```c
#include "interrupt.h"

#define PA15_FREQ 1000000
#define PB4_FREQ 1000000

struct keys key[4] = {0};

uint PA15_freq,PA15_duty;
uint PA15_rise,PA15_fall;

uint PB4_freq,PB4_duty;
uint PB4_rise,PB4_fall;

char rx_arry[50];
char rx_data;
char rx_pointer;

void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart)
{
    if(huart->Instance == USART1)
    {
        rx_arry[rx_pointer++] = rx_data;
        HAL_UART_Receive_IT(huart, (uint8_t *)&rx_data, 1);
    }
}
void HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef *htim)
{
    if(htim->Instance == TIM8)
    {
        if(htim->Channel == HAL_TIM_ACTIVE_CHANNEL_1)
        {
            PA15_rise = __HAL_TIM_GetCounter(htim);
            __HAL_TIM_SetCounter(htim,0);
            PA15_freq = PA15_FREQ/PA15_rise;
            PA15_duty = PA15_fall*100/PA15_rise;
        }
        else if(htim->Channel == HAL_TIM_ACTIVE_CHANNEL_2)
        {
            PA15_fall = __HAL_TIM_GetCounter(htim);
        }
    }

    if(htim->Instance == TIM3)
    {
        if(htim->Channel == HAL_TIM_ACTIVE_CHANNEL_1)
        {
            PB4_rise = __HAL_TIM_GetCounter(htim);
            __HAL_TIM_SetCounter(htim,0);
            PB4_freq = PB4_FREQ/PB4_rise;
            PB4_duty = PB4_fall*100/PB4_rise;
        }
        else if(htim->Channel == HAL_TIM_ACTIVE_CHANNEL_2)
        {
            PB4_fall = __HAL_TIM_GetCounter(htim);
        }
    }
}
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{
    if(htim->Instance == TIM6)
    {
        key[0].value = HAL_GPIO_ReadPin(GPIOB,GPIO_PIN_0);
        key[1].value = HAL_GPIO_ReadPin(GPIOB,GPIO_PIN_1);
        key[2].value = HAL_GPIO_ReadPin(GPIOB,GPIO_PIN_2);
        key[3].value = HAL_GPIO_ReadPin(GPIOA,GPIO_PIN_0);

        for(int i=0;i<4;i++)
        {
            switch(key[i].state)
            {
                case 0:
```

```
72                      if(key[i].value == 0) key[i].state = 1;
73                      break;
74                  case 1:
75                      if(key[i].value == 0)
76                      {
77                          key[i].state = 2;
78                          key[i].click_time = 0;
79                      }
80                      else key[i].state = 0;
81                      break;
82                  case 2:
83                      if(key[i].value == 0)
84                          key[i].click_time++;
85                      else
86                      {
87                          if(key[i].click_time >80)
88                              key[i].long_flag = 1;
89                          else key[i].short flag = 1;
90
91                              key[i].state = 0;
92                      }
93
94                      break;
95              }
96          }
97      }
98  }
```