

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования

«ТОМСКИЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И  
РАДИОЭЛЕКТРОНИКИ» (ТУСУР)

Кафедра комплексной информационной безопасности электронно-  
вычислительных систем (КИБЭВС)

## РАБОТА С ПОТОКАМИ В MBED OS

Отчет по лабораторной работе №5  
по дисциплине «Системное программирование»

Студенты гр. 718-1

\_\_\_\_\_ Прозорова Е. А.

\_\_\_\_\_ Новокрещенных В. И.

«\_\_» \_\_\_\_\_ 2022

Принял

М.н.с. ИСИБ

\_\_\_\_\_ Калинин Е. О.

«\_\_» \_\_\_\_\_ 2022

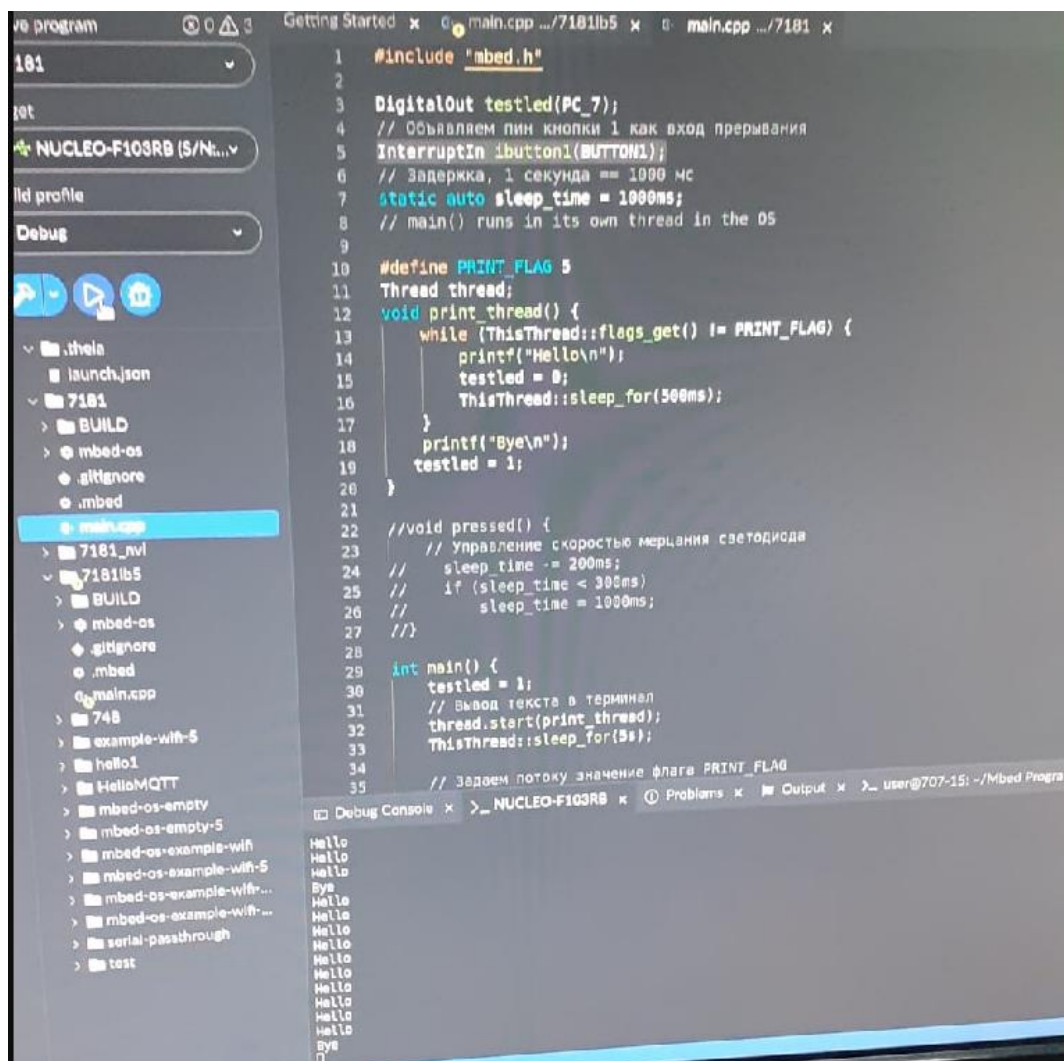
Томск 2022

## 1 Введение

Цель работы: Получение навыков работы с потоками в Mbed OS.

## 2 Ход работы

По заданию лабораторной работы была написана программа, реализующая вывод текста за определенное время через поток с помощью флага (рисунок 2.1).



```
1 #include "mbed.h"
2
3 DigitalOut testled(PC_7);
4 // Объявляем пин кнопки 1 как вход прерывания
5 InterruptIn button1(BUTTON1);
6 // Задержка, 1 секунда == 1000 мс
7 static auto sleep_time = 1000ms;
8 // main() runs in its own thread in the OS
9
10 #define PRINT_FLAG 5
11 Thread thread;
12 void print_thread() {
13     while (ThisThread::flags_get() != PRINT_FLAG) {
14         printf("Hello\n");
15         testled = 0;
16         ThisThread::sleep_for(500ms);
17     }
18     printf("Bye\n");
19     testled = 1;
20 }
21
22 //void pressed() {
23 //    // Управление скоростью мерцания светодиода
24 //    sleep_time -= 200ms;
25 //    if (sleep_time < 300ms)
26 //        sleep_time = 1000ms;
27 //}
28
29 int main() {
30     testled = 1;
31     // Вывод текста в терминал
32     thread.start(print_thread);
33     ThisThread::sleep_for(5s);
34
35     // Задаем потоку значение флага PRINT_FLAG
```

Debug Console x >\_ NUCLEO-F103RB x Problems x Output x >\_ user@707-15: ~/Mbed Program

```
Hello
Hello
Hello
Bye
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Bye
```

Рисунок 2.1 – Вывод текста за определенное время

Также, была написана программа реализующая постоянный поток, через который было реализовано мигание диода (рисунки 2.2 – 2.3).

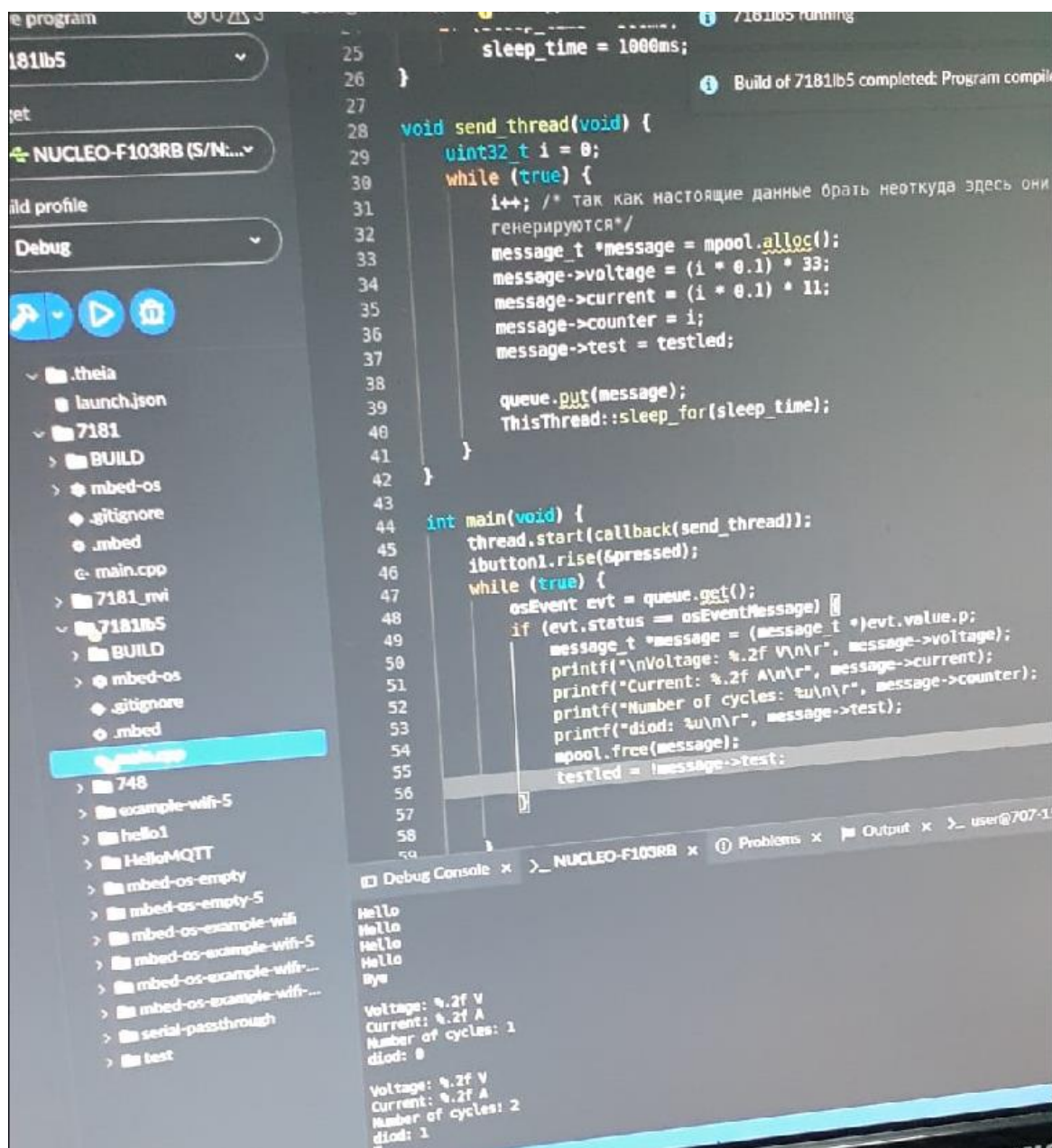


Рисунок 2.2 – Программа постоянного потока

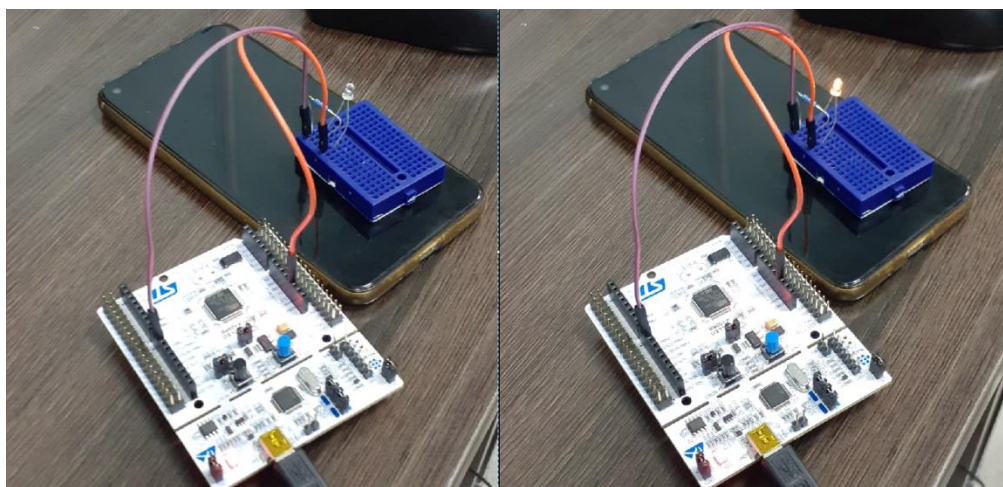


Рисунок 2.3 - Мигание диода

### 3 Заключение

В рамках выполнения лабораторной работы были получены навыки работы с потоками в Mbed OS. Была написана программа, реализующая мигание светодиода.

## Приложение А

(Код программы)

```
#include "mbed.h"

DigitalOut testled(PC_7);
// Объявляем пин кнопки 1 как вход прерывания
InterruptIn ibutton1(BUTTON1);
// Задержка, 1 секунда == 1000 мс
static auto sleep_time = 1000ms;
// main() runs in its own thread in the OS

#define PRINT_FLAG 5
Thread thread;
void print_thread() {
    while (ThisThread::flags_get() != PRINT_FLAG) {
        printf("Hello\n");
        testled = 0;
        ThisThread::sleep_for(500ms);
    }
    printf("Bye\n");
    testled = 1;
}

//void pressed() {
    // Управление скоростью мерцания светодиода
    // sleep_time -= 200ms;
    // if (sleep_time < 300ms)
    //     sleep_time = 1000ms;
    //}
```

```
int main() {  
    testled = 1;  
    // Вывод текста в терминал  
    thread.start(print_thread);  
    ThisThread::sleep_for(5s);  
  
    // Задаем потоку значение флага PRINT_FLAG  
    thread.flags_set(PRINT_FLAG);  
  
    // Прикрепляем функцию для обработки нажатия  
    //ibutton1.rise(&pressed);  
    //while (true) {  
        // testled = !testled;  
        // ThisThread::sleep_for(sleep_time);  
  
    //}  
  
}
```

```
#include "mbed.h"
```

```
DigitalOut testled(PC_7);
```

```
InterruptIn ibutton1(BUTTON1);
```

```
static auto sleep_time = 1000ms;
```

```
typedef struct {
```

```
    float voltage; /* результат измерения напряжения */
```

```
    float current; /* результат измерения тока */
```

```
    uint32_t counter;
```

```
    int test;
```

```
} message_t;
```

```
/*здесь используется объект класса MemoryPool, для определения и  
управления пулом памяти фиксированного размера, информацию об этом  
классе изучите в документации */
```

```
MemoryPool<message_t, 16> mpool;
```

```
Queue<message_t, 16> queue;
```

```
Thread thread;
```

```
/* поток отправитель */
```

```
void pressed() {
```

```
    // Управление скоростью мерцания светодиода
```

```
    sleep_time -= 200ms;
```

```
    if (sleep_time < 300ms)
```

```
        sleep_time = 1000ms;
```

```
}
```

```
void send_thread(void) {
```

```
    uint32_t i = 0;
```

```
    while (true) {
```



```

    i++; /* так как настоящие данные брать неоткуда здесь они
    генерируются*/
    message_t *message = mpool.alloc();
    message->voltage = (i * 0.1) * 33;
    message->current = (i * 0.1) * 11;
    message->counter = i;
    message->test = testled;

    queue.put(message);
    ThisThread::sleep_for(sleep_time);
}
}

int main(void) {
    thread.start(callback(send_thread));
    ibutton1.rise(&pressed);
    while (true) {
        osEvent evt = queue.get();
        if (evt.status == osEventMessage) {
            message_t *message = (message_t *)evt.value.p;
            printf("\nVoltage: %.2f V\n\r", message->voltage);
            printf("Current: %.2f A\n\r", message->current);
            printf("Number of cycles: %u\n\r", message->counter);
            printf("diod: %u\n\r", message->test);
            mpool.free(message);
            testled = !message->test;
        }
    }
}

```

## Приложение Б

(Ссылка на репозиторий)

Ссылки на репозитории gitlab:

<https://github.com/SlavaNovok/SystemProgram.git>

<https://github.com/ledastro/SystemProgramm.git>