实验八 定时器计时实验

```c
#include "stm32f10x.h" // 根据您的 STM32 型号和库版本调整
#include <stdio.h>     // 用于 sprintf 函数将数值转换为字符串

// USART_SendString 函数 (来自实验指导书)
void USART_SendString(USART_TypeDef* USARTx, char *DataString) {
    unsigned char i = 0;
    while (DataString[i] != '\0') {
        USART_SendData(USARTx, DataString[i]);
        while (USART_GetFlagStatus(USARTx, USART_FLAG_TXE) == RESET);
        i++;
    }
}

// 需要测量其执行时间的延时函数
void Delay_ms_Target(volatile unsigned int nms) {
    volatile unsigned int i, j;
    for (i = 0; i < nms; i++) {
        for (j = 0; j < 12000; j++); // 此值需要根据实际时钟和目标精度调整
    }
}

// KEY0 初始化 (PE4, 上拉输入, 低电平有效)
void KEY0_Init(void) {
    GPIO_InitTypeDef GPIO_InitStructure;

    // 使能 GPIOE 时钟 (KEY0 连接在 PE4)
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOE, ENABLE);

    // 配置 PE4 为上拉输入模式
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_4;        // KEY0 在 PE4
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IPU;    // 上拉输入
    GPIO_Init(GPIOE, &GPIO_InitStructure);           // 初始化 GPIOE
}

// USART1 初始化 (PA9 TX, PA10 RX) - 保持不变
void USART1_Init_For_Experiment(unsigned int bound) {
    GPIO_InitTypeDef GPIO_InitStructure;
    USART_InitTypeDef USART_InitStructure;

    // 使能 GPIOA 和 USART1 时钟 (GPIOA 用于 PA9/PA10)
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOA, ENABLE);
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_USART1, ENABLE);

    // 配置 PA9 (USART1_TX)
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_9;
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF_PP;
    GPIO_Init(GPIOA, &GPIO_InitStructure);

    // 配置 PA10 (USART1_RX)
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_10;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN_FLOATING;
    GPIO_Init(GPIOA, &GPIO_InitStructure);

    // USART1 参数配置
    USART_DeInit(USART1);
    USART_InitStructure.USART_BaudRate = bound;
    USART_InitStructure.USART_WordLength = USART_WordLength_8b;
    USART_InitStructure.USART_StopBits = USART_StopBits_1;
    USART_InitStructure.USART_Parity = USART_Parity_No;
    USART_InitStructure.USART_HardwareFlowControl =
USART_HardwareFlowControl_None;
    USART_InitStructure.USART_Mode = USART_Mode_Rx | USART_Mode_Tx;
    USART_Init(USART1, &USART_InitStructure);

    USART_Cmd(USART1, ENABLE);
}

// TIM2 初始化用于计时 (配置为10微秒每计数一次) - 保持不变
void TIM2_Base_Init_For_Timing(void) {
    TIM_TimeBaseInitTypeDef TIM_TimeBaseStructure;
    unsigned short prescaler_value;
    unsigned short period_value;

    RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM2, ENABLE);

    prescaler_value = 719; // (72MHz / (719+1)) = 0.1MHz -> 10μs tick
    period_value = 0xFFFF;

    TIM_TimeBaseStructure.TIM_Period = period_value;
    TIM_TimeBaseStructure.TIM_Prescaler = prescaler_value;
    TIM_TimeBaseStructure.TIM_ClockDivision = TIM_CKD_DIV1;
    TIM_TimeBaseStructure.TIM_CounterMode = TIM_CounterMode_Up;
    TIM_TimeBaseInit(TIM2, &TIM_TimeBaseStructure);
```

```
}

int main(void) {
    unsigned short timer_raw_count;
    unsigned long measured_time_us;
    char message_output_buffer[60];

    KEY0_Init();                         // 初始化 KEY0 (PE4)
    USART1_Init_For_Experiment(115200);     // 初始化 USART1
    TIM2_Base_Init_For_Timing();          // 初始化 TIM2

    USART_SendString(USART1, (char*)"Experiment 8: Program Run Time
Measurement\r\n");
    USART_SendString(USART1, (char*)"Press KEY0 (PE4) to measure
Delay_ms_Target(200) time.\r\n");

    while (1) {
        // 检测 KEY0 是否按下 (PE4 低电平表示按下)
        if (GPIO_ReadInputDataBit(GPIOE, GPIO_Pin_4) == Bit_RESET) { //
检查 PE4
            Delay_ms_Target(25); // 消抖

            if (GPIO_ReadInputDataBit(GPIOE, GPIO_Pin_4) == Bit_RESET)
{ // 再次检查 PE4
                USART_SendString(USART1, (char*)"KEY0 pressed. Starting
measurement...\r\n");

                TIM_SetCounter(TIM2, 0);
                TIM_Cmd(TIM2, ENABLE);

                Delay_ms_Target(200);

                TIM_Cmd(TIM2, DISABLE);
                timer_raw_count = TIM_GetCounter(TIM2);

                measured_time_us = (unsigned long)timer_raw_count * 10;

                sprintf(message_output_buffer, "Delay_ms_Target(200)
execution time: %lu us\r\n", measured_time_us);
                USART_SendString(USART1, message_output_buffer);

                while (GPIO_ReadInputDataBit(GPIOE, GPIO_Pin_4) ==
Bit_RESET); // 等待 PE4 释放
```

```
                USART_SendString(USART1, (char*)"KEY0 released. Ready for next measurement.\r\n\r\n");
            }
        }
    }
}
```