```c
#include "interrupt.h"
#define PA7_FREQ 1000000
uint pa7_freq,pa7_duty;
uchar pa7_flag;
uint pa7_rise,pa7_fall;

struct keys key[4] = {0};

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:
                    if(key[i].value == 0) key[i].state = 1;
                    break;
                case 1:
                    if(key[i].value == 0)
                    {
                        key[i].state = 2;
                        key[i].click_time = 0;
                    }
                    break;
                case 2:
                    if(key[i].value == 0)
                        key[i].click_time++;
                    else
                    {
                        if(key[i].click_time > 90)
                            key[i].long_flag = 1;
                        else key[i].short_flag = 1;
                        key[i].state = 0;
                    }
                    break;
//                  case 2:
//                  if(key[i].value == 0)
//                      key[i].click_time++;
//                  else if(key[i]. click_time >70)
//                  {
//                      key[i].long_flag = 1;
//                      key[i].state = 0;
//                  }
//                  else
//                  {
//                      switch(key[i].double_state)
//                      {
//                          case 0:
//                              key[i].double_state = 1;
//                              break;
//                          case 1:
//                              key[i].double_state = 0;
//                              key[i].double_flag = 1;
//                              break;
//                      }
//                      key[i].state = 0;
//                      key[i].double_time = 0;
//                  }
//                  break;
            }
//          if(key[i].double_state == 1)
//          {
//              key[i].double_time++;
//              if(key[i].double_time >30)
```

```
72  //                {
73  //                    key[i].short_flag = 1;
74  //                    key[i].double_state = 0;
75  //                }
76  //            }
77          }
78      }
79  }
80  void HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef *htim)
81  {
82      if(htim->Instance == TIM17)
83      {
84          if(htim->Channel == HAL_TIM_ACTIVE_CHANNEL_1)
85          {
86              if(pa7_flag == 0)
87              {
88                  pa7_rise = __HAL_TIM_GetCounter(htim);
89                  pa7_freq = PA7_FREQ/pa7_rise;
90                  pa7_duty = pa7_fall*100/pa7_rise;
91                  __HAL_TIM_SetCounter(htim,0);
92                  __HAL_TIM_SET_CAPTUREPOLARITY(htim,TIM_CHANNEL_1,TIM_INPUTCHANNELPOLARITY_FALLING);
93              }
94              else
95              {
96                  pa7_fall = __HAL_TIM_GetCounter(htim);
97                  __HAL_TIM_SET_CAPTUREPOLARITY(htim,TIM_CHANNEL_1,TIM_INPUTCHANNELPOLARITY_RISING);
98              }
99              pa7_flag = !pa7_flag;
100         }
101
102     }
103 }
104
105
```