

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное  
учреждение высшего образования  
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ  
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)  
Кафедра безопасности информационных систем (БИС)

### **ЗНАКОМСТВО С ОСРВ MBED OS**

Отчёт по лабораторной работе №4  
по дисциплине «Системное программирование»

Студенты гр. 748

\_\_\_\_\_ Е. В. Граборова

\_\_\_\_\_ Д. Е. Мануилова

«\_\_» \_\_\_\_\_ 2022

Принял

М.н.с ИСИБ

\_\_\_\_\_ Е. О. Калинин

«\_\_» \_\_\_\_\_ 2022

## Введение

Целью работы является ознакомление с операционной системой реального времени (ОСРВ) Mbed OS.

Изучить методы и функции работы с пинами платы, механизм прерывания, таймеров и управления потоками в ОСРВ Mbed OS, на примере рассмотренных программ.

Познакомиться с инструментами отладки, имеющимися в Mbed Studio.

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

1. мигание светодиодом;
2. обработка нажатия кнопок.

## 2 Ход работы

Реализуем простую программу с использованием ОСРВ Mbed OS и официального инструмента разработки Mbed Studio. Откроем Mbed Studio. Используя меню программы «File > New Program...», откроем диалоговое меню «New Program». В списке «Select example program» выберем пустой проект «Mbed OS 6: empty Mbed OS program». В поле «Program name» зададим имя приложения (рисунок 2.1).

Определим местонахождения файлов Mbed OS в директорию проекта, выбрав пункт «Store Mbed OS in the program folder (+1GB)». После, нажимаем «Add Program». Копия дистрибутива Mbed OS будет загружена в папку проекта, предоставляя возможность вносить свои правки.

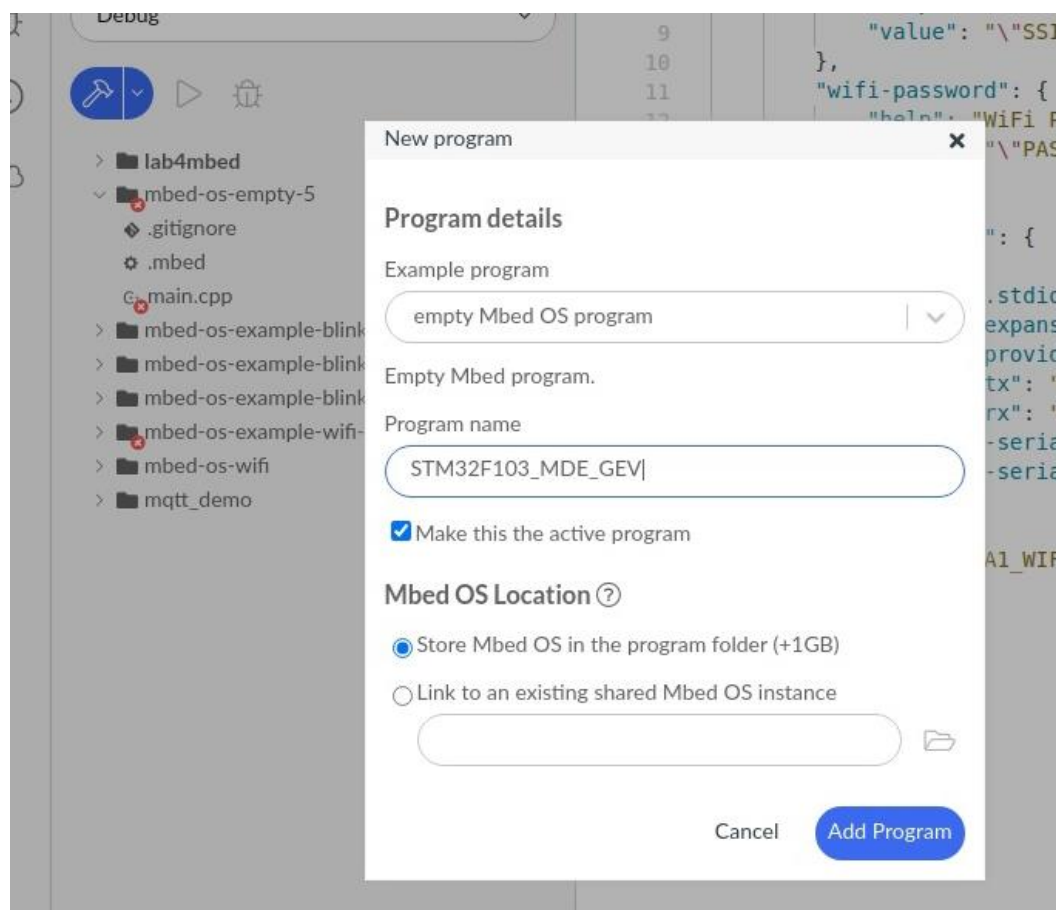


Рисунок 2.1 – Окно создания нового проекта

Далее необходимо выбрать целевую платформу из списка поддерживаемых плат. Для этого подключаем плату к компьютеру, в списке

«Target» нажимаем на иконку «Manage custom targets»т (рисунок 2.2).

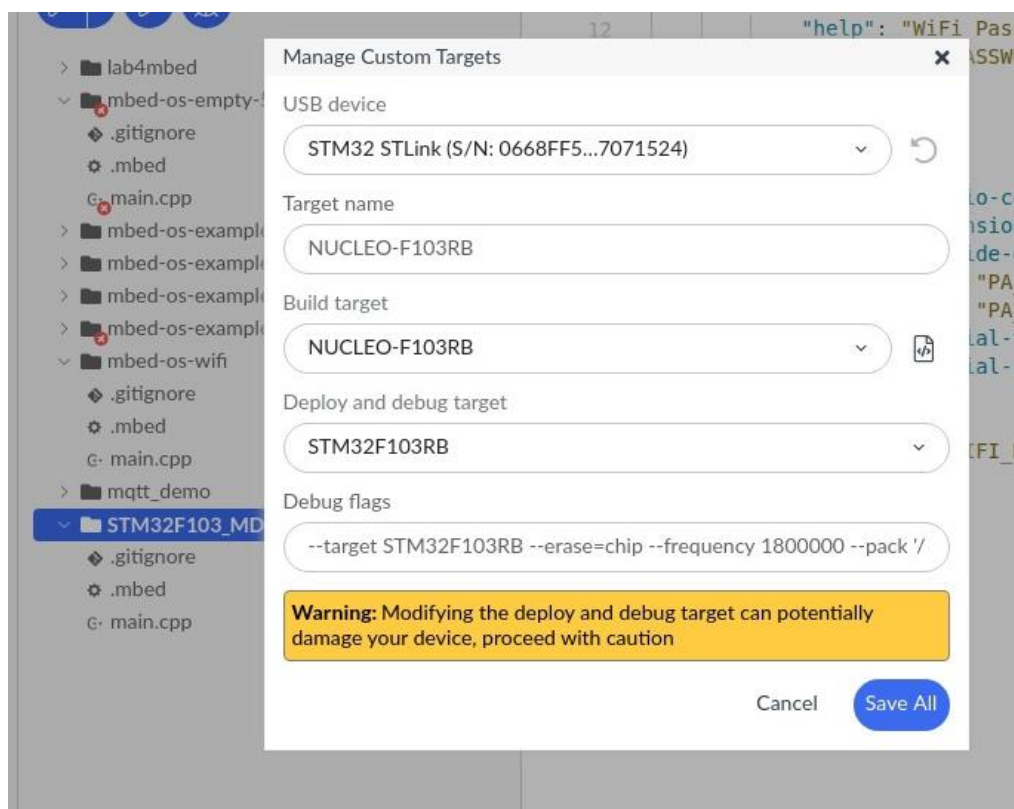


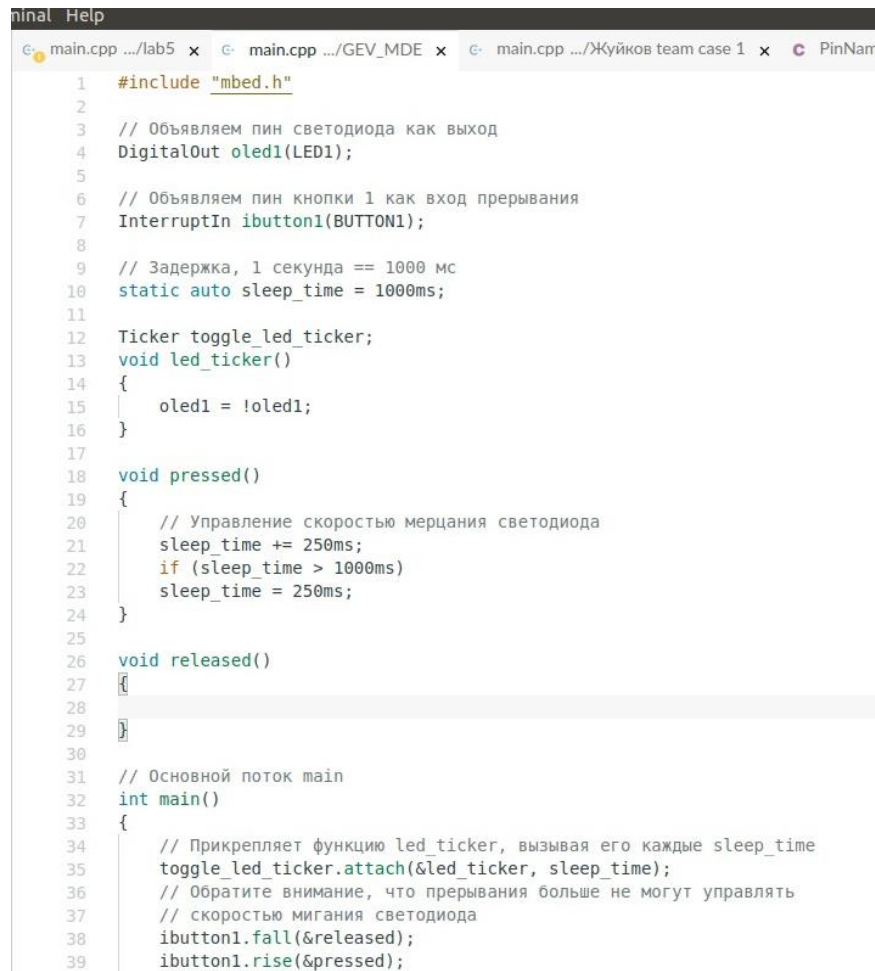
Рисунок 2.2 – Окно подключения платы

После успешного создания проекта, будет выведена простая пустая программа, содержащая лишь определение главной функции «main» и бесконечного цикла (рисунок 2.3).



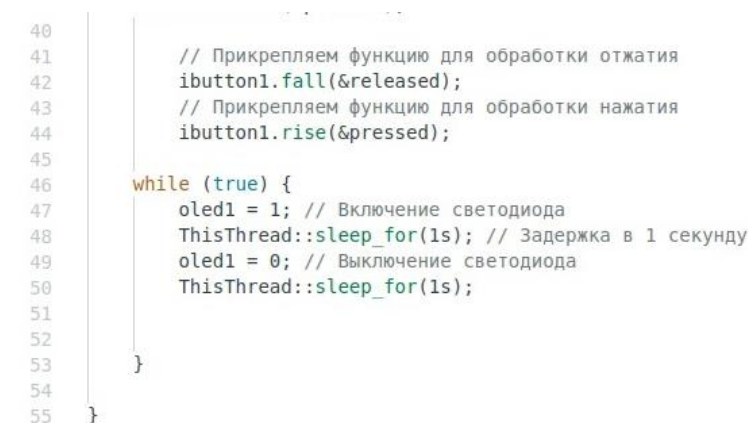
Рисунок 2.3 – Сгенерированная программа

Модифицируем программу так, чтобы при нажатии на пользовательскую кнопку 1 загорается светодиод, а при отпускании кнопки светодиод затухал (рисунки 2.4-2.5).



```
1  #include "mbed.h"
2
3  // Объявляем пин светодиода как выход
4  DigitalOut oled1(LED1);
5
6  // Объявляем пин кнопки 1 как вход прерывания
7  InterruptIn ibutton1(BUTTON1);
8
9  // Задержка, 1 секунда == 1000 мс
10 static auto sleep_time = 1000ms;
11
12 Ticker toggle_led_ticker;
13 void led_ticker()
14 {
15     oled1 = !oled1;
16 }
17
18 void pressed()
19 {
20     // Управление скоростью мерцания светодиода
21     sleep_time += 250ms;
22     if (sleep_time > 1000ms)
23         sleep_time = 250ms;
24 }
25
26 void released()
27 {
28 }
29
30
31 // Основной поток main
32 int main()
33 {
34     // Прикрепляет функцию led_ticker, вызывая его каждые sleep_time
35     toggle_led_ticker.attach(&led_ticker, sleep_time);
36     // Обратите внимание, что прерывания больше не могут управлять
37     // скоростью мигания светодиода
38     ibutton1.fall(&released);
39     ibutton1.rise(&pressed);
```

Рисунок 2.4 – Первая часть кода программы



```
40
41     // Прикрепляем функцию для обработки отжатия
42     ibutton1.fall(&released);
43     // Прикрепляем функцию для обработки нажатия
44     ibutton1.rise(&pressed);
45
46     while (true) {
47         oled1 = 1; // Включение светодиода
48         ThisThread::sleep_for(1s); // Задержка в 1 секунду
49         oled1 = 0; // Выключение светодиода
50         ThisThread::sleep_for(1s);
51     }
52 }
53
54
55 }
```

Problems x Output x Libraries x >\_ NUCLEO-F103RB x

Рисунок 2.5 – Вторая часть кода программы

### 3 Индивидуальное задание

Изучив методы и функции работы с пинами платы, механизмы прерывания, таймеры и управление потоками в OSCPB Mbed OS, была реализованная программа с модификацией работы светодиодов (Приложение А).

Откроем Mbed Studio. Используя меню программы «File > New Program...», откроем диалоговое меню «New Program». В списке «Select example program» выберем пустой проект «Mbed OS 6: empty Mbed OS program». В поле «Program name» зададим имя приложения (рисунок 3.1)

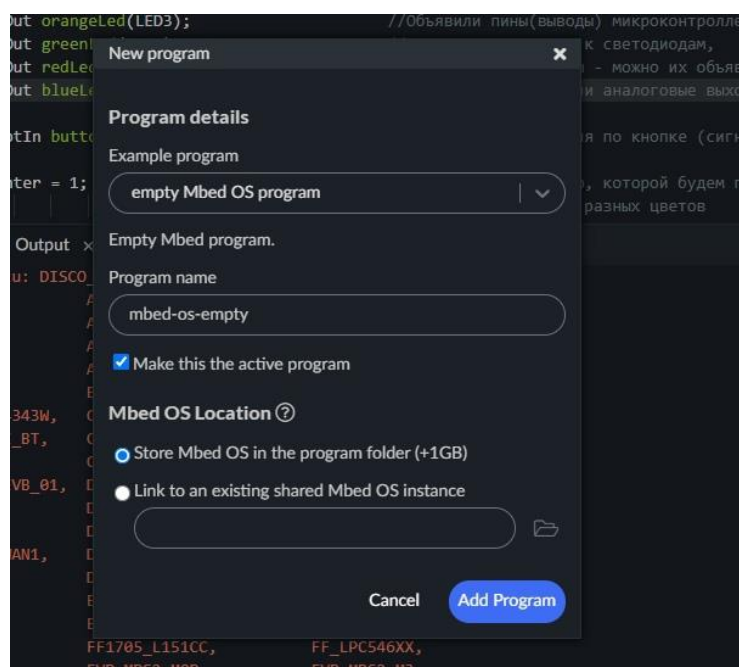


Рисунок 3.1 – Окно создания нового проекта для задания

Написав код программы, предполагающий выполнение задания преподавателя: осуществление мигания светодиодов с разными временными промежутками и цветами посредством нажатия кнопки, который представлен в приложении А, столкнулись с ошибкой. Выяснилось, что программирование данного микропроцессора не поддерживается в инструменте разработки Mbed Studio (рисунок 3.2).

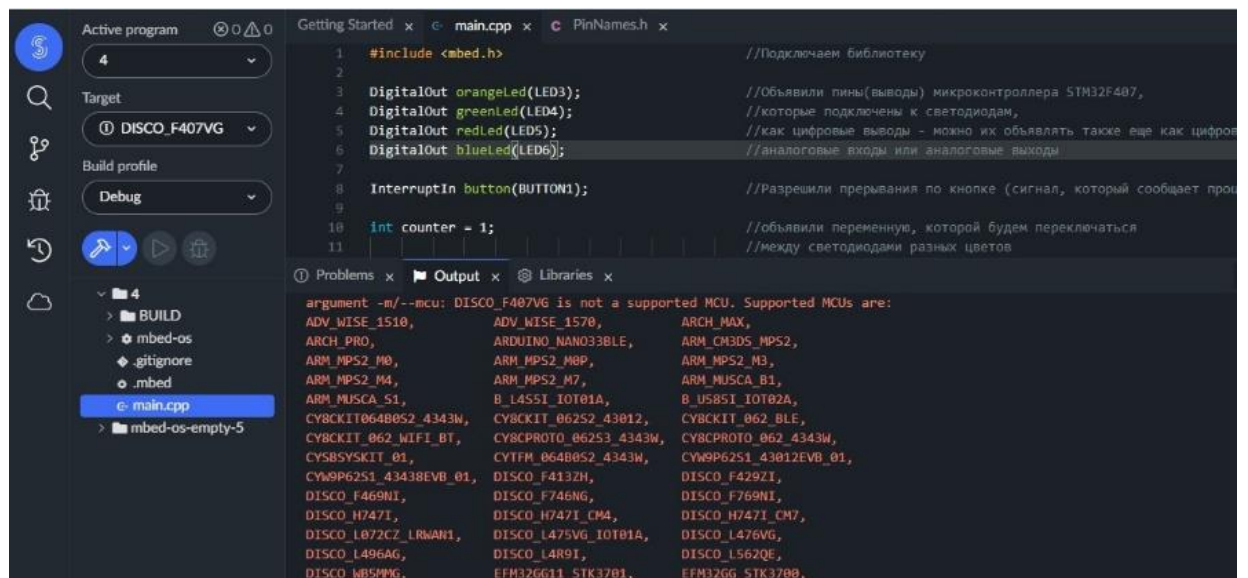


Рисунок 3.2 – Не поддерживается микропроцессор в MBED STUDIO

Было принято решение перенести выполнение задания в среду разработки visual studio code (рисунок 3.3-3.5).

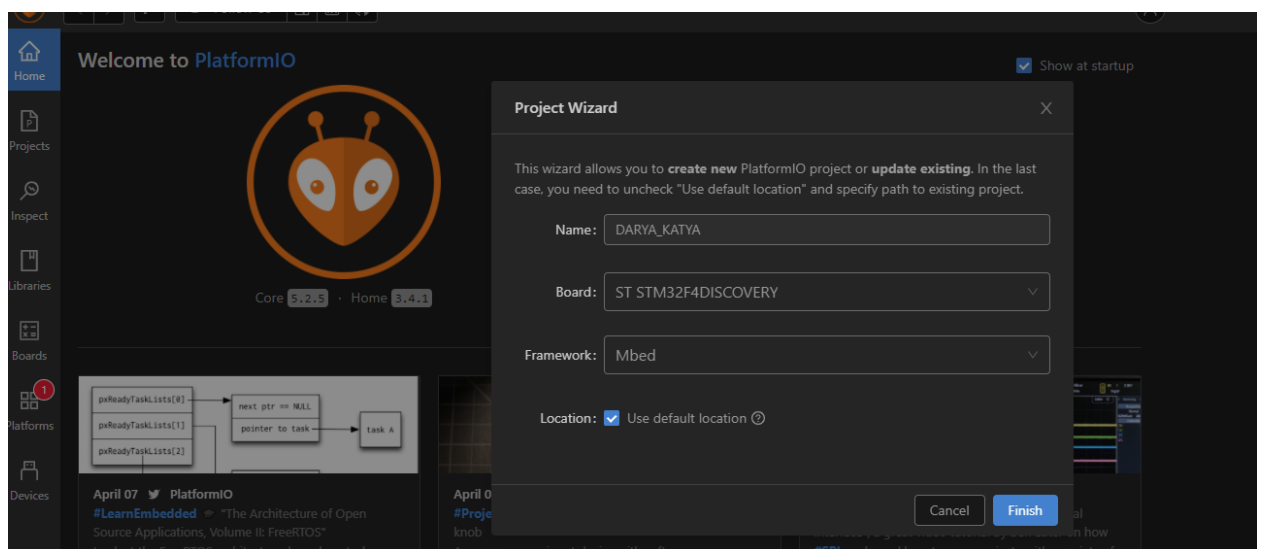


Рисунок 3.3 – Создание проекта в visual studio code с MBED OS



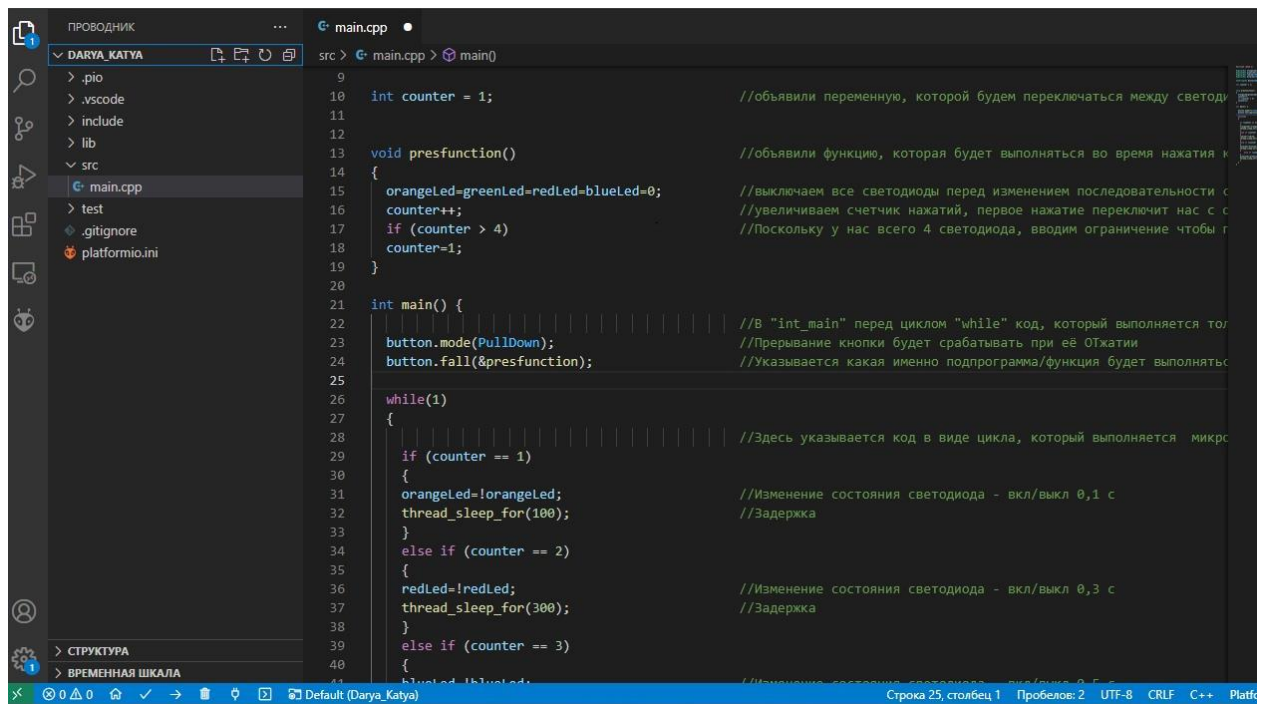


Рисунок 3.4 – Код программы в visual studio code

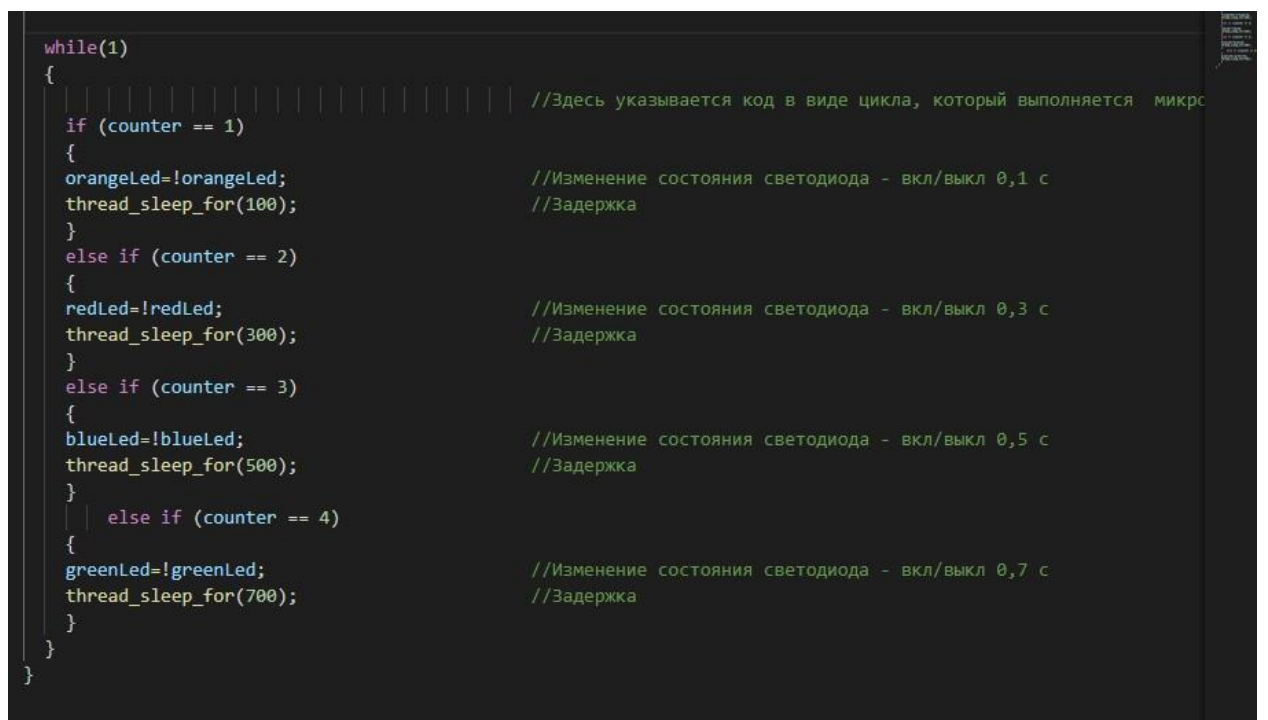
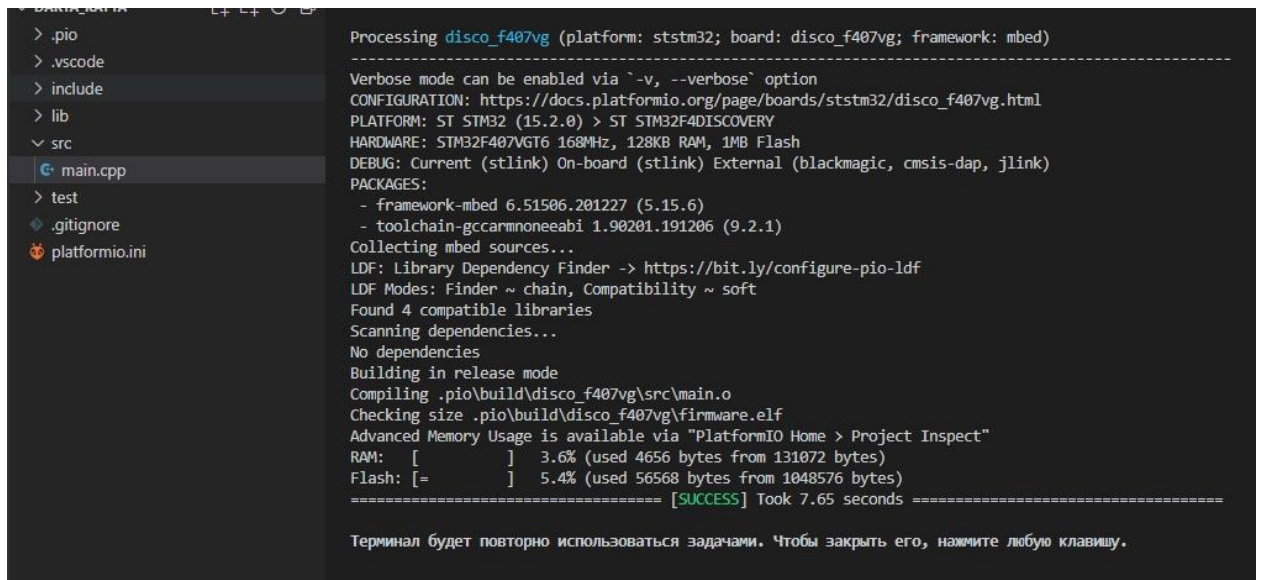


Рисунок 3.5 – Код программы в visual studio code

Была осуществлена сборка проекта, без загрузки на плату (рисунок 3.6) и с загрузкой на плату (рисунок 3.7).



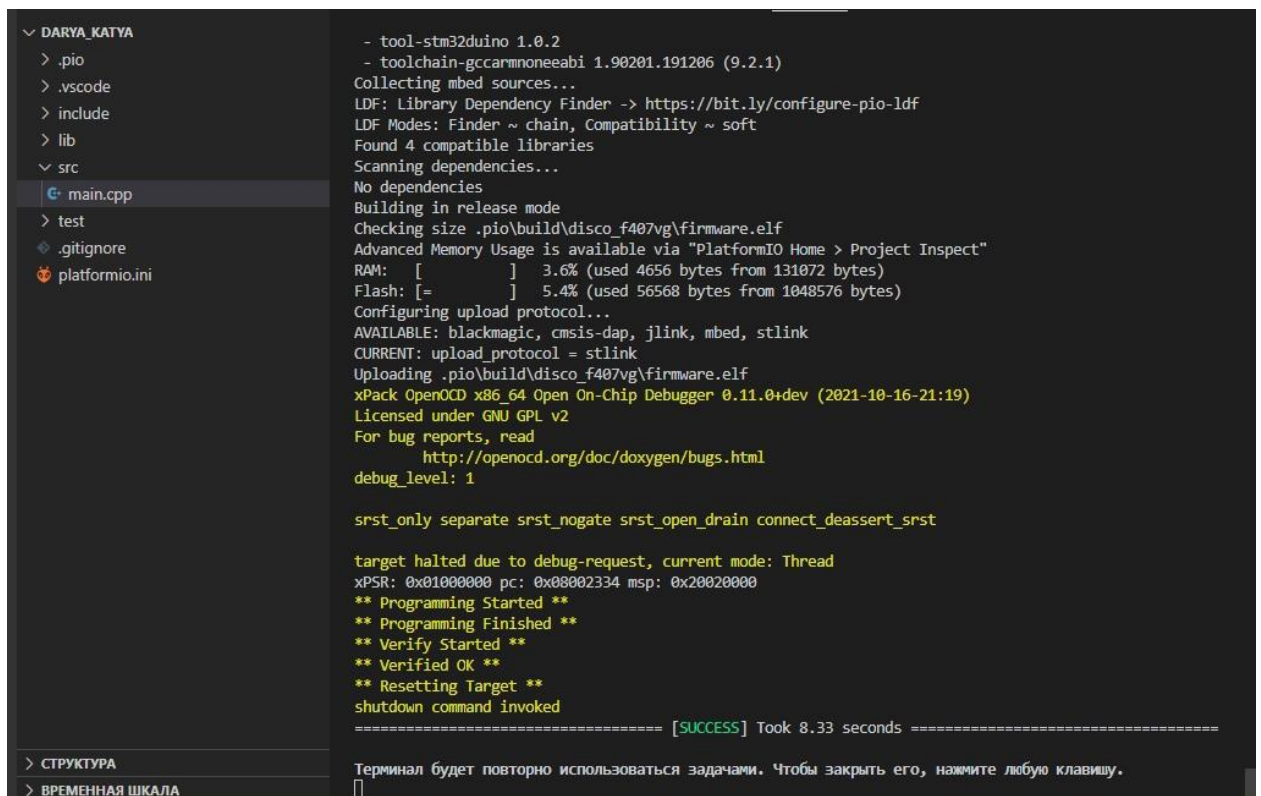


```
> .pio
> .vscode
> include
> lib
> src
main.cpp
> test
.gitignore
platformio.ini

Processing disco_f407vg (platform: ststm32; board: disco_f407vg; framework: mbed)
-----
Verbose mode can be enabled via `-v, --verbose` option
CONFIGURATION: https://docs.platformio.org/page/boards/ststm32/disco_f407vg.html
PLATFORM: ST STM32 (15.2.0) > ST STM32F4DISCOVERY
HARDWARE: STM32F407VGT6 168MHz, 128KB RAM, 1MB Flash
DEBUG: Current (stlink) On-board (stlink) External (blackmagic, cmsis-dap, jlink)
PACKAGES:
- framework-mbed 6.51506.201227 (5.15.6)
- toolchain-gccarmnoneabi 1.90201.191206 (9.2.1)
Collecting mbed sources...
LDF: Library Dependency Finder -> https://bit.ly/configure-pio-ldf
LDF Modes: Finder ~ chain, Compatibility ~ soft
Found 4 compatible libraries
Scanning dependencies...
No dependencies
Building in release mode
Compiling .pio\build\disco_f407vg\src\main.o
Checking size .pio\build\disco_f407vg\firmware.elf
Advanced Memory Usage is available via "PlatformIO Home > Project Inspect"
RAM: [ ] 3.6% (used 4656 bytes from 131072 bytes)
Flash: [= ] 5.4% (used 56568 bytes from 1048576 bytes)
===== [SUCCESS] Took 7.65 seconds =====

Терминал будет повторно использоваться задачами. Чтобы закрыть его, нажмите любую клавишу.
```

Рисунок 3.6 – Отражение успешной сборки, без загрузки на плату



```
> DARYA_KATYA
> .pio
> .vscode
> include
> lib
> src
main.cpp
> test
.gitignore
platformio.ini

- tool-stm32duino 1.0.2
- toolchain-gccarmnoneabi 1.90201.191206 (9.2.1)
Collecting mbed sources...
LDF: Library Dependency Finder -> https://bit.ly/configure-pio-ldf
LDF Modes: Finder ~ chain, Compatibility ~ soft
Found 4 compatible libraries
Scanning dependencies...
No dependencies
Building in release mode
Checking size .pio\build\disco_f407vg\firmware.elf
Advanced Memory Usage is available via "PlatformIO Home > Project Inspect"
RAM: [ ] 3.6% (used 4656 bytes from 131072 bytes)
Flash: [= ] 5.4% (used 56568 bytes from 1048576 bytes)
Configuring upload protocol...
AVAILABLE: blackmagic, cmsis-dap, jlink, mbed, stlink
CURRENT: upload_protocol = stlink
Uploading .pio\build\disco_f407vg\firmware.elf
xPack OpenOCD x86_64 Open On-Chip Debugger 0.11.0+dev (2021-10-16-21:19)
Licensed under GNU GPL v2
For bug reports, read
http://openocd.org/doc/doxygen/bugs.html
debug_level: 1

srst_only separate srst_nogate srst_open_drain connect_deassert_srst

target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x08002334 msp: 0x20020000
** Programming Started **
** Programming Finished **
** Verify Started **
** Verified OK **
** Resetting Target **
shutdown command invoked
===== [SUCCESS] Took 8.33 seconds =====

Терминал будет повторно использоваться задачами. Чтобы закрыть его, нажмите любую клавишу.
```

Рисунок 3.7 – Отражение успешной сборки и загрузки на плату

На рисунке 3.8 представлено успешное выполнение индивидуального задания.

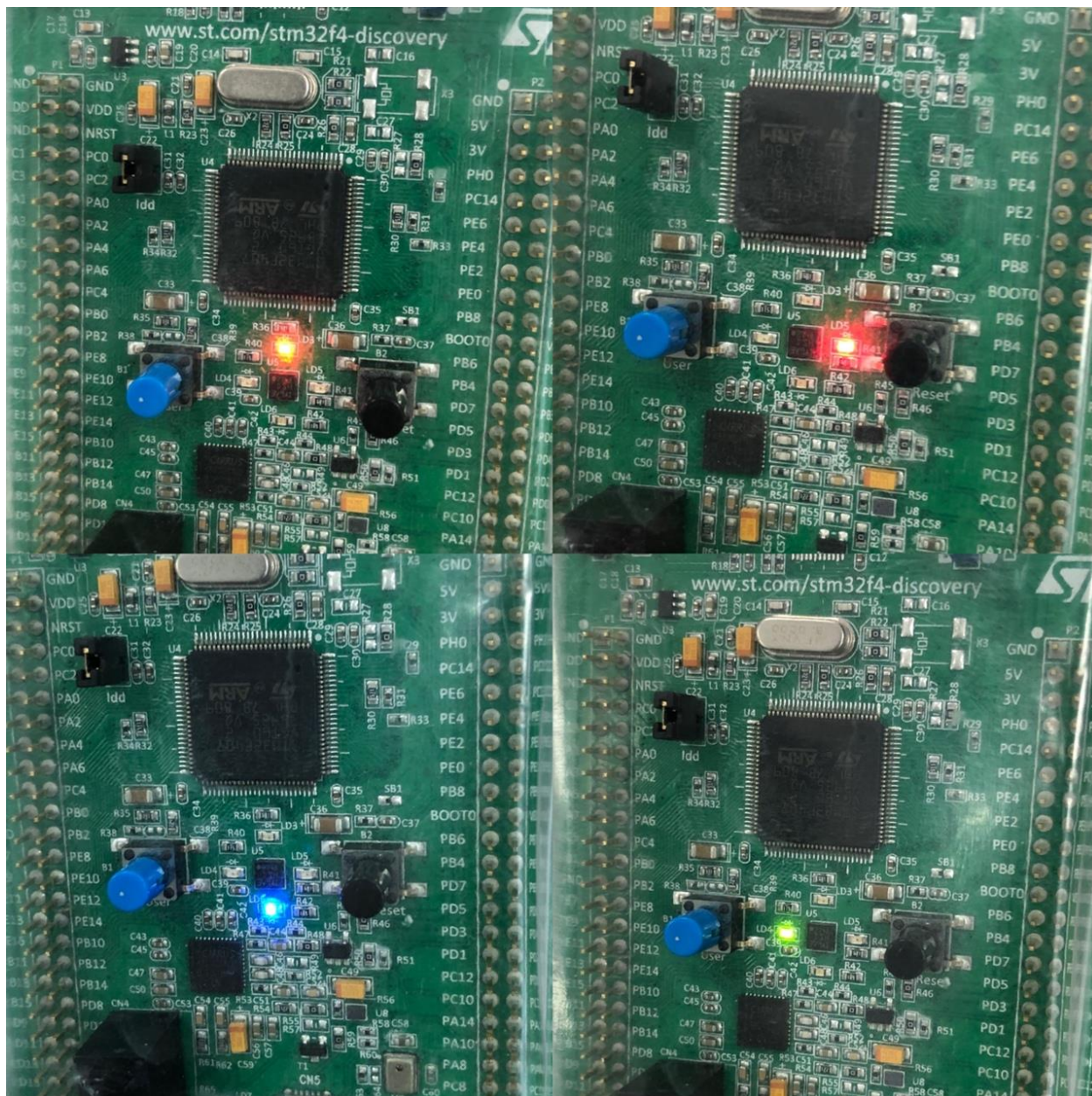


Рисунок 3.8 – Успешное выполнение индивидуального задания

## Заключение

В ходе выполнения лабораторной работы ознакомились с операционной системой реального времени (ОСРВ) Mbed OS. Изучили инструменты отладки, имеющиеся в Mbed Studio.

Изучили методы и функции работы с пинами платы, механизмы прерывания, таймеры и управление потоками в ОСРВ Mbed OS, на примере рассмотренных программ.

На основании рассмотренного примера реализовали 2 программы с модификациями по заданию преподавателя:

1. мигание светодиодом;
2. обработка нажатия кнопок.

Приложение А  
(обязательное)  
Реализации программы

```
#include <mbed.h> //Подключаем библиотеку

DigitalOut orangeLed(LED3); //Объявили пины(выводы) микроконтроллера
STM32F407, которые подключены к светодиодам,
DigitalOut greenLed(LED4);
DigitalOut redLed(LED5);
DigitalOut blueLed(LED6);

InterruptIn button(BUTTON1); //Разрешили прерывания по кнопке

int counter = 1; //объявили переменную, которой будем переключаться между
светодиодами разных цветов
void presfunction() //объявили функцию, которая будет выполняться во время
нажатия кнопки
{
orangeLed=greenLed=redLed=blueLed=0; //выключаем все светодиоды перед
изменением последовательности светодиодов
counter++; //увеличиваем счетчик нажатий, первое нажатие переключит нас с
оранжевого светодиода на красный и т.д.
if (counter > 4) //Поскольку у нас всего 4 светодиода, вводим ограничение
чтобы переменная counter после значения 4 принимала заново значение 1
counter=1;
}

int main() {
//В "int_main" перед циклом "while" код, который выполняется только 1 раз,
когда плата только включилась
button.mode(PullDown); //Прерывание кнопки будет срабатывать при её
```

Отжати

button.fall(&presfunction); //Указывается какая именно  
подпрограмма/функция будет выполняться во время прерывания

```
while(1)
{
//Здесь указывается код в виде цикла, который выполняется
микроконтроллером
if (counter == 1)
{
orangeLed=!orangeLed; //Изменение состояния светодиода - вкл/выкл 0,1 с
thread_sleep_for(100); //Задержка
}
else if (counter == 2)
{
redLed=!redLed; //Изменение состояния светодиода - вкл/выкл 0,3 с
thread_sleep_for(300); //Задержка
}
else if (counter == 3)
{
blueLed=!blueLed; //Изменение состояния светодиода - вкл/выкл 0,5 с
thread_sleep_for(500); //Задержка
}
else if (counter == 4)
{
greenLed=!greenLed; //Изменение состояния светодиода - вкл/выкл 0,7 с
thread_sleep_for(700); //Задержка
}
}
}
```