

Министерство науки и высшего образования РФ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ
И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра комплексной информационной безопасности электронно-
вычислительных систем (КИБЭВС)

РАБОТА С ПОТОКАМИ В MBED STUDIO
Отчет по лабораторной работе №5
по дисциплине «Системное программирование»

Выполнил:

Студент гр. 718

_____ Керноз. И.С.

_____ Буравский Н.С.

____. ____ 2022

Принял:

м.н.с. ИСИБ

_____ Калинин Е.О.

____. ____ 2022

Введение

Цель работы: изучить работу с потоками. Научиться разбивать задачу на части, для последующего их выполнения различными потоками в Mbed OS.

Задача: на основании рассмотренных примеров реализуйте 2 программы с модификациями по заданию преподавателя:

- мигание светодиода, параллельно работающему потоку;
- мигание светодиода, основываясь на соседнем потоке данных.
- .

2 ХОД РАБОТЫ

Для работы имеется плата NUCLEO-F103RB (рисунок 2.1).

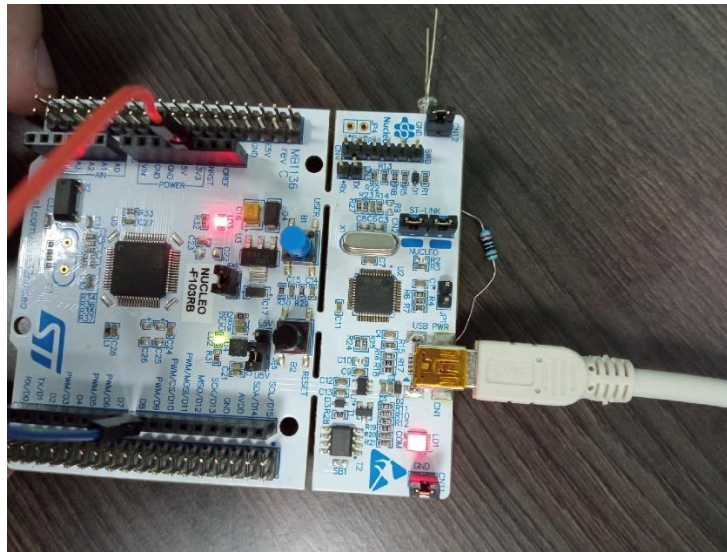


Рисунок 2.1 – Плата NUCLEO-F103RB

Напишем код для работы первого задания – мигание светодиодов и запустим программу (рисунок 2.2). Листинг приведен в приложении А.

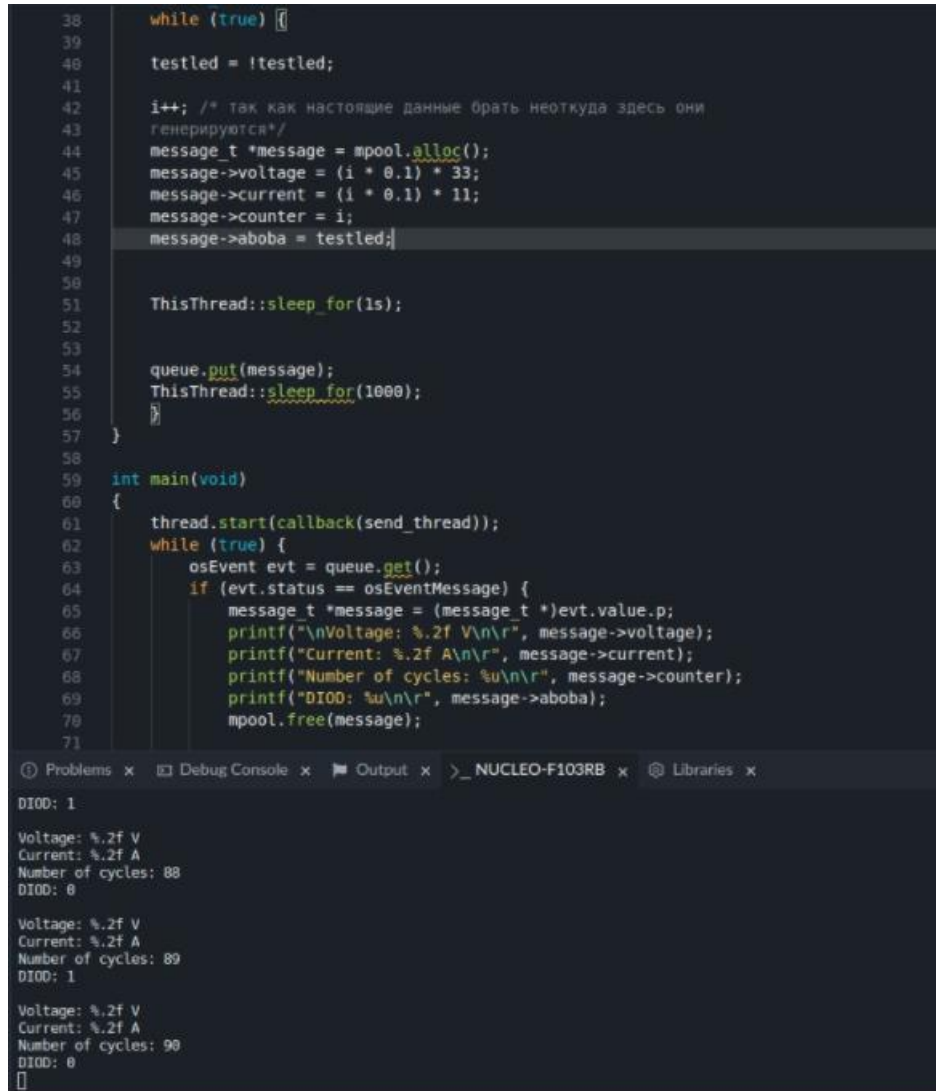
```
11 #include "stm32f10x.h"
12 #include "ThisThread.h"
13 #include "mbed.h"
14 #include "pinmap.h"
15 #include <stdio>
16 #define FLAG 2
17
18 DigitalOut oled1(LED1);
19 DigitalOut testled(PC_7);
20 // Задержка, 1 секунда == 1000 мс
21 Thread thread;
22
23 void print_term()
24 {
25     while (ThisThread::flags_get() != FLAG)
26     {
27         printf("Hello\n");
28         ThisThread::sleep_for(1500ms);
29     }
30     printf("Goodbye\n");
31 }
32
33 int main()
34 {
35     testled = 1;
36     thread.start(print_term);
37     ThisThread::sleep_for(5s);
38     // Задаем потоку значение флага PRINT_FLAG
39     thread.flags_set(FLAG);
40     while (true) {
41         oled1 = !oled1;
42         testled = !testled; // Включение/выключение светодиода
43         ThisThread::sleep_for(1s);
44     }
45 }
```

Problems x Debug Console x Output x >_ NUCLEO-F103RB x Libraries x

Goodbye
Hello
Hello
Hello
Hello
Goodbye

Рисунок 2.2 – Код для работы светодиодов

Для второй части задания необходимо реализовать программу с потоком, где меняется значение переменной для мигания диодов. Напишем код программы (рисунок 2.3). Листинг приведен в приложении Б.



```
38 while (true) {
39
40     testled = !testled;
41
42     i++; /* так как настоящие данные брать неоткуда здесь они
43     генерируются*/
44     message_t *message = mpool.alloc();
45     message->voltage = (i * 0.1) * 33;
46     message->current = (i * 0.1) * 11;
47     message->counter = i;
48     message->aboba = testled;
49
50
51     ThisThread::sleep_for(1s);
52
53
54     queue.put(message);
55     ThisThread::sleep_for(1000);
56 }
57
58
59 int main(void)
60 {
61     thread.start(callback(send_thread));
62     while (true) {
63         osEvent evt = queue.get();
64         if (evt.status == osEventMessage) {
65             message_t *message = (message_t *)evt.value.p;
66             printf("\nVoltage: %.2f V\n\r", message->voltage);
67             printf("Current: %.2f A\n\r", message->current);
68             printf("Number of cycles: %u\n\r", message->counter);
69             printf("DI00: %u\n\r", message->aboba);
70             mpool.free(message);
71         }
72     }
73 }
```

Problems x Debug Console x Output x >_ NUCLEO-F103RB x Libraries x

DI00: 1

Voltage: %.2f V
Current: %.2f A
Number of cycles: 88
DI00: 0

Voltage: %.2f V
Current: %.2f A
Number of cycles: 89
DI00: 1

Voltage: %.2f V
Current: %.2f A
Number of cycles: 90
DI00: 0

Рисунок 2.3 – Поточное применение значений для диодов

Проверим работоспособность кода на плате (рисунок 2.4).

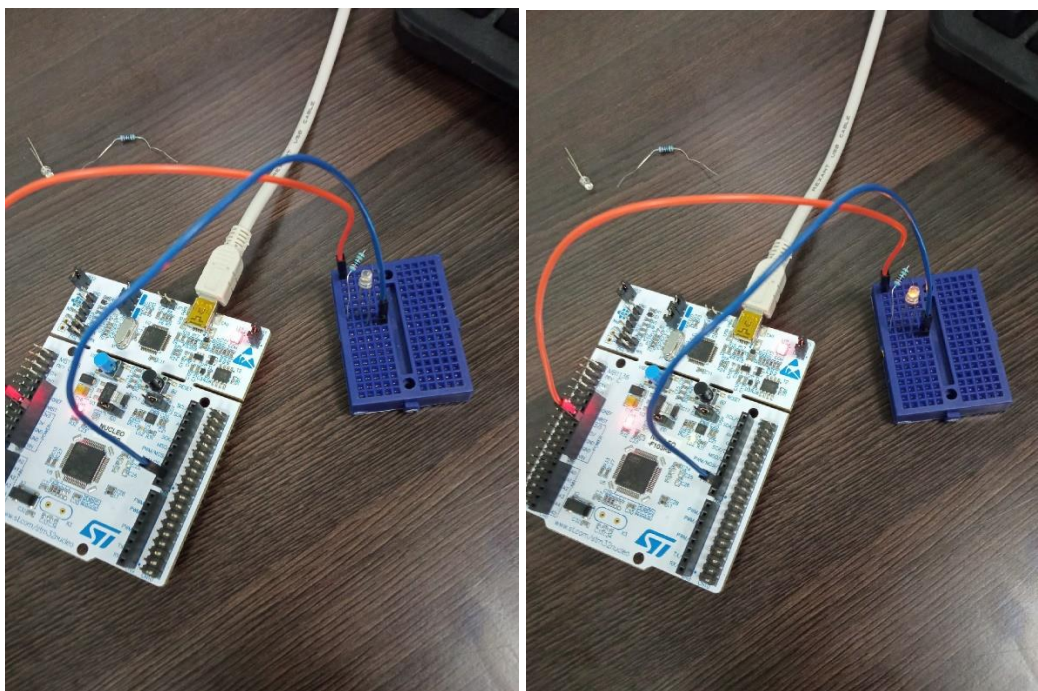


Рисунок 2.4 – Работоспособность кода на плате.

Заключение

В результате выполнения данной лабораторной работы было произведено изучить работу с потоками, научились разбивать задачу на части, для последующего их выполнения различными потоками в Mbed OS. Написаны программы для работы платы: поток мигания диодов и вывода в терминал, и присвоение значений для диодов из потока.

Ссылка на гитхаб: <https://github.com/Kernoz-Igor/splab5>

Приложение А
(обязательное)
Листинг программы led.c

```
#include "DigitalOut.h"
#include "PinNameAliases.h"
#include "PinNamesTypes.h"
#include "ThisThread.h"
#include "mbed.h"
#include "pinmap.h"
#include <stdio>
#define FLAG 2

DigitalOut oled1(LED1);
DigitalOut testled(PC_7);
// Задержка, 1 секунда == 1000 мс
Thread thread;
void print_term()
{
while (ThisThread::flags_get() != FLAG)
{
printf("Hello\n");
ThisThread::sleep_for(1500ms);
}
printf("Goodbye\n");
}
int main()
{
testled = 1;
thread.start(print_term);
ThisThread::sleep_for(5s);
// Задаем потоку значение флага PRINT_FLAG
thread.flags_set(FLAG);
while (true) {
oled1 = !oled1;
testled = !testled; // Включение/выключение светодиода
ThisThread::sleep_for(1s);
}
}
```

Приложение Б

(обязательное)

Листинг программы button.c

```
#include "DigitalOut.h"
#include "PinNameAliases.h"
#include "PinNamesTypes.h"
#include "ThisThread.h"
#include "mbed.h"
#include "pinmap.h"
#include <stdio>
#include "mbed.h"

DigitalOut oled1(LED1);
DigitalOut testled(PC_7);

typedef struct {
float voltage; /* результат измерения напряжения */
float current; /* результат измерения тока */
uint32_t counter;
int aboba;
} 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 send_thread(void)
{
uint32_t i = 0;
while (true) {

testled = !testled;

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->aboba = testled;
```

```
ThisThread::sleep_for(1s);
```

```
queue.put(message);
ThisThread::sleep_for(1000);
}
}
```

```
int main(void)
{
    thread.start(callback(send_thread));
    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->aboba);
            mpool.free(message);

            oled1 = message->aboba;
            testled = message->aboba;
        }

        //oled1 = !oled1; // Включение/выключение светодиода
        ThisThread::sleep_for(1s);
    }
}
```