Федеральное государственное автономное образовательное
учреждение высшего образования
«Национальный исследовательский университет ИТМО»

Дисциплина: Проектирование вычислительных систем

Лабораторная работа 1

Вариант 4

Выполнили:
Барсуков Максим Андреевич,
группа P3415
Стригалев Никита Сергеевич,
группа P3412

Преподаватель:
Пинкевич Василий Юрьевич

2025 г.

Санкт-Петербург

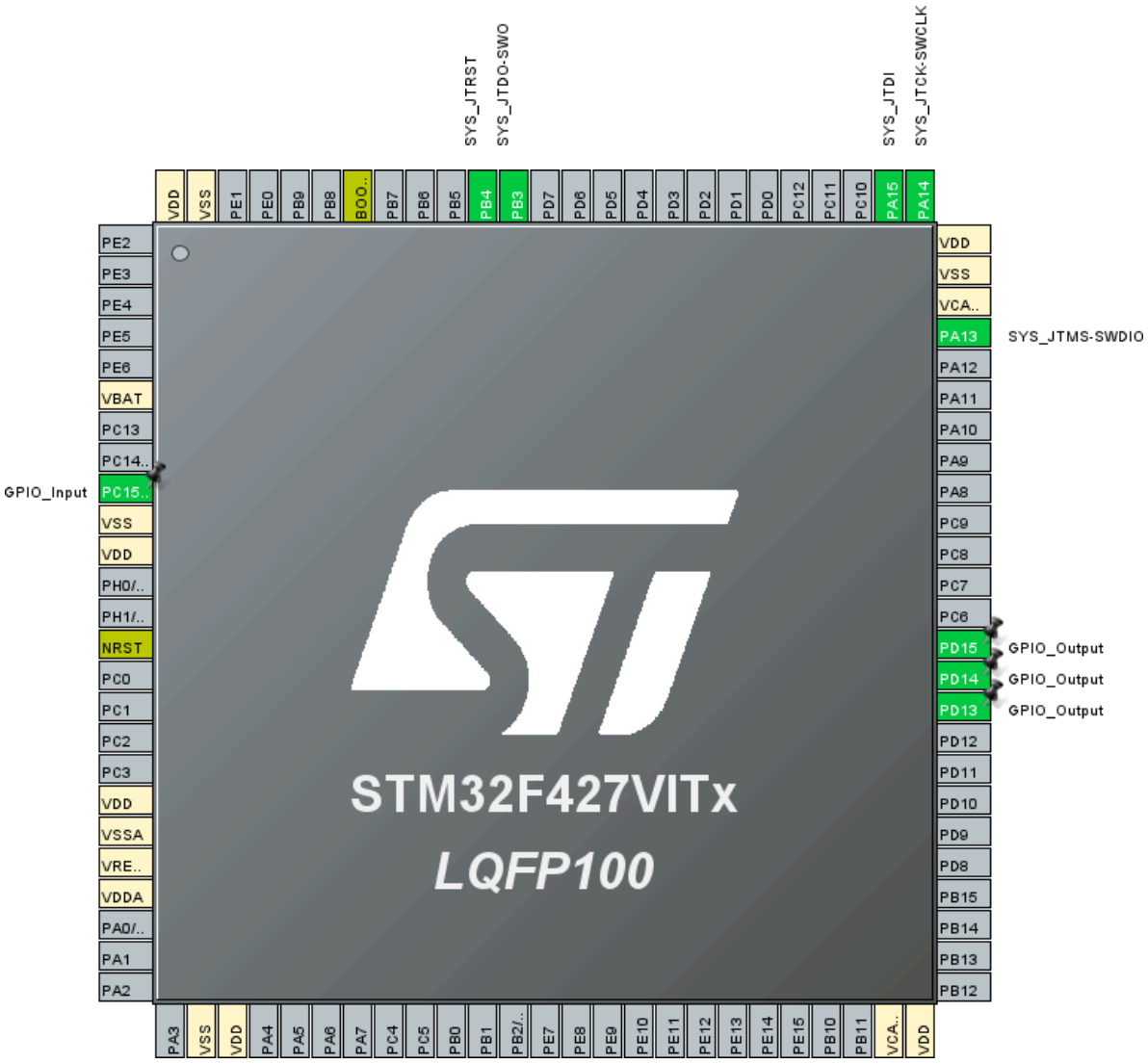
Содержание

Задание

Разработать и реализовать драйверы управления светодиодными индикаторами и чтения состояния кнопки стенда SDK-1.1M (индикаторы и кнопка расположены на боковой панели стенда). Написать программу с использованием разработанных драйверов в соответствии с вариантом задания.

Реализовать двоичный двухразрядный счетчик на светодиодах с возможностью вычитания (использовать зеленый светодиод и один из двух цветов двухцветного). Быстрое нажатие кнопки должно прибавлять единицу к отображаемому на светодиодах двоичному числу. По переполнению счетчика должна отображаться простая анимация: мигание обоими светодиодами, затем количество миганий зеленым светодиодом, равное количеству переполнений с момента перезагрузки микроконтроллера. Долгое нажатие кнопки должно вычитать единицу из отображаемого на светодиодах двоичного числа. Если происходит вычитание из нуля, количество переполнений уменьшается на единицу, и отображается анимация, аналогичная анимации при переполнении.

Используемые контакты

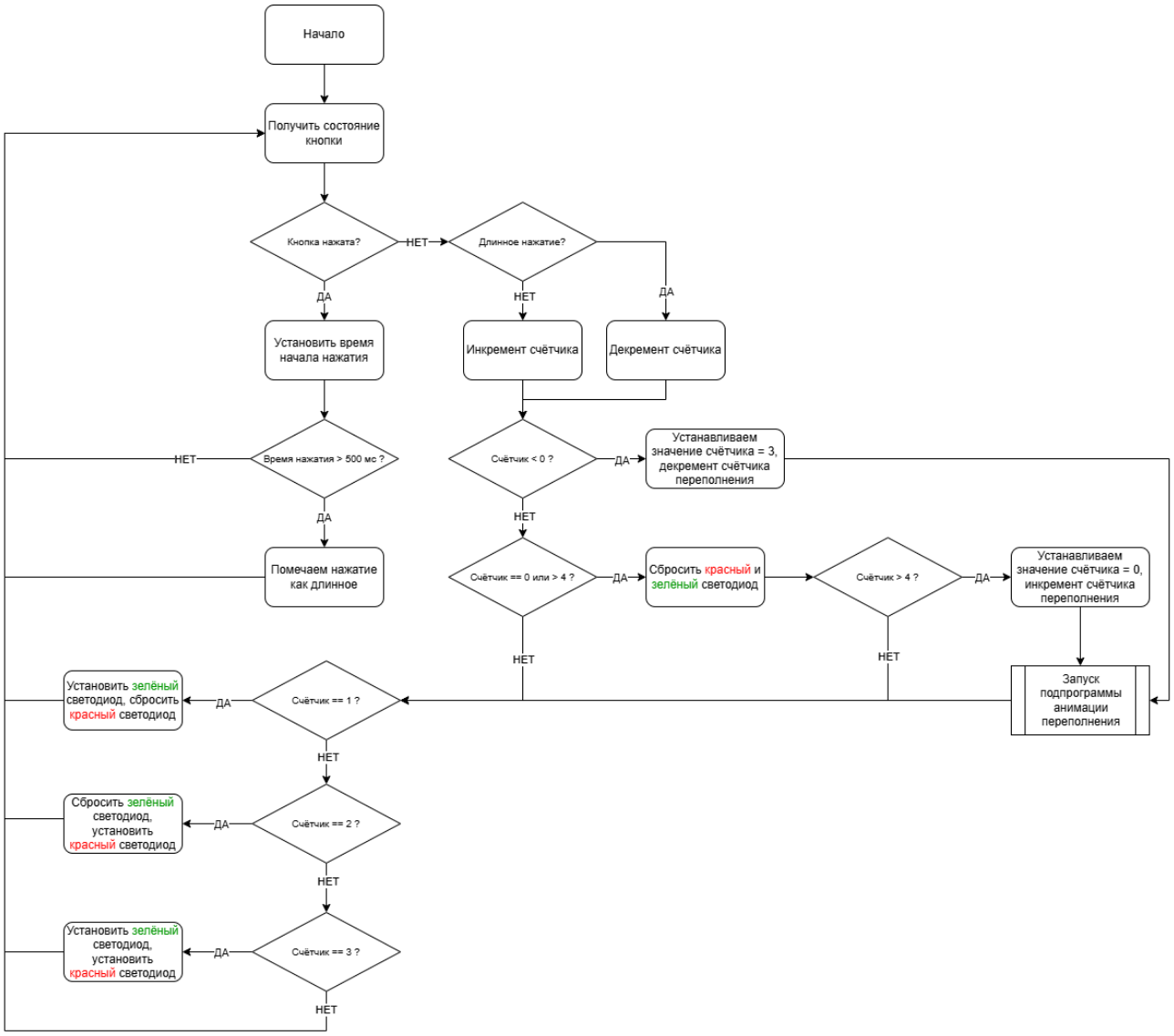


Описание контактов

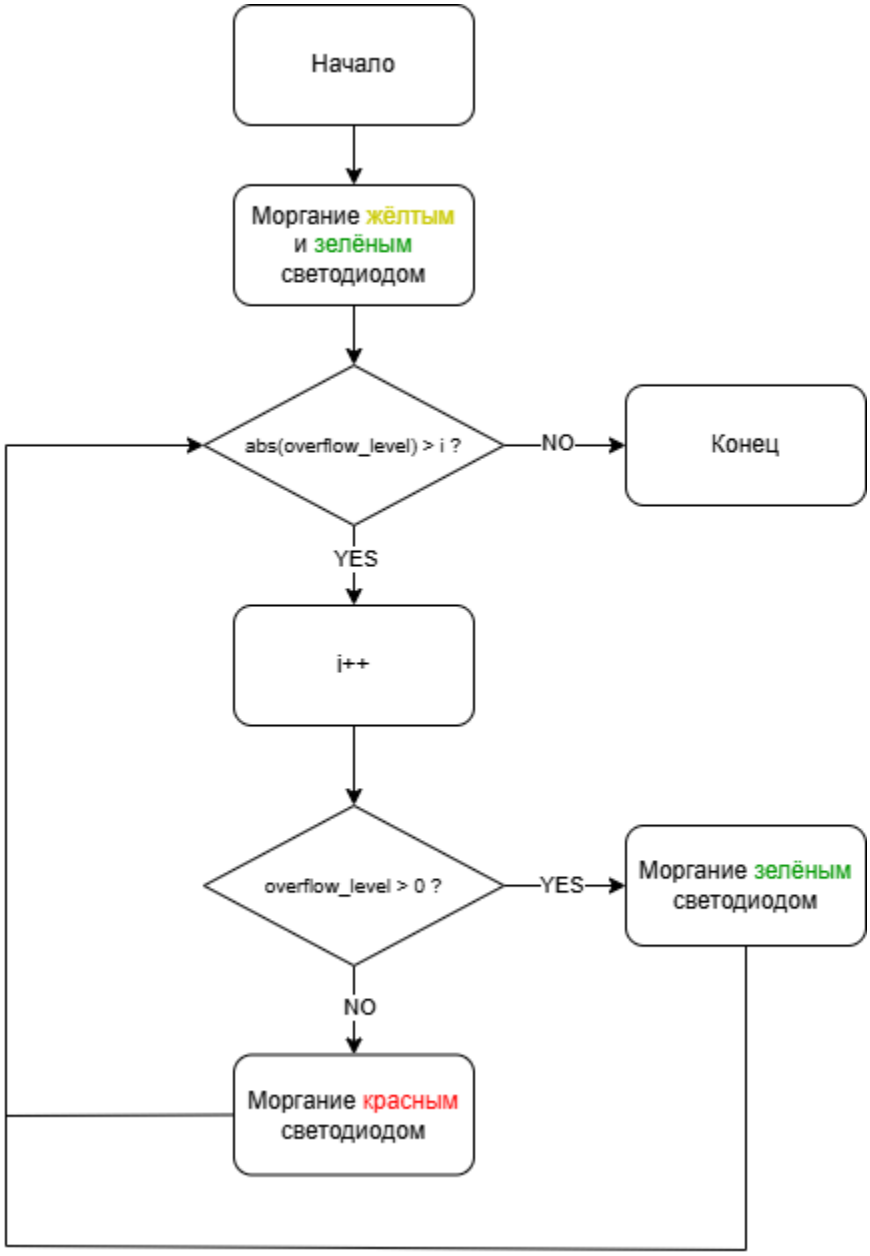
- **PC15** - перехватывает нажатие кнопки (настроен на GPIO_Input)
- **PD13** - управляет зеленым светодиодом (настроен на GPIO_Output)
- **PD14** - управляет желтым светодиодом (настроен на GPIO_Output)
- **PD15** - управляет красным светодиодом (настроен на GPIO_Output)
- **PB3, PB4, PA13, PA14, PA15** - J-Tag для отладки

Блок-схемы

Схема алгоритма:



Подпрограмма анимации переполнения:



Описание работы алгоритма

Алгоритм работает в цикле, постоянно отслеживая состояние кнопки. При обнаружении нажатия система регистрирует этот факт и измеряет его длительность. Различаются два типа нажатий: короткое и долгое. В момент отпускания кнопки анализируется тип произошедшего нажатия: краткое нажатие увеличивает значение счётчика, а долгое — уменьшает его.

Счётчик ограничен диапазоном от 0 до 3, так как для отображения доступны только две лампочки. При попытке выйти за эти границы (например, при значении 4 или -1) возникает переполнение. В этом случае активируется специальная анимация, которая сигнализирует о выходе за пределы диапазона: зелёный светодиод указывает на переполнение "сверху", а красный — "снизу".

Код драйвера

```
void turn_off_leds() {
    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13, GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_14, GPIO_PIN_RESET);
    HAL_Delay(150);
}

void pulse_dual_leds() {
    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13, GPIO_PIN_SET);
    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_14, GPIO_PIN_SET);
    HAL_Delay(150);
    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13, GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_14, GPIO_PIN_RESET);
    HAL_Delay(150);
}

void pulse_single_led() {
    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13, GPIO_PIN_SET);
    HAL_Delay(150);
    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13, GPIO_PIN_RESET);
    HAL_Delay(150);
}

void show_overflow_feedback(int times) {
    turn_off_leds();
    pulse_dual_leds();
    for (int i = 0; i < times; i++) {
         pulse_single_led();
    }
}

int main(void)
{
    int led_state = 0;
    int overflow_level = 0;
    const int press_threshold = 500;

  HAL_Init();

  SystemClock_Config();

  while (1)
    {
      int button_is_down = !HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_15);
      int tick_start = 0;
      if (button_is_down) {
```

```
            tick_start = HAL_GetTick();
            while (!HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_15));
            int press_time = HAL_GetTick() - tick_start;
            if (press_time < press_threshold) {
                  led_state++;
                  if (led_state > 3) {
                        led_state = 0;
                        overflow_level++;
                        show_overflow_feedback(overflow_level);
                  }
            } else {
                  led_state--;
                  if (led_state < 0) {
                        led_state = 3;
                        overflow_level--;
                        if (overflow_level < 0) {
                              overflow_level = 0;
                        } else {
                              show_overflow_feedback(overflow_level);
                        }
                  }
            }
      }

      switch (led_state) {
            case 0:
                  HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13,
GPIO_PIN_RESET);
                  HAL_GPIO_WritePin(GPIOD, GPIO_PIN_14,
GPIO_PIN_RESET);
                  break;
            case 1:
                  HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13, GPIO_PIN_SET);
                  HAL_GPIO_WritePin(GPIOD, GPIO_PIN_14,
GPIO_PIN_RESET);
                  break;
            case 2:
                  HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13,
GPIO_PIN_RESET);
                  HAL_GPIO_WritePin(GPIOD, GPIO_PIN_14, GPIO_PIN_SET);
                  break;
            case 3:
                  HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13, GPIO_PIN_SET);
                  HAL_GPIO_WritePin(GPIOD, GPIO_PIN_14, GPIO_PIN_SET);
                  break;
      }
    }
}
```

Вывод

В результате выполнения лабораторной работы были освоены принципы работы с портами ввода-вывода общего назначения (GPIO). На практике эти знания были применены для разработки и программирования цифрового счётчика, использующего два светодиода в качестве индикатора состояния.