浙江大学

本科实验报告

课程名称：	电子电路系统综合实验
姓　　名：	王天亮、王一言、徐建东
学　　院：	信息与电子工程学院
专　　业：	信息工程
学　　号：	3200102777、3200300758、3200103285
指导教师：	施红军、邓靖靖

2023 年 7 月 12 日

便携式脉搏测量仪

第 11 组

一、实验目的

1. 掌握并熟悉放大电路、比较电路及偏置电压的原理。

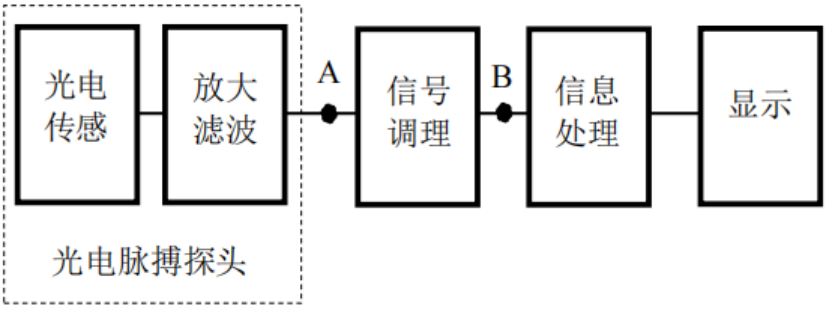
2. 利用 Altium Designer 来设计电路原理图与 PCB 设计图。

3. 学习 STM32F103 型单片机，利用其的软件代码设计来实现其所带来的功能。

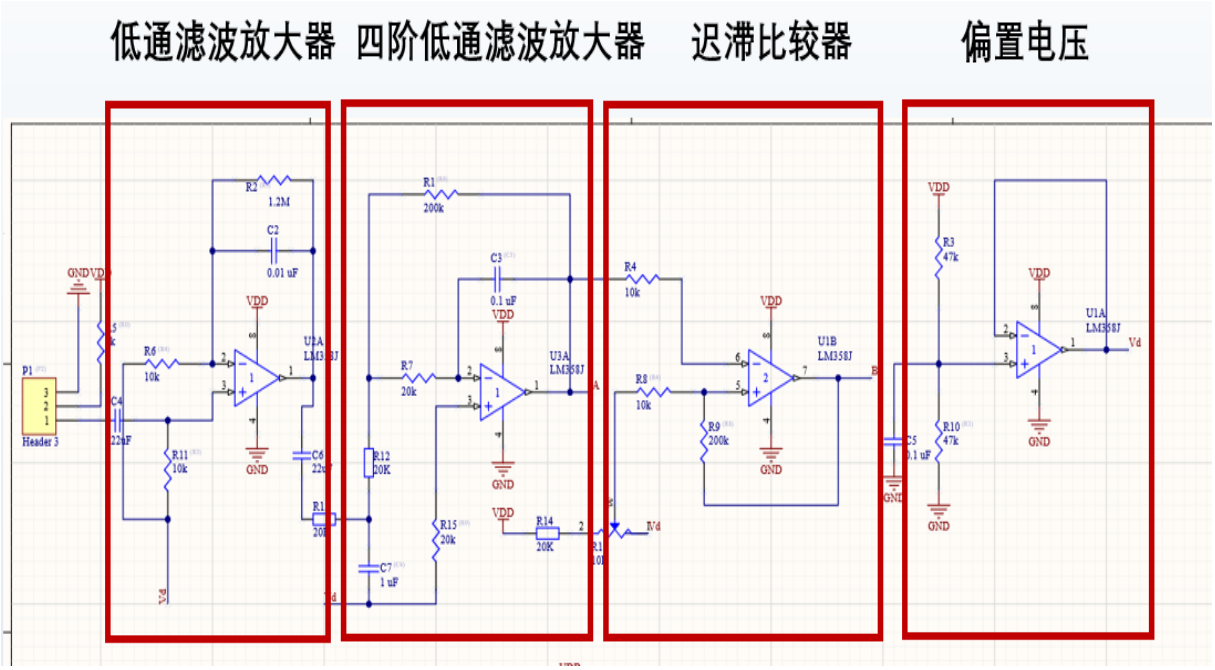
二、实验原理

1. 系统设计

本设计主要是通过 STM32F103 单片机实现了便携式脉搏的测试，由光电传感器采集到脉冲信号，经过信号的放大、滤波、调理和比较后，将输出的信号通过单片机的获取并进行处理，最终在 OLED 显示屏上显示。



2. 设计电路



3. 电路分析

（1）低通滤波器与四阶低通滤波器

容许低频信号通过，但减弱（或减少）频率高于截止频率的信号的通过。 对于不同滤波器而言，每个频率的信号的减弱程度不同。 当使用在音频应用时，它有时被称为高频剪切滤波器，或高音消除滤波器。在这个脉搏测试仪当中，主要是过滤掉其他除脉搏以外的任何其他干扰信号。然后，四阶低通滤波器，主要是把被滤波后的波形从高电平到低电平可以在短时间来实现。

（2）迟滞比较器

主要功能是将最终输出结果来决定是否为高电平或低电平。

（3）偏置电压

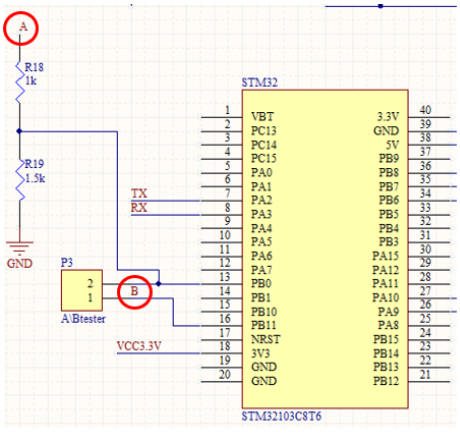
通过两个 4.7kΩ 的电阻的分压，将 5V 电压输出为 2.5V 电压，把输入信号抬高 2.5V，便于单电源供电的运放工作。

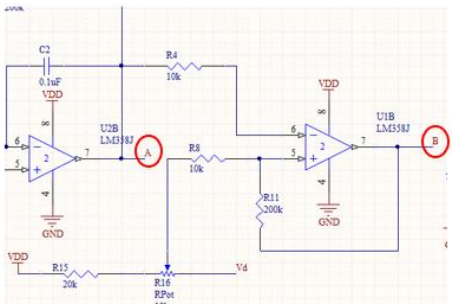
（4）参数计算

放大倍数 A=（1+R_2/R_6）*（1+R_1/R_7）≈1200；滤波$\frac{1}{2\pi R_6 C_4} \leq f \leq \frac{1}{2\pi R_2 C_2}$，即限定心率为 43.4—800 之间。

4. 单片机输入输出端口

①

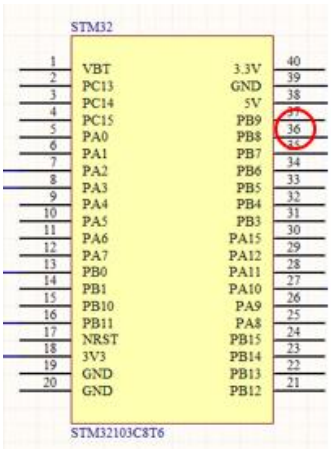


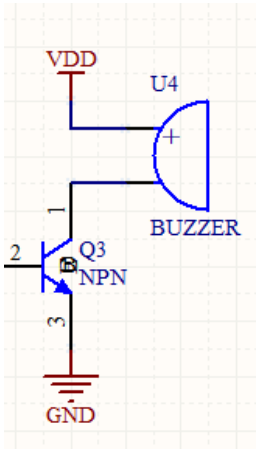


A：滤波放大后的 5V 信号通过电阻分压变为 3V 信号，用于显示波形

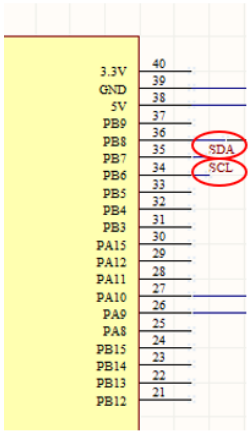
B：比较器之后的信号用于脉搏测量，经过 ADC 接口进入单片机

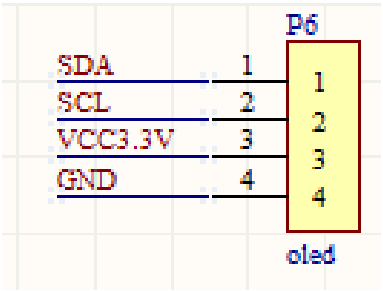
②超出阈值后蜂鸣告警





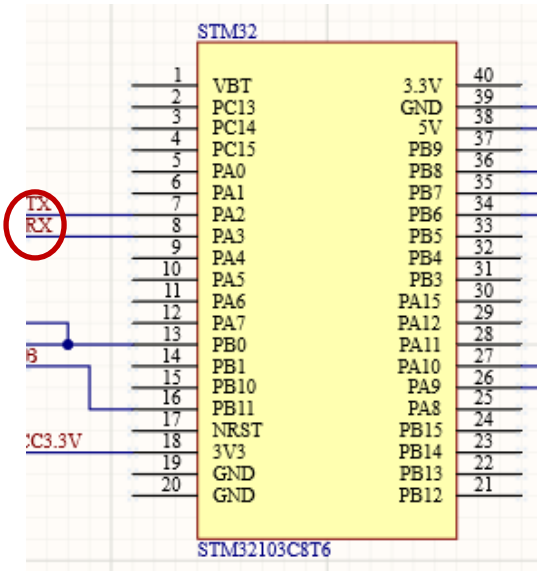
③OLED 显示：

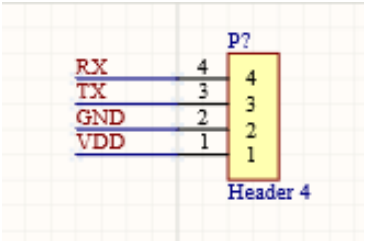




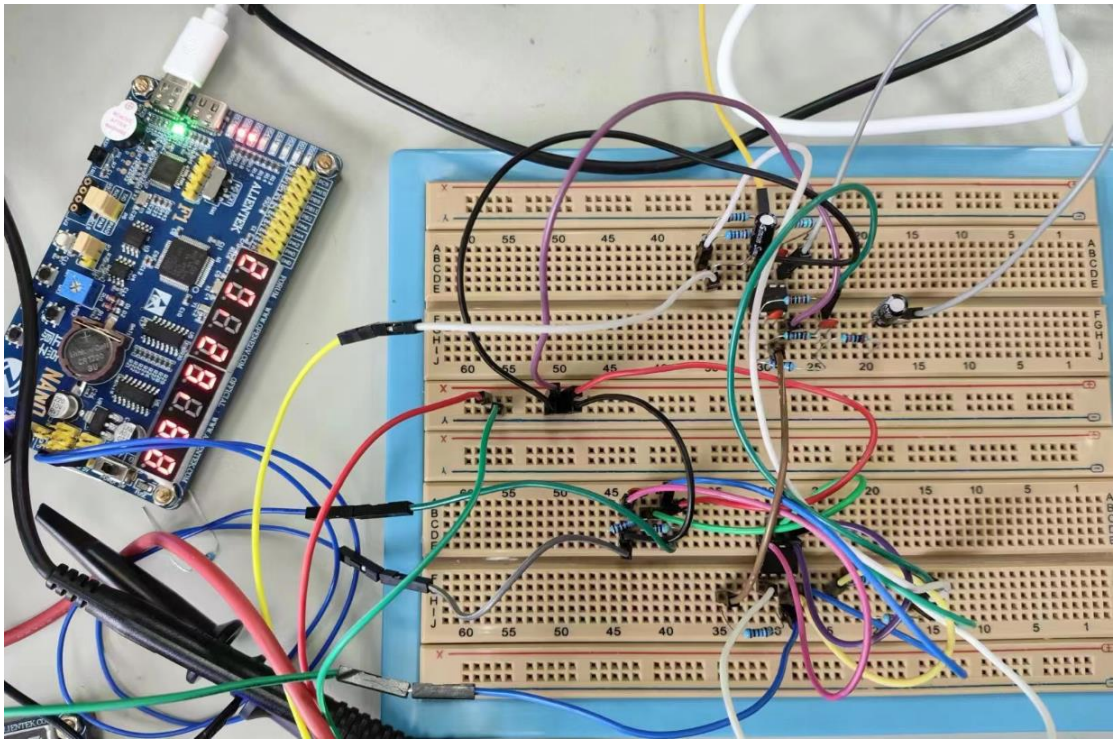
控制自动启动测量，显示心率，一分钟测量完自动待机

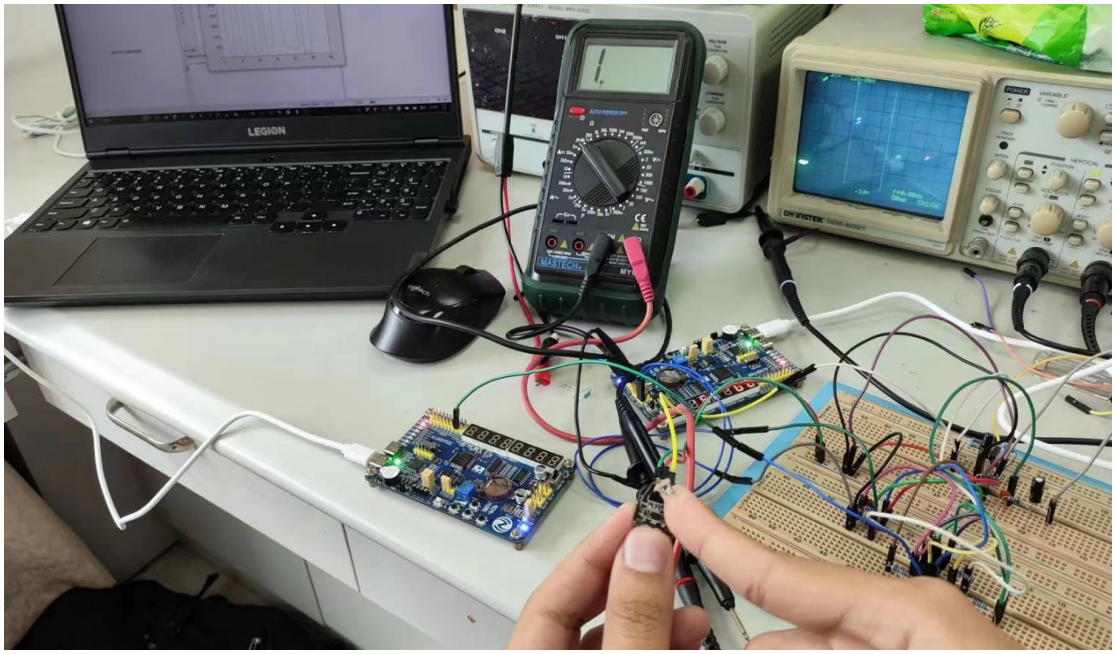
④蓝牙模块



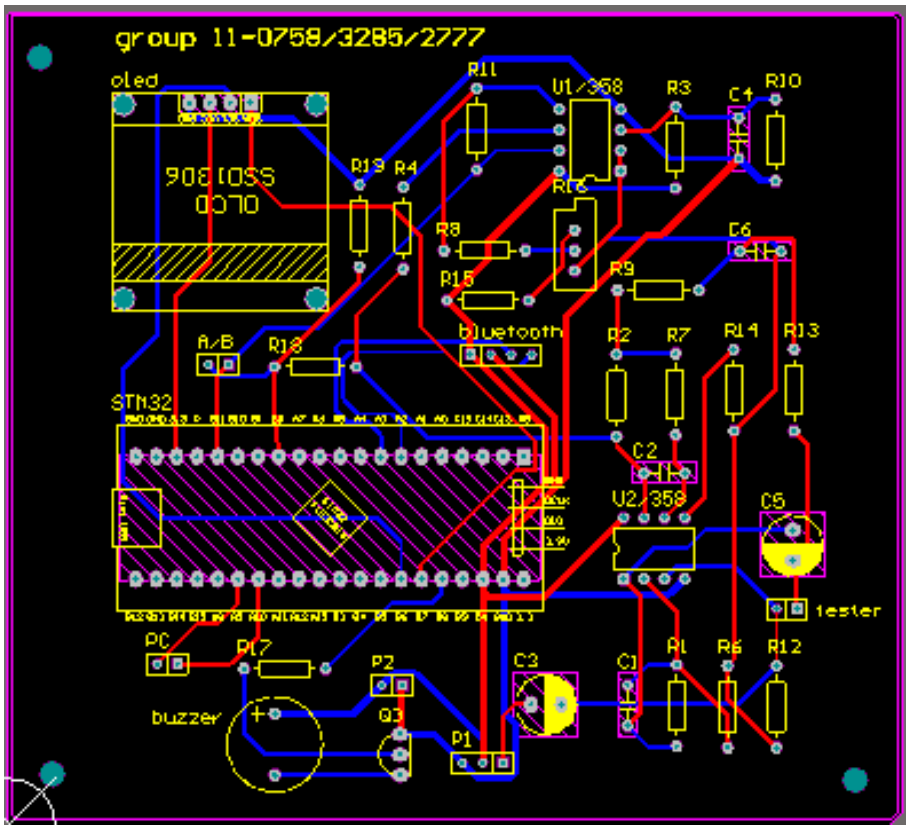


5.面包板调试

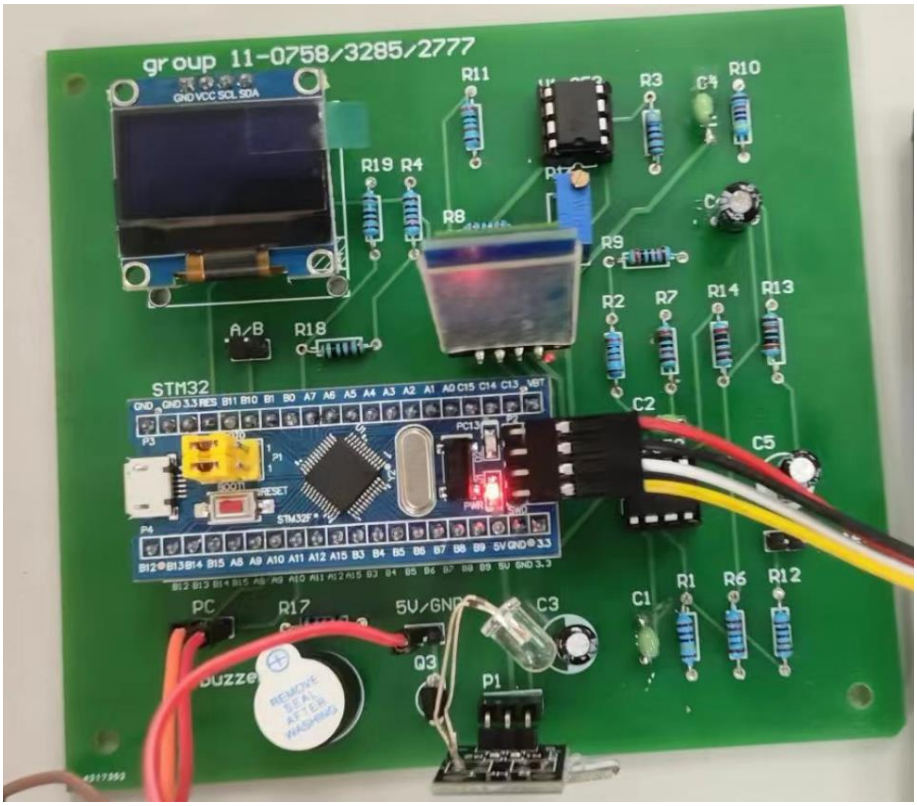




6.PCB 电路设计图

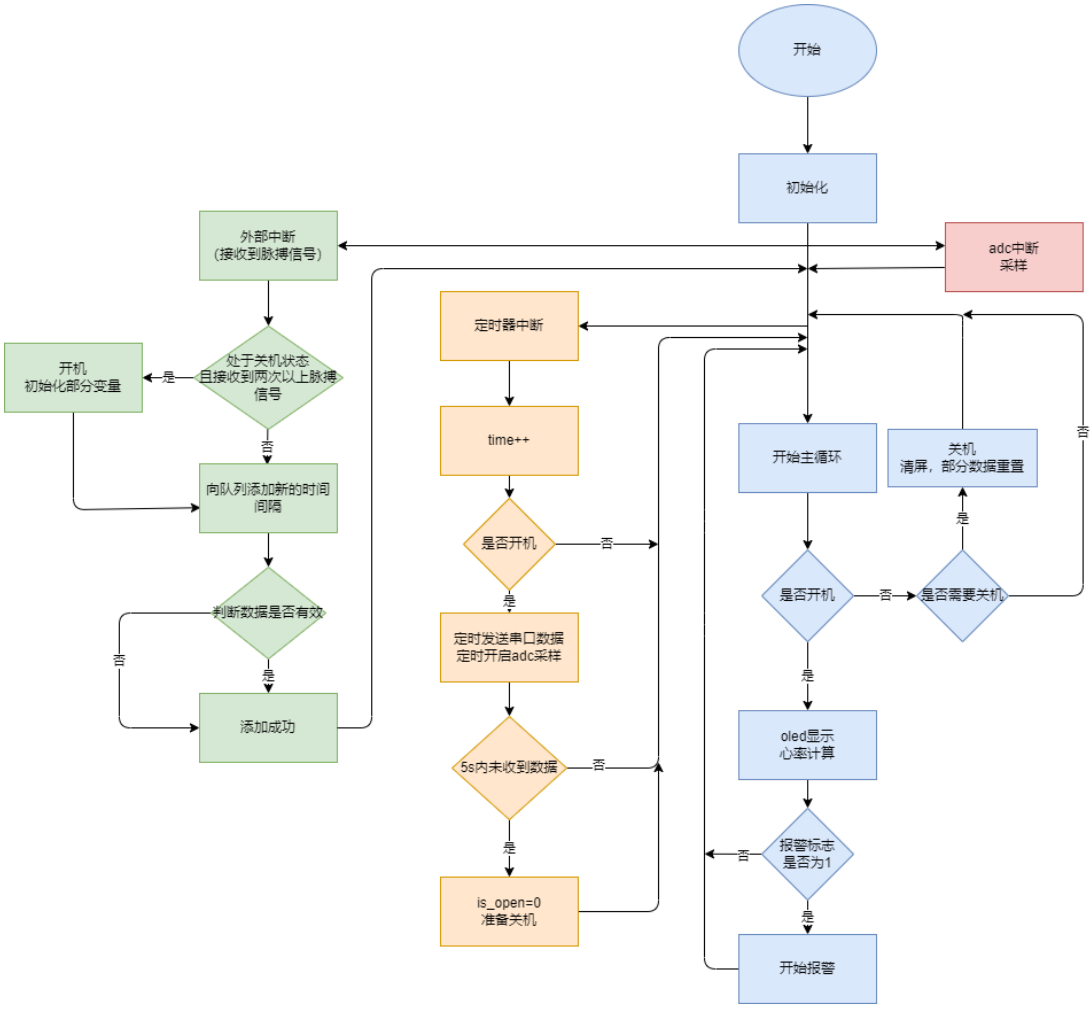


焊接完成后的电路图



三、 软件部分

1. 软件总体方案



a) 程序主要分为两个部分，一部分为数据的接受与处理，一部分为结果的计算与显示。

b) 为保证测量结果更为准确，接受到的数据会经过一系列条件的过滤，最终被存储到一个长度为 60 的队列中，最终会根据该队列中时间间隔的平均值计算最终心率。

c) 结果的显示分为了三个部分

i. 使用 oled 屏显示心跳动画，心率以及告警信号。

ii. 心率和心率波形由 adc 采样后通过串口 1 发送到电脑，然后由 matlab 编写的程序进行显示。

iii. 心率数据以两秒一次的速度通过串口二连接的蓝牙模块发送到手机进行显示。

2. 核心代码（为保证简洁，展示代码均省略 STM32CubeMX 自动生成部分）

a) 部分变量与宏定义

宏定义

```
// 时间差队列相关
#define MINLISTSIZE 15 // 最小列表长度（超过该值视为心率开始稳定，开启报警）
#define MAXLISTSIZE 60 // 最大列表长度（维护时间差队列的长度）
```

```
// 告警相关
#define MAXSPHYFMUS 150 // 告警心率上阈值
#define MINSPHYFMUS 50  // 告警心率下阈值
// 计时器相关
#define MAXTIME 100000 // 计时器的最大值
#define UART1TIME 20   // 串口 1 发送时间间隔
#define UART2TIME 2000 // 串口 2 发送时间间隔
#define ADC1TIME 20    // adc 采样时间间隔
#define CLOSETIME 5000 // 关机时间间隔
// 数据过滤相关
#define MIN_DELTA 333       // 最小时间间隔
#define MAX_DELTA 1000      // 最大时间间隔
#define MAX_DELTA_DELTA 250 // 最大时间间隔变化量
```

重要变量定义

```
int too_high; // 报警标志
int too_low;  // 报警标志
int is_open = 0;   // 是否开机
int is_closed = 0; // 是否已经关机
int heart_beat = 0; // 心跳标志
uint8_t sphygmus_num; // 测试到的心跳次数
uint16_t adc_value;   // adc 采样值
uint16_t last_value;  // 上个 adc 采样值
double sphygmus;      // 脉搏
uint32_t time;   // 当前时间
uint32_t l_time; // 上次心跳时间
uint32_t delta_time_list[MAXLISTSIZE]; // 脉搏间的时间间隔列表
UART_HandleTypeDef current_uart;       // 当前使用的串口
```

b) Main 函数

```
int main(void)
{
  /* USER CODE BEGIN 2 */
  init();      // 初始化系统
  is_open = 0; // 默认关闭
  HAL_ADC_Start_IT(&hadc1);
  /* USER CODE END 2 */
  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
```

```
  while (1)
  {
    /* USER CODE END WHILE */
    /* USER CODE BEGIN 3 */
    if (is_open) // 若处于开机状态
    {
      display();      // oled 显示
      give_alarm();   // 满足条件时报警
      get_sphygmus(); // 计算当前心率
    }
    else if (!is_closed) // 若退出开机状态且还没有完成关机
    {
      HAL_GPIO_WritePin(GPIOC, Alarm_Pin, GPIO_PIN_SET); // 关闭蜂鸣
器
      OLED_Clear(0x00);                                  // oled 清屏
      sphygmus_num = 0;                                  // 清除心跳计数
      is_closed = 1;                                     // 完成关机
    }
  }
  /* USER CODE END 3 */
}
```

main 函数包括了部分需要持续运行的逻辑，包括 oled 的显示，心率的计算，以及报警。除此之外在主循环中若发现系统已经关闭，则会执行关机逻辑，并将已关机标志置为 1。

c） **定时器中断**

```
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{
  if (htim == (&htim1))
  {
    time++;
    if (time > MAXTIME)
    {
      time = 0;
    }
    if (is_open)
    {
      // 测量一次
      if (time % ADC1TIME == 0)
      {
```

```
        HAL_ADC_Start_IT(&hadc1);
      }
      // 串口 1 输出
      if (time % UART1TIME == 0)
      {
        current_uart = huart1;
        printf("%d %f\r\n", adc_value, sphygmus);
      }
      // 串口 2 输出
      if (time % UART2TIME == 0)
      {
        current_uart = huart2;
        printf("%f\r\n", sphygmus);
      }
      if (get_delta_time() > CLOSETIME)
      {
        on_close();
      }
    }
  }
}
```

定时器中断负责整个程序的计时与调度，一方面维护一个 **time** 变量用于计时，一边在抵达设定好的时间时定时调用对应的程序，比如以 **50hz** 的频率进行 **adc** 采样，以 **2s** 一次的频率向蓝牙发送信息，已经失去脉搏信号一段时间后自动待机等功能

c）外部中断

```
// 外部中断（用于测量脉搏）
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
  if (GPIO_Pin & GPIO_PIN_11)
  {
    if (!is_open && sphygmus_num > 2) // 接收到两次以上的脉搏信号后，系统开机
    {
      start(); // 开机初始化
      l_time = time;
      sphygmus_num = 0;
    }
```

```
    add_delta_time(); // 向队列添加新的时间间隔
    heart_beat = 1;   // 心跳标志置 1
    l_time = time;    // 记录上一次脉搏的时间
  }
}
```

对脉搏信号的接受主要通过外部中断实现。

接收到两次以上脉搏信号时，系统结束待机状态，进行开机初始化。每次接收到脉搏信号时，向队列中添加一个新的时间间隔，交由数据处理部分进行处理，同时记录本次心跳的时间，便于下次心跳时求平均值

3. 数据的处理

数据的处理主要由以下几个部分组成：

a) **添加数据同时过滤有问题的数据：**

```
// 添加时间间隔
void add_delta_time()
{
  uint32_t delta_time = get_delta_time(); // 获取时间间隔
  // 过滤掉间隔过短和过长的脉搏
  if (delta_time < MIN_DELTA || delta_time > MAX_DELTA)
  {
    return;
  }
  // 过滤掉间隔不稳定的脉搏
  int delta_delta = delta_time > delta_time_list[MAXLISTSIZE -
1] ? delta_time - delta_time_list[MAXLISTSIZE - 1] :
delta_time_list[MAXLISTSIZE - 1] - delta_time;
  if (sphygmus_num >= MINLISTSIZE && delta_delta >
MAX_DELTA_DELTA)
  {
    return;
  }
  // 队列元素左移一位
  for (int i = 0; i < MAXLISTSIZE - 1; i++)
  {
    delta_time_list[i] = delta_time_list[i + 1];
  }
  // 添加新的时间间隔
  delta_time_list[MAXLISTSIZE - 1] = delta_time;
  // 脉搏数加一
```

```
      sphygmus_num++;
    }
```

b) 根据队列元素计算心率平均值

```
// 脉搏间隔平均值
double average()
{
  uint32_t sum = 0;
  for (uint8_t i = 0; i < MAXLISTSIZE; i++)
  {
    sum += delta_time_list[i];
  }
  int num = sphygmus_num > MAXLISTSIZE ? MAXLISTSIZE :
sphygmus_num;
  return (float)sum / num;
}
// 获取脉搏
void get_sphygmus()
{
  double avg_time = average();
  double sp = 1000.0 / avg_time * 60;
  if (sphygmus_num > MINLISTSIZE)
  {
    too_high = (sp > MAXSPHYFMUS);
    too_low = (sp < MINSPHYFMUS);
  }
  sphygmus = sp;
}
```

4. 结果的显示

a) oled 显示

```
// oled 显示
void display()
{
  static char buf[] = {"rate: "};
  static char buf1[] = {"rate:"};
  static char sphygmus_str[5];
  Oled_Display_String(0, 80, buf1);
  Oled_Display_Pic(50, 50, 0, 15, heart_small);
```

```
  Oled_Display_String(0, 80, buf);
  Oled_Display_String(3, 80, to_string(sphygmus, sphygmus_str, 5));
  if (heart_beat)
  {
    heart_beat = 0;
    Oled_Display_Pic(50, 50, 0, 15, heart_large);
  }
}
```

Oled 显示基本使用提供的 demo 工程，但添加了一个心脏跟随脉搏信号跳动的功能，外部中断接收到心跳信号时会将 heart_beat 标志位设置为 1，display 时，若该变量为 1 则播放一次心脏变大的动画，并将标志位设置为 0

b) 串口 2 显示（蓝牙模块）

```
// 输出
int fputc(int ch, FILE *f)
{
  HAL_UART_Transmit(&current_uart, (uint8_t *)&ch, 1, 10);
  return ch;
}
```

由于涉及到两个串口，但 fputc 函数的参数不能改变（该函数在标准库中定义），所以设置了一个中间变量 current_uart，在每次 printf 时将该值设置为对应的串口，即可实现两个串口显示不同数据。

```
    // 串口 1 输出
    if (time % UART1TIME == 0)
    {
      current_uart = huart1;
      printf("%d %f\r\n", adc_value, sphygmus);
    }
    // 串口 2 输出
    if (time % UART2TIME == 0)
    {
      current_uart = huart2;
      printf("%f\r\n", sphygmus);
    }
```

c) 串口 2 的显示

Stm32 部分显示代码与串口 2 类似，故此处仅展示 matlab 的代码,代码如下：

```
if exist('s1', 'var')  %判断上一次打开的端口有没有关闭
    clear s1;
end
```

```
%新建串口对象
s1=serialport('COM9',115200);      %设置串口波特率
s1.InputBufferSize = 8000;         %输入缓冲区长度 8000 字节
s1.Timeout=4;
%ADC 采样
index1=201;                %adc 采样 bufffer 的大小
buffer1=zeros(1,index1);  %定义 adc  buffer
%配置图标的显示方式
subplot(2,1,1);
plot1=plot(buffer1);
axis([0 inf 0 4096]);
%心率
index2=51;                %心率数据 bufffer 的大小
buffer2=zeros(1,index2); %定义心率 buffer
%配置图标的显示方式
subplot(2,1,2);
plot2=plot(buffer2);
axis([0 inf 50 150]);
yticks(50:10:150);
while 1
    %从串口读取一行数据
    str=readline(s1);
    if not (isempty(str))    %若成功读取到数据
      datas=strsplit(str); %以空格分割字符串
      data1=datas(1);                    %第一部分为 adc 数据
      buffer1=circshift(buffer1,-1);      %将 buffer 左移一位
      buffer1(index1) = str2double(data1);%将 buffer 的最后一位设置为新的值
      set(plot1, 'YData', buffer1);       %设置折线图表新值
      data2=datas(2);                    %第二部分为心率数据
      buffer2=circshift(buffer2,-1);      %将 buffer 左移一位
      buffer2(index2) = str2double(data2);%将 buffer 的最后一位设置为新的值
      set(plot2, 'YData', buffer2);       %设置折线图表新值
      drawnow
    end
end
```

串口显示的代码主要分为两部分：

1. 第一部分是串口的设置以及变量和图表的设定

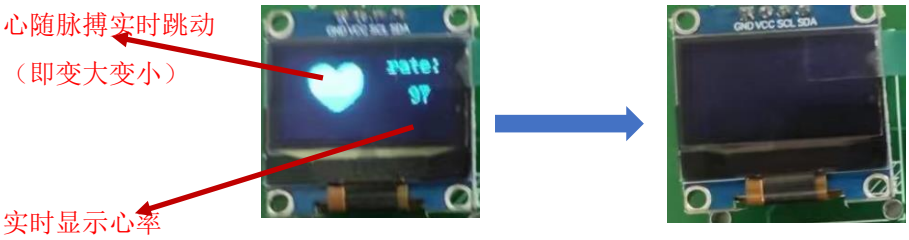
2. 第二部分为一个 while 循环，不断接受串口数据，将新的数据添加到数组末端，并设置新的折线图进行显示

四、 实验结果

1. 各项功能实现

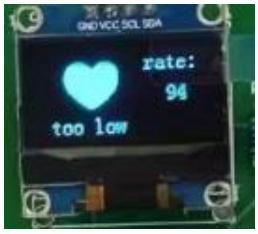
① 自动启动测量和自动待机

将手指放入探头内会自动启动并实时测量心率，离开探头会自动待机



② 脉搏次数上下告警门限

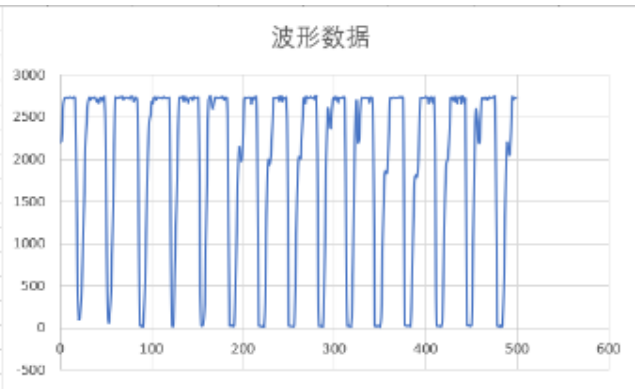
因正常门限不会触发，因此更改代码，分别使超过 80 和低于 100 发出“too hign”和“too low”警告，验证其警告功能。上下告警门限功能正常。

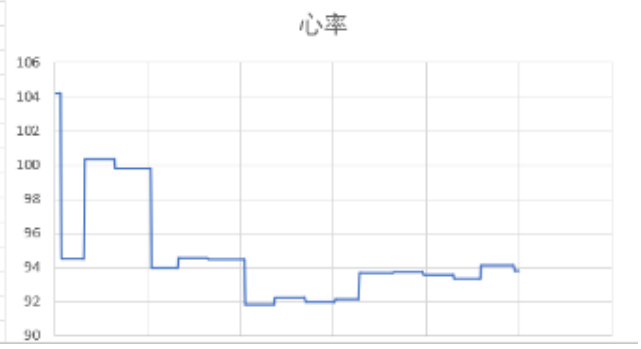


③ 异步串口传输数据和显示波形

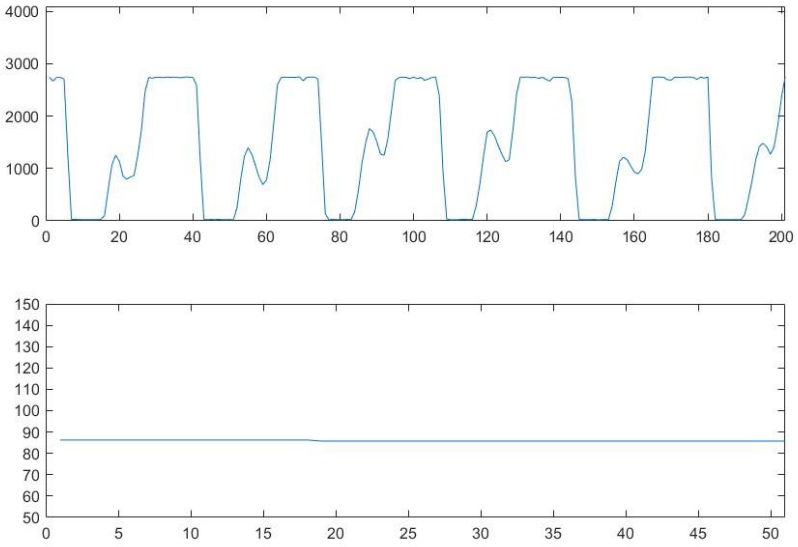
将数据传输到串口，并导入 excel 生成波形

	波形数据	心率
2	2207	104.167
3	2397	104.167
4	2654	104.167
5	2730	104.167
6	2735	104.167
7	2738	104.167
8	2742	94.4882
9	2735	94.4882
10	2740	94.4882
11	2745	94.4882
12	2739	94.4882
13	2739	94.4882
14	2735	94.4882
15	2740	94.4882
16	2745	94.4882
17	2735	94.4882
18	2139	94.4882
19	1147	94.4882
20	449	94.4882
21	106	94.4882
22	111	94.4882
23	239	94.4882
24	430	94.4882
25	739	94.4882
26	1181	94.4882
27	1580	94.4882
28	2063	94.4882
29	2350	94.4882
30	2531	94.4882





④ MATLAB 实时显示波形



⑤ 蓝牙模块无线传输

1、 测试数据处理

组员 1	1	2	3	4	5	6
测试数据	88	82	89	83	86	77
手环数据	85	80	84	83	86	76
误差	3.53%	2.50%	5.95%	0.00%	0.00%	1.32%

组员 2	1	2	3	4	5	6
测试数据	97	100	91	96	95	89
手环数据	94	98	93	95	97	88
误差	3.19%	2.04%	-2.15%	1.05%	-2.06%	1.14%

组员 3	1	2	3	4	5	6
测试数据	85	83	90	86	86	88
手环数据	84	81	89	84	88	88
误差	1.19%	2.47%	1.12%	2.38%	-2.27%	0.00%

五、 实验调试

整个实验过程收获很多，但过程中也遇到了很多问题，出现的问题有硬件、软件和最终调试的部分。

硬件：

①电解电容方向标反，本实验电解电容用于直流电路电容滤波，如果装反，会出现滤波作用大大降低，更会出现电解电容过压的问题。

②面包板调试时，万用表测试电压明显不对，再用万用表测杜邦线,发现线是断的。

软件：

①因为开始是在大板 STM32F103RBT6 上学习的，后面移植到小板 CBT6 上出现了很多引脚、电源和信号的设置与连接问题。在查阅资料后成功修复

②oled 显示屏的移植也出现了微秒延迟、管脚配置和初始化的问题（由于开始时间较早，此时还没有发布修改版的 oled 代码，我们独立解决了 oled 代码的移植问题）。

③使用蓝牙模块时，起初没有分清楚串口通信和蓝牙协议通信的关系，后来发现是单片机与蓝牙模块通过串口通信，蓝牙和手机通过蓝牙协议通信。此外串口传输时蓝牙模块与单片机的波特率设置不同也导致了许多乱码，最后同步蓝牙模块和板子的波特率后解决。

④Matlab 起初接收数据显示时非常慢，在 adc 采样大于约 10hz 时 matlab 的显示就已经跟不上串口发送数据的速度了，通过查阅资料发现不应该每次刷新使用 plot 函数显示图表，在将原本程序修改为在循环外定义 plot 变量，然后在循环中使用 set 函数设置后解决了该问题。

最终调试:

①红色的瓷片电容相比于最终使用的青色电容误差较大，更换后实时心率跳动变化趋于稳定。

②异常数据处理，尽管硬件电路已经实现了足够的滤波放大和调理，但实验测试时仍会出现测出两次脉搏跳动的时间变化很大，因此我们在软件上对此做出了限制，剔除了明显的异常数据。

③最终调试时会出现 oled 显示屏乱码后来不显示的问题，最后发现直接用四并杜邦线连接两个板子是 3.3V 供电，供电不足，最后在我们提前准备的电源引脚上接上 5V 供电可以正常工作。

六、个人心得

王天亮

本次项目中，我主要负责和完成了软件部分的工作，此外还与徐建东同学一起完成了面包板和 pcb 的调试，以及 pcb 上部分芯片座和排母的焊接。

在短学期期间，我收获了许多，一方面是知识和技能的收获，我了解到并实践了单片机这样运用范围广泛的强大芯片，在软件方面，我学会了 stm32 的使用和编程，认识到了中断这一机制的必要性与优雅性，中断这一机制的加入使得原本完全单线程的单片机程序有了部分多线程程序的特性，使得单片机可以“同时”进行多项任务，为单片机的编程带来了极大的灵活性。也在项目中对定时器中断，外部中断和 adc 中断等进行了实践。在硬件方面，我也对运放组成的放大器，迟滞比较器和偏置电压生成电路等常见的电路模块的原理和应用有了更深的了解。

另一方面，我也收获到了很多技能或知识之外的提升，经历了一次完整的项目实践后，我对电子系统的设计，仿真，编程，调试等流程有了更深的认识。除此之外我认为对于我提高最大的是自学能力和调试能力的提升。在短时间内学习一项新的技能，从寻找学习资料，了解原理，开始尝试实践，再到出现问题，定位问题，查找资料，解决问题的过程十分令人着迷，这个过程虽然充满了困难，会遇到电脑环境没配置好，查到的资料版本有问题，硬件接触不良以及一些代码逻辑有误等问题，但在漫长的调试，定位问题，查找资料，解决问题的过程中，我很有收获，调试过程中我可以积累到很多经验，避免下次再犯这样的问题，查找资料的过程中，我能学到很多知识，等到这个问题解决的一瞬间，当系统按照我们的意愿正常运行的那一刻，喜悦的表情便出现在我们的脸上，心跳加速，肾上腺素飙升，那一刻我们的成就感是无与伦比的。回想起高中的时光，我想这样的实践带来的成就感正是我选择这一专业的原因。

至于对这门短学期课程的建议的话，首先我认为对于我们工科学生来说，像这样的实践课程是非常有必要的，毕竟我们专业理论所学最终还是要落实到真正的实物上，非常感谢各位老师能够给我们这样一门有趣而富有意义的课程。可能唯一的建议是希望类似的课程能提供更多的选题范围以及更多的自由发挥部分。我想这样可能会更加有趣。

最后还要感谢各位老师的辛勤付出和耐心辅导，以及两位队友尤其是徐建东同学的努力，我们共同度过了一段充实且富有意义的日子。

徐建东

主要工作介绍：

①电路原理图设计，虽然最终还是采用了参考电路的设计方法，但我也通过学在浙大和钉钉的资料学会了用运放设计放大滤波电路、电源供电设计。

②电路仿真，在 orCAD 上进行仿真，遇到的最大问题是如何设计一个有不同频率不同幅度的待测信号源，中途试过信号发生器无果，最后发现直接用几个交流电源串联起来即可。

③面包板设计，因为仍是害怕电路设计出问题，我们向老师借用了面包板进行调试，好在发现面包板搭起来能够明显看到脉冲，才放下心来。期间遇到的问题就是有的杜邦线是断的，开始时却没有发现。

④PCB 板设计，遵循了各个模块布局在一块的原则，在后面焊接和调试时发现还是方便了许多。当时设计 PCB 板时发现很多东西都忘了，还是找到了电设 II 的课件重新学习，按步骤设计，复习着感觉又学到很多这方面的知识。

⑤焊接，板子到手后，我们小组都很开心，迫不及待一起来焊接。

⑥软件代码学习，跟着课件学习了小灯闪烁、按键按钮、外部中断、定时中断、串口通信、ADC 和 oled。但是我们小组王天亮同学太强了，代码基本都设计好了，故最后代码都是用他的。但我也不想软件代码一点不写，于是还是在网上查找资料，学习了串口通信并在 MATLAB 上实时显示波形，发现结果显示很慢，很卡顿，并且 MATLAB 因很多数据来不及处理会导致延时很高。最终和王天亮同学一起学习，将每次生成图像由接收数据的循环内移到循环外，并在循环内通过 set 函数刷新图像的“YData”，动态图像的显示明显流畅顺滑起来。

⑦最终调试和展示，虽然各个步骤我们都尽量让它能正常工作，在最终调试的时候，仍是遇到许多问题。比如因电源 5V 供电的问题导致显示屏乱码和不能工作，异常太多数据导致心率很不稳定，蓝牙模块的信号干扰导致手指离开红外探头仍不能正常待机，等等。但好在我们小组三人通力合作，共同努力，最终将各项问题统统解决，在我心中也算是完美完成了这次实验。

这次试验虽然过程困难，各项过程对我们来说都异常艰难，但好在我们一起学习一步步解决各个问题，终于在完成实验后，也是收获很多，复习了模电电路的知识，熟练了电路图和 PCB 设计，学习了很多 cubeMX 软件和许多代码设计，

积累一些调试的经验。总的来说学习了许多知识，面对困难也有了更多自信，更要谢谢老师们的帮助和解惑。

王一言

本次实验让我获益匪浅，认识了很多关于电路方面的知识。短学期开始之际，我自身感觉要做的东西非常迷茫，搞不懂接下来要做什么事情。老师也介绍了很多关于各种实验的大概概念，随后我找到了我的实验小组，其他两位成员是徐建东和王天亮。实验初期，我和徐建东负责通过参考电路图来修改我们所要实现的电路，而王天亮同学则深入学习 STM32 要在电路里的实现。当然，大家也是学这个单片机的原理和应用。通过实验讲义也认识了很多 STM32 的强大功能，比如 ADC 转换功能和 OLED 显示等，ADC 转换功能可以通过 STM32CubeMX 的软件来编程及下载，以 C 语言为编程环境，此软件功能极强，可通过可视化的时钟设置和其他功能设置，可以生成以 C 语言为主的代码，从那个代码编写我们想要实现的东西，代码上传后，可通过 XCOM 软件显示采样的信号数字，我也学了通过 MATLAB 来实时显示波形，通过其的 serialport()函数来显示结果，也学到了 x 轴的显示滚动结果。我的 ADC 实验结果已在学在浙大上上传。电路设计图首先是我通过参考电路来进行画的。幸亏，施红军老师和邓靖靖老师的细心指导下，也改了很多器件，比如以为显示用的是 LCD 灯，施老师才说我们要用拥有 4 个引脚的 OLED 显示灯，于是徐建东同学帮了我优化电路图。PCB 板的设计主要是以原理电路图为主的，我画了 PCB 板电路图，徐建东同学也帮了我优化

PCB 板电路图。最终，通过我俩的共同努力下，终于把正式版电路板提交给施红军老师。印刷完电路板后，我们就开始焊接阶段，这是我第一次用实验室的电烙铁来焊接，之前因为疫情关系，之前是在家用便携式手枪焊接的，所以我的电子工程训练是在家完成的。到校后，终于能体验这里的温控电烙铁了，也发现这里的吸焊枪跟在家的不太一样，这里的吸焊枪吸得比较强，一下子把所有的锡都吸光。然后在最后碰到了一些电路问题，就是脉搏显示的不够稳定，原因是我们用的电容精度不够高，于是通过调试完后，成功把部分电容替换掉，从而得到更稳定的结果。最后，我想感谢两位与我并肩作战的队友，虽然我的贡献比是最少的，但是通过本次综合实验，让我感受到玩单片机的过瘾，电路设计的乐趣。