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

ИНСТИТУТ МАШИНОСТРОЕНИЯ
(наименование института полностью)

Кафедра «Промышленная электроника»
(наименование кафедры)

13.06.01 «Электро- и теплотехника»
(код и наименование направления подготовки)
«Силовая электроника»
(направленность (профиль))

НАУЧНО-КВАЛИФИКАЦИОННАЯ РАБОТА (ДИССЕРТАЦИЯ)

на тему: Разработка системы и алгоритмов определения положения локального источника звука для энергетической эффективности систем контроля коммерческих зданий и сооружений

Аспирант	С. А. Мальцев	
	(И.О. Фамилия)	(личная подпись)
Руководитель	А. А. Шевцов	
	(И.О. Фамилия)	(личная подпись)

Допустить к представлению научного доклада

Заведующий кафедрой	к.т.н., доцент. А.А. Шевцов	
	(ученая степень, звание, И.О. Фамилия)	(личная подпись)

«_____»_____2021 г.

Тольятти 2021

Содержание

Введение

С течением времени и с развитием прогресса в повседневной жизни человека стали появляется разнообразные инженерные системы и бытовые приборы. Стартовала эпоха автоматизации, поскольку очень многие процессы, которые раньше выполнялись руками, стали работать автоматически. И количество самых разнообразных инженерных систем стало исчисляться не просто десятками, а сотнями единиц. Поэтому управлять и контролировать подобные системы становится все более проблематично. Традиционные инструменты перестают справляться с объемом неоднородной и очень быстро поступающей цифровой информацией.

С помощью компьютерного анализа можно увидеть как определенные, так и малозаметные закономерности, которые не сможет найти человек. Это оптимизирует все сферы жизни, начиная государственным управлением, заканчивая телекоммуникациями и производством. Возникает необходимость исследования систем управления и систематизации обработки данных (в том числе использования подходов, инструментов и методов обработки Big Data (больших данных)).

В систему управления инфраструктурой зданий внедряются новые методы контроля положения объектов в помещении. Например, искусственное зрение, распознавание лиц, дорожных знаков. Кроме того, существуют и другие методы, такие как оптическая, радиолокация, тепловая локация и звуковая локация. У каждого этого метода есть свои преимущества и недостатки. В данной научно-квалификационной работе будет уделено внимание пассивной звуковой локации, когда сам объект является источником звуковых волн. Это актуально, потому что представленные в настоящее время системы звуковой локации имеют высокую стоимость. К

тому же в разрабатываемой системе будет применена новая, современная технология «Internet вещей».

Таким образом, целью научно-квалификационной работы является разработка доступной системы определения положения локального источника звука. Система будет применяться в коммерческих зданиях и сооружениях для функций оптимизации и контроля расхода электрической энергии.

Научная новизна предстоящего исследования состоит в том, что для определения положения локального источника звука применяется современная технология «Internet вещей».

1 Состояние вопроса

1.1 Основные задачи и функции систем АЗУС

В завершении семидесятых годов прошлого двадцатого века ученые в Японии и США начали проводить работы по внедрению автоматизации в управление инженерных систем, результаты которых в настоящее время принято называть «Умным домом». Это определение, «Умный дом» первые было сформулировано и озвучено в Вашингтонском Институте интеллектуального здания. Его озвучили в следующем изложении: Умный дом – это здание, обеспечивающее продуктивное и эффективное использование рабочего пространства [1]. Изначально имелись в виду исключительно системы оптимизации потребления электричества и воды, но с течением времени смысл значительно увеличился. На сегодняшний день в систему «Умный дом» также входят системы поддержания климата, охранная система и система видеонаблюдения, контроль доступа к комнатам и помещениям, интеграцию электроприборов и многое другое.

В настоящее время определение звучит следующим образом: Система «Умный дом» – это программно-аппаратный комплекс, упрощающий проживание людей и обеспечивающий автоматизацию процессов и операций в современном здании. А другое название системы «Умный дом» представляет собой аббревиатуру АСУЗ (Автоматизированная Система Управления Зданием).

Предполагается, что основными и главными задачами систем АЗУС это создание безопасной и комфортной среды проживания, минимизация расхода ресурсов, предоставление средств управления и автоматизации и как следствие улучшение качества жизни конечного пользователя [2]. Особенно стоит отметить еще одну важную задачу систем «Умный дом» – помощь

людям с ограниченными возможностями и пожилым людям [3]. Согласно исследованиям, в среднем человек тратит около 1 часа в день на выполнение бытовых операций, таких как включение света, открывание штор и так далее [4]. Поэтому потребность в системах такого рода неуклонно растет.

Как показано на рисунке 1, на сегодняшний день АСУЗ имеют следующие возможности:

– управление температурой,

– управление освещением,

– управление работой электроприборов,

– управление водоснабжением,

– контроль доступа к помещениям (системы безопасности),

– видеонаблюдение,

– имитация присутствия человека,

– управление телекоммуникационными системами,

– техническое зрение.

Рисунок 1 – Функции АСУЗ

Главной особенностью систем «Умный дом» является объедение всех перечисленных функций в единый автоматизированный комплекс.

В настоящее время продвигается несколько проектов по развитию систем «Умный дом» (АСУЗ). Как оценивает Федеральная сетевая компания, благодаря внедрению технологий «интеллектуальных» сетей и зданий произойдет уменьшение потери в электрических сетях России всех классов напряжения до 25%, и это позволит достигнуть экономии 34-35 млрд. 1 кВт · ч в год [1].

Как было сказано ранее, системы «Умный дом» активно применяются для улучшения условий жизни людей в возрасте (пожилые) [3]. По данным Организации Экономического Сотрудничества и Развития процент людей, чей возраст превысит 65 лет, увеличится к 2050 году на 20% [5], что, несомненно, увеличит востребованность «Умного дома».

Неудивительно, что многие крупные фирмы, такие как Apple и Xiaomi заинтересованы в разработке систем «Умного дома». Последняя фирма уже имеет ряд продуктов в соответствующей линейке (рисунок 2).

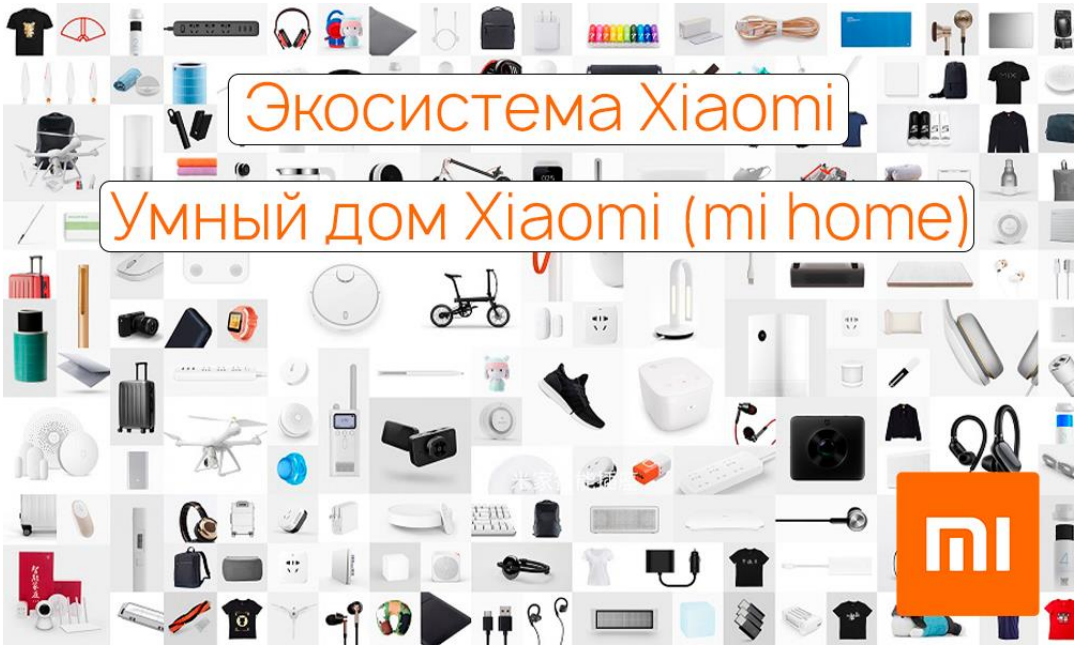


Рисунок 2 – Продукты «Умный дом» Xiaomi

1.2 Анализ актуальности разработки интеллектуальных зданий

В мае 2003 года в публикации журнала «Автоматизация в промышленности» активно обсуждали тему «Интеллектуальные здания: приборы и системы». Генеральный директор компании «Интеллектуальные дома» О.Е. Павлов считает, что главным содержанием подобных проектов является диспетчеризация, учет ресурсов, энерго- и ресурсосбережение, повышение рыночной стоимости здания [6]. Представитель компании «Оптима» Б.М. Либерман написал, что «интеллектуальное здание» – это комплекс систем, интегрированных в единое информационное и управляющее пространство [7]. Такого же мнения придерживается А.В. Фрейдман (компания Науцилус), отметив, что в случае оснащения здания системами и оборудованием разных производителей важно, исключить «конфликт» технических устройств между собой, они должны быть совместимы и представлять единое целое [8]. Компания ЗАО «Нанко» представила Автоматизированную Систему Управления Зданием (АСУЗ) основной задачей которой является создание комфортных и продуктивных условий проживания и (или) работы, обеспечение нужного уровня безопасности (защита от несанкционированного доступа, стихийных бедствий, обеспечение сохранности и целостности имущества); снижение эксплуатационных расходов за счет рационального использования всех потребляемых ресурсов [9]. В свою очередь Е.К. Беспалов и Е.Г. Левина из компании «Инда Софт» снова отмечают, что системы здания, такие как жизнеобеспечения, учета и безопасности необходимо интегрировать в единую диспетчерскую систему [10]. Таким образом, согласно итогам обсуждения темы, актуальностью интеллектуального здания является создание комфортного интеллектуального здания для ресурсосбережения с единой системой управления.

В октябре 2006 года журнал «Автоматизация в промышленности» вернулся к интеллектуальным зданиям, на этот раз обсуждалась тема «Автоматизированное управление инженерными системами зданий: снижение расходов при эксплуатации». А.В. Уваров из компании «ДЭП» пишет о необходимости создания системы диспетчеризации, для контроля огромного объема инженерного оборудования [11]. Менеджер по продукции компании «Прософт» Н.Н. Жиленков предложил разделить системы автоматики по функциональным особенностям и степени важности для жизнеобеспечения [12]. А.В. Паршиков (Компания ФИОРД) снова отмечает, что помимо обеспечения комфорта и функциональности, одной из основных задач, которые должны и призваны решать интеллектуальные системы, является снижение затрат управляющей компании и владельцев на эксплуатацию здания [13]. Представители ООО НПКФ "ДЭЙТАМИКРО" сравнили интеллектуальное здание с салатовым зданием, из комплекса решений, направленных на обеспечение увеличения качества благоприятных условий работы и жизни людей в целом. Такое здание разумно потребляет воду, электричество и прочие ресурсы, полностью убирает негативное воздействие на окружающую среду [14]. Следовательно, за прошедшие три года к ранее существующим проблемам добавилась новая – возрастающие объемы «умного» оборудования и способы управления им.

Спустя два года, в октябре 2008 года журнал «Автоматизация в промышленности» затронул тему «Интеграция систем автоматизации интеллектуального здания». Руководитель группы разработки стандартов НП АВОК по АСУЗ В.В. Ильин отметил, что одной из основных задач АСУЗ является уменьшение затрат на эксплуатацию, в первую очередь, на персонал за счет повышенного уровня автоматизации, так же после анализа сделаны выводы, что АСУЗ нуждается в достойном нормативном обеспечении [15]. В статье А.В. Широкова из компании КРОК сказано, что для эффективной эксплуатации инженерных систем внедряется автоматизация управления зданием, которая может экономить электроэнергию и другие ресурсы,

уменьшает технические и финансовые риски, является основой для защиты инвестиций в недвижимость и находящееся в здании имущество, а также находящимся в здании людям обеспечиваются безопасные условия пребывания. Получается, что с течением времени актуальность разработки АСУЗ сохраняется.

В сентябре 2011 года журналом «Автоматизация в промышленности» была рассмотрена родственная тема «Автоматизированные системы управления освещением». Главный редактор журнала Н.И. Аристова пишет, что внедрение подобных систем способствуют решению двух актуальных для общества задач – энергосбережение и сохранение качества освещения [17]. Последующие статьи описывают различные способы их реализации, с главной задачей – энергоэффективности. Помимо этого, А.Ю. Угреватов и В.Ю. Угреватов (НПФ «КРУГ») выделили еще и значение повышения надежности эксплуатации системы уличного освещения и уровня безопасности пешеходов и водителей [18].

В 2015 году в журналах появляются статьи, посвященные концепции «Internet вещей» (Internet of Things) как особой сенсорной, вычислительной и коммуникационной сети, построенной на базе датчиков и подобных им автономных устройств, связанных и обменивающихся друг с другом информацией [19]. С помощью данной концепции можно повысить качество управления различными процессами, в том числе и АСУЗ. Компания PTC приводит в своей статье [20] уровни внедрения технологии в предприятие.

Последующий анализ уже более современных статьей приводит к выводам, что разработка АСУЗ не теряет своей актуальности, более того, появляются новые аспекты, которые необходимо изучать. Становится необходимым появление отечественной «умной» системы управления, выполняющей необходимый набор функций по доступной представителям среднего класса населения цене [21]. Многие статьи также указывают на актуальность создания интеллектуальных зданий в наше время, в связи с необходимостью обеспечить помощь пожилым людям и людям с

ограниченными возможностями. В статье [22] указано, что наблюдается рост стареющего населения и значительное снижение рождаемости, таким образом, создание «умного дома» позволит уменьшить растущие расходы на социальные нужды.

В результате анализа научных публикаций в период с 2003 по 2019 года можно сделать выводы о том, что разработка систем автоматизированного управления зданиями является темой, не теряющей своей актуальности с течением времени. Это обусловлено тем, что подобные системы позволят значительно экономить природные ресурсы, помогут создать более комфортные условия проживания и работы конечным пользователям, обеспечат более комфортные условия жизни пожилым людям и людям с ограниченными возможностями. При этом сложность создания систем АСУЗ состоит в том, что система управления должна быть единой, с возможностью обработки больших объемов данных, и должна быть доступной для большинства слоев населения. Одним из решений, в процессе реализации описанной выше системы, является использование концепции «Internet вещей» (рисунок 3).

Рисунок 3 - Концепция «Internet вещей»

1.3 Анализ подходов, инструментов и методов обработки Big Data (больших данных)

В настоящее время традиционные инструменты не справляются с обработкой разнообразной неоднородной и быстро поступающей цифровой информацией. С помощью компьютерного анализа можно увидеть как определенные, так и малозаметные закономерности, которые не сможет найти человек. Оптимизируется все сферы нашей жизни, начиная государственным управлением, заканчивая производством и телекоммуникациями.

Термин Big Data появился относительно недавно. Как показано на рисунке 4, активный рост его употребления приходится на 2011 год.

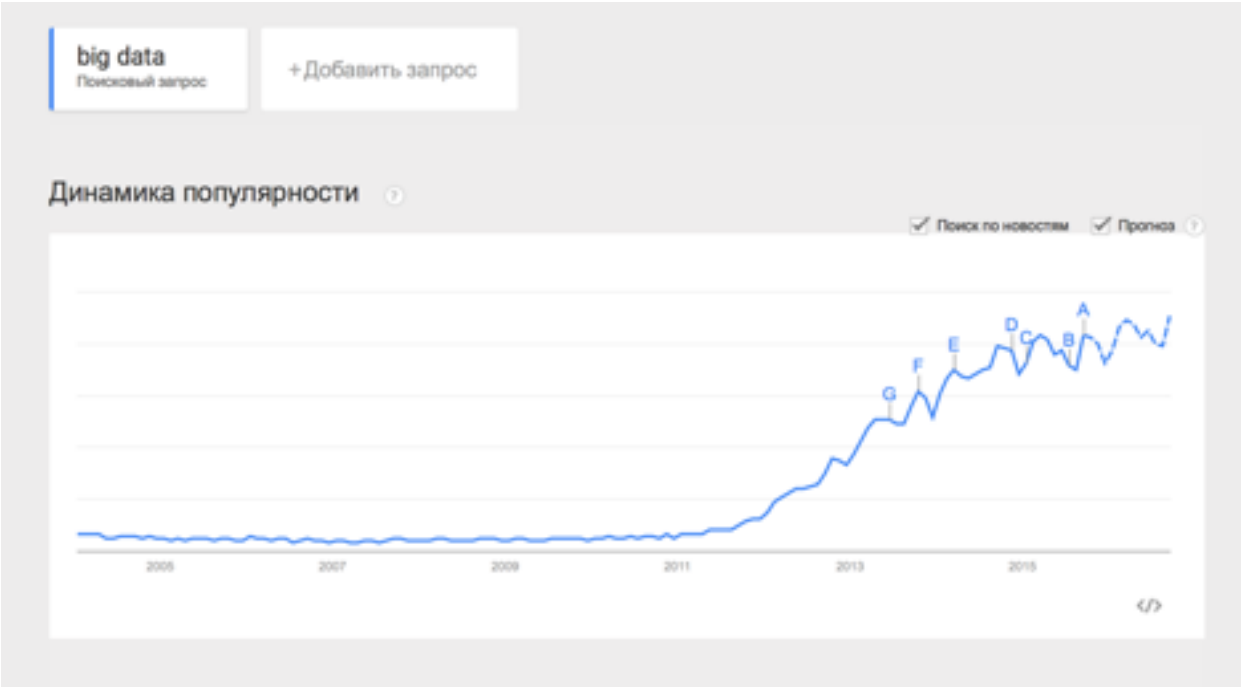


Рисунок 4 – Динамика популярности запроса Big Data

Однако, не смотря на его активное употребление, существуют заблуждения в его корректном определении. Например: BigData это когда

данные превышают какой либо объем (100 Гб, 500 Гб, 1 Тб и т.д.), или BigData это такие данные, которые нужно обязательно обрабатывать исключительно большим множеством компьютеров, а одной единицей.[23].

Тщательно проанализируем определение Big Data: Большие данные (англ. big data) – это инструменты, алгоритмы, различные подходы, а так же методы обработки данных огромных объемов. Подобные данные, не смотря на то были они структурированы или нет, должны быть представлены в формате, понятном человеку.

Делаем выводы: Big Data это далеко не конкретный объем данных, и уж тем более не просто данные. Big Data это методы обработки данных группами самых разных систем. Преимущество и достоинством данных методов состоит в том, что они применяются как к маленькому объему данных (книга в электронной библиотеке, или единственная страница в интернете), так и к огромным массивам данных (электронная библиотека или весь интернет).

Термин «большие данные» впервые сформулировал и озвучил в далеком 2008 году редактор журнала Nature Клиффорд Линч, и указал, что их нужно обрабатывать для получения нужных и конкретных результатов для дальнейшего эффективного и оптимального применения [24].

После того, как мы разобрались с определением Big Data, представим их главные принципы:

– Горизонтальная масштабируемость. Система для обработки больших данных имеет возможность изменять свои размеры, и подстраиваться под объем обрабатываемых данных. В случае изменения размера данных, как в большую, так и в меньшую сторону, аппаратное обеспечение должно измениться пропорционально.

– Отказоустойчивость. Первый принцип означает, что не идет в работе не только одна машина, таких машин может быть много. Получается, что с течением времени компьютеры начнут выходить из строя, и их нужно будет ремонтировать. Это необходимо учитывать во время

проектирования методов работы с данными. Серьезных последствий из-за сбоев быть не должно.

– Локальность данных. Большие данные не могут располагаться только на одном компьютере. Для этого требуется большое количество машин. Так же нужны машины и для обработки данных. Необходимо все спланировать так, что бы данные находились и обрабатывались на одном и том же компьютере.

В состав Google входит MapReduce, которая представляет собой модель обработки данных. Она может обрабатывать большие объемы данных на компьютерных кластерах. Логика ее работы представлена на рисунке 5.

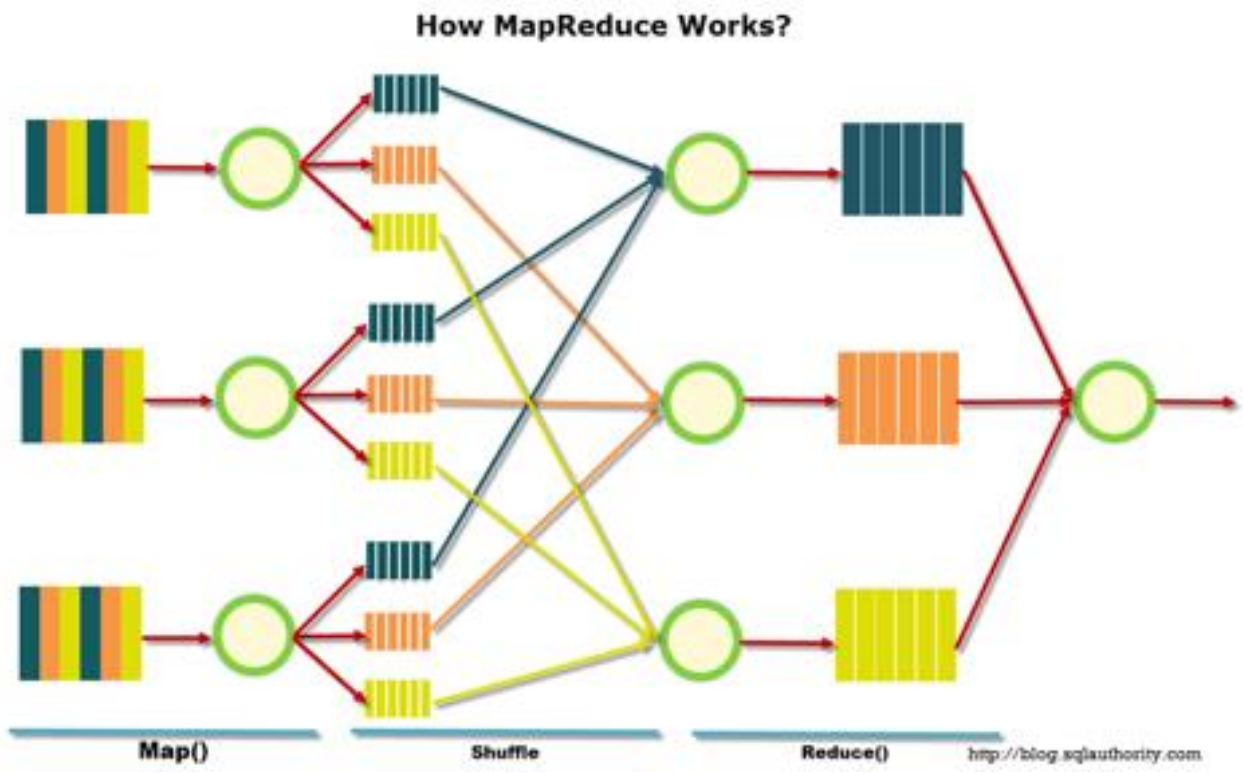


Рисунок 5 – Логика работы MapReduce

MapReduce исходит из того, что данные представлены в форме записей.

Обработка данных происходит в три стадии [25]:

– Первая стадия предобработки данных map(), которая представляет собой созданную пользователем функцию. Эта стадия должна отфильтровать данные. Функция применяется ко всем входным записям. После применения функции map() к входной записи выдается множество ключей-значений. Их может быть как несколько, так ключи могут отсутствовать в принципе. Пользователю необходимо грамотно выбирать что будет располагаться в ключе, поскольку нельзя делать так, что бы были данные с одинаковым ключем, в противном случае они окажутся в едином экземпляре функции reduce.

– Стадия Shuffle. Ее выполнение происходит скрыто для пользователя. Результат этой функции сортируется по корзинам. В одной корзине находится один ключ стадии map.

– Стадия Reduce. «Корзины» с поступают на вход функции reduce(), где по ключас-значениям для каждой «корзины» производится расчет. В итоге этой функции получится множество всех значений, которое представляет собой конечный результат поставленной ранее задачи.

Источниками больших данных являются:

– интернет (социальные сети, форумы, месседжеры, блоги, СМИ и прочие сайты),

– архивы документов,

–показания датчиков, приборов и других устройств.

Последнее напрямую относится к исследуемой теме – системе управления инфраструктурой зданий. Очевидно, что в системе АСУЗ (Автоматизированная Система Управления Зданием), только в одном офисе как минимум должны быть установлены системы:

– управлением температурой,

– управлением освещением,

– контроля доступа к помещению,

– видеонаблюдение.

Учитывая, что только на одном этаже может находиться несколько десятков офисов, да и сами этажи могут исчисляться десятками, получается, что у нас может быть несколько тысячи единиц оборудования, данные которого необходимо обрабатывать. Следовательно, необходимо использовать инструменты и методы обработки больших объемов данных.

1.4 Анализ информационной безопасности интеллектуальных зданий

Какая бы совершенная не была бы технология, у нее равно или поздно обнаруживаются уязвимости. При росте числа устройств и технологий, используемых в системе, растет и уязвимость. Более того, в процессе интеграции разных решений возможна ошибка в процессе проектирования. Из-за этого могут появиться дополнительные слабые места в системе [26]. Что бы технологии «Умный дом» и «Интернет вещей» получили весь требуемый функционал, они наделяются огромными возможностями и хорошим уровнем доступа, они собирают о пользователе большое количество информации, и могут управлять всеми важными функциями жизнеобеспечения. На сегодняшний день «Умный дом» может управлять температурой, освещением, работой электроприборов, водоснабжением, системой безопасности, видеонаблюдением, имитацией присутствия человека, телекоммуникационными системами и техническим зрением. Вероятность возникновения событий по угрозам информационной безопасности выше с более сложных системах, и ниже в простых системах.

Существующие угрозы делятся на два типа: непреднамеренные (факторы внешней среды) и преднамеренные (действия злоумышленников) [27]. Более подробное распределение угроз представлено на рисунке 6.

«Умные дома» подключены к интернету, поэтому данные неминуемо передаются удаленно между системой и гаджетом пользователя. Или бывает так, что большая часть данных обрабатывается удаленно, на сервере производителя системы. Его защищенность вызывает большие вопросы, а проверять ее мы никак не можем [28]. Если современные компьютеры или смартфоны, как правило, защищены должным образом, то Интернет вещей – относительно новая технология, и вопросы ее защищенности в настоящий момент проработаны слабо. Стоит еще учитывать, что «Умный дом» – это не обязательно жилой дом, это может быть государственное учреждение, стадион и даже аэропорт. Все это очень привлекает злоумышленников.

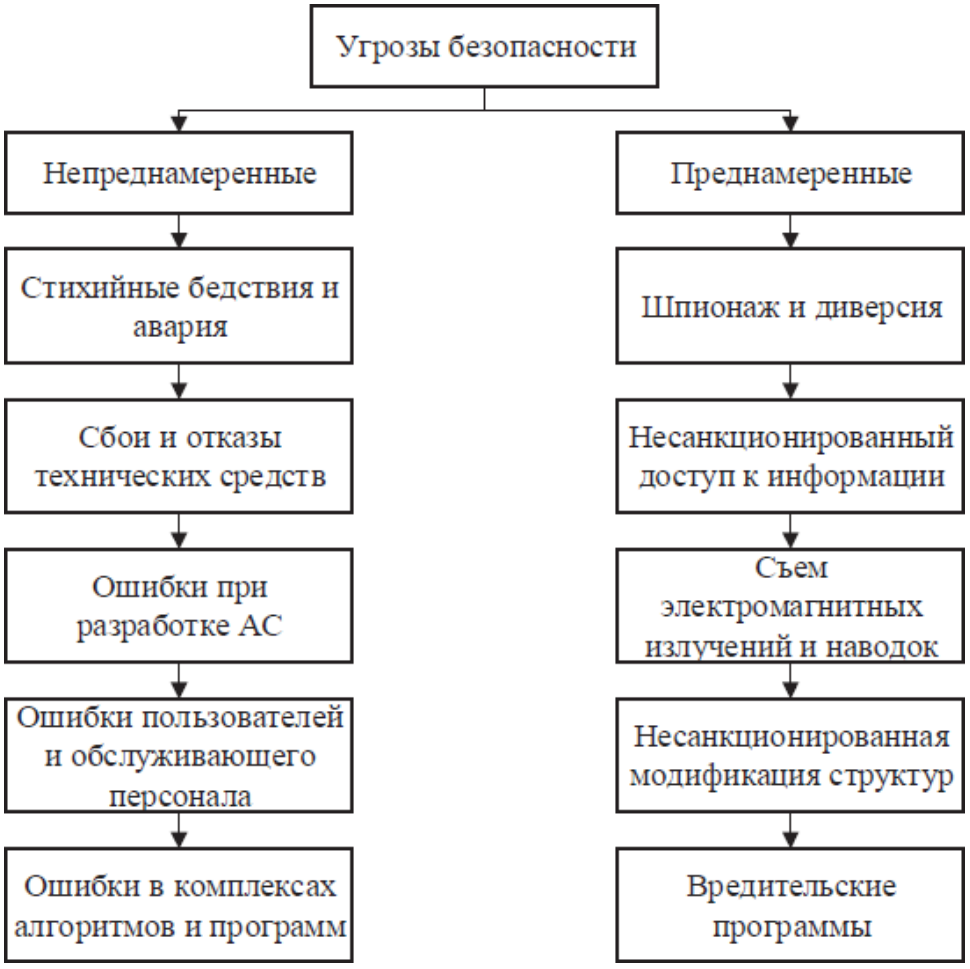


Рисунок 6 – Угрозы безопасности

Слабыми местами современных систем автоматизированного контроля жизнеобеспечения умного дома могут воспользоваться мошенники и предпринять попытку атаки. Крупные объекты, такие как, например аэропорт, могут быть надолго заблокированы преступниками. Кроме того, они могут посеять панику, выведя из строя систему жизнеобеспечения, заблокировать систему безопасности изнутри, что приведет к серьезным последствиям. Выполнив взлом системы, хакеры от лица собственников могут распоряжаться помещением, и проникнут в него всего за пару минут без привлечения внимания. Или еще хуже, можно запереть хозяина дома и просить денежное вознаграждение за свободу.

Доступ в помещение не единственное, что может интересовать злоумышленников. Атака может проводиться для получения доступа к ценным данным (личные фотографии, видео, цифровые дневники) или к видеокамерам и микрофонам для прослушивания частных разговоров. Некоторые системы собирают такую подробную персональную информацию как: имя, адрес, дату рождения, номер телефона и другие данные, используя которые можно получить доступ к данным других систем, например, банковским.

Учитывая все выше сказанное, необходимо обеспечить эффективную и надежную защиту подобных систем. Рассмотрим возможные варианты:

– Система управления информацией и событиями (SIEM). Данные системы используются для управления событиями безопасности. Они просматривают и анализируют различные события на предмет подозрительной модели поведения и реагируют на них. Например, если логин входит в систему только по утрам или по вечерам, и неожиданно авторизовался в середине дня, система обратит вникание. Или если за малый промежуток времени было слишком много неудачных попыток авторизации (введен неверный пароль). Выявляя подобное, SIEM информирует администраторов о том, что происходят/имели место атаки или странное поведение [29];

– Важно защитить связь между смартфоном пользователя и системой «Умный дом». Например, шифрованием передаваемого контента или использованием VPN;

– Необходимо усложнить проверку подлинности, используя более сложный пароль или двухфакторную аутентификацию;

– Максимально протестировать облачные интерфейсы на три уязвимости: возможность последовательного перебора значений, слабой политики паролей и отсутствия блокировки учетных записей [30];

– По возможности осуществлять подключение к Интернету по витой паре. Данное решение необходимо для дополнительной безопасности, так как передача данных по каналу Wi-Fi не является безопасным и надежным подходом [31];

– Поддержка автоконфигурации. Для ее реализации требуется реализовать функциональность в шлюзе и облачных сервисах. Когда новое устройство подключается к сети, шлюз будет использовать его индефикатор для опроса доверенного веб-сервиса и получит все сведения об устройстве: его функциональные возможности, команды, протоколы шифрования. Большинство подходов автоматической конфигурации требуют, чтобы большая часть этой информации была сохранена на самих устройствах. Благодаря данному подходу простой идентификатор устройства и веб-служба гарантируют, что эта информация легкодоступна и остается актуальной [32];

– Обновление программного обеспечения и прошивки Интернет вещей (IoT). Настольные операционные системы и мобильные устройства (смартфоны) регулярно и автоматически обновляются по мере выявления и исправления уязвимостей, в том числе механизмы проверки подлинности изменений. Это дает дополнительную экономическую жизнеспособность системам и устройствам. Но у сотен различных устройств IoT регулярная служба обновления отсутствует

[31]. Для обеспечения целостности и подлинности обновлений и предотвращения возможного вмешательства в микропрограммное обеспечение, такое как внедрение вредоносных программ, для обновлений рекомендуется применять цифровые подписи на основе сертификатов. Цифровые подписи позволяют защититься от злоумышленников, которые решат замаскировать вредоносную программу под легальную прошивку;

– Привязка MAC-адресов удаленных устройств, исключив возможность подключения к сети недоверенного устройства.

1.5 Перспективные направления развития АСУЗ

Кроме безопасности, у «Умного дома» существуют еще несколько перспективных направлений развития:

1) Распознавание поз человека, с помощью внедрения системы технического зрения. Как сказано в источниках [33] и [34], за элементарную единицу поведения человека в пространстве принимается «позирование». Оно характеризует положение частей тела человека по отношению друг к другу. А это означает, что если фиксировать пространственные характеристики корпуса тела и его конечностей, то тогда можно осуществить взаимодействие с системами «Умного дома», и сделать выводы о потребностях объекта и о психическом состоянии личности.

2) Внедрение многопользовательской системы. Для каждого пользователя создается профиль, определяющих в каких пределах поддерживать те или иные параметры (температура воздуха, уровень освещенности и т.д.). В случае присутствия в одном помещении нескольких жильцов – устанавливаются параметры более приоритетного профиля [35]. Приоритетность определяется по доминирующей в системе личности, но при

этом учитывается проверка на уязвимые категории людей: маленькие дети, беременные женщины, люди с ограниченными возможностями и пр.

3) Создание принципиально новых классов умных устройств, например «Умное зеркало», и другие устройства, встраиваемые в интерьер [36].

1.6 Задачи научно-квалификационной работы

Подводя итоги первого раздела, делаем выводы, что в настоящее время разработка эффективной и безопасной структуры системы «Умный дом», для управления как жилыми, так и коммерческими зданиями, является актуальной научной темой. Система «Умный дом» позволит уменьшить расход энергоресурсов и повысить качество жизни его пользователя. Таким образом, разработка доступной системы определения положения локального источника звука необходима для реализации некоторых структурных элементов «Умного дома».

Для достижения цели диссертации необходимо решить следующие задачи:

- разработка системы сбора информации с датчиков,
- разработка и выполнение научного эксперимента по тестированию системы,
- выполнение математического расчета координат источника звука,
- создание системы акустической локализации,
- разработка программных алгоритмов расчета местоположения источника звука.

2 Разработка системы сбора информации с датчиков

2.1 Создание структуры системы сбора информации

Ключевым моментом в разработке прототипа системы «Умный дом» является выбор управляющего устройства и типа системы управления. Можно использовать готовые системы, но у них есть следующие недостатки: дороговизна и закрытый программный код, делающий невозможным тонкую настройку под нужды каждого потребителя [37]. Поэтому, наиболее оптимальным решением будет разработка прототипа «Умного дома» самостоятельно. В настоящее время в роли управляющего устройства все чаще используются микроконтроллеры.

Типы систем управления можно разделить на три категории:

– Централизованная система управления. В самом центре такой системы располагается центральный процессор (контроллер), имеющий в своем составе исполнительные блоки. К этому главному контроллеру подключаются абсолютно все устройства системы. Не исключено, что некоторые эти устройства могут оснащаться своими микроконтроллерами, но управление исходит только из главного контролера и его программы.

– Децентрализованная система управления – в этой системе главный контроллер отсутствует. Ее смысл в том, что шина соединяет все ее устройства и компоненты. В каждом устройстве есть свой собственный контролер с программой, и работает самостоятельно. Каждый элемент программируется независимо от остальных.

– Смешанная система – в ней присутствуют элементы как обоих рассмотренных систем.

«Умный дом», построенный на централизованной системе управления плохо показал себя на практике, поскольку в случае возникновения неисправности центрального контроллера, система отключается полностью. Известны случаи, что в подобных ситуациях владельцы «Умных домов» не могли войти внутрь здания до приезда ремонтной бригады, их не пускала собственная охранная система. Поэтому систему управления необходимо спроектировать таким образом, чтобы в случае выхода одного элемента из строя «Умный дом» отключался не полностью, а только частично в точке неисправности. Такое, возможно, реализовать на децентрализованной или смешанной системе управления. Поэтому каждый элемент «Умного дома» в разрабатываемой структуре управляется собственным микроконтроллером, который представляет собой микросхему, предназначенную для управления различными электронными устройствами [38].

Исследование многочисленных литературных источников в данной области техники, позволило прийти к выводу, что наиболее оптимальным будет использование микроконтроллера на платформе Arduino (рисунок 7), поскольку он обладает такими преимуществами как: низкая стоимость, легкость программирования, поддержка многочисленных датчиков и исполнительных механизмов широкого спектра применения.

Arduino – это торговая марка, в которую входит аппаратная вычислительная платформа. Основными компонентами этой платформы являются весьма простые платы ввода-вывода на микроконтроллерах семейства ATmega. Кроме того, в торговую марку входит и специальная интегрированная среда разработки.

Среда разработки, представленная на рисунке 8, базируется на программе Processing и спроектирована так, что интуитивно понятна даже для тех, кто только начал программировать и не близко знаком с разработкой программного обеспечения.

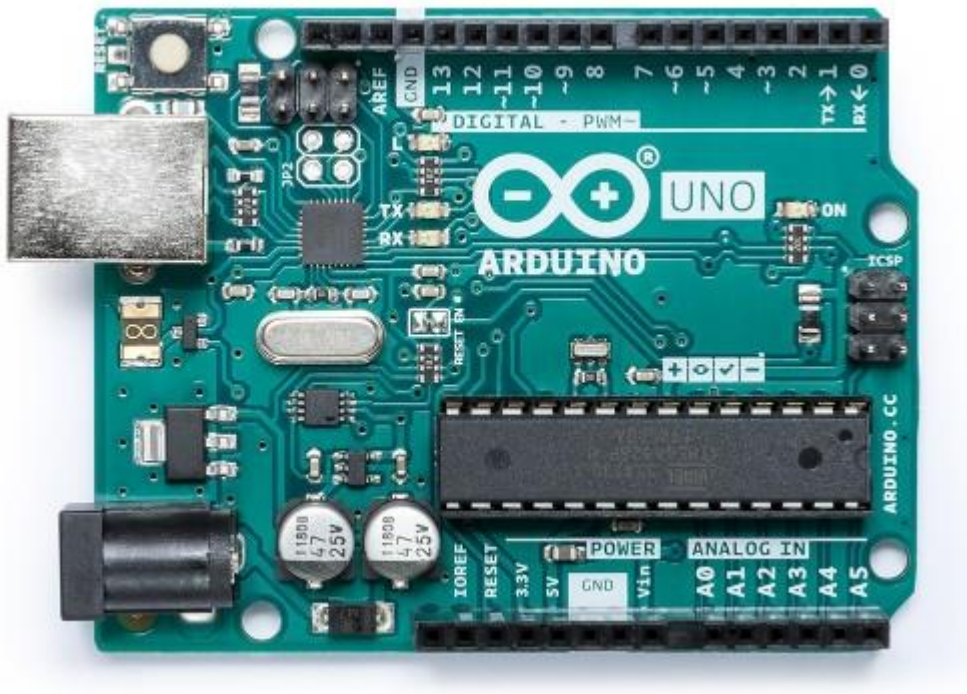


Рисунок 7 – Arduino UNO

Arduino и прочие совместимые платы сделаны так, что при необходимости расширяются, а в схему добавляются новые устройства и компоненты. Последними являются устройства ввода-вывода (например SD-карты, светодиодные индикаторы, ЖК-дисплеи), огромное количество датчиков широкого спектра применения, устройства коммуникации (такие как GSM, Wi-Fi, Ethernet и прочие). Такие платы расширений подключаются к Arduino с помощью установленных на них специальных штыревых разъёмов, и это позволяет самостоятельно проектировать сложные технические устройства [39].

Arduino это открытая система. В интернете опубликованы и ее принципиальные схемы, и ее программное обеспечение находится полностью в свободном доступе.

Рисунок 8 – Интегрированная среда разработки Arduino

При выборе способа передачи данных, были рассмотрены два варианта: проводной и беспроводной. Как правило, необходимость монтажа системы «Умный дом» редко возникает в начале проведения ремонтных работ в помещении. Зачастую монтаж «умного дома» осуществляется тогда, когда помещение уже полностью отремонтировано.

В этой ситуации прокладка новых проводов даже если и возможна, то проблематична и трудоемка. Поэтому необходимо сделать структуру системы «Умный дом» максимально гибкой и в любой ситуации доступной

для пользователя. В результате данных рассуждений был выбран беспроводной способ передачи данных.

Существуют различные технологии беспроводной передачи данных, такие как ZigBee, BlueTooth, Wi-Fi и т.д. [40]. Было принято решение на создание беспроводной локальной сети на основе технологии Wi-Fi.

Для платформы Arduino существуют специальные платы расширения Wi-Fi, так же на рынке представлены Wi-Fi чипы (например ESP8266), которые можно подключить к Arduino и осуществить совместную работу. Однако подобные решения существенно увеличивают стоимость проекта, усложняется монтаж и сам процесс программирования.

В процессе анализа решений приоритет был отдан Arduino совместимым платам для домашней автоматизации и интернета вещей производства WeMos на чипе ESP8266EX, представленный на рисунке 9.

Рисунок 9 – Плата WeMos на чипе ESP8266EX

Плата построена на основе Wi-Fi модуля ESP-12E. ESP-12E – это одна из последних модификаций модулей ESP8266, различающихся количеством

выводов и вариантами исполнения. И это не простой Wi-Fi модуль, а полноценный 32 битный микроконтроллер ESP-8266EX, в состав которого входит набор интерфейсов ввода/вывода (SPI, UART, I2C). При этом сама схема модуля состоит из минимального количества необходимых деталей: самого чипа ESP8266, flash памяти, кварца.

Плата WeMos D1 R2 имеет формфактор Arduino Uno и работает в большинстве случаев как UNO, но имеет одно огромное преимущество над стандартным Arduino – она обладает встроенным Wi-Fi. Это позволяет создавать умные устройства, управляемые с любого смартфона, планшета или ноутбука, имеющего встроенный модуль Wi-Fi.

Таким образом, в помещении каждой группой объектов будет управлять плата WeMos D1. Она будет считывать информацию с датчиков и передавать их в локальную сеть, и принимать различные команды управления из локальной сети, обрабатывать, и подавать выходной сигнал на исполнительные механизмы.

Однако конечный пользователь может установить большое количество групп управления, и будет проблематично управлять в локальной сети каждой группой в отдельности. К этому может добавиться необходимость ведения статистики больших объемов данных, что будет сложно реализовать на микроконтроллере, поскольку микроконтроллеры применимы там, где нужна не мощность процессора, а, скорее нужен баланс между ценой и достаточной функциональностью [41].

Универсальное решение данной проблемы – установка в локальной сети web-сервера на персональном компьютере или ноутбуке. WeMos платы будут передавать GET запросы на сервер, в которых будет содержаться текущая информация о состоянии контролируемых объектов.

В свою очередь сервер будет принимать эти запросы, обрабатывать и сохранять в базе данных. Кроме того, на сервере будет реализован пользовательский интерфейс, в котором будет отображаться состояние

объектов, строиться статистика, и расположены экранные кнопки управления.

При активации пользователем экранной кнопки, сервер будет генерировать GET запрос команды управления и передавать его на плату WeMos.

Сам сервер будет реализован на Apache HTTP Server и PHP. Именно с помощью языка PHP будет выполнена описанная выше функциональность. Данные будут храниться в базе данных mysql. Данное программное обеспечение является бесплатным, поэтому его использование не потребует дополнительных затрат от конечного пользователя. Общая стандартная структура работы подобной системы представлена на рисунке 10.

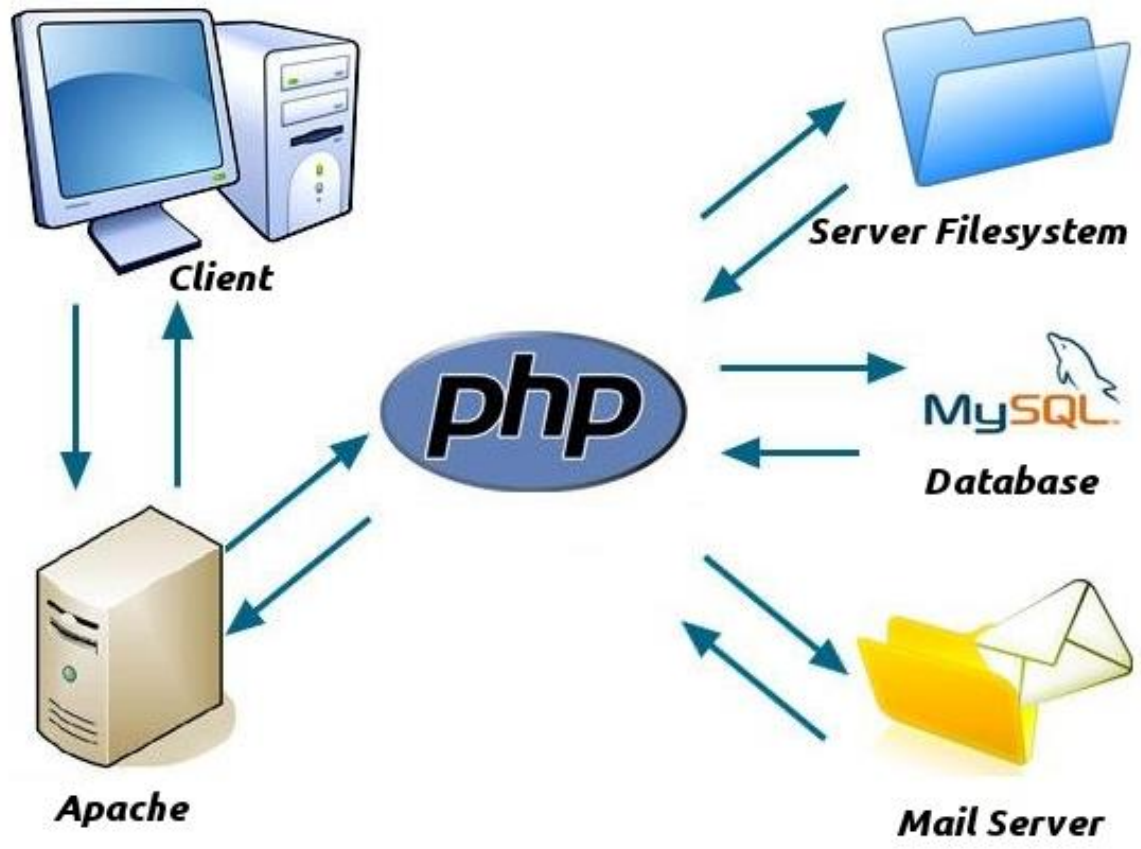


Рисунок 10 – Стандартная структура работы связки Apache, PHP и MySql

Для того, что бы получить доступ к серверу и, соответственно, к управлению системой «Умный дом», пользователю будет достаточно подключиться к локальной сети помещения. Тут стоит уделить внимание безопасности. Внутри помещения к локальной сети можно подключиться как через провод, так и через Wi-Fi. В первом случае безопасность обеспечивает сам способ подключения, во втором – Wi-Fi аутентификация.

Но для выполнения некоторых потребностей пользователя может потребоваться удаленное управление через интернет. В этом случае, для обеспечения безопасности, подключение будет происходить с помощью виртуальной частной сети VPN и ее аутентификации. Конечная структура системы «Умного дома» представлена на рисунке 11.

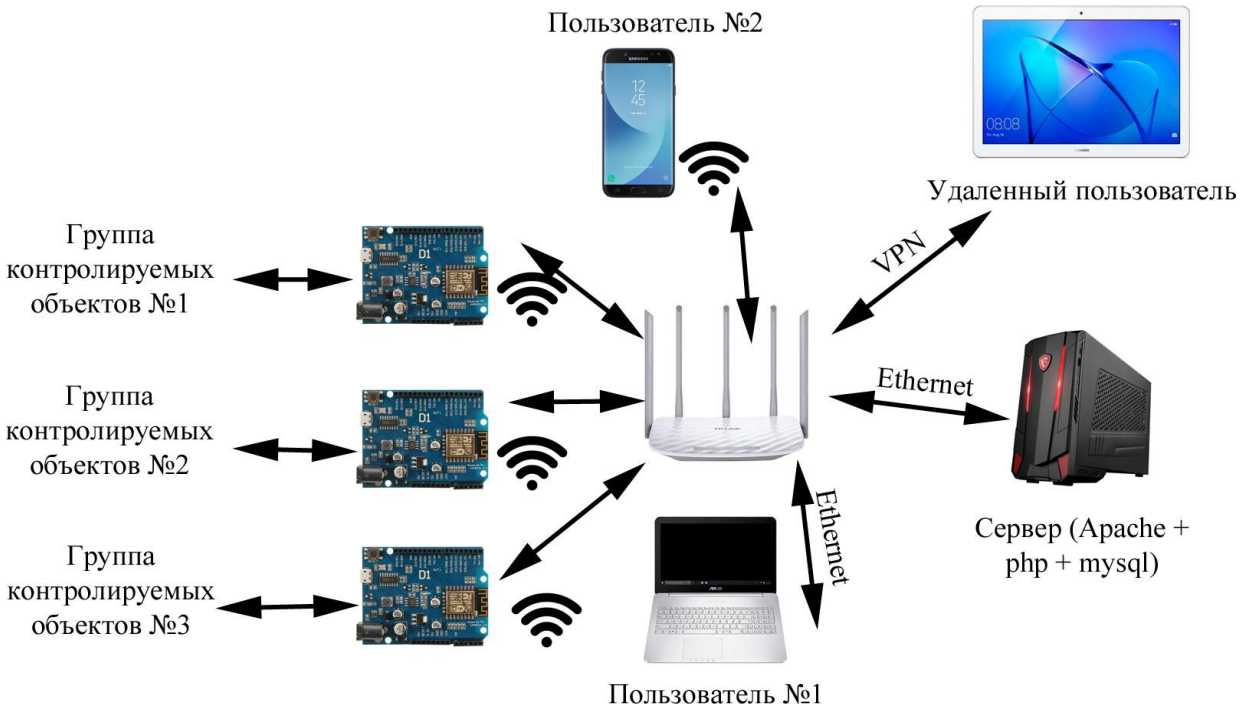


Рисунок 11 – Структура системы сбора информации с датчиков

Из недостатков описанной системы можно выделить следующий: в конечном итоге система управления получилась все таки централизованной, построилась на основе центрального сервера. В случае выхода из строя

сервера, система перестанет функционировать. Следовательно, получившийся структура нуждается в доработке.

2.2 Проект научного эксперимента по тестированию системы

В процессе реализации Автоматизированной Системы Управления Зданием (АСУЗ) необходимо разработать систему сбора информации с датчиков в единую СУБД. Целью эксперимента будет получить и отладить механизм беспроводного сбора, сохранения и отображения данных.

Будет использовано следующее оборудование:

– arduino совместимые платы для автоматизации и интернета вещей производства WeMos на чипе ESP8266EX (32 битный микроконтроллер ESP-8266EX со своим набором интерфейсов ввода/вывода, в том числе SPI, UART, I2C, построена на основе Wi-Fi модуля ESP-12E [42];

– датчики температуры и влажности DHT11;

– Wi-Fi роутер для организации беспроводной локальной сети;

– ПК с предустановленным программным обеспечением (Apache HTTP Server, PHP, СУБД MySQL).

В процессе эксперимента будет разработана:

– программа на языке PHP, которую установим на Web сервер. Программа будет состоять из двух модулей, первый принимает защищенные паролем GET запросы, обрабатывает и сохраняет результат в базе данных, второй отображает на экране пользователя сохраненную информацию с датчиков из базы данных в виде, понятном для человека;

– прошивка для платы WeMos, алгоритм которой будет подключаться к локальной сети по Wi-Fi, а затем с интервалом в 60 секунд опрашивать

датчик температуры и отправлять GET запросом результат на Web сервер;

– система будет запущена и протестирована на надежность.

2.3 Методика проведения экспериментальных исследований

Экспериментальные исследования начнем с разработки программного алгоритма, принимающего и обрабатывающего GET запросы. Т.к. полученные данные сохраняются в СУБД MySQL необходимо создать саму базу данных, а в ней таблицу, структура которой представлена в таблице 1.

Таблица 1 – Таблица полученных показаний датчика температуры

Имя поля	Тип данных	Описание
t_id	Счетчик	Идентификатор данных.
temp	Числовой	Полученная температура
time	Числовой	Отметка времени обработки показаний на сервере
sensor	Числовой	Номер датчика температуры

Для того чтобы создать БД с таблицей в файле setup.php напишем скрипт, который сначала проверит на существование базу данных, а в случае его отсутствия сначала выполнит SQL запрос «CREATE DATABASE ESP_DATA CHARACTER SET utf8 COLLATE utf8_general_ci» а после этого запрос «CREATE TABLE temp (t_id int(11) unsigned NOT NULL auto_increment, temp int(11) unsigned NOT NULL, time int(11) unsigned NOT NULL, sensor int(11) unsigned NOT NULL, PRIMARY KEY (t_id)) AUTO_INCREMENT=1». Полная версия алгоритма создания БД представлена в приложении А. Результат выполнения алгоритма представлен на рисунке 12.

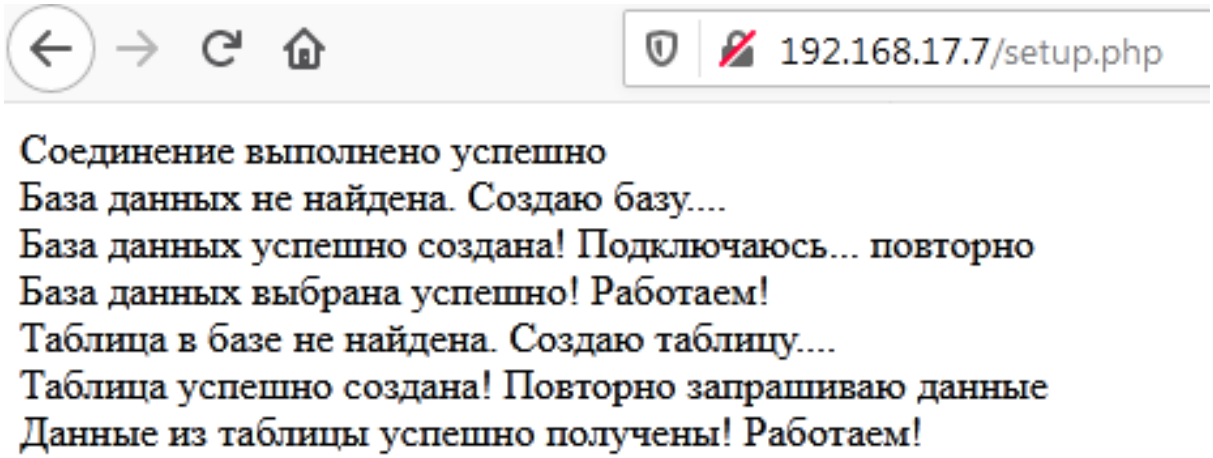
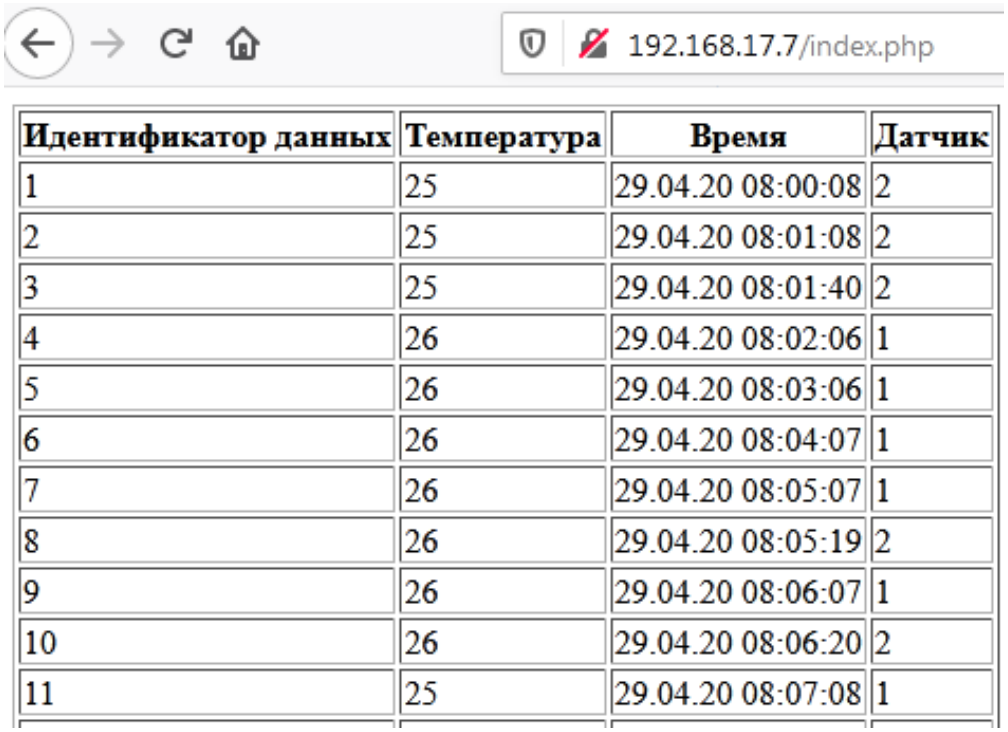


Рисунок 12 – Создание базы данных с таблицей

Все настройки для подключения к серверу баз данных (хост, имя пользователя, пароль) вынесены в отдельный файл config.php, который представлен в приложении Б.

Поскольку программный код подключения к серверу баз данных впоследствии будет использоваться в начале каждого программного алгоритма, его так же вынесем в отдельный файл connect.php, который представлен в приложении В. Затем перейдем к созданию файла add.php (приложение Г), считывающего данные из GET запроса, проводящего проверку корректности пароля доступа в запросе, и, в случае успешной проверки, выполняющего SQL запрос добавления данных в таблицу temp.

На завершающем этапе разработки алгоритма обработки GET запросов, напишем программный код, выводящий на экране монитора содержимое таблицы базы данных. Для корректного отображения кодировки создадим конфигурационный файл веб-сервера .htaccess с содержимым: «AddDefaultCharset utf-8», а после этого в файле index.php выполним подключение к БД, запросим содержимое таблицы и выведем ее на экран. Программный алгоритм представлен в приложении Д. Результат выполнения алгоритма представлен на рисунке 13.



Идентификатор данных	Температура	Время	Датчик
1	25	29.04.20 08:00:08	2
2	25	29.04.20 08:01:08	2
3	25	29.04.20 08:01:40	2
4	26	29.04.20 08:02:06	1
5	26	29.04.20 08:03:06	1
6	26	29.04.20 08:04:07	1
7	26	29.04.20 08:05:07	1
8	26	29.04.20 08:05:19	2
9	26	29.04.20 08:06:07	1
10	26	29.04.20 08:06:20	2
11	25	29.04.20 08:07:08	1

Рисунок 13 – Результат вывода содержимого таблицы базы данных

Для отладки работы алгоритма, самостоятельно напишем и отправим GET запрос (http://192.168.17.7/add.php?temp=30&sensor=0&pass=qwerty) через адресную строку браузера. В случае успешной работы, перейдем к следующему этапу эксперимента – создание прошивки для платы WeMos.

Подобная прошивка представлена в источнике [43], следовательно, ее необходимо адаптировать под текущий эксперимент. Итоговый вариант прошивки представлен в приложении Е. Алгоритм работы прошивки следующий: подключаются библиотеки для работы с чипом ESP8266 и с датчиком температуры DHT, затем создается необходимые переменные и объекты, в процедуре настройки setup происходит подключение к Wi-Fi сети, и наконец, в циклической процедуре loop, с интервалом 50 секунд производится считывание показаний с датчика, формирование и отправка GET запроса. Поскольку выполнение последней операции занимает 10 секунд, каждое новое показание температуры в базе данных появляется с интервалом в 1 минуту.

2.4 Итоги эксперимента

Разработанная система сбора информации с датчиков в единую СУБД в ходе эксперимента показала отличные результаты. Возможные варианты продолжения исследований в этом направлении:

– провести эксперимент с имитацией различных сбоев (временная остановка WEB сервера, потеря сигнала Wi-Fi и т.д.), и усовершенствовать программные алгоритмы, что бы они автоматически корректировали логику работы в подобных ситуациях;

– увеличить количество датчиков и контролируемых единиц в системе в сумме до сотни единиц;

– добавить алгоритмы обработки полученной информации, автоматическое построение статистики.

3 Определение положения локального источника звука

3.1 Звук и оценка его восприятия

Нахождение источника звука в трехмерном пространстве имеет очень важное практическое значение. Необходимо увеличить возможности взаимодействия роботов с человеком, и прочими объектами. Функция определения положения источника звука так же особо применима в области автоматизации, машиностроении и в обычной повседневной бытовой жизни человека [44].

Также данная область применима и в системе «Умного дома» например, для своевременного включения исполнительных механизмов (открытие окон и дверей) и выполнения функций систем безопасности.

Как физическое явление, звук это колебательные движения любого тела, которые передаются окружающему воздуху, вызывая в нем последовательные сгущения и разрежения его частиц, которые распространяются в виде продольной звуковой волны [44]. Органы чувств человека, в частности уши, воспринимают эти волны, и результатом этого восприятия является звук. Субъективным восприятием силы звука называют громкость звука. Самым оптимальным способом будет использование физической характеристики громкости звука, которая называется уровнем звукового давления. Давление, которое возникает в газообразной или жидкой среде во время прохождения через нее измеряемых в Па звуковых волн, называется звуковым давлением. Для того что бы произвести расчеты более удобно вводим понятие «уровень звукового давления». Это давление измеряется в дБ, и представляет собой значение звукового давления, которое было измерено по относительной школе, и отнесено к опорному давлению 20 мкПа.

Мозг человека анализирует сдвиг фаз звуковых колебаний в ушах и различает направление прихода звуковых сигналов в горизонтальной плоскости. Это называется бинауральный. Когда сигнал приходит в оба уха одновременно, то мозг делает выводы, что источник звука находится строго перед ним, либо строго позади него. Если есть смещение источника звука от центра головы, как это представлено на рисунке 14, то фиксируется какое ухо раньше обнаружило звуковой сигнал, и производится анализ.

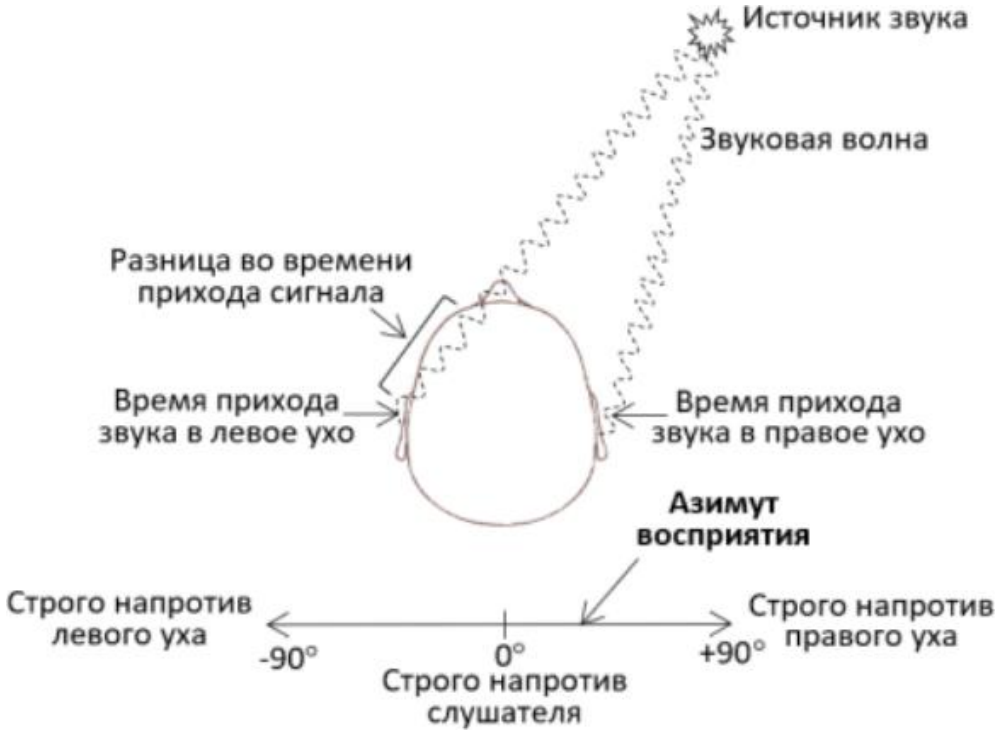


Рисунок 14 – Механизм локализации в горизонтальной плоскости по разности во времени

Обратим внимание на то, что мозг анализирует только источники звука в диапазоне частот от 300 Гц до 1 кГц. Если частота выше 1000 Гц и больше, то тогда анализируется громкость звука. Но такие большие волны имеют свойство очень быстро затухать, и как следствие в ушах слушателя интенсивность сигналов отличается. Последнее, используя разницу амплитуд, позволяет мозгу вычислять направление прихода сигнала. Как представлено на рисунке 15, источник звука находится с той стороны, с которой лучше слышан звук.

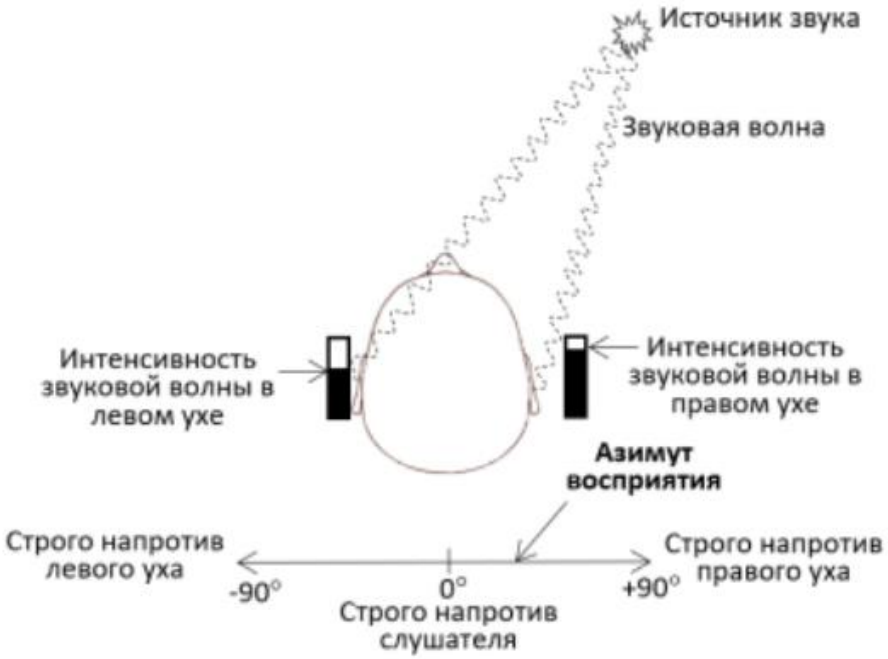


Рисунок 15 – Механизм локализации в горизонтальной плоскости по разности амплитуд

3.2 Математический расчет координат источника звука

В основу многих алгоритмов акустической локализации входит следующий принцип: звук от источника достигает каждого датчика звука (микрофона) за разное время. Если измерить задержку по времени сигнала для всех датчиков, то с помощью этих данных можно рассчитать местоположение источника звука. Представим, что доступно два датчика звука. Источник звука достигнет первого датчика в момент времени t_1, а второго в момент времени t_2. Относительная временная задержка составляет $\tau = t_1 - t_2$. При правильном расчете времени τ, получится такая точка пространства, что $t_1 - t_2. = \tau$. В этой точке и находится источник звука.

Для нахождения координат источника звука существуют два метода. Первый – триангуляция, это когда известны угловые координаты источника звука относительно микрофонов на плоскости. Второй, уже описанный ранее, трилатерация, это когда известно расстояние от источника звука до нескольких микрофонов [45].

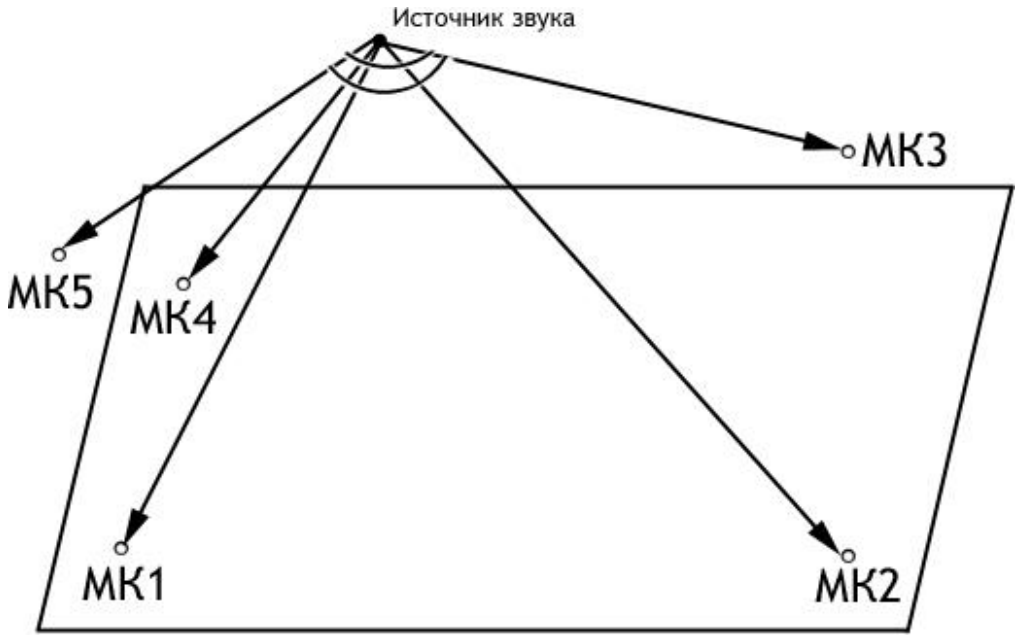


Рисунок 16 – Пример расположения источников звука и микрофонов в пространстве

Для определения используется информация о временных задержках между парами микрофонов, исходя из того, что координаты каждого микрофона в пространстве известны. Точность алгоритма повышается путем увеличения числа микрофонов и расстояния между ними.

Обозначим координаты:

– I={x;y} –источник звука;

– $M_i=\{x_i;y_i\}$ – однин из микрофонов, где i=1..n.

Расстояние от источника звука до i-го микрофона рассчитывается по формуле [46]:

$$R_i = \sqrt{(x_i - x)^2 + (y_i - y)^2 + (z_i - z)^2} \quad (1)$$

Время распространения звука t_i от источника звука до микрофона i определяется скоростью звука ($V_{зв}$ = 340.29 м/с при нормальных условиях):

$$t_i = \frac{R_i}{V_{ЗВ}} \quad (2)$$

Между двумя микрофонами будет следующая разность:

$$t_{ij} = \frac{R_i - R_j}{V_{ЗВ}}, \quad (3)$$

Так же для каждой пары микрофонов должно выполняться условие:

$$R_i - R_j - V_{ЗВ} \times t_{ij} = 0, \quad (4)$$

Но если действительно получится равенство нулю, это будет означать, что погрешность измерений отсутствует.

Следовательно, для всех попарно сработавших микрофонов можно записать данное соотношение через сумму:

$$\Phi_1 = \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} (R_i - R_j - V_{ЗВ} \times t_{ij}), \quad (5)$$

Эта сумма должна стремится у нулю.

В финальных расчетах добавим знакопостоянный функционал оценки суммарной погрешности:

$$\Phi_2 = \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} (R_i - R_j - V_{ЗВ} \times t_{ij})^2, \quad (6)$$

Однако в текущей ситуации данные о координатах источника звука отсутствуют. Следовательно, осуществим подбор этих координат, а критерием правильности подбора считается минимизация функционала Φ_2. Следовательно, необходимо найти минимум этой функции:

$$\Phi_2 = \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} (\sqrt{(x_i - x)^2 + (y_i - y)^2 + (z_i - z)^2} - \sqrt{(x_j - x)^2 + (y_j - y)^2 + (z_j - z)^2} - V_{ЗВ} \times t_{ij})^2 \quad (7)$$

Для этого на следующем этапе разработаем алгоритм. Входными данными алгоритма будут числа, которые являются временными задержками попарно сработавших микрофонов. Так же на вход алгоритма поступят и сами координаты микрофонов. И для полученного функционала будет произведен поиск координат минимума (x, y, z).

4 Разработка системы определения положения локального источника звука

4.1 Система акустической локализации

В основу системы акустической локализации заложим разработанную ранее систему сбора информации с датчиков в единую СУБД. Вместо датчиков температуры и влажности DHT11 будем использовать датчик звука KY-037, представленный на рисунке 17.

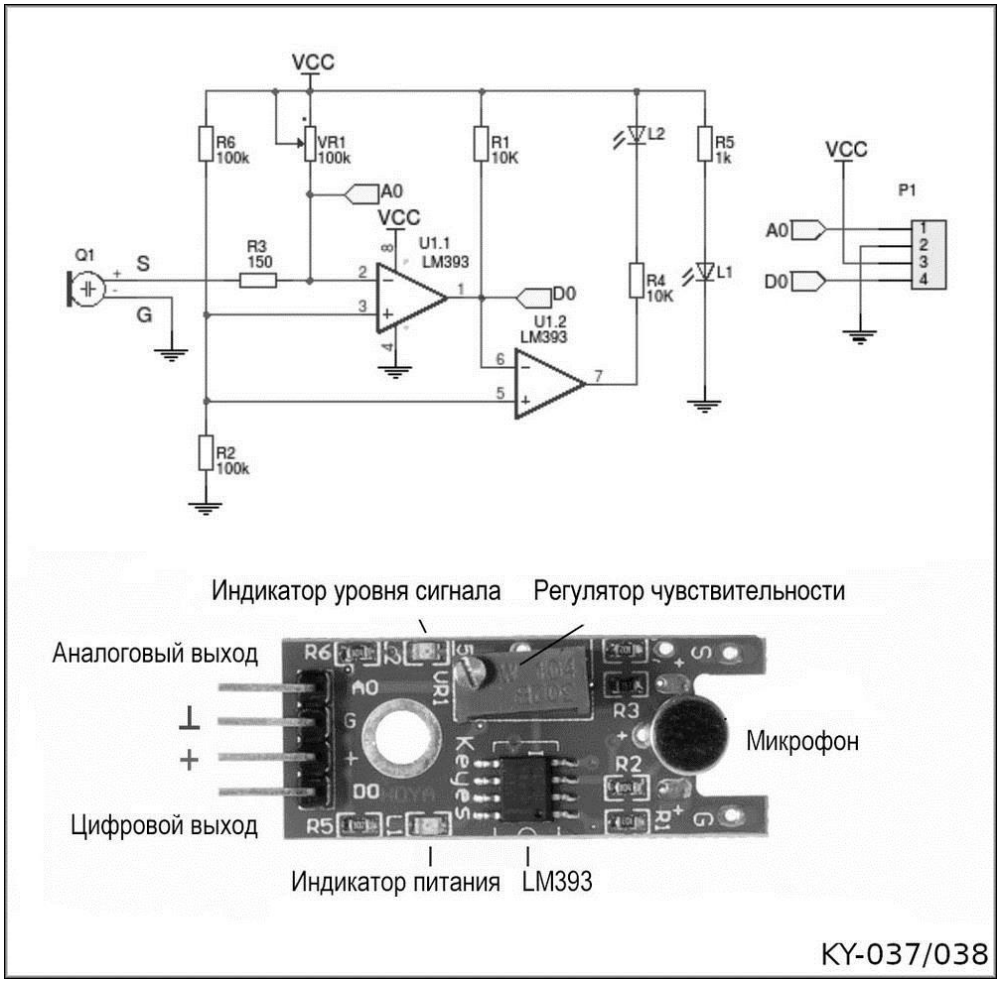


Рисунок 17 – Датчик звука KY-037

На рисунке 18 представлена принципиальная схема подключения датчика звука к плате Wemos, а на рисунке 19 показана логика взаимосвязи компонентов разрабатываемой системы.

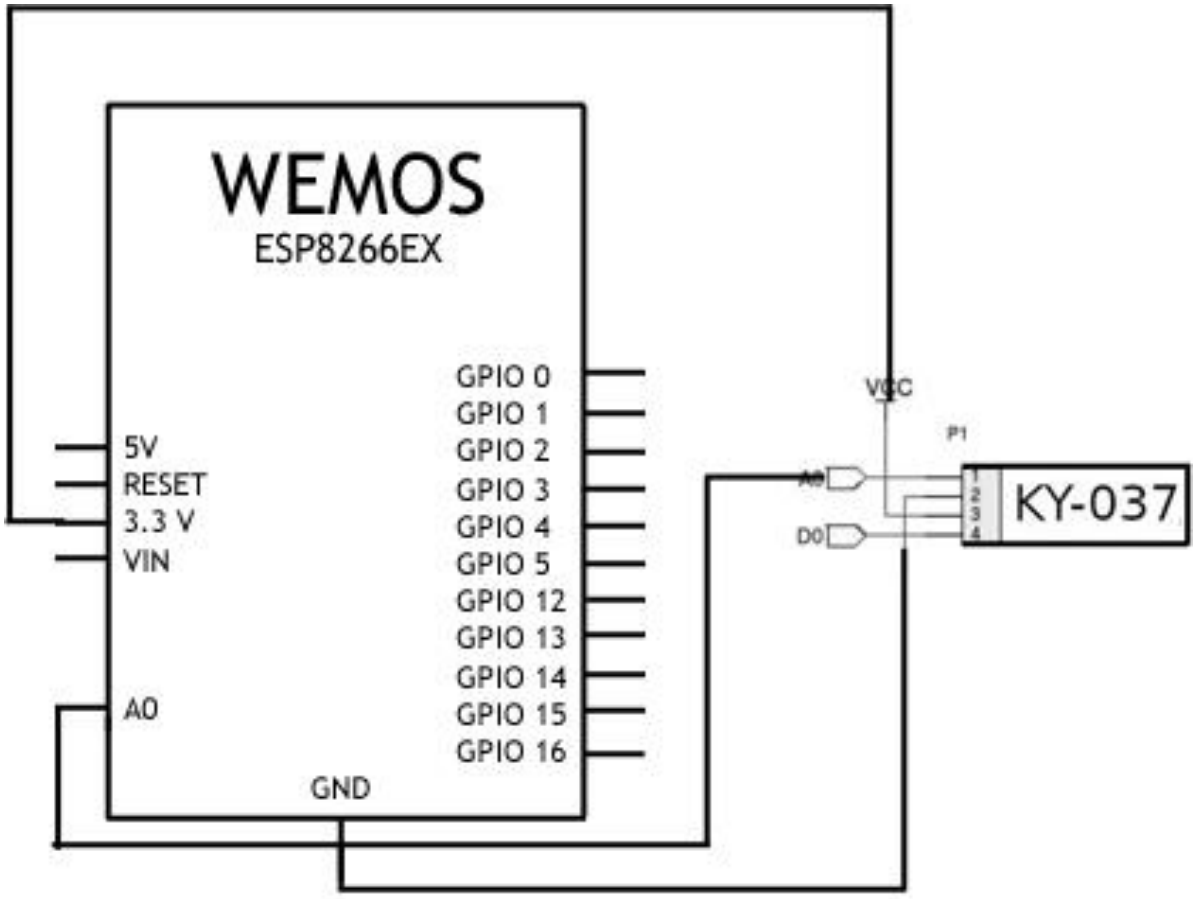


Рисунок 18 – Принципиальная схема подключения датчика звука в плате Wemos

Принцип работы системы состоит в следующем: при запуске платы Wemos подключаются к роутеру, синхронизируются через интернет по протоколу сетевого времени NTP, и подают световую индикацию об успешно выполненной синхронизации. После этого микроконтроллер начинает постоянно опрашивать датчик звука. Это единственная операция, которую выполняет микроконтроллер. И, как показано, на схеме ниже, на один микроконтроллер приходится один датчик. Из результатов работ в источнике [47] мы видим, что успеть уловить звук является достаточно сложной

задачей. Если микроконтроллер будет выполнять какие либо другие задачи, он может пропустить звук, и не обработать сигнал. Например, пока будем опрашивать один микрофон, звук в это время будет на другом.

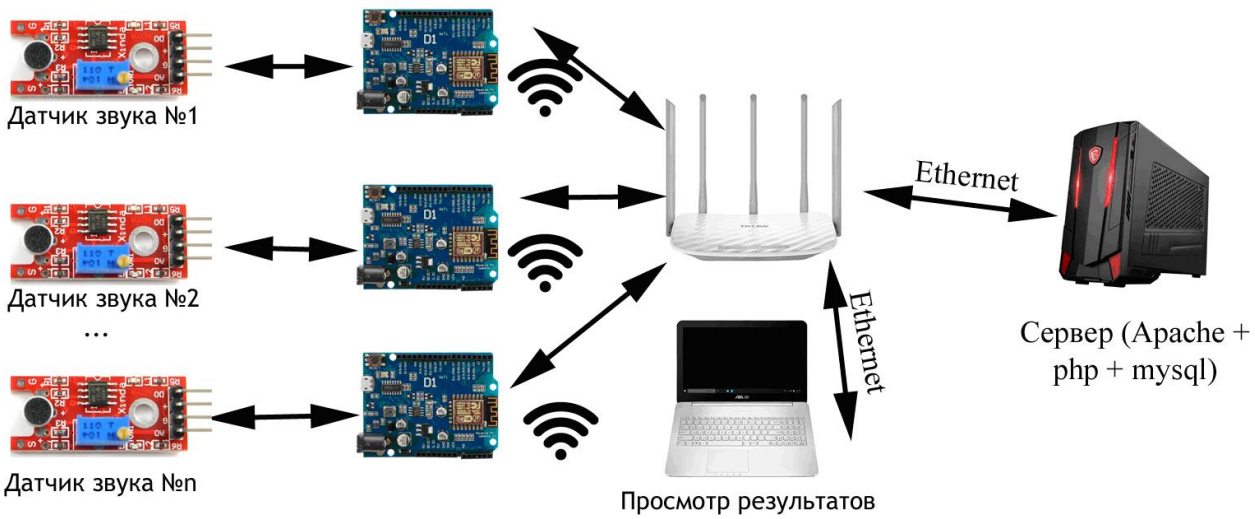


Рисунок 19 – Логика взаимосвязи компонентов системы акустической локализации

Как только происходит срабатывание датчика звука, микроконтроллер формирует GET запрос и отправляет его на сервер. GET запрос включает в себя следующие параметры:

– unix время, в которое произошла синхронизация платы с сервером NTP;

– время работы микроконтроллера, в момент синхронизации;

– время работы микроконтроллера в момент срабатывания датчика;

– номер датчика;

– пароль доступа в систему.

Итоговая прошивка платы, в которой представлен описанный выше алгоритм, представлен в приложении Ж.

С помощью первых трех параметров сервер рассчитает точное время срабатывания датчика. В свою очередь сервер принимает GET запрос, и

сохраняет полученный результат в СУБД MySQL. В последствии специальная программа на сервере анализирует полученные значения, и рассчитывает местоположение источника звука в пространстве.

Проведем следующие исследования: возьмем помещение (комнату) размером 2,5 м X 2,5 м, и расположим в ней на одной высоте 8 датчиков звука следующим образом (рисунок 20):

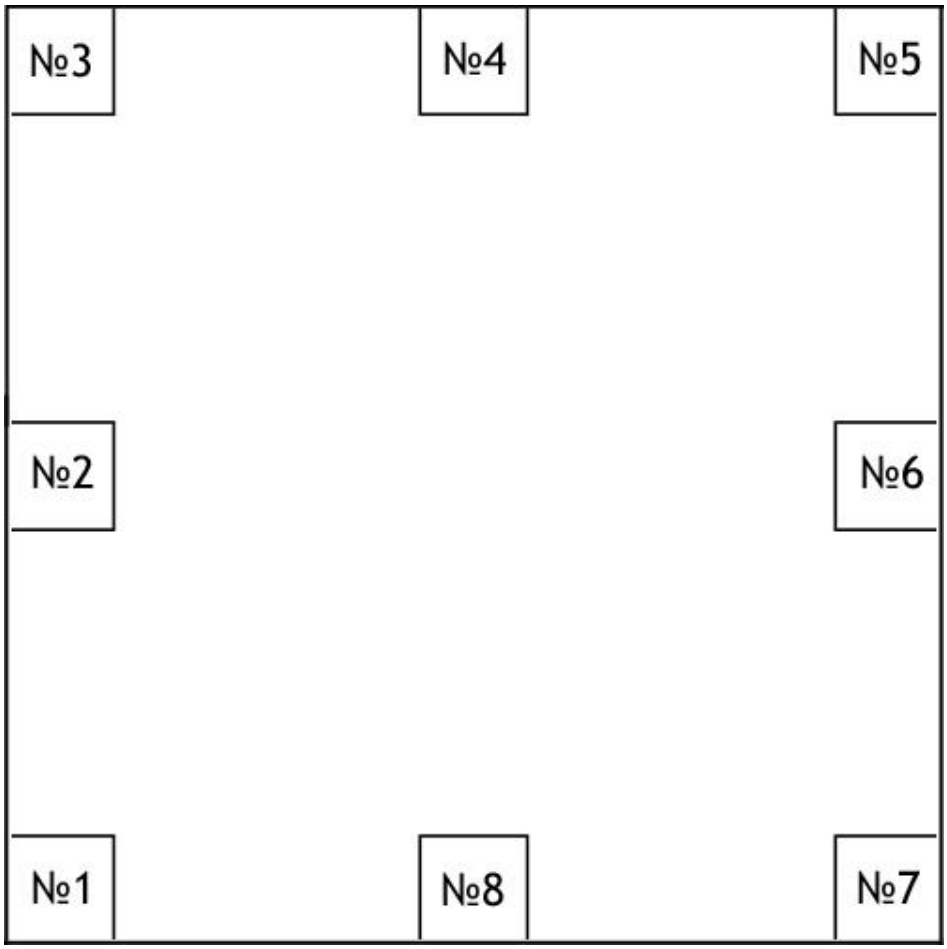


Рисунок 20 – Расположение датчиков в помещении

Для точности исследований, все время все датчики будут работать одновременно. Однако, рассчитывать координаты источника звука будем четырьмя разными способами.

Способ первый представлен на рисунке 21, в нем будут задействованы три датчика: №1, №4 и №7, образуя треугольник.

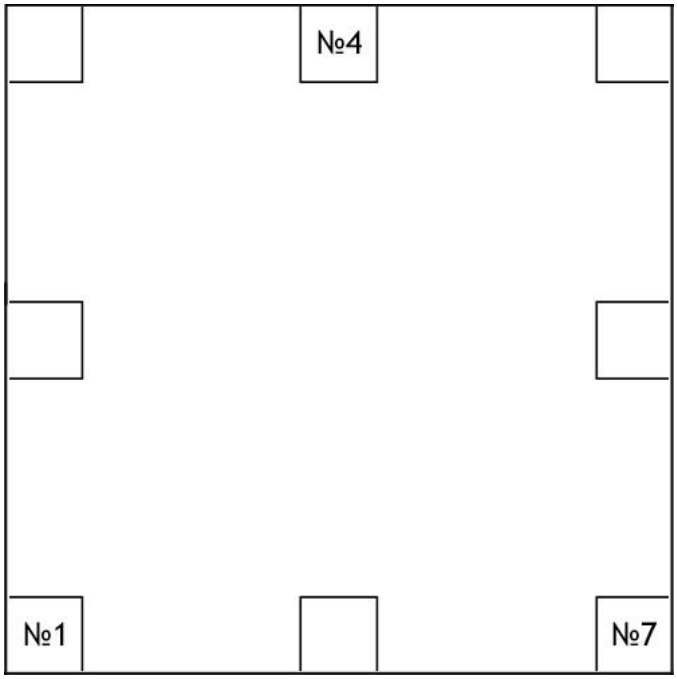


Рисунок 21 – Анализируемые датчики в способе №1

Во втором способе, представленном на рисунке 22, будет уже задействовано четыре датчика, находящиеся на углах: №1, №3, №5 и №7.

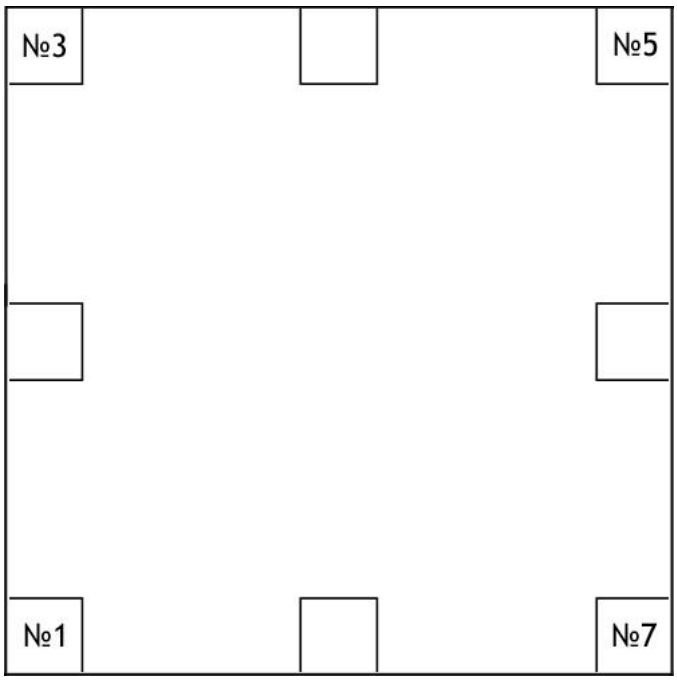


Рисунок 22 – Анализируемые датчики в способе №2

В третьем способе, представленном на рисунке 23, снова задействовано четыре датчика, но уже №2, №4, №6 и №8, которые находятся в середине сторон.

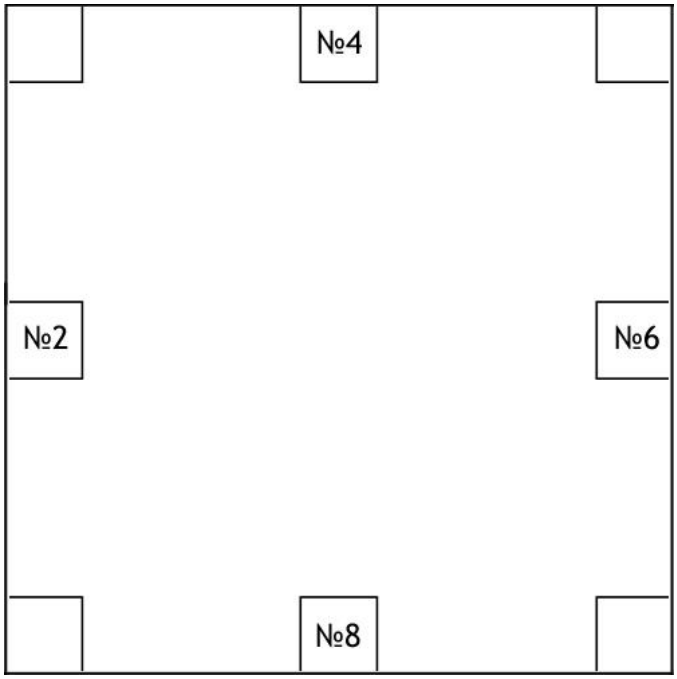


Рисунок 23 – Анализируемые датчики в способе №3

В четвертом способе, который не нуждается в иллюстрации, будут использованы все датчики одновременно.

4.2 Программа расчета местоположения источника звука

На данном этапе исследований разработана система, состоящая из восьми датчиков звука с Wi-Fi микроконтроллерами. Датчики фиксируют сигнал и отправляют GET запрос на сервер. Во втором разделе были разработаны алгоритмы, принимающие GET запрос, и сохраняющие результаты в системе управления базами данных. Используем разработки, и

сформируем на их основе структуру новой таблицы в базе данных. Результат представлен в таблице 2.

Таблица 2 – Таблица полученных показаний с датчиков звуков

Имя поля	Тип данных	Описание
s_id	Счетчик	Идентификатор данных.
epoch_last	Числовой	unix время, в которое произошла синхронизация платы с сервером NTP
synx	Числовой	время работы микроконтроллера, в момент синхронизации
millis	Числовой	время работы микроконтроллера в момент срабатывания датчика
sensor	Числовой	Номер датчика звука
time	Числовой	Отметка времени обработки показаний на сервере

Для того, что бы создать таблицу, напишем алгоритм в файле setup_z.php, который выполнит следующий запрос к базе данных: «CREATE TABLE sound (s_id int(11) unsigned NOT NULL auto_increment, epoch_last int(11) unsigned NOT NULL, synx int(11) unsigned NOT NULL, millis int(11) unsigned NOT NULL, sensor int(11) unsigned NOT NULL, time int(11) unsigned NOT NULL, PRIMARY KEY (s_id)) AUTO_INCREMENT=1». Полная версия алгоритма создания таблицы БД представлена в приложении И. Результат выполнения алгоритма представлен на рисунке 24.

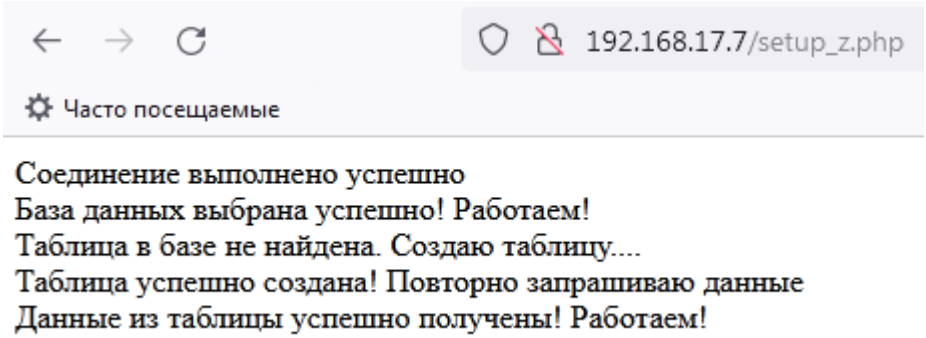


Рисунок 24 – Создание таблицы в базе данных

Далее разработаем файл add_z.php. Принцип его работы аналогичен файлу add.php, только вместо показаний температуры, он принимает параметры для расчета времени срабатывания датчика звука и сохраняет их в таблицу sound. Программный алгоритм представлен в приложении К.

Финальным этапом является сам алгоритм расчета местоположения источника звука, он представлен в приложении Л. В самом начале алгоритма указывается количество микрофонов, используемых для расчета, скорость звука при нормальных условиях и координаты микрофонов в пространстве. Затем алгоритм отправляет запросы в базу данных, и по очереди получает из базы значения для расчета момента срабатывания датчика звука, и сразу же производит расчет, сохраняя полученные значения в переменные. Далее, для формулы 7 расчета функционала, подсчитываются временные задержки между всеми парами микрофонов. Затем в алгоритме прописана функция подсчета функционала по выведенной формуле 7. Ну и наконец, производится поиск минимального значения функции, методом перебора. Мы находим источник звука в пространстве с точностью да 10 см, поэтому перебираем тремя вложенными циклами каждую координату от 0 до 2,5 с шагом 0,1, каждый раз вызывая созданную функцию и проверяя значение на минимум. В конце алгоритм выводит результат – координаты источника звука.

4.3 Регулировка датчика звука

В процессе подключения датчика звука KY-037 к микроконтроллеру, его необходимо отрегулировать с помощью регулятора чувствительности. Для этого была написана специальная прошивка, представленная в приложении М.

На вход микроконтроллера поступает напряжение от 0В до 3,3В, в программном коде эти значение интерпретированы от 0 до 1023. Показания неотрегулированного датчика звука представлены на рисунке 25.

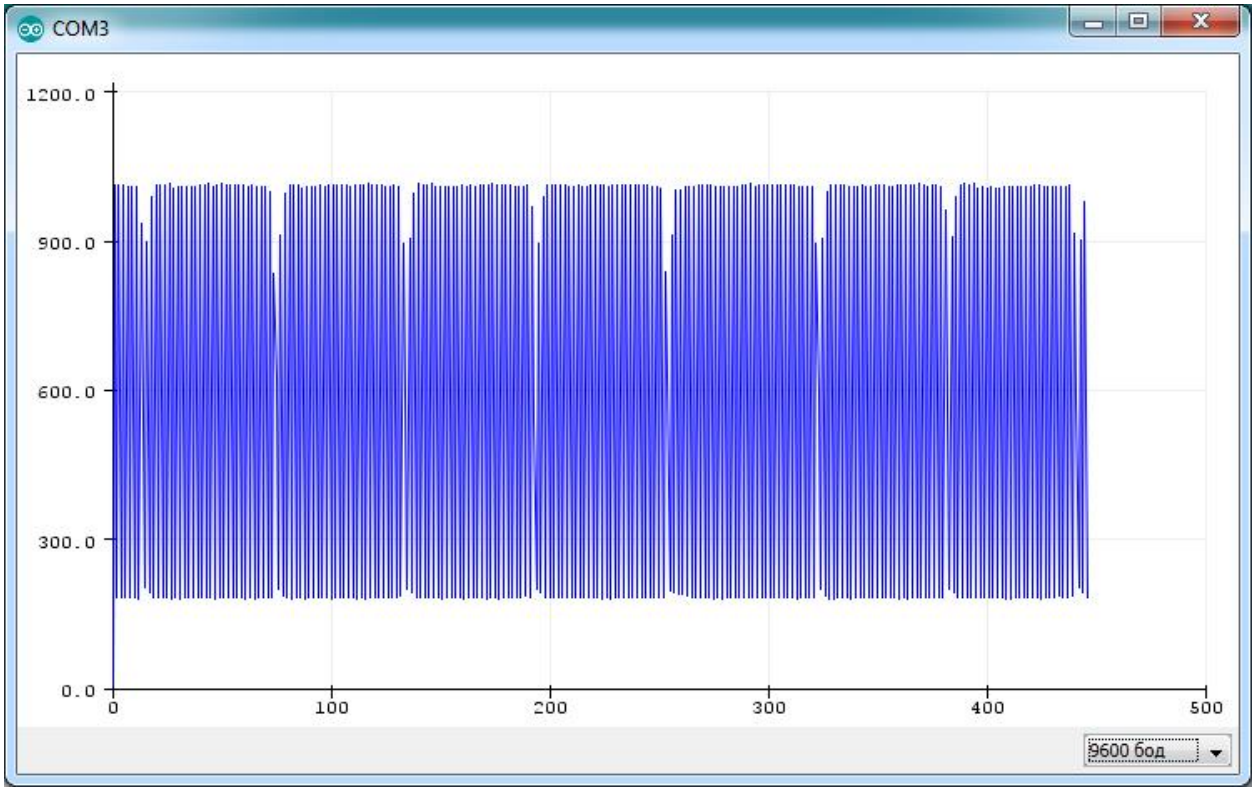


Рисунок 25 – Показания неотрегулированного датчика звука

Как видно из графика, когда датчик отрегулирован на более высокое выходное напряжение, он очень чувствительный, и срабатывает постоянно. При понижении чувствительности до сильно низкого выходного напряжения датчик срабатывает только при очень громком звуке вблизи него, что тоже не подходит для нашей системы. Экспериментальным путем, было установлено, что лучше всего для выполнения задачи научно-исследовательской работы датчик работает, когда отрегулирован на выходное значение около 330 единиц по программе Arduino IDE (рисунок 26).

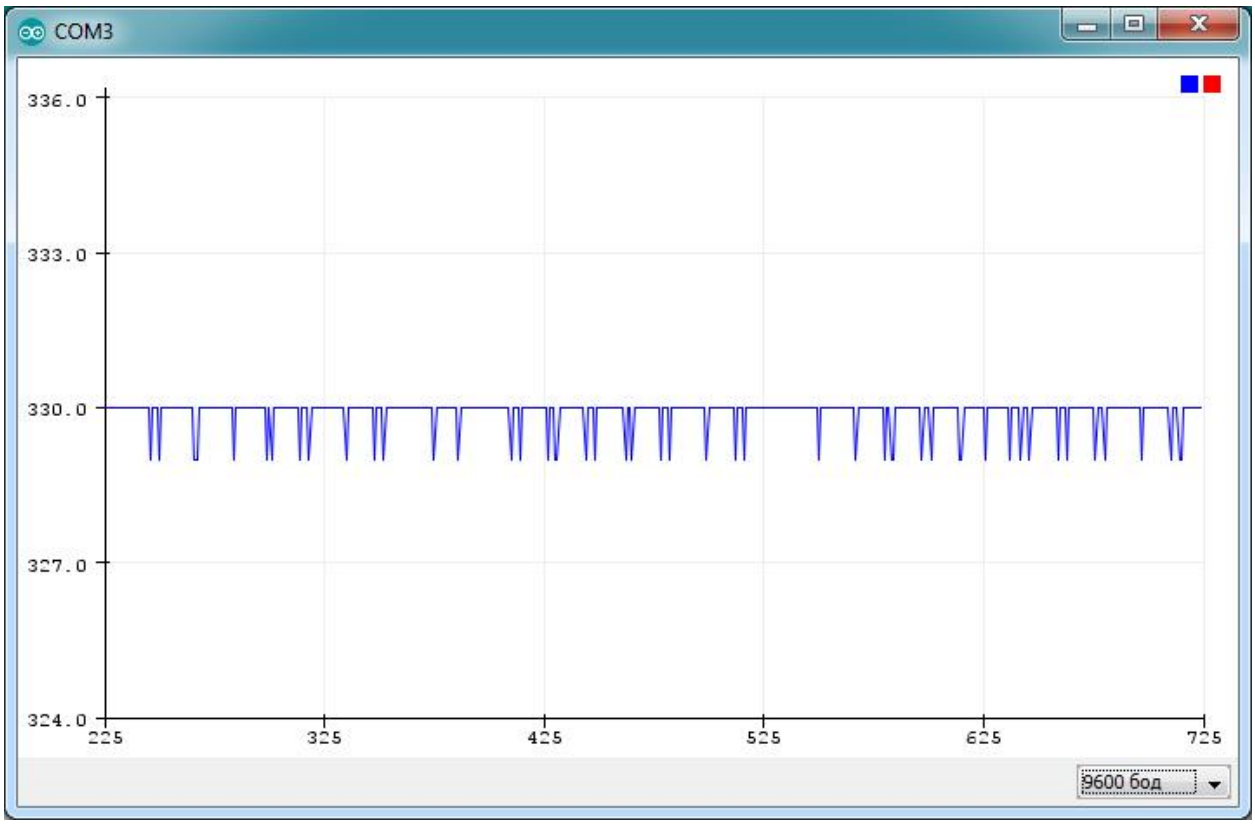


Рисунок 26 – Показания отрегулированного датчика звука

При данной регулировке в момент срабатывания датчика звука, его выходной сигнал в большинстве случаев сначала опускается ниже 310 единиц, а затем резко поднимается до 340 единиц, а после чего снова возвращается в состояние «покоя» (рисунки 27 и 28).

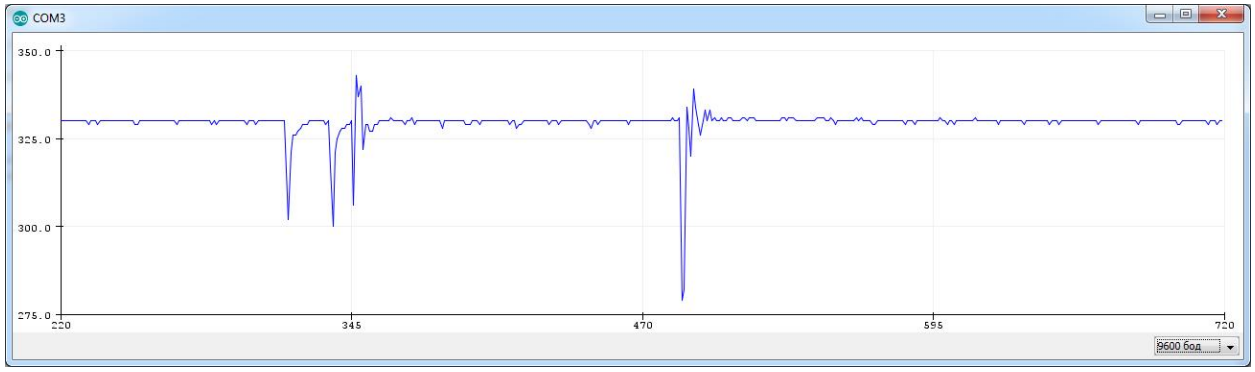


Рисунок 27 – Срабатывание датчика звука

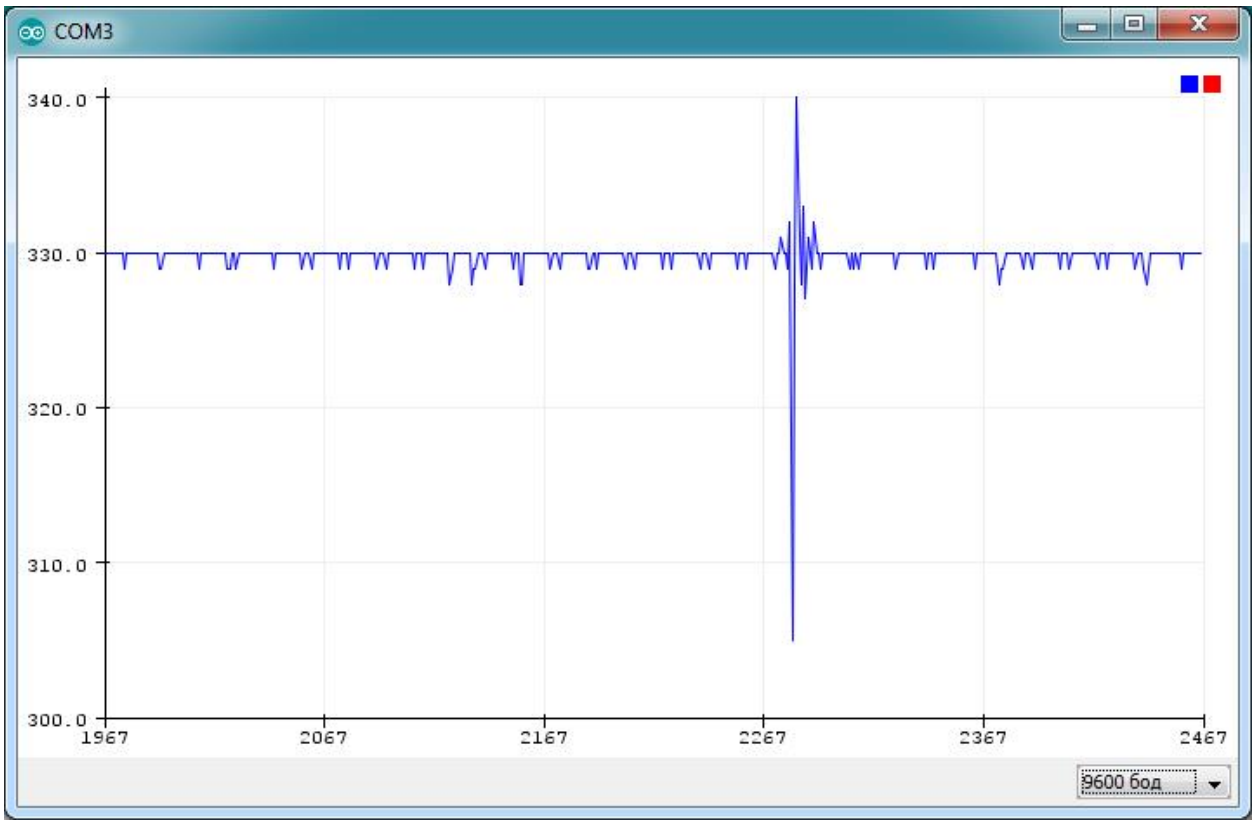


Рисунок 28 – Срабатывание датчика звука

4.4 Тестирование датчика на предмет внешний воздействий

Проверим полученную регулировку на предмет внешних воздействий на показания датчика (ложные срабатывания). На рисунке 29 представлен график, когда около помещения есть небольшие звуковые воздействия в виде разговора людей. Как видно из графика, датчик фиксирует разговоры, но значения не выходят за границы допустимого для срабатывания датчика (максимальное значение 328 единиц, минимальное 313 единиц). Это означает, что данное воздействие не вызывает ложных срабатываний датчика.

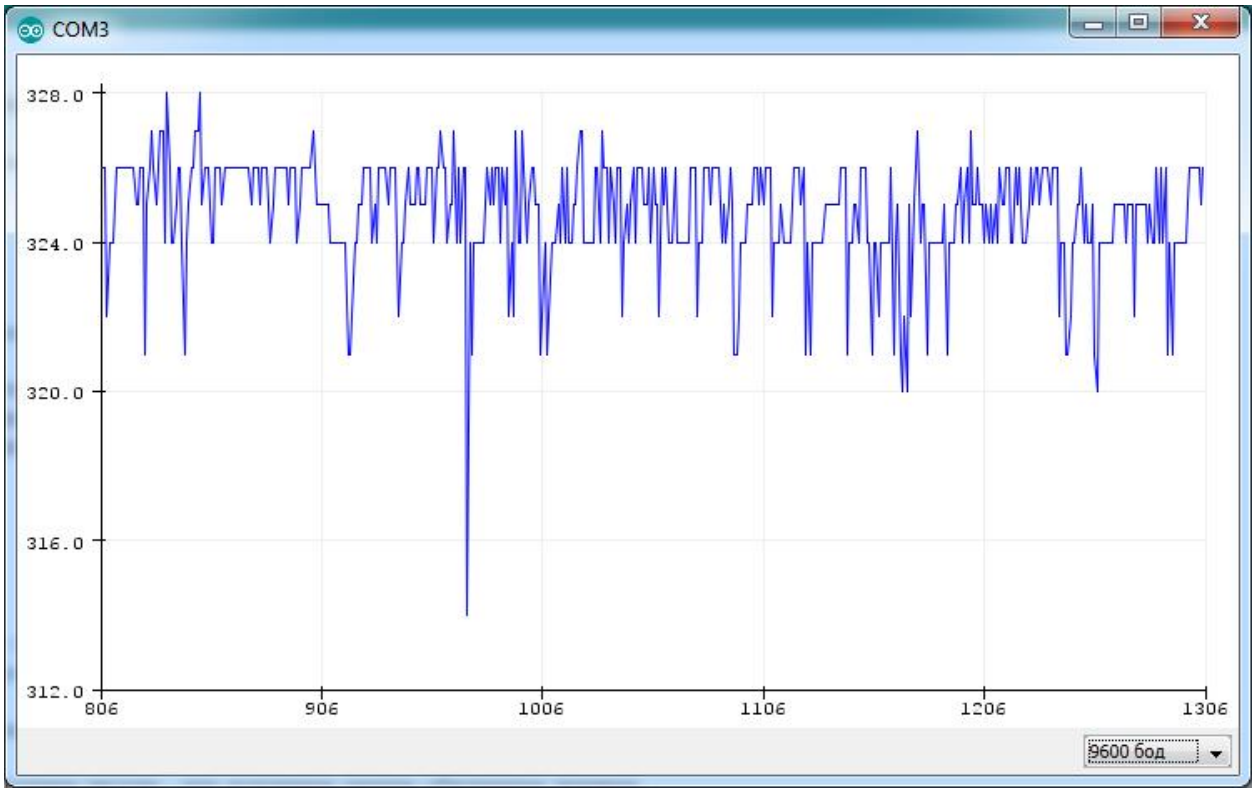


Рисунок 29 – Показания датчика звука во время разговора вне помещения

Следующее внешнее воздействие представлено на рисунке 30 – это выполнение ремонтных работ вне помещения, таких как сверление. Опять же, как видно из графика, датчик снова фиксирует звуковые колебания, но при этом, не переходя границы срабатывания. Максимальное значение 334 единицы, а минимальное 330 единиц.

Ну и наконец, третье воздействие – это музыка. Данное воздействие было разделено на два вида, первое – музыка без низких частот (рисунок 31), и музыка с низкими частотами (рисунок 32). Из графиков видно, что музыка без низких частот не вызывает отрицательного воздействия на датчик, а вот музыка с низкими частотами (басами), к сожалению вызывает у датчика ложные срабатывания. Подведя итоги, можно сделать выводы, что только одно из четырех внешних воздействий в процессе эксперимента вызвало ложное срабатывание, а это означает высокую надежность полученной системы.

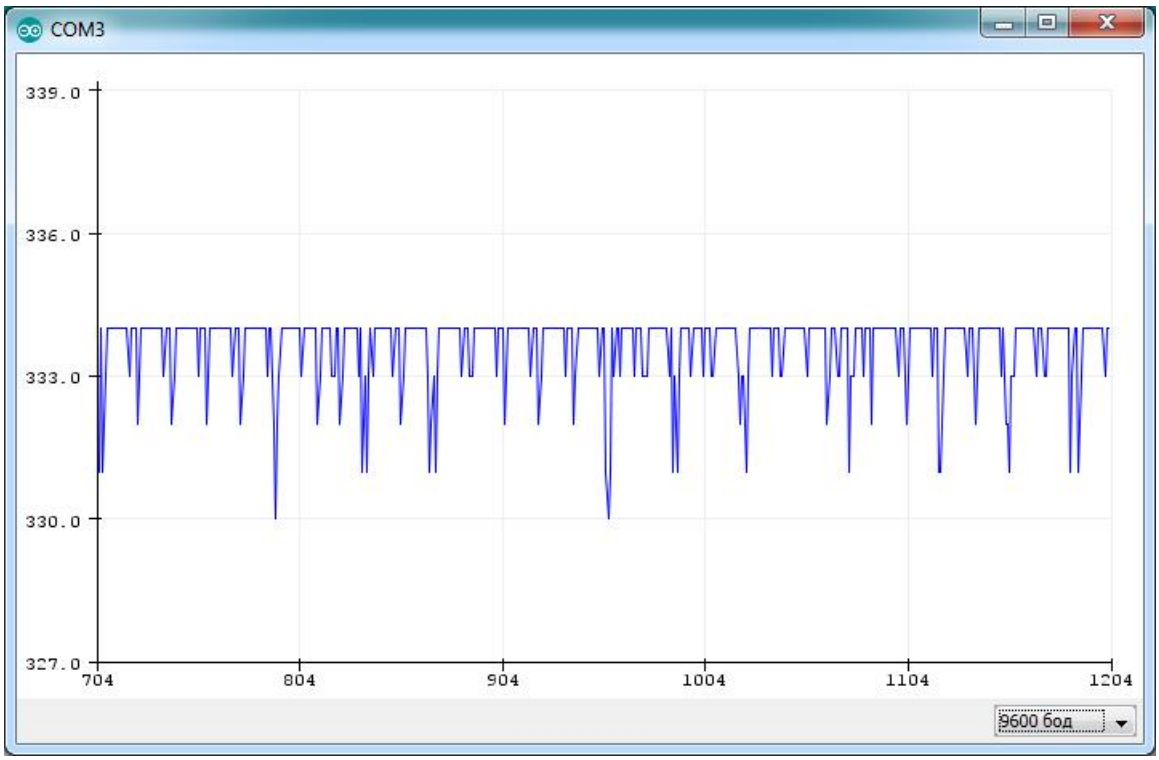


Рисунок 30 – Показания датчика во время ремонтных работ вне помещения (сверление)

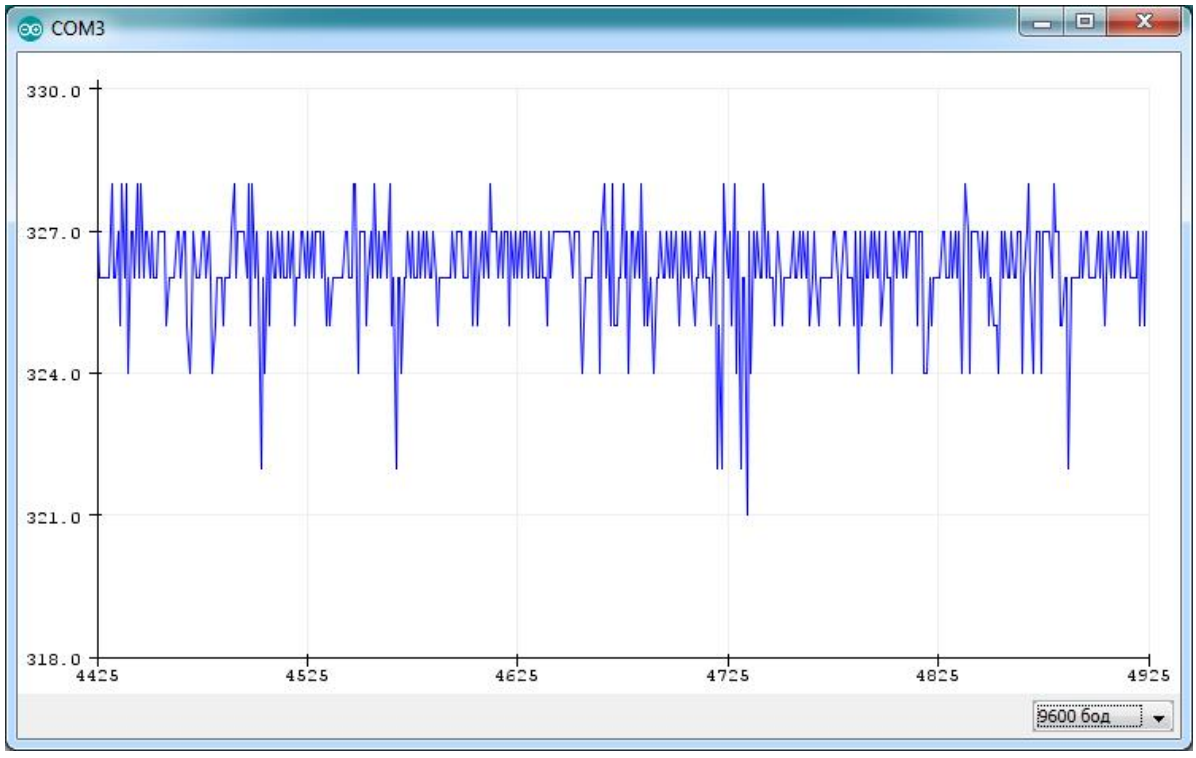


Рисунок 31 – Показания датчика звука при проигрывании музыки вне помещения без низких частот (басов)

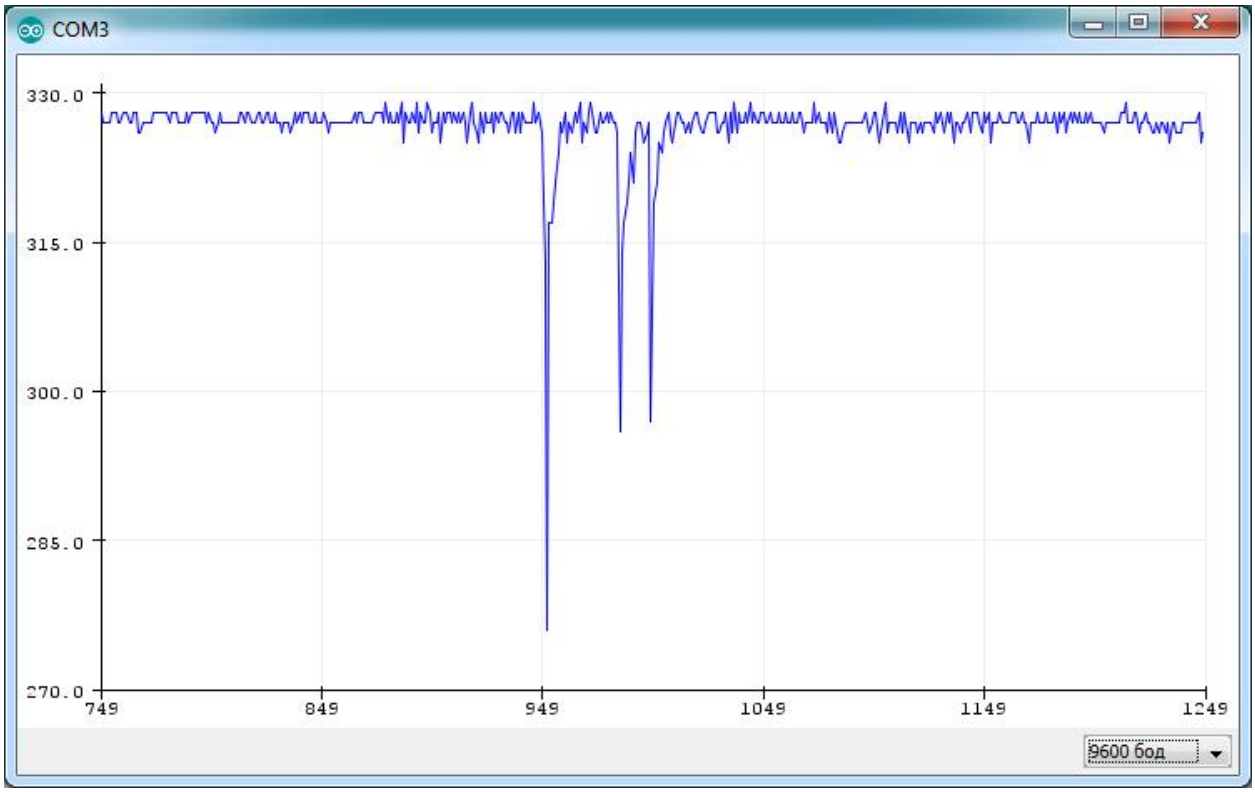


Рисунок 32 – Показания датчика звука при проигрывании музыки вне помещения с низкими частотами (басами)

4.5 Проведение итоговых исследований

Вернемся к исследованиям. Ранее была выполнена подготовка к исследованию, а именно в помещении размером 2,5 м X 2,5 м были расположены 8 датчиков звука (4 датчика расположены в углах, а еще 4 датчика расположены в середине сторон), подключенные к микроконтроллерам Wemos. Все датчики были отрегулированы, а в каждую плату была загружена прошивка со своим номером датчика.

Расположим в комнате источник звука, и запустим его, тем самым вызовем срабатывание датчиков звука, и отправку на сервер необходимой информации для анализа.

Проверим содержимое базы данных, и убедимся, что данные были успешно получены. Пример полученных данных представлен на рисунке 33.

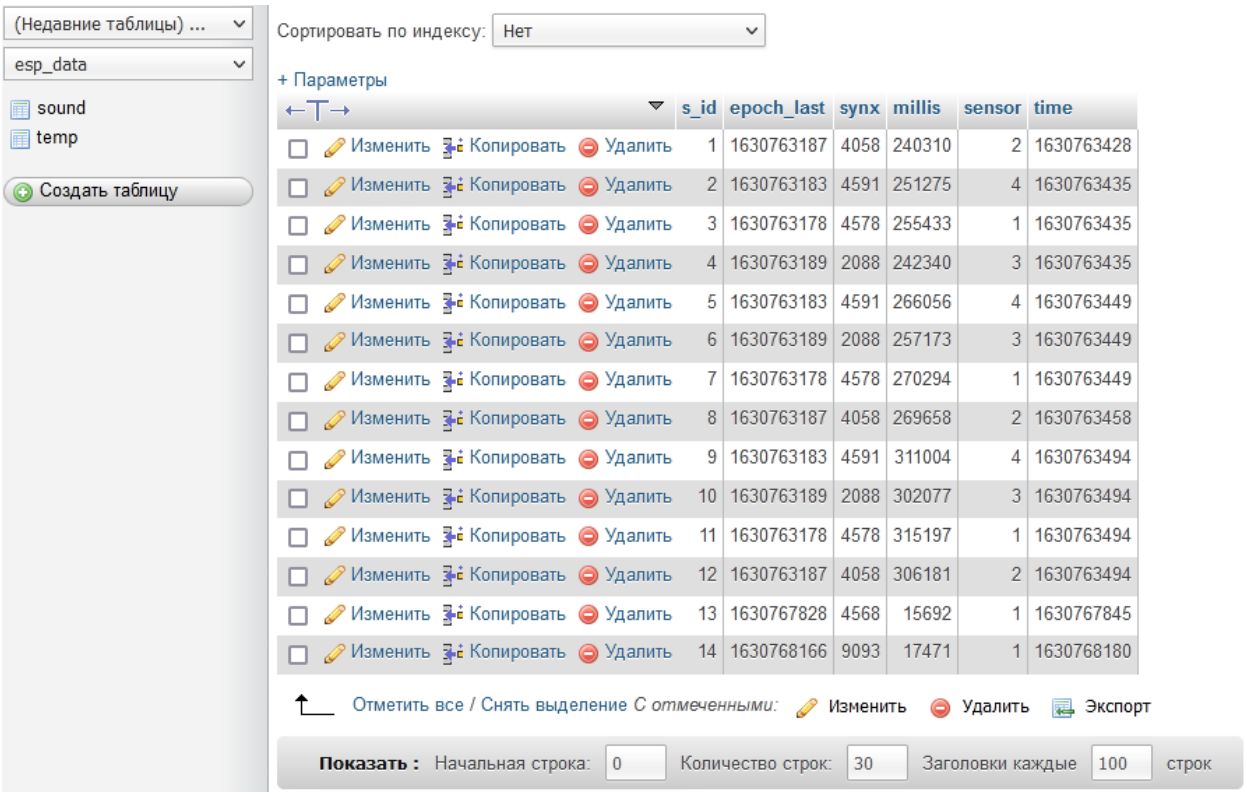


Рисунок 33 – Информация о времени срабатывания датчиков звука в базе данных

Запустим алгоритм расчета местоположения источника звука по трем датчикам (два в углу на одной стороне, и третий на середине противоположной стороны, датчики расположены по форме треугольника). Данный способ отлично фиксирует факт присутствия источника звука в помещении, однако по полученным данным рассчитать его местоположение не получилось.

Следующие два способа рассчитывают местоположение источника звука в помещении по четырем датчикам. В первом случае датчики располагаются в углах помещения, во втором на середине сторон. Как показала практика, оба способа работают одинаково. Они однозначно

фиксируют факт наличия источника звука в помещении, но не всегда срабатывают все четыре датчика, и примерные координаты источника звука были получены только в половине случаев.

Последний, четвертый способ, который учитывает сразу восемь звуковых датчиков, тоже однозначно фиксирует факт присутствия источника звука в помещении, но получить сразу с восьми датчиков информацию, по которой возможно рассчитать местоположение источника звука в помещении так и не получилось.

Не смотря на то, что система для точного определения координат локального источника звука нуждается в доработке, она отлично может определять факт присутствия источника звука в каком либо помещении и использоваться в работе некоторых систем «Умного дома», таких как охранная система, управление освещением, управление электроприборами и прочее.

Заключение

В ходе выполнения научно-квалификационной работы, целью которой являлась разработка доступной системы определения положения локального источника звука, было выполнено следующее:

– произведен анализ актуальности разработки автоматизированных систем управления зданием («Умный дом»), в том числе и подсистемы определения локального источника звука;

– разработана система сбора информации с датчиков на основе технологии технология «Internet вещей»;

– выполнен математический расчет координат источника звука;

– разработаны программные алгоритмы расчета местоположения источника звука;

– создана система акустической локализации, способная точно определять, в каком именно помещении находится источник звука, с возможностью доработки до точного определения координат источника звука в самом помещении.

Исходя из содержания и объема проделанных работ, поставленная цель научно-квалификационной работы выполнена. Полученная система может быть использована для выполнения некоторых функций автоматизированной системы управления зданием с позиции энергетической эффективности.

Список используемой литературы и используемых источников

1. «Умный дом»: методические указания для слушателей курсов повышения квалификации / А.Н. Стариков, С.И. Рощина, А.В. Власов. – Владимир: Изд-во ВЛГУ, 2014.

2. Zipperer A, et al. Electric Energy Management in the Smart Home: Perspectives on Enabling Technologies and Consumer Behavior // NREL/JA-5500-57586. – 2013/

3. Morris M.E., et al. Smart-Home Technologies to Assist Older People to Live Well at Home // Aging Sci. – 2013

4. С.П. Фомин. Возможность использования методов определения позы человека в системе «Умный дом». Алгоритмы, методы и системы обработки данных. Муромский институт (филиал) ГОУ ВПО ВлГУ - 2015 — № 1 (30) — с. 75-86.

5. Д.С. Сорокин, М.А. Калоев Использование микроконтроллеров в программно-аппаратных комплексах «Умный дом». Межвузовская научно-техническая конференция студентов, аспирантов и молодых специалистов им. Е.В. Арменского. Материалы конференции. - М.:МИЭМ НИУ ВШЭ, 2016. – С. 205 – 207.

6. Павлов О.Е., Состояние и развитие рынка "Интеллектуальных зданий" в России / О.Е. Павлов // Автоматизация в промышленности. 2003. № 5. – С. 37-38.

7. Либерман Б.М., Интеллектуальное здание. Некоторые вопросы интеграции систем безопасности и жизнеобеспечения / Б.М. Либерман // Автоматизация в промышленности. 2003. № 5. – С. 44-46.

8. Фрейдман А.В., Протокол передачи данных в стандарте АВОК "Интеллектуальные здания" / А.В. Фрейдман // Автоматизация в промышленности. 2003. № 5. – С. 48-49.

9. Интеллектуальное здание от компании Нанко // Автоматизация в промышленности. 2003. № 5. – С. 53-54.

10. Беспалов Е.К., Применение универсального SCADA-пакета iFIX в системах диспетчеризации зданий / Е.К. Беспалов, Е.Г. Левина // Автоматизация в промышленности. 2003. № 5. – С. 55-56.

11. Уваров А.В., Эффективные системы диспетчеризации современных зданий и комплексов / А.В. Уваров // Автоматизация в промышленности. 2006. № 10. – С. 21-26.

12. Жиленков Н.Н., Умные деревни / Н.Н. Жиленков // Автоматизация в промышленности. 2006. № 10. – С. 26-29.

13. Паршиков А.В., Как построить "Интеллектуальное здание"? / А.В. Паршиков // Автоматизация в промышленности. 2006. № 10. – С. 31-33.

14. Демченко Д.А., Распределенная управляющая сетевая платформа для построения систем автоматизации зданий / Д.А. Демченко, В.Б. Ланский, С.А. Третьяков // Автоматизация в промышленности. 2006. № 10. – С. 34-39.

15. Ильин В.В., Стандарты для автоматизации зданий - время для изменений / В.В. Ильин // Автоматизация в промышленности. 2008. № 10. – С. 16-19.

16. Широков А.В., Интеллектуальное здание для диспетчерского центра управления / А.В. Широков // Автоматизация в промышленности. 2008. № 10. – С. 57-58.

17. Аристова Н.И., Автоматизация освещения: стандарты, источники освещения, реализованные проекты / Н.И. Аристова // Автоматизация в промышленности. 2011. № 9. – С. 03-09.

18. Угреватов А.Ю., АСУ наружным освещением города / А.Ю. Угреватов, В.Ю. Угреватов // Автоматизация в промышленности. 2011. № 9. – С. 16-18.

19. Щербина С.В., ГИС и «Internet вещей» для управления промышленными предприятиями / С.В. Щербина // Автоматизация в промышленности. 2015. № 1. – С. 21-24.

20. Internet вещей: от фантастики к реальности // Автоматизация в промышленности. 2015. № 8. – С. 39-41.

21. Пучкова А.А., Разработка системы управления освещением в интеллектуальном здании / А.А. Пучкова // Потенциал интеллектуально одаренной молодежи - развитию науки и образования Материалы V Международного научного форума молодых ученых, студентов и школьников. Под общей редакцией Д. П. Ануфриева. 2016. – С. 145-148.

22. Упоров Е.И., Коммуникационные технологии для умного дома / Е.И. Упоров, С.В. Рыбкин // Международный студенческий научный вестник. 2019. № 1. – С. 44-51.

23. Big Data от А до Я. Часть 1: Принципы работы с большими данными, парадигма MapReduce [Электронный ресурс] / Режим доступа: https://habr.com/company/dca/blog/267361/ – (Дата обращения: 04.06.2018).

24. Что такое Big data: собрали всё самое важное о больших данных [Электронный ресурс] / Режим доступа: https://rb.ru/howto/chto-takoe-big-data/ – (Дата обращения: 07.06.2018).

25. MapReduce [Электронный ресурс] / Режим доступа: https://ru.bmstu.wiki/MapReduce – (Дата обращения: 08.06.2018).

26. И.И. Кусакин. Программно-аппаратный комплекс автоматизированного контроля целостности инфраструктуры жилых помещений для социального обеспечения. XV Международная телекоммуникационная конференция молодых ученых и студентов «МОЛОДЕЖЬ И НАУКА». Тезисы докладов. В 3-х частях. Ч. 3. – М.:НИЯУ МИФИ, 2012. – С. 156 – 157.

27. Сатаев Д.Г., Угрозы информационной безопасности системы «Умный дом» / Д.Г. Сатаев // Электронный научный журнал. 2019. № 6 (26). – С. 24-28.

28. Алефиренко В.М., Проблемы конфиденциальности и безопасности в системах «Умный дом» / В.М. Алефиренко, В.В. Костюченко // Актуальные научные исследования в современном мире. 2018. № 3-2 (35). – С. 196-204.

29. Балуева А.В., Управление событиями безопасности и построение защищенного протокола передачи данных для системы интернета вещей /

А.В. Балуева, Ю.А. Свинцов // Тенденции развития науки и образования. 2018. № 37-6. – С. 25-29.

30. Филиппов И.Е., Уязвимости и недостатки системы Умный дом / И.Е. Филиппов // В сборнике: European Scientific Conference сборник статей VIII Международной научно-практической конференции : в 3 ч.. 2018. – С. 171-173.

31. Чайко А.А., Разработка системы охраны и контроля объекта в концепции Умного дома / А.А. Чайко // Моя профессиональная карьера. 2019. Т. 3. № 5. – С. 271-277.

32. Гусейнов А.А., Будущие вопросы безопасности интернета вещей Умного дома // А.А. Гусейнов, А.А. Османов // В сборнике: OPEN INNOVATION сборник статей XI Международной научно-практической конференции. 2019. – С. 23-25.

33. Система Viziware, [Электронный ресурс]. – Режим доступа: URL http://www.comvisionsys.ru/product/viziware/ (дата обращения 18.12.2017)

34. С.П. Фомин. Возможность использования методов определения позы человека в системе «Умный дом». Алгоритмы, методы и системы обработки данных. Муромский институт (филиал) ГОУ ВПО ВлГУ - 2015 — № 1 (30) — с. 75-86.

35. Харке В. Умный дом. Объединение в сеть бытовой техники и систем коммуникаций в жилом помещении / В. Харке. – М.: Техносфера, 2006. – 288 с.

36. В.В. Казаков. Разработка умного зеркала Smart Mirror. Межвузовская научно-техническая конференция студентов, аспирантов и молодых специалистов им. Е.В. Арменского. Материалы конференции. - М.:МИЭМ НИУ ВШЭ, 2016. – С. 348 – 349.

37. Умный дом. Фирменный магазин продукции Xiaomi [Электронный ресурс]. – Режим доступа: URL https://ru-mi.com/device/umnyiy-dom/ (дата обращения 21.12.2017)

38. Что такое микроконтроллеры - назначение, устройство, софт», [Электронный ресурс]. – Режим доступа: URL http://electrik.info/main/automation/549-chto-takoe-mikrokontrollery-naznachenie-ustroystvo-princip-raboty-soft.html дата обращения (16.12.2018)

39. Америка. Вики [Электронный ресурс]. – Режим доступа: URL http://wiki.amperka.ru/ (дата обращения 18.12.2017)

40. Технологии беспроводной передачи данных ZigBee, BlueTooth, Wi-Fi, [Электронный ресурс]. – Режим доступа: URL http://wireless-e.ru/articles/bluetooth/2006_1_10.php дата обращения (16.12.2018)

41. Что такое микроконтроллеры - назначение, устройство, софт», [Электронный ресурс]. – Режим доступа: URL http://electrik.info/main/automation/549-chto-takoe-mikrokontrollery-naznachenie-ustroystvo-princip-raboty-soft.html дата обращения (16.12.2018)

42. WeMos D1 Wi-Fi UNO (ESP8266 ESP-12E) [Электронный ресурс]. – Режим доступа: URL https://radioprog.ru/shop/merch/57 дата обращения (10.02.2020)

43. Wi-fi модуль NodeMcu v3 с чипом ESP8266 (ESP-12e) — отправка GET запроса с сохранением в базу данных [Электронный ресурс]. – Режим доступа: URL https://rcl-radio.ru/?p=50109 дата обращения (18.02.2020)

44. А. С. Ковтуненко. Технология обнаружения источников звука методом акустической визуализации. Всероссийская научная конференция "Информационные технологии интеллектуальной поддержки принятия решений", Уфа-Ставрополь-Ханты-Мансийск, Россия - 2019. – С. 165-168.

45. Гусев А.Е., Пространственная локализация источника звука с использованием микрофонных решёток / А.Е. Гусев, О.Е. Бизин, Д.А. Токарев // Перспективные технологии в средствах передачи информации - ПТСПИ-2017. 2017. – С. 204-206.

46. Львов А.В., Триангуляционная система определения координат источника звука / А.В. Львов, М.Н. Агапов, А.И. Тищенко // Ползуновский вестник. 2010. № 2 — с. 159-162.

47. Определение положения точечного источника звука с помощью трёх микрофонов [Электронный ресурс]. – Режим доступа: URL http://wiki.amperka.ru/%D1%81%D0%B5%D0%BD%D1%81%D0%BE%D1%80%D1%8B:microphones_plus_servoprivod дата обращения (10.03.2021)

Приложение А

Алгоритм создания БД

```
<?php
	// Подключение к БД
	include "config.php"; // Считываем настройки

	$ls = mysql_connect($bd_n, $u_n, $u_p); // Подключаемся к БД

	if (!$ls)
	{
		echo "Нет соединения с БД";
		exit();
	}

	echo "Соединение выполнено успешно";

	if (!mysql_select_db($db_name))
	{
		echo "<br> База данных не найдена. Создаю базу....";

		if (!mysql_query("CREATE DATABASE ".$db_name." CHARACTER SET utf8 COLLATE utf8_general_ci"))
		{
			echo "<br> Не могу создать базу данных. Выход из программы";
			exit();
		}
```

```
            echo "<br> База данных успешно создана! Подключаюсь... повторно";

            if (!mysql_select_db($db_name))
            {
                  echo "<br>Так и не подключился, что-то не то. Выход.";
                  exit();
            }
      }

      echo "<br> База данных выбрана успешно! Работаем!";

      mysql_query("SET NAMES 'utf8'");
      mysql_query("SET CHARACTER SET 'utf8'");
      mysql_query("SET SESSION collation_connection = 'utf8_general_ci'");



      if (!mysql_query("SELECT * FROM temp"))
      {
            echo "<br> Таблица в базе не найдена. Создаю таблицу....";

            if (!mysql_query("CREATE TABLE temp ( t_id int(11) unsigned NOT NULL auto_increment, temp int(11) unsigned NOT NULL, time int(11) unsigned NOT NULL, sensor int(11) unsigned NOT NULL, PRIMARY KEY (t_id) ) AUTO_INCREMENT=1 ;"))
            {
                  echo "<br> Не могу создать таблицу. Выход из программы";
                  exit();
```

```
}

            echo "<br> Таблица успешно создана! Повторно запрашиваю
            данные";

            if (!mysql_query("SELECT * FROM temp"))
            {
                  echo "<br> Так и не получил данные, что-то не то. Выход.";
                  exit();
            }
      }

      echo "<br> Данные из таблицы успешно получены! Работаем!";
?>
```

Приложение Б

Настройки для подключения к БД

```
<?php
	$sdb_n = "localhost";  // Сервер
	$u_n = "root";		// Пользователь
	$u_p = "";		// Пароль
	$db_name = "ESP_DATA";	// База данных

	$esp_pass = "qwerty";	// Пароль на доступ ESP к БД
?>
```

Приложение В

Алгоритм подключения к БД

```
<?php
	// Подключение к БД
	include "config.php"; // Считываем настройки

	mysql_connect($bd_n, $u_n, $u_p); // Подключаемся к БД
	mysql_select_db($db_name);

	mysql_query("SET NAMES 'utf8'");
	mysql_query("SET CHARACTER SET 'utf8'");
	mysql_query("SET SESSION collation_connection = 'utf8_general_ci'");

?>
```

Приложение Г

Алгоритм добавления данных в БД

```
<?php
	// Если есть данные
	if(isset($_GET['pass']))
	{
		include "connect.php"; // Подключаемся к БД
		// Проверяем пароль

		if ( ($_GET['pass']) != $esp_pass) exit();

		mysql_query("INSERT INTO temp (temp, time, sensor) VALUES (".$_GET['temp']."', ".time()."', ".$_GET['sensor']."');");

	}
?>
```

Приложение Д

Алгоритм вывода данных в БД

```
<?php
	include "connect.php"; // Подключаемся к БД

?>

	<table border='1'>
	<tr>
		<th>Идентификатор
данных</th><th>Температура</th><th>Время</th><th>Датчик</th>
	</tr>
<?php
	$query = mysql_query("SELECT * FROM temp");

	while ($data = mysql_fetch_assoc($query))
	{
		echo
"<tr><td>".$data['t_id']."</td><td>".$data['temp']."</td><td>".date("d.m.y  H:i:s",
$data['time']-4000000)."</td><td>".$data['sensor']."</td></tr>";
	}
?>
	</table>
```

Приложение Е

Алгоритм прошивки микроконтроллера

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>

#include "DHT.h"
#define DHTPIN 13
DHT dht(DHTPIN, DHT11);

const char* td = "TEST_X";
const char* pa = "09041990";

const char* host = "192.168.17.7";

void setup()
{
  Serial.begin(9600);
  delay(10);

  // Подключаемся к сети WI-FI
  Serial.println();
  Serial.println();
  Serial.print("Connect…….");
  Serial.println(td);

  WiFi.mode(WIFI_STA);
  WiFi.begin(td, pa);
```

```
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.print(".");
  }

  Serial.println("WiFi ok");
  Serial.println(WiFi.localIP());
}

void loop()
{
  float temp = dht.readTemperature(); // Считываем температуру
  Serial.println(temp);
  int t=temp;

  HTTPClient http;
  Serial.println("http://192.168.17.7/add.php?temp=" + (String)t +
"&sensor=1&pass=qwerty");
  http.begin("http://192.168.17.7/add.php?temp=" + (String)t +
"&sensor=1&pass=qwerty");
  delay(10000);
  int httpCode = http.GET();
  if (httpCode > 0)
  {
    Serial.print("HTTP ");
    Serial.print(httpCode);
    if (httpCode == 200) {
      Serial.println(" OK");
```

```
    } else {
      Serial.println(" Error");
    }
  }
  http.end();
  delay(50000);
}
```

Приложение Ж

Алгоритм прошивки микроконтроллера с синхронизацией с сервером времени

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include <WiFiUdp.h> // Библиотека для обновления времени


// Сюда название вашей Wi-FI сети и пароль!!!
const char* td = "TEST_X";
const char* p = "09041990";

const char* host = "192.168.17.7";


unsigned int localPort = 2390;      // локальный порт для получения пакетов обновления

unsigned long epoch = 0;            // Текущее время, изначально это ноль

unsigned long synx = 0;             // Время (ардуино) последней синхронизации

unsigned long epoch_last;           // Время, полученное от сервера времени последней синхронизации

int a1 = A0;

int sensor = 1; // Номер датчика
```

```
// Сервер обновления времени
IPAddress timeServerIP;

const char* ntpServerName = "time.nist.gov";

const int NTP_PACKET_SIZE = 48; \

byte packetBuffer[ NTP_PACKET_SIZE]; //Создаем массив, для получения пакета обновления времени

// Создаем объект для передачи и получения сообщений обновления
WiFiUDP udp;

void setup()
{
  /* WeMos способен подключаться к последней использованной WiFi-сети
    (либо после запуска, либо после сброса), используя настройки, хранящиеся в специальных участках flash-памяти.
    По умолчанию эти настройки записываются на flash-память каждый раз, когда используются в функциях
    вроде WiFi.begin(ssid, password). Причем запись происходит всякий раз, независимо от того,
    изменились ли SSID и пароль или нет.

    Это может привести к износу flash-памяти – в зависимости от того, как часто вызываются эти функции.
```

Если выставить аргумент persistent на false, то SSID и пароль будут записаны на flash-память только в

том случае, если новые значения не будут соответствовать тем, что хранятся во flash-памяти.

```
 */

 WiFi.persistent(false);

 Serial.begin(9600);

 // Подключаемся к сети WI-FI
 Serial.print("……. ");
 Serial.println(td);
 WiFi.begin(td, p);

 while (WiFi.status() != WL_CONNECTED)
{
   delay(1000);
   Serial.print(".……");
 }


 Serial.println("WiFi connected");

 Serial.println("IP address: ");
 Serial.println(WiFi.localIP());

 Serial.println("Start UDP!!!!");
```

```
  udp.begin(localPort);
  Serial.print("Local port: ");
  Serial.println(udp.localPort());

}

void loop()
{

  // Если синхронизации еще не было, или она была, но более 24 часов назад, пора сделать новую синхронизацию

  if ((synx == 0) || ( (millis() - synx) / 1000 > 86400) )
  {
    // Получаем адрес сервера из пулла ип адреса
    WiFi.hostByName(ntpServerName, timeServerIP);

    sendNTPpacket(timeServerIP); // отправляем NTP пакет на сервер времени
    // ждем, чтобы узнать, доступен ли ответ
    delay(1000);

    int cb = udp.parsePacket();

if (!cb)
    {
      Serial.println("no packet yet");
    }
    else
```

```
{
    Serial.print("packet received, length=");
    Serial.println(cb);
    udp.read(packetBuffer, NTP_PACKET_SIZE);

    // Дальше идет работа с байтами
    unsigned long highWord = word(packetBuffer[40], packetBuffer[41]);

    unsigned long lowWord = word(packetBuffer[42], packetBuffer[43]);

    // combine the four bytes (two words) into a long integer
    // this is NTP time (seconds since Jan 1 1900):
    unsigned long secsSince1900 = highWord << 16 | lowWord;
    Serial.print("Seconds since Jan 1 1900 = " );
    Serial.println(secsSince1900);

    // В конце которой мы получаем количество секунд, прошедших с 1 января 1970 года

// Начинаем конвертацию
    Serial.print("Unix time = ");

    // Время интернета Unix time начинается 1 января 1970.

    const unsigned long seventyYears = 2208988800UL;

    // subtract seventy years:
    epoch_last = secsSince1900 - seventyYears;
```

```
    // print Unix time:
    Serial.println(epoch);


    synx = millis(); // Запоминаем время ардуино последней синхронизации
  }
}
else
{
  // Если не надо синхронизироваться, опрашиваем датчик
  int d = analogRead(a1);
  if ( (d < 310) || (d > 340) )
  {
    // Если мы видим, что датчик сработал

    HTTPClient http;
```

```
Serial.println("http://192.168.17.7/add_z.php?epoch_last=" + (String)epoch_last + "&synx=" + (String)synx + "&millis=" + (String)millis() + "&sensor=" + (String)sensor + "&pass=qwerty");

http.begin("http://192.168.17.7/add_z.php?epoch_last=" + (String)epoch_last + "&synx=" + (String)synx + "&millis=" + (String)millis() + "&sensor=" + (String)sensor + "&pass=qwerty");

delay(10000);


int httpCode = http.GET();
if (httpCode > 0)
{
 Serial.print("HTTP ");
 Serial.print(httpCode);
 if (httpCode == 200)
 {
  Serial.println(" OK");
 }
  else
  {
  Serial.println(" Error");
 }
}


http.end();
```

```
    delay(1000);

   }
  }
}


// отправляем запрос NTP серверу времени по данному адресу
unsigned long sendNTPpacket(IPAddress & address)
{
  Serial.println("sending NTP packet...");

  // Очищаем буфер
  memset(packetBuffer, 0, NTP_PACKET_SIZE);

  // Инициализируем значения, необходимые для формирования запроса NTP
  // (см. URL-адрес выше для получения подробной информации о пакетах)
  packetBuffer[0] = 0b11100011;   // LI, Version, Mode
  packetBuffer[1] = 0;     // Stratum, or type of clock
  packetBuffer[2] = 6;     // Polling Interval
  packetBuffer[3] = 0xEC;  // Peer Clock Precision

  // 8 bytes of zero for Root Delay & Root Dispersion
  packetBuffer[12]  = 49;
  packetBuffer[13]  = 0x4E;
  packetBuffer[14]  = 49;
  packetBuffer[15]  = 52;
```

```
  // Когда все инициализировано
  // вы можете отправить пакет с запросом метки времени
  udp.beginPacket(address, 123); // Запросы NTP предназначены для порта 123
  udp.write(packetBuffer, NTP_PACKET_SIZE);
  udp.endPacket();
}
```

Приложение И

Алгоритм создания таблицы в БД

```
<?php
	// Подключение к БД
	include "config.php"; // Считываем настройки

	$link = mysql_connect($sdb_n, $u_n, $u_p); // Подключаемся к БД

	if (!$link)
	{
		echo "Нет соединения с БД";
		exit();
	}

	echo "Соединение выполнено успешно";

	if (!mysql_select_db($db_name))
	{
		echo "<br> База данных не найдена. Создаю базу....";

		if (!mysql_query("CREATE DATABASE ".$db_name." CHARACTER SET utf8 COLLATE utf8_general_ci"))
		{
			echo "<br> Не могу создать базу данных. Выход из программы";
			exit();
		}
```

```
            echo "<br> База данных успешно создана! Подключаюсь... повторно";

            if (!mysql_select_db($db_name))
            {
                echo "<br>Так и не подключился, что-то не то. Выход.";
                exit();
            }
        }

        echo "<br> База данных выбрана успешно! Работаем!";

        mysql_query("SET NAMES 'utf8'");
        mysql_query("SET CHARACTER SET 'utf8'");
        mysql_query("SET SESSION collation_connection = 'utf8_general_ci'");



        if (!mysql_query("SELECT * FROM sound"))
        {
            echo "<br> Таблица в базе не найдена. Создаю таблицу....";

            if (!mysql_query("CREATE TABLE sound ( s_id int(11) unsigned NOT NULL auto_increment, epoch_last int(11) unsigned NOT NULL, synx int(11) unsigned NOT NULL, millis int(11) unsigned NOT NULL, sensor int(11) unsigned NOT NULL, time int(11) unsigned NOT NULL, PRIMARY KEY (s_id) ) AUTO_INCREMENT=1;"))
            {
                echo "<br> Не могу создать таблицу. Выход из программы";
```

```
                exit();
            }

            echo "<br> Таблица успешно создана! Повторно запрашиваю данные";

            if (!mysql_query("SELECT * FROM sound"))
            {
                echo "<br> Так и не получил данные, что-то не то. Выход.";
                exit();
            }
        }

        echo "<br> Данные из таблицы успешно получены! Работаем!";
?>
```

Приложение К

Алгоритм добавления данных с датчиков звука в БД

```
<?php
	// Если есть данные
	if(isset($_GET['pass']))
	{
		include "connect.php"; // Подключаемся к БД
		// Проверяем пароль

		if ( ($_GET['pass']) != $esp_pass) exit();

		echo "INSERT INTO sound (epoch_last, synx, millis, sensor, time) VALUES (".$_GET['epoch_last']."., ".$_GET['synx']."., ".$_GET['millis']."., ".$_GET['sensor']."., ".time().");";
		mysql_query("INSERT INTO sound (epoch_last, synx, millis, sensor, time) VALUES (".$_GET['epoch_last']."., ".$_GET['synx']."., ".$_GET['millis']."., ".$_GET['sensor']."., ".time().");");

	}
?>
```

Приложение Л

Алгоритм расчета местоположения источника звука

```
<?php

$m=4; // кол-во микрофонов
$v=340.29; // скорость звука

$s=2; // выбор способа анализа
// 1 - три датчика
// 2 - четыре датчика в углах
// 3 - четыре датчика по сторонам
// 4 - восемь датчиков

// Координаты микрофонов для каждого способа
if ($s==1)
{
	$x[1]=0;
	$y[1]=0;
	$z[1]=1;

	$x[2]=1.25;
	$y[2]=2.5;
	$z[2]=1;

	$x[3]=2.5;
	$y[3]=0;
	$z[3]=1;
}
```

```
if ($s==2)
{
	$x[1]=0;
	$y[1]=0;
	$z[1]=1;

	$x[2]=0;
	$y[2]=2.5;
	$z[2]=1;

	$x[3]=2.5;
	$y[3]=2.5;
	$z[3]=1;

	$x[4]=2.5;
	$y[4]=0;
	$z[4]=1;
}

if ($s==3)
{
	$x[1]=0;
	$y[1]=1.25;
	$z[1]=1;

	$x[2]=1.25;
	$y[2]=2.5;
	$z[2]=1;
```

```
        $x[3]=2.5;
        $y[3]=1.25;
        $z[3]=1;

        $x[4]=1.25;
        $y[4]=0;
        $z[4]=1;
}

if ($s==4)
{
        $x[1]=0;
        $y[1]=0;
        $z[1]=1;

        $x[2]=0;
        $y[2]=1.25;
        $z[2]=1;

        $x[3]=0;
        $y[3]=2.5;
        $z[3]=1;

        $x[4]=1.25;
        $y[4]=2.5;
        $z[4]=1;

        $x[5]=2.5;
```

```
        $y[5]=2.5;
        $z[5]=1;

        $x[6]=2.5;
        $y[6]=1.25;
        $z[6]=1;

        $x[7]=2.5;
        $y[7]=0;
        $z[7]=1;

        $x[8]=1.25;
        $y[8]=0;
        $z[8]=1;
}

// Теперь сделаем запросы в БД, и вытащим от туда время

include "connect.php"; // Подключаемся к БД

$query = mysql_query("SELECT * FROM sound ORDER BY s_id DESC"); // Запрашиваем последнюю запись

if ($data = mysql_fetch_assoc($query)) $time=$data['time']; // Сохраним время последнего срабатывания датчика
echo "$time <br>";
```

```
if ($s==1)
{
	// Время срабатывания 1 датчика
	$query = mysql_query("SELECT * FROM sound WHERE sensor=1 and time=".$time);
	if ($data = mysql_fetch_assoc($query)) $time_1=$data['epoch_last'] + ($data['millis']-$data['synx'])/1000;
	echo "$time_1 <br>";

	// Время срабатывания 2 датчика
	$query = mysql_query("SELECT * FROM sound WHERE sensor=4 and time=".$time);
	if ($data = mysql_fetch_assoc($query)) $time_2=$data['epoch_last'] + ($data['millis']-$data['synx'])/1000;
	echo "$time_2 <br>";

	// Время срабатывания 3 датчика
	$query = mysql_query("SELECT * FROM sound WHERE sensor=7 and time=".$time);
	if ($data = mysql_fetch_assoc($query)) $time_3=$data['epoch_last'] + ($data['millis']-$data['synx'])/1000;
	echo "$time_3 <br>";
}

if ($s==2)
{
	// Время срабатывания 1 датчика
```

```
$query = mysql_query("SELECT * FROM sound WHERE sensor=1 and time=".$time);
if ($data = mysql_fetch_assoc($query)) $time_1=$data['epoch_last'] + ($data['millis']-$data['synx'])/1000;
echo "$time_1 <br>";

// Время срабатывания 2 датчика
$query = mysql_query("SELECT * FROM sound WHERE sensor=3 and time=".$time);
if ($data = mysql_fetch_assoc($query)) $time_2=$data['epoch_last'] + ($data['millis']-$data['synx'])/1000;
echo "$time_2 <br>";

// Время срабатывания 3 датчика
$query = mysql_query("SELECT * FROM sound WHERE sensor=5 and time=".$time);
if ($data = mysql_fetch_assoc($query)) $time_3=$data['epoch_last'] + ($data['millis']-$data['synx'])/1000;
echo "$time_3 <br>";

// Время срабатывания 4 датчика
$query = mysql_query("SELECT * FROM sound WHERE sensor=7 and time=".$time);
if ($data = mysql_fetch_assoc($query)) $time_4=$data['epoch_last'] + ($data['millis']-$data['synx'])/1000;
echo "$time_4 <br><br>";
}
```

```
if ($s==3)
{
	// Время срабатывания 1 датчика
	$query = mysql_query("SELECT * FROM sound WHERE sensor=2 and time=".$time);
	if ($data = mysql_fetch_assoc($query)) $time_1=$data['epoch_last'] + ($data['millis']-$data['synx'])/1000;
	echo "$time_1 <br>";

	// Время срабатывания 2 датчика
	$query = mysql_query("SELECT * FROM sound WHERE sensor=4 and time=".$time);
	if ($data = mysql_fetch_assoc($query)) $time_2=$data['epoch_last'] + ($data['millis']-$data['synx'])/1000;
	echo "$time_2 <br>";

	// Время срабатывания 3 датчика
	$query = mysql_query("SELECT * FROM sound WHERE sensor=6 and time=".$time);
	if ($data = mysql_fetch_assoc($query)) $time_3=$data['epoch_last'] + ($data['millis']-$data['synx'])/1000;
	echo "$time_3 <br>";

	// Время срабатывания 4 датчика
	$query = mysql_query("SELECT * FROM sound WHERE sensor=8 and time=".$time);
	if ($data = mysql_fetch_assoc($query)) $time_4=$data['epoch_last'] + ($data['millis']-$data['synx'])/1000;
```

```
        echo "$time_4 <br><br>";
}

if ($s==4)
{
        // Время срабатывания 1 датчика
        $query = mysql_query("SELECT * FROM sound WHERE sensor=1 and time=".$time);
        if ($data = mysql_fetch_assoc($query)) $time_1=$data['epoch_last'] + ($data['millis']-$data['synx'])/1000;
        echo "$time_1 <br>";

        // Время срабатывания 2 датчика
        $query = mysql_query("SELECT * FROM sound WHERE sensor=2 and time=".$time);
        if ($data = mysql_fetch_assoc($query)) $time_2=$data['epoch_last'] + ($data['millis']-$data['synx'])/1000;
        echo "$time_2 <br>";

        // Время срабатывания 3 датчика
        $query = mysql_query("SELECT * FROM sound WHERE sensor=3 and time=".$time);
        if ($data = mysql_fetch_assoc($query)) $time_3=$data['epoch_last'] + ($data['millis']-$data['synx'])/1000;
        echo "$time_3 <br>";

        // Время срабатывания 4 датчика
```

```
$query = mysql_query("SELECT * FROM sound WHERE sensor=4 and time=".$time);
if ($data = mysql_fetch_assoc($query)) $time_4=$data['epoch_last'] + ($data['millis']-$data['synx'])/1000;
echo "$time_4 <br><br>";

// Время срабатывания 5 датчика
$query = mysql_query("SELECT * FROM sound WHERE sensor=5 and time=".$time);
if ($data = mysql_fetch_assoc($query)) $time_5=$data['epoch_last'] + ($data['millis']-$data['synx'])/1000;
echo "$time_5 <br>";

// Время срабатывания 6 датчика
$query = mysql_query("SELECT * FROM sound WHERE sensor=6 and time=".$time);
if ($data = mysql_fetch_assoc($query)) $time_6=$data['epoch_last'] + ($data['millis']-$data['synx'])/1000;
echo "$time_6 <br>";

// Время срабатывания 7 датчика
$query = mysql_query("SELECT * FROM sound WHERE sensor=7 and time=".$time);
if ($data = mysql_fetch_assoc($query)) $time_7=$data['epoch_last'] + ($data['millis']-$data['synx'])/1000;
echo "$time_7 <br>";

// Время срабатывания 8 датчика
```

```
	$query = mysql_query("SELECT * FROM sound WHERE sensor=8 and time=".$time);
	if ($data = mysql_fetch_assoc($query)) $time_8=$data['epoch_last'] + ($data['millis']-$data['synx'])/1000;
	echo "$time_8 <br><br>";
}

if ($s==1)
{
	// Считаем разницу времени для датчиков
	$t[1][2]=abs($time_1-$time_2);
	$t[1][3]=abs($time_1-$time_3);
	$t[2][3]=abs($time_2-$time_3);

	echo $t[1][2]. "<br>";
	echo $t[1][3]. "<br>";
	echo $t[2][3]. "<br>";
}

if (($s==2) || ($s==3))
{
	// Считаем разницу времени для датчиков
	$t[1][2]=abs($time_1-$time_2);
	$t[1][3]=abs($time_1-$time_3);
	$t[1][4]=abs($time_1-$time_4);
	$t[2][3]=abs($time_2-$time_3);
	$t[2][4]=abs($time_2-$time_4);
	$t[3][4]=abs($time_3-$time_4);
```

```
        echo $t[1][2]. "<br>";
        echo $t[1][3]. "<br>";
        echo $t[1][4]. "<br>";
        echo $t[2][3]. "<br>";
        echo $t[2][4]. "<br>";
        echo $t[3][4]. "<br>";
}

if ($s==4)
{
        $t[1][2]=abs($time_1-$time_2);
        $t[1][3]=abs($time_1-$time_3);
        $t[1][4]=abs($time_1-$time_4);
        $t[1][5]=abs($time_1-$time_5);
        $t[1][6]=abs($time_1-$time_6);
        $t[1][7]=abs($time_1-$time_7);
        $t[1][8]=abs($time_1-$time_8);

        $t[2][3]=abs($time_2-$time_3);
        $t[2][4]=abs($time_2-$time_4);
        $t[2][5]=abs($time_2-$time_5);
        $t[2][6]=abs($time_2-$time_6);
        $t[2][7]=abs($time_2-$time_7);
        $t[2][8]=abs($time_2-$time_8);

        $t[3][4]=abs($time_3-$time_4);
        $t[3][5]=abs($time_3-$time_5);
        $t[3][6]=abs($time_3-$time_6);
```

```
$t[3][7]=abs($time_3-$time_7);
$t[3][8]=abs($time_3-$time_8);

$t[4][5]=abs($time_4-$time_5);
$t[4][6]=abs($time_4-$time_6);
$t[4][7]=abs($time_4-$time_7);
$t[4][8]=abs($time_4-$time_8);

$t[5][6]=abs($time_5-$time_6);
$t[5][7]=abs($time_5-$time_7);
$t[5][8]=abs($time_5-$time_8);

$t[6][7]=abs($time_6-$time_7);
$t[6][8]=abs($time_6-$time_8);

$t[7][8]=abs($time_7-$time_8);

echo $t[1][2]. "<br>";
echo $t[1][3]. "<br>";
echo $t[1][4]. "<br>";
echo $t[1][5]. "<br>";
echo $t[1][6]. "<br>";
echo $t[1][7]. "<br>";
echo $t[1][8]. "<br>";

echo $t[2][3]. "<br>";
echo $t[2][4]. "<br>";
echo $t[2][5]. "<br>";
```

```
        echo $t[2][6]. "<br>";
        echo $t[2][7]. "<br>";
        echo $t[2][8]. "<br>";

        echo $t[3][4]. "<br>";
        echo $t[3][5]. "<br>";
        echo $t[3][6]. "<br>";
        echo $t[3][7]. "<br>";
        echo $t[3][8]. "<br>";

        echo $t[4][5]. "<br>";
        echo $t[4][6]. "<br>";
        echo $t[4][7]. "<br>";
        echo $t[4][8]. "<br>";

        echo $t[5][6]. "<br>";
        echo $t[5][7]. "<br>";
        echo $t[5][8]. "<br>";

        echo $t[6][7]. "<br>";
        echo $t[6][8]. "<br>";

        echo $t[7][8]. "<br>";
}
```

```
function calc($X,$Y,$Z)
{
	global $m, $v, $x, $y, $z, $t;
	$sum = 0;
	for ($i=1; $i<=($m-1); $i++)
	{
		for ($j=$i+1; $j<=$m; $j++)
		{
			$Ri = sqrt(pow(($x[$i]-$X),2) + pow(($y[$i]-$Y),2) + pow(($z[$i]-$Z),2));
			$Rj = sqrt(pow(($x[$j]-$X),2) + pow(($y[$j]-$Y),2) + pow(($z[$j]-$Z),2));
			$F = $Ri - $Rj - $v*$t[$i][$j];
			$F=$F*$F;
			$sum=$sum+$F;
		}
	}
	return $sum;
}


$min=10000000;
for ($i=0; $i<=2.55; $i=$i+0.1)
{
	for ($j=0; $j<=2.55; $j=$j+0.1)
	{
		for ($k=0; $k<=2.55; $k=$k+0.1)
		{
```

```
                        if (calc($i,$j,$k)<$min)
                        {
                                $min = calc($i,$j,$k);
                                $mx=$i;
                                $my=$j;
                                $mz=$k;

                        }
                    //echo "<br>";
                    //echo "$i $j $k ";
                    //echo calc($i,$j,$k);
                    }
                }
            }

            echo "$min $mx $my $mz";



?>
```

Приложение М

Прошивка микроконтроллера для регулирования датчика звука

```
int a1 = A0;

void setup()
{
  Serial.begin(9600);
}
void loop()
{
 Serial.print("A0");
 Serial.print(" ");

  Serial.print(analogRead(a1));
  Serial.println(" ");

}
```