Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: компьютерные системы и сети (КСиС)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

на тему:

«Встроенное программное обеспечение для парных сетевых светильников»

БГУИР КП  1-40 01 01 19 ПЗ

Студент:  гр. 051005 Сидоренко И.Д.

Руководитель: Шимко И.В.

Минск 2022

**СОДЕРЖАНИЕ**

Введение 3

1 Аналитический обзор существующих аналогов 4

1.1 Обзор аналогов 4

1.2 Постановка задачи 5

2 Проектирование устройства 6

2.1 Проектирование программной части 6

2.2 Проектирование аппаратной части 8

2.3 Связь компонентов 9

3 Разработка устройства 10

3.1 Разработка программной части 10

3.2 Разработка аппаратной части 12

4 Тестирование 14

5 Руководство пользователя программы 15

5.1 Подключение 15

5.2 Настройка 15

5.3 Управление 15

5.4 Поведение лампы 15

Заключение 16

Список использованных источников 17

Приложение А. Исходный код программы 18

**ВВЕДЕНИЕ**

Современный уровень технологий позволяет любому человеку в любой точке мира общаться со своими родственниками, друзьями, партнёрами и даже абсолютно незнакомыми людьми, но в этом есть и свои минусы. Каждый день человек получает десятки звонков и сотни сообщений, всё это превращается в рутину, обыденность и то, без чего современный человек не сможет прожить и дня в наше время. Это всё говорит о том, что цифровая жизнь всё больше поглощает людей и отбирает силу осязания, которая была у людей старшего поколения во время написания письма или получения бумажного чека зарплаты. Фотография дорогого Вам человека имеет большую ценность, чем гигабайты медиа в облачном хранилище. Социальные сети подарили нам удобство и комфорт, но мы в это время потеряли связь.

Всё это можно легко изменить, привнеся в Вашу жизнь пару сетевых светильников. Они возвращают те прикосновения, ту связь, что не получить в обычной переписке или звонке. Это и подойдёт для семьи, которая живёт далеко друг от друга; дети смогут «подмигивать» своим бабушкам и дедушкам, а те – им.

Цель данного проекта – объединить людей, которых разделяет расстояние. Всё, что нужно, – это простое прикосновение руки к устройству, чтобы почувствовать радость от мгновенной связи с другими людьми. Это мощный, но очень простой способ отправить сообщение любимому человеку, что вы думаете о нём.

1. **АНАЛИТИЧЕСКИЙ ОБЗОР СУЩЕСТВУЮЩИХ АНАЛОГОВ**

Для начала работы требуется узнать о нынешних вариантах исполнения похожих систем и продуктов и правильно поставить задачу для проекта. Незнание об аналогах разрабатываемого продукта зачастую приводит к неправильной постановке задачи, а неправильная постановка задачи влечёт за собой трату ресурсов на разработку продукта, который не превосходит, иногда даже уступает существующим аналогам, представленным уже на рынке.

В данном разделе будут рассмотрены существующие на данный момент «Парные лампы» или «Лампы дружбы» и поставлена задача для данного проекта, на основании изученного материала.

**1.1 Обзор аналогов**

Самый доступный вариант на данный момент был найден мной на сайте <https://www.friendlamps.com> . Но данный продукт стоит слишком много для обычного человека, а именно 179,99$ за две лампы и 94,99$ за одну штуку.

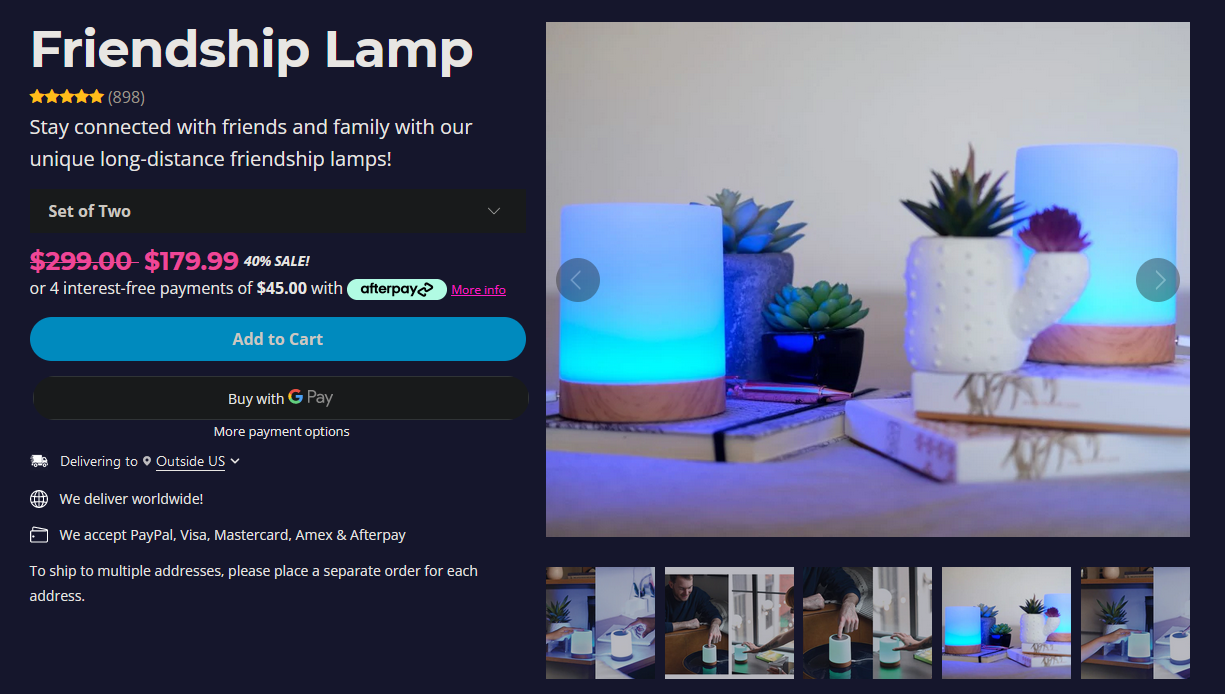


Рис. 1.1 – Фрагмент с сайта «Лампы дружбы»

Исполнение ламп приятное, смотрятся они очень красиво. В дополнение к ним идёт приложение, которое можно скачать для операционной системы Andriod и IOS. Единственное, что может оттолкнуть покупателя приобрести данный товар – это цена.

**1.2 Постановка задачи**

В рамках данного курсового проекта было принято решение разработать программное средство и аппаратную его часть «Лампы дружбы» на платформе аппаратного языка Arduino с использованием MQTT сервера от HiveMQ. Для аппаратной реализации была выбрана плата Wemos D1 mini, сенсорная кнопка и адресная светодиодная лента.

В обязательные требования входят:

* Возможность подключения устройства к любой точке доступа WiFi.
* Включение/выключение, смена цвета, мерцание и смена яркости одной кнопкой.
* Изменение поведения лампы в зависимости от подключения удалённой лампы-партнёра.
* Гибкая настройка лампы в браузере любого устройства, подключенного к одной сети с лампами/лампой.
* Использование протокола MQTT, для обмена данными между лампами.
* Обмен данными двух устройств через публичный сервер HiveMQ.
* Возможность самостоятельной доработки лампы в домашних условиях (возможность добавления ИК датчика для отслеживания движения в комнате, в которой стоит лампа).

1. **ПРОЕКТИРОВАНИЕ УСТРОЙСТВА**

Сам проект будет состоять из двух основных частей:

* Написание прошивки для платы (программная часть)
* Сбор требуемых компонентов и их прошивка (аппаратная часть)

**2.1 Проектирование программной части**

Для написания прошивки для платы Wemos D1 mini был выбран язык Arduino и приложение для написания Arduino IDE. Прошивка включает в себя 8 файлов расширения .ino, где описана вся программная часть продукта. Также было использовано 5 сторонних библиотек, которые можно найти в открытом доступе на Git Hub.

Две лампы должны общаться друг с другом с помощью протокола MQTT. MQTT — это протокол обмена сообщениями по шаблону издатель-подписчик (pub/sub). Этот протокол стал для потоковой передачи данных между устройствами с ограниченной мощностью CPU и/или временем автономной работы, а также для сетей с дорогой или низкой пропускной способностью, непредсказуемой стабильностью или высокой задержкой. Именно поэтому MQTT известен как идеальный транспорт для IoT. Он построен на протоколе TCP/IP.

Система связи, построенная на MQTT, состоит из сервера-издателя, сервера-брокера и одного или нескольких клиентов. Издатель не требует каких-либо настроек по количеству или расположению подписчиков, получающих сообщения. Кроме того, подписчикам не требуется настройка на конкретного издателя. В системе может быть несколько брокеров, распространяющих сообщения.

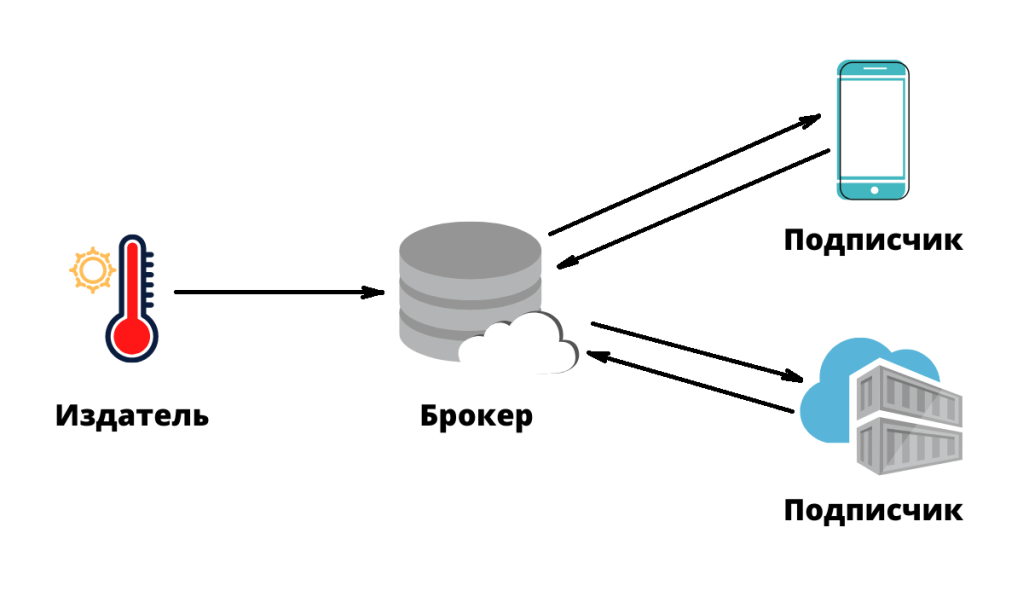


Рис 2.1 – Графическое представление работы MQTT

В качестве сервера был выбран бесплатный публичный сервер HiveMQ. Сообщения между сервером и получателем отправляются через веб-сокет в виде фреймов. Эти кадры имеют 2 байта служебных данных. Теперь все сообщения MQTT отправляются при помощи инфраструктуры веб-сокет. HiveMQ выступает в данной цепочке в качестве брокера.

Также было разработан одностраничный сайт для настройки и подключения ламп к сети WiFi. В панели «STATUS» можно узнать статус своей лампы и лампы-партнёра. В панели «SETTINGS» есть возможность с телефона изменить яркость своей лампы и цвет её свечения при помощи ползунков. В панели «WIFI» вводится имя и пароль от роутера для подключения к нему и последующей корректной работы. В панели «MQTT» есть возможность поменять имя своей лампы и лампы-партнёра, на которую будет ссылаться первая, адрес сервера (broker.mqttdashboard.com) и порт подключения к нему (1883).

**2.2 Проектирование аппаратной части**

Для начала была создана примерная схема аппаратной части, которая и использовалась в будущем с небольшими изменениями.

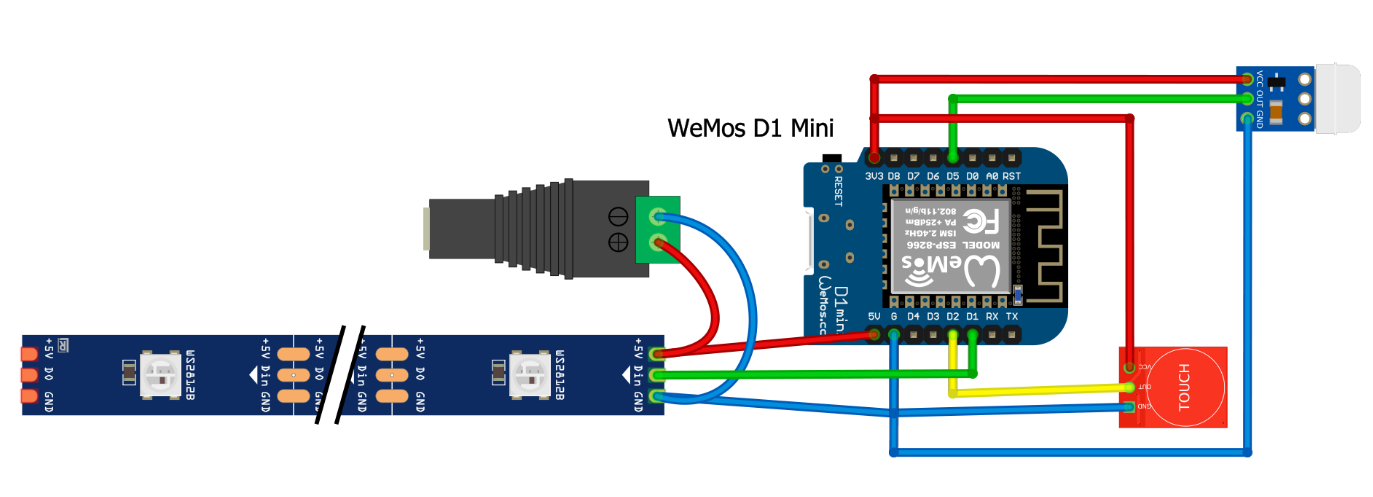


Рис. 2.2 – схема аппаратной части устройства с ИК датчиком

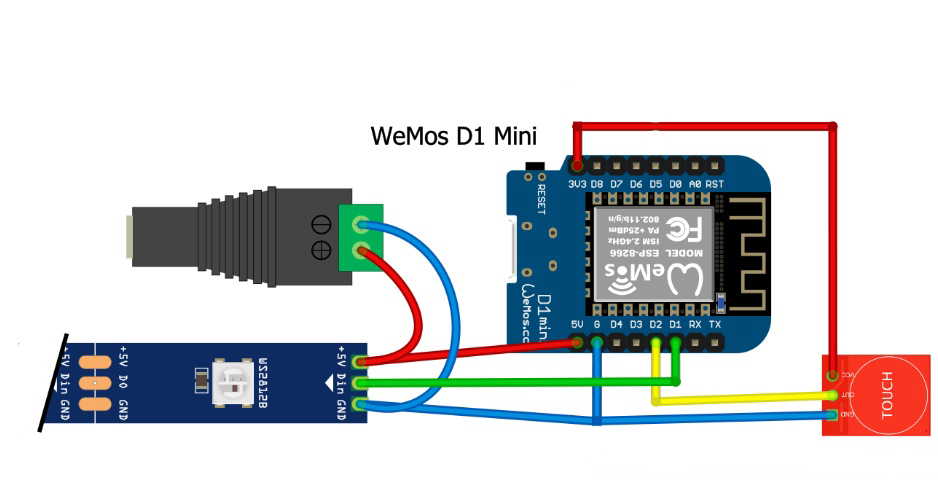


Рис. 2.3 – схема аппаратной части устройства без ИК датчика

Бюджета для внедрения ИК датчиков не хватило, поэтому было принято решение отказаться от него. Датчик можно будет в будущем припаять в соответствии с рисунком 2.2, а проект будет строиться вокруг схемы без него, представленной на рисунке 2.3.

**2.3 Связь компонентов**



Рис 2.4 – Фотография корпусов для проекта

Для сбора схемы были выбраны две лампы с питанием от USB, все компоненты были извлечены из корпуса, кроме платы питания и антенны, которая нам нужна будет для сенсорной кнопки.

Питание со входа 5V подаётся на адресную светодиодную ленту и на Wemos в соответствующие порты. Отдельное питание для сенсорной кнопки идёт от Wemos через выход 3V3 и подаётся на вход кнопки VCC. Минус питания отходит от входа и подаётся на адресную светодиодную ленту на пин GND, на плату Wemos на пин G и на кнопку на пин GND. Вход/выход (I/O) кнопки подключается к пину D2 платы. Канал управления адресной светодиодной лентой отходит от пина D1 на плате и входит на пин DIN ленты. Также было принято решение удлинить антенну сенсорной кнопки, для этого была перерезана одна дорожка, которая вела к стандартной антенне, оставшаяся часть дорожки была зачищена, залужена и к ней был припаян провод антенны. Также к схеме можно добавить ИК датчик, его подключение и все соответствующие пины продемонстрированы на рисунке 2.2.

1. **РАЗРАБОТКА УСТРОЙСТВА**

Оставшаяся часть работы заключается в том, чтобы собрать всю схему вместе, разработать и загрузить прошивку на плату Wemos, а после протестировать работу устройства. Все возникнувшие проблемы на данном этапе будут описаны в пункте 4 Тестирование.

**3.1 Разработка программной части**

Программная часть представляет собой программу на языке Arduino, для начала нам требуется указать в начале главного файла указать пины для подключения тех или иных компонентов к плате Wemos:

#define LED\_PIN D1 // пин ленты

#define BTN\_PIN D2 // пин кнопки

#define PIR\_PIN D5 // пин PIR (ИК датчика)

#define LED\_AMOUNT 5 // кол-вл светодиодов

#define BTN\_LEVEL 1 // 1 - кнопка подключает VCC, 0 - подключает GND

#define USE\_PIR 0 // 1 - использовать PIR (ИК датчик) на этой лампе

#define IGNORE\_PIR 1 // 1 - игнорировать сигнал PIR (ИК датчика) с удалённой лампы

На основании того, что лампы используют публичный сервер, требуется определить уникальные имена, чтобы не обращаться случайно к другим устройствам, использующих этот же сервер. Имена первой лампы – Asdfg125, второй – Asdfg325.

Также для подключения ламп к роутеру и последующего управления ими через телефон было принято решение при первом включении лампы пытаться подключиться к роутеру, если этого не происходит, то через 15 секунд или по нажатию пользователем кнопки на лампе включается точка доступа WiFi, при подключении к которой вас перекидывает на сайт «авторизации», где уже можно ввести имя и пароль от вашего роутера. Сайт был создан на основе библиотеки, которая находится в открытом доступе (название: GyverPortal.h). Само построение сайта и его функционал определён в файле webface.ino, там прописаны блоки, установлен их порядок и реализовано управление с телефона.

Основная часть программы управляет всеми остальными файлами, в которых прописаны функции для каждого действия пользователя. В файле Timer.h при вызове функции проверяется тайминг (к примеру для нажатия кнопки). В файле button.ino прописаны все функции для лампы:

* Включение/выключение по однократному нажатию на кнопку
* Двойное нажатие на кнопку меняет цвет света лампы.
* Тройное нажатие на кнопку посылает сигнал лампе-партнёру с заданием помигать три раза.
* Импульсное удержание кнопки увеличивает/уменьшает яркость лампы, но при этом не меняет интенсивность свечения лампы-партнёра.

Также было определено поведение лампы при различных сценариях:

* Лампа-партнёр не в сети: локальная лампа горит ровным светом.
* Удалённая лампа в сети: локальная лампа плавно пульсирует с периодичностью в одну секунду, тайм-аут 18 секунд.
* Сработал ИК датчик на лампе-партнёре: локальная лампа плавно пульсирует три раза в одну секунду, тайм-аут 60 секунд.

Функции управляющие адресной светодиодной лентой прописаны в файле func.ino, а также запуск портала и анимация работы локального портала.

Данные для передачи через MQTT прописаны в файле mqtt.ino:

Протокол:

GWL:0,ir // пакет heartbeat, состояние ИК

GWL:1,power,color,wink // пакет данных

GWL:2 // запрос настроек

В этом же файле мы опрашиваем mqtt, подключаемся к нему, принимаем пакеты с лампы-партнёра и отправляем пакеты ей.

В файле startup.ino прописаны действия для старта веб-лампы, анимация ленты при условиях, при подключении к точке доступа прописываем IP в память, выданный роутером при удачном подключении и стартуем веб-сокет с порталом.

**3.2 Разработка аппаратной части**

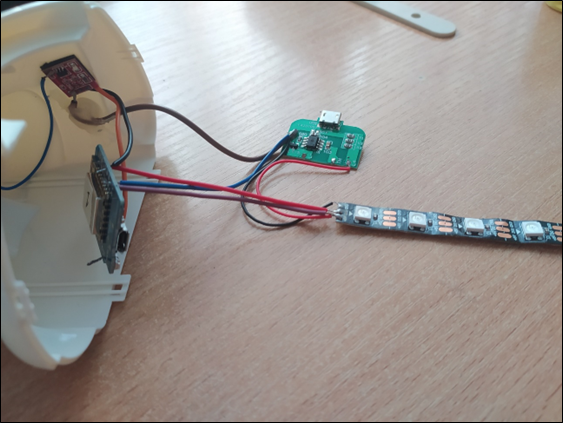


Рисунок 3.1 – Фотография собранной схемы проекта

Схема была разработана и собрана на основании схемы, представленной на рисунке 2.3. Все компоненты были соединены навесным монтажом и аккуратно сложены в корпус лампы. Также для корректной работы от стандартной схемы питания минус питания был проведён в обход диода, распаянного на плате. На фото видно все компоненты, применённые для реализации:

* Сенсорная кнопка (приклеена внутри корпуса; маленькая красная)
* Стандартная плата питания (слева сверху зелёная)
* Плата Wemos D1 mini (большая синяя плата с цифрой 1)
* Светодиодная адресная лента (слева чуть ниже стандартной платы питания)

Для данного проекта можно было бы выбрать обычную светодиодную ленту, а не адресную, но тогда пришлось бы прописывать для неё драйвер, а так получается намного меньше проводов и более удобное программирование компонентов.

Очень важным при прошивке платы Wemos было то, что при спаянной схеме нельзя просто подключить плату к компьютеру для прошивки, иначе всё может сгореть и мы потеряем дорогостоящие компоненты. Чтобы правильно прошить плату при собранной схеме требуется предварительно подать дополнительное питание на стандартную плату питания в размере 5V, а уже после подключать плату Wemos к компьютеру и прошивать. Также можно прошить плату до сбора всех комплектующих и уже собирать схему с загруженной прошивкой, что я и сделал. Но лучше всего во избежание разного рода ошибок прошить плату до сборки схемы и после, тогда шанс некорректной загрузки прошивки во много раз снижается.

Также стоит быть крайне внимательным с выходами и проводами, идущими от компонентов к плате, иначе она может сгореть или в лучшем случаи не работать. А также с правильным подключением светодиодной ленты (требуется соблюдать правильное направление при установке).

Дополнительно к этому было перерезана дорожка на сенсорной кнопке, идущая к встроенной антенне, остаток дорожки зачистили, залудили и припаяли антенну, которая шла в стандартной комплектации ламы. Также для снижения чувствительности антенны можно заменить металлическую накладку на какую-либо неметаллическую или запять конденсатор, для снижения чувствительности и предотвращения самопроизвольного нажатия кнопки.

После сборки все компоненты были приклеены к корпусу на двухсторонний скотч для предотвращения замыкания и неприятных шумов во время использования устройства.

1. **ТЕСТИРОВАНИЕ**

Тестирование производилось на протяжении всех этапов разработки и проектирования. Основными ошибками было то, что при загрузке прошивки на Wemos и дальнейшей работы лампы первый светодиод самопроизвольно мигал. Эта проблема была решена установкой ядра esp8266 более старой версии (2.7.4).

Также была решена проблема самопроизвольного срабатывания кнопки заменой металлической площадки на корпусе на неметаллическую, конденсатор для снижения чувствительности кнопки не понадобился.

При сборке схемы изначально питание шло с перебоями, из-за чего возникали ошибки в работе лампы, эта проблема была решена подачей минуса питания в обход диода, для этого была зачищена дорожка, ведущая к диоду, залужена и к ней припаяны повода-минусы.

Также было сделано несколько прошивок с доработкой функционала и исправлением ошибок:

* Отсутствие света на ленте
* Некорректная передача пакетов данных
* Большая задержка второй лампы при срабатывании кнопки на первой лампе
* Отсутствие подключения к роутеру
* Некорректная работа кнопки

Все эти ошибки были исправлены при написании программной части проекта. Единственная ошибка, которую не удалось исправить, – это задержка в срабатывании сценария на лампе-партнёре. Как оказалось проблема была в сервере. К сожалению, это нельзя как-то исправить, не прибегнув к замене публичного сервера на собственный или не публичный, но не публичные сервера требуют оплату, а для такого проекта нецелесообразно тратиться на него.

Также были проблемы с пайкой, некоторые провода отрывались от пинов плат, замыкались. А компоненты перегревались при пайке и приходилось их заменять на новые, а в нынешнее время их достаточно проблематично достать.

1. **РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ**

**5.1 Подключение**

* При подаче питания начнётся подключение к роутеру (лампа мигает зелёным)
  + Если подключение не будет установлено за 15 секунд или пользователь кликнет по кнопке, лампа запустит свою точку доступа WiFi (лампа мигает синим)
  + Точка называется **WLamp <ip адрес>**, где ip адрес – последний полученный от роутера адрес (успешное подключение)
  + При подключении к точке доступа должна автоматически открыться страница конфигурации (веб-интерфейс). Если не открылась – перейди в браузере по адресу **192.168.4.1**
  + При сохранении настроек ИЛИ при клике по кнопке лампа будет перезагружена и процесс подключения начнётся заново
* При успешном подключении лампа перейдёт в режим работы и включится настроенным цветом (если включена)
* Страница конфигурации лампы (веб-интерфейс) доступна в локальной сети роутера по IP адресу, который роутер выдал лампе. Чтобы его узнать, можно повторить процедуру подключения

**5.2 Настройки**

Нужно указать на странице конфигурации данные для подключения к роутеру, а также имя своей лампы (Local Name) и её удалённой пары (Remote Name)

**5.3 Управление**

* 1 клик: вкл/выкл
* 2 клика: сменить цвет
* 3 клика: подмигнуть
* Удержание: сменить яркость

**5.4 Поведение лампы**

* Удалённая лампа не в сети: локальная лампа горит ровным светом
* Удалённая лампа в сети: локальная лампа плавно мерцает раз в секунду, тайм-аут 18 секунд
* Сработал ИК датчик на удалённой лампе: локальная лампа плавно мерцает 3 раза в секунду, тайм-аут 60 секунд

**ЗАКЛЮЧЕНИЕ**

В результате курсового проекта было разработано программное обеспечение для парных сетевых ламп. И также собрано два рабочих макета.

Разработка приложения включала в себя решение множества задач и проблем, как итог было изучено множество веб-сайтов с продажей, отзывами и списком функционала аналогов, произведён их анализ и сформировано требование для курсового проекта.

В ходе разработки было изучено и задействовано множество новых технологий, таких как: протокол MQTT, работа с esp8266 и Wemos D1 mini, создание веб-сайта для управления и настройки ламп, работа с публичным бесплатным сервером HiveMQ. Также были приобретены навыки в пайке, подключении и проектировании прошивки для микроконтроллера.

Проект был отлажен живыми пользователями, так как любая лампа может работать в любой точке мира и провести тест на большом расстоянии с разными роутерами не составило никаких проблем. Требования и пожелания пользователей были учтены в данном проекте.

Как результат, мы имеем большой список освоенных технологий, новый язык программирования, отлаженный проект, с помощью которого два пользователя могут напоминать о себе и общаться даже без слов, только лишь прикосновениями.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

* Книга «Изучаем Arduino. Инструменты и методы технического волшебства» Джереми Блум
* Электронный ресурс «Wemos D1 R2 и mini на основе esp8266» <https://arduinomaster.ru/datchiki-arduino/esp8266-wemos-d1-mini-raspinovka/> - Дата доступа 08.04.2022
* Электронный ресурс «**WeMos D1 R2 на ESP8266 обзор и примеры**» <https://geekelectronics.org/arduino/wemos-d1-r2-na-esp8266.html> - Дата доступа 08.04.2022
* Электронный ресурс «Книга – ESP 8266 01 – для начинающих Arduino / ArduBlock» <https://www.youtube.com/watch?v=ZxmOS7jRRg4> – Дата доступа 10.04.2022
* Электронный ресурс «Начало работы с Arduino, установка и разбор ошибок» <https://www.youtube.com/watch?v=G7bZJhWD01c> – Дата доступа 01.04.20224
* Книга «Интернет вещей с ESP8266» Марко Шварц
* Книга «Создание умного дома на базе ARDUINO»
* Книга «Internet of Things Architecture and Research App with MQTT Protocol»
* Книга «MQTT: A Concise and Practical Tutorial»
* Электронный ресурс «Протокол MQTT: концептуальное погружение» <https://habr.com/ru/post/463669/> - Дата доступа 20.04.2022
* Электронный ресурс «HiveMQ getting started» <https://www.hivemq.com/docs/hivemq/4.8/user-guide/getting-started.html> - Дата доступа - 15.04.2022

**Приложение А. Исходный код программы**

**WebLamp\_v1.0.ino**

#define LED\_PIN D1 // пин ленты

#define BTN\_PIN D2 // пин кнопки

#define PIR\_PIN D5 // пин PIR (ИК датчика)

#define LED\_AMOUNT 5 // кол-вл светодиодов

#define BTN\_LEVEL 1 // 1 - кнопка подключает VCC, 0 - подключает GND

#define USE\_PIR 0 // 1 - использовать PIR (ИК датчик) на этой лампе

#define IGNORE\_PIR 1 // 1 - игнорировать сигнал PIR (ИК датчика) с удалённой лампы

/\*

Запуск:

Клик или >15 секунд при анимации подклчения: запустить точку доступа

Кнопка сохранить или клик: перезагрузить систему

Анимация:

- Мигает зелёным: подключение к роутеру

- Мигает синим: запущена точка доступа WLamp <IP>

Работа:

1 клик: вкл/выкл

2 клика: сменить цвет

3 клика: подмигнуть

Удержание: сменить яркость

\*/

// ============= ВСЯКОЕ =============

#define MQTT\_HEADER "GWL:" // заголовок пакета данных

// ============= БИБЛЫ =============

#include <GyverPortal.h>

#include <EEPROM.h>

#include <ESP8266WiFi.h>

#include <PubSubClient.h>

#include <EEManager.h>

#include <FastLED.h>

#define EB\_STEP 100 // период step шага кнопки

#include <EncButton.h>

#include "Timer.h"

// ============= ДАННЫЕ =============

#if 1

#define DEBUG(x) Serial.print(x)

#define DEBUGLN(x) Serial.println(x)

#else

#define DEBUG(x)

#define DEBUGLN(x)

#endif

struct LampData {

char ssid[32] = "";

char pass[32] = "";

char local[20] = "Asdfg325";

char remote[20] = "Asdfg125";

char host[32] = "broker.mqttdashboard.com";

uint16\_t port = 1883;

uint8\_t ip[4] = {0, 0, 0, 0};

bool power = 1;

uint8\_t bright = 50;

uint8\_t color = 0;

};

LampData data;

EncButton<EB\_TICK, BTN\_PIN> btn;

CRGB leds[LED\_AMOUNT];

WiFiClient espClient;

PubSubClient mqtt(espClient);

GyverPortal portal;

EEManager memory(data);

bool pirFlag = 0;

bool winkFlag = 0;

bool startFlag = 0;

const uint8\_t hLen = strlen(MQTT\_HEADER);

Timer onlineTmr(18000, false); // 18 секунд таймаут онлайна

Timer pirTmr(60000, false); // 1 минута таймаут пира

Timer hbTmr(8000); // 8 секунд период отправки пакета

void setup() {

startup(); // запускаем всё

}

void loop() {

if (USE\_PIR && digitalRead(PIR\_PIN)) pirFlag = 1; // опрос ИК датчика

heartbeat(); // отправляем пакет что мы онлайн

memory.tick(); // проверяем обновление настроек

animation(); // эффект ленты

buttonTick(); // действия кнопки

mqttTick(); // проверяем входящие

portal.tick(); // пинаем портал

checkPortal(); // проверяем действия

}

**func.ino**

// подмигнуть

void wink() {

if (data.power) {

brightLoop(data.bright, 0, 20);

brightLoop(0, 255, 20);

brightLoop(255, 0, 20);

brightLoop(0, 255, 20);

brightLoop(255, 0, 20);

brightLoop(0, data.bright, 20);

}

}

void brightLoop(int from, int to, int step) {

int val = from;

for (;;) {

FastLED.setBrightness(val);

FastLED.show();

delay(10);

if (from > to) {

val -= step;

if (val < to) return;

} else {

val += step;

if (val > to) return;

}

}

}

// выводим эффект на ленту

void animation() {

static Timer tmr(30);

static bool breath; // здесь отвечает за погашение яркости для дыхания

static uint8\_t count; // счётчик-пропуск периодов

if (tmr.period()) {

// переключаем локальную яркость для "дыхания"

count++;

if (!onlineTmr.elapsed()) { // удалённая лампа онлайн

if (!pirTmr.elapsed()) { // сработал ИК на удалённой

if (count % 10 == 0) breath = !breath;

} else {

if (count % 30 == 0) breath = !breath;

}

} else {

breath = 1;

}

uint8\_t curBr = data.power ? (breath ? 255 : 210) : 0;

// здесь делаем плавные переходы между цветами

CRGB ncol = CHSV(data.color, 255, curBr);

CRGB ccol = leds[0];

if (ccol != ncol) ccol = blend(ccol, ncol, 17);

// выводим на ленту

fill\_solid(leds, LED\_AMOUNT, ccol);

FastLED.setBrightness(data.bright);

FastLED.show();

}

}

// локальный запуск портала. При любом исходе заканчивается ресетом платы

void localPortal(IPAddress ip) {

// создаём точку с именем WLamp и предыдущим успешным IP

Serial.println(F("Create AP"));

WiFi.mode(WIFI\_AP);

String s(F("WLamp "));

s += ip.toString();

WiFi.softAP(s);

portal.start(WIFI\_AP); // запускаем портал

while (portal.tick()) { // портал работает

loadAnimation(CRGB::Blue); // мигаем синим

btn.tick();

// если нажали сохранить настройки или кликнули по кнопке

// перезагружаем ESP

if (checkPortal() || btn.click()) ESP.reset();

}

}

// анимация работы локал портала

void loadAnimation(CRGB color) {

static int8\_t dir = 1;

static uint8\_t val = 0;

static Timer tmr(20);

if (tmr.period()) {

val += dir;

if (val >= 100 || val <= 0) dir = -dir;

fill\_solid(leds, LED\_AMOUNT, color);

FastLED.setBrightness(val);

FastLED.show();

}

yield();

}

// парсер инта из указанного индекса в строке

int getFromIndex(char\* str, int idx, char div = ',') {

int val = 0;

uint16\_t i = 0;

int count = 0;

bool sign = 0;

while (str[i]) {

if (idx == count) {

if (str[i] == div) break;

if (str[i] == '-') sign = -1;

else {

val \*= 10L;

val += str[i] - '0';

}

} else if (str[i] == div) count++;

i++;

}

return sign ? -val : val;

}

**mqtt.ino**

/\*

Протокол:

GWL:0,ir // пакет heartbeat, состояние ИК

GWL:1,power,color,wink // пакет данных

GWL:2 // запрос настроек

\*/

// опрашиваем mqtt

void mqttTick() {

if (WiFi.status() != WL\_CONNECTED) return; // wifi не подключен

if (!mqtt.connected()) connectMQTT();

else {

if (!startFlag) {

startFlag = 1;

char str[] = MQTT\_HEADER "2"; // +2

mqtt.publish(data.remote, str);

}

}

mqtt.loop();

}

void connectMQTT() {

// задаём случайный ID

String id("WebLamp-");

id += String(random(0xffffff), HEX);

//DEBUGLN(id);

// подписываемся на своё имя

if (mqtt.connect(id.c\_str())) mqtt.subscribe(data.local);

delay(1000);

}

// тут нам прилетел пакет от удалённой лампы

void callback(char\* topic, byte\* payload, uint16\_t len) {

payload[len] = '\0'; // закрываем строку

char\* str = (char\*)payload; // для удобства

DEBUGLN(str);

// не наш пакет, выходим

if (strncmp(str, MQTT\_HEADER, hLen)) return;

str += hLen; // смещаемся для удобства чтения

switch (getFromIndex(str, 0)) {

case 0: // heartbeat

if (!IGNORE\_PIR && getFromIndex(str, 1)) pirTmr.restart();

break;

case 1: // управление

data.power = getFromIndex(str, 1);

data.color = getFromIndex(str, 2);

if (getFromIndex(str, 3)) wink();

break;

case 2: // запрос

sendPacket();

break;

}

onlineTmr.restart(); // перезапуск таймера онлайна

}

// отправляем пакет

void sendPacket() {

// GWL:1,power,color,wink

String s;

s.reserve(10);

s += MQTT\_HEADER "1,"; // +1,

s += data.power;

s += ',';

s += data.color;

s += ',';

s += winkFlag;

winkFlag = 0;

// отправляем

mqtt.publish(data.remote, s.c\_str());

}

void heartbeat() {

if (hbTmr.period()) {

// GWL:0,pir

char str[hLen + 4] = MQTT\_HEADER "0,"; // +0,

str[hLen + 2] = pirFlag + '0';

pirFlag = 0;

mqtt.publish(data.remote, str);

}

}