МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**Национальный исследовательский ядерный университет «МИФИ»**



**Институт интеллектуальных кибернетических систем**

**КАФЕДРА КИБЕРНЕТИКИ (№22)**

Направление подготовки 09.03.04 Программная инженерия

|  |
| --- |
| **Курсовая работа** |
| по дисциплине «Микропроцессорные системы»  на тему «Реализация управления диодом через терминал» |

Выполнил:

студент группы Б20-514

Юмагулов М. И.

Преподаватель: Кононов В.М.

**Москва 2022**

Содержание

[Содержание 2](#_Toc123081794)

[1. Постановка задачи 3](#_Toc123081795)

[2. Теоретическая информация 4](#_Toc123081796)

[3. Реализация 6](#_Toc123081797)

[4. Ссылка на реализацию 9](#_Toc123081798)

[5. Заключение 9](#_Toc123081799)

1. Постановка задачи

Написать программу для управления светодиодом через терминал на базе микроконтроллера STM32F401CDU6. Терминал дает следующие возможности при работе со светодиодом: включить, выключить, моргнуть заданное число раз, моргнуть заданное число раз с заданной пользователем периодичностью, завершить работу программы. Работа выполнялась в программе STM32CubeIDE на языке программирования C с использованием интерфейса USART и библиотеки HAL.

2. Теоретическая информация

Микроконтроллер— микросхема, предназначенная для управления электронными устройствами. Микроконтроллер сочетает на одном кристалле функции процессора и периферийных устройств, содержит ОЗУ и (или) ПЗУ.

Микроконтроллер семейства STM32F401СD построен на базе ядра Cortex-M4. В Programming Manual для платы на официальном сайте STM [1] находится описание данного ядра:

Процессор Cortex-M4 — процессор, предназначенный для рынка микроконтроллеров и микропроцессоров. Он предлагает значительные преимущества для разработчиков, в том числе: хорошая производительность, усовершенствованная отладка системы. эффективное ядро процессора, система и память, сверхнизкое энергопотребление и безопасность платформы.

Arm® Cortex®-M4, способен работать на частоте до 168 МГц. Ядро Cortex-M4 оснащено модулем с плавающей запятой (FPU) одинарной точности, который поддерживает все инструкции и типы данных обработки данных Arm с одинарной точностью.

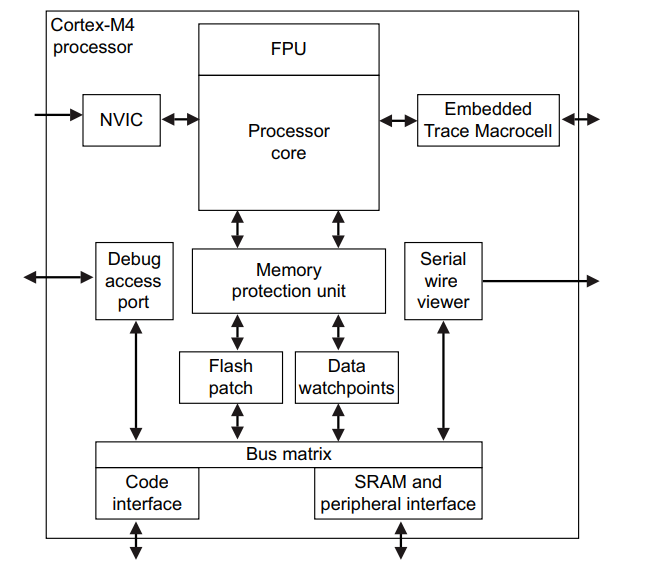


рис. 1 Реализация STM32 Cortex-M4 [1]

В STM32 для некоторого устройства ассоциирован определенный участок памяти. Записывая в эти участки «0» или «1» можно управлять определенной периферией. При работе на Cortex-M4 c адреса 0х8000000 начинается Flash-память - это та память, которая будет использована для прошивки.

На плате присутствует 2 светодиода, один из которых используется как индикатор питания. Соответственно, остается только 1 доступный светодиод. Он подключен к пину PC13 (рис.3).

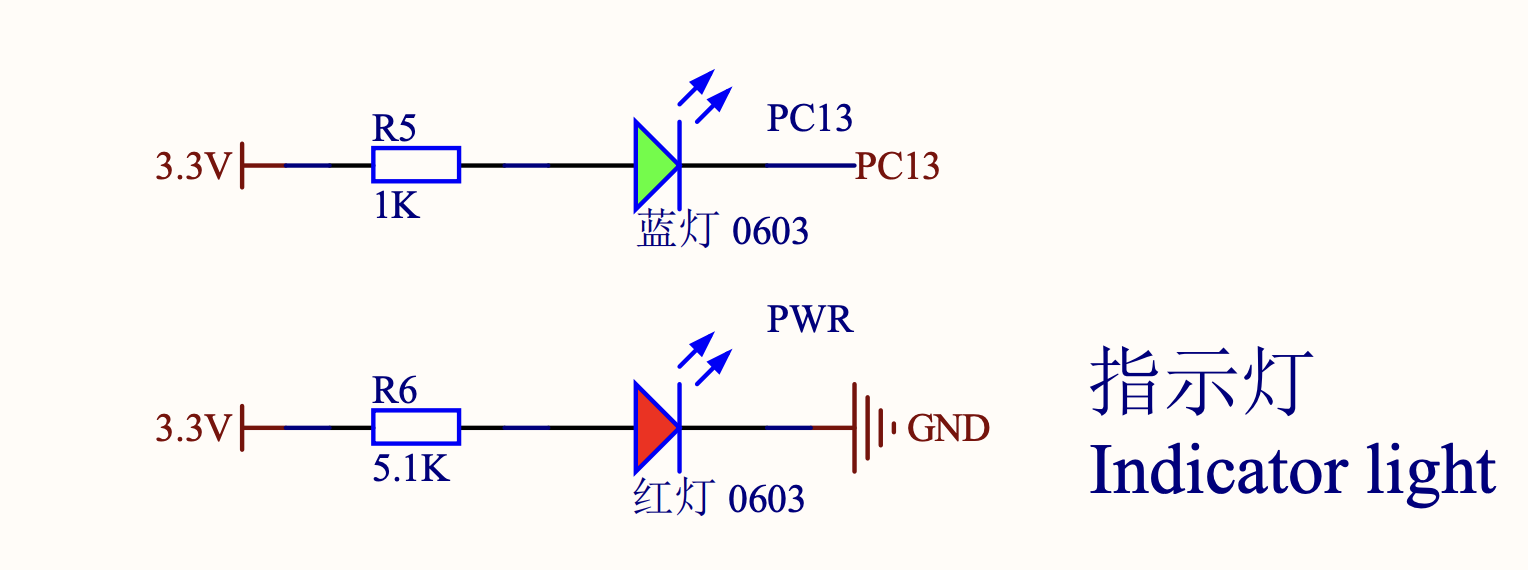


рис. 2 Схема диодов [3]

Чтобы включить эти диоды, нужно включить тактирование на GPIO и направить сигнал на пин PC13.

3. Реализация

Плата подключается к ноутбуку с помощью ST-LINK-V2 и USB-USART. Universal synchronous asynchronous receiver transmitter (USART) — универсальный синхронный/асинхронный приемопередатчик. Настроим его в выбранной IDE.

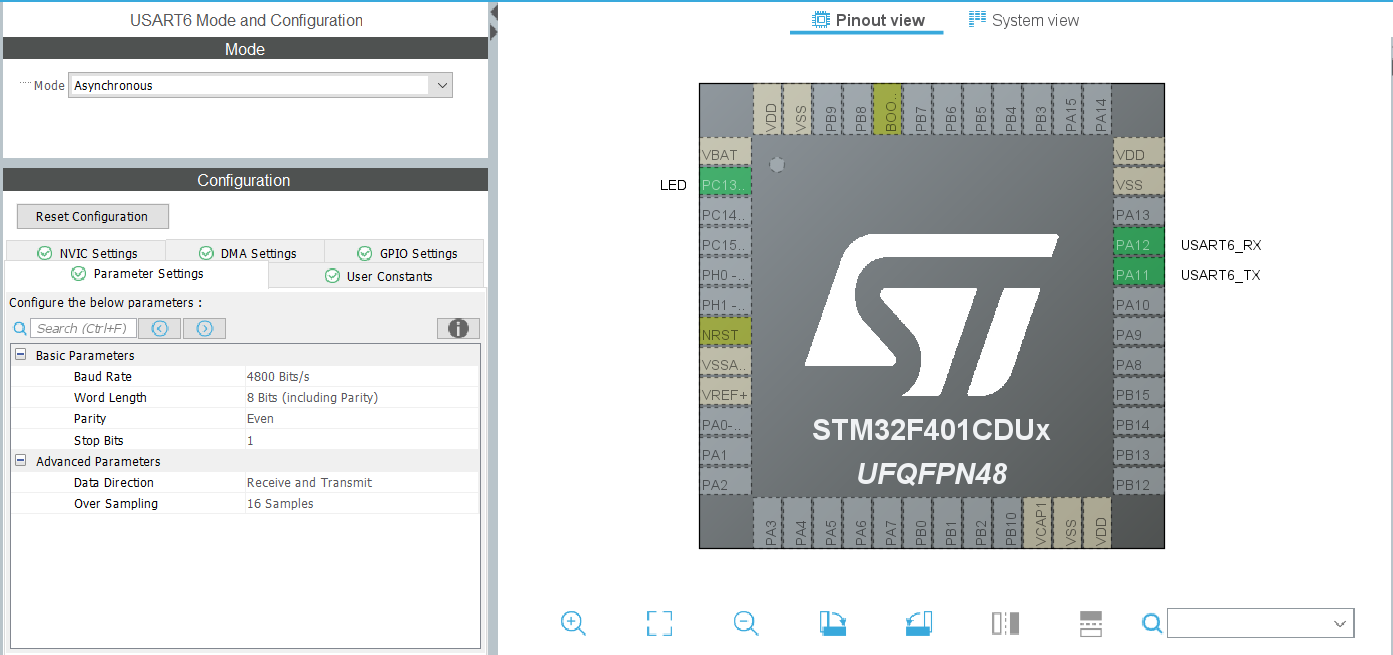


рис. 3 Настройка USART

На рис. 3 видны выбранные настройки: включаем модуль в асинхронном режиме, скорость передачи данных (Baud Rate) - 4800 бит/с, размерность данных (Word Length) - 8 бит (с учетом бита четности), четность (parity) – Even (четное), количество стоп-битов – 1. Также оставляем настройки направления потока данных и over sampling.

Также в NVIC настройках мы включаем прерывания, чтобы использовать их при вводе и выводе данных. После выбранных настроек IDE подсвечивает пины, к которым необходимо подключить USB-USART: передатчик (TX) устройства к приемнику (RX) микроконтроллера и наоборот.

Следующим шагов настроим GPIO. Соответствующему светодиоду пину PC13 присваиваем GPIO\_output, а в настройках GPIO оставляем следующие свойства (рис.4), присваивая ему пользовательское название LED. Автоматическая генерация кода в STM32CubeIDE сама настроит инициализацию портов ввода и вывода, оставив там пользовательское название LED.

Изображение выглядит как текст

Автоматически созданное описание

рис. 4 Настройки пина светодиода

После этих настроек плата готова к работе.

Определим основной функционал терминала по управлению диодом.

1. При вводе «1» диод загорается.
2. При вводе «2» диод гаснет.
3. При вводе «3» терминал предлагает ввести количество морганий светодиода со стандартным периодом.
4. При вводе «4» терминал предлагает ввести количество морганий светодиода и их периодичность.
5. При вводе 5 программа завершается.

Включение и выключение светодиода осуществляется с помощью функции HAL\_GPIO\_WritePin(LED\_GPIO\_Port, LED\_Pin, GPIO\_PIN\_RESET), третий аргумент функции - GPIO\_PIN\_RESET отвечает за включение диода, в то время как GPIO\_PIN\_SET - за его выключение. В начале работы программы светодиод выключен.

Мигание диода реализуется последовательным включением и выключением диода с паузой между ними, которая осуществляется функцией HAL\_Delay(delay).

Ввод и вывод осуществляется пользовательскими функциями read(uint8\_t \*val, const uint8\_t size) и print(const uint8\_t \*val, const uint8\_t size) через HAL с прерываниями. В отличие от функции HAL\_UART\_Transmit, которая среди аргументов имеет тайм-аут операции и работает в блокирующем режиме, для передачи и получения данных будем использовать функцию HAL\_UART\_Transmit\_IT, которая не принимает на вход задержку. При работе этой функции система не дожидается окончания передачи данных, а продолжает работать и выполнять следующие команды. Функция возвращает состояние USART - HAL\_BUSY, что означает что передача данных не была еще закончена. Состояние процесса передачи UART также можно проверить с помощью функции HAL\_UART\_GetState. При занятом передачей UART функция возвращает значение HAL\_UART\_STATE\_BUSY\_TX. Поэтому следующий код: while( HAL\_UART\_GetState (&huart1) == HAL\_UART\_STATE\_BUSY\_TX ) будет выполняться до тех пор, пока передача не остановится, что по функционалу схоже с блокирующим режимом. Прием будет осуществляться похожим же образом, только проверяется уже состояние приемника.

Функции ввода принимают всего 1 символ, таким образом моргнуть светодиодом можно от 0 до 9 раз, аналогично с пользовательским периодом – его можно назначить равным 100мс, 200мс и т.д. до 900мс.

Изображение выглядит как текст

Автоматически созданное описание

рис. 5 Пример работы терминала

Терминал на рис.5 работает следующим образом: после нажатия на выбор в меню, если это включение – выключение диода, сразу предлагает нажать на любую клавишу, чтобы снова вывести меню, а при выборе помигать светодиодом – терминал просит подождать, пока не выполнится цикл, после чего также ждет нажатия любой клавиши, чтобы снова вывести меню.

4. Ссылка на реализацию

<https://github.com/archermarat/mps2022>

5. Заключение

На базе микроконтроллера STM32F401CDU6 с применением интерфейса USART в STM32CubeIDE на языке программирования С была написана программа, осуществляющая работу терминала по управлению светодиода: реализованы функции включения, выключения, а также мигание светодиода с заданными параметрами – количество миганий и их периодичность.

### Список литературы

1. 1. STM32 Cortex®-M4 MCUs and MPUs programming manual [Электронный ресурс] // URL: <https://www.st.com/resource/en/programming_manual/dm00046982-stm32-cortexm4-mcus-and-mpus-programming-manual-stmicroelectronics.pdf>
2. 2. High-performance access line, Arm Cortex-M4 core with DSP and FPU, 384 Kbytes of Flash memory, 84 MHz CPU, ART Accelerator [Электронный ресурс] // URL: <https://www.st.com/en/microcontrollers-microprocessors/stm32f401cd.html>
3. 3. Board-user manual [Электронный ресурс] // URL: <https://stm32-base.org/assets/pdf/boards/original-schematic-STM32F401CEU6_WeAct_Black_Pill_V3.0.pdf>
4. 4. Reference manual [Электронный ресурс] // URL: <https://www.st.com/resource/en/reference_manual/rm0368-stm32f401xbc-and-stm32f401xde-advanced-armbased-32bit-mcus-stmicroelectronics.pdf>