МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование)

02.03.03 Математическое обеспечение и администрирование информационных систем
(код и наименование направления подготовки, специальности)

Мобильные и сетевые технологии
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему Разработка имитационной модели системы управлением склада на базе Mesh сети

Студент В.И. Рыбин
(И.О. Фамилия) (личная подпись)

Руководитель С.В. Митин
(ученая степень, звание, И.О. Фамилия)

Консультант М.В. Дайнеко

Тольятти 2021

Аннотация

Тема выпускной квалификационной работы: «Разработка имитационной модели системы управления освещением склада на базе MESH сети».

Целью данной бакалаврской работы является разработка имитационной системы управления освещением на базе Mesh-сетей, которая должна экономить электроэнергию, автоматизировать процесс жизнедеятельности компании.

Для достижения поставленной цели решались следующие задачи изучение особенности работы Mesh-сетей, разработка алгоритма по нахождению кратчайшего пути, разработка программного обеспечения для внедрения Mesh-сети в процесс работы освещением, тестирование работы Mesh-сети и алгоритма нахождения кратчайшего пути.

В данной бакалаврской работе исследуются построение Mesh-сети на базе контроллеров esp32 с применением алгоритма кратчайшего пути.

Структура бакалаврской работы представлена введением, три главы, заключением, приложением, списком литературы.

Во введение описывается актуальность дистанционного управления освещением, даётся краткая характеристика работы.

В первой главе проводится обзор известных решений по организации построения Mesh-сети.

Во второй главе проводится построение математической модели построение кратчайшего пути с применением алгоритма Э. Дейкстры.

В третьей главе проводиться разработка программного кода на стороне сервера (рабочая станция), клиент (микроконтроллер esp32).

В заключение представлены выводы по проделанной работе.

В работе использовано 4 таблиц, 24 рисунка, список литературы содержит 20 литературных источников. Общий объём выпускной квалификационной работы составляет 53 страниц

Abstract

The theme of bachelor's work: "Development of a software model for a warehouse lighting control system based on the MESH network."

In this bachelor's work, we study the construction of a Mesh network in a warehouse and the development of an algorithm for controlled lighting depending on a person’s route

The first chapter provides an overview of known solutions for organizing the construction of a Mesh network.

In the introduction, the relevance of remote lighting control is described, a brief description of the work is given.

In the first chapter, a review of known solutions for organizing the construction of a Mesh network is carried out.

In the second chapter, the construction of a mathematical model is carried out, the construction of the shortest path using E. Dijkstra's algorithm.

The third chapter is the software implementation and testing of the prediction of the current route.

In conclusion, conclusions on the work done are presented.

In the work 4 tables, 24 figures were used, the list of references contains 20 literary sources. The total volume of final qualification work is 53 pages.

Содержание

Введение

Информационные технологии занимают ключевые позиции в области электроники и развиваются быстрыми темпами. Появляется возможность создания сложных беспроводных сетей, умных бытовых приборов, других электронных устройств, которые способны взаимодействовать друг с другом. Благодаря этим сетям реализуется автоматизация производств, например: при охране и защите различных объектов, системы освещения, транспортировка товаров. Грамотная автоматизация позволяет снизить затраты электроэнергии в сетях, влияния человеческого фактора.

Используя уникальные технологии передачи сигнала по сети, удаётся решиться проблему экономии электроэнергии, автоматизацию процессов, контроля и эксплуатацию технологии в режиме реального времени. Примером такой разработки, предназначенной для автоматизации процессов предприятия и экономии электроэнергии, является – освещение на базе Mesh-сетей. Что это даёт? При помощи дистанционного освещение на базе mesh-сети предоставляется возможность экономить электроэнергию на предприятии, при помощи алгоритмов достигается экономия электроэнергии т.к. алгоритмы будут сопоставлять потребности потребителя и включать только необходимые лампочки. Внедрение такой технологии позволит повысить эффективность предприятия и значительно уменьшить потребление электроэнергии предприятия.

В сфере производства электроэнергии применение IoT позволит сократить расход топлива, что в настоящее время составляет половины операционных расходов станций. На смену стандартной системы освещения придёт гибкая система освещения на базе Mesh сети. Каждый элемент сети сможет видеть другие элементы и взаимодействовать с ними, что позволит при нарушении целостности сети самовосстановиться и продолжить работу в штатном режиме. Данные изменения позволят выйти на новый уровень

надежности и эффективности энергетической системы. С каждым годом регулирующие органы требуют сокращение тарифов и обслуживающих их затрат сетевых компаний. Такие затраты можно сократить за счёт улучшения наблюдаемости объектов сети.

Новизна исследовательской работы заключается в экономии электроэнергии при помощи использования Mesh-сетей на базе алгоритма по нахождению кратчайшего пути.

Объект исследования – системы управления освещением на базе Mesh-сетей.

Предмет исследования – экономия энергии при использовании Mesh-сетей с использованием нахождения кратчайшего пути.

Целью данной бакалаврской работы является разработка имитационной системы управления освещением на базе Mesh-сетей, которая должна экономить электроэнергию, автоматизировать процесс жизнедеятельности компании.

Для достижения поставленной цели необходимо решить следующие задачи:

- изучить особенности работы Mesh-сетей;
- изучить существующие алгоритмы по нахождению кратчайшего пути;
- разработать программное обеспечение для внедрения Mesh-сети в процесс работы освещением;
- протестировать фактическую работу Mesh-сети и алгоритма нахождения кратчайшего пути.

1 Обзор известных решений по организации построения Mesh-сети

1.1 Актуальность Mesh-сетей

Для интеграции Mesh-сети на устройства используют контроллеры и микроконтроллеры. WiFi несомненный лидер беспроводных соединений устройств. Технология беспроводного соединения устрой WiFi очень похожа на Mesh-сеть, но Mesh-сети часто используют для организации городских или локальных сетей.

Mesh-сеть обрела быструю популярность среди предметов микроэлектроники. Появилось огромное количество устройств, которые работают автономно и имеют смену режимов online/ofline. Данным сетям необходим обмен информации со своим окружением.

1.2 Примеры технологий на базе mesh-сетей

Далее будут рассмотрены технологии mesh-сетей. Из основных можно выделить следующие:

- Z-Wave – это протокол, который разрабатывается с 2001 г для устройств в умном доме. Z-Wave используют для домашней автоматизации. Технология использует радиочастотные модули, которые встроены в бытовую технику, освещение, отопление. Через эти модули выполнятся, такие команды, как включение или выключение света, оповещение о протечке воды, автоматическое движение роль ставней. В основе технологии Z-Wave лежит ячеистая топология mesh, в данной технологии каждое устройство может являться приёмником или передатчиком информации. Главное преимущество – это совместимость с другими устройствами. Данный протокол подходит для объектов площадью от 10 до 5000м. При

помощи пульта пользователь контролирует бытовую технику, освещение, отопление. Пользоваться такой технологией можно через интернет. Преимуществом данной технологии является низкое электропотребление, которую разработали специально для дистанционного управления. Z-Wave имеет низкий радиочастотный диапазон (1 ГГц), что уменьшает потенциальные источники помех. Другие технологии Wi-Fi, ZigBee, Bluetooth работают в диапазоне 2.4 ГГц, в данном диапазоне много источников помех и необходимо прибегать к мероприятиям, которые уменьшают помехи от других бытовых устройств;

- Ruckus – данная технология была создана для коммерческих предприятий, она предназначена для умного дома. Данная сеть имеет такие плюсы как: самовосстановление, самоорганизация, самооптимизация. Данная сеть состоит из двух устройств. Первое устройство – это контроллер сети Ruckus mesh, а второе устройство Ruckus mesh – это точки доступа Root AP;

- AirTies Mesh – эта технология была создана для решения проблемы со слабым сигналом в локальной сети и ограниченного покрытия. Несколько точек доступа организуют mesh-сеть. При подключении к данной сети пользователю автоматически назначается точка доступа с наиболее сильным сигналом.

1.3 Технология беспроводной передачи данных Wi-Fi

Первый стандарт, благодаря которому были организованны беспроводные сети – это WiFi или RadioEthernet 802.11. Однако наиболее распространенным стандартом является IEEE 802.11 с буквенными индексами a, b, g, n. Эти стандарты являются самыми популярными у производителей оборудования. Первый стандарт IEEE 802.11 был разработан в 1997 году и предоставлял пользователям доступ к данным на скорости 1 и 2

Мбит/с в диапазоне частот 2,4 ГГц. В таблице 1 представлены характеристики стандартов IEEE 802.11.

Таблица 1– характеристики основных стандартов IEEE 802.11

	IEEE 802.11a	IEEE 802.11b	IEEE 802.11g	IEEE 802.11n
Частотный диапазон	5.15 – 5.25 5.67-5.85	2.4 -2.483	2.4-2.483	2.4-2.483 5.15-5.25 5.67-5.85
Доступ к радиоканалу	CSMA-CA	CSMA-CA	CSMA-CA	CSMA-CA
Количество абонентов на один канал	64	64	64	64
Максимальная скорость обмена данными	54 Мбит/с	11 Мбит/с	54 Мбит/с	600 Мбит/с
Обычная скорость передачи данных	23 Мбит/с	4 Мбит/с	20 Мбит/с	120 Мбит/с
Ширина канала	20 МГц	22 Мбит/с	20 Мбит/с	40 Мбит/с
Дальность действия в помещении	10-20	20-100	20-50	10-20

Основные преимущества передачи данных с помощью технологии WiFi заключается в:

- Простое использование;
- Работой множества устройств в диапазоне 2,4 ГГц;
- Безопасная передача информации (64/128-битное шифрование);
- Легкая интеграция с существующими проводными сетями;

Недостатки данной технологии заключается в:

- Высокая цена на оборудование;
- Большое энергопотребление;
- Недостаточно защищенный стандарт шифрования WEP, который используется для защиты доступа к сети.

У WiFi огромная пропускная способность, поддержка пропускной способности не проблема, но в нашем случае проблему вызывает большое энергопотребление сети. WiFi потребляет много энергии для технологий по типу «Умный дом» [2]. Конфигурация данной сети называю «Клиент/Сервер», данная организация позволяет подключать до 10 устройств. Данная сеть организуется при помощи точек доступа (Access Point). Далее на рисунке 1 представлена точка доступа.

Рисунок 1 – точка доступа

При организации проводной сети точка доступа может быть использована в качестве моста между сегментами сети и беспроводными устройствами.

Точка доступа может работать в составе проводной сети и служить в качестве моста между беспроводными и проводными устройствами. При использование данной конфигурации устройства общаются с точкой доступа, которая руководит передачей пакетов данных. При росте сети, где сеть не ограничивается одной точкой доступа, образуется расширенный набор служб (Extended Service Set, ESS). При использовании сети которая поддерживает ESS точки доступа связывается между собой посредству проводных соединений или радиомостов (рисунок 2).

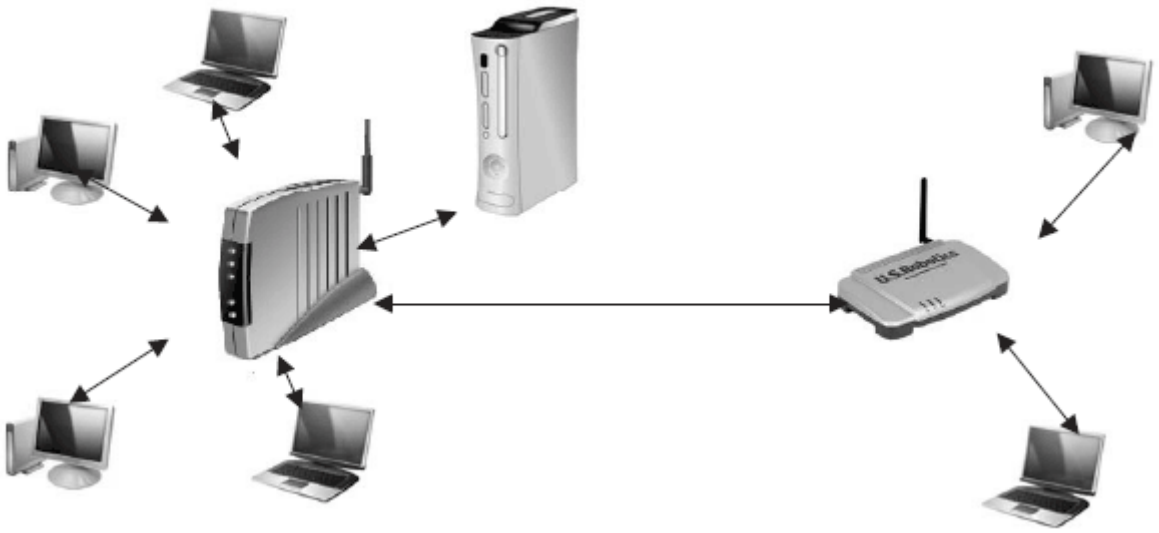


Рисунок 2 – Организация сети с поддержкой ESS

Так же недостатком данной технологии является зависимость от маршрутизатора так как если маршрутизатор сломается, то сеть перестанет работать так как точка доступа будет недоступна. Данная технология пользуется ячеистой топологией. В ячеистой топологии весь трафик

передаётся от одной точки доступа к другой при помощи беспроводного канала.

1.4 Технология беспроводной передачи данных Z-Wave

Z-Wave – протокол связи разработанный для управления умными вещами технологии «Умный дом». Для автоматизации управления и связи технологии Z-Wave с приборами используются малые частоты. Данная технология имеет низкое энергопотребление, её разрабатывали для дистанционного управления системами [12]. Z-Wave пользуется низким радиочастотным диапазоном, такое решение обусловлено малым количеством помех от других устройств.

Z-Wave позволяет создать недорогую и энергоёмкую систему. Она идеально подходит для устройств, которые работаю автономно.

Нижние слои протокола, MAC и PHY, описываются ITU-T G.9959, а также они являются обратно совместимыми. Продукты от компании Z-Wave всегда совместимы. Благодаря сертификациям Z-Wave или Z-Wave Plus происходит хорошая совместимость с другими устройствами сети. Каждое устройство Z-Wave может передавать и получать сигнал другим устройствам и выступает в роли ретранслятора. Поэтому если из одного узла сигнал не может быть отправлен напрямую в другой узел, то он будет отправлен в целевой узел через промежуточные находящиеся в радиусе действия сигнала.

1.5 Обзор стека протокола Z-Wave

В стеке Z-Wave присутствует физический и канальный уровень для доступа к среде передачи данных. Транспортный уровень отвечает за целостность данных, на этом уровне происходит проверка целостности данных, ретрансляция подтверждение. За маршрутизацию данных отвечает

сетевой уровень [12]. Стек Z-Wave работает с диапазоном от 860 МГц до 908 МГц. Данный протокол может иметь скорость до 40 Кб/с.

Транспортный уровень контролирует проверку контрольной суммы данных, после чего идёт подтверждение. Переда началом отправки данных в первую очередь передаётся пакет, в котором содержится информация об идентификаторе сети, размер данных и тип данных. Байт контрольной суммы передаётся в конце пакета данных, для проверки целостности пакета.

Сетевой уровень Z-Wave отвечает за передачу данных и проверку со списком отправителя. В нашем случае данный уровень будет отвечать за поддержку карты маршрутизации сети.

Уровень приложений Z-Wave отвечает за выполнение разных команд в сети Z-Wave и декодирования. Команды можно поделить на 2 класса:

- Команды протокола Z-Wave;
- Специальные команды приложений.

Каждый уровень анализирует информацию, поступающую с соседнего уровня, который расположен выше него единым блоком. Разные уровни протоколов не взаимодействуют между собой на прямую: их взаимодействие происходит через физический уровень.

Физический уровень – осуществляет передачу данных на частоте 869.0МГц (Россия), 919.8 (Гонконг), 865.2 (Индия), частоты уровня зависит от страны использования т.к. производство устройства производиться для разных рынков, разных стран. Разработчики данного стандарта приняли хорошее решение, которое в перспективе ожидает хорошие продажи по всему миру.

При использовании пакета информации Z-Wave можно определить возможности узла такие, как тип узла, способность узла повторять пакет и другие моменты, которые могут быть связанны с протоколом. Таким образом приложение может разрешить узлу сети получать информацию о других узлах.

Менеджмент узлов Z-Wave включает две основные операции:

- Ассоциирование;
- Включение/исключение.

Включение запускает новые сетевые узлы, а исключение описывает процесс удаление сетевых узлов. Включать и исключать узлы сети могут только первичные контроллеры. Контроллеры по средству сигналов имею возможность связываться с тем узлами сети, с которыми это необходимо. Они имеет свой персональный идентификатор, созданный на заводе изготовителя. Основной контроллер подчиняет другие узлы сети, вписывая свой собственный ID, когда узел принимает ID основного контроллера, то становиться частью сети. Вместе с присвоением ID основного контроллера, новому устройству присваивается индивидуальный идентификатор узла. Данный процесс называется – Включением. Представим, что у нас имеется 4 устройства с заводскими настройками по умолчанию. Два контроллера с заданным Home ID. Два других устройства, которые не имеют собственного Home ID т.к. устройства не могут работать в качестве контроллера (рисунок 3).

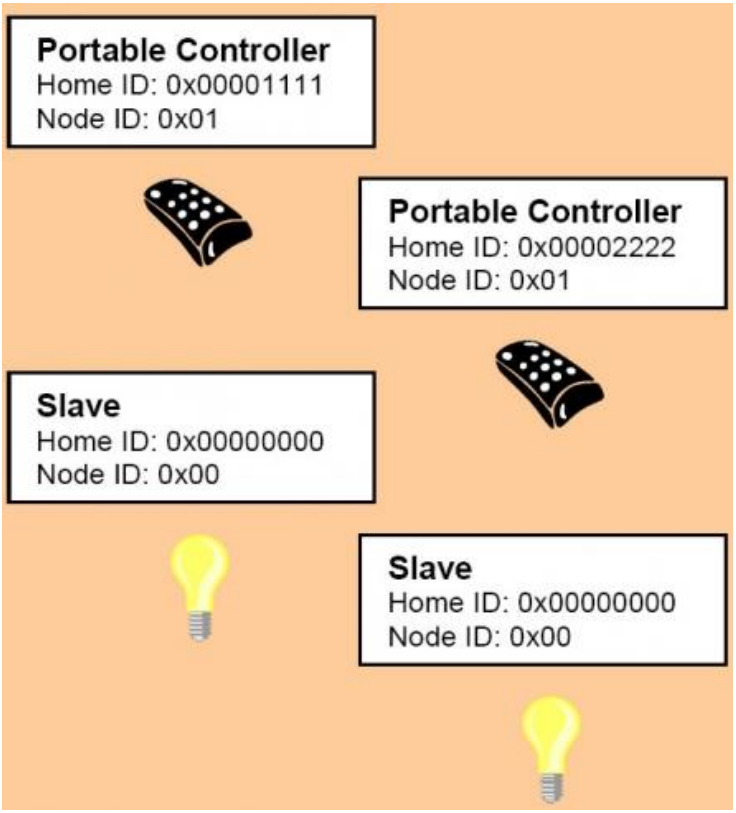


Рисунок 3 – Z-Wave устройства до включения в сеть

Главный идентификатор сети в данном примере будет либо 0x00001111, либо 0x0000222. У двух контроллеров имеется одинаковый номер узла 0x01. Так как ни один из узлов не имеет общий Home ID, то связь между узлами не может быть. Пусть один из контроллер будет выбран, как главный контроллер сети, то этот контроллер присваивает Home ID оставшимся устройствам, тем самым включает их (присваивает уникальный Node ID).

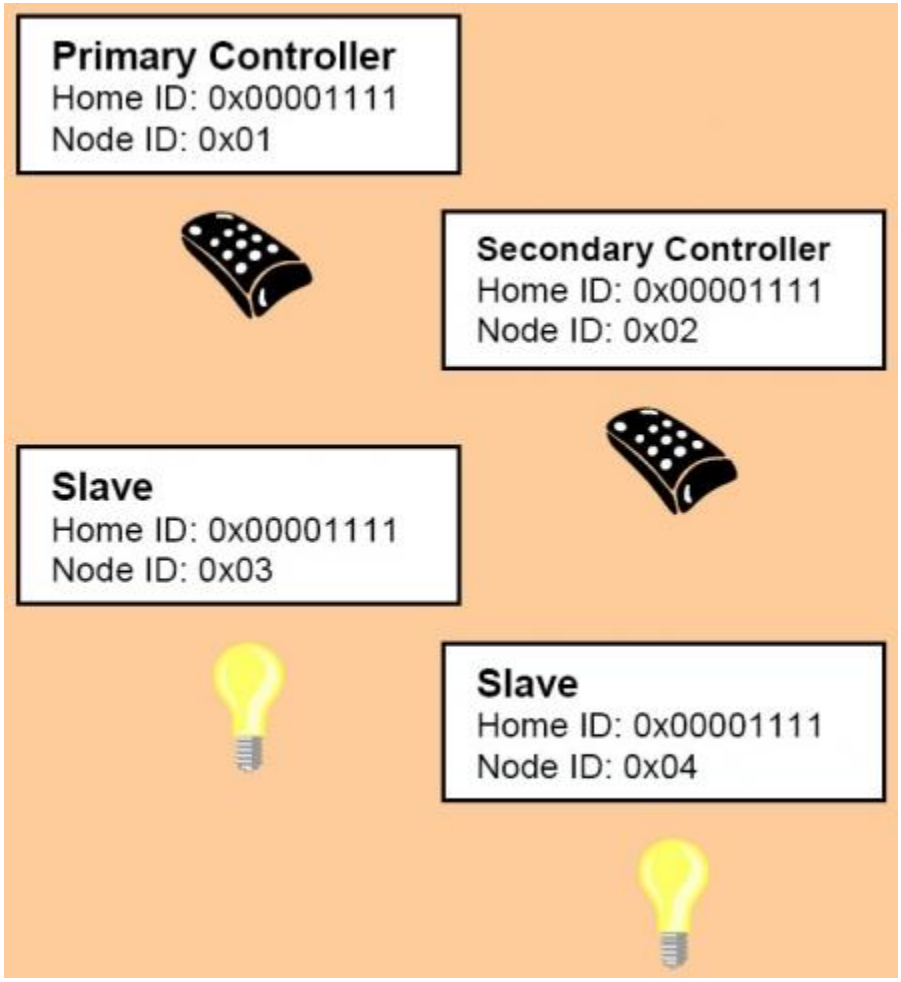


Рисунок 4 – Z-Wave устройства после включения в сеть

При запуске такого процесса начинается активация инициатора в контроллере и его устройстве одновременно. Зачастую инициатором является кнопка.

При получение данных от неинициализированного узла контроллер предоставит Home ID и Node ID контроллеру. При подключение нового контроллера к сети, ему будет предоставляться последняя информация по маршрутизации. Обновление сетевой информации может осуществляться автоматически при использовании функциональности обновления статических контроллеров (SUC).

Ассоциирование происходит при одновременной активации на контроллере и подчиненном ему устройстве.

1.6 Настройка беспроводной сети Z-Wave

Использование ячеистой технологии в протоколах Z-Wave позволило создать сеть из одного управляемого устройства и управляющего устройства. При необходимости дополнительные устройства можно подключить в любой момент времени.

Для управления устройством необходимо убедиться, что устройства подключены к сети Z-Wave. Контроллер запоминает мощность сигнала, который исходит от устройства так, что устройства необходимо устанавливать в окончательном месте. Z-Wave предлагает автоматическую конфигурацию/реконфигурацию сети [13]. Z-Wave так же предлагает конфигурацию сети реконфигурацию сети, запуск такой процедуры позволяет перераспределить маршруты и улучшить связь.

1.7 Топология и маршрутизация Z-Wave

В Z-Wave применяется ячеистая топология сети, за маршрутизацию отвечает источник, также в данной технологии присутствуют различные контроллеры, один первичный и ноль или более вторичных. Они отвечают за маршрутизацию и безопасность. На рисунке 5 представленном ниже изображена схема технологии Z-Wave

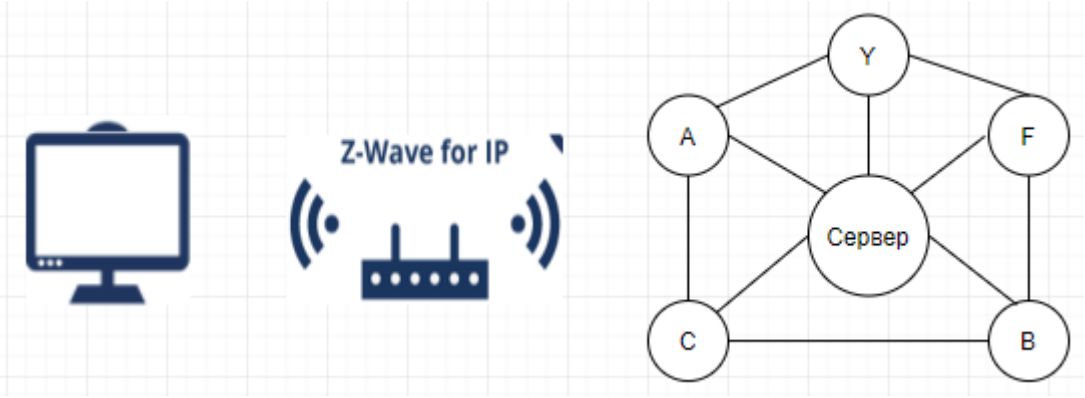


Рисунок 5 – Схема технологии Z-Wave

В данной технологии ключевую роль играет сервер, на нём описана логика по которой происходит отправка сообщения для более эффективной работы сети. Далее будет рассмотрен пример отправки сообщения. При отправке сигнала из узла A в узел B он будет успешно доставлен, не смотря на то что данные узлы могут находиться вне зоны действия связи, это возможно благодаря узлу D, который взаимодействует с данными узлами и является сервером приложения. Если из узла D невозможно напрямую отправить сообщение в узел B, то оно перенаправляется в промежуточный узел C т.к. каждое устройство выступает в роли ретранслятора и передаёт сигнал по узлам и тем самым увеличивая зону покрытия сигнала. Взаимодействия между узлами происходит при участии центрального контроллера (первичного), который хранит сведения о маршрутах, через которые передаются данные и так же запоминает расположение новых устройств. Центральный контроллер обеспечивает маршрутизацию внутри сети, такой контроллер может быть только один, но стандарт допускает использование нескольких вторичных контроллеров.

Благодаря этому Z-Wave сеть имеет большой радиус передачи сигнала. Однако из-за таких прыжков происходят небольшие задержки сети. Устройства, работающие от аккумуляторов, периодически находятся в

режиме сна, поэтому в этот момент они не могут ретранслировать сигнал. В связи с этим данные устройства не подходят для ретрансляции сигналов. Z-Wave поддерживает до 232 устройств, которые имеют возможность расширения сети.

1.8 Безопасность технологии Z-Wave

До 2008 года компания Z-Wave не опубликовывала документацию или код в открытый доступ, но в 2008 году разработчики впервые выложили в открытый доступ информацию по шифрованию. В качестве шифрования протоколов Z-Wave был выбран алгоритм AES-128. На тот момент разработчики значительно увеличили защищённость протокола т.к. предыдущий протокол шифрования Z-Wave имел лазейку. Первичный ключи шифрования состоял из 128 нулей. Злоумышленник мог спокойно взломать данную систему управлением дома. В 2016 году компания доработала специфику шифрования и выпустила новую версию Z-Wave S2 Security, где в качестве первичного ключа использовался алгоритм распределения ключей Диффи-Хелмана.

1.9 Технология беспроводной передачи данных ZigBee

ZigBee около 10 лет является главным конкурентов Z-Wave на рынке умного дома. Данный протокол разработали в 2004 году. В данном протоколе присутствует низкое энергопотребление, управление умным домом, оптимизация для удаленного мониторинга. Главным различием между этими протоколами можно посмотреть в модели OSI.

ZigBee включает в себя высокоуровневые протоколы связи, которые используют маломощные трансиверы. ZigBee основан на стандарте IEEE

802.15.4-2006 предназначенном для организации беспроводных персональных сетей, таких как беспроводные наушники. Создатели ZigBee ставили перед собой задачу разработать технологию, которая будет работать эффективнее технологии Bluetooth, для устройств, которые работают в автономном режиме.

Протоколы ZigBee были разработаны для использования в приложениях, которые требуют низкую скорость передачи данных и низкое энергопотребление [4]. Основной задачей ZigBee является создание дешевой, самоорганизующейся ячеистой сети. Созданная ZigBee сеть потребляет мало электроэнергии, благодаря этому устройства могут работать около двух лет.

Устройства ZigBee можно разделить на следующие типы:

- Координатор ZigBee (ZC) – данное устройство создаёт таблицу маршрутизации (дерево сети), которая необходима для связи с другими сетями. Для создания сети ZigBee, необходимо наличие хотя бы одного координатора;
- Маршрутизатор ZigBee (ZR) – промежуточный маршрутизатор сети;
- Конечное устройство ZigBee (ZED) – данное устройство предназначено только для обмена данными с родительским узлом, с другими узлами обмен данными не происходит.

1.10 Протоколы ZigBee

В построение протокола ZigBee участвует два протокола AODV и NeuRFon. AODV – это протокол для динамической маршрутизации мобильных сетей, а NeuRFon – это протокол, который предназначен для образования мобильных ad-hoc сетей или узлов. Большинство профилей протокола ZigBee поддерживают сеть с отключенными или включенными маячками.

Такие сети поддерживают включенными свои приёмники большой промежуток времени, что плохо сказывается на энергосберегающих способностях. Такие сети называются разнородными. Разнородные сети – это сети, где одно устройство длительный промежуток времени отправляет внешние сигналы, пока другие устройства их принимают. В качестве примера такой сети можно предоставить ламповый выключатель [5]. В сетях такого вида узел, связанный с лампой должен является либо маршрутизатором ZigBee, либо координатором.

В сетях с маяком, маршрутизаторы ZigBee периодически отправляют сигналы на другие узлы для обмена информацией с ними. Узлы могут находится в режиме сна, так как данный режим повышает время их автономной работы. В этот момент они не могут ответить на сигнал. Поэтому маршрутизатор должен точно распределять время между маяками, в связи с чем его стоимость возрастает.

В сетях без маячка отсутствует симметричное распределение энергии, в связи с чем одни устройства могут быть постоянно в рабочем режиме, а другие постоянно в режиме сна. Протоколы ZigBee позволяют оптимизировать энергопотребление, а также время включения радиопередатчиков.

Как указано в открытых источниках, а также в техническом руководстве: «все устройства ZigBee выпускаются совместимые со стандартом IEEE 802.15.4-2003 сетей, такой стандарт определён тем, что нижние слои протокола – физический слой (PHY) и контроль доступа (MAC) являются частью слоя данных (DLL), данный стандарт определяет работу на частотах 868 МГц (Европа), 2.4 ГГц в мире на диапазоне ISM, ZigBee использует широкополосную модуляцию с прямым расширением спектра, которая управляет потоками в модуляторе, офсетная квадратурная манипуляция используется на полосе 2,4 ГГц, а двоичная фазовая манипуляция на полосах в 868 и 915 Мгц, когда идёт передача информации

скорость составляет 250 кбит/с для каждого канала в диапазоне 2.4 ГГц, 20 кбит/с в диапазоне 868 Мгц» [7].

1.11 Программное и аппаратное обеспечение технологии ZigBee

Компания ZigBee занимается разработкой и внедрением своих протоколов для бюджетных микропроцессоров. Протоколы хорошо оптимизированы и удачно балансируют между ценой и качеством.

Для интеграции протоколов ZigBee с радиопередатчиками, необходимо чтобы последние прошли полную проверку требований на физическом уровне, таким образом все устройства будут обладать одинаковыми FR-характеристиками. Данное требование необходимо поскольку протоколы ZigBee накладывают жесткие ограничения на оборудование и в случае не соответствия им, аккумуляторная батарея устройства в скором времени может выйти из строя.

При некоммерческом использовании спецификация ZigBee доступна любому желающему. Однако при коммерческом использовании необходимо заключать договор с компанией ZigBee, после чего пользователю предоставляется доступ к спецификации и её разработке.

Сети, предоставляемые ZigBee на территории Российской Федерации, имеют возможность работать в диапазоне 2400-2483,5 МГц и для них не требуется получения разрешения на использование.

1.12 Выбор протокола передачи данных

На сегодняшний день лидер по совместимости устройств разных производителей – это технологии Z-Wave ZigBee. минусы протоколов, было

принято решение воспользоваться протоколом Z-Wave [2]. В таблице 2 предоставлены преимущества и недостатки протоколов Z-Wave и ZigBee.

Таблица 2 – Сравнения протоколов передачи данных

Протокол связи	Скорость передачи данных	Частота связи	Надёжность	Совместимость с другими устройствами	Диапазон работы устройств
ZigBee	250 кбит/с	2.4 ГГц	Сильные помехи от Wifi, Bluetooth, микроволновых печей	Плохая совместимость между устройствами различных производителей	От 10 до 20 метров
Z-Wave	42 кбит/с	800-900 МГц	Минимальные помехи	да	Максимальная досягаемость до 100 метров

Исходя из выше изложенного можно сделать вывод, что протокол Z-Wave за счёт реализованного уровня протоколов S2 более безопасный нежели ZigBee с протоколом AODV. Использование в протоколах ZigBee стандартных ключей подвергает систему серьёзной опасности и поэтому его использование является нежелательным.

Низкая частота сигнала Z-Wave позволяет ему быть менее чувствительным к помехам в отличии от протокола ZigBee. Устройства Z-Wave отлично работают в городе, где есть много сигналов.

Можно сделать вывод, что технология Z-Wave обгоняет технология ZigBee в планах безопасности и защищенности от помех.

В данном разделе были приведены характеристики популярных протоколов, так же сравнение протоколов Z-Wave и ZigBee. Выбор подходящего протокола для технологии умного освещения.

2 Математическая модель определения кратчайшего пути

2.1 Алгоритм Э. Дейкстры

В последнее время развитие предприятия включает в себя автоматизацию рабочего процесса и благодаря этому на предприятии появляются автоматические устройства, которые выполняют часть функций человека.

Роботы оказались слишком эффективными, так как они доставляли товары так быстро, что нормативы для людей увеличились более чем вдвое и продолжали расти. Если раньше погрузчики должны были обработать 100 товаров за час, с появлением роботов их норма выросла до 400 товаров в час. Чтобы обеспечить наибольшую эффективность необходимо решить вопрос о том, как находить кратчайший путь от точки А до точки В, было принято решение использовать алгоритм Э. Дейкстры. Данный алгоритм используется для нахождения кратчайшего пути.

Рассмотрит алгоритм Э. Дейкстры. Для реализации хранения чисел необходимо использовать массив чисел, а для хранения принадлежности множеству – массив булевых переменных. В начале алгоритмы расстояние для вершины графа равна нулю, все остальные расстоянию заполняются положительными числами. Массив флагов заполняется нулями. После идёт запуск основного цикла.

На каждом шаге цикла нам необходимо найти такую вершину V расстояние, которой будет минимальным и флаг которого будет равен нулю. После этого действия необходимо установить флаг на этой вершине и проверить все соединенные соединённые с ней листья. Если в данной вершине расстояние больше, чем сумма расстояний до текущей вершины, то уменьшаем это расстояние. Цикл прекращает свою работу после того, как флаги всех вершин становятся равным единице, или, когда у всех вершин стоит флаг:

$$d[i] = \infty \quad (1)$$

Этот случай возможен только тогда, когда граф G является несвязный.

Пусть l(V) – это длинна кратчайшего пути из вершины A в вершину V.

Благодаря индукции можно доказать посещение любой вершины:

$$d(v) = l(v) \quad (2)$$

где, $d(v)$ – окончание работы алгоритма равно по длине кратчайшего пути из вершины A в вершину V;

$l(v)$ – фактическая длинна кратчайшего пути из вершины A в вершину V.

Начальное условия посещается вершина A. В этот момент времени оно будет сопоставимо с равенством:

$$d(a) = l(a) = 0 \quad (3)$$

Допустим, что мы выбрали для посещения вершину Z≠A. Докажем, что в этот момент времени:

$$d(z) = l(z)$$

Необходимо отметить, что для любой вершины V всегда выполняется неравенство: 4)

$$d(v) \geq l(v) \quad 5)$$

Пусть P – это самый короткий путь из точки A в точку Z. Вершина Z находиться на пути P и не является на данный момент достигнутой. Поэтому множество вершин, которые мы не достигли на пути P

(6)

не является пустым. Предположим, что Y – это первая вершина, которую мы не достигли на пути P, X – вершина, которая предшествует вершине Y. Путь P является кратчайшим, следовательно, его часть, ведущая из A через X в Y, является кратчайшей:

$$l(y) = l(x) + w(xy)$$

По предположению индукции во время посещения вершины X выполнялось равенство:

$$d(x) = l(x) \quad 7)$$

Исходя из этого вершина Y получила метку не больше чем:

$$d(x) + w(xy) = l(x) + w(xy) = l(y) \quad 8)$$

От сюда следует, что $d(y) = l(y)$. Если Z = Y, то индукционный переход доказан. Поскольку сейчас выбрана вершина Z, а не Y, метка Z становиться минимальной среди достигнутых, то есть выполняется равенство:

$$d(z) \leq d(y) = l(y) \leq l(z) \quad 9)$$

Тогда при комбинации формулы (9) с формулой:

$$cd(z) \geq l(z) \quad (10)$$

В результате получаем формулу (4), что и требовалось доказать. Алгоритм заканчивает работу, когда все вершины достигнуты и в этот момент является истинным равенство $d = l$, которое выполняется для всех вершин.

Сложность данного алгоритмы зависит от способа нахождения вершин V, а также способа обновления и хранения меток. Обозначим через n – количество вершин, а через m – количество рёбер в графе G.

В самом простом случае, когда для поиска вершин просматривается всё множество вершин, а для хранения d используется массив, то при такой архитектуре время работы данного алгоритма можно вычислить с помощью формулы

(11)

$O(n^2)$.

Основной цикл будет выполняться n раз, в каждом из циклов нахождение минимума займёт n операций. На цикл посещения соседних вершин тратиться количество операций, которое пропорционально количеству ребер m. Можно сделать вывод, что общее время работы алгоритма соответствует $O(n^2 + m)$, но так как m≤n(n-1) то, оно равняется $O(n^2)$.

Рассмотрим применение алгоритма Э. Дейкстры на практике. Возьмём ориентированный граф G изображен на рисунке 6.

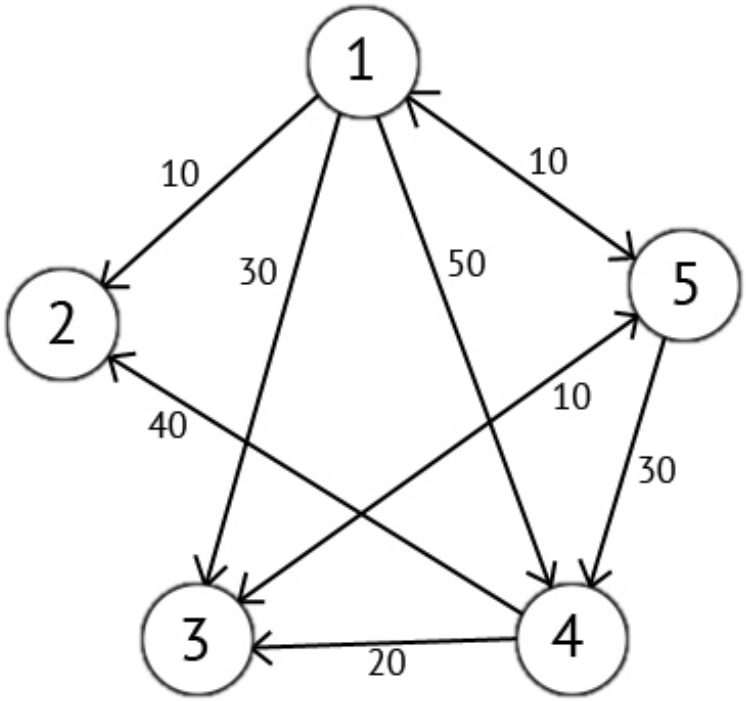


Рисунок 6 – Ориентированный граф G

Возьмём в качестве начальной точки вершину 1. Мы будем искать кратчайшие маршруты из вершины 1 в вершину 2, 3, 4, 5. Данный алгоритм пошагово перебирает все вершины графа и назначает им метки, которые являются известным минимальным расстоянием от вершины источника до конкретной вершины.

Присвоим вершине 1 метку равную 0, так как эта вершина — источник. Остальным вершинам назначим метки равные бесконечности (рис. 7).

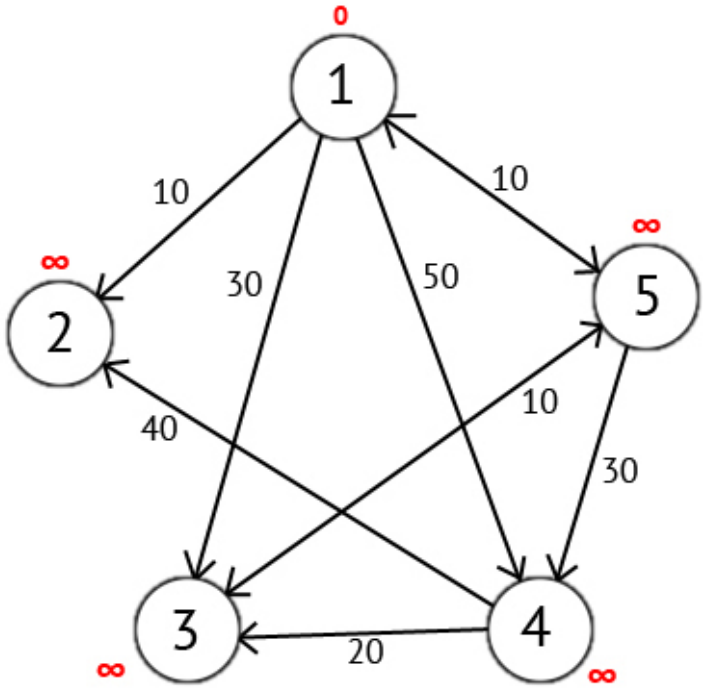


Рисунок 7 – Граф с назначенными метками

Следом выбираем такую вершину W, которая имеет минимальную метку (в данном случае это вершина 1) и рассмотрим, все остальные вершины в которые из вершины W есть путь. Каждой из рассмотренных вершин назначим метку равную сумме метки W и длину пути из W, но только случае, если полученная сумма будет меньше предыдущего значения метки. Если же сумма не будет меньше, то оставляем метку без изменений (рис. 8).

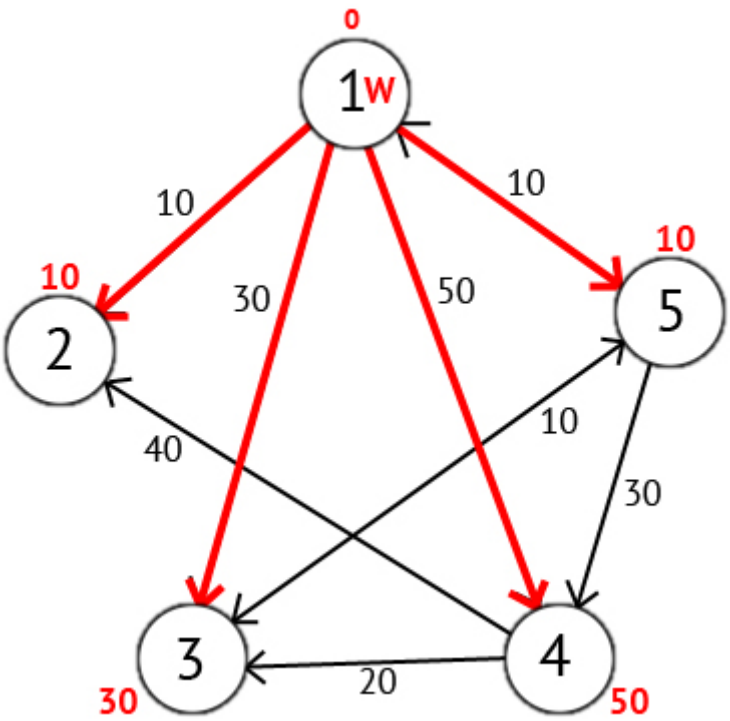


Рисунок 8 – Граф с назначенными суммами меток длинны

После того как мы рассмотрели все вершины, в которые есть прямой путь из W, то вершину W мы отмечаем как достигнутую, и выбираем ту которая имеет минимальное значение метки, она и будет следующей вершиной W. В данном случае это вершина 2 или 5. Если есть несколько вершин с одинаковыми метками, то выбираем любую из них.

Выберем вершину 2. Из нее нет ни одного исходящего пути и сразу отмечаем данную вершину, как достигнутую и переходим к следующей вершине с минимальной меткой. Выбираем вершину 5 так как она имеет минимальную метку (рисунок 9).

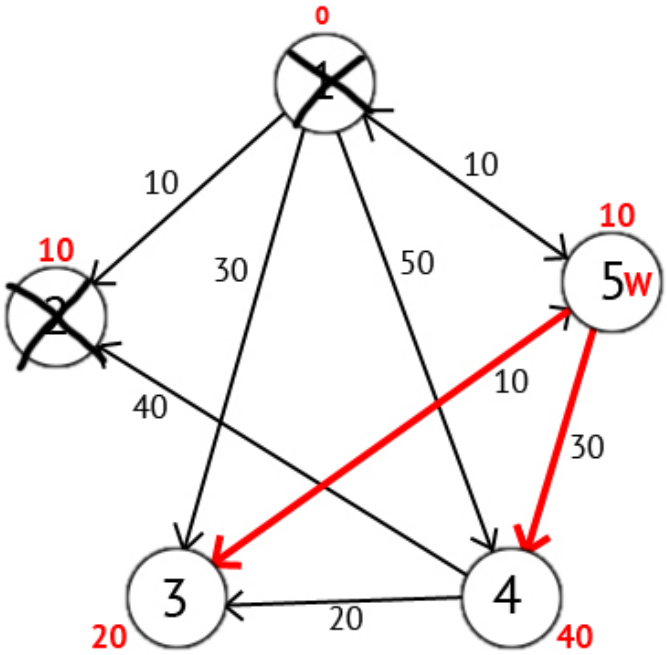


Рисунок 9 – Граф с достигнутой вершиной 5

Исходя из рисунка видно, что метки 3-ей и 4-ой вершины стали меньше, следовательно, был найден более короткий маршрут в эти вершины из вершины источника. Далее отмечаем 5-ю вершину как достигнутую и выбираем следующую вершину, которая имеет минимальную метку. Повторяем все перечисленные выше действия до тех пор, пока есть недостигнутые вершины. Выполнив данное действие получим результат, который изображен на рисунке 10.

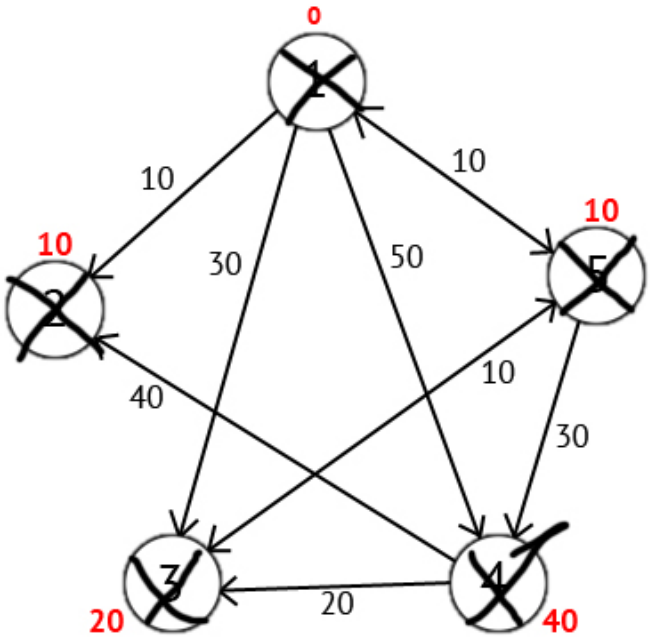


Рисунок 10 – Граф с конечными метками

Имеется вектор P, исходя из которого можно построить кратчайшие маршруты. Данный вектор по количеству элементов соответствует количеству вершин в графе. Каждый элемент содержит последнюю промежуточную вершину на кратчайшем пути между вершиной-источником и конечной вершиной. В начале алгоритма все элементы вектора P равны вершине источнику (в начале вектор P содержит такие элементы {1, 1, 1, 1, 1}). Далее на этапе пересчета значения метки для рассматриваемой вершины, в случае если метка рассматриваемой вершины меняется на меньшую, в массив P мы записываем значение текущей вершины W. Например: у 3-ей вершины была метка со значением «30», при W=1. Далее при W=5, метка 3-ей вершины изменилась на «20», следовательно мы запишем значение в вектор P - P[3]=5. Также при W=5 изменилось значение метки у 4-й вершины (было «50», стало «40»), значит нужно присвоить 4-му элементу вектора P значение W - P[4]=5. В результате получим вектор P = {1, 1, 5, 5, 1}.

Понимая, что в каждом элементе вектора P записана последняя промежуточная вершина на пути между источником и конечной вершиной, мы можем найти сам кратчайший маршрут.

3 Программная реализация и тестирование

Для создания приложение необходимо в первую очередь понять суть работы программы. В нашем случае заказчик хочет оборудовать склад Mesh-сетью, которая будет контролировать датчики света, а именно включение и выключения ламп склада. При движении роботизированной техники приложение должно координировать технику по кратчайшему пути из точки А в точку В.

Далее на рисунке 11 будет предоставлена диаграмма последовательности.

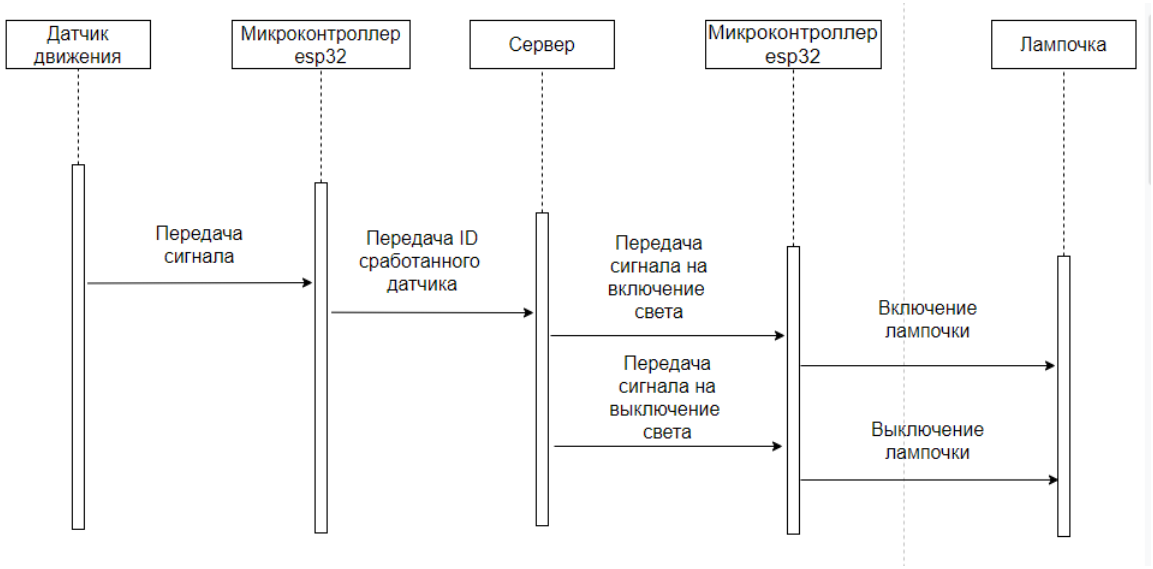


Рисунок 11 – Диаграмма последовательности

За каждым датчиком движения привязана одна лампочка, за микроконтроллером esp32 привязан один датчик движения. При срабатывании датчика движения, микроконтроллер esp32 получает сигнал, который передаётся на сервер. Сервер обработав сигнал при помощи алгоритма Э. Дейкстры, отправляет нужному микроконтроллеру esp32 сигнал

на включение или выключение света. Микроконтроллера esp32 включает или выключает определённые лампочки.

3.1 Настройка сети Z-Wave

Для установки контроллера необходимо: устанавливать контроллер в радиусе всех устройств, чтобы сигнал доставал до всех датчиков. К выбору места установки контроллера нужно подойти ответственно. Не желательно устанавливать контроллеры около металлических шкафов, за стенами. Данный фактор снижает мощность радиосигнала от 55% до 90%.

Модули с постоянным питанием (рисунок 12) – это основа построения Mesh-сети. Через них маршруты перестраиваются в случае необходимости.

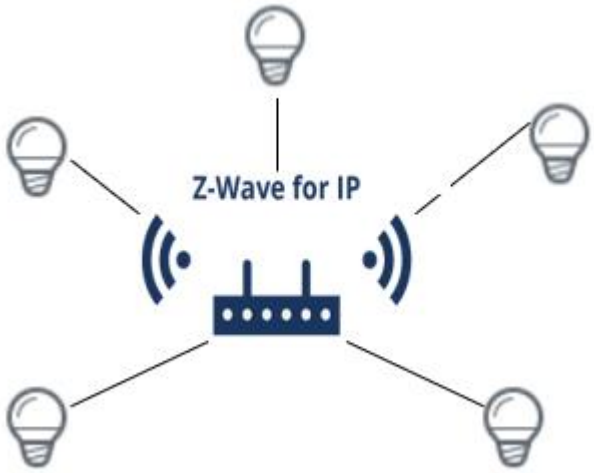


Рисунок 12 – модули с постоянным питанием

В центре рисунка установлен сервер (приложение), которое отправляет сигнал в контроллер на включение лампочек.

Подключение модулей, где один модуль – это датчик движения, контроллер и лампочка, может происходить постепенно, подключать модули необходимо в радиусе действия сети.

После того, как этап добавления всех узлов пройден необходимо проверить сеть. Необходимо убедиться в бесперебойной работе каждого модуля сети.

Если при построении сети возникают зоны с локальной низкой мощностью сигнала, необходимо рассмотреть по очереди один из вариантов повышения качества связи. Добавить вторичный контроллер или добавить модуль с постоянным питанием. Эти устройства будут играть роль дополнительных трансмиттеров и помогут решить проблему.

Узлы, которые не добавляются в сеть при построении маршрута подключаем в последнюю очередь. Данные узлы большинство своего времени находятся в спящем режиме, они ставятся на дальнем конце маршрутов передачи команд. Для быстрого конфигурирования датчика необходимо настроить интервал «пробуждения».

Последним этапом настройки является настройка ассоциации. Настройка группы прямых связей нужна для быстрого и надежного программного взаимодействия. Для отправки отчёта микроконтроллерам уже используются надежные маршруты. Если последовательность сети соблюдалась верно, то датчики и устройства уже будут знать о близости друг к другу.

3.2 Характеристики сети

Первая опция хранит в себе все сведения о чипе z-wave и изменениях, которые вы можете внести.

Пример информации, которую можно увидеть в первой опции:

- Версия: 3.20 L:1-текущая версия z-wave, установленная на устройстве;

- Locale-частота микросхемы z-wave (EU-868,42 МГц b. US-908,42 МГц c. AU-921.42 МГц d. RU-869,0 МГц e. IN-865.2 МГц f. JP-922-926 МГц);

- HouseID - демонстрирующий вам текущий сетевой идентификатор z-волны;

- - Role — это роль вашей сети z-wave (Master SIS без PRI, Master SIS вместе с PRI);

- Last update – время, когда в последний раз был обновлен чип z-wave;

- Recent backup – время последней резервной копирование сети Z-Wave;

- Port - порт блока, получаемого доступ к сети z-Wave.

- Во второй опции вы можете взаимодействовать со следующими параметрами:

- автоматическая настройка устройства - эта опция позволит устройству автоматически перенастраивать ваши устройства z-wave, когда оно запускается впервые;

- изменение версии z-wave - она даст возможность совместить более старые версии с новыми;

- Vera routing-позволяет устройству использовать свой собственный алгоритм для маршрутизации сети z-wave.

Ограничение обнаружение соседней Z- wave (требуется маршрутизация Vera) - заставит каждый узел видеть те, которые находятся поблизости, а не полагаться на те, которые находятся дальше искомого узла.

- Poll nodes-определяет, что модуль будет опрашивать все узлы z-wave из сети z-wave.

Третья опция взаимодействует с интервалами опроса сети z-wave:

- Количество секунд ожидания для начала опроса-сколько времени в секундах устройство будет ждать, пока оно не начнет опрашивать сеть z-wave (по умолчанию 20 секунд);

- Опросить узел в случае неактивности сети Z-Wave - если сеть Z-Wave неактивна ожидание некоторого времени перед опросом узла (по умолчанию 10 секунд);

- Если не указано иная сеть, опросите каждый узел не более одного раза -как часто блок будет опрашивать устройство Z- Wave не более чем в указанном интервале (по умолчанию 60 секунд);

- Частота опроса блока - как часто блок будет опрашивать устройство Z-Wave в указанном интервале (по умолчанию 30 секунд);

Четвёртая опция – это восстановление сети Z-Wave, если произошли неполадки. Данная опция включает в себя восстановление и диагностику проблемы

Восстановление сети Z-Wave позволит установить следующие параметры перед ее запуском:

- Время ожидания устройства Mios других устройств, с батарейным питанием. Когда устройства Mios восстанавливаются, то они начинают искать оборудование в сети. Большинство устройств с батарейным питанием не будут реагировать, за исключением интервалов пробуждения. Среднее время ожидание одного устройства занимает от 30 до 60 минут.

- Повторить компоновку приборов званого. Если оставить его включенным, зеленые значки состояния станут синими, поскольку устройство повторно настраивается, но, когда тест будет завершен, значки снова должны позеленеть.

- Время выполнения стресс-тест в сети. В течение этого времени устройство будет постоянно отправлять данные во все внешние узлы Z-волны и измерять задержку и точность их ответов.

После завершения процесса восстановления появляется краткий отчёт. В данном отчёте находиться информация о рейтингах сигнала для каждого из устройств. Отчёт проинформирует вас о настройке каждого устройства, было ли оно настроено или нет.

Восстановление сети занимает большой промежуток времени, около двух часов. Оно создаёт много трафика в сети и устройства Z-Wave начинают долго реагировать при использовании их во время выполнения тестирования сети. Если у вас есть устройства с батарейным питанием, которые всегда прослушивают команды, называемые Flir, такие как замок двери Schlage LiNK, тестирование будет использовать часть их батари.

Опции с дополнительными настройками позволят взаимодействовать со следующими опциями:

- Reset Z-Wave network-эта опция сбросит чип Z-Wave вашего устройства, и это изменит HouseID сети Z-Wave. Этот параметр следует использовать с осторожностью, так как он удалять сеть Z-Wave и все ранее сопряженные устройства Z-Wave;
- копировать сеть Z-Wave с главного контроллера-эта опция позволит устройству Vera копировать сеть z-wave с главного контроллера Z-wave, тем самым делая устройство вторичным контроллером Z-Wave.
- Controller shift-эта опция будет передавать состояние устройства от первичного к вторичному на другой контроллер z- Wave;

Backup Z-Wave network- данный параметр производит запуск резервного копирование микросхемы z-wave для того, чтобы она могла сохраненяться для восстановления устройства в текущем состоянии из резервной копии устройства.

3.3 Подключение датчика движения

Для подключения датчиков движения на складе понадобятся следующие детали:

- Микроконтроллер esp32;
- PIR-датчик;
- провода папа-папа.

Rip-датчик (рисунок 13) – такие датчики очень часто используются в системе сигнализации. Датчик будет фиксировать перемещение объектов. В нашем случае будет следить за движением пользователя на складе.

Рисунок 13 – Датчик движения

Для подключения датчика движения к esp32(14) требуется воспользоваться схемой подключения (рисунок 15). Провода датчика можно подключить на прямую.

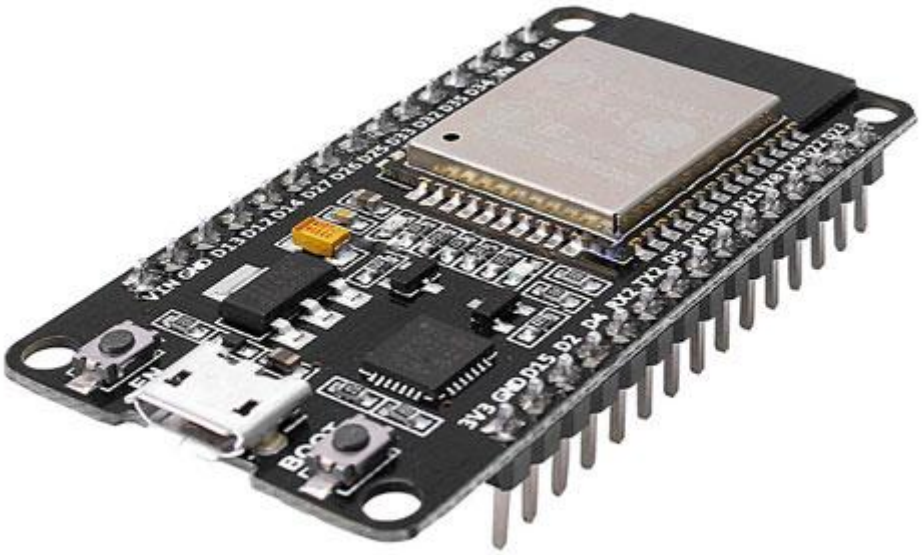


Рисунок 14 – микроконтроллер esp32

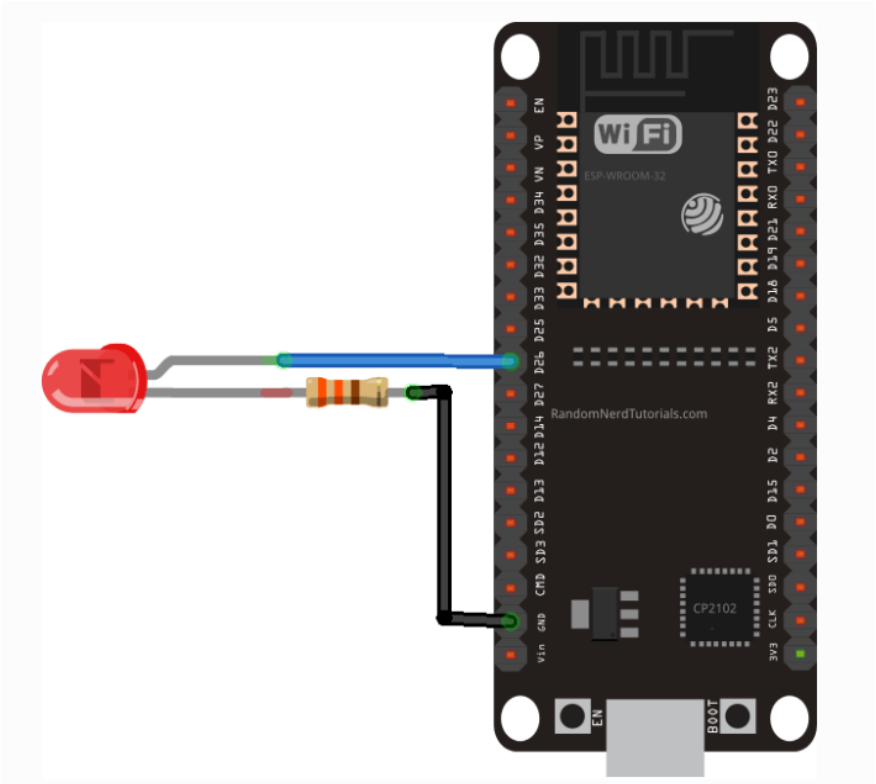


Рисунок 15 – схема подключения датчика движения к микроконтроллеру

При движении объекта возле датчика, сигнал будет поступать на микроконтроллер esp32 и передаваться посредством Mesh-сети на сервер.

3.4 Разработка программного кода

Для создания приложения в первую очередь необходимо инициализировать глобальные переменные (рисунок 16).

```
1   import JavaMeshManager.JavaMeshManager;
2
3   import java.io.Console;
4
5   public class HelloJavaMesh {
6       private static final int HELLO_PORT = 9876;
7       private JavaMeshManager mm;
8       int INF = Integer.MAX_VALUE / 2;
9       int vNum = 48;
10      MultiList graph;
11
12      List<Track> trackList;
13      List<Integer> endPointPirPinList = new ArrayList<>();
14      Map<Integer, Integer> letters = new HashMap<Integer, Integer>();
15
```

Рисунок 16 – инициализация глобальных переменных

- private static final int HELLO_PORT = 9876 – порт для подключение к ретранслятору Mesh-сети;
- private JavaMeshManager mm – объект, который отправляет сообщение в Mesh-сеть;
- int vNum = 48 – количество лампочек (граф вершин);
- MultiList graph – объект для нахождения графов;
- List<Track> trackList – массив данных, который отвечает за количество маршрутов (один маршрут содержит в себе массив сработанных датчиков);

- List<Integer> endPointPirPinList = new ArrayList<>() – количество всех сработанных лампочек;

- Map<Integer, Integer> letters = new HashMap<Integer, Integer>() - количество включения конкретных лампочек.

Далее на рисунке 17 будет предоставлен фрагмент кода, который демонстрирует инициализацию локальных переменных.

```
26
27          Console c = System.console();
28          Console d = System.console();
29
30          if (c == null || d == null) {
31              System.err.println("No console.");
32              return;
33          }
34          mm = new JavaMeshManager();
35
36          mm.bind(HELLO_PORT);
```

Рисунок 17 – инициализация локальных переменных

- Console c = System.console() – переменная, которая отвечает за инициализацию координаты начала пути;

- Console d = System.console() - переменная, которая отвечает за инициализацию координаты конца пути;

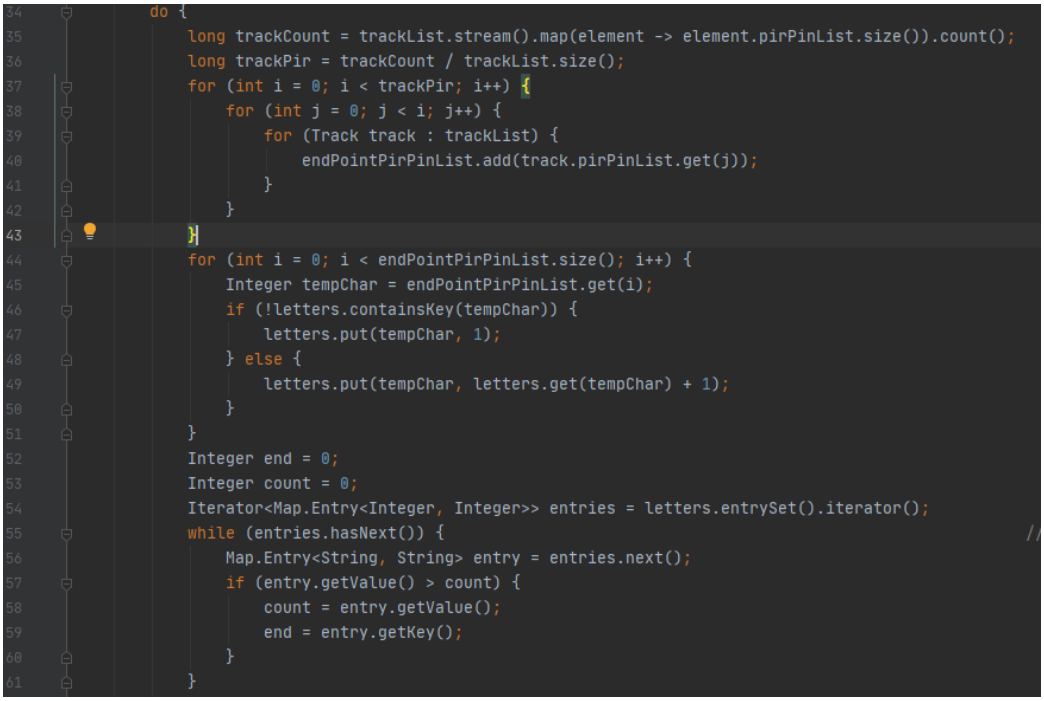
- Далее идёт проверка параметров на валидность, чтоб в дальнейшем не выбросило исключение NullPointerExeption;

- mm = new JavaMeshManager() – создание и инициализация объекта JavaMeshManager;

- mm.bind(HELLO_PORT) – инициализация параметра порта у объекта mm;

После инициализации локальных переменных необходимо приступать к кодированию подсчёта количества срабатывания конкретных лампочек и

нахождения лампочки, которая сработала больше всего раз к ней необходимо проложить маршрут с помощью алгоритма Дейкстры. Далее на рисунке 18 будет предоставлен алгоритм Дейкстры.



```
do {
    long trackCount = trackList.stream().map(element -> element.pirPinList.size()).count();
    long trackPir = trackCount / trackList.size();
    for (int i = 0; i < trackPir; i++) {
        for (int j = 0; j < i; j++) {
            for (Track track : trackList) {
                endPointPirPinList.add(track.pirPinList.get(j));
            }
        }
    }
    for (int i = 0; i < endPointPirPinList.size(); i++) {
        Integer tempChar = endPointPirPinList.get(i);
        if (!letters.containsKey(tempChar)) {
            letters.put(tempChar, 1);
        } else {
            letters.put(tempChar, letters.get(tempChar) + 1);
        }
    }
    Integer end = 0;
    Integer count = 0;
    Iterator<Map.Entry<Integer, Integer>> entries = letters.entrySet().iterator();
    while (entries.hasNext()) {
        Map.Entry<String, String> entry = entries.next();
        if (entry.getValue() > count) {
            count = entry.getValue();
            end = entry.getKey();
        }
    }
```

Рисунок 18 – цикл для нахождения пути

В данном коде происходит:

- инициализация локальных переменных
- проверка на корректность параметров (начала и конца пути)
- объявление массивов пометок (графов), массивов состояний
- подсчёт количества срабатываний конкретных лампочек и нахождения лампочки, которая сработала больше всего раз.
- инициализация RMQ (структура данных) (рисунок 4.5)

Алгоритм собирает со всех маршрутов информацию о срабатывании датчиков движения. Для предоставления информации о кратчайшем маршруте, требуется знать конечную координату. После этого отправляется сигнал для включения конкретной лампочки.

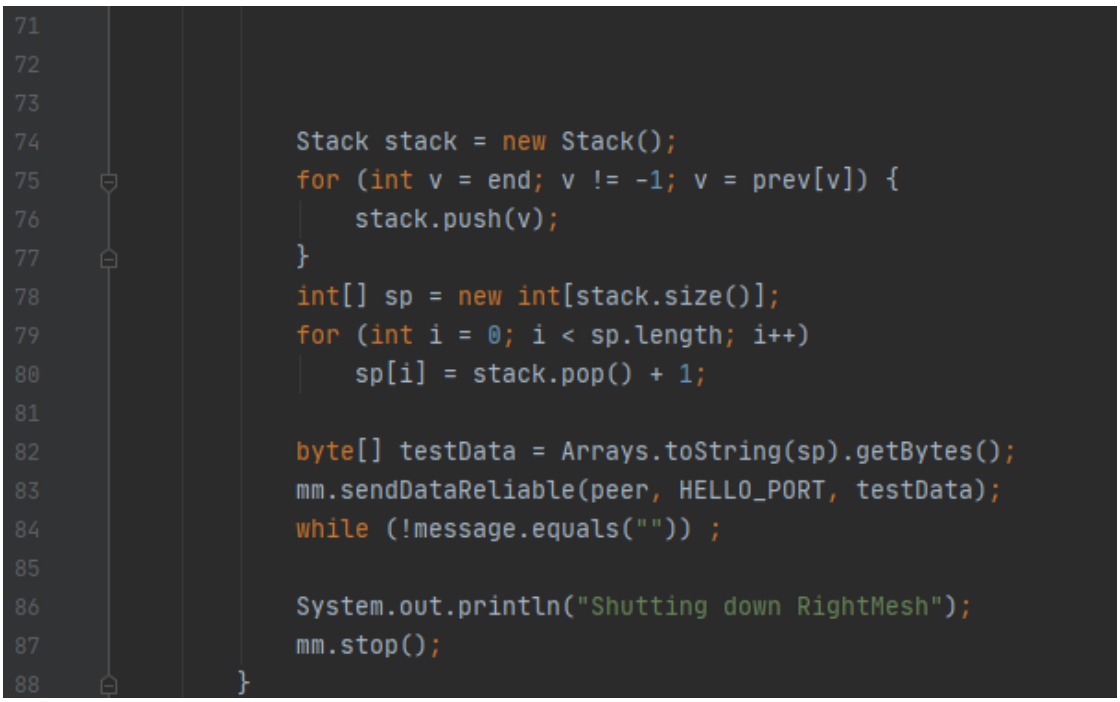
Алгоритм выбирает ближайшую вершину графа (лампочки на складе), после выбора вершины начинается проверка на соответствие, если вершина не найдена или является конечной вершиной графа, то алгоритм перестаёт работу. Информация о графов будет храниться в классе RMQ. Далее на рисунке 19 будет предоставлен класс RMQ.

```
119    class RMQ {
120        int n;
121        int[] val;
122        int[] ind;
123
124        RMQ(int size) {
125            n = size;
126            val = new int[2 * n];
127            ind = new int[2 * n];
128            fill(val, INF);
129            for (int i = 0; i < n; i++)
130                ind[n + i] = i;
131        }
```

Рисунок 19 – класс RMQ

Класс RMQ выполняет функцию структуры данных, хранение информации о графов.

После нахождения кратчайшего пути по алгоритм Дейкстры, необходимо найденные данные перевести в байты и отправить в Mesh-сеть. На рисунке 20 будет продемонстрирована отправка данных в Mesh-сеть.



```
71
72
73
74              Stack stack = new Stack();
75              for (int v = end; v != -1; v = prev[v]) {
76                  stack.push(v);
77              }
78              int[] sp = new int[stack.size()];
79              for (int i = 0; i < sp.length; i++)
80                  sp[i] = stack.pop() + 1;
81
82              byte[] testData = Arrays.toString(sp).getBytes();
83              mm.sendDataReliable(peer, HELLO_PORT, testData);
84              while (!message.equals("")) ;
85
86              System.out.println("Shutting down RightMesh");
87              mm.stop();
88          }
```

Рисунок 20 – восстановление и отправка в Mesh-сеть

3.5 Тестирование системы на складском помещении

Для тестирования системы на складском помещение необходимо установить стандартную сеть 802.11 bgn, к которой будет подключена Mesh-сеть. Mesh-сеть будет управлять микроконтроллером. Так на рисунке 21 можно наблюдать схему организации работы сети.

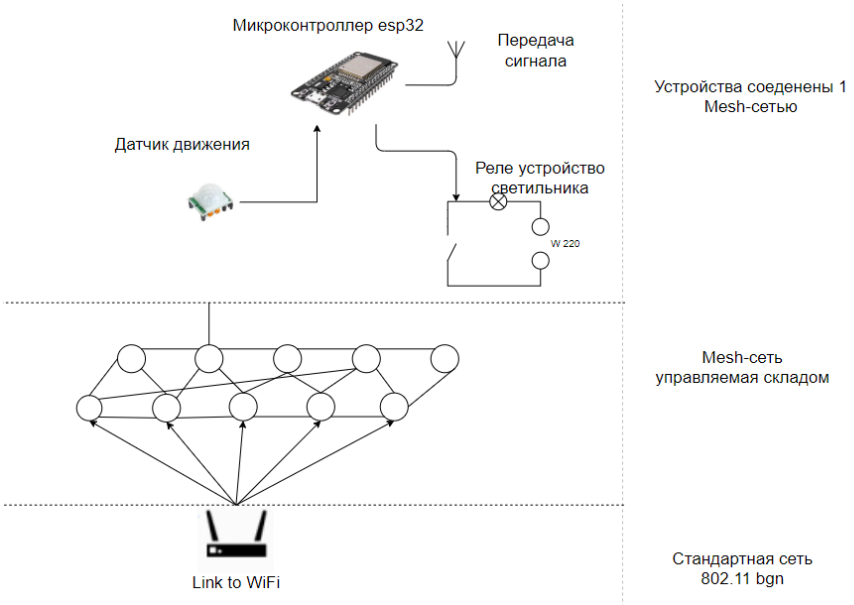


Рисунок 21 - схема организации работы сети

На рисунке видно, что датчик движения будет отправлять сигнал контроллеру микроконтроллеру esp32. Конструкция контроллеров и датчика связана с реле, которое отвечает за подачу электроэнергии. Микроконтроллер esp32 будет передавать сигнал серверу по средству mesh-сети. В таблице 3 будет предоставлено количество комплектующих для склада общей площадью 500кв/м2.

Таблица 3 – необходимо кол-во комплектующих

Наименование компонента	Кол-во, ед.
Микроконтроллер esp32	42
Датчик движения	42
Маршрутизатор Tp-link	1
Лампочки	42
Рабочая станция	1

Для проведения тестирования системы необходимо разработать unit тесты.

Модульное тестирование (unit testing) – процесс программирования, который позволяет проверить на корректность модули исходного кода программы.

Юнит тестирование будет проводиться для класса Classifer.java т.к. это основной класс при работе нашей программы. Тестирование будет проводиться на предмет корректности срабатывания алгоритма Э. Дейкстры.

Тестирование системы проводилась по заранее заготовленным восьми маршрутам. На рисунках 22, 23, 24 продемонстрирован три фрагмента кода юнит-тестирования. Класс содержит массив атрибутов (Track.java), массив ответов (List< LightBulb >).

```
import org.junit.Before;
import org.junit.Test;

import java.util.Arrays;
import java.util.List;

import static org.junit.Assert.assertEquals;

public class ClassifierTest {

    Classifier classifier = new Classifier();

    @Before
    public void setup() {
        Track track1 = new Track();
        track1.pirPinList = Arrays.asList(7,13,1,2,3,4,10,16,9,15,8,14,7,13);
        Track track2 = new Track();
        track1.pirPinList = Arrays.asList(7,13,19,20,21,22,16,10,15,9,14,8,13,7);
        Track track3 = new Track();
        track1.pirPinList = Arrays.asList(7,13,8,14,9,15,10,16,11,17,23,22,21,20,19,13,7);
        Track track4 = new Track();
        track1.pirPinList = Arrays.asList(7,13,1,2,3,9,15,8,14,7,13);
        Track track5 = new Track();
        track1.pirPinList = Arrays.asList(7,13,19,20,21,22,23,24,18,12,17,11,16,10,15,9,14,8,13,7);
        Track track6 = new Track();
        track1.pirPinList = Arrays.asList(7,13,8,14,9,15,10,16,11,17,5,4,3,2,1,7,13);
        Track track7 = new Track();
        track1.pirPinList = Arrays.asList(7,13,8,14,9,15,3,2,1,7,13);
        Track track8 = new Track();
        track1.pirPinList = Arrays.asList(7,13,19,20,21,15,9,14,8,13,7);

        classifier.trackList.add(track1);
        classifier.trackList.add(track2);
        classifier.trackList.add(track3);
        classifier.trackList.add(track4);
        classifier.trackList.add(track5);
        classifier.trackList.add(track6);
        classifier.trackList.add(track7);
        classifier.trackList.add(track8);
    }
```

Рисунок 22 – первый фрагмент кода юнит теста

Аннотация @Before отвечает за инициализацию заранее заданных восьми маршрутов.

```
33
34          classifier.trackList.add(track1);
35          classifier.trackList.add(track2);
36          classifier.trackList.add(track3);
37          classifier.trackList.add(track4);
38          classifier.trackList.add(track5);
39          classifier.trackList.add(track6);
40          classifier.trackList.add(track7);
41          classifier.trackList.add(track8);
42      }
43      @Test
44      public void start() {
45          //when
46          List<Integer> result = classifier.start();
47
48          //then
49          assertEquals(Arrays.asList(7, 13, 8, 14, 9, 15, 3, 2, 1, 7, 13), result);
50      }
51  }
```

Рисунок 23 – второй фрагмент когда юнит теста

В данном фрагменте кода производиться запуск теста и проверка ожидаемых результатов, если результат не верный, то тест падает с информативной ошибкой.

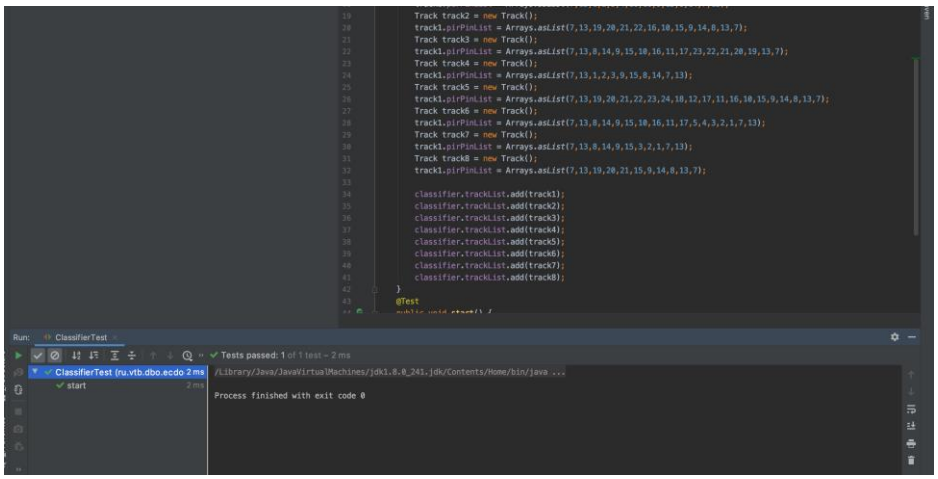


Рисунок 24 – третий фрагмент кода юнит тестирования

На рисунках показан положительный результат на тестирования Classifer.java

3.6 Окупаемость системы Z-Wave

Для подсчёта окупаемости Z-Wave системы необходимо посчитать расходы на приобретения комплектующих данной системы, а именно датчики движения, микроконтроллер, маршрутизатор, рабочая станция. В таблице 4 предоставлена информация по ценовой политике на данные комплектующие.

Таблица 4 стоимость комплектующих системы Z-Wave

Наименование компонента	Ценник на комплектующие в рублях
Микроконтроллер esp32	750
Датчик движения	410
Маршрутизатор Tp-link TP-LINK TL-WR841N	1450
Энергосберегающая лампа	85
Рабочая станция	17 500

Рассчитать сумму установки 1 лампочки можно по формуле 3.1 (3.1)

$$costlight = e + d + l$$

где costlight – это стоимость установки 1 лампочки в рублях;

e – стоимость микроконтроллера esp32 в рублях;

d- стоимость датчика движения;

l – стоимость лампочки.

Так же необходимо рассчитать количество лампочек в системе для этого введём значение countlight.

$$\text{countlight} = x/y \quad (3.2)$$

где x – это размер склада;

y – размещение одной лампочки на 1 кв/м.

Установка всей системы освещения можно рассчитать по формуле: (3.3)

$$\text{countlight} * costlight + s$$

где s – это стоимость рабочей станции.

Экономия электроэнергии в час рассчитывается по формуле: (3.4)

$$economy = u - \frac{r}{p} * g$$

где u – стоимость освещения за час работы всех лампочек

r – длинна кратчайшего маршрута в кв/м;

p – радиус освещение одной лампочки в кв/м;

g – константа, стоимость потребления одной лампочки в рублях в час.

После сравнения данных с квитанции склад стал потреблять на 27.5% меньше электроэнергии благодаря технологии Z-Wave, а это на 813 рублей меньше, чем до внедрения нашей системы. Окупаемость данной технологии за счёт электроэнергии будет происходить 87 месяцев. При больших размерах склада данная система будет быстрее окупаться, а также средний ресурс лампочки на складских помещениях составляет 8 тысяч часов, то есть 3 года, в нашем же случае средний ресурс лампочки будет увеличен с трёх

лет до четырёх т.к. в среднем в день лампочка будет работать не 8 часов, а 5-6 часов. Это так же позволит сэкономить на замене лампочек.

Заключение

В рамках данной бакалаврской работы спроектирована система управлением освещением на основе Mesh-сети. Программная реализация была выполнена в среде разработки Intelliji IDEA на языке программирования Java.

Изучены методы построение Mesh-сети на базе контроллеров esp32, а также изучены протоколы беспроводной передачи данных, для построения сети на базе этих же контроллеров.

Доказана актуальность рассматриваемой темы. Проанализированы существующие решения в области системы освещения «Умного дома», для освещения складского помещения при помощи Mesh-сети.

Разработан алгоритм работы системы освещения, связь через Mesh-сеть между контроллерами и сервером. Подобраны необходимые электронные компоненты, а также написано нужное программное обеспечение для достижения поставленной цели.

Реализована система нахождения кратчайшего пути на основе алгоритма Э. Дейкстры, который даёт возможность находить кратчайшее расстояние от искомого до целевого пути.

Тестирование разработанной системы освещения, а также алгоритма Э. Дейкстры производилось при помощи unit-тестирования, которое затрагивает важные аспекты системы.

На основе спроектированной системы освещения возможна реализация полномасштабной системы, которую можно устанавливать на склады, в квартиры, частные дома.

Произведены расчёты экономии затрат на освещение при функционировании предприятия без использования системы, а также рассчитана стоимость освещения при использовании системы освещения склада на базе Mesh-сети.

Список используемых источников

1. Ардуин и Расбери в проектах Internet of Things. — СПб.: БХВПетербург, 2016. – 320 с.
2. Блум Д., изучаем Arduino: инструменты и методы технического волшебства: Пер. с англ. – СПб.: БХВ-Петербург, 2015. — 336 с.
3. Давенпорт В. В., Рут В. Л. Введение в теорию случайных сигналов и шумов. М.: Издательство иностранной литературы. 1960. 467 с.
4. Интернет вещей: учебное пособие [текст] / А.В. Росляков, С.В. Ваняшин, А.Ю. Гребешков. – Самара: ПГУТИ, 2015. – 200 с.
5. Инструменты и методы для Arduin: учебное пособие [текст] / А.В. Рассольников, С.В. Петин, А.Ю. Гребешков. – Самара: ПГУТИ, 2015. – 200 с.
6. Кучерявый А. Е., Прокопьев А. В., Кучерявый Е. А. Самоорганизующиеся сети. СПб.: Любавич. 2011. 310 с. ISBN 978-5-86983-318-1.
7. Колбанёв М. О., Верзун Н. А., Омельян А. В. Об энергетической эффективности сетей пакетной передачи данных // Известия высших учебных заведений. Приборостроение. 2014.Т. 57. № 9. С. 42–46.
8. Петин В., Создание умного дома на базе Arduino. – М.: ДМК Пресс, 2018. – 180 с.
9. Роберт К. Элсенпитер, Тоби Дж. Велт, Умный дом строим сами / Пер. с англ. – М.: КУДИЦ-ОБРАЗ, 2005. – 384 с.
10. «Интернет вещей» (IoT) в России. Технология будущего, доступная уже сейчас // PricewaterhouseCoopers URL:
11. Arduino Mega 2560 Datasheet [Электронный ресурс] URL: https://www.robotshop.com/media/files/pdf/arduinomega2560datasheet.pdf (дата обращения: 14.03.2018).
12. HC-05-Bluetooth to Serial Port Module [Электронный ресурс] URL: http://www.electronicaestudio.com/docs/istd016A.pdf (дата обращения: 18.05.2018)

13. Specification for LCD Module 1602A-1 (V1.2) [Электронный ресурс] URL: https://www.openhacks.com/uploadsproductos/eone-1602a1.pdf (дата обращения: 17.03.2018).
14. Bangali, J. and Shaligram, A. Design and Implementation of Security Systems for Smart Home Based on GSM Technology. International Journal of Smart Home, Vol.7, No.6 (2013), pp.201-208.
15. Budijono S., Andrianto J., Axis Novradin M. Design and implementation of modular home security system with short messaging system [Text] / EPJ Web of Conferences. — 2014, Vol. 68. doi: 10.1051/epjconf/20146800025.
16. Chitnis, S., Deshpande, N. and Shaligram, A. (2016) An Investigative Study for Smart Home Security: Issues, Challenges and Countermeasures. Wireless Sensor Network, 8, 61-68. doi: 10.4236/wsn.2016.84006.
17. Choudhary V., Parab A., Bhapkar S., Jha N., Kulkarni Ms. Medha. Design and Implementation of Wi-Fi based Smart Home System, International Journal Of Engineering And Computer Science. - 2016, Vol.5 - PP.15852-15855.
18. Kumar A., Tiwari N., 2015, “Energy Efficient Smart Home Automation System” , International Journal of Scientific Engineering and Research (IJSER), Vol. 3 Issue 1 - PP. 2347-3878.
19. Krishna M., V. Narasimaha N., K. Ravi Kishore Reddy, B. Rakesh, 2015, “Bluetooth Base Wireless Home Automation System Using FPGA”, Journal of Theoretical and Applied Information Technology, Vol.77. No.3, PP. 1992-8645.
20. Shirisha Tadoju, J. Mahesh. “Bluetooth Remote Home Automation System using Android Application”, International Journal Of Advanced 48 Technology and Innovative Research, ISSN 23 48–2370 Vol.07, Issue.10, August 2015, PP.1815-1818