基于Stone-STVC070WT屏的

音乐节拍器

## 一、摘要

音乐教育发展历史悠久，近年来随着我国经济水平的不断提高，其发展态势异常迅猛，以乐器之王钢琴为例，2019年的时长规模达到了54.22亿元，较2018年同比增长10.71%。节拍器作为作为钢琴学习中不可缺少的辅助工具，起到培养速度概念、演奏更加平稳、提供练琴效率的作用。

本文基于Stone-STVC070WT屏开发了一款的音乐节拍器。该方案使用Stone-STVC070WT HMI屏作为显示控制平台，Arduino板作为节拍音频输出终端，打造出一款简单易用的音乐节拍器。

## 实现的功能

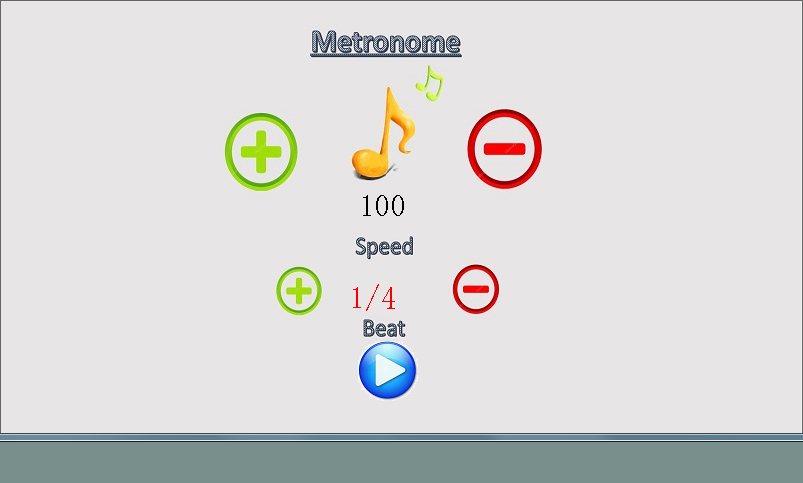


图 1 UI界面示意

1.节拍器UI功能界面，提供触控功能；

2.频率设定功能，通过触控按钮，可以实现从40-120次/分，间隔为10的调节；

2.节拍设定功能，通过触控按钮，实现1/4、2/4、3/4、4/4四种节拍的悬着设定；

3.播放控制功能，启动或暂停节拍执行；

4.Arduion板驱动蜂鸣器实现节拍的音频；

5.串口通信实现屏幕与Arduion的信息交互。

## 三、系统原理及组成

该系统主要由两部分组成，前端显控平台和后端执行终端，具体的实现原理如下。

触控按钮

UI界面

前端显控平台

串口

UART

PWM

后端执行终端

串口

蜂鸣器

蜂鸣器

定时器

图 2系统框图

Stone HMI屏作为显示控制平台，一方面负责显示UI界面，提供触控按钮；另一方面将用户的设置和播放暂停指令，通过串口发送给执行终端。



图 3 界面UI调整节拍

后端执行终端，采用Arduion板外扩蜂鸣器的方案。首先，通过串口接收Stone HMI 屏幕发送过来的设置参数，据此调整节拍函数，再根据播放和暂停指令，控制对蜂鸣器的驱动。

电气节拍的实现原理相对比较简单，通过PWM引脚驱动蜂鸣器发出两种音调（实质就是两种频率）的声音作为强弱信号，而定时器控制节拍的频率。

## 四、系统硬件设计

1.显控前端硬件设计

1.1 Stone HMI模块

本文采用北京斯通恩科技有限公司的HMI屏，型号为STVC070WT-01，集成了TFT显示器和触摸控制器。它包括处理器、控制程序、驱动器、闪存、RS232/RS485/TTL端口、触摸屏幕、电源等，是一个功能强大、操作方便的整体显示系统操作系统，可由任何单片机控制。具备了如文本显示，图像显示、曲线显示以及触摸、视频和音频播放等功能，丰富了用户界面。内置的闪存可以存储数据、配置文件、图像等。





图 4 HMI屏实物图

（1）STVC070WT-01产品特性

·由任何MCU控制；

·显示图片/文本/曲线；

·65536彩色TFT显示屏；

·带/不带触摸屏；

·RS232/RS485/TTL UART接口和USB端口；

·宽电压范围；

（2）应用范围

广泛应用于各种工业领域、医疗美容设备、工程机械及车辆设备、电子仪器、工业控制系统、电力工业、民用电子设备、自动化设备、交通运输。

1.2 电气连接；

（1）供电

HMI屏的供电是5V 600mA以上的启动电流，这部分由后端数据采集处理平台的Buck降压电路产生，具体见下节的原理图；

（2）通信

HMI显控终端采用UART口与后端平台通信，由于我使用的型号是 STVC070WT-01 ，尾缀-X1（X=0 RS232 、X=4 RS485、X=1 TTL）,因此通过TTL的UART接口与其通信，波特率设置为9600bps；

1.3 开发步骤；

使用STVC070WT的TFT-LCD模块只需4个步骤：

1. 使用TOOL-2019开发软件，设计了一组漂亮的“图形用户界面”；



图 5 Stone 的GUI-EDA软件图

（2）通过调试工具，将编辑好的图形用户界面下载到TFT-LCD屏幕中；



图 6 调试工具

（3）通过RS232、RS485或TTL电平直接与客户的MCU连接；

（4）在MCU端编写一个简单的程序，让单片机通过命令控制TFT-LCD模块（十六进制代码）仅此而已。

2.后台执行终端硬件设计

2.1 硬件组成

（1）控制电路采用Arduino UNO开发板。

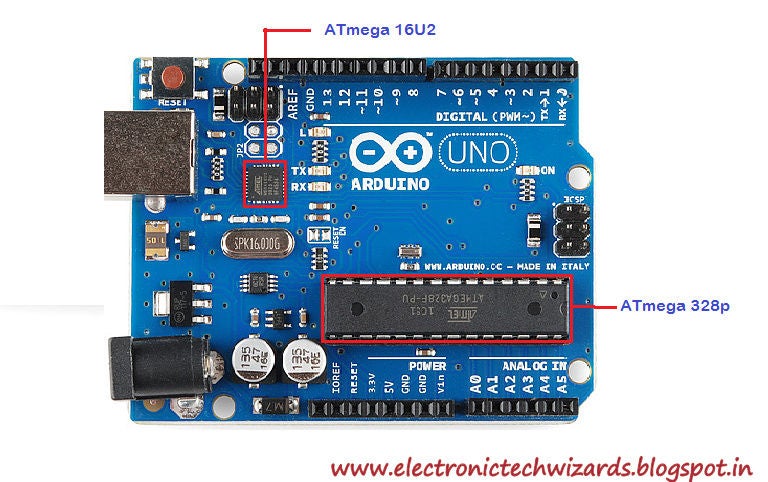


图 7 Arduion实物图

·微控制器：ATmega328p；

·工作电压：5V；

·输入电压（推荐）：7-12V；

·输入电压（限制）：6-20V；

·模拟输入引脚：6；

·每个I/O引脚的直流电流：40mA；

·3.3V引脚：50mA；

·闪存：32KB（ATmega328）；

·EEPROM：1KB（ATmega328）；

·时钟速度：16MHZ；

（2）蜂鸣器采用无源蜂鸣器



图 8 蜂鸣器实物图

蜂鸣器是一种电子发声元器件，可以发出"beep"的声音，只需要直流供电，应用非常广泛；蜂鸣器分为有源蜂鸣器和无源蜂鸣器两种，区别是内部是否有震荡源，我们选择无源蜂鸣器，需要用用2K-5K的波形脉冲信号去驱动它才能发生。

2.2主要控制芯片

ATmega328p是AVR公司生产的高性能8位单片机，拥有32KB ISP Flash、1024B EEPROM、2KB SRAM、23 GPIO、 3个定时器、10-bit ADC，以及USART和SPI接口，其资源如下所示。

􀁹 ·Flash：32(KB);

􀁹 ·CPU Speed (MIPS/DMIPS)：20;

􀁹 ·SRAM：2,048 Bytes;

􀁹 ·EEPROM: 1024 Bytes;

􀁹 ·Digital Communication Peripherals：

1-UART, 2-SPI, 1-I2C；

􀁹 ·Capture/Compare/PWM Peripherals：

1 Input Capture, 1 CCP, 6PWM；

􀁹 ·Timers：2 x 8-bit, 1 x 16-bit；

􀁹·工作温度：-40 to 85(°C)；

## 五、系统软件设计

1.显控前端软件设计；

（1）开发流程

首先，建立工程，并将所需要的图片加载到工程中去，这里我们制作一个背景图片，上面带有“速度”和“节拍”选项，通过“+”“-”来进行调节，再加上两个播放和暂停的ico图标，已满足控制的需求。

1. 利用TOOL-2019的控件制作，动态关联关系；其中主要的控件有：“text”、“Button”、“ico”、“Button return”、“Incremental adjustment”;
2. 软件模拟仿真、编译生成可执行文件；
3. 将HMI 屏幕通过调试工具连接PC端，并将可执行文件下载到屏幕中去。

（2）如何建立对“速度”的控制；

设置控件数值显示控件“Data variable display”，命名为Speed，more的值是“120”，再在其左右放置增减调节控件“Incremental adjustment”，设定增加的间隔为“10”；

（3）按照同样的方法设定“节拍”；

（4）实现旋转的播放按钮；

通信是怎么处理的，要说清楚，给出指令来；

1. 实现“播放”和“暂停”的串口通信控制；

2.后端执行终端软件设计；

（1）驱动蜂鸣器产生节拍的音调

Arduino 已经封装好了一个函数tone(),其输入参数为驱动引脚和频率，频率越高产生的音调越高，而其原理就是PWM。

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int pinBuzzer = 3; //管脚D3连接到蜂鸣器模块的信号脚

void setup()

{

pinMode(pinBuzzer, OUTPUT); //设置pinBuzzer脚为输出状态

}

void Beep(long fre,unsigned int on,unsigned int off)

{

long frequency = fre; //频率, 单位Hz

//用tone()函数发出频率为frequency的波形

tone(pinBuzzer, frequency );

delay(on); //等待1000毫秒

noTone(pinBuzzer);//停止发声

delay(off); //等待2000毫秒

}

为了实现节拍的效果，需要两种声音，我们选择比较有代表性的，300Hz和600Hz并默认600Hz作为强节拍。

（2）设定节拍的速度

节拍速度计算其实非常简单，主要是Speed就是设定的值，而节拍的方式主要是：1/4、2/4、3/4、4/4这四种，也就是在每一小节的最后一拍用强化音乐（600Hz）就可以了;

举个例子，比如120的速度、3/4的节拍，那么情况就是：各个节拍持续时间（包含鸣响和静默）为：60s÷120=0.5s，每一节共4拍，前2拍为300Hz的弱音，第3拍为600Hz的强音。

音调排序为：

Beep(300,250,250)；

Beep(300,250,250)；

Beep(600,250,250)；

再次进行函数封装，以便后续的高效设置使用。

void metronome(unsigned char metr,unsigned char speed)

{

unsigned char i;

unsigned int on=250;

unsigned int off=60000/speed-250;

for(i=0;i<metr;i++)

{

Beep(300,on,off);

}

Beep(600,on,off);

}

（3）串口通信

使用Serial.begin函数初始化串口，并编写串口类似中断接收函数和解析函数，并申明串口接收缓冲区char Uart\_RX\_Buf[9];

Serial.begin(115200);

void serialEvent()

{

  if(Serial.available()>0)

  {

char tmp = Serial.read();

//-判断是否正在接收数据报文中

if(Uart\_Packet\_IN)

{

Uart\_RX\_Buf[Uart\_Current\_Order++] = tmp;

//--接收到的数据长度 < 该数据包的设定长度

if(Uart\_Current\_Order == 9)

{

//--判断是否为包尾 && 校验和正确

Uart\_Packet\_Flag = 1; //--接收到有效数据包

Uart\_Packet\_IN = 0; //--不再处于接收数据包状态

Uart\_Current\_Order= 0; //--对序号清零

}

//-判断是否为数据包头

else if(tmp==Uart\_Packet\_Header)

{

Uart\_Packet\_IN = 1;

Uart\_RX\_Buf[0] = tmp;

Uart\_Current\_Order ++;

}

}

处理指令解析并执行的函数如下

void process(void)

{

Uart0\_Packet\_Flag=0;

switch(Uart0\_RX\_Buf[5])

{

//--判断是操作速度还是节拍

case 0x06:

{

Speed=Uart0\_RX\_Buf[8];

break;

}

//--判断是操作速度还是节拍

case 0x07:

{

Metr=Uart0\_RX\_Buf[8];

break;

}

Default:break;

}

## 六、系统运行效果测试

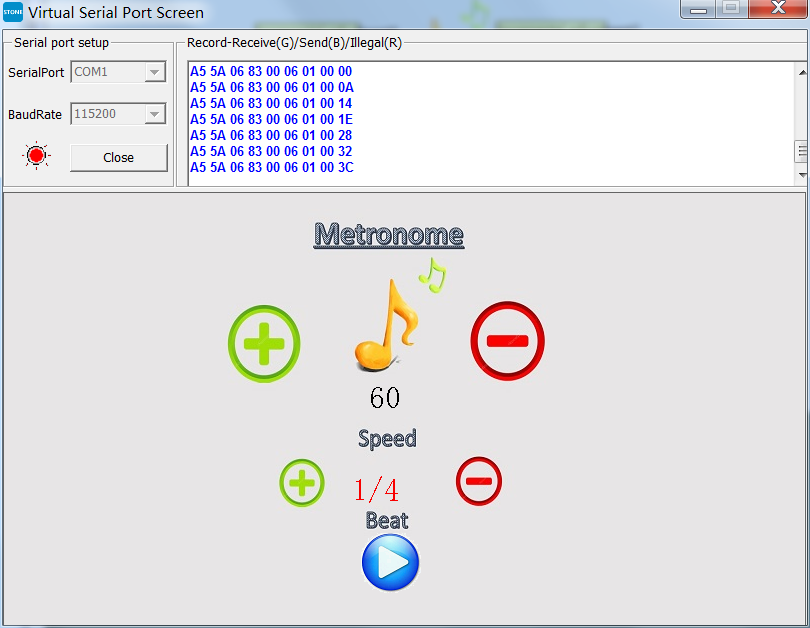
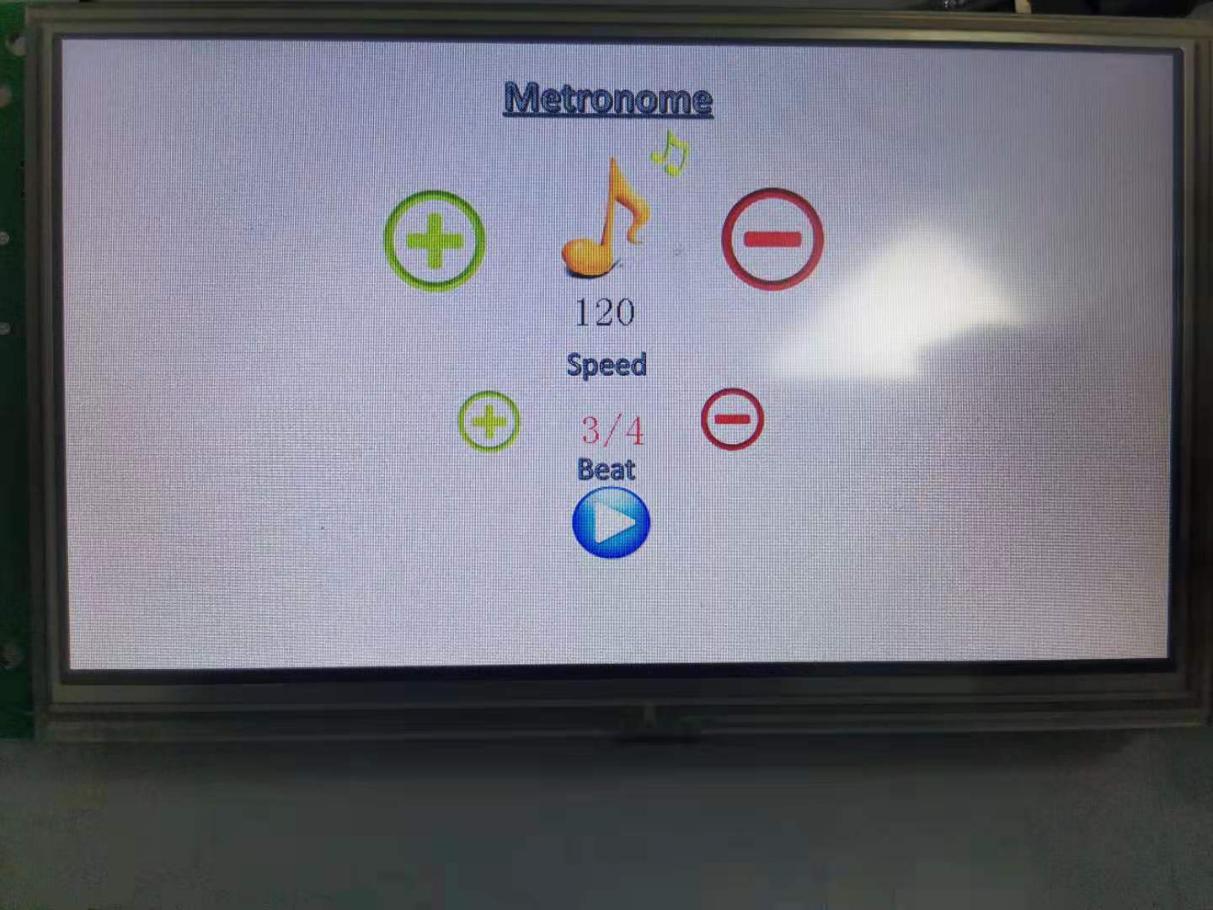


图 9 操控按键的串口指令

调整速度串口同步打印指令，连接Arduion开始实地运行，效果如下图所示。



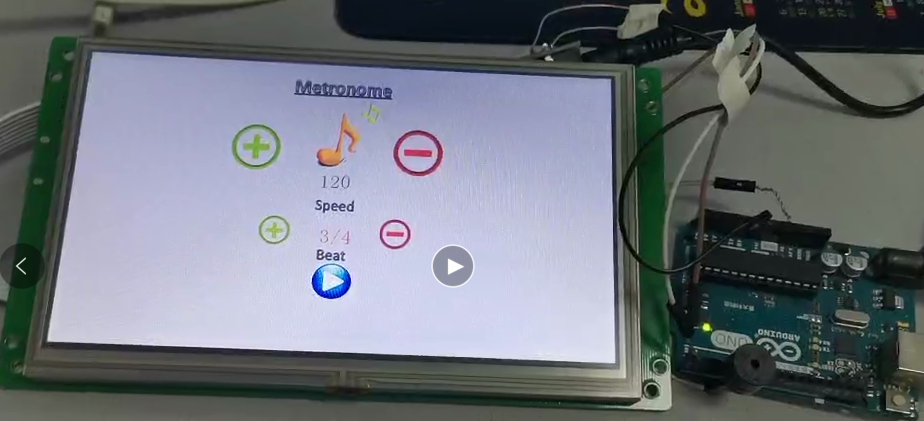


图 10 系统运行示意图

## 七、未来展望

这次的案例是使用屏幕和Arduino打造了一个节拍器，只用到了HMI屏的触控、按键、串口等功能，对于其播放音乐的功能还没有用上，后续会制作“听音乐猜歌名”的小游戏。