**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет «ЛЭТИ»**

**Кафедра САУ**

отчет

**по лабораторной работе №4**

**по дисциплине «Операционные системы реального времени»**

**Тема: «УПРАВЛЕНИЕ ЗАДАЧАМИ»**

**Вариант - 3**

|  |  |  |
| --- | --- | --- |
| Студенты гр. 8491 |  | ТуленковК.А. |
|  |  | Саламахин А. |
| Преподаватель |  | Гречухин М.Н. |

Санкт-Петербург

2022

**Цель работы:** В данной лабораторной работе предлагается создать две задачи, которые будут обращаться к одному и тому же ресурсу – расположенному на плате диоду.

**Задание:**

1. Открыть IDE Keil MDK-ARM, создать новый проект по алгоритму из работы 1. В файле конфигурации FreeRTOSConfig.h необходимо разрешить использование нужных в работе функций (с помощью директив вида #define INCLUDE\_vTaskDelay 1).
2. Создать две функции-задачи. Обе будут пытаться использовать один и тот же ресурс. Создать мьютекс для организации совместного доступа к ресурсу.
3. Сделать так, чтобы обе задачи в течение 4 с мигали диодом по своей программе по очереди.
4. Скомпилировать проект. Загрузить его на плату, наблюдать работу с подключенной периферией.
5. Намеренно «забыть» освободить мьютекс в первой задаче. Пересобрать проект, наблюдать за работой.

**Код программы:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60 | #include "stm32f4xx.h" // Device header  #include "FreeRTOS.h" // ARM.FreeRTOS::RTOS:Core  #include "FreeRTOSConfig.h" // ARM.FreeRTOS::RTOS:Config  #include "task.h" // ARM.FreeRTOS::RTOS:Core  #include "queue.h" // ARM.FreeRTOS::RTOS:Core  #include "inttypes.h"  #include "semphr.h"  **void** **move\_blink1**();  **void** **move\_blink2**();  SemaphoreHandle\_t mutex = NULL;  TaskHandle\_t move\_blink1\_, move\_blink2\_;  **int** **main**()  {  mutex = xSemaphoreCreateMutex();  RCC->AHB1ENR |= RCC\_AHB1ENR\_GPIOAEN;  GPIOA->MODER |= GPIO\_MODER\_MODER5\_1;    xTaskCreate(move\_blink1, "move\_blink", configMINIMAL\_STACK\_SIZE, NULL, **1**, &move\_blink1\_);  xTaskCreate(move\_blink2, "move\_blink2", configMINIMAL\_STACK\_SIZE, NULL, **1**, &move\_blink2\_);  vTaskStartScheduler();    **while**(**1**);  }  **void** **move\_blink1**()  {    **while**(**1**)  {  **if**(xSemaphoreTake(mutex, (TickType\_t)**5**)== pdTRUE)  {  **for**(uint8\_t i = **0**; i < **16**; i++)  {  GPIOA->ODR = (**1** << i);  vTaskDelay(**100**);  }  xSemaphoreGive(mutex);  }  vTaskDelay(**5**);  }  }  **void** **move\_blink2**()  {  **while**(**1**)  {  **if**(xSemaphoreTake(mutex, (TickType\_t)**5**)== pdTRUE)  {  **for**(uint8\_t i = **0**; i < **8**; i++)  {  GPIOA->ODR = (**1** << (**15**-i)) | (**1** << (i));  vTaskDelay(**200**);  }  xSemaphoreGive(mutex);  }  vTaskDelay(**5**);  }  } |
|  |  |

**Описание кода программы:**

move\_blink1 – мигает диодом в виде бегущей полоски.

move\_blink2 – мигает диодом в виде бегущей полоски, сходящейся к центру.

**Работа программы:**

**Задание 1, 2, 3, 4:**

Выполняем программу корректно, с освобождением мьютекса после выполнения задачи, функция xSemaphoreGive(mutex). Программа работает как положено. Вторая задача не начнет выполняться пока не выполнится первая.

**Задание 5:**

Намеренно забываем освободить мьютекс. Первая задача выполняется и дальше программа ничего не делает, так как у первой задаче мьютекс уже захвачен и новый захват

невозможен, а во второй – мьютекс не может быть захвачен, так как он занят первой

задачей.

**Вывод:** В ходе выполнения работы по созданию программы, реализующей различные вариации работы светодиода, были получены навыки работы с задачами с совместным доступом к ресурсам, а также навыки работы с мьютексами реализованными в ОСРВ FreeRTOS.