Факультет программной инженерии и компьютерной техники

Встроенные системы

Лабораторная №2 «Программирование дискретных портов ввода/вывода»

Вариант №1

Преподаватели: Ключев Аркадий Олегович, Быковский Сергей Вячеславович

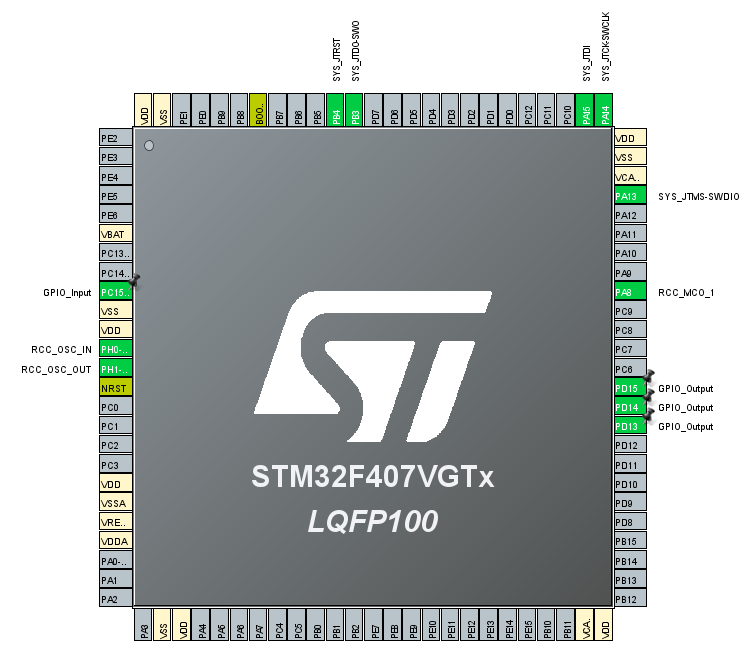
Выполнил: Кульбако Артемий Юрьевич

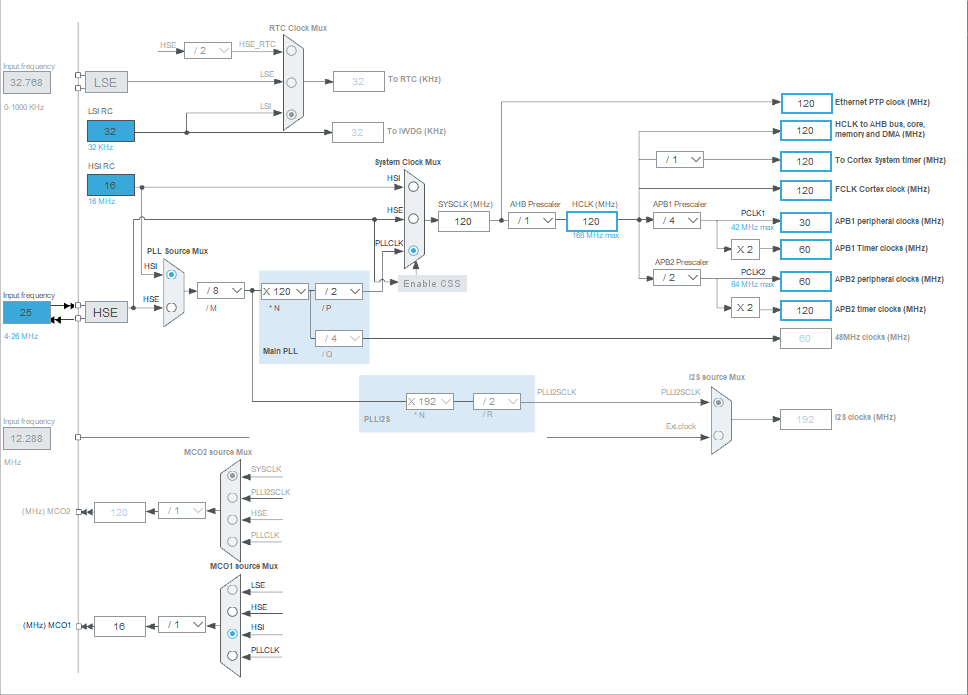
Р3212

# Описание инструментария

Разработка программного обеспечения велась в среде разработки на STM32CubeIDE, основанной на базе IDE с открытым кодом Eclipse специально для серии микропроцессоров STM. Необходимо создать STM32 Project на C/C++, скачать SDK для вашей платы, и открыть файл Core -> Src -> main.c, в котором можно начать редактировать код. Предварительно нужно установить драйвер WinUSB для дебага контроллера через универсальную последовательную шину, с помощью утилиты Zadig. После всех этих действий можно приступать к написанию кода.

**Интерфейс пинов**

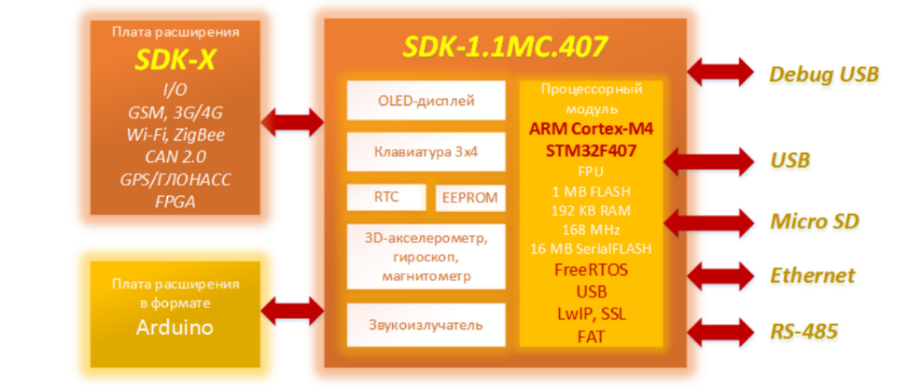


**Схема тактирования**

Аппаратная платформа - SDK-1.1M - стенд-конструктор, состоящий из платы ввода-вывода и заменяемого процессорного модуля. В настоящий момент стенд поставляется с ARM микроконтроллером STM32F407.

Основные характеристики SDK-1.1MC.407:

* Микропроцессор STM32F407VGT6;
* Внешняя EEPROM объемом 1 Кбит;
* Часы реального времени MCP79411;
* Графический OLED дисплей WEO012864DL фирмы Winstar;
* Интерфейс ввода/вывода общего назначения (GPIO) PCA9538PW;
* Инерционный модуль iNEMO LSM9DS1;
* Электромагнитный излучатель звука HC0903A;
* Набор сигнальных светодиодов (зеленый, желтый, красный);
* Клавиатура из 12 кнопок;
* RS-485;
* Ethernet 10/100;
* USB.

**Схема платформы**

# Код

**while** (1) {

HAL\_GPIO\_WritePin(GPIOD, GPIO\_PIN\_13, *GPIO\_PIN\_SET*);

HAL\_Delay(3000);

HAL\_GPIO\_WritePin(GPIOD, GPIO\_PIN\_13, *GPIO\_PIN\_RESET*);

**for** (**int** i = 0; i < 3; i++) {

HAL\_Delay(500);

HAL\_GPIO\_WritePin(GPIOD, GPIO\_PIN\_13, *GPIO\_PIN\_SET*);

HAL\_Delay(500);

HAL\_GPIO\_WritePin(GPIOD, GPIO\_PIN\_13, *GPIO\_PIN\_RESET*);

}

HAL\_GPIO\_WritePin(GPIOD, GPIO\_PIN\_14, *GPIO\_PIN\_SET*);

HAL\_Delay(2000);

HAL\_GPIO\_WritePin(GPIOD, GPIO\_PIN\_14, *GPIO\_PIN\_RESET*);

HAL\_GPIO\_WritePin(GPIOD, GPIO\_PIN\_15, *GPIO\_PIN\_SET*);

**for** (**int** i = 0; i < 5000; i++) {

**if** (HAL\_GPIO\_ReadPin(GPIOC, GPIO\_PIN\_15) == 0) {

HAL\_Delay(1250);

**break**;

}

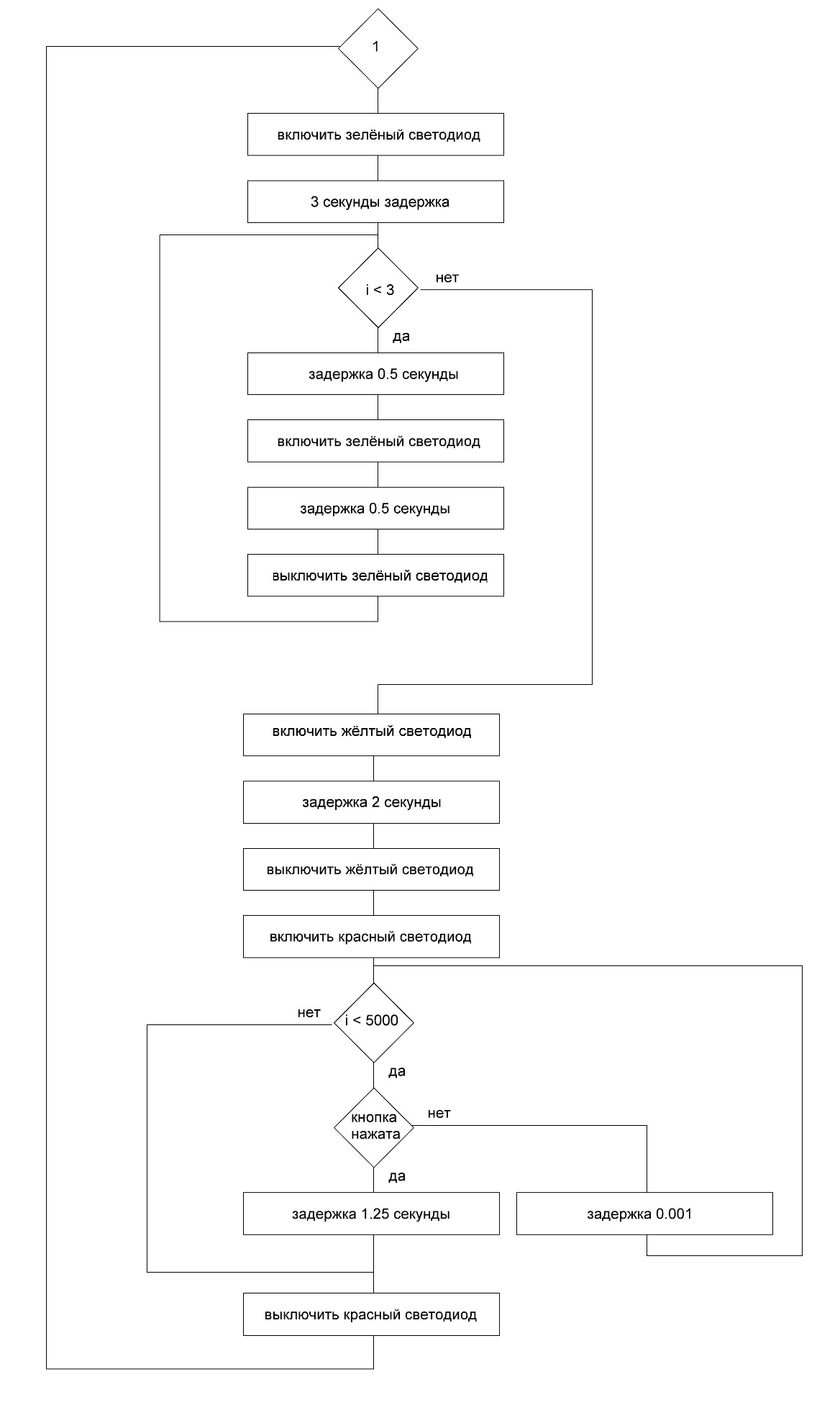
**else** HAL\_Delay(1);

}

HAL\_GPIO\_WritePin(GPIOD, GPIO\_PIN\_15, *GPIO\_PIN\_RESET*);

}

# Блок-схема



# Задание

Сымитировать работу светофора пешеходного перехода. В режиме по умолчанию светофор переключает цвета в следующем режиме: зелёный - мигающий зелёный – жёлтый – красный - зелёный…, при этом период красного значительно больше. При нажатии кнопки происходит переключение с красного на зелёный, но два включения «зелёных» не могут идти сразу друг за другом – между ними должен быть период, больше или равный ¼ периода красного.

# Выводы

В процессе работы я столкнулся с двумя ошибками, из-за которых не получалось загрузить программу в микроконтроллер. Первая решилась переустановкой специализированного драйвера WinUSB, а вторая перемещением проекта в директорию, которая не содержала кириллицы в названии.

На базе SDK-1.1M я создал управляемый светофор. Пин PC15 был настроен на Input, для получений состояния нажатия кнопки. В бесконечном цикле работы платы поочерёдно переключались режимы работы светодиодов (PD13-PD15) в порядке зелёный, зелёный мигающий, жёлтый, красный. Во время работы красного светодиода, я получал значение с PC15, и прерывал текущий цикл работы красного светодиода, если кнопка нажата.

Я научился работать с функциями:

* HAL\_GPIO\_WritePin(GPIOx, GPIO\_PIN\_NUMBER, GPIO\_PIN\_STATUS) – изменить состояние пина
* HAL\_GPIO\_ReadPin(GPIOx, GPIO\_PIN\_NUMBER) – получить состояние пина
* HAL\_Delay(milliseconds) – задержка между командами