```
#include "GPIO.h"
#include "LCD1602.h"
#include "delay.h"
#include "sys.h"
#include "timer.h"
#include "HC_SR04.h"
#include "DS18B20.h"
#include <stdio.h>
#include <math.h>

float S,SPEED = 0;
u8 song_begin = 0;
//STM32 超声波测距
int   SAFET_Distance   = 300;              //安全距离

float   temper = 0;
float   Distance = 0;

u8 state_machine = 1;    //根据按键变化使能或失能测量

u8 mode = 0;

int main(void)
{
      char displaytemp[16];           //开启字符串空间

      u8 key = 0;
      delay_init();          //延时函数初始化
      HC_SR04_IO_Init();   //超声波模块 GPIO 初始化
      Key_Init();      //按键初始化
      LCD1602_ENABLE();    //使能屏幕
      delay_ms(500);           //上电瞬间加入一定延时在初始化
      LCD_Init();              //屏幕初始化
      TIM2_Init(7199,0);      //以 10KHz 计数
      sprintf(displaytemp,"Alarm Val: %dmm",SAFET_Distance);
      LCD_Write_String(0,1,displaytemp);
      //while(1);
      while(1)
      {
            key = Key_Scan(0);

                  if(key == 2)
                  {
                        SAFET_Distance = SAFET_Distance+10;
```

```
                if(SAFET_Distance >99999)
                {
                    SAFET_Distance = 99999;
                }
                sprintf(displaytemp,"Alarm Val: %dmm",SAFET_Distance);
        LCD_Write_String(0,1,displaytemp);
            }

            if(key == 1)
            {
                SAFET_Distance = SAFET_Distance-10;
                if(SAFET_Distance<10)
                {
                    SAFET_Distance = 10;
                }
                sprintf(displaytemp,"Alarm Val: %dmm",SAFET_Distance);
        LCD_Write_String(0,1,displaytemp);
            }

                Distance = (Get_SR04_Distance() * 331) * 1.0/1000;
//Get_SR04_Distance()返回单程声波传输时间 us

                if(Distance>=4500)                          //超过传感器测距极限，该
值不能使用
                {
                    Distance = 4500.00;
                    sprintf(displaytemp,"Distance: %4.0fmm ",Distance);
                }

                 if (Distance>=1000)
                    sprintf(displaytemp,"Distance: %4.0fmm ",Distance);    //少一个
空格。防止将 mm 挤到后面
                else
                    sprintf(displaytemp,"Distance: %4.0fmm ",Distance);
                LCD_Write_String(0,0,displaytemp);                  //显示距离

                if(Distance <= SAFET_Distance)
                {
                   BEEP = !BEEP;
                    delay_ms(Distance);
                }
                else
                {
                    BEEP = 0;
```

```
                }
                delay_ms(150);                //插入延时，防止屏幕刷新太快
        }

}

#include"delay.h"

#include "GPIO.h"


//sbit date=P0^5;//数据
//sbit reset=P0^6;//复位
//sbit busy=P0^4;//查忙

void playsound_init()
{
    GPIO_InitTypeDef    GPIO_InitStructure;

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

    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_11;                    //端口配置
    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);
    GPIO_ResetBits(GPIOA,GPIO_Pin_11);                        //PB.11  输出低

    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_12;                    //端口配置
    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);
    GPIO_ResetBits(GPIOA,GPIO_Pin_12);                        //PB.12  输出低

    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_15;                    //端口配置
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IPU;        //推挽输出
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;        //IO 口速度为 50MHz
    GPIO_Init(GPIOA, &GPIO_InitStructure);
}
void sound_delayms(unsigned int p)//毫秒延时程序
{
    delay_ms(p);
}

uint8_t    PlaySoundCheckBusy(void)//语音播报函数，举例：若要播报第五个地址的内容，只
```

需调用函数 sj(5)

```
{
    uint8_t dat;

    dat=GPIO_ReadInputDataBit(GPIOA,GPIO_Pin_15);

    return dat;
}

void PlaySoundTick(int a)//语音播报函数，举例：若要播报第五个地址的内容，只需调用函数 sj(5)
{
    while(PlaySoundCheckBusy() == 0);

    if (a == 0)
        a = 10;
    //reset=1;
    GPIO_SetBits(GPIOA,GPIO_Pin_11);
    sound_delayms(1);
    //reset=0;          //语音芯片复位
    GPIO_ResetBits(GPIOA,GPIO_Pin_11);
    sound_delayms(1);
    while(a)//播报第 a 个地址的内容
    {
        //date=1;

        GPIO_SetBits(GPIOA,GPIO_Pin_12);
        sound_delayms(1);
        //date=0;
        GPIO_ResetBits(GPIOA,GPIO_Pin_12);
        sound_delayms(1);
        a--;
    }

}
```