/\*

 \* task1.c

 \*

 \*  Created on: Sep 20, 2021

 \*      Author: wangy62

 \*/

#include "init.h"

void Init\_GPIO();

void Init\_Timer();

GPIO\_InitTypeDef GPIO\_InitStruct;

int main() {

    Sys\_Init();

    HAL\_Init();

    Init\_GPIO();

    // Clear screan

    printf("\033[2J\033[;H");

    printf("start\r\n");

    // Infinite loop, wait for interrupt

    while(1);

}

void Init\_GPIO(){

    //Enable RCC clock

    RCC->AHB1ENR |= RCC\_AHB1ENR\_GPIOJEN;

    RCC->AHB1ENR |= RCC\_AHB1ENR\_GPIOCEN;

    RCC->APB2ENR |= RCC\_APB2ENR\_SYSCFGEN;

    asm ("nop");

    asm ("nop");

    //Enable GPIO J0 Pin D4 for input

    GPIOJ -> MODER &= 0xFFFFFFFCU;  //Mode input

    GPIOJ -> PUPDR |= 0x00000002U;  //Pull down

    GPIOJ -> PUPDR &= 0xFFFFFFFEU;  //Pull down

    // GPIO Interrupt

    // Enable GPIO J for interrupt

    SYSCFG->EXTICR[0] |= 0x00000009;

    SYSCFG->EXTICR[0] &= 0xFFFFFFF9;

    // Set interrupt enable for EXTI0.

    NVIC->ISER[0] = (uint32\_t) 1 << (6);

    // Enable Rinsing and Falling edge for interrupt

    EXTI->IMR |= 0x00000001U;

    EXTI->RTSR |= 0x00000001U;

    EXTI->FTSR |= 0x00000001U;

    //GPIO C8 Pin D5 for input

    \_\_HAL\_RCC\_GPIOC\_CLK\_ENABLE();

    GPIO\_InitStruct.Pin = GPIO\_PIN\_8;

    GPIO\_InitStruct.Mode = GPIO\_MODE\_IT\_RISING\_FALLING; //Rising and falling edge

    GPIO\_InitStruct.Pull = GPIO\_PULLDOWN;   // Pulldown resister

    HAL\_GPIO\_Init(GPIOC, &GPIO\_InitStruct);

    HAL\_NVIC\_EnableIRQ(EXTI9\_5\_IRQn);   //

}

// IRQHandler for EXTI0 GPIO J0 Pin D4

void EXTI0\_IRQHandler() {

    // Clear Interrupt Bit by setting it to 1.

    EXTI->PR |= 0x00000001U;

    asm ("nop");

    asm ("nop");

    // Other code here:

    printf("EXTI0 interrupt occured\r\n");

    for (int i = 0; i<15;i++){}

}

// IRQHandler for EXTI9\_5 GPIO C8 Pin D5

void EXTI9\_5\_IRQHandler() {

    HAL\_GPIO\_EXTI\_IRQHandler(GPIO\_PIN\_8);

}

void HAL\_GPIO\_EXTI\_Callback(uint16\_t GPIO\_Pin){

    asm ("nop");

    asm ("nop");

    // Other code here:

    printf("EXTI9\_5 interrupt occured\r\n");

    for (int i = 0; i<15;i++){}

}

TASK3

/\*

 \* task3.c

 \*

 \*  Created on: Sep 23, 2021

 \*      Author: wangy62

 \*/

#include "init.h"

//void blinkScreen();

void Init\_GPIO();

void Init\_Timer();

TIM\_HandleTypeDef htim6;

volatile uint16\_t timeUpdated = 0;  // Overflow counter

uint32\_t Time\_ms = 0;

int main() {

    Sys\_Init();

    Init\_Timer();

    printf("\033[2J\033[;H");

    printf("start\r\n");

    // Infinite loop

    while(1);

}

void Init\_Timer() {

    // initialize the pointer to set timer 6

    htim6.Instance = TIM6;

    htim6.Init.Prescaler = 5399;

    htim6.Init.Period = 1999;

    \_\_HAL\_RCC\_TIM6\_CLK\_ENABLE();

    // Enable NVIC interrupt for timer 6 TIM6\_DAC

    HAL\_NVIC\_EnableIRQ(TIM6\_DAC\_IRQn);

    HAL\_TIM\_Base\_Init(&htim6); //Configure the timer

    HAL\_TIM\_Base\_Start\_IT(&htim6);  // Start the timer

}

void TIM6\_DAC\_IRQHandler() {

// Pass the control to HAL, which processes the IRQ

   HAL\_TIM\_IRQHandler(&htim6);

}

void HAL\_TIM\_PeriodElapsedCallback(TIM\_HandleTypeDef \*htim) {

    // Interrupt triggers only with TIM6

    if(htim->Instance == TIM6){

        // Overflow counter increase

        timeUpdated ++;

        Time\_ms = timeUpdated\*100;  // Calculate time is ms

        printf("\033[2;0H");

        printf("overflow occured, time %dms\r\n", Time\_ms);

   }

}

Task4

/\*

 \* task4.c

 \*

 \*  Created on: Sep 23, 2021

 \*      Author: wangy62

 \*/

#include "init.h"

#include "stdlib.h"

//void blinkScreen();

void Init\_GPIO();

void Init\_Timer();

void Screen\_Init();

void blinkScreen();

GPIO\_InitTypeDef GPIO\_InitStruct;

// Interrupt flag

uint8\_t react\_flag = 0;

char c;

// Overflow counter, timer

uint32\_t Global\_Time = 0;

uint32\_t reactTime = 0;

uint32\_t player\_score[5];

float averageScore = 0;

int temp = 0;

int random\_time[5];

float totalTime = 0;

int main() {

    Sys\_Init();

    HAL\_Init();

    Init\_Timer();

    Init\_GPIO();

    printf("\033[2J\033[;H");

    printf("start\r\n");

    // Get random seed

    srand(time(NULL));

    // Generate the random value for setting the blinking start time

    for (int i=0;i<5;i++){

        random\_time[i] = 50 + 10 \* (rand()%40); // 0.5 sec to 4.5 sec to start blinking

        printf("Random number(50,450): %d\r\n", random\_time[i]);

    }

    HAL\_Delay(1000);

    // Push a key to start

    printf("push a key to start\r\n");

    c = getchar();

    // Game loop

    while(1){

        // Total 5 trails

        for (int k=0;k<5;k++){

            // Clear the screen

            Screen\_Init();

            // Calculate delay time and delay

            temp = 10\*random\_time[k];

            HAL\_Delay(temp);

            // Clear the timer

            Global\_Time =0;

            // Blink the screen until interrupt

            // react\_flag set to 1 only when there is an interrupt

            while(react\_flag == 0){

                // Blink the screen

                printf("\033c\033[36m\033[2J");

                printf("Blink!\r\n");

                HAL\_Delay(50);

                blinkScreen();

                HAL\_Delay(50);

            }

            // record the react time in 10ms

            player\_score[k] = reactTime;

        }

        printf("\033[2J\033[;H");

        // Print each score

        for (int j=0; j<5; j++){

            printf("the %d trail reatction time is %d0ms \r\n",j+1, player\_score[j]);

            totalTime += player\_score[j];

        }

        // Print total score

        averageScore = totalTime / 500;

        printf("Player average react time is %.2fs\r\n", averageScore);

        while(1);

    }

}

void Init\_Timer() {

    // Enable the TIM6 interrupt.

    // Enable NVIC interrupt for TIM6\_DAC position 54

    NVIC->ISER[1] = (uint32\_t) 1 << (22);

    // Enable TIM6 clock

    RCC->APB1ENR |= RCC\_APB1ENR\_TIM6EN;

    asm ( "nop" );

    asm ( "nop" );

    TIM6->PSC = 0x0437; // PSC = 1079

    // Set the Auto-reload Value for 100Hz overflow

    TIM6->ARR = 0x03E9; // ARR = 999

    // Generate update events to auto reload.

    TIM6->EGR |= 0x0001;

    // Enable Update Interrupts.

    TIM6->DIER |= 0x0001;

    // Start the timer.

    TIM6->CR1 |= 0x0001;

}

void Init\_GPIO(){

    // Enable RCC clock

    RCC->AHB1ENR |= RCC\_AHB1ENR\_GPIOAEN;

    RCC->AHB1ENR |= RCC\_AHB1ENR\_GPIOCEN;

    RCC->APB2ENR |= RCC\_APB2ENR\_SYSCFGEN;

    asm ("nop");

    asm ("nop");

    //Enable GPIO PA0 USER pushbutton for input

    GPIOA -> MODER &= 0xFFFFFFFCU;

    GPIOA -> PUPDR |= 0x00000002U;

    GPIOA -> PUPDR &= 0xFFFFFFFEU;

    // Enable GPIO A for interrupt

    SYSCFG->EXTICR[0] |= 0x00000000;

    SYSCFG->EXTICR[0] &= 0xFFFFFFFF;

    // Set interrupt enable for EXTI0.

    NVIC->ISER[0] = (uint32\_t) 1 << (6);

    // Interrupt on triggers for the rising edge

    EXTI->IMR |= 0x00000001U;

    EXTI->RTSR |= 0x00000001U;

    \_\_HAL\_RCC\_GPIOC\_CLK\_ENABLE();

    //GPIO C8 Pin D5 for input

    GPIO\_InitStruct.Pin = GPIO\_PIN\_8;

    GPIO\_InitStruct.Mode = GPIO\_MODE\_IT\_RISING; // Rising trigger

    GPIO\_InitStruct.Pull = GPIO\_PULLDOWN;   // Internal resistor pulldown

    HAL\_GPIO\_Init(GPIOC, &GPIO\_InitStruct); // Enable GPIO C8 Pin D5

    HAL\_NVIC\_EnableIRQ(EXTI9\_5\_IRQn);

}

// User pushbutton interrupt to record the time

void EXTI0\_IRQHandler() {

    // Clear Interrupt Bit by setting it to 1.

    EXTI->PR |= 0x00000001U;

    asm ("nop");

    asm ("nop");

    // Record the time when interrupt trigger

    reactTime = Global\_Time;

    // set the flag to quit the loop for blinking

    react\_flag = 1;

    for (int i = 0; i<10;i++){}

}

void TIM6\_DAC\_IRQHandler() {

    // Clear Interrupt Bit

    TIM6->SR &= 0xFFFE;

    // Overflow counter increases

    Global\_Time++;

    asm ( "nop" );

    asm ( "nop" );

}

void EXTI9\_5\_IRQHandler() {

    HAL\_GPIO\_EXTI\_IRQHandler(GPIO\_PIN\_8);

}

void HAL\_GPIO\_EXTI\_Callback(uint16\_t GPIO\_Pin){

    // ISR code here.

    asm ("nop");

    asm ("nop");

        // Other code here:

    for (int i = 0; i<5;i++){}

    // Restart the program work same as the reset pushbutton

    NVIC\_SystemReset();

}

void Screen\_Init(){

    // Clear the screen and begin game

    printf("\033[2J\033[;H");

    printf("Test starts in 3 seconds\r\n");

    printf("3\r\n");

    HAL\_Delay(1000);

    printf("2\r\n");

    HAL\_Delay(1000);

    printf("1\r\n");

    HAL\_Delay(1000);

    printf("0 Counting Begin!!\r\n");

    printf("0 Counting Begin!!\r\n");

    printf("0 Counting Begin!!\r\n");

    // Clear the timer and interrupt flag

    Global\_Time = 0;

    react\_flag = 0;

}

// blink function in the lab template

void blinkScreen(){

    printf("\033[30;47m");

    // Clear and redraw display (flash it & sound the bell).

    printf("\a\033[s\033[2J\033[u");

    fflush(stdout);

    HAL\_Delay(100);

    printf("\033[37;40m");

    // Clear and redraw display (flash it).

    printf("\033[s\033[2J\033[u");

    fflush(stdout);

}