**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP. HCM**



**VI ĐIỀU KHIỂN**

**BÀI BÁO CÁO 10**

**NHÓM 8**

Giảng viên : **PHẠM QUANG TRÍ**

Sinh viên :

* Trần Công Hòa 20017691
* Đỗ Tuấn Duy 20061261
* Nguyễn Quốc Dương 20123251

**TP.HCM – 2022**

**Bài tập mức độ 2:**

1. **Sơ đồ nguyên lí kết nối phần cứng tối thiểu dùng STM32:**

Diagram, schematic

Description automatically generated

1. **Cách thiết lập chức năng trong phần mềm CubeMX:**

**Bước 1**: Khởi động phần mềm CubeMX, tạo New Project và chọn mã số STM32F103C8T6.

Graphical user interface, text, application

Description automatically generated

**Bước 2**: Chọn mạch nạp:

Graphical user interface, text

Description automatically generated

**Bước 3**: Chọn bộ giao động:

**Graphical user interface

Description automatically generated**

**Bước 4:** Chọn tần số giao động:

Graphical user interface, diagram

Description automatically generated

**Bước 5**: Cấu hình ADC

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

**Bước 6**: Cấu hình Timer hoạt động ở chế độ PWM

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

**Bước 7**: Cấu hình UART

Graphical user interface, application

Description automatically generated

**Bước 8:** Cấu hình FREERTOS

Graphical user interface

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

**Bước 9:** Đặt tên cho Project và chọn nơi lưu trữ, chọn phần mềm viết chương trình

Graphical user interface, text, application, email

Description automatically generated

1. **Lưu đồ giải thuật:**

**Chương trình con:**

**Diagram

Description automatically generatedDiagram

Description automatically generatedDiagram

Description automatically generated**

**Chương trình chính:**

**Diagram

Description automatically generated**

1. **Mã nguồn chương trình:**

**Chương trình con:**

void StartDefaultTask(void const \* argument)

{

/\* USER CODE BEGIN 5 \*/

/\* Infinite loop \*/

for(;;)

{

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_3, 1);

HAL\_Delay(222);

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_3, 0);

HAL\_Delay(111);

}

/\* USER CODE END 5 \*/

}

void StartTask02(void const \* argument)

{

/\* USER CODE BEGIN StartTask02 \*/

/\* Infinite loop \*/

for(;;)

{

HAL\_ADC\_Start(&hadc2);

HAL\_ADC\_PollForConversion(&hadc2, 100);

adc\_value2 = HAL\_ADC\_GetValue(&hadc2);

v2 = adc\_value2\*3.3/4095;

sprintf(&data\_send[0], "V = %4.2f\t\n", v2);

HAL\_UART\_Transmit(&huart1, &data\_send[0], sizeof(data\_send), 10);

osDelay(1000);

}

/\* USER CODE END StartTask02 \*/

}

void StartTask03(void const \* argument)

{

/\* USER CODE BEGIN StartTask03 \*/

/\* Infinite loop \*/

for(;;)

{

HAL\_ADC\_Start(&hadc1);

HAL\_ADC\_PollForConversion(&hadc1,100);

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

pwm\_value = adc\_value/1.820444;

pwm = pwm\_value;

\_\_HAL\_TIM\_SET\_COMPARE(&htim1,TIM\_CHANNEL\_1,pwm);

}

/\* USER CODE END StartTask03 \*/

}

**Chương trình chính:**

/\* Includes ------------------------------------------------------------------\*/

#include "main.h"

#include "cmsis\_os.h"

/\* Private includes ----------------------------------------------------------\*/

/\* USER CODE BEGIN Includes \*/

#include <stdio.h>

/\* USER CODE END Includes \*/

/\* Private typedef -----------------------------------------------------------\*/

/\* USER CODE BEGIN PTD \*/

/\* USER CODE END PTD \*/

/\* Private define ------------------------------------------------------------\*/

/\* USER CODE BEGIN PD \*/

/\* USER CODE END PD \*/

/\* Private macro -------------------------------------------------------------\*/

/\* USER CODE BEGIN PM \*/

/\* USER CODE END PM \*/

/\* Private variables ---------------------------------------------------------\*/

ADC\_HandleTypeDef hadc1;

ADC\_HandleTypeDef hadc2;

TIM\_HandleTypeDef htim1;

UART\_HandleTypeDef huart1;

osThreadId defaultTaskHandle;

osThreadId myTask02Handle;

osThreadId myTask03Handle;

/\* USER CODE BEGIN PV \*/

uint32\_t adc\_value2;

float v1;

float v2;

uint8\_t data\_send[50];

uint32\_t adc\_value;

float pwm\_value;

uint16\_t pwm;

/\* USER CODE END PV \*/

…

int main(void)

{

/\* USER CODE BEGIN 1 \*/

/\* USER CODE END 1 \*/

/\* MCU Configuration--------------------------------------------------------\*/

/\* Reset of all peripherals, Initializes the Flash interface and the Systick. \*/

HAL\_Init();

/\* USER CODE BEGIN Init \*/

/\* USER CODE END Init \*/

/\* Configure the system clock \*/

SystemClock\_Config();

/\* USER CODE BEGIN SysInit \*/

/\* USER CODE END SysInit \*/

/\* Initialize all configured peripherals \*/

MX\_GPIO\_Init();

MX\_ADC1\_Init();

MX\_ADC2\_Init();

MX\_USART1\_UART\_Init();

MX\_TIM1\_Init();

/\* USER CODE BEGIN 2 \*/

HAL\_TIM\_PWM\_Start(&htim1, TIM\_CHANNEL\_1);

/\* USER CODE END 2 \*/

/\* USER CODE BEGIN RTOS\_MUTEX \*/

/\* add mutexes, ... \*/

/\* USER CODE END RTOS\_MUTEX \*/

/\* USER CODE BEGIN RTOS\_SEMAPHORES \*/

/\* add semaphores, ... \*/

/\* USER CODE END RTOS\_SEMAPHORES \*/

/\* USER CODE BEGIN RTOS\_TIMERS \*/

/\* start timers, add new ones, ... \*/

/\* USER CODE END RTOS\_TIMERS \*/

/\* USER CODE BEGIN RTOS\_QUEUES \*/

/\* add queues, ... \*/

/\* USER CODE END RTOS\_QUEUES \*/

/\* Create the thread(s) \*/

/\* definition and creation of defaultTask \*/

osThreadDef(defaultTask, StartDefaultTask, osPriorityNormal, 0, 128);

defaultTaskHandle = osThreadCreate(osThread(defaultTask), NULL);

/\* definition and creation of myTask02 \*/

osThreadDef(myTask02, StartTask02, osPriorityAboveNormal, 0, 128);

myTask02Handle = osThreadCreate(osThread(myTask02), NULL);

/\* definition and creation of myTask03 \*/

osThreadDef(myTask03, StartTask03, osPriorityHigh, 0, 128);

myTask03Handle = osThreadCreate(osThread(myTask03), NULL);

/\* USER CODE BEGIN RTOS\_THREADS \*/

/\* add threads, ... \*/

/\* USER CODE END RTOS\_THREADS \*/

/\* Start scheduler \*/

osKernelStart();

/\* We should never get here as control is now taken by the scheduler \*/

/\* Infinite loop \*/

/\* USER CODE BEGIN WHILE \*/

while (1)

{

/\* USER CODE END WHILE \*/

/\* USER CODE BEGIN 3 \*/

}

/\* USER CODE END 3 \*/

}

1. **Video minh chứng:**

Link youtube: