**Содержание**

[Введение 2](#_Toc43473456)

[1.Анализ уязвимости объектов 3](#_Toc43473457)

[2.Основные технические решения, принятые в проекте 4](#_Toc43473458)

[2.1 Пожарная сигнализация 6](#_Toc43473459)

[2.2 Охранная сигнализация 7](#_Toc43473460)

[2.3 Умный дом 8](#_Toc43473461)

[3 Микроконтроллеры в проекте 8](#_Toc43473462)

[3.1 NodeMCU 9](#_Toc43473463)

[3.2 Arduino Nano 13](#_Toc43473464)

[3.3 W5500 16](#_Toc43473465)

[3.4 SIM800L 19](#_Toc43473466)

[4 Среда передачи данных 22](#_Toc43473467)

[4.1 Свойства стандарта RS-485 23](#_Toc43473468)

[Интерфейс RS-485 обладает следующими свойствами: 23](#_Toc43473469)

[4.2 Технические характеристики RS-485 24](#_Toc43473470)

[Интерфейс RS-485 обладает следующими техническими характеристиками: 24](#_Toc43473471)

[4.3 Описание обмена данными по стандарту RS-485 24](#_Toc43473472)

[4.4 Основные принципы реализации протоколов верхнего уровня (типа MODBUS) 25](#_Toc43473473)

[5 Применяемое оборудование и его обоснование в системах охранной, пожарной сигнализации и системы умного дома 26](#_Toc43473474)

[6 Расчёт основных числовых характеристик проектируемой системы безопасности 29](#_Toc43473475)

[7 Программа взаимодействие управляющего контролера с датчиками и управляющими устройствами 30](#_Toc43473476)

[8 Демонстрация возможностей системы 32](#_Toc43473477)

[9 Разбор основных частей программного кода 35](#_Toc43473478)

[9.1 Протокол MQTT 39](#_Toc43473479)

[9.2 Интерфейс SPI 41](#_Toc43473480)

[Заключение 44](#_Toc43473481)

[Список использованных источников 45](#_Toc43473482)

[Приложение А Список компонентов 46](#_Toc43473483)

[Приложение B Исходный код ESP-12 47](#_Toc43473484)

[Приложение С Исходный код AVR 48](#_Toc43473485)

# Введение

В современном мире вопрос сохранности имущества, недвижимости, ценных бумаг стоит достаточно остро. Связано это с ростом преступности [9] и развитием технологий, которые пользуются злоумышленниками в своих деяниях. Как результат угроза взлома зданий, квартир, домов, машин и кражи вещей или их порча постоянно растет. Стражи правопорядка не в состоянии предотвращать трудности подобного рода, поэтому на помощь приходят охранные системы.

Под охранными системами принято понимать комплекс устройств, основное предназначение которых -  охрана организации от внешних и внутренних угроз и взлома квартир, домов. Вполне логично, что оборудование само по себе не ловит злоумышленников. [Датчики](https://www.forter.com.ua/datchiki/) фиксируют проникновение воров в помещение и передают сигнал на центральный компьютер или блок управления. Он в свою очередь оповещает группу быстрого реагирования или охранника о возникшей ситуации. Представители охранной компании прибывают в считанные минуты на объект и задерживают взломщиков, которые даже не успевают понять, что произошло. Это происходит при правильном проектирование и последующей установкой систем безопасности.

Система безопасности способна распознавать несанкционированное проникновение в охраняемый объект. Современные высокоточные датчики моментально срабатывают на движение или открывание дверей. Встроенная в Умный Дом сигнализация объявляет тревогу. Вам отправляется текстовое сообщение на телефон, звучит предупредительная сирена, включается освещение, ведется запись, и к дому подъезжает охрана.

При утечке газа, воды или, когда образуется дым, сработают датчики, а «умный дом» своевременно сообщит аварийную ситуацию и проинформирует об этом соответствующими сигналами (звуковыми или световыми). Под Умным Домом подразумевается система, которая обеспечивает безопасность и ресурсосбережение. Умный Дом не только добавляет автоматизации, но дает полезную информацию, которая может быть использована и в охранных целях.

# 1.Анализ уязвимости объектов

Встраивание комплексных охранных систем, получило широкое распространение, они многофункциональны, и вместе с этим, сложны. Подобными системами оборудуются не только жилые дома, но и государственные учреждения, места массового скопления людей, а также стратегические объекты, например, АЭС, аэропорты и т.д. Сочетание разных технологий при построении одной автоматизированной системы увеличивает количество возможных недостатков решения с точки зрения безопасности, что, безусловно, привлекает внимание злоумышленников.

Вход электрической сети частично контролировать невозможно, поэтому злоумышленнику не обязательно проникать во внутрь квартиры чтобы обесточить охраняемый объект

. Это действие может нарушить работу приемно-контрольного устройства в данном проекте это микроконтроллер. Против этого действия будет установлена резервная батарея для бесперебойной подачи питания.

Система пожарной безопасности как наиболее приоритетная, строилась таким образом, чтобы функционировать при потере сигнала с приёмных устройств и не зависеть от других элементов в системе безопасности.

Все объекты, которые могут быть подвергнуты угрозам безопасности или противоправным посягательства, имеют различную потенциальную уязвимость с точки зрения возможного информационного, материального или морального ущерба. Под уязвимостью объекта понимается степень его незащищенности к воздействию нарушителей. Уязвимость зависит от эффективности системы защиты (СЗ) объекта, т.е. степени его защищенности от воздействия нарушителей и вызванными ими действиями.  
 Существующая практика в области охраны объектов [7] показывает, что для подобного типового объекта существуют следующие основные виды угроз:

1) Проникновение внешнего нарушителя на охраняемый объект с целью:

* нанесения вреда жизни и здоровью человеку;
* хищения материальных ценностей, ценных бумаг, оборудование, порчи имущества.

2) Чрезвычайная ситуация (пожар), несущая:

* угрозу жизни и здоровью;
* угрозу уничтожения товарно-материальных ценностей, расположенных в объекте.

3) Чрезвычайная ситуация (потоп), несущая:

* угрозу нанесение вреда имуществу.
* угрозу нанесение вреда имуществу другим людям (соседям).

# 2.Основные технические решения, принятые в проекте

В современных условиях роста преступности в стране, активизации террористической деятельности, увеличения количества техногенных аварий, применение интегрированных систем безопасности (ИСБ) является наиболее эффективным способом организации инженерно-технической защиты объектов, поскольку создает необходимые условия для комплексного решения задач по обеспечению сохранности материальных ценностей, сохранности жизни и здоровья людей.

Система безопасности построена [4] на работе типа клиент-сервер. В её основе лежат микроконтроллеры семейства AVR и ESP. Последний из которых превосходит по производительности промышленные автоматизированные системы. ESP-12 она же NodeMCU в системе является ключевой (сервер). ESP-12 принимает и обрабатывает данные от других микроконтроллеров, и далее принимает выбранную инструкцию. Данный микроконтроллер также служит для отображения информации, выводя информацию на LCD экран. На базе этого микроконтроллера сделана реализация умного дома. Система умного дома интегрирована под охранную систему.

Микроконтроллер Arduino Nano в системе служит как клиент [5]. Он был выбран т.к. у него значительно больше портов ввода-вывода в сравнении с ESP с которыми могут взаимодействовать датчики. Он может отправлять и принимать данные с сервера. На одном таком микроконтроллере можно поставить на охрану крыло здание или большой дом. В проекте клиентов типа Arduino Nano или других подобных микроконтроллеров может быть до 16 т.е. данный проект может охранять целое здание или корпус, например, корпус№5 ГГУ им. Ф. Скорины. Связь между микроконтроллерами осуществляется по RS-485, данный протокол является помехозащищённым с максимальной длиной в 1200 метров, промышленным протоколом передачи данных.

Данный микроконтроллер семейства AVR на сегодняшний день повсеместно используются в компьютерах, для автоматизации управления электронной аппаратурой, различными приборами и механизмами, применяемыми в промышленных, коммерческих, а также бытовых целях. Невысокая стоимость, широкий ассортимент и богатые возможности микроконтроллеров этой серии способствовали их большой популярности.

К клиенту в свою очередь крепятся различные датчики различных систем, которые могут понадобится на определённой территории. К ним можно подключить все возможные датчики, существующие на данный момент с разными способами подключения. Это делает систему максимально конфигурированной.

В обычной топологии клиент-сервер существует один большой недостаток – это единая точка отказа, если выйдет из строя сервер, данные не кому будет обрабатывать. В данном проекте эту проблему пришлось обойти из-за большого риска отказа или плановой перезагрузки сервера. Для этого в клиент добавлен Ethernet-контроллер который реализует аппаратный стек TCP/IP, он отправляет данные по средствам веб интерфейса в пункт охраны без обработки данных на сервере. Данный Ethernet-контроллер может быть в системе в зависимости от потребности.

Самая быстрая система оповещения является GSM-модуль, который в случае тревоги отправит смс-оповещение, он также может устанавливаться по желанию, но хотя бы одно устройство которое закрывает проблему топологии клиент-сервер должна быть.

Программное обеспечение разрабатывалось таким образом, чтобы в дальнейшем его можно было обновлять и дополнять. Код был написан на языке С++ используя процедурное программирование. К каждому микроконтроллеру написано своё ПО которое уже аппаратно реализует взаимодействие микроконтроллеров. ПО в данной проекте является более ключевой, чем его аппаратная часть т.к. именно под разработанную программу строилась аппаратная часть.

В данном проекте реализован источник бесперебойного питания на девять вольт с последующим понижением до пяти вольт из-за требований по питанию некоторой обвязки датчиков и системы вывода. Систему также можно запитать и от 12V, поэтому проблем с выбором другого ИБП не возникнет. ИБП в системы служит чтобы отойти от возможных перебоев в сети.

Основные подсистемы ИСБ:

* Охранная сигнализация
* Пожарная сигнализация
* Умный дом

Все системы безопасности при комплексном подходе связаны между собой. Таким образом создается интегрированная среда обмена сигналами между различными элементами. Все системы работают в комплексе, выполняя одновременно функции контроля, сдерживания, обнаружения опасности, ее оценки и реагирования на нее, обеспечивая защиту сразу по нескольким направлениям.

## **2.1 Пожарная сигнализация**

Основные задачи функционирования системы пожарной сигнализации в совокупности с организационными мероприятиями — это задачи спасения жизни людей и сохранения имущества. Минимизация ущерба при пожаре напрямую зависит от своевременного обнаружения и локализации очага возгорания.

Пожарная сигнализация будет построена с помощью компонентов:

1. LM393 ИК модуль датчика обнаружения пламени.
2. MQ-2 Служит для детектирования горючего газа и дыма.

Настроить порта ввода-вывода на микроконтроллере можно под разные задачи по-разному. В систему безопасности можно использовать и любые другие открытые компоненты, например, датчики компании [8] Болид:

1. Извещатель пожарный комбинированный адресно-аналоговый газовый и тепловой максимально-дифференциальный С2000-ИПГ. Способен оповещать на раннем этапе воспламенение. В количестве двух.
2. Извещатель пожарный тепловой максимально дифференциальный адресно-аналоговый С2000-ИП-03. Узкоспециализированный, применяется для контроля инженерных коммуникаций. В количестве один.

## **2.2 Охранная сигнализация**

Охранная сигнализация — совокупность технических средств для обнаружения появления нарушителя на охраняемом объекте и подачи извещения о тревоге для принятия мер по задержанию нарушителя. Из определения можно выделить несколько основных задач охранной сигнализации:

1. Обнаружение нарушителя;

2. Формирование извещения об обнаружении нарушителя в нужном информационном формате;

3. Передача извещения в нужном формате в определённое место;

4. Обеспечение процедуры постановки на охрану и снятия с охраны (взятия/снятия).

Охранная сигнализация будет построена на компонентах:

1. SR501 Инфракрасный датчик движения
2. SW-420 Вибрационный датчик
3. Gercon Магнитный датчик, один из самых распространённых изващателей.

Также систему можно построить и на других компонентах:

1. Извещатель охранный совмещенный объемный оптико-электронный и поверхностный звуковой адресный С2000-СТИК. Избавляет от установки дополнительных датчиков на окнах.
2. Адресный извещатель магнитный С2000-СМК.

## **2.3 Умный дом**

Умный дом – это комплекс решений для автоматизации дома, который избавит вас от рутины. Сюда входят системы, контролирующие всё, что происходит в квартире.

Умный дом поможет избавиться от тех тревог, что связаны с вашей квартирой. Состоянием дома с установленной технологией умного дома можно следить из любой точки мира. К этому всему добавляется и удобная автоматизация. Если запущенная перед уходом из дома стиральная машинка вышла из строя и есть шанс затопить имущество себе и соседей. Автоматизация умного дома предотвратит эту проблему уведомив вас об утечке. Если она действительно подтекает, вам об этом моментально сообщит датчик протечек.

Умный дом будет построен на следующих компонентах:

1. BME280 – Цифровой датчик температуры и влажности, барометрический датчик давления
2. Релейный модуль 4CH.

**3 Микроконтроллеры в проекте**

Микроконтроллер — [микросхема](https://ru.wikipedia.org/wiki/%D0%9C%D0%B8%D0%BA%D1%80%D0%BE%D1%81%D1%85%D0%B5%D0%BC%D0%B0), предназначенная для управления [электронными](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%B8%D0%BA%D0%B0) [устройствами](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%BE%D0%B5_%D1%83%D1%81%D1%82%D1%80%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%BE). Типичный микроконтроллер сочетает на одном кристалле функции [процессора](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D1%80) и [периферийных устройств](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B8%D1%84%D0%B5%D1%80%D0%B8%D0%B9%D0%BD%D0%BE%D0%B5_%D1%83%D1%81%D1%82%D1%80%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%BE), содержит [ОЗУ](https://ru.wikipedia.org/wiki/%D0%9E%D0%97%D0%A3) и (или) [ПЗУ](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%81%D1%82%D0%BE%D1%8F%D0%BD%D0%BD%D0%BE%D0%B5_%D0%B7%D0%B0%D0%BF%D0%BE%D0%BC%D0%B8%D0%BD%D0%B0%D1%8E%D1%89%D0%B5%D0%B5_%D1%83%D1%81%D1%82%D1%80%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%BE). По сути, это однокристальный [компьютер](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80), способный выполнять относительно простые задачи.

Отличается от [микропроцессора](https://ru.wikipedia.org/wiki/%D0%9C%D0%B8%D0%BA%D1%80%D0%BE%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D1%80) интегрированными в микросхему устройствами ввода-вывода, таймерами и другими периферийными устройствами.

Разработчики микроконтроллеров придумали остроумную идею – объединить процессор, память, ПЗУ и периферию внутри одного корпуса, внешне похожего на обычную микросхему. С тех пор производство микроконтроллеров ежегодно во много раз превышает производство процессоров, а потребность в них не снижается.

Все датчики и периферийные устройства в проекте можно легко поменять, кроме выбранных микроконтроллеров, изменить возможно только их модель, но не семейство. Например, Arduino Nano можно заменить на Arduino Uno или Mega, ESP-12 на STM32. Т.к. они являются ядром системы на которой связывается программная и аппаратная часть.

Микроконтроллеры в проекте выбраны с запасов мощности, чтобы в бедующем система не испытывала трудность при увеличении функционала и добавления новых модулей. Программное обеспечение было сконфигурировано именно под данные микроконтроллеры, для увеличения безопасности и отказоустойчивости. Возможно лишь изменение модели, без изменения семейства.

Для лучшего понимание выбранных микроконтроллеров далее будет их описание.

Основные микроконтроллеры в проекте:

1. NodeMCU(ESP8266).
2. Arduino Nano
3. W5500
4. SIM800L
   1. **NodeMCU**

NodeMCU – это платформа на основе модуля ESP8266. ESP8266 – микроконтроллер китайского производителя [Espressif Systems](https://ru.wikipedia.org/w/index.php?title=Espressif_Systems&action=edit&redlink=1) с интерфейсом [Wi-Fi](https://ru.wikipedia.org/wiki/Wi-Fi) [10]. Помимо Wi-Fi, микроконтроллер отличается отсутствием флеш-памяти в SoC, программы пользователя исполняются из внешней [флеш-памяти](https://ru.wikipedia.org/wiki/%D0%A4%D0%BB%D0%B5%D1%88-%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C) с интерфейсом [SPI](https://ru.wikipedia.org/wiki/Serial_Peripheral_Interface).  Плата предназначена для удобного управления различными схемами на расстоянии посредством передачи сигнала в локальную сеть или интернет через Wi-Fi. Возможности применения этой платы ограничивается лишь вашей фантазией. К примеру, на базе NodeMCU можно создать «умный дом», настроив управление светом или вентиляцией через телефон, регистрацию показаний датчиков и многое другое.

Существует несколько поколений плат NodeMCU – V1(версия 0.9), V2(версия 1.0) и V3 (версия 1.0). Обозначения V1, V2, V3 используются при продаже в интернет-магазинах. Нередко происходит путаница в платах – например, V3 внешне идентична V2. Также все платы работают по принципу open-source, поэтому их могут производить любые фирмы. Но в настоящее время производством плат NodeMcu занимаются Amica, DOIT и LoLin/Wemos. Портов ввода-вывода (рисунок 1) на данном микроконтроллере меньше чем на тех же семейства AVR.

Программные средства разработки (программный комплект разработчика, SDK) состоят из: [Компилятора](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%82%D0%BE%D1%80). Компилятор для Xtensa LX106 входит в пакет компиляторов [GNU Compiler Collection](https://ru.wikipedia.org/wiki/GNU_Compiler_Collection). Компилятор имеет открытые исходные тексты. В разных SDK могут содержаться разные сборки этого компилятора, немного отличающиеся поддерживаемыми опциями.

Библиотек для работы с периферией контроллера, стеков протоколов WiFi, [TCP/IP](https://ru.wikipedia.org/wiki/TCP/IP).

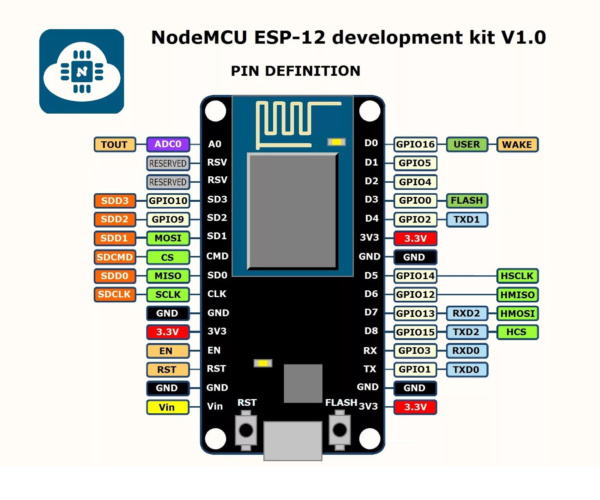


Рисунок 1 – Распиновка NodeMcu [10]

Типовое применение ESP8266 как аппаратной основы Internet of Things чаще всего подразумевает установку в домах или офисах. При этом сетевое подключение осуществляется к домашней/офисной локальной сети с выходом в интернет через [роутер](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%80%D1%88%D1%80%D1%83%D1%82%D0%B8%D0%B7%D0%B0%D1%82%D0%BE%D1%80). Пользователь устройства может контролировать его с помощью планшета или компьютера через свою локальную сеть либо удаленно, через Интернет.

Wi-Fi на ESP8266 может работать как в роли [точки доступа](https://ru.wikipedia.org/wiki/%D0%A2%D0%BE%D1%87%D0%BA%D0%B0_%D0%B4%D0%BE%D1%81%D1%82%D1%83%D0%BF%D0%B0) так и оконечной станции. При нормальной работе в локальной сети ESP8266 конфигурируется в режим оконечной станции. Для этого устройству необходимо задать [SSID](https://ru.wikipedia.org/wiki/SSID) Wi-Fi сети и, в закрытых сетях, пароль доступа. Для первоначального конфигурирования этих параметров удобен режим точки доступа. В режиме точки доступа устройство видно при стандартном поиске сетей в планшетах и компьютерах. Остается подключиться к устройству, открыть HTML страничку конфигурирования и задать сетевые параметры. После чего устройство штатно подключится к локальной сети в режиме оконечной станции.

В случае исключительно местного использования возможно всегда оставлять устройство в режиме точки доступа, что снижает необходимые усилия пользователя по его настройке.

Локальная сеть. После подключения к Wi-Fi сети устройство должно получить IP-параметры локальной сети. Эти параметры можно задать вручную вместе с параметрами Wi-Fi либо активизировать какие либо [сервисы автоматического конфигурирования IP-параметров](https://ru.wikipedia.org/wiki/Zeroconf) (например, [DHCP](https://ru.wikipedia.org/wiki/DHCP)).

После настройки IP параметров обращение к серверу устройства в локальной сети обычно осуществляется по его IP адресу, сетевому имени (в случае если имена поддержаны какой либо технологией, например [NBNS](https://ru.wikipedia.org/wiki/NBNS)) или сервису (в случае если поддержан автоматический [поиск сервисов](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D0%BD%D0%B0%D1%80%D1%83%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5_%D1%81%D0%B5%D1%80%D0%B2%D0%B8%D1%81%D0%BE%D0%B2), например через протокол [SSDP](https://ru.wikipedia.org/wiki/Simple_Service_Discovery_Protocol)).

Интернет. Зачастую доступ к устройству требуется из Интернета. Например пользователь с мобильного телефона удаленно проверяет состояние своего «[умного дома](https://ru.wikipedia.org/wiki/%D0%A3%D0%BC%D0%BD%D1%8B%D0%B9_%D0%B4%D0%BE%D0%BC)», обращаясь напрямую к устройству. В этом случае устройство работает в режиме сервера, к которому обращается внешний клиент.

Как правило, устройство на основе ESP8266 находится в локальной сети офиса или дома. Выход в Интернет обеспечивает [роутер](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%80%D1%88%D1%80%D1%83%D1%82%D0%B8%D0%B7%D0%B0%D1%82%D0%BE%D1%80), подключенный с одной стороны к локальной сети а с другой к сети провайдера интернета. Провайдер назначает роутеру свой статический или динамический IP адрес и роутер осуществляет [трансляцию адресов](https://ru.wikipedia.org/wiki/NAT) локальной сети в сеть провайдера. По умолчанию правила этой трансляции обеспечивают свободную видимость интернет-адресов из локальной сети, но не позволяют обратиться к локальным адресам со стороны Интернета. Есть несколько способов обойти это ограничение.

Конфигурирование NAT. Большинство современных роутеров позволяют задать дополнительные правила [трансляции сетевых адресов](https://ru.wikipedia.org/wiki/NAT) между локальной и глобальной сетями. Как правило для этого используются технологии [Virtual server](https://ru.wikipedia.org/w/index.php?title=Virtual_server&action=edit&redlink=1) или [DMZ](https://ru.wikipedia.org/wiki/DMZ_(%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D1%8B%D0%B5_%D1%81%D0%B5%D1%82%D0%B8)). Обе технологии позволяют обратиться к серверу в локальной сети из глобальной сети, зная лишь IP адрес, выданный роутеру провайдером. В случае статического IP адреса роутера — это зачастую может быть удовлетворительным решением для ограниченного круга пользователей системы. Однако такой подход не всегда удобен: необходимо вручную конфигурировать роутер и выяснять IP-адрес роутера, который может регулярно меняться. Относительно легко решить проблему неизвестного IP адреса можно с помощью механизма [DDNS](https://ru.wikipedia.org/wiki/DDNS).

DDNS. Чтобы обратиться к серверу устройства конечный пользователь должен знать IP адрес, по которому находится устройство. Однако получить у провайдера Интернета для устройства статический IP адрес не всегда возможно, да и пользоваться таким адресом неудобно. Для решения этой проблемы были созданы специальные интернет-сервисы под общим наименованием [динамический DNS](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_DNS). Эти сервисы работают как специальные серверы с фиксированными именами в интернете. Разработчик заводит на таком сервисе свой аккаунт с уникальным именем. Параметры этого аккаунта он прописывает в устройстве. Устройство в режиме клиента периодически обращается к серверу сервиса, сообщая ему имя своего аккаунта и свой текущий IP адрес. Конечный пользователь в интернете обращается к этому же сервису и получает от него текущие IP параметры устройства. В таком случае устройство в сети видно с доменным именем третьего уровня, например, esp8266.ddns.org. Существует множество DDNS сервисов. Основная проблема DDNS сервисов это гарантии существования конкретного сервиса. Как правило, гарантируется только коммерческий сервис, когда за его использование взимается плата.

Внешние IoT сервисы. Чтобы облегчить проблему доступности устройства в Интернете и сделать инсталляцию устройства легкой для пользователя были разработан ряд решений. Механизм этих решений базируется на существовании в Интернете специального сервера, к которому может подключиться как IoT устройство, так и планшет/компьютер пользователя. При этом устройство работает в режиме клиента, никаких специальных настроек роутера или особых навыков от инсталлятора и пользователя устройства не требуется. Обмен данными с устройством осуществляется при посредничестве этого специального сервиса, параметры которого в устройство должен заложить разработчик. Распространение использования таких сервисов сдерживается необходимостью длительно поддерживать свой сервис в Интернете или пользоваться чужими сервисами с непонятными перспективами длительного существования бесплатных возможностей или регулярной оплатой коммерческих вариантов.

Основное применение ESP8266 находит в управлении разнообразными бытовыми приборами через беспроводные сети. Концепцию такого управления часто называют «Internet of Things» (IoT, «интернет вещей»). Верхний уровень IoT представлен разнообразными приложениями под популярные платформы (Android, iOS, Windows, …). Эти приложения позволяют разработчику прибора адаптировать приложение под управление его прибором и передать пользователю готовое решение. Существует несколько популярных реализаций концепции IoT в плане обмена данными по сети: HTML сервер на ESP8266. Контроль и управление устройством ведется через браузер. Тяжеловесное решение, подходит автономным устройствам автоматики.

AllJoyn – набирающий популярность открытый IoT протокол крупного альянса производителей цифровой техники «Allseen». Поддержка встроена в Windows 10.

HTTP запросы с использованием протоколов типа REST, XML-RPC (SOAP). Для этого на ESP8266 запускают упрощенный HTTP сервер, без HTML. Достоинство метода — отсутствие проблем с настройкой файрволлов, HTTP обычно открыт всегда.

MQTT. Это простой протокол поверх TCP/IP. Очень популярное решение. Существует большое количество IoT приложений верхнего уровня для Android, iOS и других платформ, поддерживающих этот протокол.

SNMP. Расширяемый протокол управления сетевыми устройствами. Основной недостаток в том что в большинстве сетей файрволлы блокируют прохождение SNMP.

ModBus и другие протоколы промышленной автоматизации.

## **3.2 Arduino Nano**

Общие сведения

Платформа Nano, построенная на микроконтроллере ATmega328 (Arduino Nano 3.0) или ATmega168 (Arduino Nano 2.x), [11] имеет небольшие размеры и может использоваться в лабораторных работах. Она имеет схожую с Arduino Duemilanove функциональность, однако отличается сборкой. Отличие заключается в отсутствии силового разъема постоянного тока и работе через кабель Mini-B USB. Nano разработана и продается компанией Gravitech.

Питание:

Arduino Nano может получать питание через подключение Mini-B USB, или от нерегулируемого 6-20 В (вывод Vin), или регулируемого 5 В (вывод 5V), внешнего источника питания. Автоматически выбирается источник с самым высоким напряжением.

Микросхема FTDI FT232RL получает питание, только если сама платформа запитана от USB. Таким образом при работе от внешнего источника (не USB), будет отсутствовать напряжение 3.3 В, генерируемое микросхемой FTDI, при этом светодиоды RX и TX мигаю только при наличии сигнала высокого уровня на выводах 0 и 1.

Память:

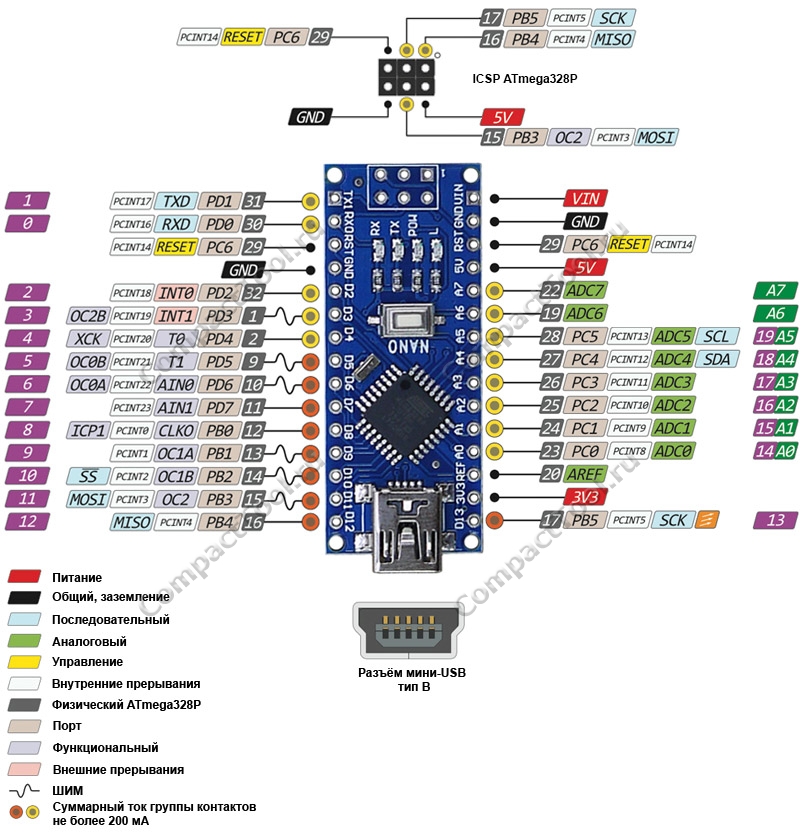
Микроконтроллер ATmega168 имеет 16 кБ флеш-памяти для хранения кода программы, а микроконтроллер ATmega328, в свою очередь, имеет 32 кБ (в обоих случаях 2 кБ используется для хранения загрузчика). ATmega168 имеет 1 кБ ОЗУ и 512 байт EEPROM (которая читается и записывается с помощью [библиотеки EEPROM](http://arduino.ru/Reference/Library/EERPOM)), а ATmega328 – 2 кБ ОЗУ и 1 Кб EEPROM.

Входы и Выходы:

Каждый из 14 цифровых выводов (рисунок 2) Nano, используя функции [pinMode()](http://arduino.ru/Reference/PinMode),[digitalWrite()](http://arduino.ru/Reference/DigitalWrite), и [digitalRead()](http://arduino.ru/Reference/DigitalRead), может настраиваться как вход или выход. Выводы работают при напряжении 5 В. Каждый вывод имеет нагрузочный резистор (стандартно отключен) 20-50 кОм и может пропускать . Некоторые выводы имеют особые функции:

Последовательная шина: 0 (RX) и 1 (TX). Выводы используются для получения (RX) и передачи (TX) данных TTL. Данные выводы подключены к соответствующим выводам микросхемы последовательной шины FTDI USB-to-TTL.

Внешнее прерывание: 2 и 3. Данные выводы могут быть сконфигурированы на вызов прерывания либо на младшем значении, либо на переднем или заднем фронте, или при изменении значения. Подробная информация находится в описании функции attachInterrupt().



**Рисунок 2 – Распиновка Arduino Nano [11]**

ШИМ: 3, 5, 6, 9, 10, и 11. Любой из выводов обеспечивает ШИМ с разрешением 8 бит при помощи функции analogWrite().

SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Посредством данных выводов осуществляется связь SPI, которая, хотя и поддерживается аппаратной частью, не включена в язык Arduino.

LED: 13. Встроенный светодиод, подключенный к цифровому выводу 13. Если значение на выводе имеет высокий потенциал, то светодиод горит.

На платформе Nano установлены 8 аналоговых входов, каждый разрешением 10 бит (т.е. может принимать 1024 различных значения). Стандартно выводы имеют диапазон измерения до 5 В относительно земли, тем не менее имеется возможность изменить верхний предел посредством функции analogReference(). Некоторые выводы имеют дополнительные функции:

I2C: A4 (SDA) и A5 (SCL). Посредством выводов осуществляется связь I2C (TWI). Для создания используется библиотека Wire (информация на сайте Wiring).

Дополнительная пара выводов платформы:

AREF. Опорное напряжение для аналоговых входов. Используется с функцией analogReference().

Reset. Низкий уровень сигнала на выводе перезагружает микроконтроллер. Обычно применяется для подключения кнопки перезагрузки на плате расширения, закрывающей доступ к кнопке на самой плате Arduino.

Связь:

На платформе Arduino Nano установлено несколько устройств для осуществления связи с компьютером, другими устройствами Arduino или микроконтроллерами. ATmega168 и ATmega328 поддерживают последовательный интерфейс UART TTL (5 В), осуществляемый выводами 0 (RX) и 1 (TX). Установленная на плате микросхема FTDI FT232RL направляет данный интерфейс через USB, а драйверы FTDI (включены в программу Arduino IDE) предоставляют виртуальный COM порт программе на компьютере. Мониторинг последовательной шины (Serial Monitor) программы Arduino позволяет посылать и получать текстовые данные при подключении к платформе. Светодиоды RX и TX на платформе будут мигать при передаче данных через микросхему FTDI или USB подключение (но не при использовании последовательной передачи через выводы 0 и 1).

Библиотекой SoftwareSerial возможно создать последовательную передачу данных через любой из цифровых выводов Nano.

ATmega168 и ATmega328 поддерживают интерфейсы I2C (TWI) и SPI. В Arduino включена библиотека Wire для удобства использования шины I2C. Более подробная информация находится в документации. Для использования интерфейса SPI обратитесь к техническим данным микроконтроллеров ATmega168 и ATmega328.

Программирование:

Платформа программируется посредством ПО Arduino. Из меню Tools > Board выбирается «Arduino Diecimila, Duemilanove или Nano w/ ATmega168» или «Arduino Duemilanove или Nano w/ ATmega328» (согласно установленному микроконтроллеру). Подробная информация находится в справочнике и инструкциях.

Микроконтроллеры ATmega168 и ATmega328 поставляются с записанным загрузчиком, облегчающим запись новых программ без использования внешних программаторов. Связь осуществляется оригинальным протоколом STK500.

Имеется возможность не использовать загрузчик и запрограммировать микроконтроллер через выводы блока ICSP (внутрисхемное программирование). Подробная информация находится в данной инструкции.

**3.3 W5500**

Микросхема W5500 - последнее дополнение к линейке популярных Ethernet-чипов от компании WIZNet - представляет собой функционально законченный чип встраиваемого Ethernet-контроллера [12] для обеспечения проводного соединения TCP/IP, позволяющий упростить подключение к интернету устройств и систем, использующих SPI (высокоскоростной последовательный периферийный интерфейс). W5500 позволяет удовлетворить потребности пользователей в стабильном подключении к Internet, с помощью единого чипа для реализации стека TCP/IP, 10/100, Ethernet MAC и PHY. Аппаратный стек TCP/IP поддерживает TCP, UDP, IPv4, ICMP, ARP, IGMP, PPPoE...

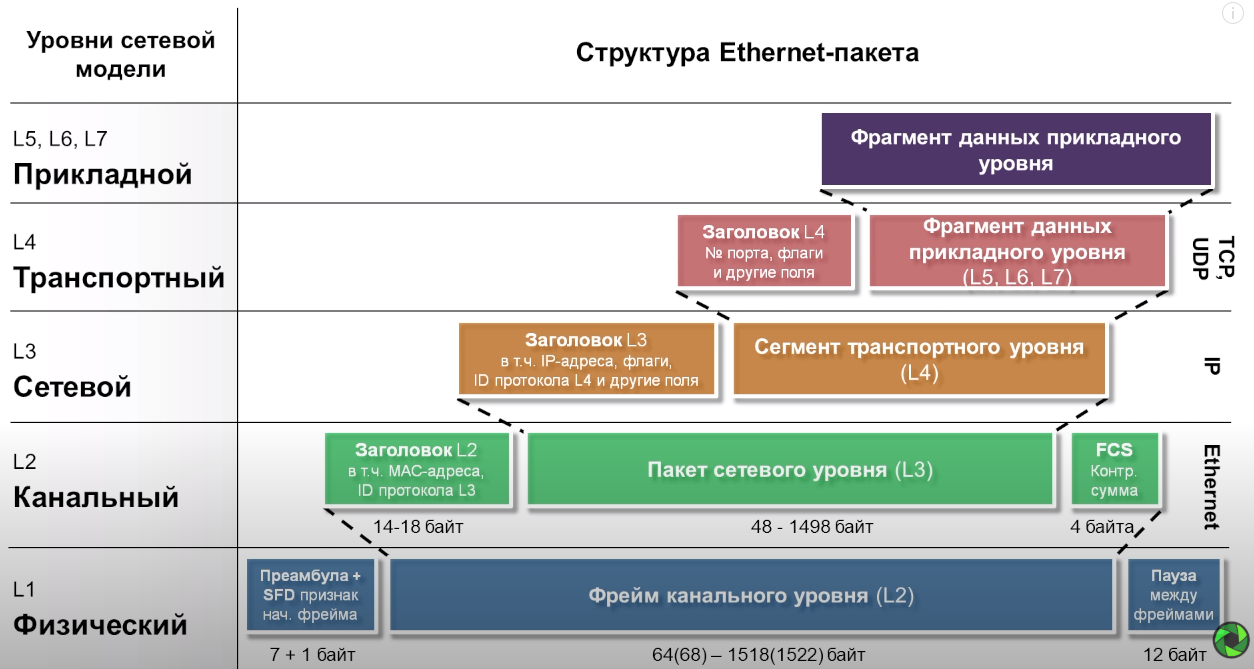
Новый усовершенствованный сетевой контроллер W5500 имеет более быстрый и универсальный интерфейс SPI (до 80 МГц), а также возможность более гибкого использования буфера оперативной памяти (ОЗУ) для целей временного хранения данных общего назначения, что является несомненным плюсом при использовании дешевых управляющих микроконтроллеров (MCU) с ограниченной RAM. Положительным является и сокращение энергопотребления компонентов аппаратного уровня и функционирующего ПО, реализованного за счет продвинутых решений физического уровня (PHY). W5500 поддерживает Ethernet-приложения, использующие популярные недорогие 8/16-разрядные микроконтроллеры. Ethernet возможности могут быть добавлены в существующие приложения без необходимости перехода на более дорогой MCU и полной передки аппаратного и программного обеспечения.

Благодаря отличающемуся улучшенным дизайном кристаллу 3-го поколения, уменьшению его геометрии (повышения компактности решения за счёт использования методов проектирования ‘SiP’ (System-in-Package), для контроллера W5500 значительно уменьшено не только энергопотребление, но и размер его корпуса.

Также W5500 может функционировать в качестве обычного Ethernet-трансивера с использованием стороннего программного реализованного TCP/IP-протокола. Его Ethernet трансивер имеет простой способ обновления прошивки (добавления новых функций) с помощью предпрошитого программного обеспечения (OS и RTOS ‘стека TCP/IP’), использующего преимущества реализации аппаратного обеспечения TCP/IP в новом чипе.

В отличие от конструкций, которые требуют стороннего программного обеспечения для обработки основных сетевых операций, W5500 предоставляет меры ‘firewall’-защиты на аппаратном уровне, что не может не сказываться на надежности функционирования разрабатываемого оборудования.

Функциональные возможности и технические преимущества W5500:

* Аппаратно поддерживаемые коммутационные протоколы обработки проводного TCP/IP стека: TCP, UDP, MAC, ICMP, IPv4, ARP, IGMP, PPPoE
* Простая реализация других сетевых протоколов
* Аппаратно реализованная сетевая превентивная защита от Un-типа атак, таких как переполнение, а также подмена и вброс пакетов
* Host интерфейс: высокоскоростной последовательный периферийный интерфейс SPI
* Основной интерфейс SPI, режим 0,3 / тактовая частота SPI до 80 МГц
* Требуется менее производительный MCU для работы с GPIO и SPI (резервные выходы GPIO могут использованы для других функций самостоятельно)
* Пониженное тепловыделение (для W5500: до 40℃ / другие аналогичные чипы : 60~70℃)
* Поддержка встроенного OS драйвера: Linux (поддержка ядра 2.4. xx, 2.6.xx, 3.1,xx) & RTOS
* Аппаратное обеспечение 8 независимых TCP/UDP сокетов (канальных соединений)
* Внутренний сокет 32 кбайт буферной памяти RAM для обеспечения процесса передачи TCP/IP пакетов
* More efficient power operation: power down & wake-on LAN over UDP
* 10BaseT / 100Base TX Ethernet со встроенным MAC/PHY
* Поддержка автоматического определения полярности принимаемого сигнала (полный & полудуплекс, 10 & 100 Based)
* Выводы под светодиодную LED индикацию (полный/полудуплекс, активное состояние)
* **Рисунок 3 – Уровни сетевого обмена [12]**

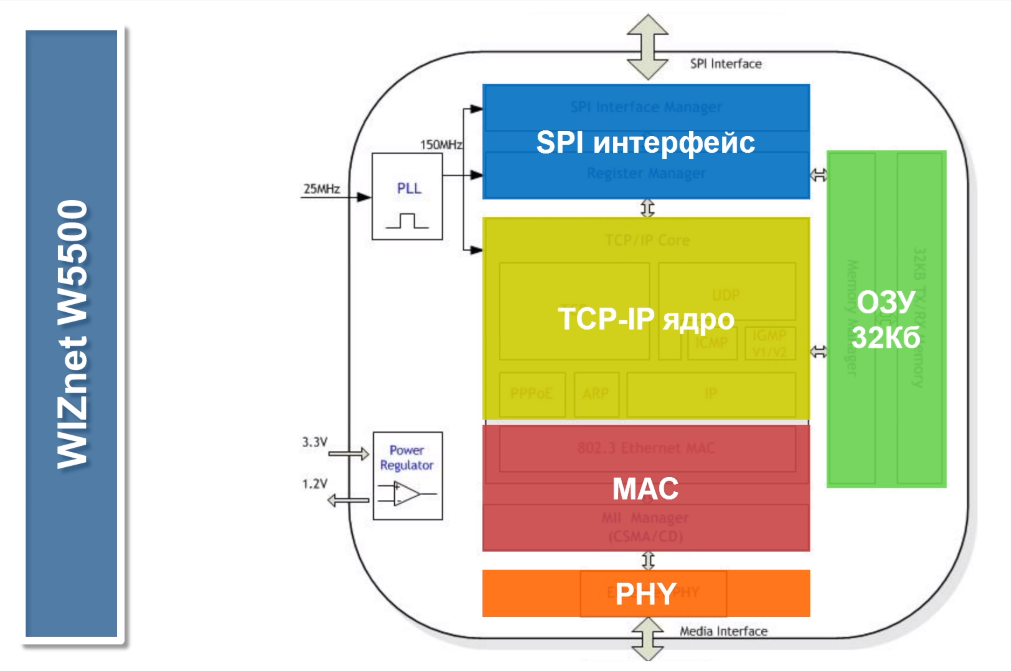
Рассмотрим протоколы реализующие сетевой обмен (рисунок 3). Делятся они на различные уровни, разделение это функциональное, то есть протоколы 1-го уровня решают какие-то конкретные задачи в общем сетевом обмене. В идеале они не должны знать детали реализации других уровней. Если несколько узлов в сети, то протоколы одного уровня общаются с такими

же протоколами того же уровня в другом узле. По разделению на уровни существует несколько сетевых моделей, самой популярная эта модель OSI.

В физическом фрейме содержатся данные различных уровней, таким образом они могут быть разобраны на приемнике и таким образом пакет может быть собран на передатчике. По сути пакет – это массив байт всего лишь линейная штука и так по физической линии передается фрейм, обрамленный преамбулой и паузой в конце и внутри этого фрейма.

Канальный уровень это ethernet фрейм у которого в заголовке есть mac-адрес, то есть физические адреса устройств. Есть контрольная сумма и внутри содержится пакет следующего уровня, сетевого уровня в данном случае айпи пакет. В нём также есть заголовок с адресами. Но адреса уже здесь не физические, а логические. Айпи адреса также, как и в теле, содержится сегмент транспортного уровня, то есть следующего уровня. У данного уровня есть свой заголовок номер порта, там и флаги различные и есть свое тело, внутри которого уже содержится фрагмент данных прикладного уровня. В нём, данные более высокого уровня ложатся во внутрь пакета более низкого уровня.

Взглянем на блок-схему (рисунок 4) w5500, выделим на ней основные блоки - это SPI интерфейсы, блок ОЗУ, MAC, TCP-IP.Данные блоки, ключевая особенность w5500.



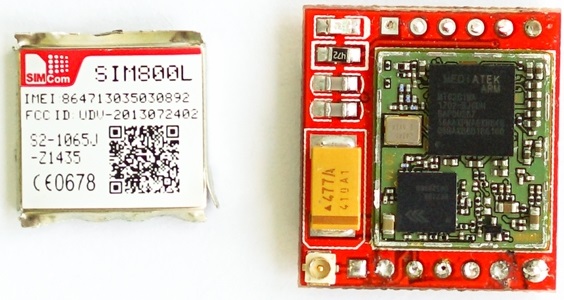
**Рисунок 4 – Блок схема W5500 [12]**

Буфер в оперативной памяти объемом 32 килобайта на самом деле эта память разделена на 2 блока по 16 килобайт. 16 килобайт могут распределяться между используемыми сокетами по 16 килобайт на передачу и 16 на прием.

## **3.4 SIM800L**

GSM-модулем SIM800L или модемом (GSM-модемом) будет называться модуль SIMCom SIM800L распаянный на плате (рисунок 5):

Но на самом деле это некорректно – модуль SIM800L скрывается под металлической крышкой с бело-красной наклейкой:



**Рисунок 5 – GSM модуль [13]**

Сердцем модуля SIM800L является чип Mediatek ARM MT6261. За GSM/GPRS-связь отвечает 4-х диапазонный (GSM850/ GSM900/CS1800) приемопередатчик RF7198

Исходя из этого, важно понимать, что различные модули 800 серии обладают разными характеристиками и функционалом.

Подавляющее большинство начинающих радиолюбителей не готовы работать с GSM/GPRS/GNSS-модулями напрямую [13]. Именно поэтому производители радиоэлектронных компонентов предлагают большое количество плат с уже размещенными на них GSM-модулями для удобной работы.  
 Каждая печатная плата — это переходник между GSM-модулем и пользователем. Самые простые из них предоставляют пользователю минимальный функционал — обмен данными с GSM-модулем по UART. В том числе, каждая из плат, в обязательном порядке имеет слот для внешней SIM-карты, таким образом, реализуя для пользователя этот интерфейс. Более сложные варианты, включая Arduino-шилды, наоборот, делают доступ к расширенному функционалу модуля более удобным.

Отсюда, второй по важности задачей, после того как выбран необходимый GSM-модуль, является выбор платы, на которой он распаян.

Питание:

Для начала работы понадобится рабочая SIM-карта формата microSIM.

Для питания модуля требуется напряжение из диапазона 3.4-4.5 В, рекомендованное 4 В. Как видно, это нестандартные для Arduino/Raspberry Pi 3, напряжения (5 В / 3.3 В), рассчитанные на питание от литиевых аккумуляторов.

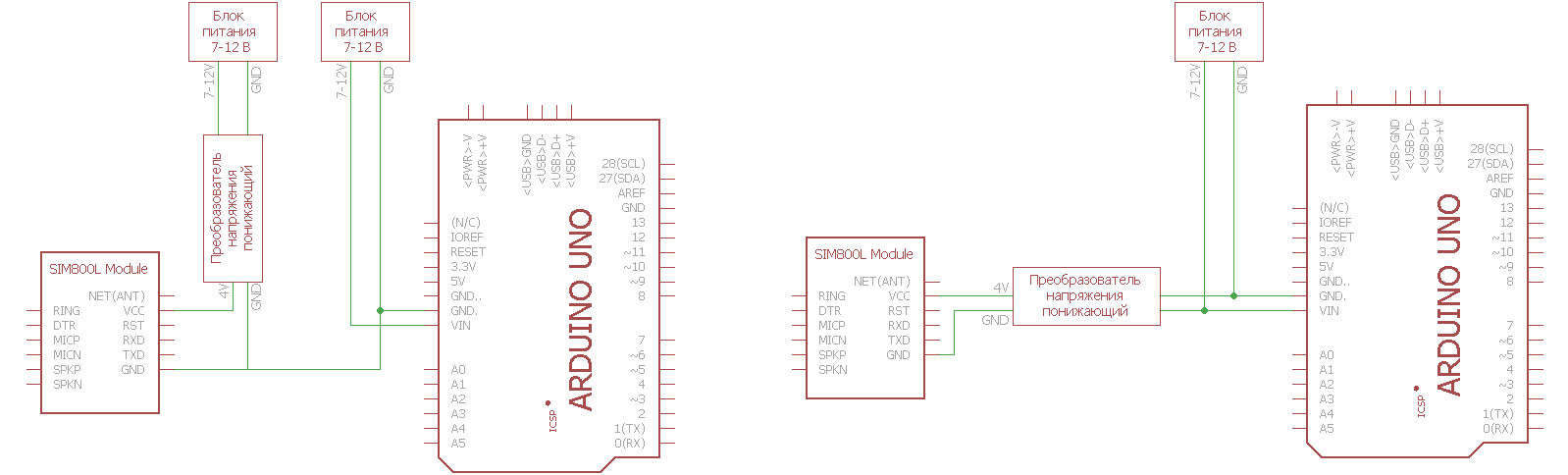
При подключении питания, не соответствующего требуемому (рисунок 6), модем выдает два типа сообщений. В случае, если питание находится на пороговом уровне (≤3.5 В, ≥4.4 В), модуль выдает предупреждение UNDER-VOLTAGE WARNING, OVER-VOLTAGE WARNING. В случаях, когда пороговый уровень превышен (≤3.4 В, ≥4.5 В), модуль сообщает об этом — UNDER-VOLTAGE POWER DOWN, OVER-VOLTAGE POWER DOWN и выключается.



**Рисунок 6 – Питания модуля [13]**

Для питания, модулю понадобится внешний источник питания и, если он не обеспечивает требуемого напряжения, то и DC-DC понижающий преобразователь напряжения.

Нельзя запускать модуль SIM800L от питания Arduino. Дело в том, что датчик бывает очень прожорлив (по даташиту максимальное потребление тока достигает 2А). А, как известно, [Arduino не в состоянии обеспечить его таким током](http://codius.ru/articles/195), поэтому очень вероятен сбой в работе — будет работать некорректно или модем, или Arduino, или оба вместе взятые (вполне вероятен выход Arduino из строя). Здесь может быть 2 выхода (рисунок 7) — либо SIM800L обеспечивается своим отдельным питанием (при этом земля GRN обоих источников должна быть общая), либо и Arduino, и GSM-модуль питаются одним мощным источником питания (7−12В).



**Рисунок 7 – Схемы по питанию [13]**

Перед подключением модуля к преобразователю напряжения, необходимо, при подключенном к преобразователю источнике питания, при помощи мультиметра установить заданное выходное напряжение.

Обмен данными:

Взаимодействие с модулем осуществляется по интерфейсу UART (Serial) при помощи специальных AT-команд. Для реализации обмена по UART-интерфейсу на пинах, отличных от стандартных RX (0)/TX (1) (они как правило заняты подключением к компьютеру), понадобится стандартная Arduino-библиотека SoftwareSerial.h

Диагностические команды:

Существует набор команд, при помощи которых можно определить состояние/готовность GSM-модуля совершать конкретные действия, а также получить другую полезную информацию диагностического характера. Некоторые из них представлены в таблице:

**Таблица №1** – **Команды GSM-модуля**

|  |  |  |  |
| --- | --- | --- | --- |
| Описание | Команда | Структура ответа | Пример ответа |
| Готовность модуля к работе | AT | OK | OK |
| Запрос информации об устройстве | ATI | *<info>* OK *<info>* — модель и версия модуля | SIM800 R14.18  OK |
| Запрос версии установленного ПО | AT+CGMR | *<revision>* OK *<revision>* — модель и версия ПО | Revision:1418B04SIM800L24  OK |

**4 Среда передачи данных**

RS-485 (Recommended Standard 485 или EIA/TIA-485) – рекомендованный стандарт передачи данных по двухпроводному полудуплексному многоточечному последовательному симметричному каналу связи. Совместная разработка ассоциаций: Electronic Industries Alliance (EIA) и Telecommunications Industry Association (TIA). Стандарт описывает только физические уровни передачи сигналов (т.е. только 1-й уровень модели взаимосвязи открытых систем OSI). Стандарт не описывает программную модель обмена и протоколы обмена [14]. RS-485 создавался для расширения физических возможностей интерфейса RS232 по передаче двоичных данных.

Достоинства стандарта RS-485**:**

1. Хорошая помехоустойчивость.
2. Большая дальность связи.
3. Однополярное питание +5 В.
4. Простая реализация драйверов.
5. Возможность широковещательной передачи.
6. Многоточечность соединения.

Недостатки RS485:

1. Большое потребление энергии.
2. Отсутствие сервисных сигналов.
3. Возможность возникновения коллизий.

## **4.1 Свойства стандарта RS-485**

Интерфейс RS-485 обладает следующими свойствами:

1. Двунаправленная полудуплексная передача данных**.**

Поток последовательных данных передаётся одновременно только в одну сторону, передача данных в другую сторону требует переключения приёмопередатчика. Приёмопередатчики принято называть "драйверами"(driver), это устройство или электрическая цепь, которая формирует физический сигнал на стороне передатчика.

2. Симметричный канал связи.

Для приёма/передачи данных используются два равнозначных сигнальных провода. Провода означаются латинскими буквами "А" и "В". По этим двум проводам идет последовательный обмен данными в обоих направлениях (поочередно). При использовании витой пары симметричный канал существенно повышает устойчивость сигнала к синфазной помехе и хорошо подавляет электромагнитные излучения, создаваемые полезным сигналом.

3. Дифференциальный (балансный способ передачи данных**).**

При этом способе передачи данных на выходе приёмопередатчика изменяется разность потенциалов, при передаче "1" разность потенциалов между AB положительная при передаче "0" разность потенциалов между AB отрицательная. То есть, ток между контактами А и В, при передаче "0" и "1", течёт (балансирует) в противоположных направлениях.

4. Многоточечность.

Допускает множественное подключение приёмников и приёмопередатчиков к одной линии связи. При этом допускается подключение к линии только одного передатчика в данный момент времени, и множество приёмников, остальные передатчики должны ожидать освобождения линии связи для передачи данных.

5. Низкоимпендансный выход передатчика.

Буферный усилитель передатчика имеет низкоомный выход, что позволяет передавать сигнал ко многим приёмникам. Стандартная нагрузочная способность передатчика равна 32-м приёмникам на один передатчик. Кроме этого, токовый сигнал используется для работы "витой пары" (чем больше рабочий ток "витой пары", тем сильнее она подавляется синфазные помехи на линии связи).

6. Зона нечувствительности**.**

Если дифференциальный уровень сигнала между контактами АВ не превышает ±200мВ, то считается, что сигнал в линии отсутствует. Это увеличивает помехоустойчивость передачи данных.

## **4.2 Технические характеристики RS-485**

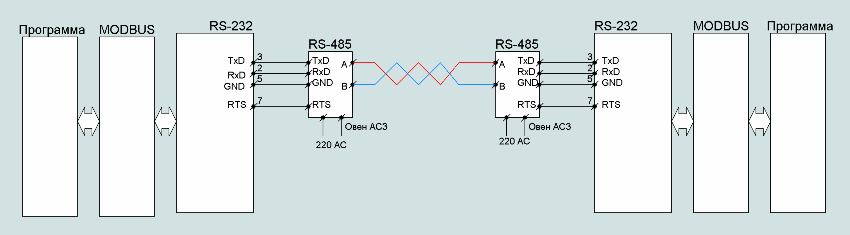
Интерфейс RS-485 обладает следующими техническими характеристиками:

* Допустимое число приёмопередатчиков (драйверов) 32
* Максимальная длина линии связи 1200 м (4000ft)
* Максимальная скорость передачи 10 Мбит/с
* Минимальный выходной сигнал драйвера ±1,5 В
* Максимальный выходной сигнал драйвера ±5 В
* Максимальный ток короткого замыкания драйвера 250 мА
* Выходное сопротивление драйвера 54 Ом
* Входное сопротивление драйвера 12 кОм
* Допустимое суммарное входное сопротивление 375 Ом
* Диапазон нечувствительности к сигналу ±200 мВ
* Уровень логической единицы (Uab) >+200 мВ
* Уровень логического нуля (Uab) <-200 мВ

Входное сопротивление для некоторых приёмников может быть более 12 кОм (единичная нагрузка). Например, 48 кОм (1/4 единичные нагрузки) или 96 кОм (1/8), что позволяет увеличить количество приёмников до 128 или 256. При разных входных сопротивлениях приёмников необходимо, чтобы общее входное сопротивление не было меньше 375 Ом.

## **4.3 Описание обмена данными по стандарту RS-485**

Каждый приёмопередатчик (драйвер) RS-485 может находиться в одном из двух состояний: передача данных или приём данных. Переключение драйвера RS-485 происходит с помощью специального сигнала. Например, на (рисунок 8) показан обмен данными с использованием преобразователя АС3 фирмы Овен. Режим преобразователя переключается сигналом RTS. Если RTS=1 (True) АС3 передает данные, которые поступают к нему от СОМ порта в сеть RS-485. При этом все остальные драйверы должны находиться в режиме приёма (RTS=0). По сути дела, RS-485 является двунаправленным буферным мультиплексированным усилителем для сигналов RS-232.



**Рисунок 8 - Пример использования преобразователя Овен АС3 [14]**

Ситуация, когда в одно время будет работать более одного драйвера RS-485 в режиме передатчика приводит к потере данных. Эта ситуация называется "коллизией". Чтобы коллизии не возникали в каналах обмена данными необходимо использовать более высокие протоколы (OSI). Такие, как MODBUS, DCON, DH485 и др. Либо программы, которые напрямую работают с RS-232 и решают проблемы коллизий. Обычно эти протоколы называют 485-тыми протоколами. Хотя на самом деле, аппаратной основой всех этих протоколов служит, конечно, RS-232. Он обеспечивает аппаратную обработку всего потока информации. Программную обработку потока данных и решение проблем с коллизиями занимаются протоколы высшего уровня (Modbus и др.) и ПО.

## **4.4 Основные принципы реализации протоколов верхнего уровня (типа MODBUS)**

Кратко рассмотрим эти протоколы, хотя они не имеют отношение к стандарту RS-485. Обычно протокол верхнего уровня включает в себя пакетную, кадровую или фреймовую организацию обмена. То есть, информация передаётся логически завершенными частями. Каждый кадр обязательно маркируется, т.е. обозначается его начало и конец специальными символами. Каждый кадр содержит адрес прибора, команду, данные, контрольную сумму, которые необходимы для организации многоточечного обмена. Чтобы избежать коллизий, обычно применяют схему "ведущий" (master)-"ведомый" (slave). "Ведущий" имеет право самостоятельно переключать свой драйвер RS-485 в режим передачи, остальные драйверы RS-485 работают в режиме приёма и называются "ведомыми". Чтобы "ведомый" начал предавать данные в линию связи "ведущий" посылает ему специальную команду, которая дает прибору с указанным адресом право переключить свой драйвер в режим передачи на определенное время.

После передачи разрешающей команды "ведомому", "ведущий" отключает свой передатчик и ждет ответа "ведомого" в течение промежутка времени, который называется "таймаут". Если в течении таймаута ответ от "ведомого" не получен, то "ведущий" снова занимает линию связи. В роли "ведущего" обычно выступает программа, установленная на компьютер. Существуют и более сложная организация пакетных протоколов, которая позволяет циклически предавать роль "ведущего" от прибора к прибору. Обычно такие приборы называют "лидерами", либо говорят, что приборы передают "маркер". Владение "маркером" делает прибор "ведущим", но он должен будет обязательно передать его другому прибору сети по определённому алгоритму. В основном, указанные выше протоколы, отличаются по этим алгоритмам.

Как мы видим, верхние протоколы имеют пакетную организацию и выполняются на программном уровне, они позволяют решить проблему с "коллизиями" данных и многоточечную организацию обмена данными.

# Применяемое оборудование и его обоснование в системах охранной, пожарной сигнализации и системы умного дома

Таблица компонентов представлена в приложении А. В проекте были использованы следующие компоненты:

NodeMcu V2 – Микроконтроллер семейства ESP, в проекте является главным звеном(сервером), его характеристики в несколько раз превосходят характеристики других компонентов в системе. Ещё один немаловажный плюс – это невысокая стоимость, распространённость, а также открытость системы.

Arduino Nano - Микроконтроллер семейства AVR, в проекте является ведомым(клиентом), его мощность и функционал слабее NodeMcu, но количества портов ввода-вывода делает его отличным кандидатом на роль клиента. Немаловажная особенность этого компонента эта низкая стоимость и широкое распространение. Из-за своих характеристик встроить Arduino Nano можно практически куда угодно с минимальными переделками. Также большое количество примеров и библиотек дает возможность не программисту работать с ним, без сильного ограничение в его функционале.

SIM800L-GSM модуль, позволяющий работать с беспроводной сетью операторов телефонной связи. Также у данного модуля есть возможность не только принимать данные, но и отправлять. Данная серия модуля одна из самых дешёвых из серии. Но из-за этого, удобной обвязки по питанию у неё нет. Также из минусов эксплуатации это бывают проблемы с оператором мобильной связи т.к. их система считает этот модуль модемом и блокирует её сеть. Из плюсов есть разъем внешней антенны, используется для улучшения сигнала вдали от вышек.

XL6009 – Повышающий модуль напряжение, способен выдавать стабильное напряжение на выходе (от 1.25 до 35 вольт). Это значит, что на выходе импульсного преобразователя напряжения XL6009, всегда будет заданное стабильное напряжение. Притом, что на его входе может быть напряжение как меньше, так и больше чем на его выходе. Данный модуль имеет тепловую защиту и короткого замыкание. В проекте применяется для питания GSM модуля.

SR501 – Инфракрасный датчик движения, датчик используется для обнаружения движения объектов, излучающих инфракрасное излучение. На плате расположены два потенциометра, с помощью первого настраивается дистанция обнаружения объектов (от 3 до 7 м), с помощью второго – задержка после первого срабатывания датчика (5 - 300 сек). На модуле установлена линза Френеля, которая используется для увеличения радиуса обзора, SR501 часть системы охранной безопасности.

LM393 – Модуль предназначен для обнаружения пламени с длиной волны в диапазоне от 760nm-1100 нм. Главным преимуществом является то, что он способен обнаружить пламя, которое не выделяет дым. Часто такие датчики устанавливаются возле каминов или за камерой горения и при выходе огня за данную область формируется тревожный сигнал. Данный датчик работает в пожарной системе.

HB100 – датчик, принцип действия которого базируется на эффекте Доплера. Датчик работает по принципу эффекта Доплера – изменение частоты и длины волны, регистрируемых приёмником, вызванного движением их источника и/или движением приёмника. Главное его преимущество в том, что его можно устанавливать скрыто т.к. волны от него проходят через всё кроме металла. А скрытое размещение датчиков затрудняет работу злоумышленников.

Зуммер – сигнальное устройство, электромеханический представляет собой реле с парой нормально замкнутых контактов. При появлении тока в цепи катушка реле возбуждает магнитное поле, под действием магнитного поля контакты реле размыкаются. При размыкании контактов реле цепь разрывается, и контакты замыкаются. В проекте служит как сигнальное устройство.

Источник бесперебойного электропитания – источник электропитания, обеспечивающий при отключении основного источника мощности питания, а также защиту от помех в сети основного источника. В проекте работает в паре с 4мя Li-ion батареями 1860 подключенных параллельно. Поддерживает питание в 9В.

Датчик bme280 содержит в себе 3 устройства – для измерения давления, влажности и температуры. Разрабатывался для малого потребления тока, высокой надежности и долгосрочной стабильной работы. Датчик фирмы Bosch Sensortec. Работает по интерфейсу I2C. В проекте работает в системе умного дома.

SW-420 – Датчик вибрации, собран на основе датчика вибрации пружинного типа SW-420 и компаратора LM393. В состоянии покоя датчик находится в разомкнутом состоянии и ток через него не протекает, а на выходе имеем логическую единицу. При внешнем воздействии (толчке, ударе и т.п.) пружина раскачивается и кратковременно замыкает контакты, а на выходе вывода получаем логический ноль. Используется часто в авто сигнализациях. Также устанавливается на дверь или сейф, и при небольшом приложении силы к этому объекту датчик сработает. Используется в системе безопасности.

NANO V3.0 – Плата расширения ввода/вывода Arduino Nano. Контроллеры Arduino Nano очень маленькие. Плата расширения для Arduino NANO V3.0 разработана для облегчения подключения и позволяет исправить этот недостаток. На ней используются быстрозажимные винтовые клеммы, которые надежно держат контакт что повышает надежность системы.

DS3231 – модуль реального времени, построен на микросхеме DS3231, в котором установлен кварцевый генератор и датчик температуры, который компенсирует изменения температуры, поэтому время отсчитывается более точно. Погрешность составляет ±2 минуты за год. Подключается к плате по интерфейсу I2C.В системе служит для вывода времени.

TF card reader – Модуль чтения записи MicroSD TF используется для подключения карт памяти (записи, чтения и хранения данных) в проектах на микроконтроллерах. Подключается по SPI интерфейсу. В проекте служит для записи Log данных.

Shield For V2 – Плата расширение для NodeMcu, пиновые контакты на плате продублированы, также есть дополнительные контакты для питания различных датчиков. В проекте повышает надёжность системы и удобность монтажа.

RS485 – Модуль передачи данных, по промышленной технологии. В проекте был выбран из-за способности работать на больших расстояниях и помехозащищённости.

Gercon – электромеханическое коммутационное устройство, изменяющее состояние подключённой электрической цепи при воздействии магнитного поля от постоянного магнита. Устанавливается на двигающиеся объекты, данный вид датчиков очень распространён. В проекте используется в системе безопасности.

MQ-2 – датчик определит концентрацию углеводородных газов (пропан, метан, н-бутан), дыма (взвешенных частиц, являющихся результатом горения) и водорода в окружающей среде. Датчик MQ-2 относиться к полупроводниковым приборам. Принцип работы датчика основан на изменении сопротивления тонкопленочного слоя диоксида олова SnO2 при контакте с молекулами определяемого газа. Из минусов – это довольно большое потребление из-за потребности накала диоксида олова. Используется в пожарной системе.

W5500 Ethernet – Ethernet модуль для работы с web, он берет на себя обработку пакетов данных для работы в Ethernet сетях. Подключен к микроконтроллеру по SPI интерфейсу. На модуле реализован весь стек протоколов OSI. Данный модуль разгружает систему и повышает безотказность системы.

Релейный модуль 4CH – коммутационное устройство (КУ), соединяющее или разъединяющее цепь электрической или электронной схемы при изменении входных величин тока. Модуль позволяет управлять микроконтроллеру силовой нагрузкой. В данном модуле присутствует гальваническая развязка и индикаторы включения. Используется в системе умного дома.

Клавиатура 4x4 – Если требуется использовать много кнопок, а иногда это просто невозможно из-за количества портов ввода-вывода. Один из вариантов экономии входов Arduino при подключении кнопок – это использование матричных клавиатур. Матричная клавиатура 4x4 в проекте служит для ввода информации.

LCD2004 + I2C – Жидкокристаллический экран. Самым главным недостатком этого экрана является тот факт, что дисплей имеет 16 цифровых выводов, из которых обязательными являются минимум 6, чтобы обойти эту проблему в экран добавлен модуль I2C (последовательная асимметричная шина), теперь экрану нужно лишь два контакта для вывода информации и два для питания. На экран можно вывести 20 символов на 4 линии. В проекте используется для вывода информации.

LM2596 – Понижающий DC-DC преобразователь на основе чипа LM2596S-ADJ. Выходное напряжение модуля регулируется подстрочным резистором. В проекте используется для преобразования питание от ИБП на 9V на нужные для микроконтроллера и его обвязки 5V. Присутствует защита от переполюсовки в качестве одного диода.

# 6 Расчёт основных числовых характеристик проектируемой системы безопасности

Создать проблему в корректной работе систем умного дома могут две причины – внезапное прекращение подачи питания и скачки напряжения. Для предотвращение этого в систему будет встроена ИБП. В случае отключения тока в сети благодаря встроенной батарее они могут обеспечивать электричеством микроконтроллер и его компонентов на протяжении временного промежутка от нескольких минут до нескольких часов. Также ИБП защищают и от скачков напряжения. Для подбора ИБП необходимо рассчитать общий потребляемый ток в системе. Все значения будут взяты по максимальному потреблению.

Расчёт тока микроконтроллеров и его обвязки:

Arduino Nano – 20 мА.

LCD дисплей – 500 мА.

Nodemcu – 70 мА.

Zummer – 20 мА.

DS3231 – 4 мА.

TF кардридер– 3 мА.

Модуль RS485 – 25 мА (2).

W5500 Ethernet – 200 мА.

Датчик влажности – 5 мА.

Общий потребляемый ток микроконтроллеров и его обвязки: 847 мА.

Расчёт тока системы пожарной безопасности:

MQ-2– 150 мА.

LM393– 10 мА.

Общий потребляемый ток пожарной безопасности: 160 мА.

Расчёт тока системы охранной безопасности:

SR501– 50 мА.

SW-420– 5 мА.

Gercon – 2 мА.

Общий потребляемый ток системы охранной безопасности: 57 мА.

Расчет тока системы умного дома:

Релейный модуль 4CH – 240 мА.

BME280 – 20 мА.

Общий потребляемый ток системы умного дома: 260 мА.

Ток потребление всей системы: 1324 мА = 1.3 А. Это ток при пиковой нагрузки, в обычной нагрузке потребление будет на 60% меньше.

Потребляемая мощность в пике, системы микроконтроллеров и его обвязки 6,5 Вт при 5 В.

Теоретически можно рассчитать время автономной работы по формуле: t=C /I, где:

t – продолжительность «работы» аккумуляторной батареи (ч);

С – емкость аккумулятора (мА\*ч);

I – ток нагрузки (мА).

Но, проблема заключается в том, что при увеличении нагрузки на АКБ, также увеличивается скорость разряда емкости. Например, аккумулятор емкостью 800 мА\*ч при нагрузке 800 мА, будет работать не час, а несколько меньше. В связи с этим используется следующая формула для расчета: t=C /(I × K), то есть, добавлен коэффициент K это табличное значение, подбираемое в зависимости от типа АКБ. В данном случае используется

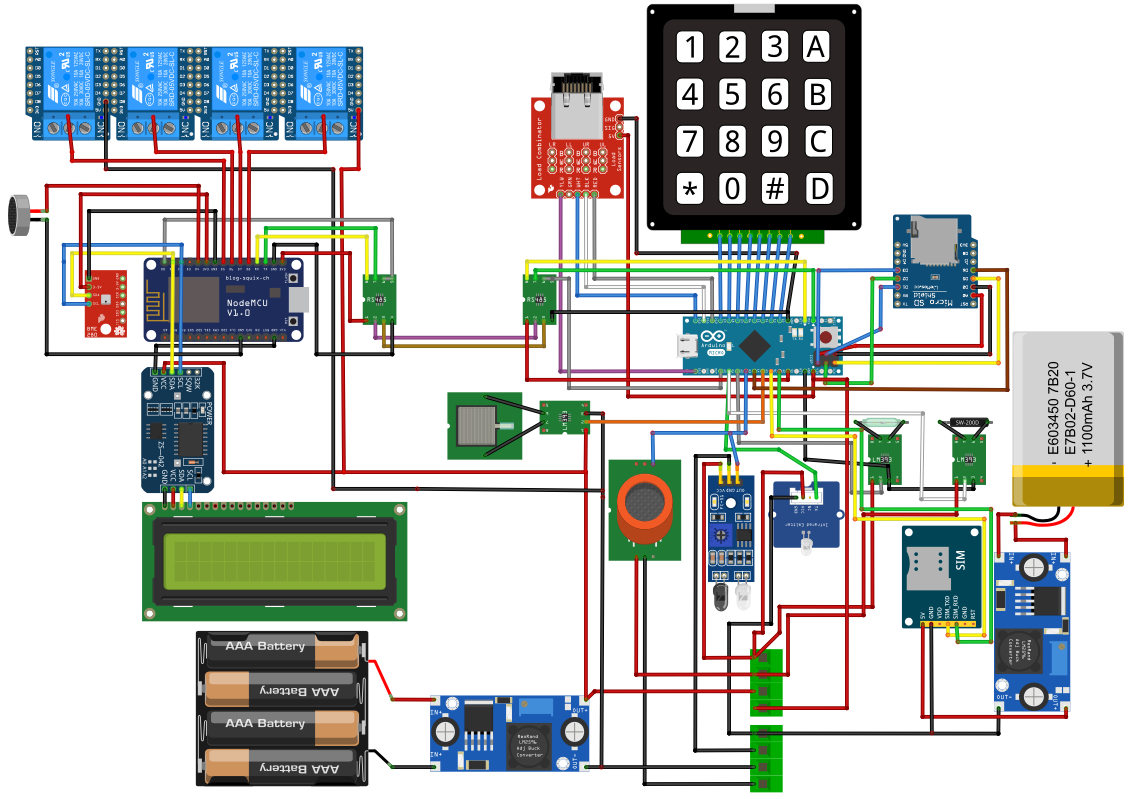
усредненное значение коэффициента для Li-ion K, равное 1,1. Подставив свои значение в форму 1200/(1324 × 1,12)= 8 часов работы от источника бесперебойного питания при максимальной нагрузке системы. В обычном режиме эксплуатации выйдет 22 часа.

У GSM модуля свой источник питания и батарея 5000 мА\*ч, его потребление в пике может составлять до 2А, а в обычном режиме 140 мА.

Что дает также 22 чеса при возможных скачках потребления.

# 7 Программа взаимодействие управляющего контролера с датчиками и управляющими устройствами

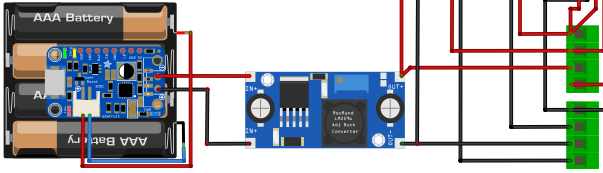
Для системы умного дома, который включает в себя разные подсистемы. С последующей обработкой всех процессов в микроконтроллере. Была спроектирована моделированная схема (рисунок 9).



**Рисунок 9 – Моделированная схема**

Каждая система безопасности: пожарная, умного дома, охранной разделены на разные каналы. Это сделано, для того чтобы повысить надежность системы. На каждый канал свои датчики, также к одному каналу можно подключить несколько датчиков используя аналого-цифровое преобразование. Для такого подключения понадобятся резисторы.

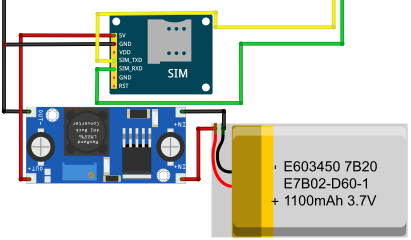
Электропитание системы идет от блока, выдающего 9V, далее который понижается до 5V (рисунок 10) и распределяется по всем модулям.



**Рисунок 10 – Схема питания**

Главное в постройке данных электронных схем является общий минус заземления. В данном проекте используется источник бесперебойного питания, работающий на 4-х параллельных литий-ионных батареях размера 18650.

У GSM модуля в отличие от других схем, требование по питанию отличается (рисунок 11) от общей схемы.



**Рисунок 11 – Схема питания SIM800L**

Данный модуль не будет работать от общих 5ти вольт. Поэтому у него есть свой повышающий блок, работающий от батареи и реализующий 4 вольта. Для корректной работы нужно объединить минусовой контакт питания с минусом основного источника.

Связь между микроконтроллерами реализовано с помощью промышленного протокола RS-485 (рисунок 12).

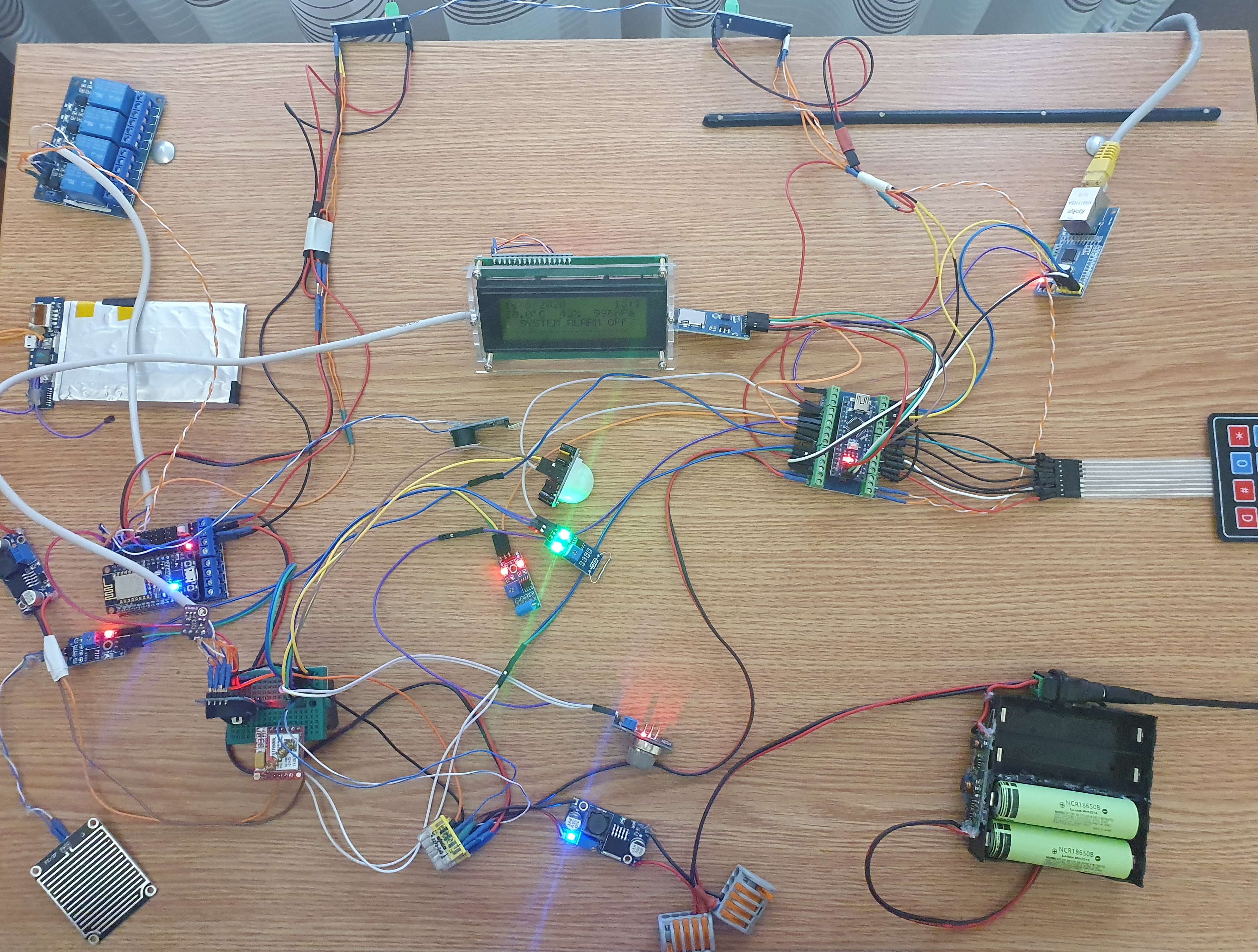


**Рисунок 12 – Схема подключение RS-485**

Соединив вход DE со входом RE и подключили их к цифровому выводу микроконтроллера, то подавая на него сигнал HIGH, модуль будет работать только на передачу, а подав сигнал LOW, только на приём. Таким способам реализуем полудуплексный режим работы.

# 8 Демонстрация возможностей системы

Проведя работу со смоделированной схемой, приступил к сборке готовой макетной схемы (рисунок 13). Для сборки понадобилась 45 проводов типа мама-папа, 20 проводов типа мама-мама и 7 проводов типа папа-папа.



**Рисунок 13 – Макетная схема**

Общая протяжённость всех проводов в проекте составила 16 метров [1]. Многие провода пришлось паять вручную, так как стандартные не подходили по длине или не было достаточно в наличии.

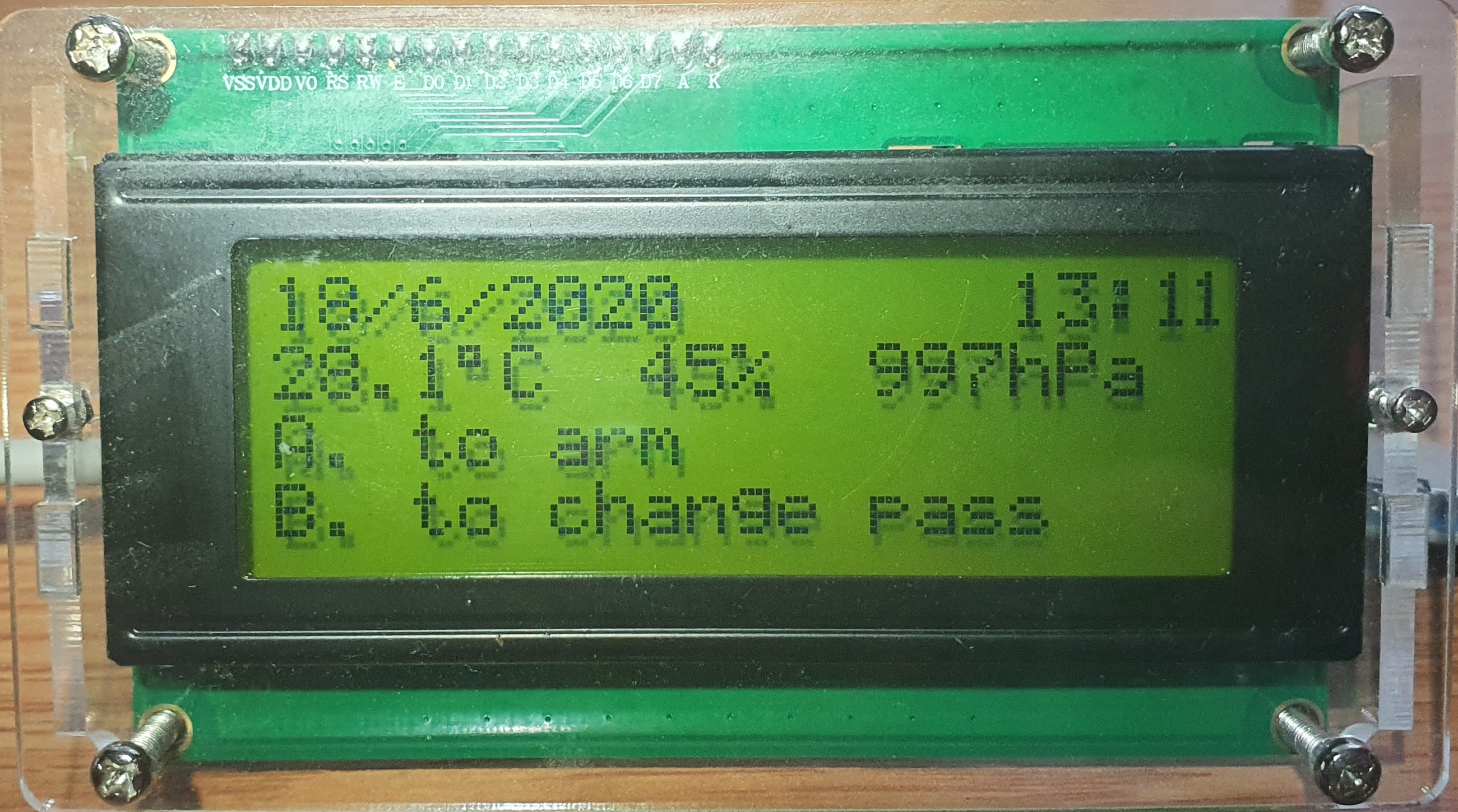
Спроектировав сначала моделированную схему с помощью программы, удалось избежать ошибок [6], которые могли привести к короткому замыканию и выхода из строя датчиков и микроконтроллеров.

Первое, что видит пользователь при включении (рисунок 14) системы на LCD экране – это дата, время, температура, влажность и давление. Данный интерфейс, дает пользователю системы много полезной информации. Даже при комфортной температуре не всегда приятно находится в помещении из-за недостаточной влажности. Изменение показателя давления, означает о скором изменении погоды.

.

**Рисунок 14 – Интерфейс LCD**

Нижние две строки интерфейса представлены как меню. При нажатии на клавишу А система через 10 секунд, давая пользователю выйти из помещения перейдет в охранный режим. При нажатии на B, перейдёт в меню смены пароля, для изменения пароля нужно ввести первоначальный пароль. В данном меню ещё отображается состояние системы, заменяя визуально меню время от времени.



Датчик геркон устанавливается на входную дверь. При входе пользователя на LCD экран выведется меню с вводом пароля и при правильном вводе, охранная система отключится. Если сработаю другие охранные датчики, то система включит извещение о проникновении нарушителя также передаст информацию на охранный Web интерфейс и отправит смс оповещение.

Состояние системы также дублируется на охранном Web интерфейсе. На который можно перейти, введя локальный айпи адрес. По задумке, к данному Web интерфейсу подключены охранные компании. Состояние - «System disabled» показывает, что охраняемому объекту ничего не угрожает. Остальные состояния Web интерфейса:

* «Security system Attact» показывает, что на объект совершена атака, то есть при активации системы сработал датчик.
* «FIRE» показывает, что на объекте сработал пожарный датчик.
* «Water leak» показывает, что на объекте сработал датчик протечки воды.

При изменении состояния охранного Web интерфейса, такие же данные передаются пользователю по смс сообщениям. Все действия который происходят в системе, записываются на карту памяти в лог файл формата txt.

Вида:

* Logg.txt – записывает время работу системы в штатном режиме.
* LogFire.txt - записывает время работу системы при активации пожарной системы.
* LogArmed.txt - записывает время работу системы при срабатывании охранных датчиков

Время в этих файлах хранится в секундах, которое берётся от функции Millis().

# 9 Разбор основных частей программного кода

При разработке программного кода, для достижений оптимальности были использованы разные технологии программирования, реализованных на языке СИ++ с помощью среды разработки Microsoft Studio Visual.

Первые строки кода начинаются с подключением библиотек это набор классов, компонентов или модулей для разных часто использующихся задач или для работы компонентов. Весь исходный код для ESP 12 представлен в приложении В.

#include <Arduino.h>

#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

#include "Adafruit\_Keypad.h"

#include <NewPing.h>

#include <ESP8266WiFi.h>

#include "Adafruit\_MQTT.h"

#include "Adafruit\_MQTT\_Client.h"

#include <SPI.h>

#include <Adafruit\_BME280.h>

#include <Adafruit\_Sensor.h>

Для лучшей работы скетча использовал вместо функции delay() функцию millis() [2] который не останавливает работу всего микроконтроллера. Millis() работает на системном таймере (Timer 0) и переполняется раз в 50 суток.

unsigned long timing; // Переменная для хранения точки отсчета

if (millis() - timing > 5000){

timing = millis() // сброс таймера

}

В теле данного условия код выполнится каждые 5 секунд. Не прерывая работу остального скетча.

Для отправки и приема данных между микроконтроллерами по протоколу RS-456 передаём состояние портов у ESP это D1 у Arduino это порт A1.

digitalWrite(PIN\_direction\_TX\_RX, HIGH); // переводим модуль в режим передачи данных

delay(1);

Serial.write('T'); // отправляем символ кнопки в последовательный порт (для вывода цифры кнопки используйте KB.getNum)

delay(1);

digitalWrite(PIN\_direction\_TX\_RX, LOW); // переводим модуль в режим приёма данных

delay(1);

Таким способам реализуется полудуплексная связь. Весь исходный код для Arduino Nano представлен в приложении C.

Для безопасного ввода и хранения пароля использую строго типизированный алгоритм и использую её в функции:

if (keypressed == '0' || keypressed == '1' || keypressed == '2' || keypressed == '3' ||keypressed == '4' || keypressed == '5' || keypressed == '6' || keypressed == '7' ||

keypressed == '8' || keypressed == '9' )

Таким образом в данную функцию которая хранит пароль не получится записать буквы или символов вызвав при этом сбой программы.

Для работы с Ethernet сетью с микроконтроллером W5500 подключаем библиотеку #include <Ethernet2.h>, следующий шаг — это создание Mac и IP адреса:

byte mac[] = {

0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED};

IPAddress ip(192, 168, 1, 177);

EthernetServer server(80);

Строка server(80) – это подключение пора для протокола HTTP по умолчанию. После подключаемся к клиенту:

while (client.connected()) {

if (client.available()) {

char c = client.read();

Serial.write(c);

if (c == '\n' && currentLineIsBlank) {

client.println("HTTP/1.1 200 OK");

client.println("Content-Type: text/html");

Ниже формируем страницу на html языке.

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

client.println("<html>");

client.print("<h1>Security system</h1>");

client.print("System disabled ");

client.println("</html>");0

Обмен данными между микроконтроллерами происходит с помощью передачи флагов в Serial порт, а затем по RS-485. Это сделано для увеличения безопасности и надежности, также данный тип связи позволяет работать на большом расстоянии. Передача флагов для установки состояний происходит в определённом порядке. Даже если злоумышленник перехватит флаги текущей сессии и переотправит их, проблем это не вызовет.

keypressed = Serial.read(); //Read the pressed button

if (keypressed =='A'){ //If A is pressed, activate the system

buzzer.sound(500, 200);

systemIsArmed(); //by calling the systemIsArmed function

}

else if (keypressed =='B'){//If B is pressed, change current password

doublecheck=0;

buzzer.sound(500, 200);

storedPassword=false;

if(!changedPassword){//By calling the changePassword function

changePassword();

}

}

if (keypressed == 'F' ){

alarmFunctionFire(); //Call alarm!

}

Каждый флаг 'A', 'B', 'F' и др. включают или выключают определённую функцию, которая в последующем работает с другими.

В коде присутствует код для работы Eeprom – это энергонезависимая память. Которая позволяет сохранять изменение пароля в памяти микроконтроллера.

/\*String qsid = password;

int charLength=qsid.length();

EEPROM.begin(512); // Ячейка памяти

Serial.println("writing eeprom ssid:");

for (int i = 0; i < qsid.length(); ++i) {

EEPROM.write(i, qsid[i]);

Serial.print("Wrote: ");

Serial.println(qsid[i]);

}

Serial.println("Reading EEPROM ssid");

for (int i = 0; i < charLength; ++i) {

esid += char(EEPROM.read(i));

}

EEPROM.end();

esid.trim();

Serial.println(esid.length());

Serial.print("SSID: ");

Serial.println(esid);

password = esid ;

\*/

В данном проекте она не работает, так как при тестировании данные ячейки были зациклены перезаписью на большой промежуток времени и память повреждена. Но на другом микроконтроллере ESP всё работает.

В данном проекте на ESP не установлены охранные датчики. Но их можно реализовать раскомментировав строки и установив нужные датчики.

lcd.print("SYSTEM IS ARMED!");

lcd.setCursor(0,3);

lcd.print("----------------");

//int door = digitalRead(doorMagSen);//Read magnetic sensros

//int window = digitalRead(windowMagSen);

//int curr\_distanse = sonar.ping\_cm();

По умолчанию они отключены т. к. не используются в проекте.

Управлением состоянием системы управляются с помощью условий включение функций.

if (armed){

systemIsArmed(); //Run function to activate the system

}

else if (!armed){

systemIsUnarmed(); //Run fuction to de activate the system

}

Которые в свою очередь управляются флагами другого микроконтроллера.

if(Serial.available()>0){ // если в последовательном порту есть данные

char n = Serial.read();

if( n =='G' ){

Serial.print(n);

armed = true;

}

if( n =='T' ){

armed = false;

}

}

Они устанавливают состояние переменных armed.

Чтобы система ещё больше отображала полезную информацию, в неё добавлена дата и время с выводов на экран. Также в ней выводится на экран информация по температуре, влажности и давлении от модуля bme280.

void dataTemp (){

RtcDateTime currentTime = rtcObject.GetDateTime(); //get the time from the RTC

char str[20]; //declare a string as an array of chars

sprintf(str, "%d/%d/%d %d:%d", //%d allows to print an integer to the string

currentTime.Day(), //get year method

currentTime.Month(), //get month method

currentTime.Year(), //get day method

currentTime.Hour(), //get hour method

currentTime.Minute() //get minute method

);

while (dT == 0)

{

lcd.setCursor(0,0);

lcd.print(str); //print the string to the serial port

lcd.setCursor(0,1);

lcd.print(temp\_event.temperature,1);

lcd.print((char)223);

lcd.print("C");

lcd.setCursor(8,1);

lcd.print(humidity\_event.relative\_humidity,0);

lcd.print("%");

lcd.setCursor(13,1);

lcd.print(pressure\_event.pressure,0);

lcd.print("hPa");

dT++;

}

dT = 0;

}

Эта функция работает с модулем DS3231 у которого есть отдельное от микроконтроллера питание, на него не действует перезагрузки или сбои в работе остальных систем. Строчка кода ниже устанавливает дату и время при записывании скетча [3] в микроконтроллер.

RtcDateTime currentTime = RtcDateTime(\_\_DATE\_\_, \_\_TIME\_\_);

## **9.1 Протокол MQTT**

Подробнее о технологии MQTT который используется для системы умного дома на NodeMCU и реле, и датчики. MQTT — это протокол обмена сообщениями по шаблону издатель-подписчик (pub/sub). он особенно актуален благодаря росту IoT устройств которым и является ESP-12.

Система связи, построенная на MQTT, состоит из сервера-издателя [15], сервера-брокера и одного или нескольких клиентов. Издатель не требует каких-либо настроек по количеству или расположению подписчиков, получающих сообщения. Кроме того, подписчикам не требуется настройка на конкретного издателя. В системе может быть несколько брокеров, распространяющих сообщения. MQTT передаёт учётные данные безопасности открытым текстом, иначе не поддерживается аутентификация или функции безопасности. Поэтому при активации охранной системы связь с сервером прерывается.

Для удаленной работы MQTT в проекте использую сервис с созданным аккаунтом от Adafruit [17]. Он не единственный, таких сервисов много.

Для работы сначала нужно получить доступ к интернету, делаем это подключением к беспроводной точки доступа.

#define WLAN\_SSID

#define WLAN\_PASS

Далее необходимо подключится и авторизоваться к серверу

#define AIO\_SERVER "io.adafruit.com"

#define AIO\_SERVERPORT 1883 // use 8883 for SSL

#define AIO\_USERNAME "AMD12"

#define AIO\_KEY "aio\_UHwC48AOrT6WueGvAVHhaoUplRvp"

Строка с AIO\_KEY в ней вводим ключ идентификации для работы с удаленным сервером.

Каждая ячейка, созданная в adafruit имеет свою ссылку чтобы посылать в них данные используем данные команды:

// Setup the MQTT client class by passing in the WiFi client and MQTT server and login details.

Adafruit\_MQTT\_Client mqtt(&client, AIO\_SERVER, AIO\_SERVERPORT, AIO\_USERNAME, AIO\_USERNAME, AIO\_KEY);

Adafruit\_MQTT\_Publish valLight1 = Adafruit\_MQTT\_Publish(&mqtt, AIO\_USERNAME "/feeds/lampa-1");

Adafruit\_MQTT\_Publish valLight2 = Adafruit\_MQTT\_Publish(&mqtt, AIO\_USERNAME "/feeds/lampa-2");

Adafruit\_MQTT\_Publish valLight3 = Adafruit\_MQTT\_Publish(&mqtt, AIO\_USERNAME "/feeds/lampa-3");

Adafruit\_MQTT\_Publish valLight4 = Adafruit\_MQTT\_Publish(&mqtt, AIO\_USERNAME "/feeds/lampa-4");

Adafruit\_MQTT\_Publish valLight5 = Adafruit\_MQTT\_Publish(&mqtt, AIO\_USERNAME "/feeds/temp");

Adafruit\_MQTT\_Publish valLight6 = Adafruit\_MQTT\_Publish(&mqtt, AIO\_USERNAME "/feeds/humidit");

С помощью отправки состояний мы будем видим обратную связь на изменение состояний, так мы можем видеть состояние датчиков температуры и влажности и состояние реле.

Далее происходит прием данных при изменение их на сервере adafruit, так мы можем управлять состоянием реле в проекте.

Adafruit\_MQTT\_Subscribe light1 = Adafruit\_MQTT\_Subscribe(&mqtt, AIO\_USERNAME "/feeds/lampa-1");

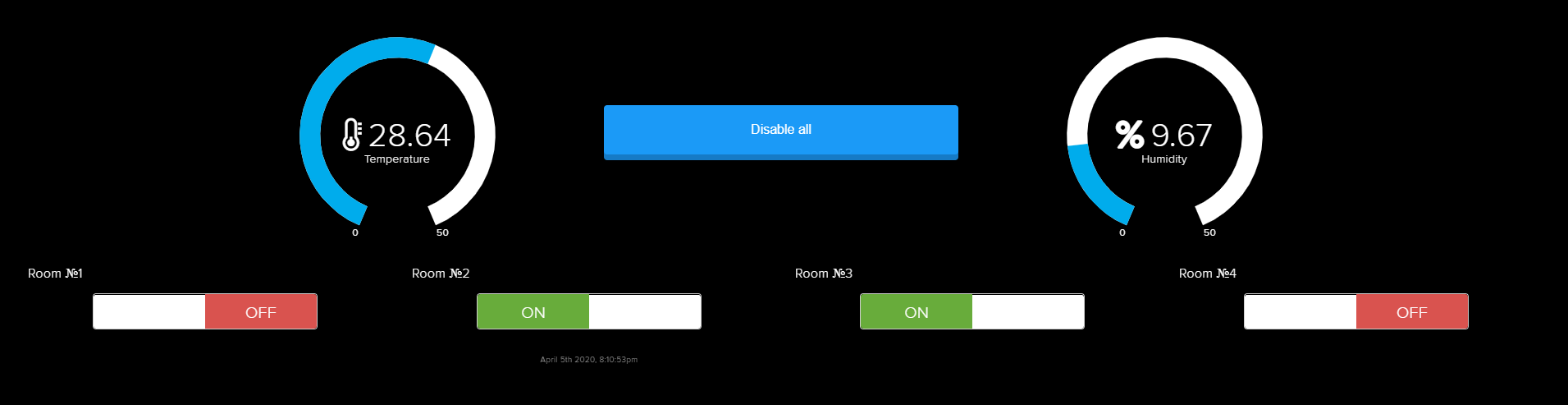
Adafruit\_MQTT\_Subscribe light2 = Adafruit\_MQTT\_Subscribe(&mqtt, AIO\_USERNAME "/feeds/lampa-2");

Adafruit\_MQTT\_Subscribe light3 = Adafruit\_MQTT\_Subscribe(&mqtt, AIO\_USERNAME "/feeds/lampa-3");

Adafruit\_MQTT\_Subscribe light4 = Adafruit\_MQTT\_Subscribe(&mqtt, AIO\_USERNAME "/feeds/lampa-4");

Adafruit\_MQTT\_Subscribe light5 = Adafruit\_MQTT\_Subscribe(&mqtt, AIO\_USERNAME "/feeds/disable-all");

Так выглядит Web интерфейс:



**Рисунок 15 – Web интерфейс**

Данный протокол реализован и в мобильном приложении (рисунок 15) под управлением любых современных операционных системах



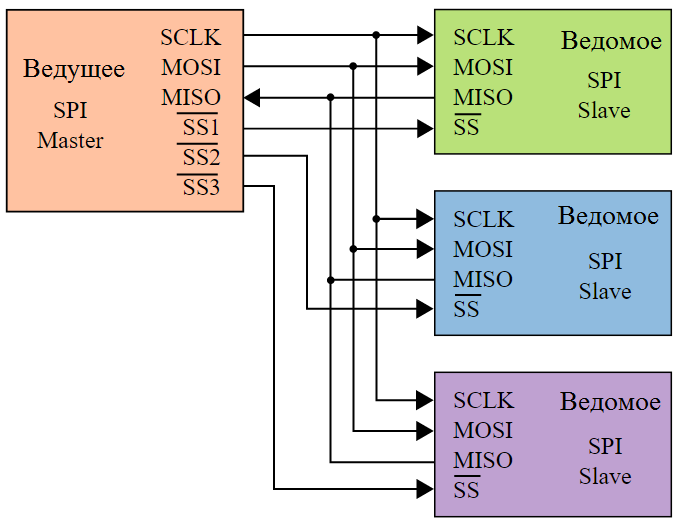
**Рисунок 16 – Android приложение**

В данном случае используется приложение IoT OnOff (рисунок 16) под управлением системы анероид. Данный протокол очень удобен для разработчика т.к. предоставляет широкий выбор инструментов.

## **9.2 Интерфейс SPI**

Последовательный периферийный интерфейс (SPI) - это синхронный протокол последовательной передачи данных [16], используемый для связи микроконтроллера с одним или несколькими периферийными устройствами. Интерфейс SPI отличается относительно высокой скоростью и предназначен для связи близко расположенных устройств. Он также может использоваться для взаимодействия двух микроконтроллеров.

Согласно протоколу SPI, одно из взаимодействующих устройств (обычно микроконтроллер) всегда является ведущим и контролирует ведомые



**Рисунок 17 – Интерфейс SPI [16]**

периферийные устройства (рисунок 17). Как правило, все взаимодействующие устройства объединены тремя общими линиями:

MISO (Master In Slave Out) - линия для передачи данных от ведомого устройства (Slave) к ведущему (Master),

MOSI (Master Out Slave In) - линия для передачи данных от ведущего устройства (Master) к ведомым (Slave),

SCK (Serial Clock) - тактовые импульсы, генерируемые ведущим устройством (Master) для синхронизации процесса передачи данных.

Помимо перечисленных, на каждое устройство отводится отдельная линия:

SS (Slave Select) - вывод, присутствующий на каждом ведомом устройстве. Он предназначен для активизации Мастером того или иного периферийного устройства.

Периферийное устройство (Slave) взаимодействует с ведущим (Master) тогда, когда на выводе SS присутствует низкий уровень сигнала. В противном случае данные от Master-устройства будут игнорироваться. Такая архитектура позволяет взаимодействовать с несколькими SPI-устройствами, подключенными к одной и той же шине: MISO, MOSI и SCK.

В проекте микроконтроллер Arduino Nano использует интерфейс SPI для логирования времени работы системы и Web интерфейс который отображает состояние системы в реальном времени. Что бы два устройства не конфликтовали между собой управляющий вывод SS переведён в низкий уровень, для устройства которому требуется SPI в определенный момент времени. Когда как на другом устройстве, вывод SS переведен в высокое положение, что переводит его в нерабочее состояние.

void logDis() {

digitalWrite(10 , HIGH); // Выкличем Web страницу

digitalWrite(A4, LOW); //Включаем SD

SD.begin(A4);

myFile = SD.open("Logg.txt", FILE\_WRITE);

if (myFile) {

myFile.println("------------");

myFile.print( "Seconds ");

myFile.print( millis());

myFile.println("");

myFile.close();

delay(10);

WebUnarmed();

}

digitalWrite(A4, HIGH); // Выключаем SD

digitalWrite(10 , LOW); // Включаем Web страницу

}

Данный протокол широко используется, его скорость выше чем у протокола I2C. Из недостатков, требуется большее количество портов, сложнее в реализации при большом количестве устройств по сравнению с I2C.

# Заключение

Спроектировав комплексную систему умного дома, используя современные датчики, научились проектировать гибкую систему безопасности, комбинируя различные датчики и разрабатывая их программное обеспечение. Система способна удовлетворить любого пользователя. Спроектировав системы сигнализации:

* Пожарной
* Охранной
* Систему умного дома

Вместе они образуют современную интегрированную систему безопасности (ИСБ). Научились подбирать подходящие компоненты к поставленной работе. В ходе проектирования ознакомились с различными датчиками и их работой, различных тонкостей и нюансов их правильной установки. Научились правильно подходить к выбору компонентов.

Получил опыт в работе с микроконтроллерами семейства AVR и ESP. Для этого разработали программное обеспечение, которое контролирует все подключенные системы с возможностью их дальнейшего расширения. В построении принципиальной схемы использовали программу Fritzing. Данная программа была выбрана из-за простоты использования и возможности бесплатного пользования. Для написания ПО была выбрана Microsoft Studio Visual Code, так как удовлетворяет всем потребностям в разработке. Графические приложения были нарисованы при помощи AutoCAD 2017. Для реализации таких больших систем у которых при разных задачах должны формироваться приоритеты и по возможности реализовывать много поточность лучше подходят семейства микроконтроллеров STM.

В проекте было настроено Android приложение, которое одновременно работает с Web интерфейсом умного дома. Данная система не только выдает полезную информацию, но и управляет реле в доме.

Получил опыт в разработке и построение IoT (Интернет вещей) устройств, которые могут взаимодействовать друг с другом или с внешней средой. Использовал протоколы взаимодействие доступных в микроконтроллерах.

Тема дипломной работы является разработка проекта электронной системы безопасности на базе программируемых микроконтроллеров AVR. Для выполнения работы была подобрана нужная литература. Литература помогла с подбором нужных компонентов и в правильном написании исходного кода для микроконтроллеров.

Для того, чтобы данный проект можно было повторить, исходный код микроконтроллеров и все библиотеки для работы модулей выложены на GitHub [17]. В данном репозитории будет и описание работы системы, для последующего конфигурирования её под нужные задачи. Данный проект доведен до логического завершения.

# Список использованных источников

1. Ю. Ревич. Занимательная электроника, БХВ-Петербург, 2015.
2. С.Монк. Программируем Arduino: профессиональная работа со скетчами. Питер, 2017.
3. С.Монк. Программируем Arduino: основы работы со скетчами. Питер, 2017.
4. Д.Бокселл. Изучаем Arduino: 65 проектов своими руками. Питер, 2017.
5. В. Петин, Проекты с использованием контроллера Arduino, 2-е издание, БХВ-Петербург, 2015.
6. Т. Карвинен, К. Карвинен, В. Валтокари. Делаем сенсоры. Проекты сенсорных устройств на базе Arduino и Raspberry Pi, Вильямс, 2015.
7. Международный Интернет портал [Электронный ресурс] –URL: http://azbsec.ru/ (дата обращения: 13.06.2020)
8. Международный Интернет портал [Электронный ресурс] –URL: http://studbooks.net/1030843/pravo/klassifikatsiya\_narushiteley\_osnove\_modeley\_deystviy (дата обращения: 13.05.2020).
9. Belarus focus[Электронный ресурс] рост преступности –URL:https://belarusinfocus.info/by/byaspeka/v-belarusi-vpervye-za-mnogo-let-otmechaetsya-sereznyy-rost-prestupnosti
10. Занимательная робототехника [Электронный ресурс] NodeMCU –URL: http://edurobots.ru/2017/04/nodemcu-esp8266/(дата обращения: 05.05.2020).
11. ArduinoMaster [Электронный ресурс] Arduino Nano–URL: https://arduinomaster.ru/platy-arduino/plata-arduino-nano// (дата обращения: 05.05.2020).
12. Электронные компоненты [Электронный ресурс] W5500–URL: http://mt-system.ru/catalog/interfejsnye-moduli-i-mikroshemy/kompanija-wiznet/mikroshemy-dlja-postoroenija-setej-ether-1/ (дата обращения: 03.05.2020).
13. GSM-модуль SIM800L [Электронный ресурс] часть 1. Введение –URL: http://codius.ru/articles/GSM\_\_SIM800L (дата обращения: 03.06.2020).
14. Национальная библиотека им. Н. Э. Баумана [Электронный ресурс] RS-485 –URL: https://ru.bmstu.wiki/index.php?title=RS485&mobileaction (дата обращения: 02.05.2020).
15. Adafruit\_IoT [Электронный ресурс] –URL:https://io.adafruit.com/ (дата обращения: 02.05.2020).
16. Википедия [Электронный ресурс] Serial Peripheral Interface –URL:https://ru.wikipedia.org/wiki/Serial\_Peripheral\_Interface
17. GitHub [Электронный ресурс] SmartHomeSecuritySystem –URL:https://github.com/amd12/SmartHomeSecuritySystem (дата обращения 13.04.2020).

# Приложение А Список компонентов

# Приложение B Исходный код ESP-12

# Приложение С Исходный код AVR