

简谱是大众化的音乐记谱方式，比较容易理解和掌握。我们可以把一首乐谱（score）看成是由若干个基本的音符（note）单元组成。一个音符由音名和音长组成。音名就是低音、中音、高音的1234567（唱作do re mi fa sol la si），其本质是音符的发声频率。用L1～L7、M1～M7、H1～H7定义了低音、中音、高音所对应的发声频率。音长是音符的发声时间长短。

2.基于C语言的乐谱表示与实现

2.1乐谱的表示

简谱中音高（或音调）的表示是通过1到7这7个阿拉伯数字作为音名，再加上表示升降的符号“．”来表示的。例如，中音re写成2，低音re写成 ，高音re写成 ，最高音re写成 ,这些都是C语言无法识别的符号。怎样把这些符号变成C语言能识别的数据呢？基本方法是用一个结构体的两个域来表示，一个域用阿拉伯数字“2”表示音名2，一个域表示升降符号，分别用符号“L”、“M”、“H”和“Z”表示低音、中音、高音和最高音。

2

2

2

简谱中音长（或节拍）的表示是通过音名加后划线和下划线来表示的。例如，2、2-、2—和2、2分别表示1拍、2拍、3拍和半拍、四分之一拍，假设1拍的时值为1秒，那么，上面5个音的音长分别为1秒、2秒、3秒和0.5秒、0.25秒。右边的是小附点·比如5·就是唱完一个拍延续它自身音长的一半 。

例如，下面是歌曲《生日快乐》中的一段曲子：



用C语言表示如下：

SOUND music[11]=｛｛5,M,0.5｝，｛5,M,0.25｝，｛5,M,0.25｝，｛6,M,1｝，｛5,M,1｝，｛1,M,1｝｝；

5·表示为{5,M,0.5},｛5,M,0.25｝。5表示音名为5，音调为中音M，二分之一节拍。点表示为延续其自身的一半的节拍，就是二分之一的二分之一，为四分之一节拍｛5,M,0.25｝

要演唱一首乐曲，首先要实现单音的发声。一个单音由两部分控制，一部分是音调（或音高），一部分是节拍（或时值）。音调又由7个音名值和4个音高升降值确定，分别对应28个不同的频率。下表为音调和频率的对应表：

表1 音调和频率的对应表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | L | M | H | Z |
| 1 | 131 | 262 | 523 | 1047 |
| 2 | 147 | 296 | 587 | 1175 |
| 3 | 165 | 330 | 659 | 1319 |
| 4 | 176 | 349 | 699 | 1397 |
| 5 | 196 | 392 | 784 | 1568 |
| 6 | 220 | 440 | 880 | 1760 |
| 7 | 247 | 494 | 988 | 1967 |

C语言用控制函数sound( 频率值)来控制28个音调，如中音2可以通过函sound(296)来控制。在STM32中调整PWM的周期为这28歌音调的频率，占空比可自己调整到合适的，一般为50%。

C语言中可以用延时函数delay(音长值)来控制节拍。假如你要控制音乐的速度为60，即一分钟60拍，也就是1拍为1秒（即1000微妙），如果节拍数为x，那么，节拍控制函数为：

delay(x\*1000);

void Timer4\_Init(u16 Period)//用来产生音调的频率

{

GPIO\_InitTypeDef GPIO\_Init\_Structure;

TIM\_TimeBaseInitTypeDef TIM\_TimeBaseInit\_Structure;

TIM\_OCInitTypeDef TIM\_OCInit\_Structure;

RCC\_APB1PeriphClockCmd(RCC\_APB1Periph\_TIM4, ENABLE);//打开定Timer4时钟

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOB, ENABLE);//打开GPIOB时钟

GPIO\_Init\_Structure.GPIO\_Pin = GPIO\_Pin\_8;

GPIO\_Init\_Structure.GPIO\_Mode = GPIO\_Mode\_AF\_PP;//手册8.1.11复用推挽输出

GPIO\_Init\_Structure.GPIO\_Speed = GPIO\_Speed\_50MHz;

GPIO\_Init(GPIOB, &GPIO\_Init\_Structure);

TIM\_TimeBaseInit\_Structure.TIM\_CounterMode = TIM\_CounterMode\_Up;//向上计数模式

TIM\_TimeBaseInit\_Structure.TIM\_Period = Period;//周期

TIM\_TimeBaseInit\_Structure.TIM\_Prescaler = 7199;//分频

TIM\_TimeBaseInit(TIM4, &TIM\_TimeBaseInit\_Structure);

TIM\_OCInit\_Structure.TIM\_OCMode = TIM\_OCMode\_PWM2;

TIM\_OCInit\_Structure.TIM\_OCPolarity = TIM\_OCPolarity\_High;

TIM\_OCInit\_Structure.TIM\_OutputState = TIM\_OutputState\_Enable;

TIM\_OCInit\_Structure.TIM\_Pulse = Period;//占空比

TIM\_OC3Init(TIM4, &TIM\_OCInit\_Structure);

TIM\_OC3PreloadConfig(TIM4, TIM\_OCPreload\_Enable);

TIM\_Cmd(TIM4, ENABLE);

}#define proport 10000 //Tclk/(psc+1)=72000000/(7199+1)

#define L1 ((proport/131)-1)//低调　do 的周期根据Tout= ((arr+1)\*(psc+1))/Tclk推出arr值就是本句define定义的值，Tout为音调频率131Hz的倒数，Tclk=72MHz

#define L2 ((proport/147)-1)//低调　re 的周期

#define L3 ((proport/165)-1)//低调　mi 的周期

#define L4 ((proport/176)-1)//低调　fa 的周期

#define L5 ((proport/196)-1)//低调　sol的周期

#define L6 ((proport/220)-1)//低调　la 的周期

#define L7 ((proport/247)-1)//低调　si 的周期

#define M1 ((proport/262)-1)//中调　do 的周期

#define M2 ((proport/296)-1)//中调　re 的周期

#define M3 ((proport/330)-1)//中调　mi 的周期

#define M4 ((proport/349)-1)//中调　fa 的周期

#define M5 ((proport/392)-1)//中调　sol的周期

#define M6 ((proport/440)-1)//中调　la 的周期

#define M7 ((proport/494)-1)//中调　si 的周期

#define H1 ((proport/523)-1)//高调　do 的周期

#define H2 ((proport/587)-1)//高调　re 的周期

#define H3 ((proport/659)-1)//高调　mi 的周期

#define H4 ((proport/699)-1)//高调　fa 的周期

#define H5 ((proport/784)-1)//高调　sol的周期

#define H6 ((proport/880)-1)//高调　la 的周期

#define H7 ((proport/988)-1)//高调　si 的周期

int music[]=

{

M5,50,M5,25,M5,25,

M6,100,M5,100,H1,100,

M7,100,M7,100,M5,50,M5,25,M5,25,

M6,100,M5,100,H2,100,

H1,100,H1,100,M5,50,M5,25,M5,25,

H5,100,H3,100,H1,100,

M7,100,M6,100,H4,50,H4,25,H4,25,

H3,100,H1,100,H2,100,H1,100,H1,100

};//生日快乐歌C调简谱C语言表示方法(M5,50)为(音调，音长)100表示为一个音长(节拍)，50位半个音长(节拍)，25位四分之一音长(节拍)

int length = sizeof(music)/sizeof(music[0]);//计算数组长度

int main()

{

int i=0;

delay\_init();//延时初始化

while(100)

{

for(i=0; i<(length/2); i++)//取数组数据

{

Timer4\_Init(music[2\*i]);//PWM波周期

delay\_ms(5\*music[2\*i+1]);//音长的时间都乘以5即一拍为500微秒，此值"5"可调整，只是播放的整个快慢而已，有点类似于视频快进和后退

}

}

}