УНИВЕРСИТЕТ ИТМО

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

P3212

Санкт-Петербург
2020

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

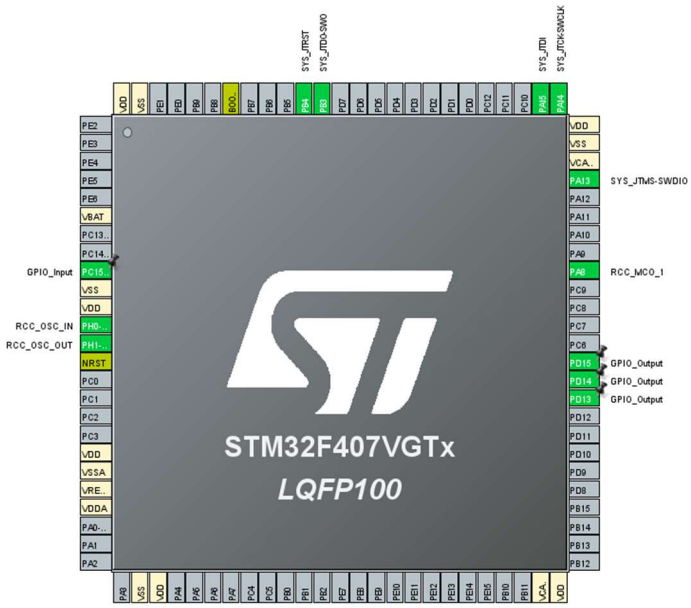
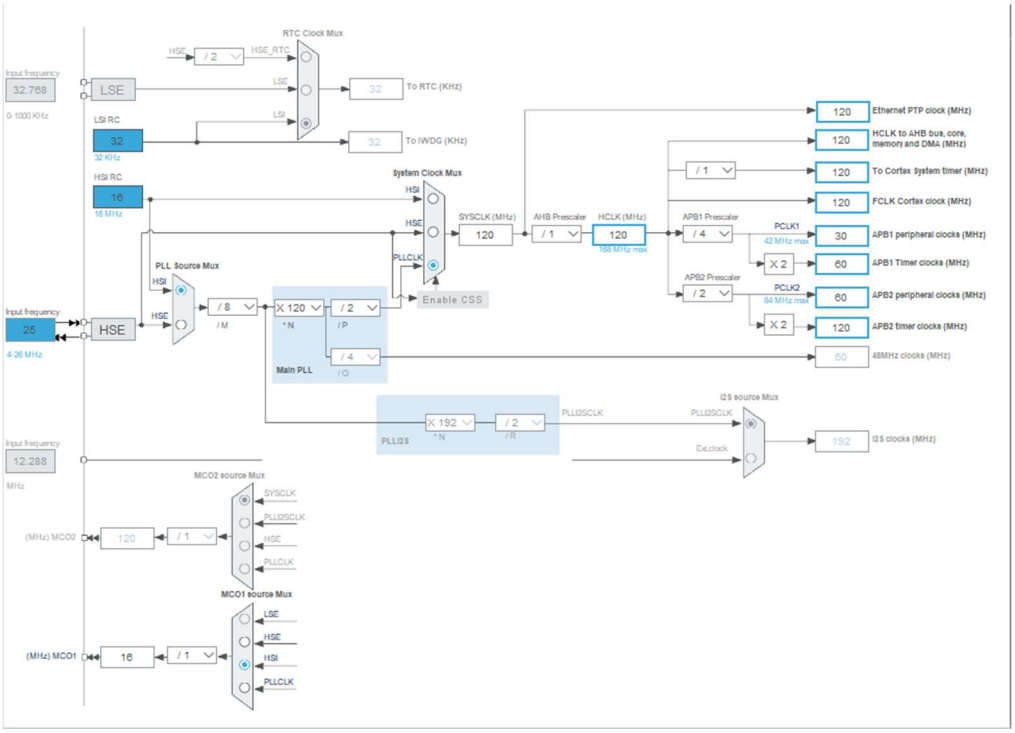


Схема тактирования

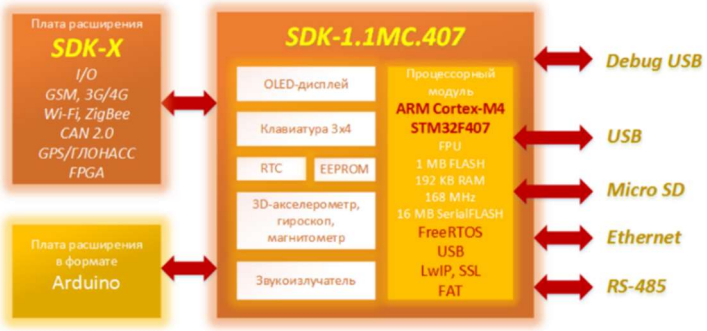


Аппаратная платформа - SDK-1.1M - стенд-конструктор, состоящий из платы ввода-вывода и заменяемого процессорного модуля. В настоящий момент стенд поставляется с ARM микроконтроллером STM32F407.

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.

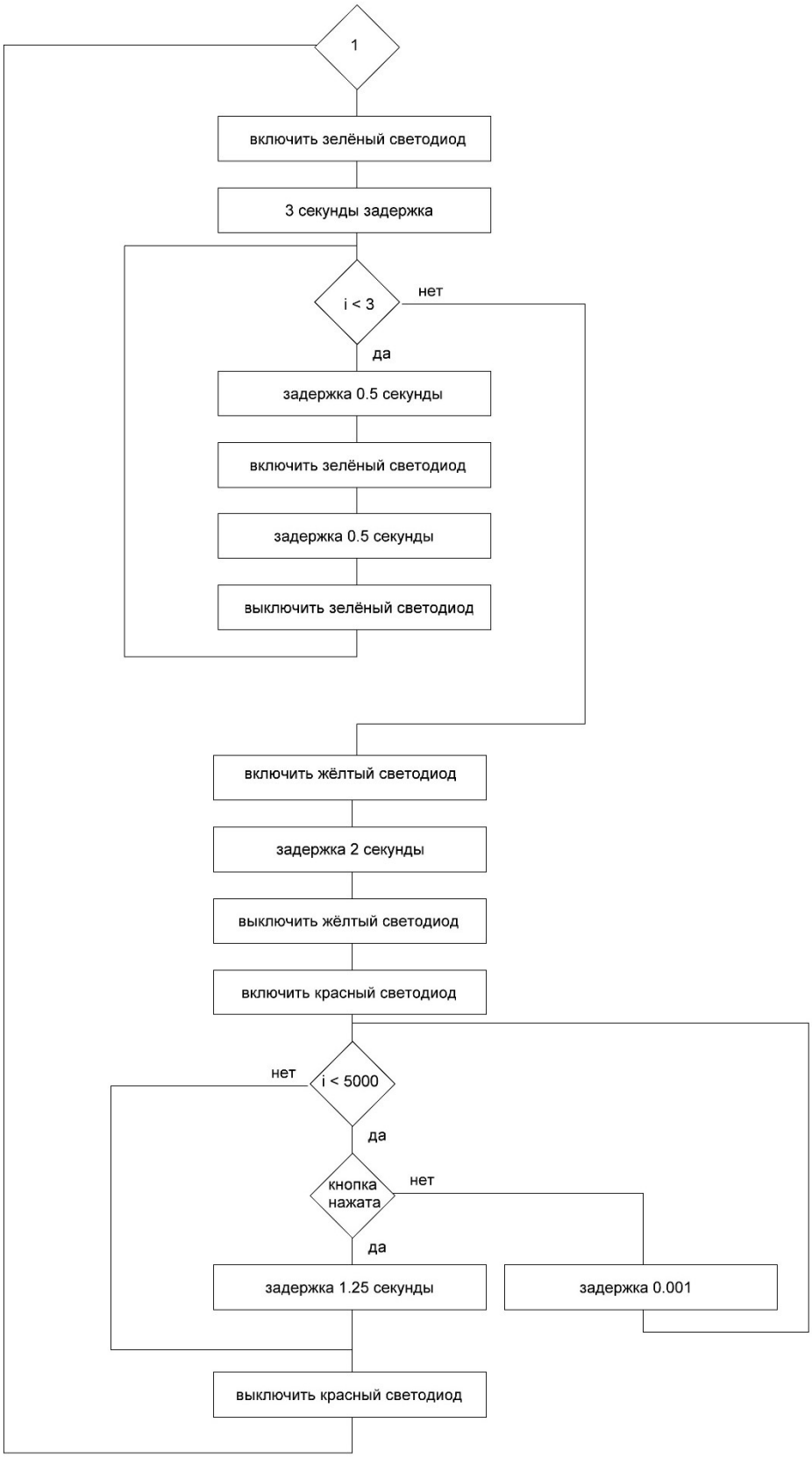
Схема платформы



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


1
включить зелёный светодиод
3 секунды задержка
i < 3
нет
да
задержка 0.5 секунды
включить зелёный светодиод
задержка 0.5 секунды
выключить зелёный светодиод
включить жёлтый светодиод
задержка 2 секунды
выключить жёлтый светодиод
включить красный светодиод
нет
i < 5000
да
кнопка нажата
нет
да
задержка 1.25 секунды
задержка 0.001
выключить красный светодиод

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