MIDI 键盘、简易电子琴和音乐播放器项目报告

（陈奕玮 未央书院 chen-yw20@mails.tsinghua.edu.cn）

一）项目名称：

MIDI 键盘、简易电子琴和音乐播放器

二）项目设计：

1. 功能：

A 由按键，通过 MIDI 协议，向电脑中实时输入指定音符（时长由按下按键的时间控制）。

B 实现了以 USB 转串口模块与 MIDI 线材两种实体连接方式，将 MIDI 信号输入电脑。

C 弹奏三个八度的 C 大调音阶，用蜂鸣器播放。

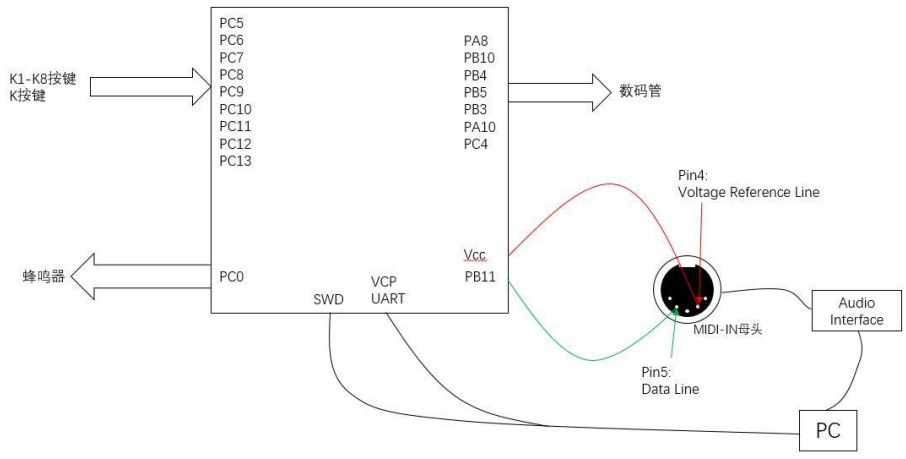
D 用蜂鸣器播放三首歌曲，中止播放曲目。

E 在用蜂鸣器播放曲目时，数码管显示对应曲目编号；蜂鸣器电子琴模式下，弹奏音时，数码管显示对应音。

附播放的音乐曲目名称：

1. 一路向北
2. 郭源潮
3. 清华大学校歌(指定曲目)

2. 系统硬件连接示意图：



3. 程序中的类与函数：

A 类：

头文件 typedef.h 中实现了四个枚举类：Tune 表示音调，该类包括空拍，以及 C 大调的 DO-SI 音符;Mode 表示键盘此时的模式,用于区分每个模式下程序需要执行的功能;Rhythm 表示一个音符的长度，实现了全音符~十六分音符（不包括三连音、五连音等）；Song 表示目前正在播放的歌曲，分为 SONG_1 到 SONG_3。

B 函数：

set_tune()函数实现了控制蜂鸣器发出音高的功能。该函数更改了 TIM1 的 PWM 参数，使其输出对应频率的 PWM 波形，从而操纵蜂鸣器发出对应音高的声音。

play_song()函数实现了播放一首歌曲的功能。其本质是通过一个循环，遍历一个数组中的所有音符。对于每一个音符，调用 set_tune()函数设置音高，结合 bpm 与 Rhythm 参数控制音符的长短。

display()函数实现了控制数码管显示的功能。在参数列表中，输入一个’0’~’7’之间的字符（或’^’），即可显示对应的数字。其中’^’表示什么都不显示。

control_midi()表示控制 midi 信号。其功能有两个，包括发出一个“音符开始：note on”信号与发出一个“音符结束：note off”信号。目前支持演奏一个八度的 C 大调音阶，预留了 octave 参数接口，后续可以扩充不同的八度。

中断函数 HAL_TIM_PeriodElapsedCallback()实现了按键消抖功能。通过设置时钟，保证每 2ms 访问一次该函数。每次调用该函数，则记录一次当前的开关状态。当连续八次的开关状态相同时，则可以认为开关已经稳定，此时，就记录开关的状态，作为 debounced_key_status 。这样就实现了按键的消抖功能。思路见：https://www.likecs.com/show-189434.html

主循环以查询方式实现了两个模式的核心功能。在 MIDI 模式下，根据按钮的变化情况，控制 MIDI 信号。具体而言：当按钮目前的状况与该音符目前是 on 还是 off 不一致时，则调用 control_midi()函数，改变音符的 on 与 off 状态，实现音符的开与关。将 MIDI 功能做成查询方式而不采用中断方式，主要是担心在中断函数的触发信号实际上是抖动。如果触发中断的信号是抖动，而恰巧抖动的方向与按键的方向相反，就可能导致按键按下时不演奏音符，或按键松开时音符仍在演奏的情况发生。这一点在舞台上是绝对不可接受的，因此必须采用查询方式。在 MUSIC 模式下，根据全局变量 song_to_play 的取值，调用 play_song()实现了播放对应歌曲的功能。

按键中断函数 HAL_GPIO_EXTI_Callback()一方面实现了模式切换。另一方面，在 PIANO 模式下，实现了用蜂鸣器播放一个音符的功能；在 MIUSIC 模式下，实现了改变全局变量 song_to_play 的选择的功能；在 MIDI 模式下不实现任何功能（因为对应功能放在查询中实现）。

三）项目实现：

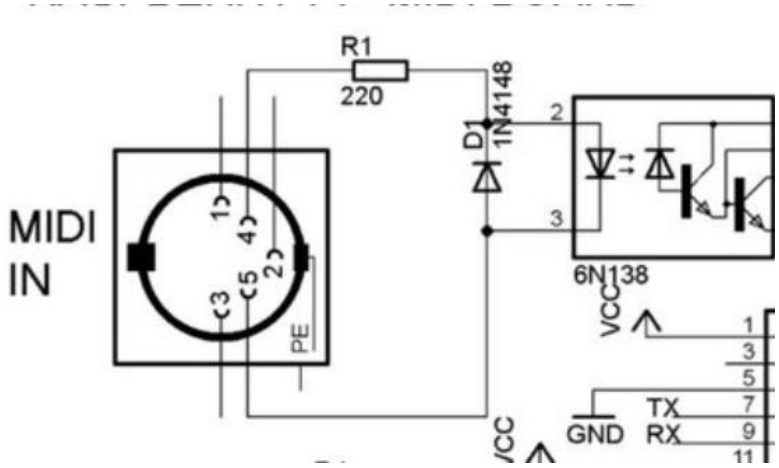
1. 描述在调试过程中遇到的 3 个难点问题及解决方法；

A 对电路与 MIDI 的硬件部分没有了解。

由于电子产品的开发并不是一见极为安全的事情，在没有学习过电路原理的情况下，其实不敢贸然对未知的设备进行接线的操作。

在这个问题上，一方面询问了老师，获得了学长学姐当年在开发时所积攒的一些资料。

另一方面，上网自行搜索 MIDI Cable Wiring（MIDI 线的接线方法），搜索完后仍然担心，于是就继续查询声卡上的 MIDI-IN 接口内部是什么样的电路。就这个电路与电机系学生交流，进行初步的估算，发现即使是在 5V 接反的情况下，由于分压电阻的存在，光电耦合器并不会有损坏，于是便打消了对于损坏器件的恐惧。MIDI-IN 接口内部的电路图如下。



B 硬件购买不及时。

由于硬件购买不及时，担心硬件可能无法在答辩前到达手里。与老师交流后，了解到，可以运用软件将 USB 转串口模块输入的信号转变为软件可识别的 MIDI 信号，实现向软件输入 MIDI 信号的功能。具体而言，信号通过单片机串口转 USB 模块输入电脑，由电脑上的 Hairless MIDI Serial Bridge 软件将该信号由串行通讯信号转换为符合 MIDI 协议波特率的信号，并将这个 MIDI 信号连接到 loop MIDI 软件产生的 MIDI 端口上。这样一来，相关音乐软件就可以通过 loop MIDI 端口，获取通过 USB 转串口模块输入的 MIDI 信号。

幸运的是，硬件在展示前到达了手中，于是就有了第二种输出 MIDI 信号的方法，即通过 MIDI 插座与 MIDI 线材将 MIDI 信号输入电脑音乐软件的方法。将单片机的 LPUART 引脚与高电平引脚接入 MIDI 插座的 5Pin 与 4Pin，将 MIDI 线材一端插入单片机侧插座，另一端插入电脑或声卡，电脑即可识别出该 MIDI 设备。第二种方式是目前市面上专业产品的实现方式。其具体接线见“硬件连接示意图”。

C 输入曲谱的过程过于麻烦。

由于大家普遍采用数组的方式储存音符，如果纯手工输入数组，则会产生诸多不便，例如，可能就需要输入很多个逗号以分割不同音符。因此，额外编写了一个 python 程序，将简谱转化为数组的形式，使得人在输入曲谱时只用输入简谱。例如，输入[123]就会输出[DO, RE, MI,]。**实现了关于唱鸣，八度，节奏这三个数组的转换程序。**其中的一个程序如下。这一 python 程序会随报告附上，并附上输入与输出例子。

```python
def convert_tune(directory, index):
    # output directory
    output = open(directory + r"\tune_" + str(index) + ".out", 'w')
    # input directory
    f = open(directory + r"\tune_" + str(index) + ".in", 'r').read()
    # example: 123 -> DO, RE, MI,
    for char in f:
        if char == '0':
            print("REST", end=', ', file=output)
        elif char == '1':
            print("DO", end=', ', file=output)
        elif char == '2':
            print("RE", end=', ', file=output)
        elif char == '3':
            print("MI", end=', ', file=output)
        elif char == '4':
            print("FA", end=', ', file=output)
        elif char == '5':
            print("SOL", end=', ', file=output)
        elif char == '6':
            print("LA", end=', ', file=output)
        elif char == '7':
            print("SI", end=', ', file=output)
        else:
            print(char, end='', file=output)
```

2. 不足和可改进之处

该 MIDI 设备的开关十分简单，仅仅存在两个状态，开与关。实际上，MIDI 信号的一个重要信息是演奏的力度。下一步，可以考虑更换可以识别“力度”这一属性的按钮，以控制 MIDI 音符的力度大小。

其实，MIDI 设备不仅能输出音符，其实还能输出各类控制信号，用于控制支持该协议的设备。例如，在演出的过程中，乐手可以通过 MIDI 信号切换音色。下一步可以扩充代码，支持这些控制信号，使得其可以完成更多功能。但其实这些功能都是基于 UART 的，格式也类似音符，因此在此框架内补充，即可比较容易地实现。

四）附

附件的源代码的 LPUART1 端口为 PB11，波特率为 31250。采用的是“MIDI 插口与 MIDI 线直接连接电脑”的连接方式。

若要改为通过 USB 转串口模块、Hairless MIDI Serial Bridge 与 loop MIDI 实现输入，首先需要调整 PA2 为 LPUART1 的端口，然后设波特率为 115200（即 Hairless MIDI Serial Bridge 的 Preference 中设置的值）。