前四位代表的信息类型如表2所示。

用Arduino处理MIDI

使用Arduino的最大优势是能够找到很多开源的库或者代码。一个叫做MIDIUSB的库可以直接将Arduino Leonardo板变成一个USB接口的MIDI设备。MIDIUSB库中封装了一个名称为MidiUSB的类，它可以直接用MidiUSB.read()读取MIDI信号，并返回一个midiEventPacket_t类的变量。库中midiEventPacket_t的定义如图1所示。

可以看到这个类（实际上只是结构体）中只存了3个字节的内容（header实际上是byte1的前4位，即

表2

高位	信息类型
8	停止发声
9	开始发声
A	轮指
B	改变控制器
C	改变音色
D	通道演奏压力（类似音量）
E	音高（调号）
F	不属于任何通道，总控制

```
typedef struct
{
        uint8_t header;
        uint8_t byte1;
        uint8_t byte2;
        uint8_t byte3;
}midiEventPacket_t;
```

图1

表1

音	C4	Db4	D4	Eb4	E4	F4	Gb4	G4	Ab4	A4	Bb4	B4
频率	261.63	277.81	293.66	311.13	329.63	349.23	369.99	392	415.3	440	466.16	493.88

```
void loop () {
 static byte note;
 static byte lastCommand = MIDI_IGNORE;
 static byte state;
 static byte lastByte;
 while (Serial.available()) {
   byte incomingByte = Serial.read();
   if (incomingByte & 0b10000000) {
     if (respondAllChannels ||
             (incomingByte & 0x0F) == myChannel) {
       lastCommand = incomingByte & 0xF0;
     } else {
       lastCommand = MIDI_IGNORE;
     }
     state = MIDI_STATE_BYTE1;
   } else if (state == MIDI_STATE_BYTE1) {
     if ( lastCommand==MIDI_CMD_NOTE_OFF )
     {if (note == incomingByte) noTone(tonePin);
       state = MIDI_STATE_BYTE2;
     } else if ( lastCommand == MIDI_CMD_NOTE_ON ){
       lastByte=incomingByte;
       state = MIDI_STATE_BYTE2; }
   } else {
     if (lastCommand == MIDI_CMD_NOTE_ON) {
       if (incomingByte != 0) {
         note = lastByte;
         tone(tonePin,(unsigned int)pgm_read_word(&frequency[note]));
       } else if (note == lastByte) {
         noTone(tonePin);
       }}
     state = MIDI_STATE_BYTE1; }
 }}
```

图2

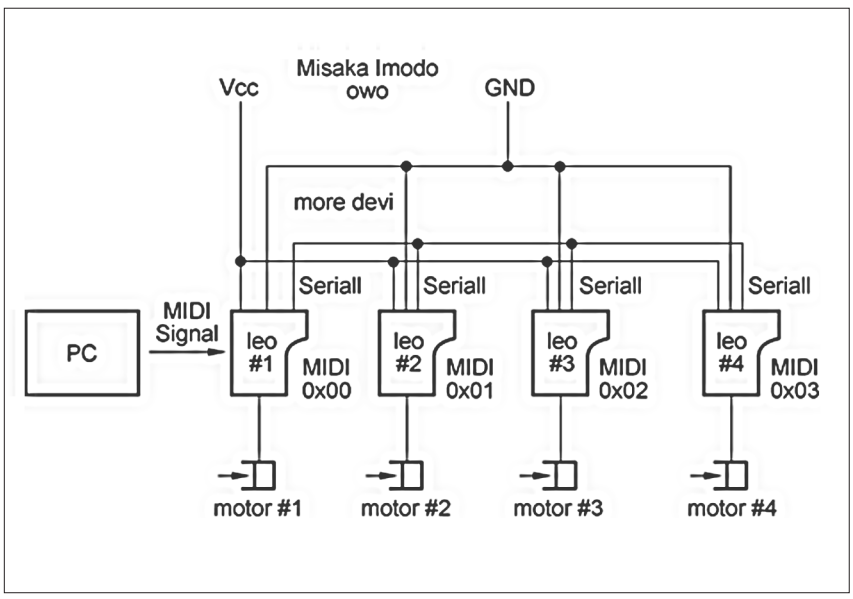

图3

第一字节的高位，表示当前指令）。之后需要的就是对读取的MIDI信号进行处理。处理MIDI的部分同样可以在一些论坛（建议找国外论坛）上找到，改改就能用了（如图2）。

从这段代码可以看出，在处理MIDI信号时要针对状态字节的具体指令进行分类操作。实际上这段代码只涉及了两个会用到的操作：发声和停止发声。代码中还使用了tone函数。tone在英语中原意是音调，在Arduino中的作用就是对一个端口发送一定频率的脉冲。其原型为tone(pin, frequency[, duration])，即（在duration的时间内）向pin端口发送频率为frequency Hz的脉冲。与之对应的还有停止发送脉冲的noTone()。tone函数的缺点在于不能同时对一个以上的端口发送信号。而采用分时系统（姑且称之为“伪多线程”）就无法及时停止脉冲，会导致不同步，这可能与Leonardo的性能和分时系统本身的缺陷有关。

硬件搭建

播放设备使用了Arduino Leonardo和DFrobot出品的双路步进电机驱动板和步进电机。播放设备的结构如图3所示。

先将电机、驱动板、Leonardo组装在一起。因为步进电机驱动板需要超过8V的电压才能工作，所以先拿洞洞板焊了一个并联4个电机的部件。同时4块Leonardo使用串口通信来传输信号，就把通信的线一

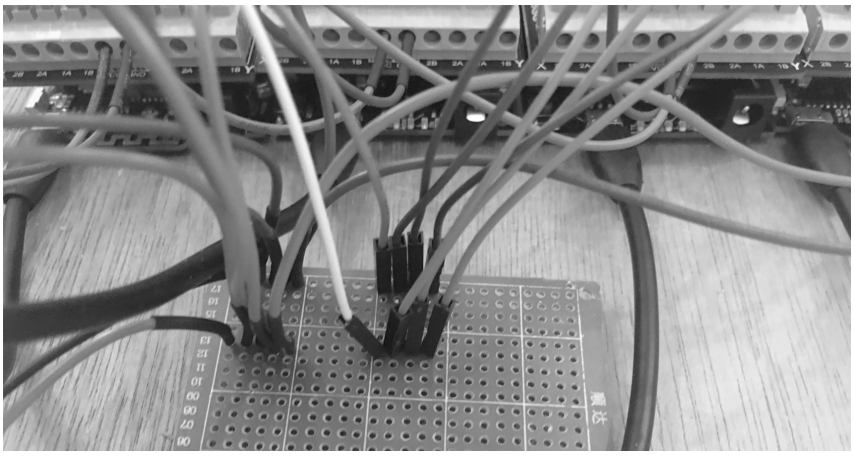

图4

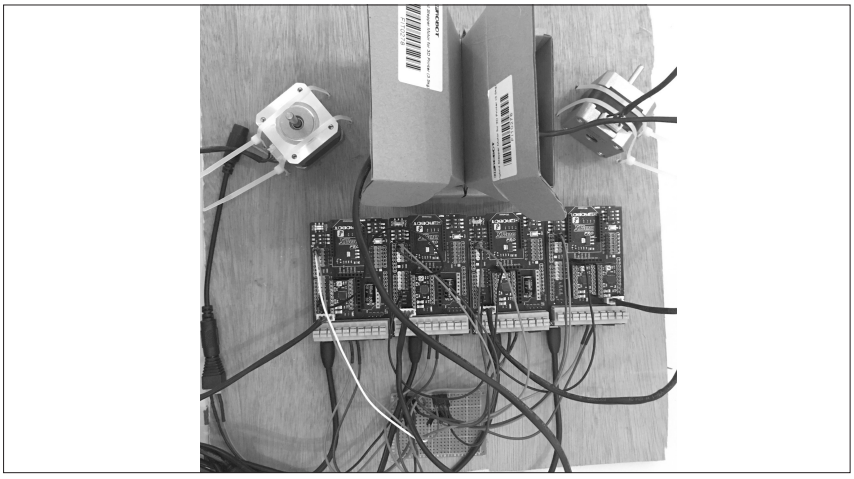

图5

起焊在洞洞板上了（如图4）。

将MIDI处理的代码烧录到各个Leonardo板子上。但是一号板子烧写的代码里要使用MIDIUSB，其他的板就直接用串口就行了。

试着用WIDI播放改编过的曲子“千本樱”，大家居然都说挺好听。电机振动居然能够发出有规律的乐音，的确挺酷的，吸引了很多同学围观。视频地址为http://www.bilibili.com/video/av4596330/。

在使用过程中我们还发现，要让电机发出足够响的声音，最好做个共鸣腔——用小纸箱就行了。还要将电机用扎带固定在木板上，因为电机发声的音量和与之共振的物体大小有关（如图5）。

完成基于步进电机的MIDI播放设备后，再对比原来论坛上的帖子，其在兼容性方面的确要好一些。原帖子介绍要在Linux下运行，而这个方案可以运行在任何操作系统下，只要能运行支持MIDI设备的音乐软件，如WIDI。但这一方案也存在一些不足，如不能够播放和弦、不够直观等。

如果对相关内容感兴趣，请关注主持人博客。

参考文献：

[1]ChoirBot，桌子上的迷你机械乐队[J/OL].http://www.dfrobot.com.cn/community/thread-14112-1-1.html.

[2]步进电机_百度百科[EB/OL].http://baike.baidu.com/link?url=e6O45jcE2J1mb0n-yOxCr3sZv_HXz1uibWk7g2q-22xv_ccM8f7CughGI4AsDyYoOkvoK8kerpmrXyMGxCQNCq.

[3]Tone() + MIDI = ToneMIDISynth[J/OL].http://forum.arduino.cc/index.php?topic=79326.0.