```
#include "delay.h"
#include "sys.h"
#include "usart.h"
#include "adc.h"
#include "timer.h"//片内资源

#include "key.h"
#include "led.h"
#include "oled.h"
#include "dht11.h"
#include "beep.h"
#include "motor.h"//外设

int compare;//报警阀值，最初会有调零动作，故用变量

void mainInit(){

	char t=10;
	delay_init();	     	//延时函数初始化
	uart_init(115200);
	NVIC_PriorityGroupConfig(NVIC_PriorityGroup_2);//设置 NVIC 中断分组 2:2 位抢占优先级，2 位响应优先级
	LED_Init();		  		//LED 端口初始化
	Adc_Init();
	motorInit();

	TIM3_Int_Init(20,719);   //1Khz 的计数频率，计数到 5000 为 500ms
	beepInit();

	OLED_Init();			//初始化 OLED
	OLED_ShowString(0	, 0,"Initing... DHT11",16);
	OLED_Refresh_Gram();//更新显示到 OLED
	while(DHT11_Init())		//DHT11 初始化
	{
		delay_ms(200);
	}

	OLED_Clear();
	while(t-->0)			//给时间，让气体传感器加热充分
	{
		OLED_ShowString(0	, 0,"Initing... MQ7",16);
		OLED_ShowNum(0, 16,t,2,16);
		OLED_Refresh_Gram();//更新显示到 OLED
		delay_ms(1000);
```

```
    }
    compare=300+Get_Adc_Average(ADC_Channel_9,10);  //以当前气体传感器值+100 为报警阀值，进行调零
    OLED_Clear();
    OLED_ShowString(0, 0,"TEM:     HUM:     ",16);
    OLED_ShowString(0, 0,"TEM:     HUM:     ",16);
    OLED_ShowString(0,16,"CO:               ",16);
    OLED_Refresh_Gram(); //更新显示到 OLED

    MOTOR = 0;
}

int main(void)
{

    /*adc*/
    u16 adcx;
    //float temp;
    /*温湿度*/
    u8 temperature = 0;
    u8 humidity = 0;
    /*系统初始化*/
    mainInit();



    /*主循环，200ms 更新一次*/
    while(1){

      /*获取 CO 浓度*/
      adcx=Get_Adc_Average(ADC_Channel_9,10);

      /*获取温湿度*/
      if(!DHT11_Read_Data(&temperature,&humidity)){
        OLED_ShowNum(2*16,0,temperature,2,16);
        OLED_ShowNum(6*16,0,humidity,2,16);
        OLED_ShowNum(2*16,16,adcx,4,16);
        OLED_Refresh_Gram();     //更新显示到 OLED
      }

      /*CO 浓度大于预警值*/
      if(adcx > compare){
        redLightOn();
        beepOpen();
```

```
                MOTOR = 0;
            }else{
                beepClose();
                MOTOR = 1;
            }

            delay_ms(200);

        }
}


void beepInit(void){

  GPIO_InitTypeDef   GPIO_InitStructure;
  RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOA, ENABLE);  //使能 PA 端口时钟

  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_7;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out_PP;          //推挽输出
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;         //IO 口速度为 50MHz
  GPIO_Init(GPIOA, &GPIO_InitStructure);                //根据设定参数初始化GPIOC.13
  GPIO_SetBits(GPIOA,GPIO_Pin_7);             //PA.7 输出低

}

void beepClose(void){
    TIM_Cmd(TIM3, DISABLE);
    GPIO_SetBits(GPIOA,GPIO_Pin_7);              //PA.7 输出低
}

void beepOpen(void){
    TIM_Cmd(TIM3, ENABLE);
}

void TIM3_IRQHandler(void)    //TIM3 中断
{
    if (TIM_GetITStatus(TIM3, TIM_IT_Update) != RESET)   //检查 TIM3 更新中断发生与否
        {
        TIM_ClearITPendingBit(TIM3, TIM_IT_Update);   //清除 TIMx 更新中断标志
        BEEP = !BEEP;
        }
}
```