Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Программное обеспечение встроенных систем (ПОВС)

ОТЧЕТ

по лабораторной работе № 5

Тема работы: игровое приложение с использованием STM32

Выполнили

студенты: гр. 151003 Барановский Р.А.,

Матошко И.В.

Проверил: Деменковец Д.В.

Минск 2024

**lib.h**

#ifndef LIB\_DECLARATION

#define LIB\_DECLARATION

#include "stm32f1xx\_hal.h"

#include "stm32f1xx\_hal\_adc.h"

#include "math.h"

#include "stdio.h"

#include "string.h"

#define CORRECTION\_TIME 500

extern volatile int previousJumpPressedTime;

extern volatile int previousShootPressedTime;

extern volatile int previousRunPressedTime;

extern volatile uint16\_t adc;

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

void HAL\_ADC\_ConvCpltCallback(ADC\_HandleTypeDef\* hadc);

#endif // LIB\_DECLARATION

**lib.c**

#include "lib.h"

#include "main.h"

#include "display.h"

volatile int previousJumpPressedTime = 0;

volatile int previousShootPressedTime = 0;

volatile int previousRunPressedRime = 0;

volatile uint16\_t adc = 0;

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

{

if(GPIO\_Pin == GPIO\_PIN\_5 && (HAL\_GetTick() - previousJumpPressedTime) > CORRECTION\_TIME)

{

BaseType\_t xHigherPriorityTaskWoken = pdFALSE;

xTaskNotifyFromISR(butPressedHandle, GPIO\_Pin, eSetValueWithOverwrite, &xHigherPriorityTaskWoken);

portYIELD\_FROM\_ISR(xHigherPriorityTaskWoken);

previousJumpPressedTime = HAL\_GetTick();

}

else if (GPIO\_Pin == GPIO\_PIN\_4 && (HAL\_GetTick() - previousShootPressedTime) > CORRECTION\_TIME)

{

BaseType\_t xHigherPriorityTaskWoken = pdFALSE;

xTaskNotifyFromISR(butPressedHandle, GPIO\_Pin, eSetValueWithOverwrite, &xHigherPriorityTaskWoken);

portYIELD\_FROM\_ISR(xHigherPriorityTaskWoken);

previousShootPressedTime = HAL\_GetTick();

} else if (GPIO\_Pin == GPIO\_PIN\_3 && (HAL\_GetTick() - previousRunPressedRime) > CORRECTION\_TIME)

{

BaseType\_t xHigherPriorityTaskWoken = pdFALSE;

xTaskNotifyFromISR(butPressedHandle, GPIO\_Pin, eSetValueWithOverwrite, &xHigherPriorityTaskWoken);

portYIELD\_FROM\_ISR(xHigherPriorityTaskWoken);

previousRunPressedRime = HAL\_GetTick();

}

}

void HAL\_ADC\_ConvCpltCallback(ADC\_HandleTypeDef\* hadc)

{

if(hadc->Instance == ADC1)

{

adc = HAL\_ADC\_GetValue(&hadc1);

if (adc >= 3500)

{

const char\* msg = "moveRight\n";

HAL\_UART\_Transmit\_DMA(&huart2, (uint8\_t\*)msg, strlen(msg));

}

else if (adc <=1500)

{

const char\* msg = "moveLeft\n";

HAL\_UART\_Transmit\_DMA(&huart2, (uint8\_t\*)msg, strlen(msg));

}

else if (adc > 1500 && adc < 3500)

{

const char\* msg = "stand\n";

HAL\_UART\_Transmit\_DMA(&huart2, (uint8\_t\*)msg, strlen(msg));

}

adc = 0;

}

}

**main.c**

/…

/\* creation of adcStartConv \*/

adcStartConvHandle = osThreadNew(adcStartConvHandler, NULL, &adcStartConv\_attributes);

/\* creation of butPressed \*/

butPressedHandle = osThreadNew(butPressedHandler, NULL, &butPressed\_attributes);

//…

void adcStartConvHandler(void \*argument)

{

/\* USER CODE BEGIN adcStartConvHandler \*/

/\* Infinite loop \*/

for(;;)

{

HAL\_ADC\_Start\_IT(&hadc1);

osDelay(100);

}

/\* USER CODE END adcStartConvHandler \*/

}

void butPressedHandler(void \*argument)

{

/\* USER CODE BEGIN butPressedHandler \*/

uint32\_t pinValue;

char\* msg;

/\* Infinite loop \*/

for(;;)

{

ulTaskNotifyTake(pdTRUE, portMAX\_DELAY);

xTaskNotifyWait(0, 0, &pinValue, portMAX\_DELAY);

if (pinValue == GPIO\_PIN\_5)

{

msg = "jump\n";

}

else if (pinValue == GPIO\_PIN\_4)

{

msg = "shoot\n";

}

else if (pinValue == GPIO\_PIN\_3)

{

msg = "switchRun\n";

}

HAL\_UART\_Transmit\_DMA(&huart2, (uint8\_t\*)msg, strlen(msg));

}

/\* USER CODE END butPressedHandler \*/

}