

ГУАП

КАФЕДРА № 41

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ
ПРЕПОДАВАТЕЛЬ

доц., канд. техн. наук		О.А. Кононов
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №2

по курсу: ОСНОВЫ МИКРОПРОЦЕССОРНОЙ ТЕХНИКИ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. №	4711		Хасанов Б.Р.
		подпись, дата	инициалы, фамилия

Санкт-Петербург 2020

Цель работы

Изучить управление светодиодами при помощи нажатия кнопки на примере процессора STM32F407VG и написать программу, которая будет поочерёдно, друг за другом включать и выключать светодиоды по нажатию кнопки.

1 Теоретические сведения

Схема подключения кнопки на плате STM32F4Discovery представлена на рисунке 1.1

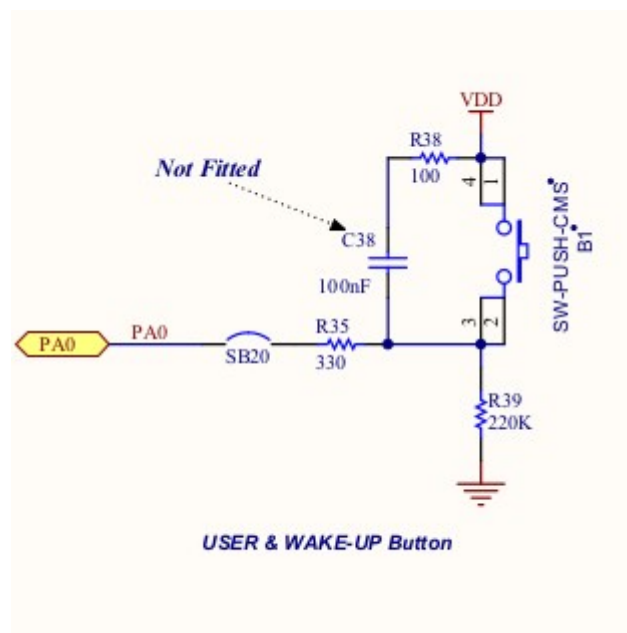


Рисунок 1.1 - Схема подключения светодиода на плате STM32F4 Discovery

Как видно, кнопка находится на нулевом выводе порта A. GPIO (General Purpose Input/Output) - самый простой и примитивный способ организации работы с внешними устройствами. Порты могут работать в двух режимах: вход (прием сигнала) и выход (передача сигнала). Работают они только с логическими уровнями 0 (Low) или 1 (Hight). Если включить вывод в режим входа, то с помощью микроконтроллера можно отслеживать ее состояние: нажатое или отжатое.

2 Практическая часть

В рамках работы была написана программа с помощью библиотеки CMSIS

Файл “main.c”

```
#include "main.h"
```

```
void Delay(volatile uint32_t nCount)
```

```
{  
    while(nCount--) {}  
}
```

```
int main(void){
```

```
    int led_switch_count = 0; //Переменная считающая количество  
нажатий кнопки
```

```
    LEDs_ini(); //Инициализация диодов
```

```
    BUTTON_ini(); //Инициализация кнопки
```

```
    while(1){
```

```
        // Фиксируем смену состояния кнопки
```

```
        if (READ_BIT(GPIOA->IDR, GPIO_IDR_IDR_0) !=  
switch_state){
```

```
            // Запоминаем его
```

```
            switch_state = READ_BIT(GPIOA->IDR, GPIO_IDR_IDR_0);
```

```
            if (READ_BIT(GPIOA->IDR, GPIO_IDR_IDR_0) == 1){
```

```
                led_switch_count++;
```

```
                switch(led_switch_count){ //На каждое нажатие  
зажигается свой диод
```

```
                    case 1:
```

```

CLEAR_REG(GPIOD->ODR); //очищаем
регистр выходных значений диодов
SET_BIT(GPIOD->ODR,
GPIO_ODR_ODR_12); //Зажигаем оранжевый диод
break;
case 2:
CLEAR_REG(GPIOD->ODR);
SET_BIT(GPIOD->ODR,
GPIO_ODR_ODR_13); // Зажигаем красный диод
break;
case 3:
CLEAR_REG(GPIOD->ODR);
SET_BIT(GPIOD->ODR,
GPIO_ODR_ODR_14); // Зажигаем синий диод
break;
case 4:
CLEAR_REG(GPIOD->ODR);
SET_BIT(GPIOD->ODR,
GPIO_ODR_ODR_15); // Зажигаем зелёный диод
led_switch_count = 0;
break;
}
}
Delay(90000); // Задержка нужна для того, чтобы не
происходило повторного срабатывания кнопки
}
}
}

```

Файл “main.h”

```
#include "init.h"
```

```
#ifndef MAIN_H
```

```
#define MAIN_H
```

```
//
```

```
#endif
```

Файл “init.c”

```
#include "init.h"
```

```
void LEDs_ini(void)
```

```
{
```

```
    RCC->AHB1ENR |= RCC_AHB1ENR_GPIODEN;
```

```
    GPIOD->MODER=          GPIO_MODER_MODER12_0      |  
GPIO_MODER_MODER13_0      |          GPIO_MODER_MODER14_0      |  
GPIO_MODER_MODER15_0;
```

```
}
```

```
void BUTTON_ini(void)
```

```
{
```

```
    RCC->AHB1ENR |= RCC_AHB1ENR_GPIOAEN;
```

```
    GPIOA->OSPEEDR |= GPIO_OSPEEDER_OSPEEDR0_0;
```

```
    GPIOA->PUPDR |= GPIO_PUPDR_PUPDR0_1;
```

```
}
```

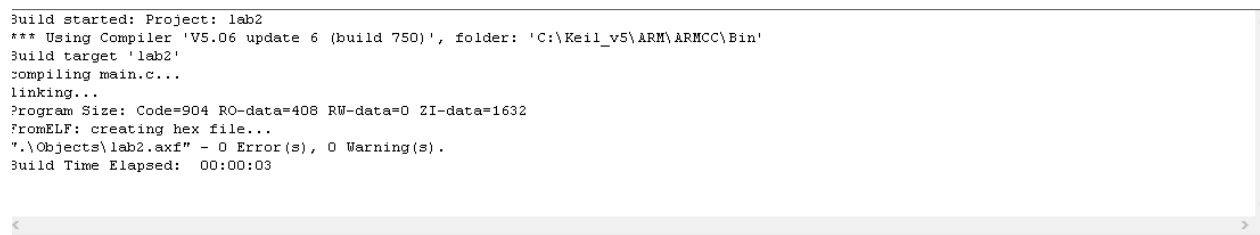
Файл “init.h”

```
#include "stm32f4xx.h"
```

```
void LEDs_ini(void);  
void BUTTON_ini(void);
```

3 Результаты работы программы.

Вывод компилятора показан на рисунке 3.1



```
Build started: Project: lab2  
*** Using Compiler 'V5.06 update 6 (build 750)', folder: 'C:\Keil_v5\ARM\ARMCC\Bin'  
Build target 'lab2'  
compiling main.c...  
linking...  
Program Size: Code=904 RO-data=408 RW-data=0 ZI-data=1632  
FromELF: creating hex file...  
".\Objects\lab2.axf" - 0 Error(s), 0 Warning(s).  
Build Time Elapsed: 00:00:03
```

Рисунок 3.1 – Компиляция программы (build output)

Вывод

В рамках данной работы ознакомился с режимом работы GPIO в режиме ввода, закрепил навыки работы с библиотекой CMSIS. Результат соответствует цели работы