TALLINNA TEHNIKAÜLIKOOL
INSENERITEADUSKOND
Virumaa kolledž

Valgustuse automatiseerimine ja kaugjuhtimine eluruumides

TOOTMISE AUTOMATISEERIMINE ÕPPEKAVA LÕPUTÖÖ

Üliõpilane: Denis Samokhvalov

Üliõpilaskood: 178665RDDR

Juhendaja: Sergei Pavlov, Lektor

Kohtla-Järve, 2021

TALLINNA TEHNIKAÜLIKOOL
INSENERITEADUSKOND
Virumaa kolledž

Автоматизация и удаленный контроль освещения в жилых помещениях

TOOTMISE AUTOMATISEERIMINE ÕPPEKAVA LÕPUTÖÖ

Üliõpilane: Denis Samokhvalov
Üliõpilaskood: 178665RDDR
Juhendaja: Sergei Pavlov, Lektor

Kohtla-Järve, 2021

AUTORIDEKLARATSIOON

Olen koostanud lõputöö iseseisvalt.

Lõputöö alusel ei ole varem kutse- või teaduskraadi või inseneridiplomit taotletud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on viidatud.

“..” ……. 2022.

Autor: .....

/ allkiri /

Töö vastab rakenduskõrgharidusõppe lõputööle/magistritööle esitatud nõuetele
“..” juuni 2022.

Juhendaja: .....

/ allkiri /

Kaitsmisele lubatud “..” …….. 2022.

Kaitsmiskomisjoni esimees Sergei Pavlov

/ nimi ja allkiri /

LIHTLITSENTS LÕPUTÖÖ ÜLDSUSELE KÄTTESAADAVAKS TEGEMISEKS JA REPRODUTSEERIMISEKS

Mina Denis Samokhvalov (sünnikuupäev: 27.02.1983)

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose elamute ruumides valgustuse automaatika ja kaugjuhtimispult, mille juhendaja on …… .

1.1. reprodutseerimiseks säilitamise ja elektroonilise avaldamise eesmärgil, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;

1.2. üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.

2. Olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.

3. Kinnitan, et lihtlitsentsi andmisega ei rikuta kolmandate isikute intellektuaalomandi ega isikuandmete kaitse seadusest ja teistest õigusaktidest tulenevaid õigusi.

TalTech Inseneriteaduskond Virumaa kolledž

LÕPUTÖÖ ÜLESANNE

Üliõpilane: Denis Samokhvalov, RDDR 178665

Õppekava, peaeriala: Tootmise automatiseerimine

Juhendaja(d): lektor, ……..., ……..@taltech.ee

Konsultant: -

Lõputöö

teema:

(eesti keeles) **Elamute ruumides valgustuse automaatika ja kaugjuhtimispult.**
(inglise keeles) **Automation and remote control of lighting in residential premises.**

Lõputöö põhieesmärgid:

1. programmeerida elamute ruumides valgustuse algoritmi
2. Luua rakenduse kaugjuhtimispuldi algoritmi jaoks.
3. Luua rakenduses märguande süsteem

Lõputöö etapid ja ajakava:

Nr	Ülesande kirjeldus	Tähtaeg
1.	Automaatse valgustuse algoritm kirjeldus	20.02.21
2.	Varustuse valik	01.03.21
3.	Kommunikatsiooni korraldamine mikrokontrolleri ja rakenduse vahel	15.03.21
4.	Mikrokontrolleri programmeerimine	30.03.21

Töö keel: vene keel **Lõputöö esitamise tähtaeg:** “..” …….. 2021. a

Üliõpilane: D.Samokhvalov “..”……. 2021a

/allkiri/

Juhendaja: ……. “..”…… 2021a

/allkiri/

Konsultant: - “.....”..... 20.....a

/allkiri/

Programmijuht: ……. “..”……. 2021a

/allkiri

СОДЕРЖАНИЕ

ПРЕДИСЛОВИЕ 10

АББРЕВИАТУРЫ И ТЕРМИНЫ 11

ВВЕДЕНИЕ 13

1. СУЩЕСТВУЮЩИЕ СИСТЕМЫ УПРАВЛЕНИЯ ОСВЕЩЕНИЕМ 14

1.1 ДИММИРОВАНИЕ 14

1.2 Протокол 0–10В 14

1.3 Протоколы AMX192 и D54 15

1.4 Протокол DSI 15

1.5 Протокол DALI 16

1.6 ZigBee 17

1.7 Открытый стандарт автоматизации инженерных систем – KNX 20

1.8 LoRaWAN в управлении уличным освещением 21

1.9 Управление освещением по PLC 24

2. ПРОЕКТ 25

2.1 Объект автоматизации 25

2.2 Техническое задание 26

2.3 Система 26

3. ОБОРУДОВАНИЕ 27

3.1 Датчики 27

3.1.1 Фоторезистор 27

3.1.2 Пир датчик 28

3.1.3 Датчик тока 29

3.2 Исполнительные механизмы 31

3.2.1 Реле 31

3.2.2 Редуктор 33

3.2.3 Концевой выключатель 34

3.2.4 Микроконтроллер 34
4. РАСПОЛОЖЕНИЕ ДАТЧИКОВ И ИСПОЛНИТЕЛЬНЫХ МЕХАНИЗМОВ ... 37
4.1 Принципиальная электросхема освещения 38
4.2 Принципиальная электросхема жалюзей 39
4.3 Монтаж силовой части системы освещения 40
4.4 Монтажн датчиков и привода жалюзей 41
4.5 Щита автоматики 42
4.6 Питание датчиков и ESP32 43
4.7 Принципиальная схема автоматики 44
5. АЛГОРИТМ ПРОГРАММЫ 45
6. СРЕДА ПРОГРАММИРОВАНИЯ 47
6.1 Среда программирования Arduino IDE 48
6.2 Установка платы ESP32 в IDE Arduino 48
7. НАПИСАНИЕ ПРОГРАММЫ 49
7.1 Язык программирования 49
7.2.1 Присвоение имен, выводам микроконтроллера 50
7.2.2 Объявление переменных 50
7.2.3 Прерывания пир датчиков 51
7.2.4 Инициализация выводов 52
7.2.5 Программа датчиков тока 52
7.2.6 Программа фоторезистора 55
7.3.1 Алгоритм включения выключения света 56
7.3.2 Аварии при включении и выключении света 58
7.3.3 Программа работы жалюзей и аварии работы жалюзей 59
8. УДАЛЕННОЕ УПРАВЛЕНИЕ 60
8.1 Веб – сервер 61
8.2 HTML страница 61

8.3 Защита паролем 64
9. УПРАВЛЕНИЕ ЧЕРЕЗ ГЛОБАЛЬНУЮ СЕТЬ ИНТРНЕТ 65
9.1 ИНТЕРНЕТ ВЕЩЕЙ Arduino IoT Cloud 65
9.2 REMOTEXY 68
ЗАКЛЮЧЕНИЕ 70
KOKKUVÕTE 71
SUMMARY 72
СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 73

ПРЕДИСЛОВИЕ

Дипломная работа была сделана по теме, которую я выбрал сам. На выбор темы повлияла тяжелая экологическая обстановка в мире, обращение глав крупнейших стран мира и президента Эстонии к жителям планеты, с площадок экологических форумов, а также мои моральные принципы и убеждения.

Проект по автоматизации и удаленному контролю освещения в жилых помещениях, это проект, который позволяет экономить электроэнергию и тем самым сохранять природные ресурсы и улучшать экологию.

Автор выражает благодарность: лекторам колледжа, за переданные ими знания, которые помогли создать этот проект. Так же благодарю свою жену, которая узнав, что эта работа позволит не только улучшить экологию в мире и сберечь природные ресурсы, но и сэкономить электроэнергию, а значит и семейный бюджет, окончательно утвердила тему моей дипломной работы.

Результатом проекта автора стала дипломная работа на тему «Автоматизация и удаленное управление освещением в жилых помещениях».

Võtmesõnad: mikrokontroller, rakendus, valgustus, algoritm lõputöö.

АББРЕВИАТУРЫ И ТЕРМИНЫ

AMX192 - аналоговый протокол передачи данных [1].

D54 - аналоговый протокол передачи данных [2].

DSI (Digital Serial Interface) - протокол последовательной передачи цифровых данных [3].

DALI (Digital Addressable Lighting Interface) - цифровой интерфейс освещения с возможностью адресации [4].

ZigBee - протокол беспроводной связи [5].

KNX - Открытый стандарт автоматизации инженерных систем [6].

ETS (Engineering Tool Software/Инструментальный программный пакет для инженерного обеспечения) - это уникальный, универсальный инструментальный программный пакет для разработки и настройки “умной” KNX системы автоматизированного управления и контроля зданий и квартир [7].

CSMA/CA - сетевой протокол [8].

LoRa (Long Range) - технология, преобразующая данные в электромагнитные волны, представляет собой физический уровень протокола [9].

LoRaWAN (Long Range Wide Area Networks) представляет собой беспроводной протокол, соединяющий устройства, работающие на батарейках, к сети интернет [10].

PLC (Power Line Communication) - телекоммуникационная технология, базирующаяся на использовании силовых электросетей для высокоскоростного информационного обмена [11].

Щ.А. – Щит автоматики.

COM – общий контакт реле.

NO - нормально открытый контакт.

NC – нормально закрытый контакт.

IDE (Integrated Development Environment) – это интегрированная, единая среда разработки, которая используется разработчиками для создания различного программного обеспечения [12].

SRAM(static random access memory) - статическая память с произвольным доступом (энергозависимый тип памяти) [13].

HTML (Hypertext Markup Language) - это код, который используется для структурирования и отображения веб-страницы и её контента [14].

GET - метод для чтения данных с сайта [16].

ASCII (American standard code for information interchange) — название таблицы, в которой некоторым распространённым печатным и непечатным символам сопоставлены числовые коды [18].

ВВЕДЕНИЕ

Задача данной дипломной работы состоит в автоматизации и удаленном управление освещения в жилых помещениях, для эффективного и экономного расхода электроэнергии при её повседневном использовании в бытовых условиях. В этой работе будут рассмотрены уже существующие системы автоматизации освещения и методы его удаленного контроля, проведен анализ их плюсов и минусов, а также создана и описана собственная система на примере стандартной квартиры.

В проекте собственной системы, рассмотрен и аргументирован выбор оборудования, языка программирования алгоритма, способ визуализации системы и инструменты удаленного управления, метод монтажа оборудования. При программировании алгоритма учтены возможные аварийные ситуации, блокировки системы при возникновении опасностей или выхода из строя какого-либо элемента оборудования, разработана система оповещения пользователя.

Так как система предназначена для использования в жилых помещениях и ее пользователи не обязаны разбираться в автоматизации, а автоматика как в принципе и любое другое оборудование имеет свойство выходить из строя, система, разработанная автором интегрирована в обычную привычную нам механическую систему управления светом и электричеством в квартире. Для максимального комфорта предусмотрена параллельная работа автоматического, удаленного и ручного управления освещением в жилом помещение.

По итогам работы сделан анализ эффективности системы, протестирована надежность и удобство её работы в реальных условиях, рассчитана стоимость оборудования и программного обеспечения, проведено сравнение с существующими системами.

1. СУЩЕСТВУЮЩИЕ СИСТЕМЫ УПРАВЛЕНИЯ ОСВЕЩЕНИЕМ

Система управления освещением - это интеллектуальная сеть, которая позволяет обеспечить нужное количество света, где и когда это необходимо [1].

1.1 ДИММИРОВАНИЕ

Диммирование представляет собой процесс регулирования яркости освещения. Функционал современных моделей довольно обширен и позволяет плавно или автоматически включать/выключать свет, затемнять в соответствии с выбранным режимом, дистанционно управлять всем процессом [2].

Плюсы диммирования

1) Позволяет создать мягкий приглушенный свет или, напротив, акцентировать внимание на конкретных зонах, чего нельзя добиться со стандартными выключателями [2].
2) Чем меньше яркость, тем менее интенсивно эксплуатируется осветительный прибор, соответственно продлевается срок его службы [2].
3) Уменьшается потребление электроэнергии, выделение тепла [2].

Благодаря диммированию можно управлять каждой отдельной группой светильников.

Среди стандартов и протоколов диммирования наиболее распространенными являются 0-10 V, DALI, AMX192 и D54, DSI.

1.2 Протокол 0–10В

Стандарт 0–10В является одним из самых первых и простых протоколов управления освещением. Он был разработан в 2001 [3].

Протокол предоставляет возможность за счет изменения напряжения в диапазоне от 0 до 10 В, изменять яркость осветительного прибора. 0В свет выключен, 10В – 100% яркости. Данный протокол это односторонняя связь между диммером и осветительным устройством.

Основные плюсы аналоговых протоколов это простота и легкость в установке, не высокая стоимость, возможность использовать любой тип кабеля, для управления.

Для каждого канала требуется один кабель и один общий обратный провод. Для управления большим количеством светильников требуется соответствующее количество линий и кабелей, что делает такую систему управления светом громоздкой. Нестабильность при передаче сигнала на большие расстояния [3].

Минусом аналогового протокола является

1) Для управления большим количеством светильников требуется большое количество линий и кабелей, что делает систему громоздкой [3].
2) Нестабильность при передаче сигнала на большие расстояния [3].
3) Падение напряжения.
4) Помехи от силовых кабелей, которые могут влиять на управляющий сигнал [3].

1.3 Протоколы AMX192 и D54

Протоколы AMX192 и D54 разработаны специально для управления освещением на сценах. Оба протокола являются аналоговыми. Принципиальным отличием этих протоколов от протокола 0-10В, это возможность управлять осветительными приборами при помощи одного общего кабеля. Различие протоколов заключается лишь в максимальном количестве каналов, которыми можно управлять (AMX 192 канала и D54 384 канала) и встроенной схемой синхронизации у протокола D54.

1.4 Протокол DSI

DSI (Digital Serial Interface) - протокол последовательной передачи цифровых данных [4].

В 1991 году в первые, удалось разработать цифровое управление освещением. DSI представляет собой 8-битовый протокол, со скоростью передачи данных 960 бит/сек.

Стандарт не является адресным, с его помощью можно управлять группой светильников на одной шине. По сравнению с аналоговым интерфейсом система отличается большей гибкостью в управлении и легкостью внесения изменений [4].

Преимущества протокола DSI:

1) все светильники регулируются одинаково независимо от мощности, расстояния между ними и блоком управления [4].

2) поскольку каждое управляемое устройство подключается не к сети, а непосредственно к контроллеру, то при выходе из строя нужно лишь заменить неработающий прибор, нет необходимости перепрограммировать всю систему [4].
3) цифровой сигнал не чувствителен к помехам и искажению [4].
4) управление освещением осуществляется с помощью специальных панелей или пультов [4].

DSI открыл ряд возможностей, ранее недоступных при протоколе 0-10 В. Он позволил:

1) объединять осветительные устройства в группы и создавать простые сценарии.
2) использовать провода небольшого сечения неограниченной длины.
3) снижать яркость до полного выключения, что позволяет исключить использование выключателей [4].
4) управлять освещением через компьютер.

1.5 Протокол DALI

DALI (Digital Addressable Lighting Interface) - цифровой интерфейс освещения с возможностью адресации [5].

Сеть DALI включает в себя управляющие устройства (контроллер, датчики, переключатели) и управляемые устройства (драйвера, диммеры и ЭПРА), работающей от источника постоянного тока 16В [5].

Технические параметры шины:

1) сила тока не более 250мА
2) передача данных на скорости 1200 бит/сек
3) передача сигнала на расстоянии до 300 м

Допускается смешанная топология сети – звезда, шина, но не кольцо. Каждому прибору присваивается индивидуальный адрес, создавая возможность управлять отдельным светильником, группой или всей системой сразу. Соответственно, предусмотрено три вида команд: индивидуальные, групповые и широковещательные [5].

Корректировка яркости, включение/выключение происходит за счет сигнальных команд с использованием двунаправленного обмена данными, когда контроллер не только посылает, но и запрашивает информацию о состоянии каждого компонента [5].

Одновременно к одной линии можно подключить до 64 подчиненных устройств, для масштабных проектов возможно увеличение до 12800 с помощью DALI роутеров [5].

Преимущества DALI

1) открытый протокол, доступный всем производителям.
2) не подвержен помехам и не требует экранирования.
3) управлять одним прибором можно с помощью нескольких контроллеров
4) в памяти каждого устройства хранятся все важные настройки (адрес, принадлежность к группам, сценарии работы, скорость диммирования, уровни яркости) [5].
5) двусторонняя связь позволяет отслеживать состояние и статус любого устройства, моментально выявляя любую его неисправность [5].
6) легко устанавливается и настраивается в помещениях любого размера (от маленьких комнат до крупных зданий) [5].
7) можно интегрировать в системы умный дом.

1.6 ZigBee

ZigBee: Ячеистая топология.

После появления в 1999 году концепция «Интернета вещей» (IoT – Internet of things), для передачи данных возможностей Bluetooth и Wi-Fi было недостаточно в силу низкой надежности и дороговизны компонентов. ZigBee позволяет устройствам связываться между собой напрямую и с использованием промежуточных узлов [6]. (Рисунок 1.6)

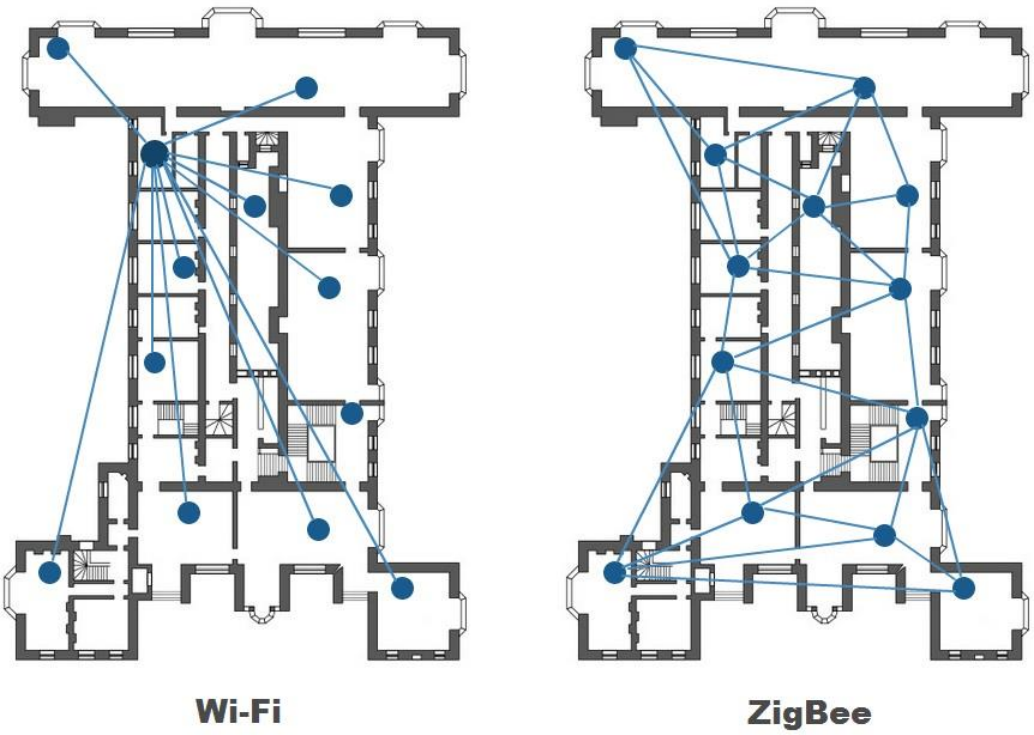


Рисунок 1.6 Структуры Wi-Fi и ZigBee. [6]

При подключении Wi-Fi, если нарушится функционирование центрального узла, то передача данных прекратится. В случае возникновения препятствий на пути сигнала устройства могут остаться без связи. Структура ZigBee предполагает передачу сигнала от одного устройства к другому, пока не достигнет конечного получателя. Если нарушится работа любого компонента или возникнут помехи, сеть предложит сигналу альтернативный маршрут [6].

Структура ZigBee:

1) Координатор - хранит всю информацию о сети. Его задачи управление и маршрутизация данных, выбор политики безопасности, разрешение/запрет на добавление новых элементов, при возникновении помех может перевести устройства на другой частотный канал [6].
2) Маршрутизатор - постоянно распределяет данные по сети, прокладывает оптимальные пути и ищет другие маршруты при перебоях в работе компонентов сети. Один маршрутизатор способен обслуживать до 32 устройств [6].
3) Конечное/выходное устройство - к ним относятся выключатели и датчики, которые обмениваются информацией с координатором или маршрутизатором, передают им сигнал по определенным событиям

(нажатие кнопки включения, открытие/закрытие окон, дверей) и отрабатывают полученные команды. Преимущественно находятся в спящем режиме, поэтому потребляют минимум электроэнергии. Работают от встроенного аккумулятора или батареек [6].

Характеристики и преимущества ZigBee

1) Рабочая частота 2,4 ГГц
2) Радиус действия - внутри помещений 75-100 м, на открытой территории до 300 м и более. Добавление новых устройств позволяют увеличить зону покрытия до нескольких тысяч квадратных метров [6].
3) До 65 000 светильников в сети
4) Передача данных на скорости 250 кбит/с
5) Моментальное включение светильников. Время реагирования 30 миллисекунд.
6) Низкое энергопотребление. Достигается за счет конечных устройств, проводящих большую часть времени в спящем режиме и способных работать от батареек типа AA/AAA в течение трех лет [6].
7) Самонастройка и самовосстановление. Структуру сети определяют характеристики профиля координатора, настройка происходит автоматически в процессе добавления устройств. Благодаря ячеистой топологии в случае неполадок/перегрузок сеть самостоятельно восстановит свое функционирование [6].
8) Широкие возможности в управлении освещением. Диммирование - плавное регулирование яркости от 0 до 100%, подключение необходимых датчиков, возможность программировать режим освещения, объединять светильники в группы, контроль работоспособности отдельных устройств [6].
9) Цена. Например, лампочки zigbee в два раза дешевле умных wifi лампочек.
10) Ключевая особенность. В одну сеть могут быть объединены устройства разных профилей (освещение, отопление, электроэнергия и любые другие компоненты «умного дома»).

1.7 Открытый стандарт автоматизации инженерных систем – KNX

Стандарт KNX разработан для контроля и управления инженерными системами зданий, таких как освещение, вентиляция и кондиционирование, безопасность [7].

Стандарт предусматривает использование следующих вариантов передачи сигнала:

1) Витая пара (трансфер на скорости до 9600 бит/с)
2) Электрическая линия (пропускная способность до 1200 бит/с)
3) Радиочастотные каналы (данные передаются на частотах 433 и 868 МГц)
4) IP-сеть (до 10 Мбит/с)

Основу структуры KNX составляет шина любой топологии за исключением кольцевой схемы, объединяющая все оборудование: (Рисунок 1.7)

1) Датчики или сенсоры (движения, присутствия, влажности, температуры, протечки воды, утечки газа, выключатели, настенные панели), они регистрируют внешние события, после чего передают команду устройству для ее исполнения) [7].
2) Исполнительные компоненты (диммеры, реле, модули), получают команду от датчиков или контроллеров, после чего изменяют свое состояние (включен/выключен, регулирование), тем самым управляя оборудованием [7].
3) Компоненты системы (блоки питания, интерфейсы программирования, мосты, контроллеры) [7].

Топология шины:

1) Линия. Последовательное объединение нескольких устройств [7].
2) Звезда. Максимум 4 сегмента с отдельным источником питания, каждый сигмет может объединить до 64 устройств с усилителем до 256. Максимальная длинна кабеля до 1000м, с усилителем до 4000м [7].
3) Дерево. Объединенные сегменты образуют линии, объединенные линии создают области до 15 штук, что увеличить количество оборудования до 57 375 компонентов [7].

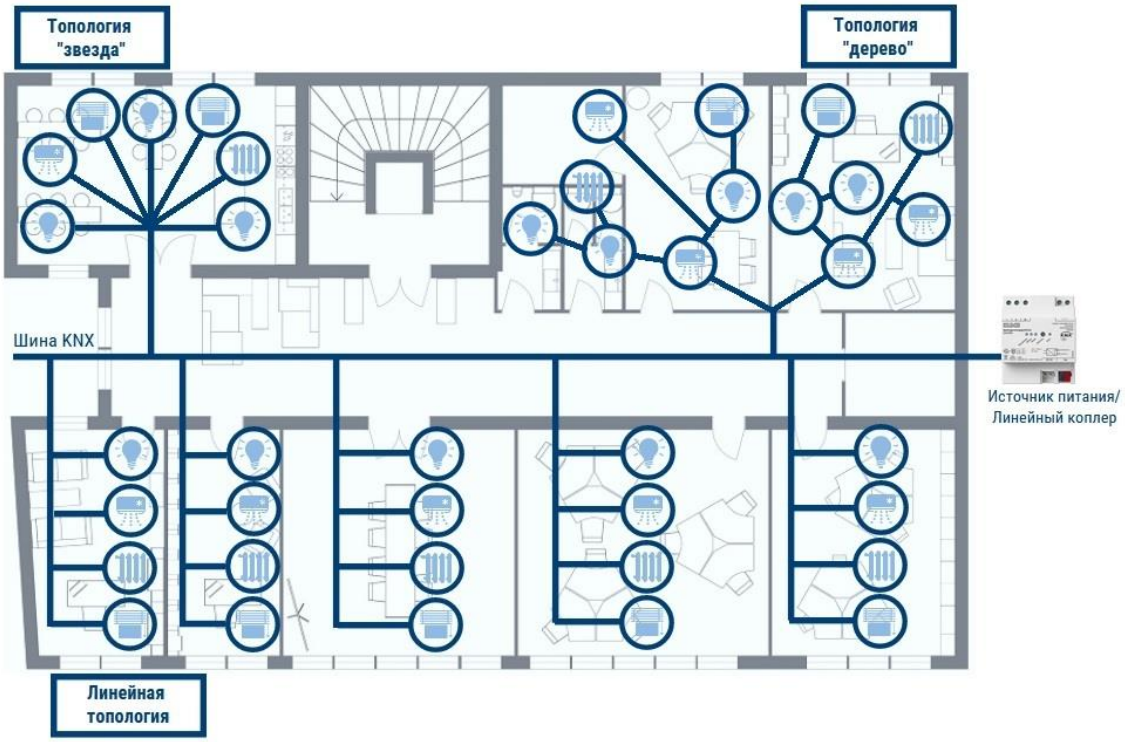


Рисунок 1.7 Топология шин. [7]

Работа всей системы программируется с помощью специального софта - ETS (Engineering Tool Software). Обмен данными между элементами KNX может происходить напрямую (без необходимости использования специального контроллера) с помощью сетевого протокола CSMA/CA, который гарантирует бесконфликтную трансляцию сигнала с сохранением скорости. Информация от датчиков формируется в телеграммы. В случае успешного прохождения исполнительное устройство подтверждает факт ее получения. Если подтверждение отсутствует, дважды срабатывает повторная передача, после этого процесс передачи прекращается [7].

Основные преимущества KNX:

1) Использование продукции разных производителей в одной системе.
2) Замена любого элемента без влияния на работу остальных компонентов.
3) Широкое применение.
4) Два режима настройки: упрощенный (с помощью контроллера, ограниченный набор функций) и системный (программирование с помощью ПК, доступ к полному функционалу - для квалифицированных проектировщиков) [7].

5) Быстрое восстановление после выключения питания за счет хранения заданных настроек в памяти устройств [7].

6) Интеграция с системами других типов. Некоторые производители выпускают специальные шлюзы для объединения KNX-систем с телефонными сетями и охранной сигнализацией [7].

7) KNX может легко взаимодействовать с компонентами, использующими протокол DALI для управления освещением. Например, выбор выключателей и пультов управления больше с поддержкой KNX, нежели с DALI. Зато светильники, диммируемые по DALI, проще найти, чем управляемые по KNX [7].

1.8 LoRaWAN

LoRa (Long Range) - технология, преобразующая данные в электромагнитные волны, представляет собой физический уровень протокола [8].

LoRaWAN (Long Range Wide Area Networks) представляет собой беспроводной протокол, соединяющий устройства, работающие на батарейках, к сети интернет [8].

Принцип работы и структура LoRaWAN.

Система состоит из центрального сервера, базовых станций (они же шлюзы, гейты или концентраторы) и конечных устройств (они же точки, узлы) [8]. (Рисунок 1.8)

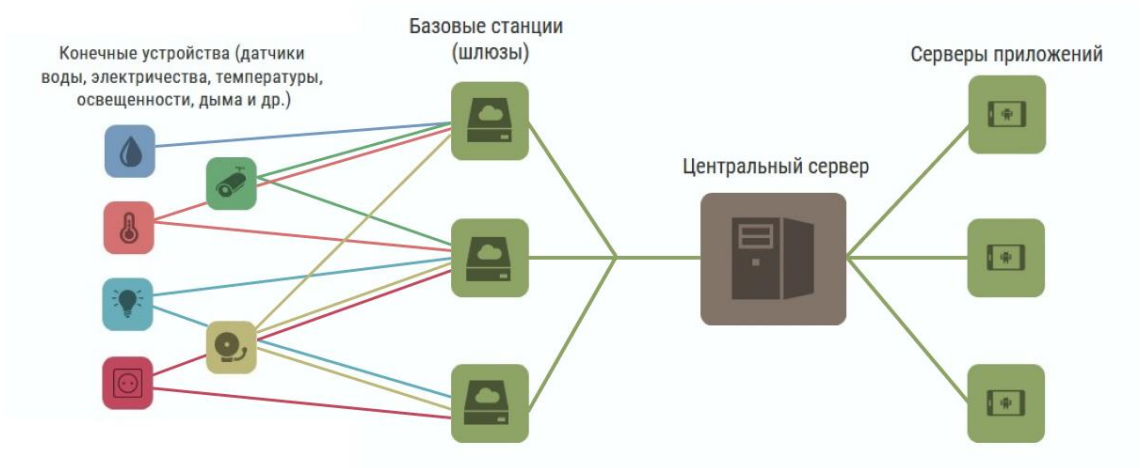


Рисунок 1.8 Структура LoRaWAN. [8]

У LoRaWAN двухзвездная топология. Каждая из конечных точек не привязана к конкретной станции. Датчики расположены удаленно, автономно и работают от батареек. Пакеты данных поступают от конечных точек к шлюзам, затем к следующему звену цепи – центральному серверу. Центральный сервер обрабатывает данные, регулирует скорость их трансляции, выполняет проверку безопасности и планирует подтверждение получения через оптимальный шлюз. После чего они попадают на сервер приложений, и уже затем к пользователю. Сигнал передается по радиоканалу. Дальность действия определяется характеристиками базовых станций. В среднем на открытой местности до 15 км, в плотной городской застройке - до 5 км [8].

Конечные устройства включаются только тогда, когда у них есть что транслировать, в остальное время они пребывают в спящем режиме [8].

Сферы применения:

1) Считывание показаний счетчиков (электричество, вода, газ)
2) Отслеживание местоположения транспорта и груза
3) Мониторинг производственных процессов (состояние оборудования, уменьшение его простоя, безопасность персонала)
4) Автоматизация зданий
5) Уличное освещение

Преимущества управления светом с помощью LoRaWAN

1) Минимальное энергопотребление (на батарейках типа АА устройства могут проработать до 10 лет) [8].
2) Дальность связи (одна БС может покрыть сотни квадратных километров).
3) Открытый протокол без необходимости получения лицензий на использование частот [8].
4) Масштабируемость. Сеть может быть развернута с минимальным количеством оборудования, при этом за счет добавления новых базовых станций можно увеличить скорость передачи и снизить нагрузку на другие шлюзы [8].
5) Устойчивость к радиопомехам
6) Безопасность сетей обеспечивается двухуровневым шифрованием [8].

1.9 Управление освещением по PLC

Существует три варианта организации каналов для систем управления освещением. Беспроводной вариант в силу низкой защищенности от помех и небольшой дальности действия редко используется в промышленных масштабах. Проводные системы освещения требуют дополнительных затрат на прокладку кабелей, что может стать ограничением на их внедрение на готовых объектах. Третий вариант подразумевает использование электросети совместно с технологией PLC (Power Line Communication) [9].

Особенности технологии PLC

PLC классифицируется на широкополосные (до 200 Мбит/с) и узкополосные (до 1 Мбит/с) системы. Первый вариант актуален для доступа в интернет, для трансляции потокового видео, цифровой телефонии и телевидения. Узкополосные системы рассчитаны для объединения электроприборов в единую систему управления и достаточны для организации полноценного управления светом [9].

Для создания системы требуется наличие PowerLine-модема, который будет связываться с аналогичным модемом, установленным в распределительном щите здания, который подключается к высокоскоростному каналу. Добавление в сеть новых устройств осуществляется с помощью специальных адаптеров [9].

С помощью PLC можно развернуть систему управления инфраструктурой предприятий (кондиционирование, водо- и теплоснабжение, электроэнергия, лифты, охранная сигнализация) [9].

Силовые кабели играют роль физической среды для передачи сигнала по тому же протоколу DALI или 0-10В [9].

Преимущества PLC:

1) Легко развернуть, не требуется прокладка кабеля;
2) Разветвленная инфраструктура;
3) Защищенность данных.

Недостатки:

1) Высокая стоимость PLC-модемов.
2) При изношенной проводке исключена качественная передача данных.
3) Единый стандарт так и не принят.

2. ПРОЕКТ

2.1 Объект автоматизации

Объектом автоматизации стало освещение в однокомнатной квартире, в которой предполагалось осуществить замену электропроводки и капитальный ремонт. (Рисунок 2.1)

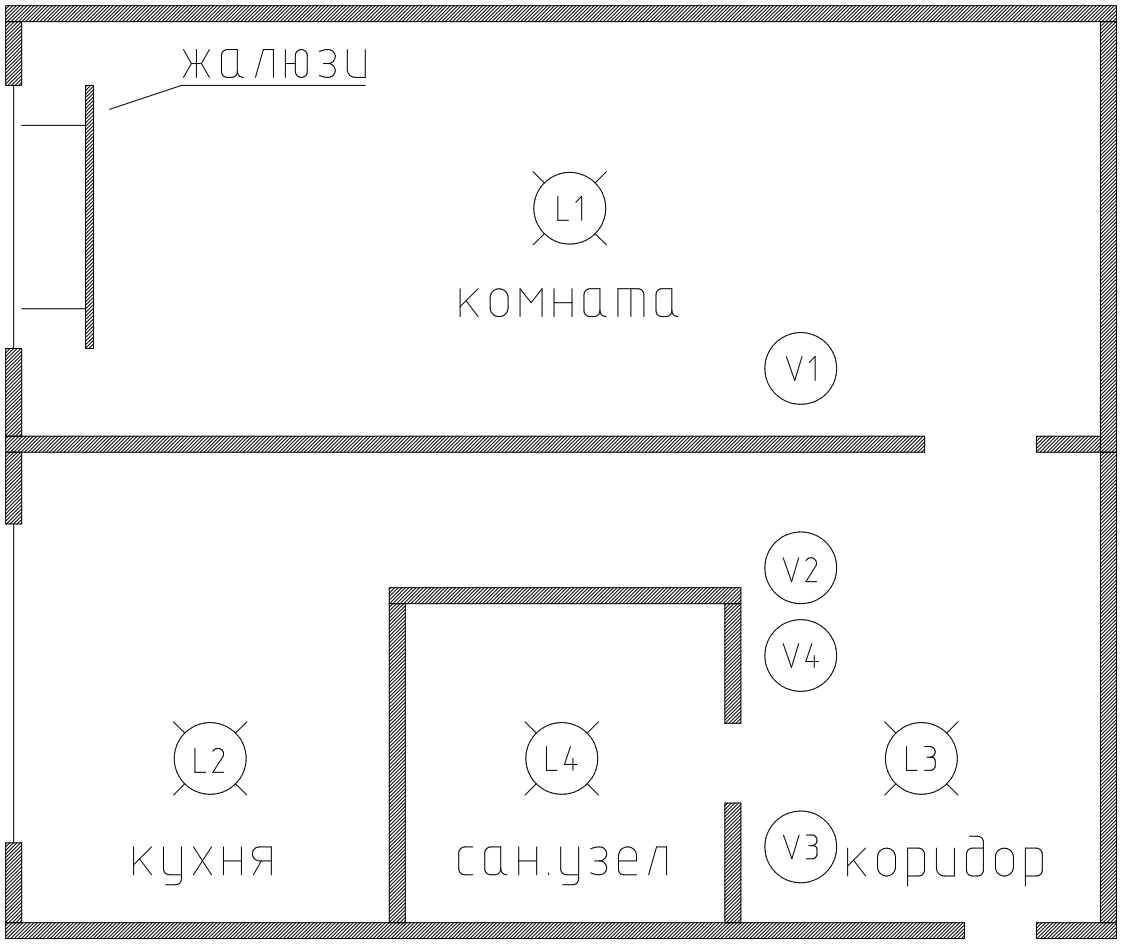


Рисунок 2.1 Объект автоматизации (фото автора)

Таблица 2. Описание элементов схемы

Обозначение	Название	Тип/модель	Место расположения
V1	Вексельный выключатель	NE IP 10-001-1 EPSILON	Комната
V2	Вексельный выключатель	NE IP 10-001-1 EPSILON	Коридор
V3	Вексельный выключатель	NE IP 10-001-1 EPSILON	Коридор
V4	Вексельный выключатель	NE IP 10-001-1 EPSILON	Коридор
L1	Лампочка	OSRAM 15W	Комната
L2	Лампочка	OSRAM 15W	Кухня

L3	Лампочка	OSRAM 15W	Коридор
L4	Лампочка	OSRAM 15W	Сан.узел

2.2 Техническое задание

В квартире необходимо автоматизировать систему освещения. При этом должна оставаться возможность осуществлять ручное управление светом в квартире, в любой момент времени. Должна быть возможность выключения света как непосредственно по месту, так и удаленно (смартфон, планшет, персональный компьютер).

Задачи автоматизации и удаленного управления освещением.

1. Экономия электроэнергии за счет включение освещения только там, где оно необходимо и когда необходимо. Выключение освещения при отсутствие людей в помещение.
2. Удобство и комфорт для пользователя системы.
3. Быть не дорогим, так как электропотребление освещением в однокомнатной квартире относительно не высоко и нет смысла устанавливать дорогостоящее оборудование и автоматизацию.
4. Доступно при приобретении, просто в установке и замене элементов.
5. Система должна по максимуму использовать естественное освещение.
6. Система должна иметь несколько режимов управления освещением.

   6.1 Ручной режим

   Включает и выключает свет не зависимо от показаний пир датчика и фоторезистора, а также не реагирует на изменения датчика тока, если свет был включен или выключен механически, вексельным выключателем.

   6.2 Автоматический режим

   Алгоритм работы авто режима: при достаточном естественном освещение на улице открывает жалюзи и закрывает их при недостаточном. Включает свет в помещение при обнаружение движения (пир датчик) и низком естественном освещение (фоторезистор). Всегда выключает свет при отсутствие движения. Горит свет или нет, считывает с (датчика тока).

2.3 Система

В проекте понадобится контролер с функциями беспроводной связи, датчики фиксирующие движение и измеряющие уровень освещенности, исполнительные

механизмы, включающие и выключающие свет, редуктор открывающий и закрывающий жалюзи, концевые выключатель фиксирующие положение жалюзей.

3. ОБОРУДОВАНИЕ

3.1 Датчики

Для выполнения поставленных задач автоматизированная система должна понимать есть, кто в помещении или нет, знать уровень освещенности в помещении и на улице, знать в любой момент времени включен свет или нет в каждом из помещений.

Для этого понадобится 3 разных типа датчиков

1) Датчик присутствия. Необходим для определения присутствия человека в помещение.
2) Датчик освещенности. Необходим для измерения освещенности помещений.
3) Датчик тока. Необходим для распознавания системой включен свет или нет.

3.1.1 Фоторезистор

Фоторезисторы используются для измерения освещенности. (Рисунок 31.1)

Основное его назначение — переводить количество света, попадающего на чувствительную площадь, в полезный электрический сигнал. Сигнал в последствии может обрабатываться аналоговой, цифровой логической схемой или схемой на базе микроконтроллера [10].

Рисунок 3.1.1 фоторезистор (модель 5516) [10]

В проекте их понадобиться 3 штуки.

Один фоторезистор для измерения естественного освещения и по одному в комнате и на кухне.

Фоторезистор естественного освещения нужен для управления штор на окнах. Зная его система сможет открывать шторы днем и обеспечивать естественное освещение помещения и закрывать их в темное время суток, когда от открытых штор нет ни какой пользы.

В сан. узле и коридоре устанавливать их не вижу смысла так как там нет окон и соответственно естественного освещения. Там всегда темно.

Таблица 3.1.1 Тех. Характеристики фоторезистора (5516)

модель	5516
Максимально потребляемая мощность	90 мВт
Максимальное напряжение	150 В
Рабочая температура	-30 – 70 С
Темновое сопротивление	1 МОм
Время оклика	30 мс
Световое сопротивление	10-20 КОм

3.1.2 Пир датчик

Для того что бы включалось освещения только там, где оно необходимо и когда необходимо, а так же что бы не горело в пустую, при отсутствие кого либо в помещение, система должна быть оснащена датчиком присутствия.

Я выбрал пир датчик HC-SR501. (Рисунок 3.1.2) На мой взгляд, он полностью соответствует всем поставленным задачам и условиям. Имеет два переменных резистора для регулировки чувствительности и времени включенного состояния после обнаружения движения. Может работать в двух режимах:

1) После обнаружения движения включается на заданный период времени и выключается по его прошествую, до следующего обнаружения движения.

2) После обнаружения движения, включается на заданный период времени и начинает его отсчет по новой, после каждого обнаружения движения.

Переключение режимов осуществляется с помощью джампера.

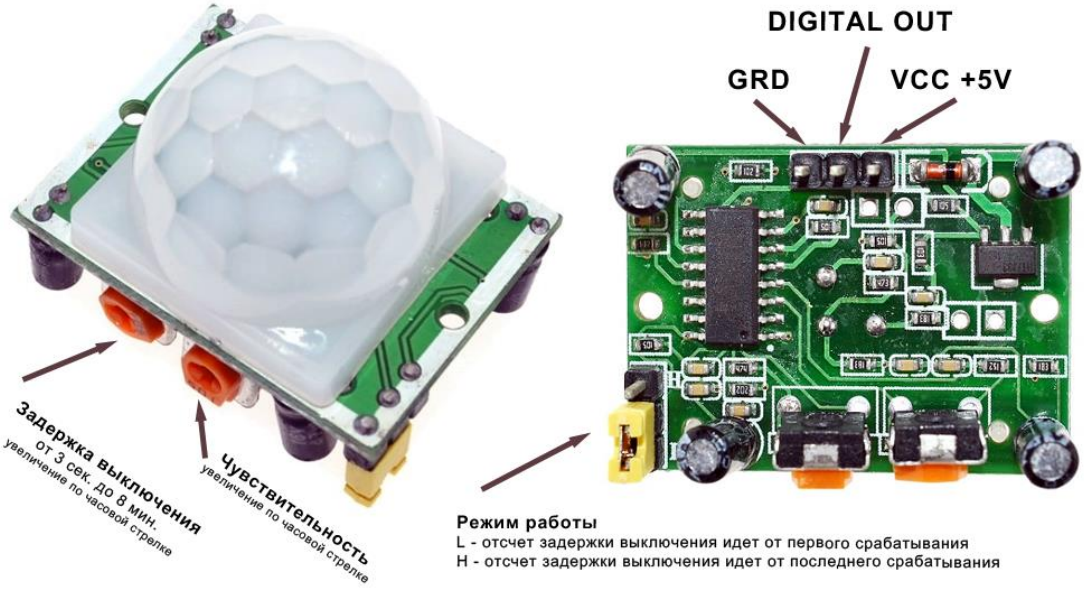


Рисунок 3.1.2 Пир датчик (HC-SR501) [11]

Таблица. 3.2 Технические характеристики (HC-SR501)

Характеристика	Значение характеристики
Расстояние обнаружения: 3-7 м	3-7 м (можно регулировать)
Диапазон обнаружения	<140 °
Время задержки	5-200 с (можно регулировать, 5s +-3%)
Время блокировки	2,5 с
Рабочая температура	-20 - + 80 °C
Напряжение питания	3–5 В
Потребляемый ток	50 мА
Габариты	28×36 мм
Вес	25 г

3.1.3 Датчик тока

Для того чтобы система точно знала, горит лампочка или нет я в своем проекте буду использовать датчик тока ACS712. (Рисунок 3.1.3)

Датчик тока ACS712 основан на эффекте Холла, суть которого в следующем: если проводник с током помещён в магнитное поле, на его краях возникает ЭДС, направленная перпендикулярно к направлению тока и направлению магнитного поля [12].

ACS712 датчик построен на эффекте Холла и имеет линейную зависимость измеряемого тока и выходного сигнального напряжения и может работает с постоянным и переменным током [12].

Рисунок 3.1.3 Датчик тока (ACS712) [12].

Таблица 3.2 Технические характеристики ACS712

Модель	ACS712 5A
Напряжение питания	+5,0 В;
Ток потребления не превышает	110мА
Сопротивление токовой шины	1,2 мОм
Температура эксплуатации	-40°C…+85°C
Размер	31мм х 13мм
Чувствительность	185 мВ/А

Я буду использовать датчик с номинальным током 5А. у него самая высокая чувствительность.

3.2 Исполнительные механизмы

Исполнительные механизмы — это устройства, механически воздействующие на физические процессы путем преобразования электрических сигналов в требуемое управляющее воздействие [13].

В проекте потребуются реле для включения и выключения освещения в автоматическом режиме. Редуктор, для открытия и закрытия жалюзей. Для ручного режима потребуются вексельные выключатели.

3.2.1 Реле

Исполняющий элемент, включающий и выключающий свет это, конечно же, электромагнитное реле, а так как по расчету их понадобится 6 штук, то для проекта был заказан релейный модуль. (Рисунок 3.2.1)

Рисунок 3.2.1 Релейный модуль [13]

Таблица 3.2.1 Технические характеристики релейного модуля

Питание	5В постоянного тока
Ток потребления	до 50 мА на катушку
Сопротивление обмотки реле	70 Ω ±10%
Сопротивление изоляции реле	100 МОм
Время срабатывания реле при выключении	до 5 мс
Время срабатывания реле при включении	до 10 мс
Скорость механических переключений	до 300 операций/мин

Рабочая температура	-25 ... +70 °C
Рабочая влажность	45 ... 85%
выходные цепи	до 30В постоянного тока 10А
выходные цепи	До 250В переменного тока 10А

Электромагнитное реле является, по сути, управляемым механическим выключателем: подали на него ток – оно замкнуло контакты, сняли ток – разомкнуло. Контакты являются именно контактами: металлическими, которые прижимаются друг к другу. Именно поэтому такое реле может управлять как нагрузкой постоянного, так и переменного тока [13].

Электромагнитное реле имеет ряд недостатков

1) Ограниченное количество переключений: механический контакт изнашивается, особенно при большой и/или индуктивной нагрузке [13].
2) Громко переключается.
3) При большой нагрузке реле может залипнуть, поэтому для больших токов нужно использовать более мощные реле, которые придётся включать при помощи маленьких реле. Или транзисторов [13].
4) Необходимы дополнительные цепи для управления реле, так как катушка является индуктивной нагрузкой, и нагрузкой самой по себе слишком большой для МК [13].
5) Очень большие наводки на всю линию питания при коммутации индуктивной нагрузки [13].
6) Относительно долгое. Время переключения [13].

И все токи в проекте буду использовать именно их, а не твердотельные реле.
(Рисунок 3.2.2)

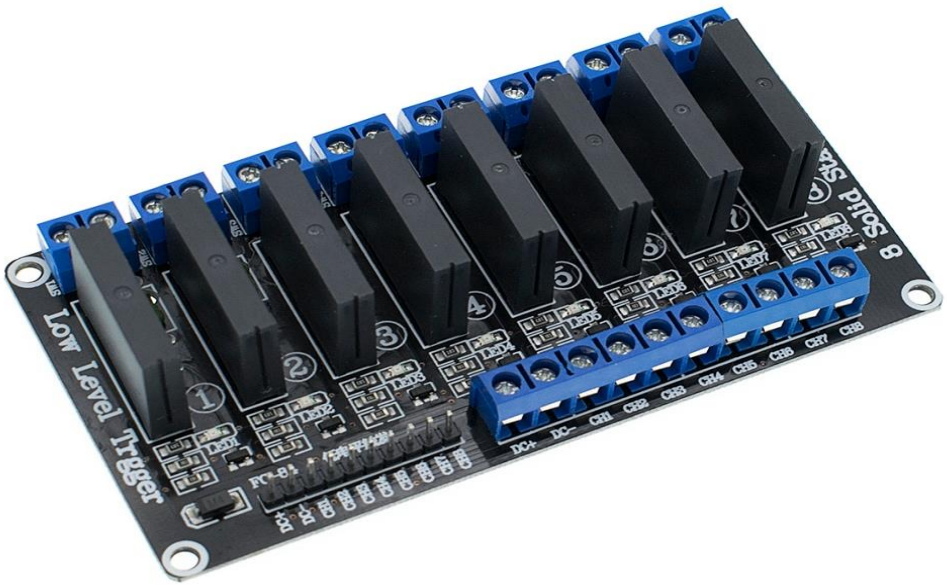


Рисунок 3.2.2 Твердотельное реле [13]

Так как в первом варианте мы имеем два выхода NO и NC, что позволит нам использовать его как проходной выключатель в паре с реальным проходным выключателем.

3.2.2 Редуктор GA12-N20

Для удаленного открытия и закрытия жалюзей понадобиться редуктор. Я выбрал редуктор GA12-N20. (Рисунок 3.2.2) На выбор повлияли его небольшие размеры, возможность выбора значения напряжения и частоты вращения.

Рисунок 3.2.2 Редуктор (GA12-N20) [14]

Талица 3.2.2 Мини редуктор

Напряжение	5В
Частота вращения на холостом ходу	100 об/мин
Частота вращения входного вала	80 об/мин
Номинальный ток	160мА
Пусковой ток	200мА
Передаточное число	150
Размер двигателя	12 мм (диаметр) * 26 мм (высота)
Размер вала	3 мм (диаметр) * 8 ~ 10 мм (длина)

3.2.3 Концевой выключатель

Концевые выключатели (см. рисунок 3.2.) в проекте не обходимы для определения положения жалюзи. Один концевой выключатель крепиться сверху, второй снизу. При достижении верхнего или нижнего положения, жалюзи, направляющей рейкой замыкают концевые выключатели и тем самым останавливают вращение редуктора.

Рисунок 3.2.3 Концевой выключатель (LXW5-1124) [15]

3.2.4 Микроконтроллер

При выборе микроконтроллера учитывалось, прежде всего, наличие необходимых выводов и возможности беспроводной связи с ним. В проекте будет регулироваться свет в четырех разделенных помещениях, для этого понадобится 4 датчика присутствия (4 дигитальных входа), 4 датчика тока для индикации включен свет или нет (4 аналоговых входа), два помещения имеют окна и там необходимо будет знать уровень освещенность, что бы не включать свет днем 2 фоторезистора, плюс один фоторезистор мониторящий уровень уличного освещения, для регулирования жалюзей (3 аналоговых входа), для переключения состояния света необходимы 4 реле и 2 реле, для управления жалюзи (6

дигитальных выходов), 2 концевых выключателя ограничивающие жалюзи (2 дигитальных входа).

И того получается, что контроллер должен иметь минимум 7 аналоговых входов, 6 дигитальных выходов и 6 дигитальных входов.

Для управления светом со смартфона или планшета понадобится контроллер с функциями WiFi или Bluetooth.

Выбор остановился на микроконтроллере ESP32, он полностью соответствует всем необходимым параметрам. (Рисунок 3.2.4)

Фирма Espressif выпустила мощный недорогой микроконтроллер ESP32 летом 2016 года. Устройство представляет собой систему на кристалле, построенную по технологии TSMC 40 нм, с Wi-Fi и Bluetooth контроллерами. Оно оснащено двухъядерным 32-битным процессором, который работает на частотах 80, 160 или 240 МГц. Также в систему интегрированы антенные коммутаторы, радиочастотные компоненты, фильтры, усилители, модули управления питанием. Подключается ESP32 к компьютеру через обычный USB провод [16].

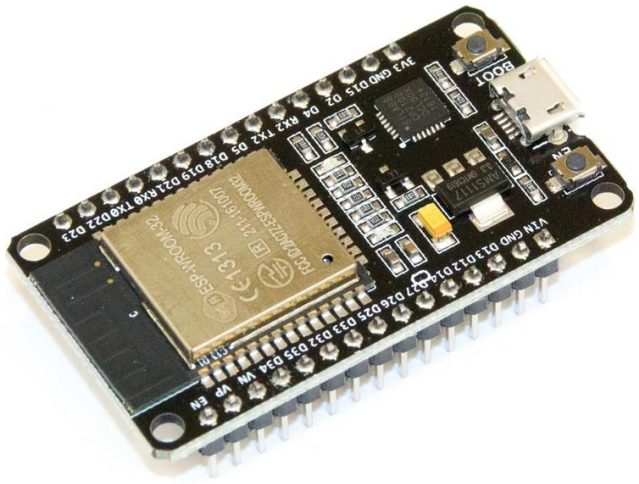


Рисунок 3.2.4 ESP32 (DEVKIT V1) [16]

Плата имеет 21 универсальный вывод, из которых 15 можно настроить как АЦП (аналого-цифровой преобразователь) с разрядностью 12 бит. 2 вывода могут быть ЦАП (цифро-аналоговый преобразователь), разрядность 8 бит. Все выводы могут

быть настроены как ШИМ выходы, а также все выводы можно использовать как прерывания [16].

На плате присутствуют 2 кнопки, EN для перезагрузки платы и BOOT для установки платы в режим прошивки.

4. РАСПОЛОЖЕНИЕ ДАТЧИКОВ И ИСПОЛНИТЕЛЬНЫХ МЕХАНИЗМОВ

Ниже для наглядности представлена схема расположения автоматики. (рисунок 4)

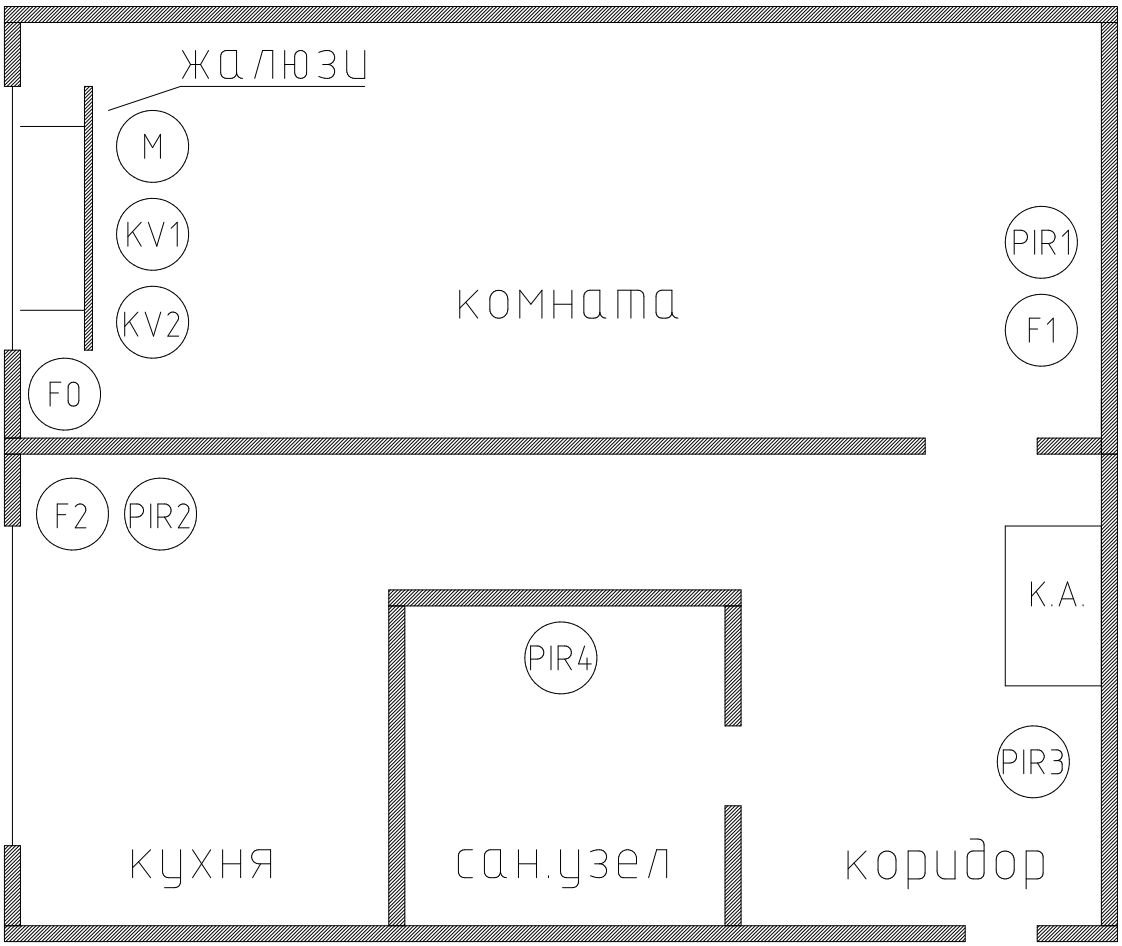


Рисунок 4. Расположение датчиков и исполнительных механизмов (фото автора)

Таблица 4. Описание элементов схемы

Обозначение	Название	Тип/модель	Место расположения
F0	Фоторезистор	5516	кухня
F1	Фоторезистор	5516	комната
F2	Фоторезистор	5516	кухня
PIR1	Пир датчик	HC-SR501	комната
PIR2	Пир датчик	HC-SR501	Кухня
PIR3	Пир датчик	HC-SR501	Коридор
PIR4	Пир датчик	HC-SR501	Сан. узел
KV1	Концевой выключатель	LXW5-1124	Комната
KV2	Концевой выключатель	LXW5-1124	Комната
M	Редуктор	GA12-N20	Комната
К. А	Шкаф автоматики	-	Коридор

4.1 Принципиальная электросхема освещения

Как видно из схемы (рисунок 4.1) управлять включением и выключением лампочек можно как автоматически с помощью реле (К1 – К4) просто переключая контакт с NO на NC, так и с помощью вексельных выключателей (V1 – V4)

При поломке какого ни будь реле или всей автоматики, останется возможность включать и выключать свет с помощью вексельных выключателей (V1 – V4).

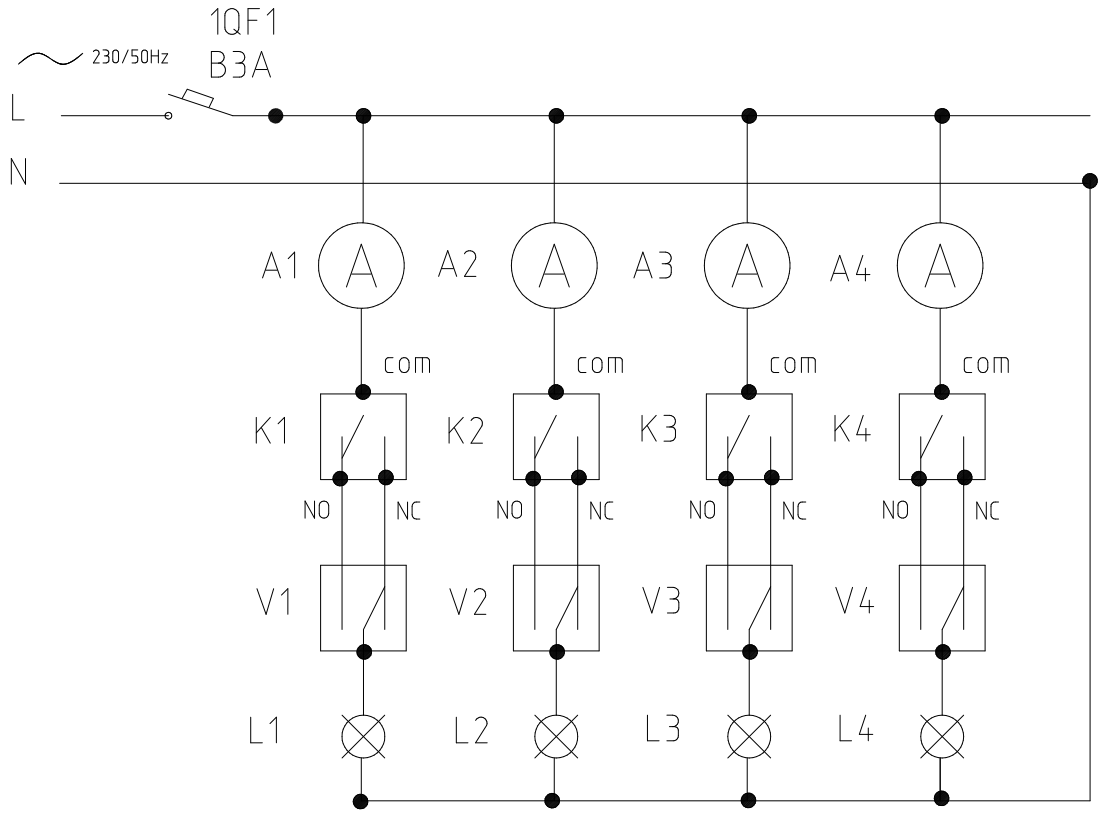


Рисунок 4.1 Принципиальная схема освещения (фото автора)

1) 1QF1B3A-автоматический выключатель (3А), 2) А1-А4 – датчики тока, 3) К1-К4 – реле, 4) V1-V4 – вексельные выключатели, 5) L1-L4 – лампочки, 6) com – общий контакт реле, 7) NO – нормально открытый контакт реле, 8) NC – нормально закрытый контакт реле.

4.2 Принципиальная электросхема жалюзей

Мотор постоянного тока меняет свое направление вращения в зависимости от полярности питания. Значит, когда мы переключим реле К5 из положения NC на положение NO мотор начнет вращаться в одном направление, допустим по часовой стрелке. Если мы вернем К5 в исходное положение и теперь уже переключим контакты реле К6 с NC на NO то мотор начнет вращение против часовой стрелки. (см. рисунок 4.2)

Переключая контакты реле К5 и К6 включаю, выключаю редуктор, а также меняю направление движения редуктора.

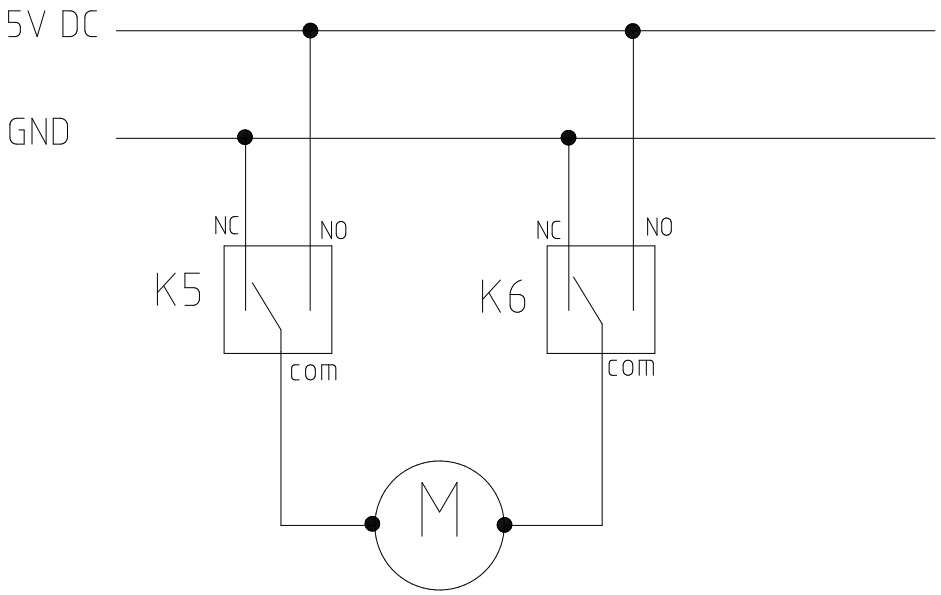


Рисунок 4.2 Принципиальная электоросхема жалюзей (фото автора)

4.3 Монтаж силовой части системы освещения

Потолки предполагается делать натяжными, поэтому от щита автоматики и вексельных выключателей до потолка для прокладки кабелей сделаны штробы. На расстоянии 200 мм от потолка монтированы распределительные коробки. От распределительных коробок у вексельных выключателей по потолку ведутся кабели до лампочек и соединительных коробок у щита автоматики. Кабели выведенный из Щ.А. заводятся в распределительные коробки и соединяются там клеммами wago с кабелями с потолка. Тоже самое делается с кабелем подключенным выключателям. (см. рисунок 4.3)

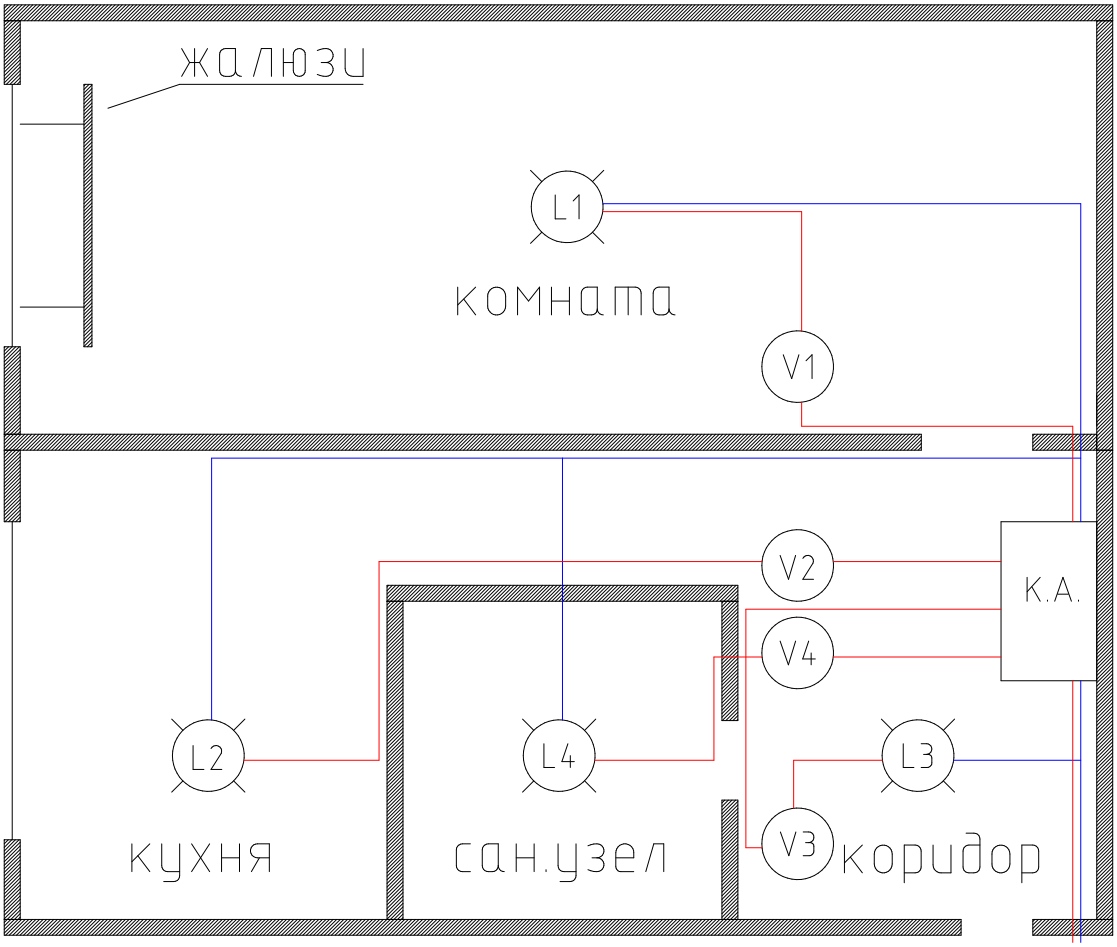


Рисунок 4.3 Монтажная схема силовой части системы освещения (фото автора)

1) красным цветом обозначена фаза, 2) синим цветом обозначена нейтраль.

4.4 Монтаж датчиков и привода жалюзей

Монтаж проводки датчиков будет осуществляться в наружном исполнении. Со щита автоматики в кабельном коробе вниз до плинтусов выводятся 3 экранированных кабеля 7х0.5 и по плинтусам ведутся до места распределительных коробок 1, 2, 3. Далее также в кабельном коробе, от плинтуса до распределительной коробки. С распределительной коробки через клеммы уже кабелями 3х0.5 доводятся до датчиков. (см. рисунок 4.4)

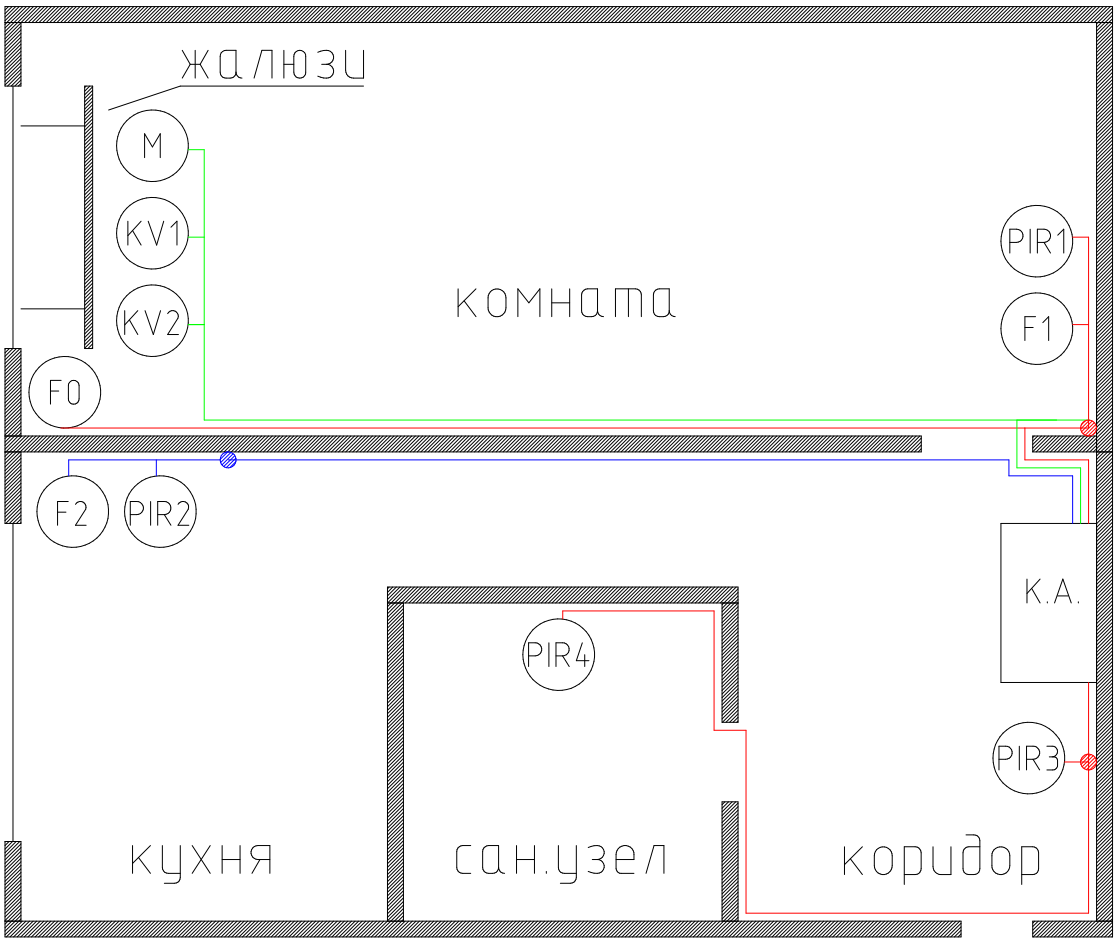


Рисунок 4.4 Монтажная схема датчиков и жалюзей (фото автора)

4.5 Щита автоматики

Для удобства, безопасности и эстетики собран щит автоматики. (Рисунок 4.5)

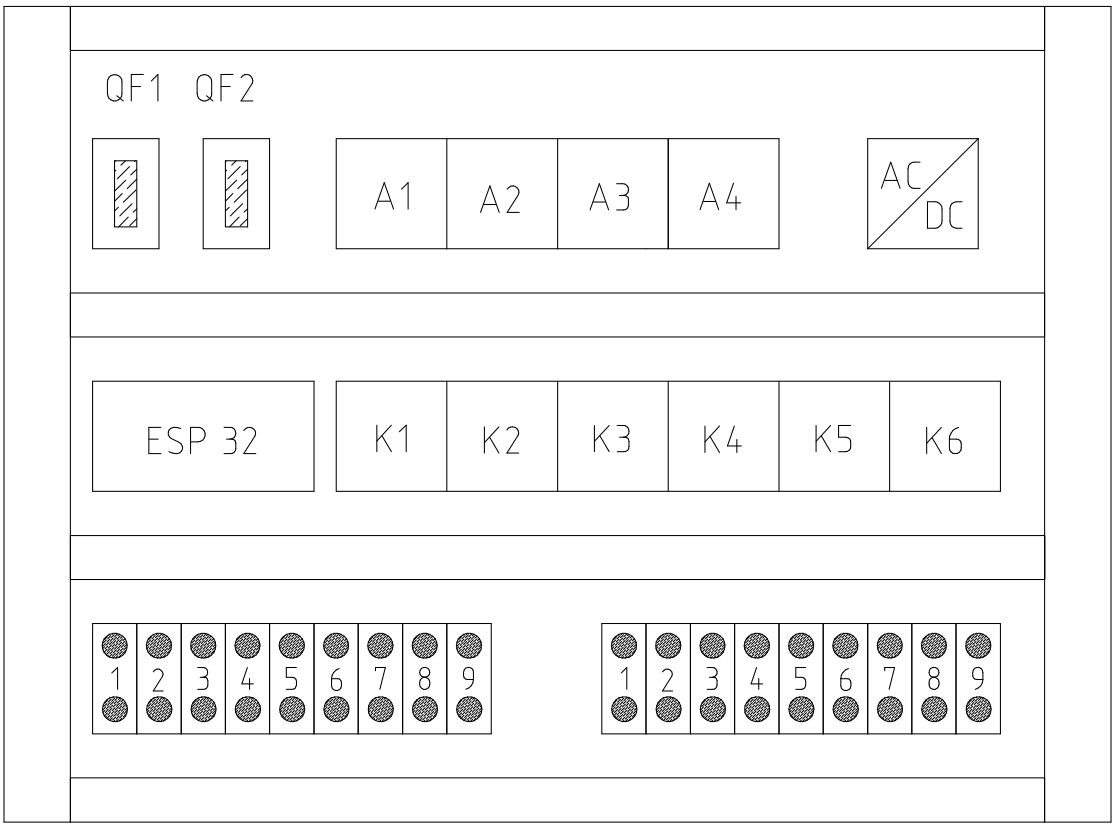


Рисунок 4.5 Щит автоматики (фото автора)

1) QF1-автоматический выключатель силовой части

2) QF2-автоматический выключатель автоматики

3) A1 –A4 датчики тока

4) AC/DC адаптер питания автоматики

5) ESP32 микроконтроллер

6) K1-K6 реле

7) клеммы соединения

4.6 Питание датчиков и ESP 32

Питание автоматики будет осуществляться с помощь адаптера. Адаптером является обычное зарядное устройство смартфона с параметрами выхода 5V DC и током не менее рассчитанного максимального, который будет потреблять автоматика. (рисунок 4.5)

Расчет произведен согласно техническим характеристикам оборудования:

ESP32 – 260мА, реле (6шт.) – 300мА, датчики тока (4шт.) – 440мА, фоторезисторы (3шт.) – 60мА, пир датчики (4шт.) – 200мА, редуктор – 200мА.

Итого получилось 1460мА максимальное потребление тока оборудованием. Для проекта достаточно было взять адаптер 2А (учтен минимальный запас по току), в проекте используется адаптер 4А для возможности расширения проекта в будущем.

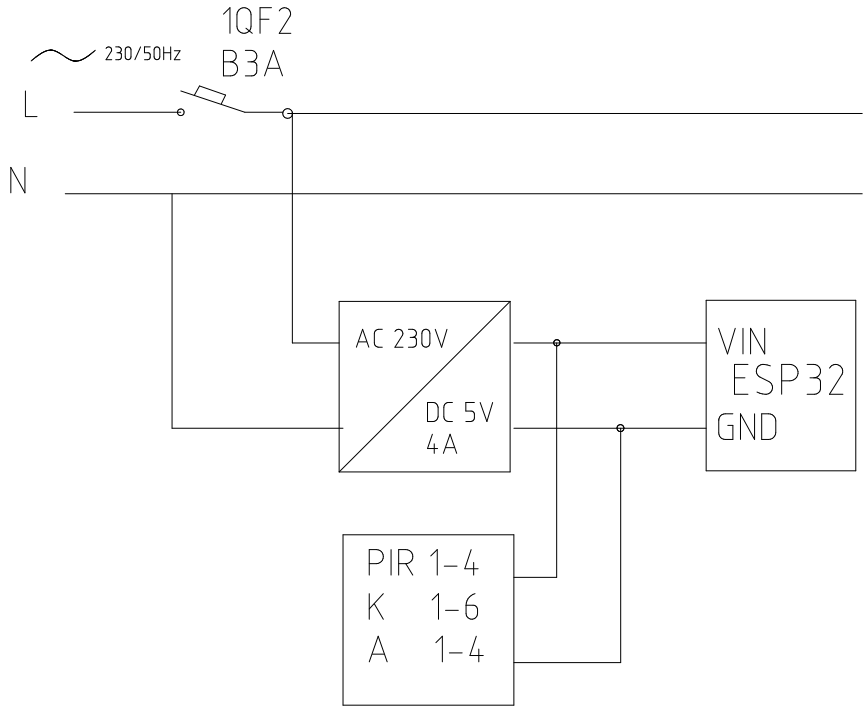


Рисунок 4.5 Питание датчиков и ESP 32 (фото автора)

4.7 Принципиальная схема автоматики

Принципиальная схема автоматики необходима при подключении датчиков и исполнительных механизмов, а также удобна при написании программы (позволяет не запутаться). (рисунок 4.6)

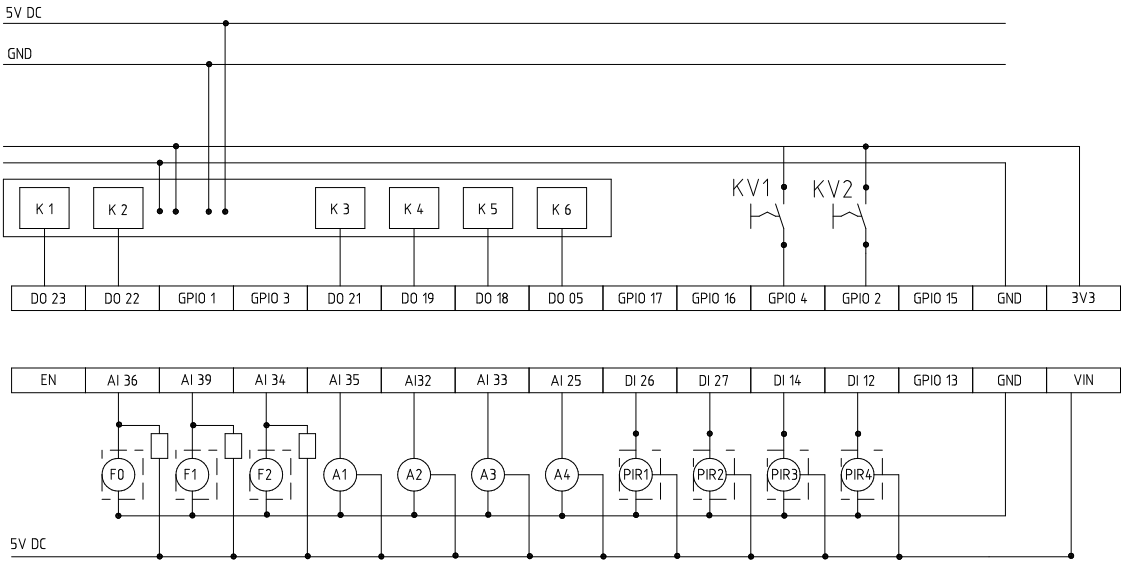


Рисунок 4.6 Принципиальная схема автоматики (фото автора)

Аналоговые входы (сверху вниз мне так удобно) 36, 39, 34, 35, 32, 33, 25

Дискретные входы (продолжаем) 26, 27, 14, 12, 4, 2

Дискретные выход (продолжаем, сверху вниз) 23, 22 21,19,18,17 (выходы 1 и 3 брать не стал)

3 пин, не может быть выходом, а первый может работать не корректно. Взяты следующие по очереди.

5. АЛГОРИТМ ПРОГРАММЫ

Алгоритмы — это основа любой программы, ведь нам нужно «объяснить» контроллеру, чего мы от него ожидаем, именно для этого и пишется программа. Программа, написанная посредством определенного языка программирования - это перевод в письменный код ожиданий оператора. Т.к. любая программа выполняется последовательно строка за строкой, за исключением прерываний, то мы должны понимать, что происходит на каждом шаге выполнения программы, чтобы получить желаемый результат [17].

Для гибкости системы и удобства пользователя, автоматика будет, состоят из двух не зависимых алгоритмов. Первый алгоритм будет отвечать за работу жалюзей (рисунок 5.1), а второй за включение и выключение света (рисунок 5.2). Это позволит, например, днем закрыть жалюзи при этом оставить включение/выключение света в автоматическом режиме и наоборот.

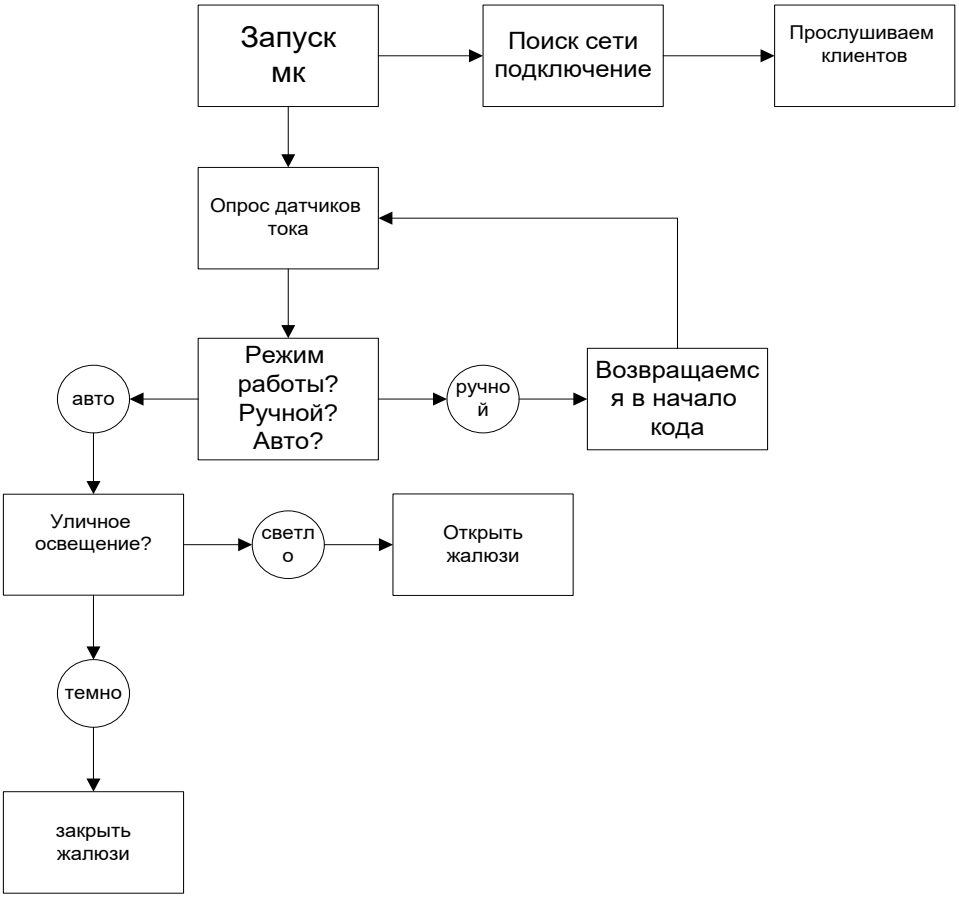


Рисунок 5.1 Алгоритм работы жалюзей (рисунок автора)

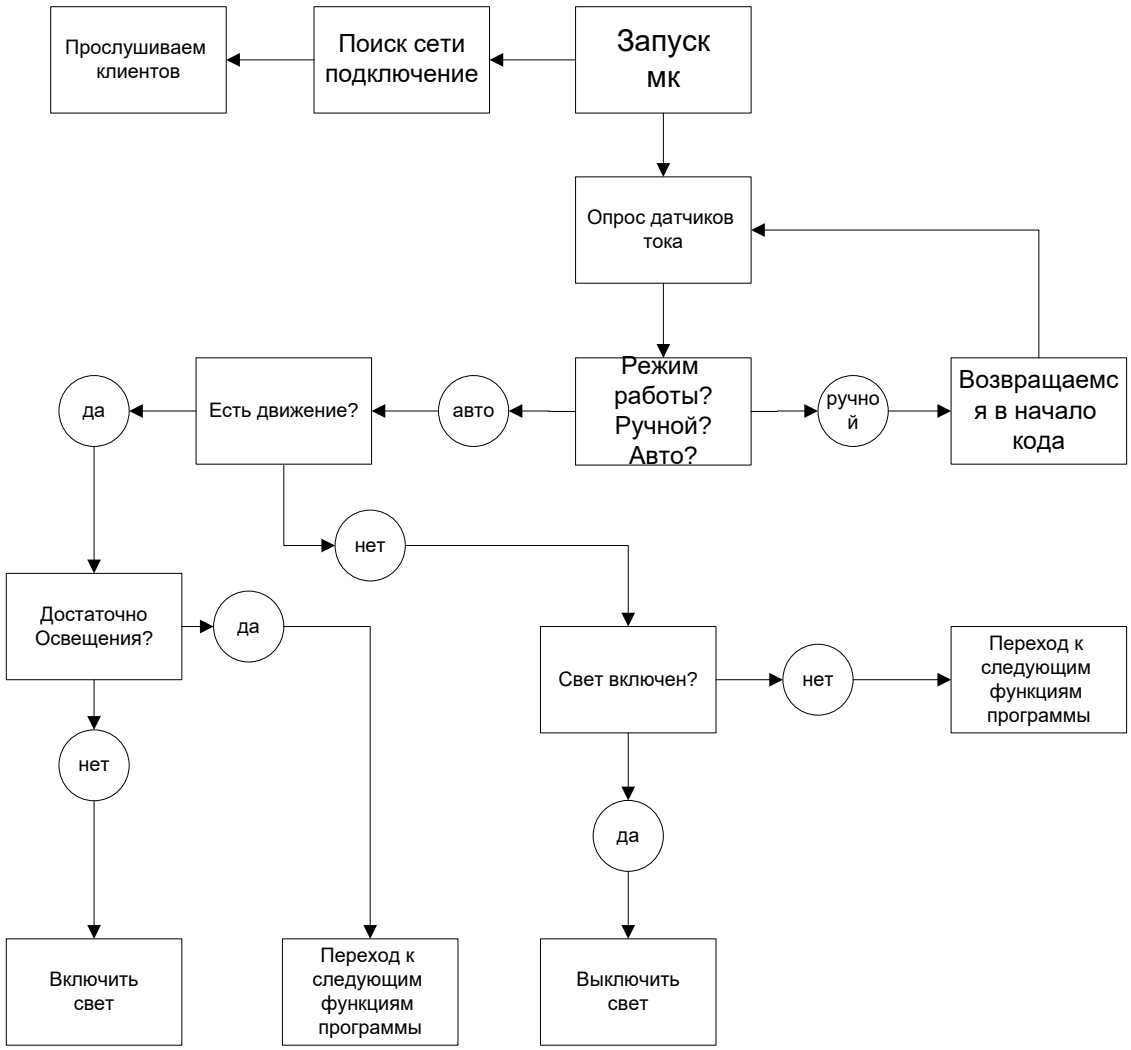


Рисунок 5.2 Алгоритм работы искусственного освещения (рисунок автора)

6. СРЕДА ПРОГРАММИРОВАНИЯ

Первое, что необходимо сделать для того, чтобы начать программировать микроконтроллер это, конечно же, скачать и установить программное обеспечение которое можно использовать с микроконтроллером который собираешься программировать.

6.1 Среда программирования Arduino IDE

Как уже приводилось выше в описании микроконтроллера, ESP32 способен взаимодействовать с большим множеством таких платформ, в том числе и с программным обеспечением Arduino с открытым исходным кодом (IDE).

Именно эта среда и была выбрана для осуществления проекта.

1) Первая и основная причина выбора данной среды стало то, что с ней уже приходилось сталкиваться раньше.
2) Огромным плюсом Arduino IDE является также ее открытость, наличие огромного количества информации, примеров, форумов, сообществ по интересу и т.д.
3) Поддержка облачных технологий: Arduino Cloud
4) Взаимодействие с популярными во всем мире платформами: LoRaWAN, The Things Stack.
5) Может быть установлена на такие операционные системы как Windows, Linux и Mac OS. При помощи соответствующих приложений, есть возможность программировать, через смартфон или планшет.
6) Среда программирования Arduino IDE является на сегодняшний день бесплатным продуктом. И скачать ее можно с официального сайта разработчиков https://www.arduino.cc. В добровольном порядке есть возможность пожертвовать на развитие проекта.

В проекте будет задействована последняя версия, программного обеспечения Arduino IDE 1.8.16

Минус платформы Arduino IDE

Программное обеспечение Arduino предоставляется вам «как есть», и мы не даем никаких явных или подразумеваемых гарантий в отношении его функциональности, работоспособности или использования, включая, помимо прочего, любые подразумеваемые гарантии товарной пригодности, пригодности

для определенной цели или нарушения прав. . Мы прямо отказываемся от какой-либо ответственности за любые прямые, косвенные, косвенные, случайные или особые убытки, включая, помимо прочего, упущенную выгоду, упущенную выгоду, убытки, возникшие в результате прерывания бизнеса или потери данных, независимо от формы иска или правовой теории в соответствии с ответственность за которые может быть возложена, даже если было сообщено о возможности или вероятности такого ущерба[18].

6.2 Установка платы ESP32 в IDE Arduino

После установки программного обеспечения, настроек для плат семейства ESP и самого ESP32 мы там не найдем. Их необходимо установить. Это можно сделать с помощью дополнения для Arduino IDE, оно позволяет программировать ESP 32 при помощи Arduino IDE на его языке программирования.

Дополнение можно найти и скачать по средством самого программного обеспечения Arduino IDE, в разделе (Менеджер плат). (рисунок 6.2)

Рисунок 6.2 Менеджер плат (Arduino IDE) (фото автора)

После установки в Arduino IDE появится все семейство плат ESP, остается тока выбрать модель на которой будет осуществлятся проект и сконфигурировать ее. (рисунок 6.2.1)

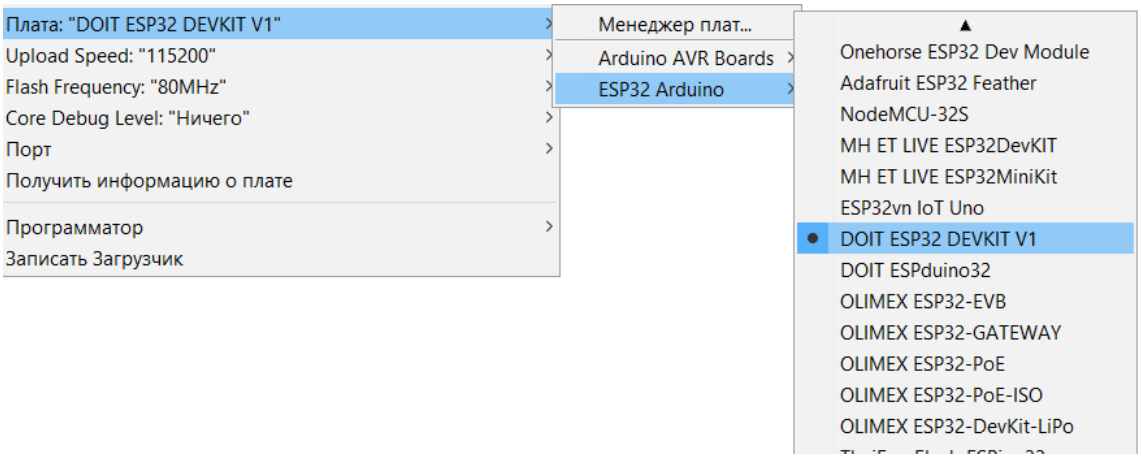


Рисунок 6.2.1 Конфигурация платы ESP 32 (фото автора)

В данном проекте будет использоваться плата ESP32 DEVKIT V1.

Конфигурация платы будет заключаться в выборе скорости соединения с последовательным портом и тактовой частоты. Выставляем их соответственно техническим характеристикам данной платы.

1) Последовательный порт 115200
2) Частота 80МГц
3) Коммуникация микроконтроллера ESP32 с компьютером осуществляется по USB кабелю.

7. НАПИСАНИЕ ПРОГРАММЫ

Теперь, когда придуман сценарий работы системы, сделан монтаж, определились со средой программирования и установили ее, можно переходить к самой интересной части проекта, к написанию программы.

7.1 Язык программирования

Язык программирования устройств Ардуино основан на C/C++ и скомпонован с библиотекой AVR Libc и позволяет использовать любые ее функции. Вместе с тем он прост в освоении, и на данный момент Arduino — это, пожалуй, самый удобный способ программирования устройств на микроконтроллерах [18].

Сами разработчики называют язык Arduino Wiring, так как в стандартной библиотеке Arduino.h используются функции и инструменты из фреймворка Wiring. Но языком, из которого берётся синтаксис, является C++ [18].

Wiring - это среда программирования с открытым исходным кодом для микроконтроллеров [18].

Саму программу можно писать как в самой среде Arduibno так и в любом удобном компиляторе, например Notepad++.

7.2.1 Присвоение имен, выводам микроконтроллера

Для упрощения читаемости и удобства универсальным входам/выходам (согласно схеме подключения датчиков и исполнительных механизмов) используемых в проекте даем название (символическую константу) с помощь директивы #define. (рисунок 7.2.1)

Директива #define это удобная директива, которая позволяет дать имя константе, перед тем как программа будет скомпилирована[18]. Определенные этой директивой константы не занимают программной памяти, прописывается исключительно для удобства.

```
1
2
3    #define f0      36  //фоторезистор отслеживающий уличное освещение
4    #define f1      39  //фоторезистор отслеживающий освещение в комнате
5    #define f2      34  //фоторезистор отслеживающий освещение на кухне
6    #define a1      35  //датчик тока комната
7    #define a2      32  //датчик тока кухня
8    #define a3      33  //датчик тока коридор
9    #define a4      25  //датчик тока сан.узел
10   #define pir1    26  //датчик тока комната
11   #define pir1    27  //пир датчик кухня
12   #define pir1    14  //пир датчик коридор
13   #define pir1    12  //пир датчик сан.узел
14   #define k1      23  //реле выключатель комната
15   #define k2      22  //реле выключатель кухня
16   #define k3      21  //реле выключатель коридор
17   #define k4      19  //реле выключатель сан.узел
18   #define k5      18  //реле жадюзи вверх
19   #define k6      5   //реле жадюзи вниз
20
```

Рисунок 7.2.1 Имена контактов (фото автора)

7.2.2 Объявление переменных

Переменная – это ячейка SRAM памяти, которая имеет своё уникальное название и хранит числа соответственно своему размеру [18]. К переменной мы можем обратиться по её имени и получить значение, либо изменить его [18].

Объявляем переменные, которые будут хранить данные с датчиков, состояние режима работы, а также таймеры, которые будут отсчитывать время и вспомогательные переменные (флаги). (рисунок 7.2.2)

```
int per_f0;          // переменная  фоторезистор отслеживающий уличное освещение
int per_f1;          // переменная  фоторезистор отслеживающий освещение в комнате
int per_f2;          // переменная отслеживающий освещение на кухне
int per_a1;          // переменная  датчик тока комната
int per_a2;          // переменная датчик тока кухня
int per_a3;          // переменная датчик тока сан.узел
int per_a4;          // переменная датчик тока комната
boolean per_pir1;    // переменная реле выключатель комната
boolean per_pir2;    // переменная реле выключатель кухня
boolean per_pir3;    // переменная реле выключатель коридор
boolean per_pir4;    // переменная выключатель сан.узел

boolean rezim_raboty = 0;                //переменная для переключения режимов

boolean startTimer1 = false;             // отсчет времени пир датчика
unsigned long taimer_pir1 = 0;           //таймер пир датчика

boolean startTimer2 = false;             // отсчет времени пир датчика
unsigned long taimer_pir2 = 0;           //таймер пир датчика
```

Рисунок 7.2.2 Переменные (фото автора)

В зависимости от объема данных, которые будут в себе хранить переменные, назначаем им разный тип данных.

Таблица 7.2 Типы данных

boolean	1 байт
int	2 байта
unsigned long	4 байта

7.2.3 Прерывания пир датчиков

Для того что бы программа не опрашивал пир датчики когда свет и так горит, настраиваем их работу по принципу прерывания. Для этого используем функцию attachInterrupt(, ,) и сразу же создадим условие if () при котором вообще возможно выполнение этой части кода. (рисунок 7.2.3)

Каждый вывод микроконтроллера ESP32 имеет возможность обрабатывать прерывания.

```
void IRAM_ATTR prerivanijaPir1() {        //прерывания пир1 датчика
    digitalWrite(k1, HIGH);
    startTimer1 = true;
    taimer_pir1 = millis();

  }
void IRAM_ATTR prerivanijaPir2() {        //прерывания пир2 датчика
    digitalWrite(k2, HIGH);
    startTimer2 = true;
    taimer_pir2 = millis();

  }
```

Рисунок 7.2.3 Прерывания пир датчиков (фото автора)

7.2.4 Инициализация выводов

Функция setup( ) – вызывается во время старта программы и используется для инициализации переменных, определения режимов работы выводов, запуска используемых библиотек и т.д. Функция запускается только один раз после подачи питания или перезарузки платы [18].

В функции setup( ) инициализируем контакты реле, пир датчиков и функцию прерывания. (рисунок 7.2.4)

```
void setup() {

pinMode(k1, OUTPUT);              // инициализируем контакты реле как выход


pinMode(k2, OUTPUT);
pinMode(k3, OUTPUT);
pinMode(k4, OUTPUT);
pinMode(k5, OUTPUT);
pinMode(k6, OUTPUT);

pinMode(pir1, INPUT_PULLUP);      // инициализируем контакты пир датчика как вход
pinMode(pir2, INPUT_PULLUP);
pinMode(pir3, INPUT_PULLUP);
pinMode(pir4, INPUT_PULLUP);
                                  // делаем контакт пир датчика контактом для прерывания
attachInterrupt(digitalPinToInterrupt(pir1), prerivanijaPir1, RISING);
attachInterrupt(digitalPinToInterrupt(pir2), prerivanijaPir2, RISING);
```

Рисунок 7.2.4 Инициализация выводов микроконтроллера (фото автора)

7.2.5 Программа датчиков тока

Функция loop( ) - основная структурная функция любой программы Arduino. Следует после функции **setup().** Функция в бесконечном цикле выполняет последовательно команды, функции, условия описанные в теле функции. По

достижении конца функция возвращается к началу и так до выключения питания микроконтроллера [18].

В этой функции будет основная программа. Здесь будут считываться показания с аналоговых входов и выполнятся основной алгоритм. Первым делом необходимо считывать показания с датчиков тока, так как не зависимо в каком режиме работает микроконтроллер, пользователь должен знать, где у него включен свет. И если это необходимо выключить его удаленно или вручную. Ведь одной из основных задачей проекта это все-таки не только комфорт и удобство, но и экономия электроэнергии.

С помощью вот этого участка кода и подключенной готовой библиотеки TroykaCurrent.h будем считывать показания с датчика тока. (рисунок 7.2.5)

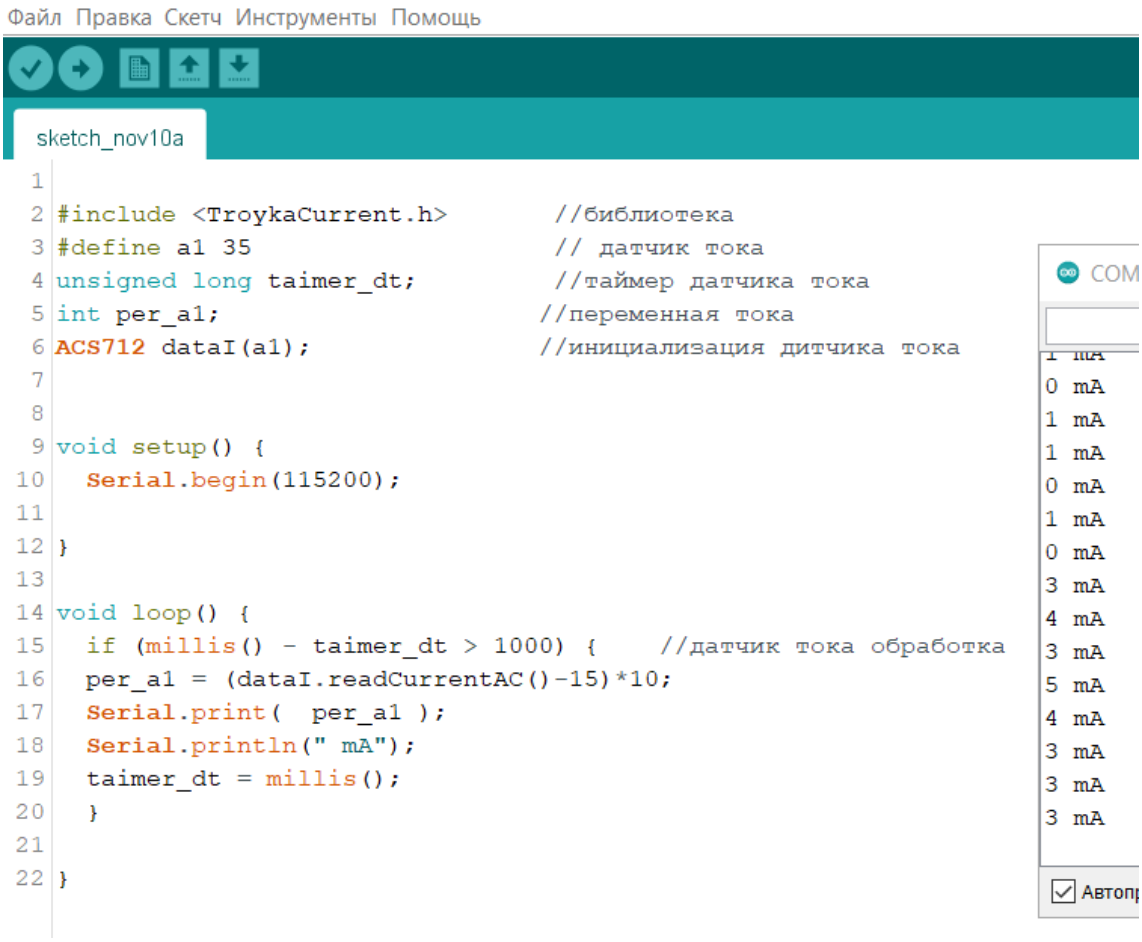


Рисунок 7.2.5 Программирование датчика тока (фото автора)

Датчик тока имеет погрешность до 2 мА поэтому для измерений он абсолютно не годится но проект и не предполагает точные измерения. От него нам нужна тока информация. Есть ток или нет, а для этого он подходит. Фиксируем показания датчика при выключенном свете и при включенном. Как видно из показаний

(рисунок 7.2.5.1) при выключенном свете датчик показывает от 0 до 1 мА, при включенном свете потребление тока составляет не меньше 3мА. Это значение и возьмем для индикации наличия тока. Теперь микроконтроллер будет знать, что если переменная «a1» > или = 3 то свет включен. Для удобства можно создать еще одну переменную с типом данных bool (принимает значения 0 или 1) назовем ее например (флаг датчика тока 1 == flag_a1) и будем хранить в ней информацию о том включен свет или нет.

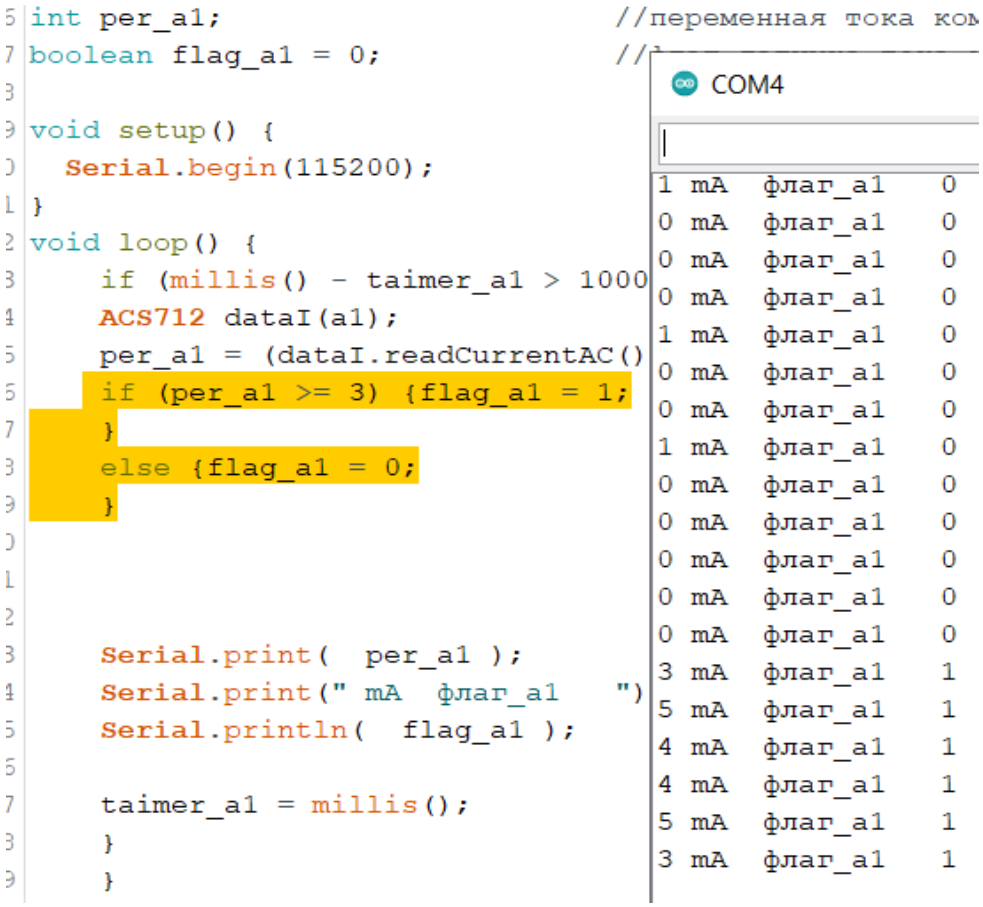


Рисунок 7.2.5.1 Флаг датчика тока (фото автора)

Точно таким же образом проверяем работу остальных трех датчиков тока и создаем свои флаги (flag_a2, flag_a3, flag_a4). Все это делается опять же для удобства и облегчения написания и читаемости программы.

7.2.6 Программа фоторезистора

Проверяем показания с фоторезисторов (рисунок 7.2.6)

```
#define f0     36  //фоторезистор отслеживающий уличное освещение
#define f1    39  //фоторезистор отслеживающий освещение в комнате

int per_f0;         // переменная  фоторезистор отслеживающий уличное освещение
int per_f1;         // переменная  фоторезистор отслеживающий освещение в комнате

boolean flag_f0;     //флаг ф улица
boolean flag_f1;     //флаг ф комнота
unsigned long taimer_f0 = 0;

void setup() {
Serial.begin (115200);

}

void loop() {
if (millis() - taimer_f0  > 1000) {          //фоторезистор
    per_f0 = analogRead(f0);
    per_f0 = map(per_f0, 0, 4095, 0, 100);  // перевести в диапазон 0.. 100
    per_f0 = constrain(per_f0, 0, 100);     // ограничить диапазон 0.. 100
    per_f0 = per_f0 + 13;

    per_f1 = analogRead(f1);
    per_f1 = map(per_f1, 0, 4095, 0, 100);  // перевести в диапазон 0.. 100
    per_f1 = constrain(per_f1, 0, 100);     // ограничить диапазон 0.. 100
    taimer_f0 = millis();
    Serial.print("  f0 ");
    Serial.print(per_f0);
  Serial.print("  f1 ");
  Serial.println(per_f1);

```

```
COM4
f0 71   f1 70
f0 71   f1 71
f0 71   f1 70
f0 71   f1 71
f0 71   f1 70
```

Рисунок 7.2.6

Показания фоторезистора считать и обработать не так сложно как показания переменного тока с датчика тока. Поэтому здесь не будем пользоваться ни какими библиотеками. Аналоговый вход ESP32 имеет 12 битное разрешение, а это значит, что диапазон показаний фоторезистора может меняться в пределах от 0 до 4095, для осуществляемого проекта это слишком высокая чувствительность, которая может повлиять на корректную работу системы в момент пограничного состояния, между светло или темно. Поэтому с помощью функции “map(, , , ,)” датчик необходимо загрубить.

Функция -map(, , , ,) преобразовывает значение переменной из одного диапазона в другой.

Меняем диапазон с 0 до 4095, на с 0 до 100. Теперь переменная не так чувствительна к изменениям и не реагирует на небольшое затенение. В дальнейшем если этого все равно будет не достаточно, можно будет еще понизить порог чувствительности переменной.

Полученные показания с фоторезистора сравниваем с показаниями люксметра и создаем флаг фоторезистора, по которому микроконтроллер будет понимать светло в помещении или темно.

7.3.1 Алгоритм включения выключения света

Основной алгоритм включения выключения света будет работать, как и в блок схеме.

Для демонстрации работы программы в порт выведены следующие переменные.

Показания фоторезистора (f1), флаг фоторезистора, состояние реле(включено/выключено), показания датчика тока в миллиамперах, флаг датчика тока и режим работы. ПИР датчик работает по прерыванию, поэтому он отображает свое значение тока, когда у него происходит какое нибудь событие, а событий у него может быть два, обнаружил движение или не обнаружил движение в промежуток времени заданный программой, в нашем случае это 8 секунд. (рисунок 7.3.1)

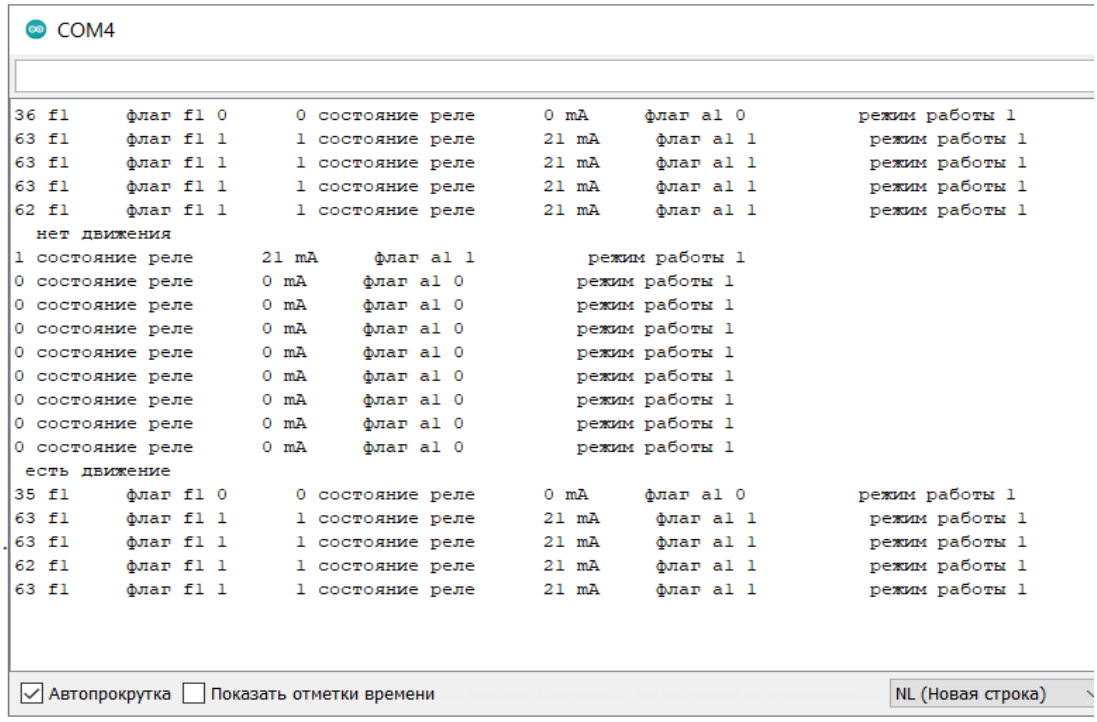


Рисунок 7.3.1 Автоматически режим работы (фото автора)

Из демонстрации видно, что при отсутствие движения микроконтроллер перестает опрашивать фоторезистор, так как какая разница светло в помещение или темно, если там все равно ни кого нет. Зато он опрашивает датчик тока, видит, что свет горит, так как через него проходит 21 миллиампер и тут же

переключает состояние реле, тем самым меняя положение контактов реле и выключая свет.

При обнаружении движения ESP 32 начинает опрашивать фоторезистор и в соответствие с их показаниями присваивает значение флагу фоторезистора 0(темно) или 1(светло). И если в помещении темно, то микроконтроллер опять меняет состояние реле, тем самым включая свет, что можно определить по показаниям датчика тока (21 мА). Вот по такому простому принципу программа будет работать в автоматическом режиме.

В ручном режиме все еще проще, микроконтроллер опрашивает только датчики тока. (рисунок 7.3.1.1)

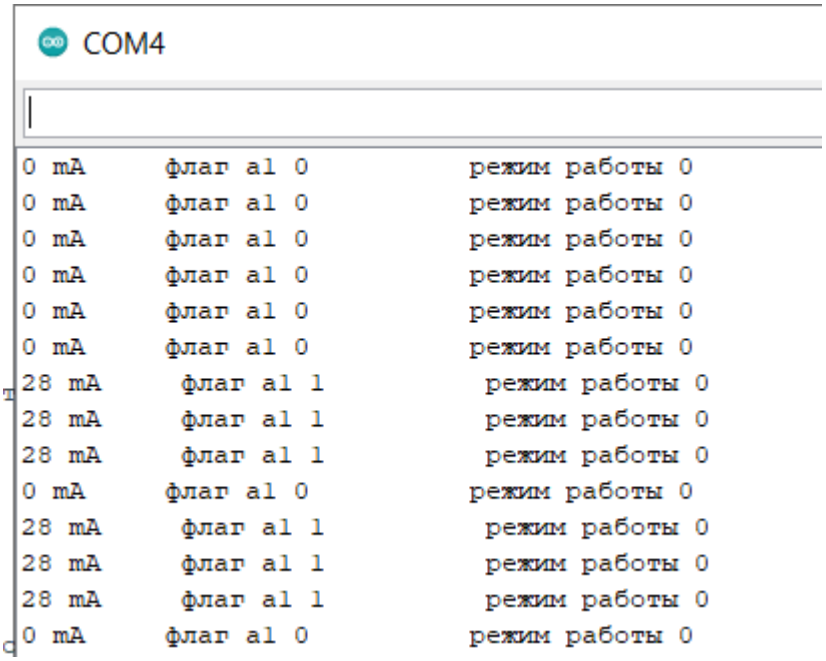


Рисунок 7.3.1.1 Ручной режим работы (фото автора)

Включаем/выключаем свет выключателями или удаленно, автоматика ни как не будет на это реагировать.

Из кода (см. рисунок 7.3.1.2) видно, что в автоматическом режиме существует всего два сценария, при которых будет происходить переключение реле.

```
if ((rezim == 1) && (flag_pir == 0) && (flag_dat_toka1 == 1))
{
  flag_rele1 = digitalRead(rele1);
  digitalWrite(rele1, !flag_rele1);

}
else if ((rezim == 1) && (flag_pir == 1) && (flag_perf1 == 0))
{
  flag_rele1 = digitalRead(rele1);
  digitalWrite(rele1, !flag_rele1);
}
```

Рисунок 7.3.1.2 Условие переключения реле (фото автора)

1) Микроконтроллер работает в автоматическом режиме И в комнате ни кого нет И свет включен. Все три условия совпадают, свет выключаем.
2) Микроконтроллер работает в автоматическом режиме И в комнате кто то есть И освещения не хватает. Все три условия совпадают, включаем свет.

При любых других сценариях, ни каких манипуляций со светом, ESP32 самостоятельно производить не будет.

7.3.2 Аварии при включении и выключении света

В программе аварии разбиты на два сценария. Первый сценарий при включении света, второй при выключении. Определять аварии будем при помощи датчика тока.

1) если включили свет и в течении заданного времени ток не появится, записывается авария и механизм помещения будет выключен и больше не будет опрашиваться до тех пор пока не будут соблюдены условия при которых аварии нет.
2) Второй сценарий осуществляется при выключении света и работает точно по такому же принципу только наоборот. Если выключить свет и датчик тока все равно показывает наличие тока, то после определенного периода времени выключаем реле.

Смотреть рисунок 7.3.2

```
//.........................аварии.....................//
if ((flag_pir == 0)&&(flag_dat_toka1 == 1)&&avarii==1&& millis()-avarijaTimer_a > 5000) {
  avarija = "есть";
  digitalWrite(rele1, LOW);
}
if ((rezim == 1) && (flag_pir == 0) && (flag_dat_toka1 == 0))
{
  avarija = "нет";
  avarii = 0;
}
if ((flag_pir == 1)&&(flag_perf1 == 0)&&avarii_vykl==1&& millis()-avarijaTimer_f >5000) {
  avarija = "есть";
  digitalWrite(rele1, LOW);
}
if ((rezim == 1) && (flag_pir == 1) && (flag_perf1 == 1))
{
  avarija = "нет";
  avarii_vykl = 0;
}
//............................................................//
```

Рисунок 7.3.2 Аварии при включении и выключении (фото автора)

7.3.3 Программа работы жалюзей и аварии работы жалюзей

Управление редуктора жалюзей происходит по средствам изменения положения контактов реле К5 - К6, концевых выключателей КВ1 – КВ2 и фоторезистора уличного освещения Ф0. Инициатором какого либо действия жалюзей (движение вверх или вниз) в автоматическом режиме является переменная фоторезистора. В зависимости от освещения на улице переменная Ф0 принимает значение 0(темно) или 1(светло). Если значение переменной = 1, то есть на улице светло необходимо открыть жалюзи, если значение переменной = 0 то есть на улице темно, необходимо закрыть жалюзи. Положение жалюзей микроконтроллер определяет по состоянию концевых выключателей.

Для определения состояния концевых выключателей создаем переменные принимающие значения 0 (разомкнут) и 1 (замкнут)

Если на улице светло и концевик верх не замкнут, микроконтроллер переключает реле К5 тем самым начав вращение редуктора до тех пор пока концевик не замкнется. Когда концевик замкнеться реле опять переключится и остановит редуктор. Аналогичным образом система работает на закрытие жалюзей.

Сразу пропишем в этом блоке программы и аварии. Авария и остановка редуктора произойдет в том случае если по истечению времени необходимому для определения расстояния от низа вверх или наоборот концевик так и не сработает, МК остановит редуктор и переведет режим работы жалюзей в ручной режим тем самым исключив какое либо движение. Для этого будем запускать таймер при работе редуктора. (рисунок 7.3.3)

```
//..........открытие жалюзей.......................//
if(rezimZaluzi == 1 && flag_f0 == 1&& per_kv1 ==0){
  digitalWrite(k5,HIGH);                  //открытие жалюзей
  per_k5 = 1;
}
 else {
  digitalWrite(k5,LOW);                   // остановка жалюзей
  per_k5 = 0;
  TimerZaluziK5 = millis();               //запуск аварийного таймера
 }
 //............авария при открытие жалюзей...........//
if ( per_k5 == 1 && millis()-TimerZaluziK5 > 7000) {
  digitalWrite(k6,LOW); //
  rezimZaluzi = 0;
  avarija  = "авария";
 }
//..........закрытие жалюзей.......................//
if(rezimZaluzi == 1 && flag_f0 == 0 && per_kv2 == 0){
  digitalWrite(k6,HIGH);
  per_k6 = 1;
}
else{
  digitalWrite(k6,LOW);
  per_k6 = 0;
  TimerZaluziK6 = millis();
}
//............авария при открытие жалюзей...........//
 if ( per_k6 == 1 && millis()-TimerZaluziK6 > 7000) {
  digitalWrite(k6,LOW);
  rezimZaluzi = 0;
  avarija  = "авария";
 }
```

Рисунок 7.3.3 Программа работы жалюзей и аварии работы жалюзей (фото автора)

8. УДАЛЕННОЕ УПРАВЛЕНИЕ

Согласно техническим характеристикам ESP32 имеет и поддерживает текие беспроводные виды связи как WiFi, Bluetooth и Bluetooth Low Energy (Bluetooth с пониженным энергопотреблением)

Wi-Fi — технология беспроводной локальной сети с устройствами на основе стандартов IEEE 802.11 [19].

Bluetooth - это беспроводной стандарт связи для обмена данными на коротком расстоянии, работает на частоте 2.4 ГГц [19].

Bluetooth Low Energy (BLE) – это энергосберегающий вариант Bluetooth, и его главная область применения – это передача маленьких порций данных на короткие расстояния [19].

Как уже видно из самих определений, самый экономный вариант для удаленного управления это Bluetooth Low Energy (BLE). Но в данном проекте хотелось попробовать не только локальное управление на коротком расстоянии, но удаленное на любое расстояние, через глобальную сеть интернет. По этому выбираем вид связи Wi-Fi.

8.1 Веб – сервер

Веб-сервер — сервер, принимающий HTTP-запросы от клиентов, обычно веб-браузеров, и выдающий им HTTP-ответы, как правило, вместе с HTML-страницей, изображением, файлом, медиа-потоком или другими данными [19].

В проекте общение с ESP32 будет происходить по схеме запрос-ответ. Для этого делаем из микроконтроллера ESP32 веб-север.

ESP32 может служить веб-сервером, прослушивающим HTTP-запросы от клиентов. То есть, если клиент сделает запрос, ESP32 отправит ему HTTP-ответы.

Соединяющим звеном между сервером и клиентом будет роутер.

Программу для создания веб-сервера из есп32, взята из интернет ресурса и интегрирована в существующую программу автора. (https://randomnerdtutorials.com/ [20]).

Теперь схема связи между пользователем и есп32 будет выглядеть вот так. (рисунок 8.1)

Рисунок 8.1 (фото автора)

8.2 HTML страница

HTML (Hypertext Markup Language) - это код, который используется для структурирования и отображения веб-страницы и её контента [18].

Создаем такой код, и интегрируем, с помощью функции client.println в программу. (рисунок 8.2)

```
client.println("<!DOCTYPE HTML><html><head>");
client.println("<meta name=\"viewport\" content=\
"width=device-width, initial-scale=1\"></head>");
client.println ("<meta charset=\"utf-8\">");

client.println ("<h1>РЕЖИМ РАБОТЫ</h1>");
client.println ("<p>ВЫБЕРИ  <a href=\"manual\">
<button>РУЧНОЙ</button></a> <a href=\"avto\">
<button>АВТОМАТИЧЕСКИЙ</button></a> " + rezim_raboty + "</p>");

client.println ("<h1>ЖАЛЮЗИ</h1>");
client.println ("<p>ВЫБЕРИ  <a href=\"manual\">
<button>РУЧНОЙ</button></a> <a href=\"avto\">
<button>АВТОМАТИЧЕСКИЙ</button></a> " + rezim_raboty_zal + "</p>");
client.println ("<p>ЖАЛЮЗИ  <a href=\"otkryt\"><button>ОТКРЫТЬ</buttor


client.println("<h1>КУХНЯ</h1>");
client.println("<p>ВЫКЛЮЧАТЕЛЬ  <a href=\"onk\"><button>ВКЛЮЧИТЬ</butt
```

Рисунок 8.2 HTML код (фото автора)

Что будет отображаться на HTML странице, при нажатии какой ни будь кнопки, задаем при помощью метода GET, действия которые должен будет выполнять микроконтроллер после нажатия кнопки прописываем ниже. (рисунок 8.2.1)

```
if (strstr(linebuf, "GET /manual") > 0) {
  rezim = 0;
  rezim_raboty = "ручной";
}
else if (strstr(linebuf, "GET /avto") > 0) {
  rezim = 1;
  rezim_raboty = "автоматический";
}
else if (strstr(linebuf, "GET /vkl_kuh") > 0)

  digitalWrite(relel, HIGH);

}
else if (strstr(linebuf, "GET /vykl_kuh") > 0)
  digitalWrite(relel, LOW);
```

Рисунок 8.2.1 Действия ESP 32 после нажатия на кнопку (фото автора)

Из логики программы видно, что например если пользователь нажмет на кнопку «ручной» в разделе «режим работы» то микроконтроллер переменной «rezim» присвоит значение 0, а текстовой переменной «rezim_raboty» слово «ручной». Если нажмем на кнопку «включить» в разделе «кухня» то микроконтроллер включит свет. (рисунок 8.2.1)

Рисунок 8.2.1HTML страница (фото автора)

8.3 ЗАЩИТА ПАРОЛЕМ

Защита паролем создается очень просто. Достаточно в существующий код добавить всего одно условие и одну переменную. indexOf осуществляет поиск символа или подстроки в строке. (рисунок 8.3.1)

Сам логин и пароль в программу заносится в кодировке base64.

Base64 — стандарт кодирования двоичных данных при помощи только 64 символов ASCII[19].

Перевести логин и пароль в кодировку можно либо по таблице, либо воспользоваться онлайн преобразователем, которых достаточно имеется в интернете.

Например denis:1234 в коде Base64 выглядит вот так ZGVuaXM6MTIzNA==

```
String parool;
if(parool.indexOf("ZGVuaXM6MTIzNA==") >= 0) {
              client.println("HTTP/1.1 200 OK");
               client.println("Content-type:text/html");
               client.println("Connection: close");
               client.println(); //  "Соединение: отключено"
               client.println();

               client.println("<!DOCTYPE HTML><html><head>");
}

               else {
               client.println("HTTP/1.1 401 Unauthorized");
               client.println("WWW-Authenticate: Basic realm=\"Secure\"");
               client.println("Content-Type: text/html");
               client.println();
               client.println("<html>пороль не верен</html>");
 }
```

рисунок 8.3.1 Защита паролем (фото автора)

Теперь после ввода в браузере IP адреса микроконтроллера, появиться окно с пустыми полями, куда необходимо будет ввести имя пользователя и пароль, для продолжения работы. (рисунок 8.3.2)

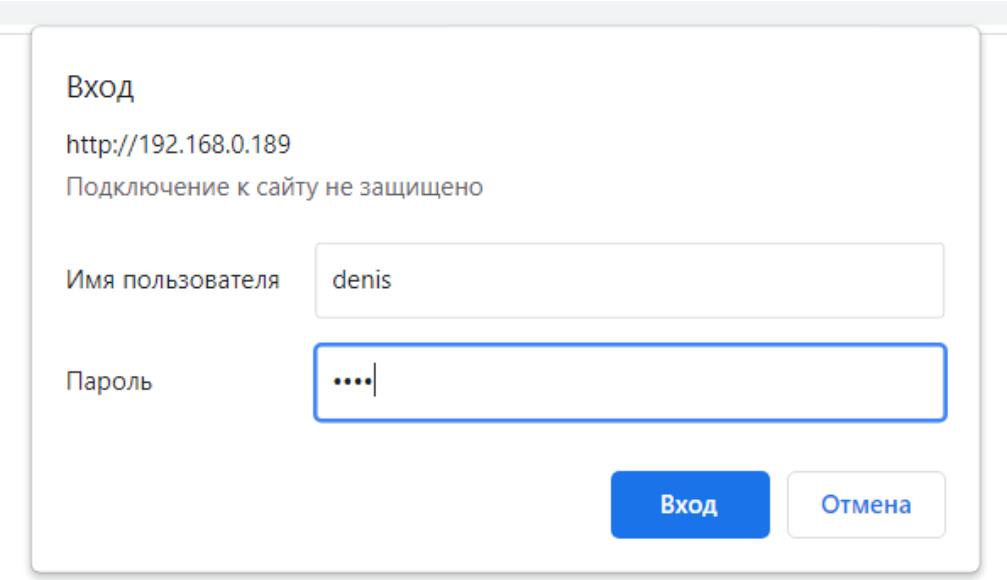


Рисунок 8.3.2 Отображение поля ввода пароля на HTML странице (фото автора)

Теперь можно пользоваться системой автоматики.

Скетч использует 669006 байт (51%) памяти устройства. Всего доступно 1310720 байт.

Глобальные переменные используют 38708 байт (11%) динамической памяти, оставляя 288972 байт для локальных переменных. Максимум: 327680 байт.

9. УПРАВЛЕНИЕ ЧЕРЕЗ ГЛОБАЛЬНУЮ СЕТЬ ИНТРНЕТ

Интернет вещей это концепция сети передачи данных между физическими объектами, оснащёнными встроенными средствами и технологиями для взаимодействия друг с другом или с внешней средой [18].

9.1 ИНТЕРНЕТ ВЕЩЕЙ Arduino IoT Cloud

Публичная версия Arduino IoT Cloud с автоматическим созданием панели управления и безопасной передачей на основе TLS предоставляет большие возможности [18].

При разработке Arduino IoT Cloud также учитывались вопросы безопасности, начиная с этапа проектирования на протяжении всего срока службы продукта. Пользователи могут безопасно подключать и авторизовать устройства в IoT Cloud, обеспечивая при этом конфиденциальность данных и устойчивость к внешним атакам [18].

Система авторизации клиента, использующая асимметричный ключ, получила сертификат X.509, и весь сетевой трафик в / из облака защищен с помощью TLS [18].

Arduino IoT Cloud автоматически создает черновой набросок программы. (рисунок 9.1)

Подключение проекта осуществляется очень просто и не занимает много времени. Так как разработчики практически обо всем позаботились.

1) Регистрация на сайте
2) Конфигурация микроконтроллера
3) Регистрация переменных
4) Визуализация переменных на панели приборов

После этой не сложной процедуры платформа автоматически создаст черновой набросок для пользовательской программы. Для элементов приборной панель которые будут задавать значения переменным программы, будут созданы свои функции.

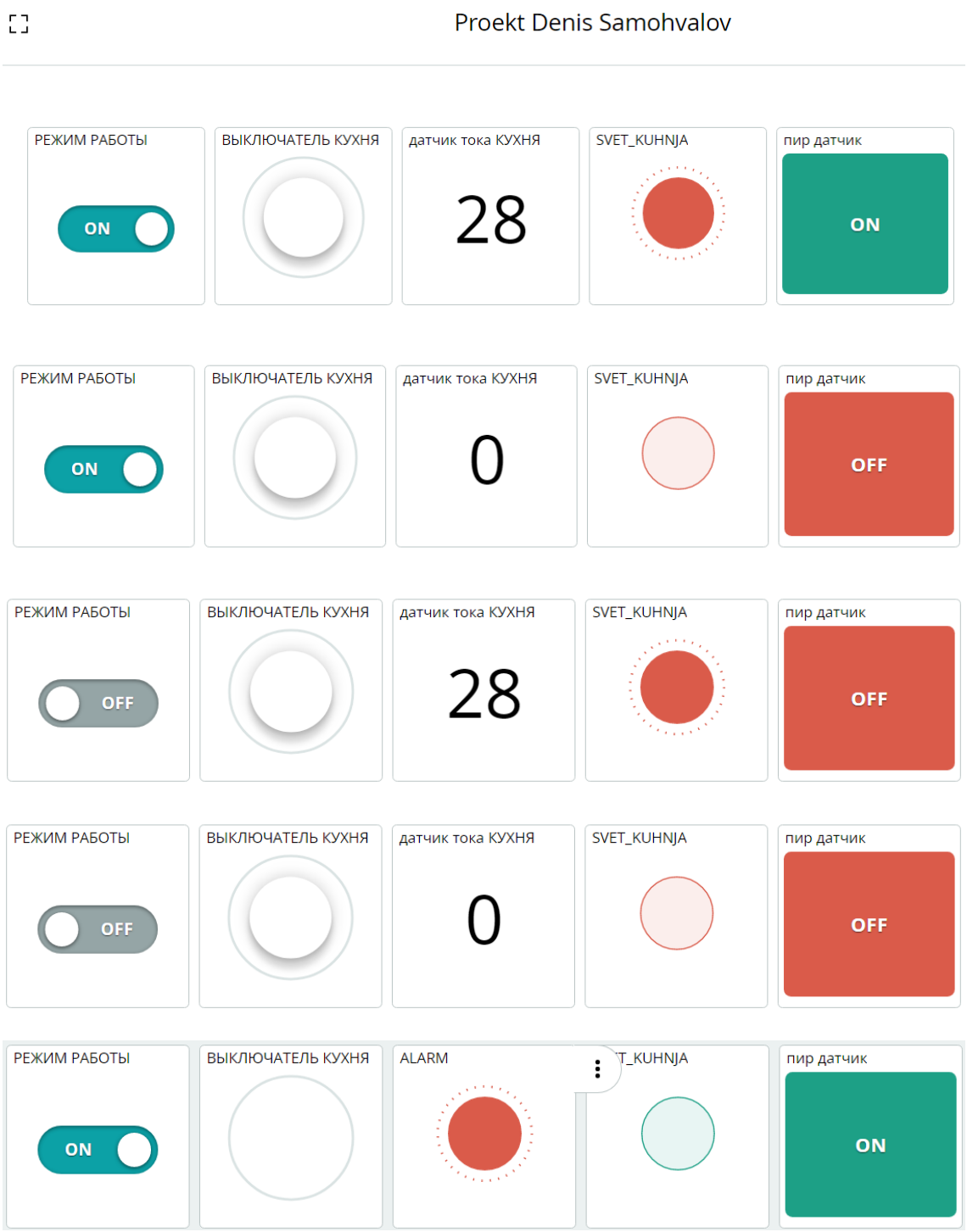
```
41 void loop() {
42   ArduinoCloud.update();
43   // Your code here
44
45
46 }
47
48 /*
49   Since Psodfiu is READ_WRITE variable, onPsodfiuChange() is
50   executed every time a new value is received from IoT Cloud.
51 */
52 void onPsodfiuChange()  {
53   // Add your code here to act upon Psodfiu change
54 }
```

Рисунок 9.1 Черновой набросок программы (фото автора)

Теперь уже написанную программу копируем в этот набросок по принципу, функцию написанной программы, в соответствующую функцию наброска. Заданным переменным в Arduino IoT Cloud присваиваем значение переменных из программы, которые мы собираемся отображать. Остается загрузив все это в микроконтроллер и пользоваться. Загружать можно напрямую из веб–редактора

Arduino IoT Cloud. Для управления с телефона или планшета существует бесплатное приложение Arduino IoT Cloud Remote.

Недостаток платформы это ограничения при использовании бесплатной версии. Не более пяти элементов. (рисунок 9.1.2) За 2 доллара в месяц можно увеличить до 10. За 6 долларов в месяц количество элементов не ограничено.



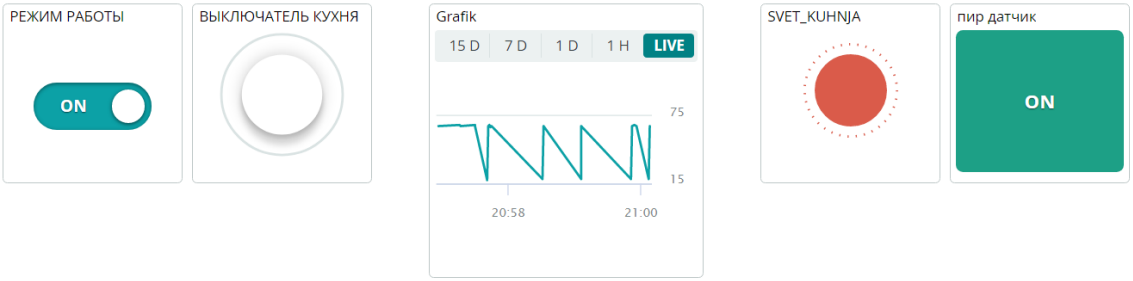


Рисунок 9.2 Панель управления (фото автора)

Программа использует 933718 байт (71%) места для хранения программ. Максимум 1310720 байт.

Глобальные переменные используют 40856 байт (12%) динамической памяти, оставляя 286824 байта для локальных переменных. Максимум 327680 байт.

9.2 REMOTEXY

RemoteXY это еще один очень интересный проект по удаленному управлению микроконтроллеров при помощи мобильных устройств. (рисунок 9.2) Особо описывать здесь нечего, этим ресурсом надо просто пользоваться.

RemoteXY - это система разработки и использования мобильных графических интерфейсов для управления контроллерами со смартфона или планшета. В состав системы входят: [21]

1) Редактор мобильных графических интерфейсов для контроллеров, размещенный на сайте remotexy.com [21].
2) Мобильное приложение RemoteXY, позволяющее подключаться к контроллеру и отображать графические интерфейсы [21].

Отличительные особенности:

1) Конфигурация графического интерфейса хранится в контроллере. При подключении нет никакого взаимодействия со сторонними серверами для того что бы загрузить графически интерфейс. Конфигурация графического интерфейса загружается в мобильное приложение из контроллера [21].
2) С одного мобильного приложения вы можете управлять всеми своими устройствами. Количество устройств не ограничено [21].

Поддерживаются следующие способы связи между контроллером и мобильным устройством:

1) Интернет из любого места через облачный сервер [21]
2) WiFi в режиме клиента и точки доступа [21]

3) Bluetooth [21]
4) Ethernet по IP адресу или URL [21]

Рисунок 9.2 RemoteXY [21]

Как видно из описания приложение дает возможность подключения не только через облако, но также WiFi в режиме клиента и точки доступа и Bluetooth, что дает возможность управлять микроконтроллером без интернета, так как весь графический интерфейс хранится в микроконтроллере. Есть возможность организации многоканальной связи, это даст возможность управлять одновременно с нескольких устройств и различными видами связи одновременно (WiFi, Bluetooth).

Графических элементов приложение имеет столько же как и Arduino IoT Cloud.

Подключение проекта осуществляется так же просто как и в Arduino IoT Cloud.

Так как вся программа храниться в микроконтроллере, то в приложении нет ограничений на их количество даже в бесплатной версии.

Бесплатная версия приложения предлагает пользователю пять графических элементов. Платная не ограничивает количество элементов и стоимость приложения составляет 7.99 на неограниченное время пользования, что гораздо дешевле Arduino IoT Cloud.

Для небольших проектов, как проект автора — это наверно лучший вариант.

ЗАКЛЮЧЕНИЕ

Данная дипломная работа была посвящена созданию системы автоматизации и удаленному управлению освещения в жилых помещениях. Целью которой было бы обеспечение удобного и надежного управления и контроля освещением, в любое время суток, с любого места при условии наличия в этом месте интернета. Вторая задача проекта, но не менее важная это эффективный и экономный расход электроэнергии, что в конечном результате должно способствовать сохранению природных ресурсов и улучшению экологии.

В результате анализа подобных существующих систем, при создании собственной системы автором был сделан акцент, во-первых, на итоговую стоимость проекта (но не в ущерб ее функциональным возможностям), так как более дешевая система доступна большему количеству людей, во-вторых на способность данной системы быть интегрированной в существующие системы управления освещением и успешно с ними взаимодействовать.

В дипломной работе, рассмотрен и аргументирован выбор оборудования (микроконтроллер, датчики, исполнительные механизмы), выбран языка программирования алгоритма и написан алгорим, выбран способ визуализации системы и инструменты удаленного управления. При программировании алгоритма учтены возможные аварийные ситуации, блокировки системы при возникновении опасностей или выхода из строя какого-либо элемента оборудования, разработана система оповещения пользователя.

KOKKUVÕTE

Käesoleva lõputöö teema on eluruumide valgustuse kaugjuhtimise automatiseerimissüsteemi loomine, mille ülesanded on:

- internetiühenduse olemasolul tagada eluruumide valgustuse mugavat ja efektiivset juhtimist ning kontrollimist olenemata kellaajast ja kontrolliva isiku asukohast;
- tagada elektrieneergia efektiivset ja ökonoomset tarbimist, mis aitab loodusressursside säilitamisele ja ökoloogia parandamisele kaasa.

Kõigepealt teostas autor tänapäeva taoliste juhtimis- ja kontrollimissüsteemide analüüsi ja saadud tulemuste alusel tuli järeldusele, et tema loodav juhtimis- ja kontrollimissüsteem peaks vastama kahele peatingimusele:

- esiteks kuna odav süsteem on paljudele inimestele kättesaadavam, siis süsteemi üldmaksumus ei pea liiga kallis olema, kuid samas ei pea kokkuhoid süsteemi funktsionaalsuse vähendamist põhjustama;
- teiseks loodaval juhtimissüsteemil peab olema koostöö- ja integreerumisvõime teiste töötavate juhtimissüsteemidega.

Autor käsitles antud lõputöös erinevaid tööseadmeid (need on mikrokontrollerid, andurid, täitemehhanismid jne), valis nende hulgast kõige sobivamad ja põhjendas oma otsust.

Samuti koostas ta algoritmi ja teostas algoritmi tõlkimist valitud programmeerimiskeelde. Peale selle valis autor süsteemi visualiseerimise võtted ja kaugjuhtimise tööriistad.

Autor peab tähtsaks, et algoritmi koostamisel arvestas ta võimalikke avariisituatsioone ja süsteemi blokeeringu toimumist avarii, ohtu või mõne elemendi rikke korral. Tema arvates oli otstarbekas süsteemi optimaalseks funktsioneerimiseks läbi mõelda tarbijate teavitamissüsteemi, mille töötas ta ka välja.

SUMMARY

This thesis was devoted to the creation of an automation system and remote control of lighting in residential premises. The purpose of which would be to provide convenient and reliable control and monitoring of lighting, at any time of the day, from any place, subject to the presence of the Internet in this place. The second task of the project, but no less important, is the efficient and economical consumption of electricity, which in the end should contribute to the preservation of natural resources and the improvement of the environment.

As a result of the analysis of such existing systems, when creating his own system, the author focused, firstly, on the final cost of the project (but not to the detriment of its functional capabilities), since a cheaper system is available to more people, and secondly, on the ability of this systems to be integrated into existing lighting control systems and successfully interact with them.

In the thesis, the choice of equipment (microcontroller, sensors, actuators) was considered and argued, the programming language of the algorithm was chosen and the algorithm was written, the method of visualizing the system and the tools for remote control were chosen. When programming the algorithm, possible emergency situations, system blocking in the event of dangers or failure of any piece of equipment were taken into account, a user alert system was developed.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Система управления освещением [Online] https://www.revolight.ru/partners/learning/documents/chto-takoe-sistema-upravleniya-osveshcheniem/ 03.09.2021
2. Диммирование [Online] https://trialight.ru/lighting-control-articles/characteristics-and-types-of-dimming.html 03.09.2021
3. Стандарт 0–10В [Online] https://trialight.ru/lighting-control-articles/standard-0-10v-dimming.html 03.09.2021 03.09.2021
4. Протокол DSI [Online] https://trialight.ru/lighting-control-articles/dsi-dimming-protocol.html 05.09.2021
5. Интерфейс DALI [Online] https://trialight.ru/lighting-control-articles/dali-dimming-protocol.html 05.09.2021
6. Протокол беспроводной связи ZigBee [Online] https://trialight.ru/lighting-control-articles/zigbee-wireless-light-control.html 10.09.2021
7. Открытый стандарт автоматизации инженерных систем – KNX [Online] https://trialight.ru/lighting-control-articles/knx-light-control.html 12.09.2021
8. LoRaWAN [Online] https://trialight.ru/lighting-control-articles/street-lighting-control-with-lorawan-protocol.html 12.09.2021
9. Управление освещением по PLC [Online] https://trialight.ru/lighting-control-articles/lighting-control-with-plc.html 12.09.2021
10. Фоторезистор [Online] https://samelectrik.ru/chto-takoe-fotorezistory.html 12.09.2021
11. Пир датчик (HC-SR501) [Online] https://asenergi.com/catalog/datchiki/datchik-dvizheniya-hc-sr501.html 12.09.2021
12. Датчик тока ACS712 [Online] https://www.rlocman.ru/shem/schematics.html?di=113339 15.09.2021

13. Исполнительные механизмы реле [Online] https://arduinomaster.ru/datchiki-arduino/podklyuchenie-rele-k-arduino/ 16.09.2021
14. Редуктор [Online] https://aliexpress.ru/item/4000244559764.html 16.09.2021
15. Концевой выключатель LXW5-1124 [Online] https://nevel.uz/koncevoj-vyklucatel-lxw5-11m-15a-p640 16.09.2021
16. Микроконтроллер [Online] https://arduinomaster.ru/platy-arduino/esp32-arduino-raspinovka-arduino-ide/ 23.09.2021
17. Алгоритм программы [Online] https://systop.ru/arduino/53-programmirovanie-na-arduino-urok-2-osnovnye-algoritmy.html 23.09.2021
18. Программное обеспечение Arduino [Online] https://www.arduino.cc 23.09.2021
19. WiFi, Bluetooth и Bluetooth Low Energy [Online] https://ru.wikipedia.org/wiki/ 23.09.2021
20. Веб – сервер [Online] https://randomnerdtutorials.com/ 23.09.2021
21. RemoteXY [Online] https://remotexy.com/ 30.09.2021