Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение  
высшего образования

«Воронежский государственный университет»

Физический факультет

Кафедра электроники

**Разработка системы управления освещением на основе**

**микроконтроллера**

ВКР «бакалаврская работа»

«03.03.03 Радиофизика»

«Информационные системы»

Допущено к защите в ГЭК \_\_.\_\_.\_\_\_\_

|  |  |  |  |
| --- | --- | --- | --- |
| Обучающийся |  |  | В. А. Сафонов |
| Руководитель |  | к. геогр. н., доц. | И.К. Астанин |
| Зав. кафедрой |  | д.ф.м . н., профессор | А. М. Бобрешов |

Воронеж

2018

**Содержание**

Введение……………………………………………………………..…………….....3

1.Теоретическая часть………………………………………………..……………...4

1.1.Выбор и обоснование компонент системы………………...……….…...4

1.2. Выбор микроконтроллера..........................................................................7

1.2.1. Типовая структура микроконтроллеров………….………….....9

1.2.2. Микроконтроллеры Motorola………...………………………..10

1.2.3. Микроконтроллеры i8051………...……………………………11

1.2.4. Микроконтроллеры PIC……...………………………………...12

1.2.5. Микроконтроллеры AVR…...………………………………….15

1.2.6. Микроконтроллеры STM............................................................16

1.3. Языки программирования........................................................................16

1.3.1. Ассемблер....................................................................................17

1.3.2. Pascal……...………………………………………………..........17

1.3.3. Basic..............................................................................................18

1.3.4. C/C++……………………………………………………………18

1.3.5. Визуальные языки.......................................................................19

1.3.6. Выбор языка программирования...............................................19

1.4. Выводы по разделу 1................................................................................20

2. Разработка системы освещения на основе микроконтроллера.........................21

2.1. Функционал системы...............................................................................21

2.2. Структурная схема...................................................................................21

2.3. Система управления.................................................................................21

2.3.1. Микроконтроллер STM32F100………………………………..22

2.3.2. Подключение модуля KY-018....................................................23

2.3.3. Wi-Fi модуль esp8266..................................................................24

2.3.4. ST-Link/V2……………………………………………….......…28

2.4. Создание программного обеспечения для управляющей системы.....29

2.4.1. Широтно-импульсная модуляция..............................................30

2.4.2. Принцип работы АЦП в STM32................................................32

2.4.3. Интерфейс UART………………………………………………34

2.5. Выводы по разделу 2................................................................................36

3. Экономическая часть.............................................................................................37

3.1. Анализ существующих систем на рынке...............................................37

3.2. Расчет себестоимости...............................................................................38

3.3. Выводы по разделу 3................................................................................38

Заключение.................................................................................................................40

Список литературы....................................................................................................41

Приложение А. Схема...............................................................................................42

Приложение Б.Рисунки.............................................................................................43

Приложение В.Листинг.............................................................................................49

ВВЕДЕНИЕ

В каждом современном доме есть электрическое освещение. Создание концепции умного дома заставило людей задуматься: как должно выглядеть умное освещение? Какие дополнительные функции можно привнести в него?

Так как существует множество компаний разрабатывающие умные дома, то каждая из них ответила на этот вопрос по-своему. Но есть у них и нечто общее. Это высокая стоимость такого оборудования. При том, что себестоимость не сильно возрастает по сравнению с обычным, «не умным», освещением. Это и является одной из основных причин почему я выбрал данную тему. В ходе данной работы будет показано, что на самом деле «умное» освещение - это недорогая и простая система, которая может быть запросто установлена в любом доме.

Для выполнения работы нам потребуются: сами предметы освещения, управляющие элементы, способ управления.

Все эти предметы будут подробно рассмотрены и выбраны в дальнейшем ходе работы.

Также система должна удовлетворять некоторым простым критериям:

1.гибкость (легкость, а также простота использования, возможность настройки)

2.надежность (автоматический контроль над состояниями системы)

3.высокая функциональная возможность и простота в обращении

4.маленькой себестоимостью, большой экономичностью

5.Безопасность использования

В результате данной работы будет создана система, удовлетворяющая данным критериям.

1. **Теоретическая часть**

**1.1. Выбор и обоснование компонент системы**

Наиболее распространенной на данным момент является традиционная система центрального освещения с управлением от обычных переключателей. Но они нас не интересуют. Нас интересует «умное» освещение. Попробуем разбить их на отдельные части. Например, по используемым осветительным приборам. Бывают:

1.Лампы накаливания

2.Люминесцентные лампы

3.Галогенные и металлогалоидные источники света

4.Светодиоды и лампы на их основе

5.Световые шнуры «дюралайт» (Duralight)

6. Оптические волокна

Рассмотрим их преимущества и недостатки:

Спектр ламп накаливания наиболее комфортен для зрения, но они лишь около 10% своей мощности расходуют на создание света, а остальную преобразуют в тепловую энергию. Поэтому в помещение со слишком большим количеством лампочек накаливания всегда будет жарко и душно, как в фотосалоне или телевизионной студии. Люминесцентные лампы не нагреваются, но до недавнего времени их неохотно использовали для освещения жилья из-за некомфортного голубовато-белого света, неприятного гула и необходимости дополнительных пусковых устройств для подключения. Современное поколение энергосберегающих люминесцентных ламп свободно от этих недостатков, и они вполне могут использоваться вместо ламп накаливания. Галогенные и металлогалоидные светильники излучают довольно узкий направленный пучок света, а поэтому больше подходят для освещения отдельных зон или подсветки ключевых элементов интерьера — картин, скульптурных групп, колонн, ниш и т.д. Светодиоды прочны, не нагреваются, в их очень широком спектре нет вредных для здоровья человека инфракрасного и ультрафиолетового излучений. Они являются одним из самых экологически чистых источников света. Светодиодные лампы не используют веществ, содержащих [ртуть](https://ru.wikipedia.org/wiki/%D0%A0%D1%82%D1%83%D1%82%D1%8C), поэтому они не представляют опасности в случае выхода из строя или повреждения колбы. Долговечность светодиодов в 80 раз превышает ресурс ламп накаливания. Но светоотдача светодиодов не настолько велика, чтобы использовать их в роли самостоятельных источников света и светят они только в одном направлении, а поэтому в интерьере их используют чаще для декоративной подсветки или в качестве ночника. Световые шнуры «дюралайт» — это декоративные источники света на основе светодиодов, широко использующиеся для светового оформления элементов интерьера или в наружном уличном освещении. Оптические волокна излучают свет только с торцов, но удаляя их на большие расстояния от основного источника света и используя специальные насадки, можно создавать волшебные фантастические световые эффекты.

Рассмотрим еще чуть более подробно светодиодное освещение. У него есть один крупный недостаток, из-за которого, казалось бы, невозможно его использовать. Но эта проблема легко разрешается использование не отдельных светодиодов, а светильников со сменными светодиодными лампами. Эти лампы имеют в своем составе несколько светодиодов, что решает проблему недостаточной светоотдачи, и рассеиватель, который решает проблему узкой направленности. Этот вариант несколько дороже, но за счет низкого энергопотребления и долгого срока службы он быстро окупится. В своей работе я планирую использовать именно этот тип освещения.

Далее рассмотрим возможные типы управляющих устройств.

1.Обычные переключатели

2. [Механические](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%BC%D0%BC%D0%B5%D1%80#%D0%9C%D0%B5%D1%85%D0%B0%D0%BD%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B4%D0%B8%D0%BC%D0%BC%D0%B5%D1%80) диммеры (на основе переменного резистора)

3. Электронные диммеры (на основе микроконтроллера)

Рассмотрим их плюсы и минусы:

Обычные переключатели обладают жестко ограниченным функционалом: включение и выключение. Этого недостаточно для «умного» освещения. Механические диммеры – это хоть и простое, но безнадежно устаревшее решение, добавляющее к предыдущему только одну функцию – регулировка уровня освещенности. А вот функционал электронных диммеров ограничен только возможностями микроконтроллера, которые, в свою очередь, ограничены только фантазией и навыками разработчика. Именно этот вариант я и буду использовать.

Остается только рассмотреть способы управления освещением:

1.Механическое управление

2.Электронное управление

2.1. Контактное

2.2. Бесконтактное

3. Дистанционное управление

4. Акустическое управление

Изучим их подробнее:

В основе механического диммера [потенциометр](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%82%D0%B5%D0%BD%D1%86%D0%B8%D0%BE%D0%BC%D0%B5%D1%82%D1%80), подключённый не непосредственно к нагрузке, а передающий сигнал через схему управления на силовой элемент ([реостат](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BE%D1%81%D1%82%D0%B0%D1%82), [дроссель](https://ru.wikipedia.org/w/index.php?title=%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D1%8B%D0%B9_%D0%B4%D1%80%D0%BE%D1%81%D1%81%D0%B5%D0%BB%D1%8C&action=edit&redlink=1), [тиристор](https://ru.wikipedia.org/wiki/%D0%A2%D0%B8%D1%80%D0%B8%D1%81%D1%82%D0%BE%D1%80)). Электронное управление устроено на использовании всевозможных датчиков. Отсюда и деление на контактный и бесконтактный. Если первый устроен на использовании, например, сенсорного управления, то второй все различные датчики движения и т.п. В дистанционном управлении используются всеразличные пульты дистанционного управления, использующие какой-либо канал связи, или излучающие инфракрасные или радио волны. Акустическое управление основано на реакции прибора на различные громкие звуки или на голосовые команды. Т.е. его можно рассматривать как голосовое управление. Но это довольно дорогая и сложная система. В своей работе я буду использовать дистанционное управление. Конкретный выбор и схема управления будет рассмотрены позже. Хотя в одном приборе и могут одновременно сосуществовать несколько различных способов управления, в своем проекте я не нахожу это целесообразным.

В конце данного пункта уже можно в общем виде обрисовать мою систему. Это система на основе светодиодных светильников с микроконтроллерным управлением, которые, в свою очередь, будут управляться дистанционно. Все эти компоненты и их реализации будут более подробно рассмотрены в отдельных пунктах далее.

**1.2 Выбор микроконтроллера**

Микроконтроллер - это самостоятельная компьютерная система, которая содержит процессор, вспомогательные схемы и устройства ввода-вывода данных, размещенные в общем корпусе.

Впервые микроконтроллеры появились в 1971 году, как и микропроцессоры общего назначения. Создатели микроконтроллеров разработали остроумную идею - совместить процессор, ПЗУ, оперативную память и периферию, поместив это внутри одного корпуса, по внешнему виду похожего на обыкновенную микросхему. С того момента производство микроконтроллеров из года в год во много раз превосходит производство процессоров, а потребность в них не уменьшается.

Но по-настоящему история микроконтроллеров начинается с выпуска компанией intel микроконтроллера intel8051 в 1980 году. Он является микроконтроллером второго поколения и является устройством с большим объемом ROM и EPROM памяти, новыми портами ввода-вывода и возможностью добавления внешней памяти. К тому же, он имел максимальную тактовую частоту 20MHz, то есть был достаточно быстрым. 8051 микроконтроллеры в скором времени начали производиться огромным количеством компаний в различных модификациях. Некоторые из этих модификаций производятся и в наше время. Микроконтроллеры с 8051 архитектурой имею следующие типовые характеристики:

* Тактовая частота 24 MHz
* Командный цикл в 12 тактов
* Объем памяти программ 4К
* Объем RAM 128байт
* Число линий ввода-вывода – 32
* Два 8\16 разрядных таймера
* Множество внутренних и внешних прерывателей
* Программируемый последовательный порт
* Интерфейс с внешней памятью и объемом до 128К

К фирмам производителям 8051 микроконтроллера относятся такие гиганты, как intel, Atmel, Dallas Semiconductor, Philips Semiconductor.

Микроконтроллеры AVR производят десятки компаний, при этом изготавливают как 32-битные современные микроконтроллеры, так и 16-битные, и даже 8-битные (такие, как i8051 и аналоги). В каждом семействе нередко можно повстречать практически одинаковые модификации, отличающиеся скоростью работы их процессора и объемом содержащейся памяти. Дело состоит в том, что микроконтроллеры используются предпочтительно во встроенных системах: в станках, в бытовой технике и автоматике, в любых автоматических устройствах - там, где необходима не столько мощность процессора, сколько, баланс между стоимостью и необходимой функциональностью.

Поэтому самые старые разновидности микроконтроллеров применяются до сих пор - они все еще многое могут: от автоматического управления дверями, включения/отключения освещения или полива газонов до управления автоматической системой «умный дом». Так же и существуют и наиболее мощные микроконтроллеры, выполняющие сотни миллионов команд в секунду и управляющие всевозможной периферией. У таких микроконтроллеров и предназначения соответствующие. Разработчик, таким образом, вначале оценивает задачу, а уж затем подбирает под нее подходящие элементы.

Сегодня существует намного более 200 разновидностей микроконтроллеров, совместимых с i8051, производимых двумя десятками фирм, и огромное количество микроконтроллеров иных типов. Большой популярностью у конструкторов пользуются 8-битные PIC-контроллеры компании Microchip Technology и AVR компании Atmel, 16-битные MSP430 компании TI, а также 32-битные микроконтроллеры с архитектурой ARM, разработку которой ведет компания ARM Limited и продаёт другим компаниям лицензии для их производства.

**1.2.1 Типовая структура микроконтроллера.**

Микроконтроллер содержит в своем составе микропроцессорное ядро с развитой системой команд, память программ IROM,

регистровый файл данных RRAM, а также набор программируемых

интерфейсных схем, выполняющих ряд важных функций внутри МК

и обеспечивающих связь с внешней средой. Большинство из перечисленных блоков являются обязательными компонентами любого

МК, некоторые, например IROM, в структуре конкретного МК могут отсутствовать. Состав и назначение интегрированных на кристалл

ПУ определяется областью преимущественного применения МК

и может широко варьироваться в зависимости от типа МК. В целях

повышения гибкости использования аппаратных средств, интегрированных на кристалле, эти средства делаются многофункциональными с программной настройкой на тот или иной режим работы.

При инициализации МК информация о типе настройки заносится

в специальные регистры и в процессе дальнейшей работы МК

обычно остается неизменной. В большинстве случаев в число встроенных ПУ входят таймеры/счетчики, последовательные и параллельные порты

ввода - вывода, контроллер прерывания, многоканальный АЦП, сторожевой таймер и пр.

**1.2.2 Микроконтроллеры Motorola.**

Motorola 6800 был одним из первых доступных 8-разрядных микропроцессоров. Несмотря на то, что его архитектуре уже много лет, он до сих пор жизнеспособен и используется в качестве процессорного ядра для микроконтроллеров.

Компания Motorola создала весьма полный функциональный ряд микроконтроллеров 68НС05, развитием которого являются микроконтроллеры семейства 68НС08. Наиболее широкие функциональные возможности обеспечивают микроконтроллеры семейства 68HCI1, которые имеют несколько другую архитектуру и реализуют расширенный набор функций, среди которых, например, имеется команда деления. Семейства микроконтроллеров 68НСхх имеют архитектуру Фон-Неймана или Принстонскую, а их ядром является CISC-процессор (компьютер со сложным набором команд). Это делает семейства 68НСхх достаточно уникальными среди микроконтроллеров, но для реализации широкого круга приложений, от пейджеров до автомобильных контроллеров, их архитектура оказывается наиболее подходящей.

Имея Принстонскую архитектуру, 68НСхх могут выполнять некоторые приложения более эффективно, чем другие микроконтроллеры. Микроконтроллеры HCl I, как их чаще всего называют, имеют не очень много отличий, но обеспечивают существенно лучшие показатели по производительности.

На 2001 год насчитывалось уже более 180 моделей 68НС05 (без учета разнообразия типов корпусов), которые обладают различными характеристиками и функциональными особенностями возможностей, предоставляемых семействами 68НСхх, поистине поражает.

Для лучшего представления о назначении отдельных микроконтроллеров, семейство НС05 разделено на серии, каждая из которых маркируется буквенным кодом. Система буквенных обозначений также используется для маркировки микроконтроллеров 68НС08 и 68HCU, однако одинаковые буквенные индексы в обозначении приборов из разных семейств соответствуют различным функциональным возможностям. Многие из этих микроконтроллеров разработаны не для общего пользования, а для специального применения. Это означает, что вы можно найти микроконтроллер, который полностью соответствует требованиям, но недоступен для использования, так как поставляется только по специальным заказам. Наилучшим способом выбора микроконтроллеров из семейств 68НС\х является выбор прибора, удовлетворяющего требованиям, из каталогов дистрибьюторов продукции компании Motorola, вместо непосредственного поиска по фирменной документации.

**1.2.3 Микроконтроллеры 8051.**

Одновременно с созданием в 70-х годах первых микропроцессоров компания INTEL вела также разработку микроконтроллеров. Усилия по разработке микроконтроллеров воплотились в архитектуру 8051, которая впервые была представлена в 1980 году и стала одной из наиболее популярных микроконтроллерных архитектур. Микроконтроллер второго поколения 8051 представляет собой законченное устройство с большим объемом встроенной памяти программ (ROM и EPROM) и данных RAM, улучшенными портами ввода-вывода и возможностью подключения внешней памяти Первые микроконтроллеры 8051 были достаточно быстрыми, имея максимальную тактовую частоту 20MHz. В то время как популярность большинства других типов микроконтроллеров определяется объемом их продаж, 8051 снискали другую славу.

Микроконтроллеры с архитектурой 8051 производятся наибольшим числом компаний. В этом заключается их несомненное преимущество, так как каждый производитель производил усовершенствования 8051 с целью расширения функциональных возможностей или повышения скорости. Благодаря этому архитектура 8051 стала весьма привлекательной для реализации широкого диапазона приложений.

Микроконтроллеры 8051 имеют следующие типовые характеристики:

* тактовая частота - 24 МГц;
* объем памяти программ - 4К;
* командный цикл - 12 тактов;
* объем памяти данных RAM - 128байт;
* число линий ввода-вывода - 32;
* два 8/16-разрядных таймера;
* множество внутренних и внешних источников прерываний;
* программируемый последовательный порт;
* интерфейс с внешней памятью объемом до 128К.

Многие производители провели усовершенствования и расширили возможности выпускаемых ими микроконтроллеров данной архитектуры.

**1.2.4 Микроконтроллеры PIC**

Микроконтроллеры семейства PICMicro относятся к числу доминирующих и наиболее эффективных микроконтроллеров. По такому параметру как рабочая скорость, которая определяется количеством команд, выполняемых за один такт.

Особенностью данного семейства является большое количество новых моделей выпускаемых фирмой Microchip - свыше 40 за 1997 год. Эти микроконтроллеры широко известны под именем PIC.

Архитектура PIC была разработана достаточно давно. Первоначально она была предложена отделением компании General Instrument, которое занималось производством микросхем. Позже это отделение выделилось в качестве самостоятельной фирмы Microchip, которая начала проводить агрессивную компанию по продвижению на рынок микроконтроллеров PICMicro и серии микросхем памяти EEPROM с последовательным доступом.

Различают 3 модели семейства PICMicro:

1.Младшие модели семейства PICMicr**o**

Младшие модели серии 16С5х реализуют нижний уровень возможностей микроконтроллеров семейства PICMicro. Они выполняют подмножество команд, реализуемых старшими моделями, и программно совместимы с ними. Не рекомендуется использовать эти устройства для разработки новых приложений для PICMicro. Недостаток прерываний, малый объем доступной памяти программ и RAM, отсутствие внутрисистемного программирования и портов ввода/вывода с расширенными функциями являются причинами трудностей, которые возникают при реализации приложений на различных моделях микроконтроллеров этого семейства.

Решающим фактором перехода от младших моделей к моделям среднего уровня послужило то обстоятельство, что последние продаются по аналогичной и даже меньшей цене и в таких же корпусах, как и первые.

Рассмотрим младшие модели микроконтроллеров с восьмью выводами.

PICMicro серии 12С5хх. Обладая 6-ю линиями ввода/вывода и объемом памяти программ 512 или 1024 команд, эти устройства используются для создания небольших и простых приложений. Микроконтроллеры этой серии, возможно, являются наилучшими простыми интерфейсными устройствами.

2.Средние модели семейства PICMicro

Модели среднего уровня образуют законченную линию микроконтроллеров со множеством различных функций. Они имеют такую же базовую архитектуру, как микроконтроллеры младших моделей, но существенно отличаются возможностями обработки прерываний.

В моделях среднего уровня реализован следующий набор «стандартных» свойств:

* количество уровней прерываний (подпрограмм) - 8;
* объем памяти программ - 512-8 К команд;
* регистровый файл (RAM) - 36-192 байт;
* число источников прерываний - 4-12;
* число таймеров - 1-3;
* возможность внутрисхемного программирования;
* напряжение питания - 2,0 - 5,0 В;
* тип корпуса - пластиковый ОТР, керамический с окном,S01C, ОFР;
* число линий ввода/вывода - 13-33;
* тактовая частота - от 0 до 4 МГц или до 20 МГц;
* тип памяти программ - масочно-программируемая, ЕРROM, EEPROM/Flash;
* возможности АЦП - Компараторы напряжения, ЦАП с резистивным делителем, АЦП интегрирующего типа, ШИМ выходы;
* последовательные порты - SPI. I2C, асинхронный.
* другие порты - Порог прямого подключения ЖКИ, параллельный ведомый порт.

Широкий диапазон возможностей, включая внутрисистемное программирование, дает этим микроконтроллером значительные преимущества, позволяй использовать один и тот же программный код при реализации разнообразных приложений.

3.Старшие модели семейства PICMicro

Старшие модели микроконтроллеров PICMicro используют 16-битные команды и могут адресовать 64К слов памяти. Так как каждое слово содержит 16 бит, то старшие модели серии 17Схх могут адресовать по 128 Кбайт памяти программ и памяти данных.

Для обеспечения чтения/записи памяти программ в серии 17Схх используется модифицированное ядро центрального процессора. Это ядро позволяет различным командам обращаться ко всем регистрам процессора. Это повышает гибкость микроконтроллера и позволяет использовать команды в широком спектре приложений. В то время как в микроконтроллерах среднего уровня используется только один вектор прерывания, в старших моделях PICMicro имеется несколько векторов прерываний. Старшие модели PICMicro разрабатывались, в основном, для взаимодействия с другими цифровыми устройствами. По этой причине в микроконтроллерах серии 1 7Схх нет АЦП и устройств, обеспечивающих непосредственное подключение датчиков, которые имеются в моделях среднего уровня.

**1.2.5 Микроконтроллеры AVR**

Микроконтроллеры AVR фирмы Amtel открывают одно из новых направлений в области разработки и архитектуры микроконтроллеров. Структура процессора AVR представляется собой RISC-архитектуру с повышенной производительностью и пониженным энергопотреблением Гарвардского типа.

Одним из основных достоинств этого контроллера является быстрое выполнение команд – он выполняет команду за один такт. АVR имеет, наиболее разносторонний по своим возможностям процессор из всех микроконтроллеров. Это означает, что при разработке приложений надо потратить немного больше времени на планирование размещения данных в памяти и регистрах, чем для других микроконтроллеров. Но благодаря своей разносторонности AVR очень прост в программировании на языке «Ассемблер» так и для языков высокого уровня, например С#.

Главным преимуществом AVR является наличие памяти EEPROM для хранения программ с возможностью программирования в системе, а также расширенный набор команд с возможностью выполнения большинства команд за один машинный цикл.

**1.2.6 Микроконтроллеры STM**

Компания STMicroelectronics одной из первых вывела на рынок семейство микроконтроллеров на ядре ARM Cortex-M3 и на сегодняшний день по праву занимает лидирующее место среди производителей микроконтроллеров на этом ядре. Все началось в 2007 году с двух семейств — Performance Line (STM32F103) и Access Line (STM32F101). Компания постоянно работает как над расширением номенклатуры семейства, так и над улучшением характеристик, не забывая при этом также пополнять программную составляющую продукта. На сегодняшний момент [STM32](http://catalog.compel.ru/mcu/list?BRAND=ST)уже состоит из 10 линеек для всевозможных применений — микроконтроллеры с высокой производительностью, недорогие микроконтроллеры общего применения, микроконтроллеры с ультранизким энергопотреблением, микроконтроллеры со встроенным радиомодулем для беспроводных решений, и все это — на одном ядре ARM Cortex-M3! Нельзя не отметить pin-to-pin и программную совместимость по всем линейкам.

**1.3 Языки программирования**

Языки программирования микроконтроллеров мало отличаются по своей структуре от классических языков для компьютеров. Единственное отличие заключается в том, чтобы сосредоточиться на работе со встроенными периферийными устройствами. Для архитектуры микроконтроллеров требуется, например, наличие бит-ориентированных команд. Последнее позволяет вам работать с отдельными линиями портов ввода-вывода или флагами регистров. Подобные команды отсутствуют в большинстве крупных архитектур. Даже ядро ARM, которое активно используется в микроконтроллерах, не содержит битовых команд, поэтому разработчикам приходилось создавать специальные методы для доступа к битам.

**1.3.1 Ассемблер**

Ассемблер - это язык низкого уровня. При этом он позволяет полностью раскрыть все возможности микроконтроллеров и получить максимальную скорость и компактный код. В некоторых случаях альтернативы ассемблеру нет, но, тем не менее, у него много недостатков. Несмотря на компактность машинного кода, программа, написанная на языке ассемблера, громоздка и трудна для понимания. Для её создания требуется отличное знание архитектуры и командной системы микроконтроллеров.

Ассемблер отлично подходит для программирования микроконтроллеров с ограниченными ресурсами, например, 8-разрядных моделей с низкой памятью. Для больших программ и, тем более, для 32-битных контроллеров лучше использовать языки более высокого уровня. Это позволит вам создавать более сложные и понятные программы.

**1.3.2 Pascal**

Язык Pascal еще более удобен для восприятия и изучения. Тем не менее, он не имеет такого распространения как C/C++, особенно при программировании микроконтроллеров. Некоторые отдельные фирмы поддерживают данный язык, с целью упрощения перехода на контроллеры с больших ПК. В частности вариант языка под названием MicroPASCAL входит в состав поставки отладочных средств фирмы Mikroelektronika.

**1.3.3 BASIC**

Старый язык начального обучения программированию, в настоящее время в значительной степени сохраняется в форме реализации Microsoft Visual BASIC. Он также используется для программирования микроконтроллеров. Реализаций этого языка намного больше, чем на том же Pascal. Это в первую очередь связано с простотой языка. BASIC часто выбирают разработчики программных и аппаратных платформ, направленных на упрощение разработки электронных устройств. Можно назвать такие проекты, как PICAXE, Amicus18, microBASIC и некоторые другие. Недостатком BASIC является плохо структурированный код. Этот язык не следует выбирать для первоначального изучения с целью дальнейшего перехода на C / C ++. Программирование микроконтроллеров на BASIC может быть рекомендовано радиолюбителям, направленным на создание, в основном, простых устройств.

**1.3.4 С/С++**

Язык программирования C / C ++ относится к языкам более высокого уровня по сравнению с ассемблером. Программа на этом языке лучше понятна человеку. Преимуществом C / C ++ является огромное количество программных средств и библиотек, которые позволяют вам запросто создать необходимый код. Фактически, C / C ++ сегодня стал основным языком разработки программ для микроконтроллеров. Компиляторы этого языка реализованы практически для всех моделей микроконтроллеров. Стандартный язык позволяет передавать программы с одной платформы на другую. Теоретически, используя разные компиляторы, вы можете конвертировать любую программу в команды микроконтроллера требуемого типа. На практике дополнительно необходимо учитывать архитектуру микроконтроллера каждого типа.

Язык C / C ++ имеет довольно сложную структуру. Полученный программный код для конкретной задачи имеет больший объем, чем код той же задачи, реализованной в ассемблере. Тем не менее, язык C / C ++ следует признать единственным правильным выбором для профессионального программирования микроконтроллеров.

**1.3.5 Визуальные языки**

В отличие от классических языков программирования, визуальные языки позволяют разрабатывать программы в виде изображений. Среди таких языков вы можете выбрать FlowCODE или Scratah. Преимущество визуальных языков - это хорошо воспринимаемая структура алгоритма. Это позволяет вам запросто понять его функционирование любому человеку, который знает основные символы языка. Перевод структурных диаграмм в инструкции микроконтроллера, как правило, не выполняется сразу. Первоначально алгоритм преобразуется в ассемблер или некоторые высокоуровневые языковые команды. Только тогда все преобразуется в машинный код. Эта схема, несмотря на ее сложность, позволяет использовать самые удобные компиляторы разных разработчиков.

Другим преимуществом визуального программирования является простота обучения. Недостатком визуального подхода является громоздкий характер исходных материалов. Тем не менее, подобные языки программирования нашли очень широкое распространение для решения специальных задач.

**1.3.6 Выбор языка программирования**

Выбор того или иного языка программирования зависит от множества факторов. В первую очередь необходимо определиться с типом решаемых задач и необходимым качеством кода. Если не требуется разработка объемных и сложных программ, то можно использовать практически любой язык. Для обеспечения компактности кода подойдет Ассемблер, а если ставятся серьезные задачи, то альтернативы С/С++ практически нет. Также необходимо учитывать доступность компилятора. В некоторых случаях, реализация языка может вообще отсутствовать, или предлагаться за солидные деньги. В итоге самым универсальным решением можно назвать связки Ассемблер и  C/C++.

**1.4 Выводы по разделу 1**

При выборе компонент для нашей системы освещения мы учитывали несколько условий:

1.гибкость (легкость, а также простота использования, возможность настройки)

2.надежность (автоматический контроль над состояниями системы)

3.высокая функциональная возможность и простота в обращении

4.маленькая себестоимость, большая экономичность

5.Безопасность использования

Исходя из этого набора условий, мы пришли к выводу, что наша система должна состоять из светодиодного освещения с микроконтроллерным беспроводным управлением. Это достаточно современная, и пользующаяся спросом система. Для частного пользования системы с рынка оказываются слишком дорогими, но при нашем исполнении этот недостаток должен исчезнуть.

**2. Разработка системы освещения на основе микроконтроллера**

**2.1. Функционал системы**

Для начала создания системы управления нам требуется определиться каким функционалом должна обладать система. Очевидно, что обычного включения и выключения освещения недостаточно. Как минимум освещение должно быть не просто включено или выключено, но еще и иметь выборочным уровень яркости. К этому также можно добавить некоторый уровень автоматизации освещения. Например, автоматическое включение света при достижении какого-то определенного порога темноты. Для этого потребуются дополнительные датчики освещенности.

**2.2 Структурная схема**

Зная требования к функционалу системы, и учитывая условия из предыдущих пунктов мы можем представить структурную схему системы (рисунок Б.1 в приложении Б.)

Структурная схема состоит из следующих элементов:

* МК – микроконтроллер
* WF – wi-fi модуль
* ДО – датчик освещенности(фоторезистор)
* ИП – источник питания
* ОУ – объект управления

Данные с датчиков и команды с телефона (через wi-fi модуль) передаются на микроконтроллер, который их обрабатывает и передает непосредственно самим объектам управления, в результате чего и достигается необходимый уровень освещения в помещении.

**2.3 Система управления**

В качестве управляющего устройства был выбран микроконтроллер

STM32F100 с необходимой периферией, подключенный к wi-fi модулю esp8266-01. В общем случае управляющая система состоит из:

* Микроконтроллер STM32F100
* Модуль KY-018 с фоторезистором
* Wi-fi модуль esp8266-01
* ST-LINK/V2 (программатор/отладчик)

Также в данном устройстве для согласования микроконтроллера (рисунок Б.2 в приложении Б.) с самим объектом управления используется N-канальный полевой транзистор IRF520. Объектом управления в данном конкретном случае будет светодиодная лента с питанием от 12 вольт.

Далее рассмотрим эти элементы подробнее.

**2.3.1 Микроконтроллер STM32F100**

STM32F100 младшая линейка дешевых микроконтроллеров от компании STMicroelectronics на базе ядра Cortex-M3. Микроконтроллеры включают в себя широкий набор интерфейсов и большой объем встроенной памяти: ядро Cortex-M3 с частотой процессора до 24 МГц, Flash до 512 кБ, до 32 кБ RAM, большее количество таймеров, часы реального времени (RTC), до 5 UART, до 2 I2C, до 3 SPI, 12-битный АЦП и 12-битный ЦАП, встроенный температурный датчик, а также контроллер внешней памяти (EMC). Микроконтроллеры семейства STM32F100х («Value Line») предназначены для различных крайне чувствительных к стоимости применений, где возможностей 16-битного микроконтроллера уже недостаточно, а функциональность обычных 32-битных микроконтроллеров избыточна. Выпускаются в корпусах: LQFP48, LQFP64, TFBGA64, LQFP100, LQFP144.

Основные характеристики линейки STM32F100х («Value Line»):

* Максимальная тактовая частота 24 МГц (30 DMIPS)
* Умножение и деление за 1 такт
* Напряжения питания 2.0 – 3.6 В
* От 4 до 8 Кб ОЗУ
* От 16 до 128 Кб флэш-памяти
* Два встроенных и откалиброванных тактовых генератора на 40 КГц и 8 МГц
* 7-канальный DMA контроллер
* 16-канальный 12-битный АЦП (1.2 мкс) с датчиком температуры
* Два 12–битных ЦАП
* До 80 быстрых портов ввода – вывода (есть совместимость с 5 В)
* 16 внешних прерываний
* Два сторожевых таймера (IWDG и WWDG)
* До 10 таймеров общего и расширенного назначений
* До 2х I2C(SMBus/PMBus), до 3х USART (Lin, IrDa, modem control), до 2 SPI(2 Мбит/с), HDMI (CEC), RTC
* Управление питанием и сбросом (3 режима низкого потребления, PVD, BOR)
* Аппаратный расчет CRC
* 96–битный уникальный идентификатор (ID)

**2.3.2 Подключение модуля KY-018**

Принцип работы данного датчика довольно прост: Чем ярче освещен фоторезистор, тем ниже его сопротивление. Сопротивление фоторезистора при изменении освещенности меняется в широких пределах от единиц кОм и до сотен кОм или МОм. Точно выяснить изменение сопротивление фоторезистора следует экспериментально с помощью омметра. Контакты и схема модуля KY-018 позволяют использовать только фоторезистор или фоторезистор в составе делителя напряжения.

Для этого на плате установлен резистор 10 кОм.  
Питание модуля подают на контакт +5 В. С увеличением освещенности на выходе модуля фоторезистора напряжение будет падать, при ярком свете напряжение выхода будет около половины напряжения питания. Величина напряжения на выходе зависит от типа фоторезистора. В темноте напряжение выхода будет близко к напряжению контакта +5 В. Схема модуля KY-018 на рисунок Б.3 в приложении Б.

При работе совместно с МК выход модуля фоторезистора соединяют с входом АЦП микроконтроллера. Так как изменение сопротивления фоторезистора при освещении значительно, то с помощью АЦП можно легко фиксировать наступление темноты или включение освещения.

**2.3.3 Wi-fi модуль esp8266**

ESP8266 — микроконтроллер китайского производителя Espressif с интерфейсом [Wi-Fi](https://ru.wikipedia.org/wiki/Wi-Fi). Помимо Wi-Fi микроконтроллер отличается возможностью исполнять программы из внешней [флеш-памяти](https://ru.wikipedia.org/wiki/%D0%A4%D0%BB%D0%B5%D1%88-%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C) с интерфейсом [SPI](https://ru.wikipedia.org/wiki/Serial_Peripheral_Interface).

Основное применение ESP8266 находит в управлении разнообразными бытовыми приборами через беспроводные сети. Концепцию такого управления часто называют «[Internet of Things](https://ru.wikipedia.org/wiki/Internet_of_Things)» (IoT, «интернет вещей»). Верхний уровень IoT представлен разнообразными приложениями под популярные платформы (Android, iOS, Windows). Эти приложения позволяют разработчику прибора адаптировать приложение под управление его прибором и передать пользователю готовое решение.

Всего существует 13 модификаций данного модуля от esp8266-01 до esp8266-13. Они отличаются размерами, антеннами, наличием или отсутствием экрана, а также количеством и назначением ножек на плате.

На плате имеется свой встроенный микроконтроллер с параметрами:

* 80 MHz 32-bit процессор [Tensilica](https://en.wikipedia.org/wiki/Tensilica) L106. Возможен негарантированный разгон до 160 МГц.
* [IEEE 802.11](https://ru.wikipedia.org/wiki/IEEE_802.11) b/g/n [Wi-Fi](https://ru.wikipedia.org/wiki/Wi-Fi). Поддерживается [WEP](https://ru.wikipedia.org/wiki/Wired_Equivalent_Privacy) и WPA/WPA2.
* 14 портов ввода-вывода(из них возможно использовать 11), [SPI](https://ru.wikipedia.org/wiki/Serial_Peripheral_Interface), [I²C](https://ru.wikipedia.org/wiki/I%C2%B2C), [I²S](https://ru.wikipedia.org/wiki/I%C2%B2S), [UART](https://ru.wikipedia.org/wiki/UART), 10-bit [АЦП](https://ru.wikipedia.org/wiki/%D0%90%D0%A6%D0%9F).
* Питание 2,2…3,6 В. Потребление до 215 мА в режиме передачи, 100 мА в режиме приема, 70 мА в режиме ожидания. Поддерживаются три режима пониженного потребления, все без сохранения соединения с точкой доступа: Modem sleep (15 мА), Light sleep (0.4 мА), Deep sleep (15 мкА).

Но в ходе данной работы использовать непосредственно этот микроконтроллер мы не будем, так как у нас уже есть внешний программируемый микроконтроллер stm32f100.

Модуль esp8266-01 является простейшей модификацией платы esp8266. Она отличается простотой в использовании, полосковой антенной (PCB antenna) с радиусом действия до 400 метров на открытой местности. Модуль управляется АТ командами.

**2.3.3.1 AT-команды**

AT-команды набор команд, разработанных в 1977 году компанией Hayes для собственной разработки, модема «Smartmodem 300 baud». Набор команд состоит из серий коротких текстовых строк, которые объединяют вместе, чтобы сформировать полные команды операций, таких как набор номера, начала соединения или изменения параметров подключения.

Для того, чтобы модем распознал эти команды, они должны быть записаны в специфической форме. Каждая команда всегда начинается буквами AT, дополненных одной или больше командой и завершаемой в конце сочетание CR&LF. Команды воспринимаются модемом только тогда, когда он находится в «командном режиме» или offline.

AT-команды обычно отправляются модему посредством коммуникационного программного обеспечения, но также могут быть введены пользователем вручную, с компьютерной клавиатуры.

Набор команд оказался весьма удачным решением и в качестве задающих установки Hayes-совместимого модема, используется для его оптимального функционирования для тех или иных целей, в различных условиях: при разном состоянии телефонной линии, частотной характеристики [линии](https://ru.wikipedia.org/wiki/%D0%9B%D0%B8%D0%BD%D0%B8%D1%8F), зашумлённости, наличии частых искровых помех и т. д.

Стандартизация набора команд AT-команд выразилась в документе под названием Data Transmission Systems and Equipment — Serial Asynchronous Automatic Dialing and Control, известном как TIA/EIA-602. ITU-T выпустила документ V.250 (так же известный как V.25ter), содержащий все команды TIA/EIA-602 и несколько дополнительных. После чего был выпущен TIA/EIA-602-A содержащий только ссылки на V.250 и небольшую историческую справку.

Для настройки (активации, деактивации и перенастройки установок) модема, АТ-команды могут использовать различные профили модема. Набор команд и архитектура оказались весьма удачными, и неоднократно расширялись, и дополнялись. ETSI выпустил ряд стандартов описывающих управление мобильными телефонами и модемами стандарта [GSM](https://ru.wikipedia.org/wiki/GSM), таких как GSM07.05 и GSM07.07. Некоторые производители коммуникационного оборудования дополняют стандартный набор AT-команд своими собственными расширениями.

Наш модуль будет конфигурироваться в режиме TCP сервера данным набором команд.

1. AT+CWMODE = 1 Устанавливаем режим работы в роли станции
2. AT+CWJAP=”Login”,”PASSWORD” подключение к точке доступа с названием Login и паролем PASSWORD
3. AT+CIFSR Отображается наш ip в сети
4. Настройка режима TCP-сервера
5. AT+CIPMODE=0 устанавливаем режим передачи в data mode
6. AT+CIPMUX=1 делаем доступным множественные соединения
7. AT+CIPSERVER=1,8888 запускаем сервер на порту 8888

После этого подключаемся к модулю и посылаем ему команды, которые он будет дублировать на выходе TX, соединенным с входом RX1 на нашем микроконтроллере.

**2.3.3.2 Описание протокола связи**

Для связи нескольких устройств мы будем использовать WiFi 802.11, разработкой стандартов которого занимается организация IEEE (Institute of Electrical and Electronic Engineers). Рассмотрим далее особенности и возможности данного протокола связи.

История стандарта IEEE 802.11 началась в 1997 году, когда была принята первоначальная спецификация для создания локальных сетей на основе беспроводных технологий. Он обеспечивает обмен информацией с помощью инфракрасного излучения и радиоволн. Теперь IEEE 802.11 - постоянно развивающееся семейство спецификаций, которые всесторонне описывают принципы и параметры беспроводной сети. В дополнение к основному, сюда включаются еще несколько пересмотров основного стандарта и группа нормативных вспомогательных документов.

Основные стандарты Wi-Fi:

* 802.11 - принят в 1997 году. Диапазон 2,4 ГГц, скорость 1 или 2 Мбит / с, с использованием модуляции с прямым расширением спектра (DSSS).
* 802.11a – 1999 г. 5 ГГц, до 54 Мбит / с. Замена DSSS на модуляцию с ортогональным частотным разделением (OFDM) увеличивает степень помехоустойчивости. С другой стороны, это уменьшает радиус покрытия и степень зависимости от преград.
* 802.11b - 1999 2.4 ГГц, до 11 Мбит / с. Утвержденный одновременно со стандартом 802.11a, этот протокол является результатом эволюционного развития базовой спецификации. Уже утвержденные технологии позволили реализоваться на рынке предложения и пользоваться довольно высоким спросом. Именно эта редакция стала называться WiFI.
* 802.11g - 2003 2,4 ГГц, до 54 Мбит / с. Преимущества технологии OFDM и разрешение Федеральной комиссии по связи (FCC) использовать ее в диапазоне 2,4 ГГц для разработки новой спецификации, которая сочетает в себе лучшие решения предыдущих стандартов. Он обратно совместим с 802.11b, но имеет более высокую помехозащищенность.
* 802.11y-2008 3,65 ГГц, до 54 Мбит / с. Развитие не получило.
* 802.11n - 2009 2,4 и 5 ГГц. Дальнейшая разработка спецификации 802.11g обратно совместима со всеми предыдущими стандартами. В этом стандарте были реализованы новые технологии, такие как, пространственное мультиплексирование и многолучевое отражение. Это позволило одновременно передавать несколько потоков данных при максимальной скорости однопотоковой передачи в 150Мбит/с. Использование WiFi из разных поколений приводит к автоматическим ограничениям скорости на уровне самого медленного объекта.
* 802.11ac - 2014 5 ГГц. Следующее эволюционное издание стандарта. В дополнение к утроенной скорости обмена, технология MU-MIMO позволяет вам общаться с несколькими «партнерами» и поддерживает до 8 потоков. Полоса пропускания была увеличена до 160 МГц путем объединения нескольких частотных каналов, теоретически достижимое значение равно 7 Гбит / с. В настоящее время выпускаются устройства с одно-, двух- и трехпотоковой антенной конфигурацией скоростей передачи 450, 900 и 1300 Мбит / с, соответственно.
  + 1. **ST-link/V2**

ST-LINK/V2 – внутрисхемный программатор/отладчик для микроконтроллеров серии STM8 и STM32 производства фирмы STMicroelectronics. Отладчик подключается к отладочным платам посредством стандартного JTAG/SWD интерфейса (микроконтроллеры на базе ядра STM32) или посредством SWIM-интерфейса (для микроконтроллеров семейства STM8).

Его особенностями являются

• программирование Flash-микроконтроллеров серий STM8 и STM32;   
• использование стандартного ARM-совместимого JTAG-разъема для STM32;   
• поддержка JTAG интерфейсом целевого напряжения 1,65 – 3,6 В;   
• поддержка всех возможностей SWIM интерфейса: режимы пониженной и повышенной скорости, подключение по стандарту ERNI, штыревой разъем с шагом 2,54 мм, поддержка целевого напряжения от 1,65 В до 5,5 В;   
• для внутрисхемной отладки микроконтроллеров семейства STM8 используется ST Visual Develop – STVD (версии 4.1.0 или более поздней);   
• для внутрисхемного программирования микроконтроллеров семейства STM8 используется ST Visual Program – STVP (версии 3.1.0 или более поздней);   
• для программирования и отладки микроконтроллеров семейства STM32 ST-LINK/V2 может использовать следующие приложения: Atollic toolset TrueSTUDIO (не ниже версии 1.0), IAR toolset EWARM (не ниже версии 5.30), Keil toolset ARM-MDK (не ниже версии 3.3) и TASKING;   
• поддержка режима самообновления (DFU);   
• USB Full Speed 2.0 интерфейс для подключения к ПК;   
• отсутствие необходимости во внешнем питании эмулятора – питание от USB-порта;   
• напряжение питания +5В.

* 1. **Создание программного обеспечения для управляющей системы**

Наше программное обеспечение напрямую зависит от нужного нам функционала. Следовательно, нам стоит снова обратить внимание на него.

Итак, функционал:

1. Включение/выключение освещения

2. Регулировка яркости

3. Автовключение/автовыключение света в зависимости от общего уровня освещенности

Первый пункт в списке не вызывает никаких вопросов или сложностей. Должен быть какой-то сигнал, при получении которого наша управляющая система поменяет состояние системы на противоположенное (включенное освещение выключит и наоборот). Функция из второго пункта может быть воплощена в жизнь двумя путями: аппаратно или программно, но в первой главе работы мы уже пришли к выводу, что более предпочтителен второй путь. Для этого нам потребуется далее рассмотреть метод широтно-импульсной модуляции. Ну, а для последнего пункта у нас есть отдельный модуль KY-018, подключенный ко входу АЦП микроконтроллера. Обрабатывая информацию от датчика, микроконтроллер будет принимать решение о включении и выключении освещения. Полный код программы находится в листинге 1.

**2.4.1 Широтно-импульсная модуляция**

Для начала ответим на вопрос: «Каким же образом ШИМ поможет изменять уровень яркости освещения?».

Человеческий глаз - сложная вещь. В нем есть палочки и колбочки, а яркость, с которой мы наблюдаем объекты, зависит от количества фотонов, которые упали на них. Особенность заключается в том, что «оцифровка» числа фотонов не происходит сразу. Они работают как сумматор, то есть накапливают «заряд», и через определенный промежуток времени показания снимаются. Это называется инерцией человеческого глаза. Другими словами, если объект мерцает быстрее, чем происходит чтение, мы просто не замечаем мерцание. Время «регистрации» фиксировано (варьируется от человека к человеку), что означает, что фиксированный количество фотонов (если мы возьмем конкретный источник света) может упасть на колбу / палочку. Если половина этого времени светодиод будет гореть, а другую - нет - тогда колба / палочка будет получать вдвое больше фотонов, чем если бы светодиод горел. Приблизительная частота, с которой средний человек не замечает мерцание, составляет 50 Гц.

Частота (период) фиксированна, а вот заполнение (англ. duty) может меняться. Пример разных уровней заполнения на рисунке Б.4 в приложении Б.

В микроконтроллерах широтно-импульсная модуляция реализуется с помощью таймеров. В режиме ШИМ для нас критичны две величины: одна отвечает за период (TIM\_Period), а вторая за заполнение (TIM\_Pulse). Счетчик стартует с «0» (на выходе низкий уровень) и считает до TIM\_Period, затем перезагружается и начинает всё сначала. До момента, когда таймер досчитает до потолка, он пройдет промежуточное значение, которое мы задали, указав TIM\_Pulse. Как только таймер это сделает, он может переключить состояние ножки на противоположное состояние. Т.е. нам необходимо выполнить несколько действий:

* Включаем тактирование порта и таймера.
* Настраиваем ножку.
* Настраиваем таймер.
* Включаем и настраиваем режим ШИМ выбранного таймера.

Непосредственно уровень яркости будет регулироваться значение переменной TIM\_Pulse. Она может принимать значения от 0 до 255(0x00-0xFF), где 0 соответствует минимальной яркости, а 255, в свою очередь, максимальную. Можно разделить этот диапазон на 5 режимов с разницей в 64. Т.е. Нам потребуется всего лишь 5 сигналов для включения определенного уровня яркости.

* 0 – 0x00
* 64 – 0x40
* 128 – 0x80
* 192 – 0xC0
* 255 – 0xFF

Когда микроконтроллер получит какой-то из этих сигналов, то он присвоит [соответствующ](https://ru.wiktionary.org/wiki/%D1%81%D0%BE%D0%BE%D1%82%D0%B2%D0%B5%D1%82%D1%81%D1%82%D0%B2%D1%83%D1%8E%D1%89%D0%B8%D0%B9)ее значение переменной TIM\_Pulse и уровень яркости измениться.

**2.4.2 Принцип работы АЦП в STM32**

АЦП преобразует аналоговый сигнал в цифровой код.

Принцип оцифровки очень прост: входное напряжение сравнивается с опорными напряжениями V\_REF- и V\_REF+:

* V\_REF- нужно подключить к земле
* V\_REF+ по желанию: либо к питанию процессора (оно плавающее и шумное, поэтому этот вариант годится только для неточных измерений), либо к внешнему источнику опорного напряжения (ИОН)

Впрочем, есть возможность программно настроить эти ноги на прямое соединение с землёй и питанием.

Входное напряжение V\_In будет измерено относительно V\_REF- и V\_REF+, и результат преобразования сложен в выходной регистр в такой пропорции (см. таблицу 1)

Таблица 2.1 Принцип работы АЦП

|  |  |
| --- | --- |
| Напряжение | Результат |
| V\_Ref- |  |
| V\_In |  |
| V\_Ref+ | 4096 |

К примеру, 1.2 В при питании АЦП от 3.3 В преобразуются в 1490.

Вот некоторые характеристики аналого-цифрового преобразователя в STM32f10x:

* АЦП является 12-ти битным
* Возможна генерация прерывания по окончанию преобразования, по окончанию преобразования с инжектированного канала, а также возможно прерывание от Analog Watchdog (что это такое расскажу чуть ниже)
* Возможно одиночное преобразование и преобразование в непрерывном режиме
* Самокалибровка
* Запуск преобразования от внешнего события
* Работа с ПДП (DMA, прямой доступ к памяти)
* скорость оцифровки — до 0.9 MSPS с программируемым временем захвата и преобразования
* режим сканирования входов по списку

Каналы АЦП делятся на регулярные и инжектированные. Причем, если запустить измерение инжектированных каналов, то измерение регулярных будет приостановлено. Использование регулярного метода опроса состоит в том, что АЦП опрашивает по очереди некоторый заранее настроенный список каналов, после каждого опроса результат записывается в один и тот же регистр. Это означает, что нужно своевременно забирать результат преобразования из этого регистра, в противном случае результат будет перетираться. Совсем иначе дело обстоит с инжектированными каналами. В случае использования этого метода опроса можно записывать результат измерения каждого канала АЦП в свой отдельный регистр ничего не перекрывая.

АЦП STM32 обладает Analog watchdog. Его предназначение в том, чтоб подать сигнал в случае если напряжение на определённом канале выйдет за допустимый диапазон, это позволяет сэкономить процессорное время за счёт того, что нам не придётся программно заставлять АЦП производить измерение и потом сравнивать полученное значение с пороговыми. Структура схема АЦП в STM32 изображена на рисунке Б.5 в приложении Б.

Для того чтоб прочитать данные с канала мы должны сделать следующее:

1) Включить тактирование порта А

2) Настроить ногу PA1 как вход без подтяжки (по умолчанию она уже в таком состоянии)

3) Включить тактирование АЦП

4) Начать калибровку и дождаться её завершения

5) Добавить канал ADC1 в состав инжектированной группы

6) Выбрать источником запуска бит JSWSTART

7) Активировать режим непрерывного преобразования

8) Включить АЦП

9) Запустить преобразование

10) Дождаться завершения первого преобразования

11) Прочитать результат

Далее, экспериментально найдем 2 пороговых значения освещенности и используем их для автоматического переключения режима освещения на противоположенное.

**2.4.3 Интерфейс UART**

Микроконтроллер в данной работе общается с модулем esp8266-01 при помощи протокола UART (Universal asynchronous receiver/transmitter) поэтому я считаю необходимым дать краткое описание данного протокола.

Изначально интерфейс UART появился в США как средство для передачи телеграфных сообщений, и рабочих бит там было пять (как в азбуке Морзе). Для передачи использовались механические устройства. Потом появились компьютеры, и коды ASCII, которые потребовали семь бит. В начале 60-х на смену пришла всем известная 8-битная таблица ASCII, и тогда формат передачи стал занимать полноценный байт, плюс управляющие три бита. Наиболее известен из семейства UART протокол RS-232.

Основные рабочие линии у нас – RXD и TXD. Передающая линия – TXD (Transmitted Data), а порт RXD (Received Data) – принимающая.  
Эти линии СОМ-порта задействованы при передаче без аппаратного управления потоком данных. При аппаратном потоке задействованы еще дополнительные интерфейсные линии (DTS, RTS и пр.). Выход передатчика TX соединен с входом приемника RX и наоборот. Электрический принцип работы RS-232 отличается от стандартной 5-вольтовой TTL логики. В этом протоколе логический нуль лежит от +3 до +12 вольт, а единица от -3 до -12, соответственно. Промежуток от -3 до +3 вольт считается зоной неопределенности. Все напряжения указаны относительно корпуса компьютера, или земли. Прием сигнала по RS-232 на рисунке Б.6 в приложении Б.

Такая большая амплитуда рабочих напряжений, целых 24 вольта, нужна в первую очередь для помехоустойчивости линий связи. По стандарту, длина кабеля может быть 15 м. Электрические параметры RS-232 – это главная отличительная характеристика RS-323 среди других протоколов семейства.  
Следующие характеристики – формат посылки и скорость передачи данных – полностью применимы ко всем видам UART и обеспечивают их совместимость через несложные схемы сопряжения. Стандартная посылка занимает 10 бит. Но правило это распространяется только на стандартные настройки СОМ-порта. В режиме простоя, когда по линии ничего не передается, она находится в состоянии логической единицы, или -12 вольт. Начало передачи обозначают передачей стартового бита, который всегда равен нулю. Затем идет передача восьми бит данных. Завершает посылку бит четности и стоповый бит. Бит четности осуществляет проверку переданных данных. Стартовый бит говорит нам, что пересылка данных завершена. Надо отметить, что STOP-бит может занимать 1, 1.5, и 2 бита. Стоповый бит, как и стартовый, равен нулю. Линейка скоростей СОМ-порта стандартизирована. Как правило, все устройства работают на трех стандартных скоростях: 9600, 19200, 115200. Но возможны другие варианты, даже использование нестандартных скоростей или скорости, меняющейся во времени. Сигнал UART на экране осциллографа на рисунке Б.7 в приложении Б.

* 1. **Вывод по разделу 2**

В данном разделе были разобраны все проблемы, и найдены решения к ним. Также, описаны и изучены протоколы связи, используемые в данной системе. Мы разработали общую схему устройства, набор рабочих кодов, и, наконец, написан рабочий код для управляющего микроконтроллера.

**3.Экономическая часть**

**3.1 Анализ существующих систем на рынке**

Беглый поиск схожих систем на рынке выдает несколько вариантов. Например, Xiaomi Philips EyeCare Smart Ceiling Lamp – это умный потолочный светильник, который является частью системы умного дома «MiHome» от Xiaomi и управляется с помощью смартфона с предустановленным приложением. Источником света выступают 64 ярких светодиода, совместная яркость которых может достигать 3000 Люмен. Xiaomi Philips EyeCare Smart Ceiling Lamp обладает широким выбором режимов работы и имеет на своем борту встроенные модули Wi-Fi и Bluetooth. В приложении можно включать и выключать светильник, регулировать яркость/температуру света и сохранять параметры, устанавливать время активации «спящего режима», расписание работы светильника и таймер выключения. Кроме того, приложение предусматривает 4 готовых сценария работы: ночник, теплый свет, максимальная яркость и просмотр ТВ. Умный светильник также работает в режимах Auto CCT и Smart Midnight Light: в зависимости от времени суток устройство регулирует яркость освещения, а также автоматически включается при наступлении ночи. Т.е. функционал данной системы практически полностью сопоставим с функционалом созданного устройства при цене в 6390 рублей за каждый светильник. Цену, созданного нами устройства, рассмотрим чуть позже.

Другим примером может служить Умная система освещения Nanoleaf Aurora Smarter Kit. Это набор из 9 светодиодных RGBW панелей, управляемых через Wi-Fi с любого android устройства версии 5.1 и выше. Цвет каждой панели Aurora, яркость и контрастность ее свечения устанавливается в приложении Nanoleaf Smarter Series. Умная система освещения Nanoleaf Aurora Smarter Kit имеет сертификацию Apple Homekit и ей можно управлять с помощью голосовых команд через помощника Siri для iOS устройств. Да, у этой системы есть некоторые функциональные преимущества. Например, голосовое управление или способность менять цвет освещения, но и её стоимость 16990 рублей.

Далее, необходимо провести расчеты стоимость созданного устройства.

**3.2 Расчет себестоимости**

Перечислим все узлы и детали, используемые в нашем готовом продукте:

* Wi-fi модуль esp8266-01
* Микроконтроллер STM32F100
* Светочувствительный модуль KY-018
* Светодиодная лента
* Полевой транзистор IRF520

Соберем все данные о стоимости данных элементов в одной таблице:

Таблица 3.1 Стоимость деталей

|  |  |  |  |
| --- | --- | --- | --- |
| Деталь | Кол-во | Цена(руб.) | Место покупки |
| Esp8266-01 | 1 | 144.45 | aliexpress.com. |
| KY-018 | 1 | 20.30 | aliexpress.com |
| STM32F100 | 1 | 69.16 | aliexpress.com |
| Светодиодная лента | 1 м. | 288.16 | aliexpress.com |
| IRF520 | 1 | 33 | chipdip.ru |

Итого 555.07 рублей. Полученная стоимость не идет ни в какое сравнение с ценами готовых продуктов из предыдущей главы.

**3.3 Выводы по разделу 3**

Сравнив рыночные цены на системы умного освещения и нашу финальную стоимость, мы можем увидеть, что использование разработанной системы является экономически более обоснованным, чем использование рыночных систем от известных компаний. А сравнение функционала, в свою очередь, показывает, что разработанное устройство практически не отстает от рыночных предложений.

**Заключение**

В процессе разработки дипломной работы получены следующие результаты:

* Рассмотрены различные варианты осветительных систем
* Выбрана оптимальная конфигурация такой системы
* Рассмотрены и проанализированы готовые варианты аналогичных систем
* Разработана собственная система управления освещением
* Подобран собственный набор управляющих сигналов

Таким образом, спроектированная система удовлетворяет всем начальным условиям, и обладает при этом дешевизной, простотой реализации и эксплуатации.

**Список литературы**

Основная литература

1. Бройдо, В. Л. Вычислительные системы, сети и телекоммуникации: Учебник для вузов / В. Л. Бройдо. – Санкт-Петербург : ПИТЕР, 2004. – 702 c.
2. Предко, М. Руководство по микроконтроллерам. Том 1 / М. Предко. – Москва : Постмаркет, 2001. – 416 c.
3. Предко, М. Руководство по микроконтроллерам. Том 2 / М. Предко. – Москва : Постмаркет, 2001. – 418 c.

Дополнительная литература

1. Joseph, Yiu. the Definitive Guide to the ARM Cortex-M3 / Yiu. Joseph. – Burlington : Elsevier Inc, 2007. – 531 c.
2. Trevor, Martin. The Insider’s Guide To The STM32 ARM®Based Microcontroller / Martin. Trevor. – Coventry : Hitex (UK) Ltd., 2008. – 96 c.

**Приложение А. Схем**

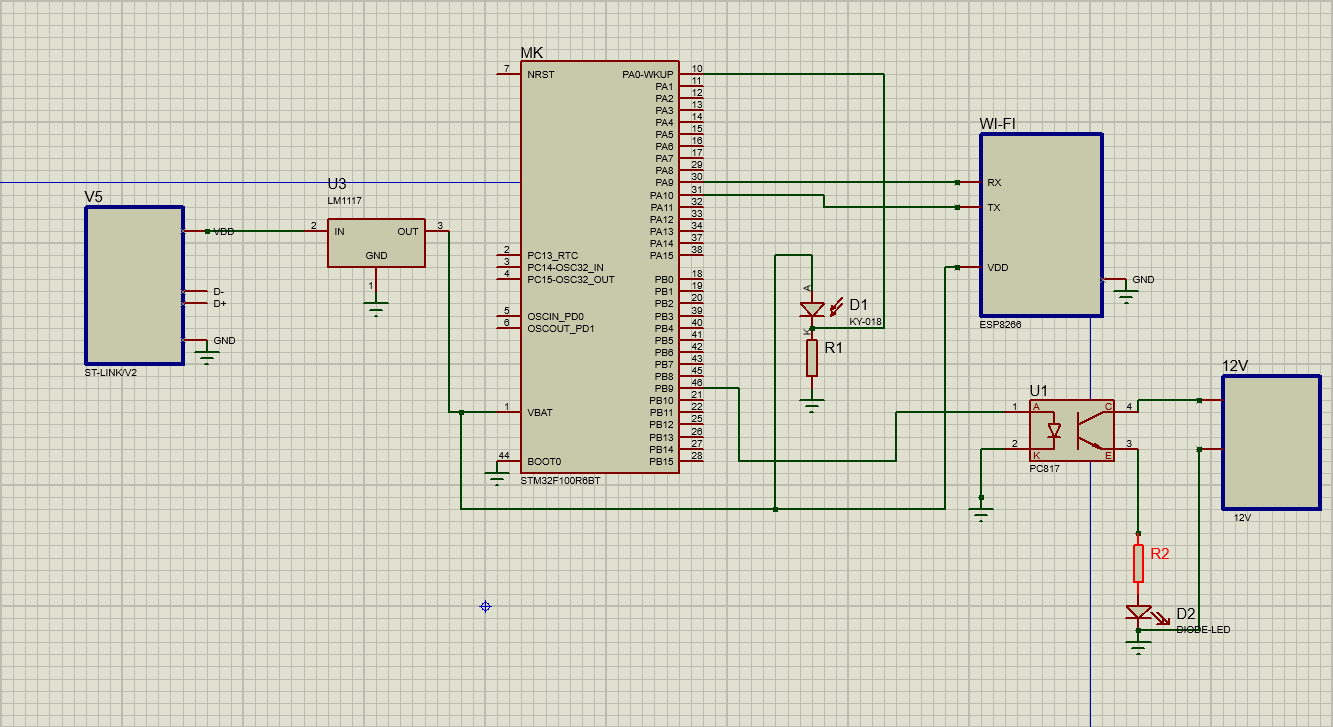


Схема А.1. Общая схема устройства

**Приложение Б. Рисунки**

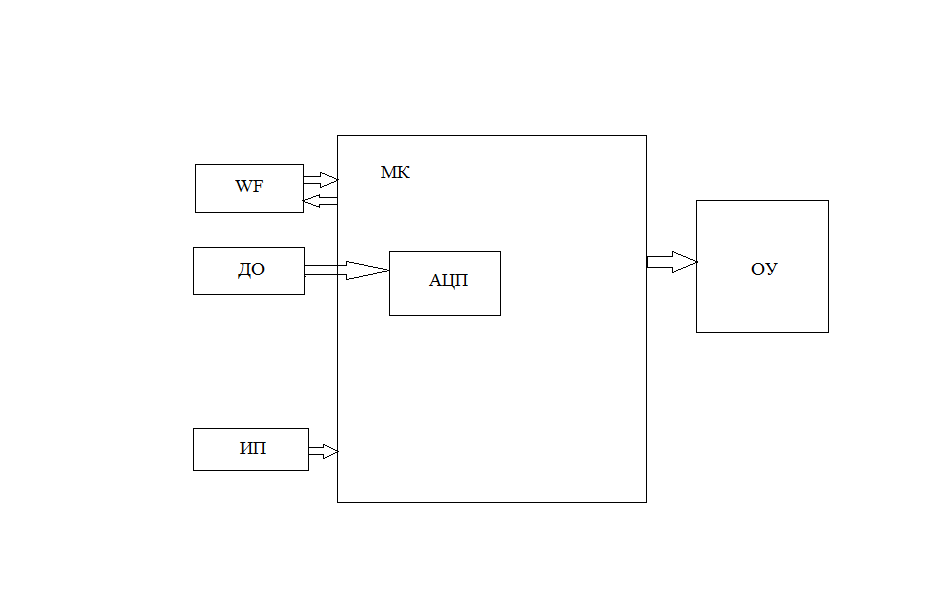


Рисунок Б.1. Структурная схема

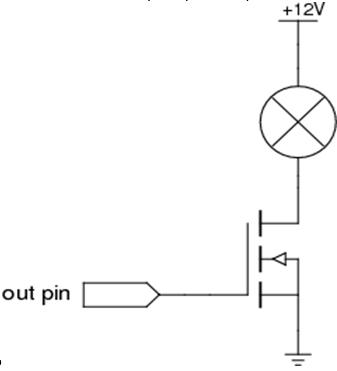
****

Рисунок Б.2.

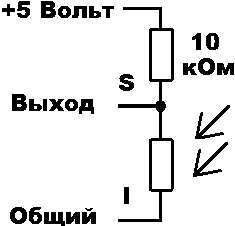


Рисунок Б.3. Модуль KY-018

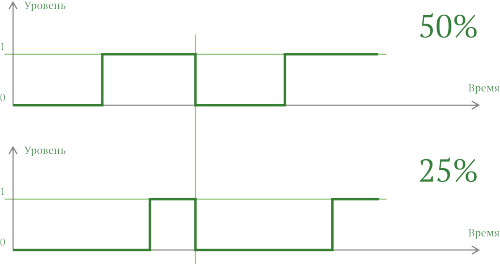


Рисунок Б.4. ШИМ

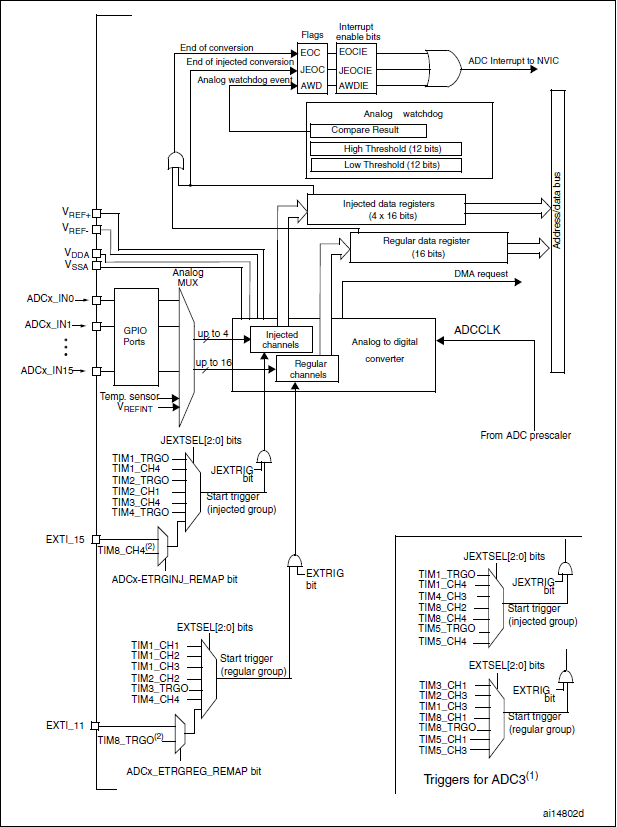


Рисунок Б.5. Схема АЦП в stm32 (http://st.com/)

Рисунок Б.6. Сигнал в UART (http://habr.com)

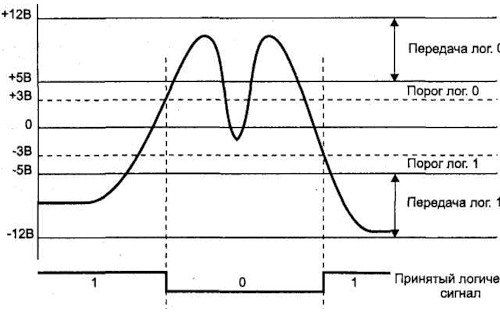


Рисунок Б.7. Передача информации через UART (http://habr.com)

**Приложение В. Листинг**

Листинг 1. Код прошивки микроконтроллера на языке С

#include <stm32f10x.h>

#include <stm32f10x\_gpio.h>

#include <stm32f10x\_rcc.h>

#include <stm32f10x\_tim.h>

#include <misc.h>

void init\_pwm();

void init\_USART1();

void init\_Usart1\_Nvic();

void adc\_init();

uint16\_t get\_adc\_value();

//n-длина команды

void Pause()

{

for(int i=0;i<100000;i++)

{}

}

void send\_command(uint8\_t command[], int n)

{

for(int i=0;i<n;i++)

{

USART\_SendData(USART1,command[i]);

Pause();

}

}

enum {TX=GPIO\_Pin\_9,RX=GPIO\_Pin\_10};

/\*41 54 2B 43 49 50 4D 55 58 3D 31 0D 0A = AT+CIPMUX=1 Режим множественных подключений \*/

/\*41 54 2B 43 49 50 53 45 52 56 45 52 3D 31 2C 38 38 38 38 0D 0A = AT+CIPSERVER=1,8888 Запустить сервер на порту 8888 \*/

//41 54 2B 43 49 46 53 52 0D 0A = AT+CIFSR Получить ip

uint8\_t cipmux[] = {0x41,0x54,0x2B,0x43,0x49,0x50,0x4D,0x55,0x58,0x3D,0x31,0x0D,0x0A};

uint8\_t cipserver[] = {0x41,0x54,0x2B,0x43,0x49,0x50,0x53,0x45,0x52,0x56,0x45, 0x52,0x3D,0x31,0x2C,0x38,0x38,0x38,0x38,0x0D,0x0A};

TIM\_OCInitTypeDef pwm;

int main(void)

{

init\_pwm();

init\_USART1();

init\_Usart1\_Nvic();

adc\_init();

//первичная настройка esp8266

send\_command(cipmux,sizeof(cipmux));

Pause();

Pause();

send\_command(cipserver,sizeof(cipserver));

uint16\_t adc;

while(1)

{

adc=get\_adc\_value();

}

}

//Инициализация шим

void init\_pwm(void)

{

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOC|RCC\_APB2Periph\_AFIO,ENABLE);

RCC\_APB1PeriphClockCmd(RCC\_APB1Periph\_TIM3,ENABLE);

GPIO\_PinRemapConfig(GPIO\_FullRemap\_TIM3,ENABLE);

//инициализируем ножку как выход с альтернативной функцией

GPIO\_InitTypeDef gpio;

gpio.GPIO\_Pin = GPIO\_Pin\_8;

gpio.GPIO\_Mode = GPIO\_Mode\_AF\_PP;

gpio.GPIO\_Speed = GPIO\_Speed\_2MHz;

GPIO\_Init(GPIOC,&gpio);

//задаем параметры таймера 3ch3 на этой ножке

TIM\_TimeBaseInitTypeDef t3;

TIM\_TimeBaseStructInit(&t3);

t3.TIM\_Period = 200;

t3.TIM\_Prescaler = 2400;

t3.TIM\_ClockDivision = 0;

t3.TIM\_CounterMode = TIM\_CounterMode\_Up;

TIM\_TimeBaseInit(TIM3,&t3);

// задаем параметры шим

pwm.TIM\_OCMode =T IM\_OCMode\_PWM1;

pwm.TIM\_OutputState = TIM\_OutputState\_Enable;

pwm.TIM\_Pulse = 0;

pwm.TIM\_OCPolarity = TIM\_OCPolarity\_High;

TIM\_OC3Init(TIM3,&pwm);

TIM\_OC3PreloadConfig(TIM3,TIM\_OCPreload\_Enable);

TIM\_ARRPreloadConfig(TIM3,ENABLE);

TIM\_Cmd(TIM3,ENABLE);

}

void init\_USART1()

{

//инициализация приема с USART(PA10)

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOA | RCC\_APB2Periph\_USART1 | RCC\_APB2Periph\_AFIO,ENABLE);

GPIO\_InitTypeDef gpio;

gpio.GPIO\_Mode = GPIO\_Mode\_IN\_FLOATING;

gpio.GPIO\_Pin = RX;

GPIO\_Init(GPIOA,&gpio);

//инициализация выхода на USART(PA9)

gpio.GPIO\_Mode = GPIO\_Mode\_AF\_PP;

gpio.GPIO\_Speed = GPIO\_Speed\_50MHz;

gpio.GPIO\_Pin = TX;

GPIO\_Init(GPIOA,&gpio);

//инициализация USART

USART\_InitTypeDef usart;

usart.USART\_BaudRate = 115200;

usart.USART\_WordLength = USART\_WordLength\_8b;

usart.USART\_StopBits = USART\_StopBits\_1;

usart.USART\_Parity = USART\_Parity\_No;

usart.USART\_HardwareFlowControl =

USART\_HardwareFlowControl\_None;

usart.USART\_Mode = USART\_Mode\_Rx | USART\_Mode\_Tx;

USART\_Init(USART1,&usart);

USART\_Cmd(USART1,ENABLE);

//прерывание

USART\_ITConfig(USART1,USART\_IT\_RXNE,ENABLE);

NVIC\_EnableIRQ(USART1\_IRQn);

}

//разрешаем и задаем приоритет прерыванию по прибытию //информации в USART1

void init\_Usart1\_Nvic()

{

NVIC\_InitTypeDef nvic;

nvic.NVIC\_IRQChannel = USART1\_IRQn;

nvic.NVIC\_IRQChannelPreemptionPriority = 1;

nvic.NVIC\_IRQChannelSubPriority = 1;

nvic.NVIC\_IRQChannelCmd = ENABLE;

NVIC\_Init(&nvic);

}

//обработчик прерывания USART1

void USART1\_IRQHandler()

{

uint8\_t data;

//Проверяем, действительно ли прерывание вызвано приемом нового байта

if (USART\_GetITStatus(USART1, USART\_IT\_RXNE) != RESET)

{

USART\_ClearITPendingBit(USART1,USART\_IT\_RXNE);

data=USART\_ReceiveData(USART1);

//обработка внешних сигналов

if(data == 0x00) pwm.TIM\_Pulse = 0;

if (data == 0x40) pwm.TIM\_Pulse = 64;

if(data == 0x80) pwm.TIM\_Pulse = 128;

if(data == 0xC0) pwm.TIM\_Pulse = 192;

if(data == 0xFF) pwm.TIM\_Pulse = 255;

if(data == 0x01) GPIO\_ResetBits(PORTC,GPIO\_Pin\_9);

if(data == 0x02) GPIO\_SetBits(PORTC,GPIO\_Pin\_9);

}

}

void adc\_init()

{

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_ADC1, ENABLE);

// настройки ADC

ADC\_InitTypeDef ADC\_InitStructure;

ADC\_StructInit(&ADC\_InitStructure);

ADC\_InitStructure.ADC\_Mode = ADC\_Mode\_Independent;

// режим работы - одиночный, независимый

ADC\_InitStructure.ADC\_ScanConvMode = DISABLE;

// просто измерить один канал

ADC\_InitStructure.ADC\_ContinuousConvMode = DISABLE;

ADC\_InitStructure.ADC\_ExternalTrigConv = ADC\_ExternalTrigConv\_None;

// без внешнего триггера

ADC\_InitStructure.ADC\_DataAlign = ADC\_DataAlign\_Right; //выравнивание битов результат - прижать вправо

ADC\_InitStructure.ADC\_NbrOfChannel = 1;

//количество каналов - одна штука

ADC\_Init(ADC1, &ADC\_InitStructure);

ADC\_Cmd(ADC1, ENABLE);

// настройка канала

ADC\_RegularChannelConfig(ADC1, ADC\_Channel\_8, 1, ADC\_SampleTime\_55Cycles5);

// калибровка АЦП

ADC\_ResetCalibration(ADC1);

while (ADC\_GetResetCalibrationStatus(ADC1));

ADC\_StartCalibration(ADC1);

while (ADC\_GetCalibrationStatus(ADC1));

}

//получение измерения с ацп

uint16\_t get\_adc\_value()

{

ADC\_SoftwareStartConvCmd(ADC1, ENABLE);

while(ADC\_GetFlagStatus(ADC1, ADC\_FLAG\_EOC) == RESET);

return ADC\_GetConversionValue(ADC1);

}