

# Введение в микроконтроллеры

Домашнее задание №6

Евгений Зеленин

30 января 2025 г.

## 1. Постановка задачи

**Условия задачи.** Реализуйте простой автомат освещения (скажем, включение светодиода при затемнении фотодатчика). Предусмотрите какой-то гистерезис включения/выключения; выведите в плагин Arduino IDE графики освещённости, порога и состояния светодиода (можно просто строкой вида "%d %d %d\n" через интерфейс UART или USB).

\* сделайте порог срабатывания настраиваемым при помощи потенциометра (т. е. понадобится опрашивать сразу два канала);

## 2. Описание устройства

Автомат освещения работает следующим образом: фотодатчик через делитель подключен к входу PA0 АЦП. При изменении уровня освещенности изменяется сопротивление датчика и, следовательно, уровень напряжения на порте PA0. К портам PA1 и PA2 АЦП подключены потенциометры, с помощью них можно регулировать верхнюю и нижнюю границу петли гистерезиса. Потенциометр PA1 отвечает за нижнюю границу, а PA2 - за верхнюю. Когда уровень на фотодатчике падает ниже границы, установленной PA1 - зажигается светодиод LED (PB0), а когда превышает границу PA2 - светодиод тухнет. На рисунке 1 показано назначение портов в CubeMX, а на рисунке 2 - схема подключений.

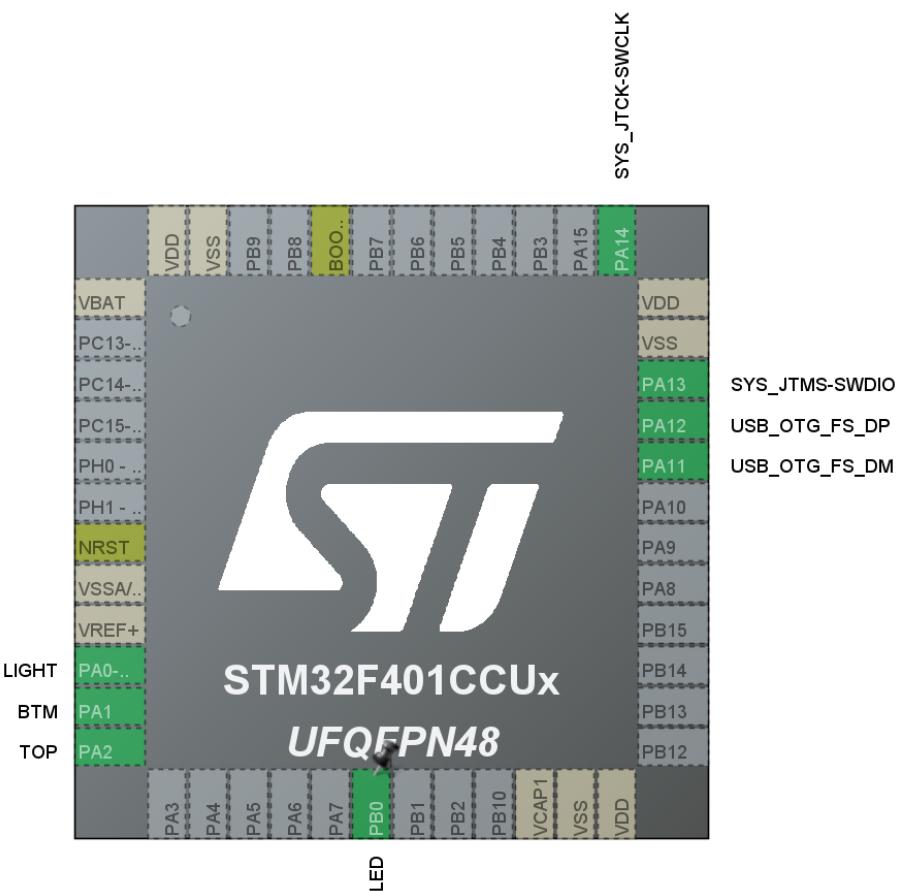


Рис. 1: Назначение портов

Дополнительно, данные с АЦП выводятся через виртуальный COM-port в терминал ПК.

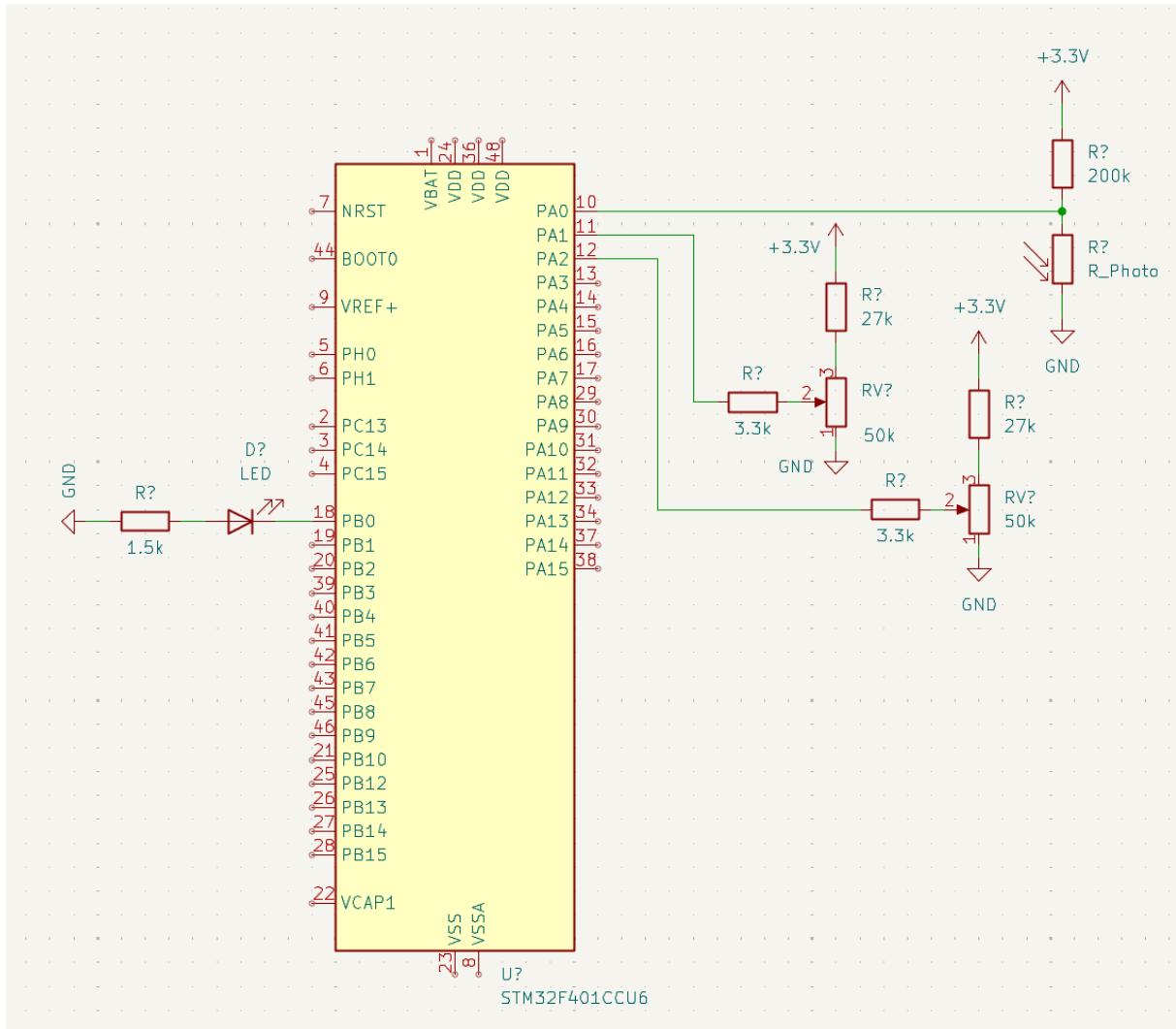


Рис. 2: Схема подключений

### 3. Исходный код проекта

Ниже можно ознакомиться фрагментами исходного кода, отвечающими за обработку АЦП. Первый вариант - с использованием прерываний.

**while(1)**

```
{
    uint32_t adc[3] = { 0, };
    for (int i = 0; i < 3; i++) {
        HAL_ADC_Start_IT(&hadc1);
        while (!is_adc)
            // waiting for data
        ;
        is_adc = 0;
        adc[i] = adc_val;
    }
}
```

```

//This will surely stop and reset ADC
    HAL_ADC_Stop_IT(&hadc1);

sprintf((char*) obuf, "%4lu %4lu %4lu\n", adc[0], adc[1], adc[2]);
CDC_Transmit_FS(obuf, strlen((char*) obuf));

//adc[1] - bottom thresold
//adc[2] - top thresold
if (adc[0] < adc[1]) {
    HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, GPIO_PIN_SET);
} else if (adc[0] > adc[2]) {
    HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, GPIO_PIN_RESET);
}
HAL_Delay(100);
}

void HAL_ADC_ConvCpltCallback(ADC_HandleTypeDef *hadc) {
    //copy ADC value to user variable
    adc_val = HAL_ADC_GetValue(&hadc1);
    is_adc = 1;
}

```

**Альтернативный вариант обработки нескольких каналов через Start/Poll**

```

{
    for (int i = 0; i < 3; i++) {
        HAL_ADC_Start(&hadc1);
        HAL_ADC_PollForConversion(&hadc1, 10);
        adc[i] = HAL_ADC_GetValue(&hadc1);
    }
    HAL_ADC_Stop(&hadc1);
}

```

Исходные коды проекта приложены к дополнительным материалам отчета.

#### 4. Демонстрация работы

На рисунке 3 показана демонстрация работы: вывод в консоль данных АЦП и статуса свето-диода. Как видно из иллюстрации, светодиод 1 загорается и тухнет при пересечении нижнего или верхнего порогов срабатывания.

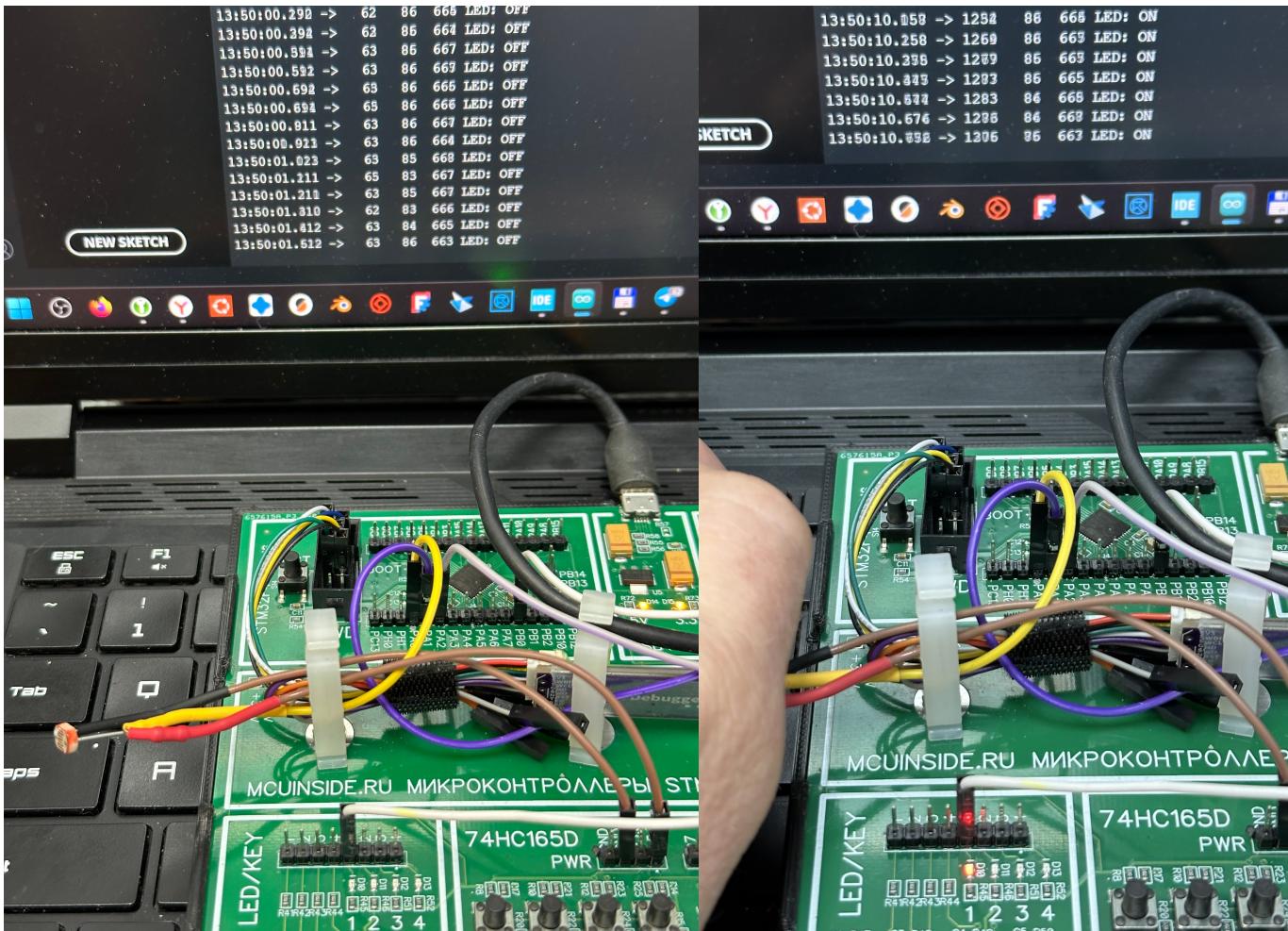


Рис. 3: Демонстрация работы

## 5. Заключение

В ходе выполнения работы получилось задействовать три канала АЦП двумя способами: через прерывания и через Start/Poll. Для обработки нескольких каналов без DMA целесообразно использовать режим Scan Conversion совместно с Discontinuous Conversion.

## 6. Дополнительные материалы

Материалы к отчету расположены в папке на google диск по следующей ссылке: Материалы к ДЗ №06