单片机课程设计报告

题目：多功能智能抢答器

姓名：魏麟 学号：2220214318 专业班级：通信工程3班

1. 摘要  
    本次课程设计要求使用stm32系列单片机实现基本实际场景功能，旨在培养学生掌握单片机系统设计的基本方法和一般步骤，设计一个基于STM32的多功能智能抢答器是为了满足教育培训、竞赛活动等场景中对抢答功能的需求。通过智能抢答器，可以实现抢答顺序管理、计分统计、用户交互等功能，提升活动的互动性和趣味性。主要内容包括硬件设计、软件设计和功能实现等方面。抢答环节是一个常见的互动环节，能够激发参与者的积极性和竞争性。
2. 任务和要求  
    设计一个多功能智能抢答器，要求能够实现定时抢答功能，加以OLED显示屏和LED灯显示其状态结果。
3. 目的和意义  
    学会掌握单片机系统设计的基本方法和一般步骤，包括软件设计，硬件设计，软硬件联调，系统综合分析。
4. 系统方案的论证和选择

硬件设计：选择适合的STM32系列芯片，根据需求选择性能和外设丰富程度。设计电源模块，保证系统稳定可靠。添加按键、OLED显示屏、LED灯等外设，用于用户交互和提示。

软件设计：编写STM32的固件程序，包括初始化设置、中断处理、外设驱动等。实现抢答器的逻辑控制，包括抢答逻辑等。设计用户界面，通过显示屏和按键实现用户交互。考虑系统的稳定性和实时性，避免出现死锁或响应延迟等问题。

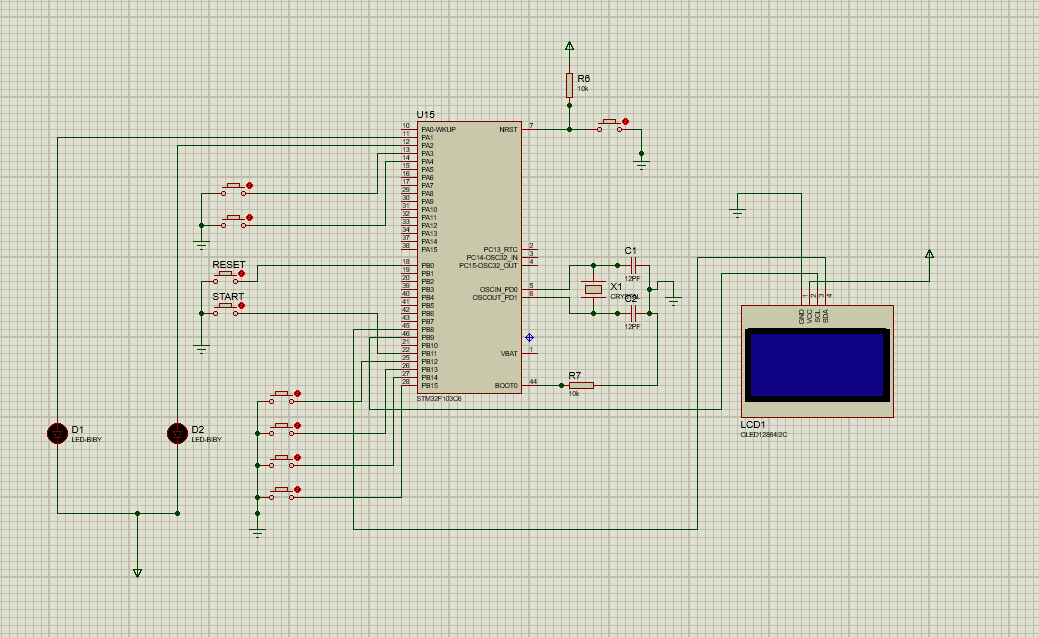
功能实现：实现抢答功能和无人抢答时的提示功能。

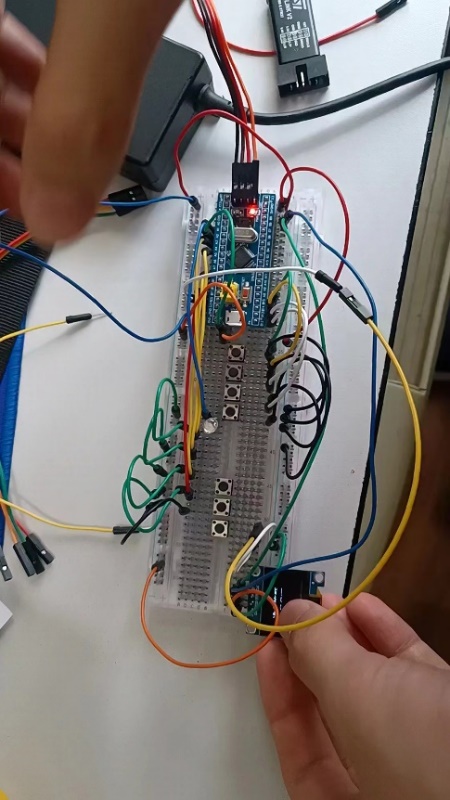
系统验证：进行系统整体功能测试，验证系统是否符合需求。进行性能测试，确保系统响应速度和稳定性满足要求。

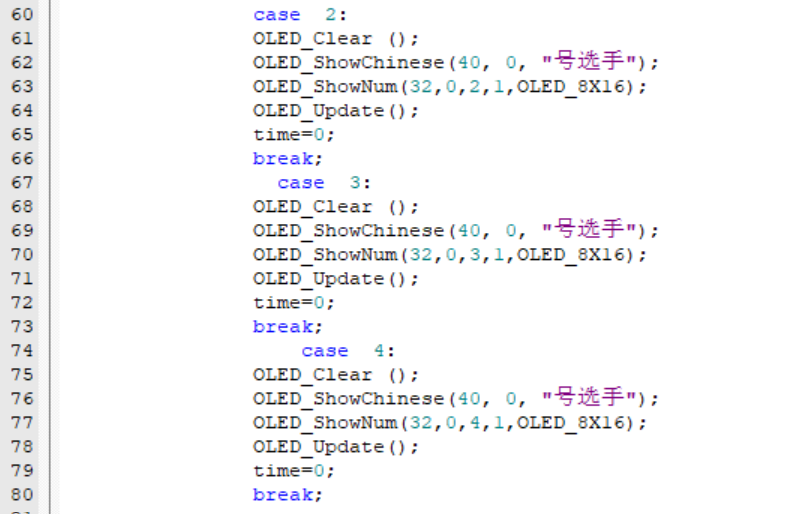
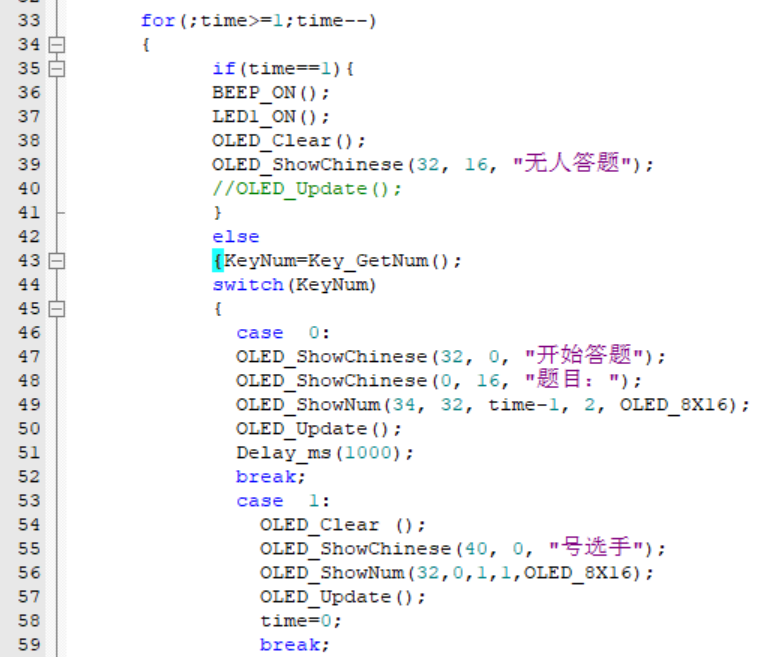
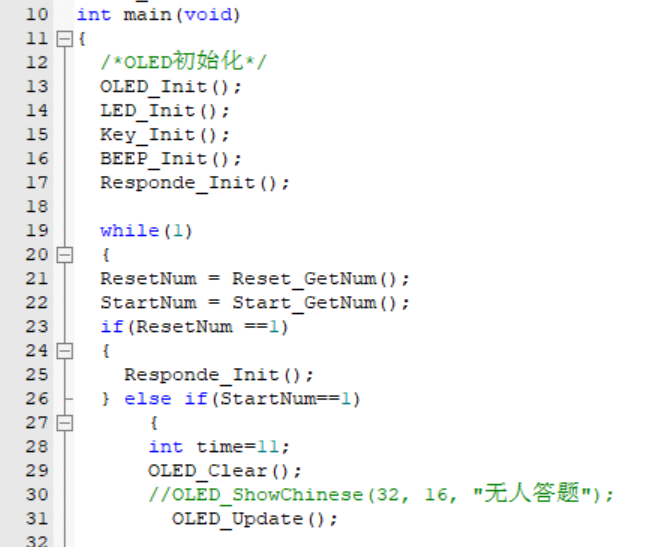
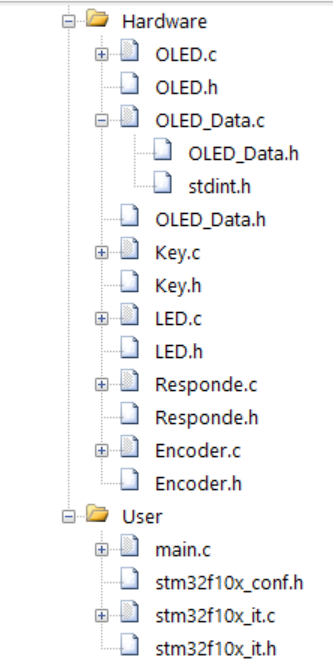
1. 硬件设计

（1）硬件平台  
stm32F103C8T6单片机、四脚按键、显示屏、LED灯、面包板。

1. 外围电路设计





1. 软件设计  
   核心代码  
     
   LED函数

void LED\_Init(void)

{

/\*开启时钟\*/

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOA, ENABLE); //开启GPIOA的时钟

/\*GPIO初始化\*/

GPIO\_InitTypeDef GPIO\_InitStructure;

GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_Out\_PP;

GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_1 | GPIO\_Pin\_2;

GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_50MHz;

GPIO\_Init(GPIOA, &GPIO\_InitStructure); //将PA1和PA2引脚初始化为推挽输出

/\*设置GPIO初始化后的默认电平\*/

GPIO\_SetBits(GPIOA, GPIO\_Pin\_1 | GPIO\_Pin\_2); //设置PA1和PA2引脚为高电平

}

void BEEP\_Init(void)

{

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOA, ENABLE);

/\*GPIO初始化\*/

GPIO\_InitTypeDef GPIO\_InitStructure;

GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_Out\_PP;

GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_8;

GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_50MHz;

GPIO\_Init(GPIOA, &GPIO\_InitStructure); //将PA1和PA2引脚初始化为推挽输出

/\*设置GPIO初始化后的默认电平\*/

GPIO\_SetBits(GPIOA, GPIO\_Pin\_8 ); //设置PA1和PA2引脚为高电平

}

void BEEP\_ON(void)

{

GPIO\_ResetBits(GPIOA, GPIO\_Pin\_8); //设置PA1引脚为低电平

}

void BEEP\_OFF(void)

{

GPIO\_SetBits(GPIOA, GPIO\_Pin\_8); //设置PA1引脚为低电平

}

/\*\*

\* 函 数：LED1开启

\* 参 数：无

\* 返 回 值：无

\*/

void LED1\_ON(void)

{

GPIO\_ResetBits(GPIOA, GPIO\_Pin\_1); //设置PA1引脚为低电平

}

/\*\*

\* 函 数：LED1关闭

\* 参 数：无

\* 返 回 值：无

\*/

void LED1\_OFF(void)

{

GPIO\_SetBits(GPIOA, GPIO\_Pin\_1); //设置PA1引脚为高电平

}

/\*\*

\* 函 数：LED1状态翻转

\* 参 数：无

\* 返 回 值：无

\*/

void LED1\_Turn(void)

{

if (GPIO\_ReadOutputDataBit(GPIOA, GPIO\_Pin\_1) == 0) //获取输出寄存器的状态，如果当前引脚输出低电平

{

GPIO\_SetBits(GPIOA, GPIO\_Pin\_1); //则设置PA1引脚为高电平

}

else //否则，即当前引脚输出高电平

{

GPIO\_ResetBits(GPIOA, GPIO\_Pin\_1); //则设置PA1引脚为低电平

}

}

/\*\*

\* 函 数：LED2开启

\* 参 数：无

\* 返 回 值：无

\*/

void LED2\_ON(void)

{

GPIO\_ResetBits(GPIOA, GPIO\_Pin\_2); //设置PA2引脚为低电平

}

/\*\*

\* 函 数：LED2关闭

\* 参 数：无

\* 返 回 值：无

\*/

void LED2\_OFF(void)

{

GPIO\_SetBits(GPIOA, GPIO\_Pin\_2); //设置PA2引脚为高电平

}

/\*\*

\* 函 数：LED2状态翻转

\* 参 数：无

\* 返 回 值：无

\*/

void LED2\_Turn(void)

{

if (GPIO\_ReadOutputDataBit(GPIOA, GPIO\_Pin\_2) == 0) //获取输出寄存器的状态，如果当前引脚输出低电平

{

GPIO\_SetBits(GPIOA, GPIO\_Pin\_2); //则设置PA2引脚为高电平

}

else //否则，即当前引脚输出高电平

{

GPIO\_ResetBits(GPIOA, GPIO\_Pin\_2); //则设置PA2引脚为低电平

}

}

#ifndef \_\_LED\_H

#define \_\_LED\_H

void LED\_Init(void);

void BEEP\_Init(void);

void LED1\_ON(void);

void LED1\_OFF(void);

void LED1\_Turn(void);

void LED2\_ON(void);

void LED2\_OFF(void);

void LED2\_Turn(void);

void BEEP\_ON(void);

void BEEP\_OFF(void);

#endif

Key函数

#ifndef \_\_LED\_H

#define \_\_LED\_H

void LED\_Init(void);

void BEEP\_Init(void);

void LED1\_ON(void);

void LED1\_OFF(void);

void LED1\_Turn(void);

void LED2\_ON(void);

void LED2\_OFF(void);

void LED2\_Turn(void);

void BEEP\_ON(void);

void BEEP\_OFF(void);

#endif

#ifndef \_\_KEY\_H

#define \_\_KEY\_H

void Key\_Init(void);

uint8\_t Key\_GetNum(void);

uint8\_t Reset\_GetNum(void);

uint8\_t Start\_GetNum(void);

#endif

1. 实物的制作调试  
    电路连接时可能会遇到一些问题，如电路接线错误、元件连接错误或接触不良等。为了解决这些问题，我们需要仔细检查电路连接是否符合设计要求，确保元件正确连接，并确保接触良好。
2. 系统功能测试  
    导入程序后进入初始化状态。按下开始按键，显示屏上出现题目和10秒倒计时，在倒计时结束前如果四个选手的按钮被按下，则显示屏出现第一个按下按钮对应选手的序号，此时可点击复位按钮回到初始化状态。在倒计时结束后一直没有选手按下按钮，则LED灯亮，表示无人抢答，此时可点击复位按钮回到初始化状态。
3. 总结  
    在进行实物的连接前，我仔细进行了仿真实验，并严谨地验证了代码的可靠性，在连接器件和电路板时，我仔细查阅了STM32F103C8T6的数据手册，并根据其引脚功能进行连接。这确保了电路的正常工作并避免了不必要的问题。在软件的配置和编程方面，我使用了STM32CubeMX配置系统时钟、GPIO、定时器等模块，并且可以直接生成初始化的代码。在编程和电路设计过程中，我遇到了一些问题，如定时器的配置还有中断函数的配置等问题，以及编译了四脚OLED显示屏的程序，却错误地使用了六脚显示屏导致看不到实验结果等问题，最终通过学习更多的资料和编程方式解决了。

通过这个实验，我更加了解了STM32单片机的硬件和软件开发，提高了我的电子设计和编程能力，令我更加熟悉了电路连接和零件的使用。

1. 参考文献  
   微机单片机实践指导教程（2023版）

NANO\_STM32F103开发指南-HAL库版本\_V1.0