

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 №5

по курсу: ОСНОВЫ МИКРОПРОЦЕССОРНОЙ ТЕХНИКИ

РАБОТУ ВЫПОЛНИЛ

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

Санкт-Петербург 2020

Цель работы

Изучить управление яркостью светодиодов с помощью кнопки на примере процессора STM32F407VG

1 Теоретические сведения

Модуляция – нелинейный электрический процесс, при котором параметры одного сигнала (несущего) изменяются при помощи другого сигнала (модулирующего, информационного). В связной технике широко применяется частотная, амплитудная, фазовая модуляция. В силовой электронике и микропроцессорной технике распространение получила широтно-импульсная модуляция.

При широтно-импульсной модуляции исходного сигнала неизменными остаются амплитуда, частота и фаза исходного сигнала. Изменению под действием информационного сигнала подвергается длительность (ширина) прямоугольного импульса. В англоязычной технической литературе обозначается аббревиатурой PWM – pulse-width modulation.

Сигнал, промодулированный по ширине импульса, формируется двумя способами:

- аналоговым;
- цифровым.

При аналоговом способе создания ШИМ-сигнала несущая в виде пилообразного или треугольного сигнала подается на инвертирующий вход компаратора, а информационный – на неинвертирующий. Если мгновенный уровень несущей выше модулирующего сигнала, то на выходе компаратора ноль, если ниже – единица. На выходе получается дискретный сигнал с частотой, соответствующей частоте несущего треугольника или пилы, и длиной импульса, пропорциональной уровню модулирующего напряжения.

В качестве примера на рисунке 1 приведена модуляция по ширине импульса треугольного сигнала линейно-возрастающим. Длительность

выходных импульсов пропорциональна уровню выходного сигнала.

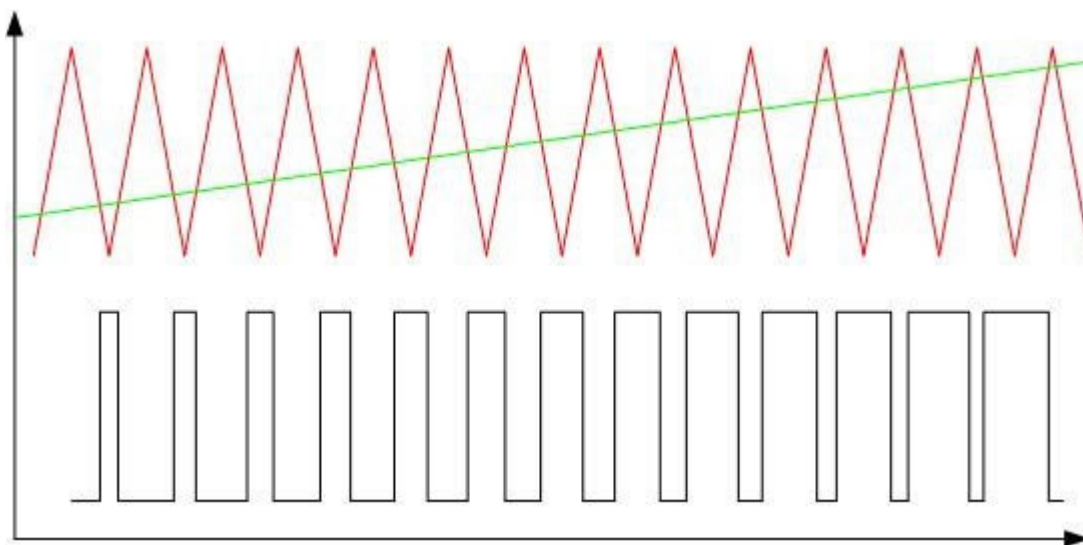


Рисунок 1 – Модуляция по ширине импульса

Аналоговые ШИМ-контроллеры выпускаются и в виде готовых микросхем, внутри которых установлен компаратор и схема генерации несущей. Имеются входы для подключения внешних частото задающих элементов и подачи информационного сигнала. С выхода снимается сигнал, управляющий мощными внешними ключами. Также имеются входы для обратной связи – они нужны для поддержания установленных параметров регулирования. Такова, например, микросхема TL494. Для случаев, когда мощность потребителя относительно невелика, выпускаются ШИМ-контроллеры со встроенными ключами. На ток до 3 ампер рассчитан внутренний ключ микросхемы LM2596.

Цифровой способ осуществляется применением специализированных микросхем или микропроцессоров. Длина импульса регулируется внутренней программой. Во многих микроконтроллерах, включая популярные PIC и AVR, «на борту» имеется встроенный модуль для аппаратной реализации ШИМ, для получения PWM-сигнала надо активировать модуль и задать параметры его работы. Если такой модуль отсутствует, то ШИМ можно организовать чисто программным методом, это несложно. Этот способ дает более широкие возможности и предоставляет больше свободы за счёт гибкого использования выходов, но задействует большее количество

ресурсов контроллера.

2 Практическая часть

В рамках работы была написана программа с помощью библиотеки CMSIS

Файл “main.c”

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

```
int      duty_cycles_array      [NUMBER_OF_DUTY_CYCLES]      =  
{0x2,0x4,0x8,0x10,0x20,0x40,0x80,0xFF};    //    Массив    коэффициентов  
заполнения
```

```
void Generate_pwm(int number_of_periods, int pwm_duty_cycle){  
    int pwm_period_number;  
    int impulse_time;  
    int pause_time;  
  
    for(pwm_period_number = 0; pwm_period_number < number_of_periods;  
pwm_period_number++){ // Количество периодов ШИМ  
        for (impulse_time = 0; impulse_time <= pwm_duty_cycle;  
impulse_time++){ //Импульс  
            // Включаем все диоды  
            SET_BIT(GPIOD->ODR, GPIO_ODR_ODR_12);  
            SET_BIT(GPIOD->ODR, GPIO_ODR_ODR_13);  
            SET_BIT(GPIOD->ODR, GPIO_ODR_ODR_14);  
            SET_BIT(GPIOD->ODR, GPIO_ODR_ODR_15);  
        }  
        // Пауза. Выключаем все диоды  
        for      (pause_time      =      0;      pause_time      <=  
duty_cycles_array[NUMBER_OF_DUTY_CYCLES - 1] - pwm_duty_cycle;  
pause_time++) CLEAR_REG(GPIOD->ODR);  
    }  
}
```

```
int main(void){  
    int duty_cycle_counter;  
    int current_duty_cycle = 0;  
  
    LEDs_ini();  
    BUTTON_ini();  
    while(1){
```

```

        if (READ_BIT(GPIOA->IDR, GPIO_IDR_IDR_0) == 1){ // Если
кнопка нажата
            duty_cycle_counter++;
            current_duty_cycle = duty_cycles_array[duty_cycle_counter %
8]; // Меняем коэффициентам заполнения
            Generate_pwm(4000, current_duty_cycle); // Запускаем
генерацию 4000 периодов ШИМ, чтобы подождать, пока не пройдет дребезг
на кнопке
        }
        Generate_pwm(1, current_duty_cycle); // Генерируем ШИМ пока не
нажмут кнопку
    }
}

```

Файл “main.h”

```

#include "init.h"

#define NUMBER_OF_DUTY_CYCLES 8
#define PWM_PERIOD 5000

#ifndef MAIN_H
#define MAIN_H
//
#endif

```

Файл “init.c”

```

#include "init.h"

void LEDs_ini(void)
{
    SET_BIT(RCC->AHB1ENR, RCC_AHB1ENR_GPIODEN);
    SET_BIT(GPIOD->MODER, GPIO_MODER_MODER12_0 |
GPIO_MODER_MODER13_0 | GPIO_MODER_MODER14_0 |
GPIO_MODER_MODER15_0);
}

void BUTTON_ini(void)
{
    SET_BIT(RCC->AHB1ENR, RCC_AHB1ENR_GPIOAEN);
    SET_BIT(GPIOA->OSPEEDR, GPIO_OSPEEDER_OSPEEDR0_0);
    SET_BIT(GPIOA->PUPDR, GPIO_PUPDR_PUPDR0_1);
}

```

```
}
```

Файл “init.h”

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

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

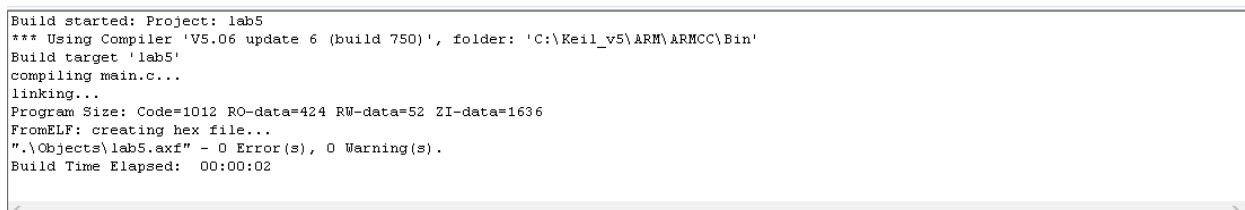
```
void BUTTON_ini(void);
```

3 Результаты работы программы.

Вывод компилятора показан на рисунке 2

Работу программы можно увидеть по ссылке

<https://imgur.com/a/5oI61z2>



```
Build started: Project: lab5
*** Using Compiler 'V5.06 update 6 (build 750)', folder: 'C:\Keil_v5\ARM\ARMCC\Bin'
Build target 'lab5'
compiling main.c...
linking...
Program Size: Code=1012 RO-data=424 RW-data=52 ZI-data=1636
FromELF: creating hex file...
".\Objects\lab5.axf" - 0 Error(s), 0 Warning(s).
Build Time Elapsed: 00:00:02
```

Рисунок 2 – Компиляция программы (build output)

Вывод

В рамках данной лабораторной работы мной был написан код на языке C, с использованием библиотеки CMSIS, для stm32f4, который заставляет диоды менять яркость при нажатии кнопки. По нажатию кнопки меняется уровень скважности ШИМ, что приводит к изменению яркости светодиода. Смена яркости реализована с помощью широтное-импульсной модуляции