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

Институт машиностроения
(наименования института полностью)

Кафедра «Промышленная электроника»
(наименование)

11.03.04 Электроника и наноэлектроника
(код и наименование направления подготовки, специальности)

Электроника и робототехника
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
(БАКАЛАВРСКАЯ РАБОТА)

на тему “Электронный замок с Android – управлением”

Обучающийся А.М. Конюхов
(И.О. Фамилия) (личная подпись)

Руководитель к.т.н., доцент, М.В. Позднов
(ученая степень, звание, И.О. Фамилия)

Консультант к.п.н., доцент, О.В. Лебединская
(ученая степень, звание, И.О. Фамилия)

Тольятти 2023

Аннотация

Объем 82 страницы, 29 рисунков, 8 таблиц, 24 источника, 3 приложения и 7 графических представлений на листах формата А1.

Объектом исследования являются схемотехнические решения, направленные на защиту физической собственности и методы взаимодействия с ними:

Цель работы — разработка автономного комплекса в сфере защиты физической собственности с применением знаний в сферах схемотехники и программирования микроконтроллеров, включающего в себя ударопрочный замок, схемотехническое решение для автономной работы от батареи и программные модули для устройства управления и микроконтроллера.

В ходе выполнения работы основные задачи разделены на два модуля – аппаратный и программный.

Программный модуль включает в себя исследование передачи данных с помощью беспроводной технологии Bluetooth и реализация программного решения для безопасной передачи приватных ключей (паролей) с применением языка программирования Java и внешних библиотек для работы с Bluetooth.

Аппаратный модуль включает в себя поиск и составление элементной базы аппаратного комплекса и проектирование схемотехнического решения для автономной работы составных элементов замка.

Областью применения данной разработки являются безопасность физической и цифровой собственности, однако установка замка возможна на любом другом механизме, работающем по принципу распахивания, однако требует твердой и крепкой площадки крепления замка.

В заключение, применение комплекса позволило ввести дополнительный метод доступа к аудитории 410 в учебном корпусе “Э” посредством беспроводной технологии Bluetooth с применением мобильного приложения.

Abstract

The title of the graduation work is: «Electric lock with Android-control».

The graduation work consists 82 pages, including 28 figures, 8 tables, 24 references, 3 appendices, and 7 graphical representations on A1-sized sheets.

The subject of research encompasses schematic solutions aimed at protecting physical property and the methods of interaction with them.

The aim is to develop an autonomous system in the field of physical property protection, utilizing knowledge in the areas of circuit design and microcontroller programming. The system includes a shock-resistant lock, a circuitry solution for autonomous battery operation, and software modules for control device and microcontroller.

The graduation work may be divided into two modules - hardware and software. The software module includes the study of native data sending using Bluetooth wireless technology and the implementation of a program solution to enable the sending of private keys (passwords) using the Java programming language and an external libraries for working with Bluetooth. The hardware module includes the search and assembly of elements of the hardware complex and the development of circuit solutions for the autonomous logic operation of the components of the lock.

The field of application of this development includes the security of physical and digital property. The installation of the developed system is possible on any other mechanism that operates on the principle of opening, but it requires a solid and sturdy surface for attaching the lock.

Содержание

Введение

В настоящее время развитие технологий приводит к улучшению качества жизни и удобства ее предоставления. Одним из направлений является разработка устройств, обеспечивающих безопасность жилья. Электронные замки являются высокотехнологичным инструментом для защиты дома, квартиры или офиса. Умные замки являются одним из наиболее популярных устройств для умного дома и систем безопасности. Они обеспечивают безопасность, контроль доступа и удобство, используя технологии беспроводной связи, мобильные приложения и интернет вещей (IoT). Умные замки также обычно предоставляют более высокий уровень безопасности, чем традиционные механические замки. Многие модели имеют функцию автоматического блокирования, которая гарантирует, что дверь будет закрыта, даже если вы забыли сделать это вручную. Некоторые умные замки также предоставляют функцию виртуальных ключей, которые можно отправлять через интернет другим людям, например, гостям, которым нужно получить доступ к вашему дому на короткое время.

Кроме того, некоторые умные замки могут работать в связке с другими устройствами умного дома, например, с камерами видеонаблюдения или сигнализацией. Это позволяет создать более интегрированную систему безопасности и контроля доступа. С помощью интеграции с камерами видеонаблюдения, умный замок может распознавать лица, что позволяет предоставить доступ только авторизованным лицам и записывать видеофрагменты, когда кто-то пытается получить несанкционированный доступ. Также замки могут интегрироваться с сигнализацией, что позволяет синхронизировать их работу.

Рост заинтересованности в умных замках может быть связан и с их дизайном. Некоторые модели имеют современный и эстетически приятный внешний вид, что может привлечь внимание покупателей, которые ценят дизайн и стиль.

1 Постановка задачи

Основной задачей выпускной квалификационной работы является разработка программного и аппаратного интерфейсов управления и поддержания стабильной работы устройства, используя полученные знания принципов объектно-ориентированного программирования микроконтроллеров и мобильных устройств, а также знания в области микроэлектроники и схемотехники.

В результате работы должны быть разработан взаимосвязанный программно-аппаратный комплекс, который позволит пользователю взаимодействовать с замком посредством беспроводной технологии Bluetooth и обеспечит автономную работу замка.

1.1 Поиск и анализ технических параметров аналогичных устройств

Рассмотрим популярные аналоги электронных замков с Bluetooth управлением. Современные электронные замки становятся все более популярными, так как они обеспечивают высокий уровень безопасности и удобство в использовании. Среди них замки с Bluetooth управлением, которые позволяют управлять замком с помощью смартфона или планшета, являются особенно популярными. Информация о самых востребованных электронных замках взята из электронного ресурса [24].

Одним из популярных аналогов электронного замка с Bluetooth управлением является August Smart Lock Pro. Он имеет элегантный дизайн, прост в установке и обладает большими функциональными возможностями. Он совместим с Amazon Alexa, Apple HomeKit и Google Assistant, что позволяет управлять им голосом. Кроме того, он поддерживает функцию автоматического открытия двери, когда вы подходите к ней. Замок изображен на рисунке 1.

Рисунок 1 – August Smart Lock Pro

Другим популярным аналогом является Schlage Sense Smart Lock. Он имеет прочный металлический корпус, который обеспечивает надежность и долговечность. Он также совместим с Amazon Alexa, Apple HomeKit и Google Assistant, что позволяет управлять им голосом. Кроме того, он имеет функцию управления доступом для нескольких пользователей, что делает его идеальным для семейного использования. Замок показан на рисунке 2.

Рисунок 2 - Schlage Sense Smart Lock

Еще одним популярным аналогом является Yale Assure Lock SL с модулем Bluetooth. Он имеет минималистичный дизайн и прост в использовании. Он также совместим с Amazon Alexa, Apple HomeKit и Google Assistant, что позволяет управлять им голосом. Он обеспечивает высокий уровень безопасности, так как имеет функцию управления доступом для нескольких пользователей и оповещение о попытках несанкционированного доступа. Замок показан на рисунке 3.

Рисунок 3 - Yale Assure Lock SL

Популярные аналоги электронного замка с Bluetooth управлением предоставляют множество функциональных возможностей управления замком и гарантируют высокий уровень безопасности. Ниша умных замков достаточно обширна, но все еще является молодой отраслью из-за постоянно обновляющихся технологических трендов.

1.2 Критерии для разработки устройства

При разработке умного замка с модулем Bluetooth необходимо учитывать ряд критериев, чтобы обеспечить его надежность, безопасность и удобство использования. Рассмотрим некоторые из них:

1. Надежность и безопасность - умный замок должен обеспечивать высокий уровень защиты от несанкционированного доступа. Это может быть достигнуто с помощью шифрования данных, которое защитит замок от взлома. Также необходимо убедиться, что механизм замка изготовлен из прочных материалов, чтобы обеспечить надежность и долговечность работы замка.

2. Удобство использования - умный замок должен быть прост в установке и использовании. Он должен иметь интуитивно понятный интерфейс и функции управления, которые легко понимаются пользователем. Например, замок может быть управляем с помощью мобильного приложения, что делает его удобным в использовании.

3. Совместимость с другими устройствами - умный замок с модулем Bluetooth должен быть совместим с другими устройствами, такими как смартфоны, планшеты и другие устройства с поддержкой Bluetooth. Это позволит пользователям управлять замком с помощью устройств, которые они уже используют, без необходимости покупать дополнительное оборудование.

4. Функциональность - умный замок должен обладать необходимыми функциональными возможностями, такими как возможность управления доступом для нескольких пользователей, история открытий/закрытий двери, возможность настройки времени доступа и т.д. Это сделает его более удобным в использовании и позволит пользователю настроить замок на свой вкус.

5. Дизайн - дизайн умного замка должен соответствовать дизайну двери и интерьера помещения. Он должен быть эстетичным и не выделяться слишком ярко среди других элементов интерьера.

При разработке умного замка с модулем Bluetooth необходимо учитывать как технические, так и эргономические критерии, чтобы обеспечить его надежность, безопасность и удобство использования.

1.3 Варианты разработки устройства и его структура

Существует несколько вариантов разработки умного замка с модулем Bluetooth. Рассмотрим некоторые из них:

1. Разработка замка с Bluetooth-модулем - первый вариант разработки заключается в создании замка, в который будет встроен Bluetooth-модуль. В этом случае, пользователь сможет управлять замком с помощью приложения на своем смартфоне, не используя дополнительные устройства. Этот подход позволит сделать устройство компактным и простым в использовании.

2. Использование Bluetooth-адаптера - второй вариант состоит в использовании отдельного Bluetooth-адаптера, который можно подключить к существующему замку. Пользователь может использовать приложение на своем смартфоне, чтобы управлять замком через Bluetooth-адаптер. Этот подход более гибкий, так как позволяет использовать Bluetooth-адаптеры разных производителей, но требует наличия дополнительного оборудования.

3. Совмещение с другими технологиями - третий вариант заключается в совмещении модуля Bluetooth с другими технологиями, такими как NFC или Wi-Fi. Это позволит расширить функциональность устройства и сделать его еще более удобным в использовании.

4. Создание умного замка с Bluetooth и датчиками - четвертый вариант состоит в создании умного замка, который будет оснащен дополнительными датчиками, такими как датчики движения или датчики окружающей среды.

Это позволит создать более интеллектуальное устройство, которое будет реагировать на окружающую среду и действия пользователя.

Каждый из этих подходов имеет свои преимущества и недостатки, и выбор зависит от конкретных потребностей и задач проекта. Независимо от выбранного подхода, необходимо обеспечить надежность, безопасность и удобство использования устройства, поэтому исходя из базовых навыков работы с микроконтроллерами и схемотехническими решениями, принято решение о разработке внешнего логического головного устройства, состоящего из микроконтроллера и Bluetooth модуля, передающего информацию через последовательный порт микроконтроллера.

В качестве основных структурных и функциональных элементов выступают микроконтроллер Arduino Nano, Bluetooth модуль HC-04 и ударопрочный замок Falcon Eye FE-2369, каждый из которых будет рассмотрен ниже.

Arduino Nano - это отладочная плата с расположенным микроконтроллером Atmel AVR на ней. Устройство предназначено для разработки интерактивных проектов и простых устройств, использующих цифровые и аналоговые выходы, расположенные на отладочной плате. Arduino Nano использует специальную среду разработки Arduino IDE, которая позволяет легко программировать и интегрировать датчики и другие компоненты в проекты с помощью внешних подключаемых библиотек. Платформа поддерживает множество языков программирования, включая C++ и предоставляет широкий набор библиотек и инструментов для упрощения разработки программного обеспечения. Arduino Nano показана на рисунке 4.

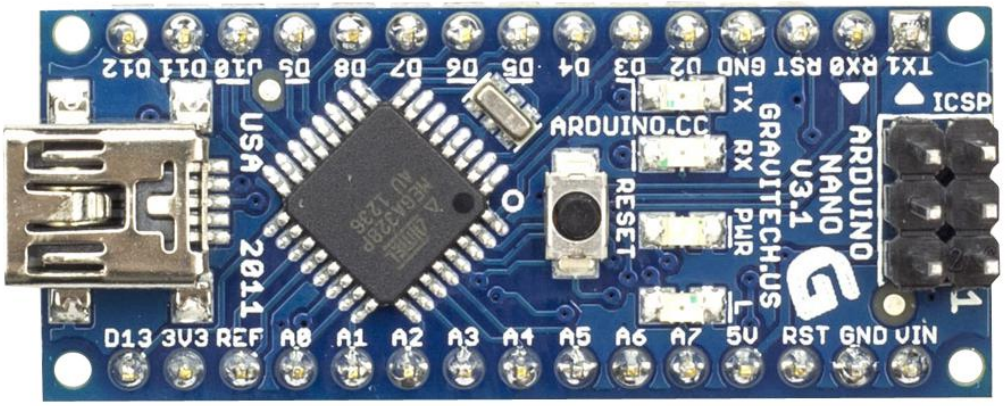

Рисунок 4 – Отладочная плата Arduino Nano

ATmega328P является 8-битным микроконтроллером компании Atmel, используемый в платформах Arduino, включая Arduino Nano. Он основан на архитектуре RISC и имеет встроенный оперативный усилитель, шину данных и аналоговый-цифровые преобразователи. ATmega328P является популярным выбором для простых проектов, так как он имеет достаточно ресурсов и возможностей для реализации многих идей, а также поддерживается большим сообществом разработчиков и активно обновляющейся документацией. Технические характеристики микроконтроллера приведены в таблице 1.

Таблица 1 – Характеристики микроконтроллера atmega328

Микроконтроллер	atmega328
FLASH память, КБ	32
EEPROM память, КБ	1
SRAM память, КБ	2
Цифровые входы/ выходы	14(6 с шим)
Аналоговые входы	8
Вес, г	15.82

Bluetooth адаптер HC-04 является популярным модулем для взаимодействия устройств Arduino с другими устройствами посредством Bluetooth. Bluetooth адаптер показан на рисунке 5.

Рисунок 5 – Bluetooth модуль HC-04

HC-04 работает в режиме Master/Slave, поддерживает стандарты Bluetooth 2.0 или выше и имеет радиус действия до 30 метров в открытом пространстве. Адаптер может быть подключен к Arduino посредством шины UART (RX, TX) и питания (Vcc, GND). Перед использованием необходимо настроить адаптер в режиме работы ведомым или ведущим адаптером и задать ему имя и пароль. Это может быть сделано посредством подключения к адаптеру через компьютер и использования специальной программы для конфигурации. После настройки адаптера может быть использован для передачи данных между Arduino и другими устройствами через беспроводную технологию Bluetooth.

Замок Falcon Eye FE-2369 является электромеханическим замком для входных дверей в квартирах, офисах и других помещениях. Он обеспечивает высокий уровень безопасности и удобство в использовании, так как может быть открыт не только с помощью физического ключа, но и посредством подачи кратковременного электрического импульса. Замок показан на рисунке 6.

Рисунок 6 – Замок Falcon Eye FE-2369i

Характеристики электромеханического замка приведены в таблице 2.

Таблица 2 – Характеристики замка Falcon Eye FE2369i

Сила удержания	500кг.
Тип	Электромеханический,нормально-закрытый,накладной
Материал	Сталь (крашенная)
Исполнение	Внутренние/уличное
Питание	9-12В (DC) 3-4А
Размеры	127x105x40 мм
Блокировка кнопки	Есть
Внутренний цилиндр	Есть
Рабочая температура	-40+50°С
Рабочая влажность	10%-90%
Область применения	Легкие/средние/тяжелые двери

Структурные элементы, приведенные выше, относятся к функциональному модулю разрабатываемого аппаратного решения, что подразумевает работу замка согласно заданному алгоритму работы, исходя из команд, переданных пользователем.

2 Конструкторское проектирование устройства

Конструкторское проектирование устройства – это процесс систематического создания и разработки технического устройства или системы, основанный на применении научных и инженерных принципов. В ходе конструкторского проектирования учитываются требования и спецификации, определяющие функциональность, производительность, надежность и другие характеристики устройства.

Основной целью конструкторского проектирования является создание оптимального технического решения, удовлетворяющего поставленным требованиям. Для этого процесс включает в себя анализ исходных данных, постановку задачи, исследование возможных вариантов решений, выбор наиболее подходящего варианта, разработку деталей и сборочных единиц, а также проверку и испытание созданного устройства.

2.1 Анализ и выбор элементной базы для аппаратной части замка

Анализ и выбор элементной базы для аппаратной части замка для управления должны основываться на требованиях к функциональности и производительности замка, а также на бюджетных ограничениях и доступности компонентов на рынке.

Ключевые структурные элементы, которые используются в аппаратной части замка, включают в себя:

- микроконтроллер, который является главным компонентом управления замком. Он отвечает за обработку входных данных и управление выходными сигналами. При выборе микроконтроллера важно учитывать такие факторы, как производительность, энергопотребление, стоимость и эргономика. Данные о актуальных микроконтроллерах взяты из электронного источника [19];

- внешние функциональные модули, способные обмениваться информацией с микроконтроллером, предоставляя дополнительный функциональный потенциал. Данные о актуальных модулях взяты из источника [5] и [10];

- замок, обладающий высокой степенью защиты и располагающий необходимым функционалом исходя из поставленных функциональных задач. Список доступных электромеханических замков и их характеристики приведены в источнике [3];

Выбор элементной базы должен основываться на совместимости компонентов между собой и на их соответствии требованиям к необходимой функциональности и эргономики замка.

Существует несколько микроконтроллерных платформ, которые могут использоваться в качестве альтернативы Arduino Nano, в зависимости от конкретных требований проекта:

- raspberry Pi Pico - это микроконтроллерная плата, основанная на процессоре ARM Cortex-M0+. Она обладает низким энергопотреблением и высокой производительностью, и может использоваться для множества различных проектов. ;

- esp32 - это микроконтроллерная плата, которая поддерживает беспроводную связь по Wi-Fi и Bluetooth. Она обладает большим количеством входов/выходов и может использоваться для создания различных проектов, таких как интернет вещей и автоматизации дома;

- stm32 - это микроконтроллерная плата, основанная на процессоре ARM Cortex-M. Она обладает высокой производительностью и множеством возможностей расширения, и может использоваться для множества различных проектов;

- teensy - это микроконтроллерная плата, которая имеет меньший размер, чем Arduino Nano, но обладает высокой производительностью и множеством входов/выходов. Она может использоваться для создания проектов, которым требуются высокие скорости обработки данных;

- adafruit Feather - это серия микроконтроллерных плат, которые обладают различными возможностями, включая беспроводную связь, GPS, датчики и т.д. Они могут использоваться для создания различных проектов, таких как интернет вещей и автоматизация дома;

Каждая плата имеет свои преимущества и недостатки, поэтому важно провести исследование и выбрать наиболее подходящую платформу для конкретного проекта. С особенностями каждой платформы можно ознакомиться в источнике [1] и [6]. Существуют некоторые преимущества, которые Arduino Nano имеет на фоне других микроконтроллеров:

- простота использования - Arduino Nano является одной из наиболее простых микроконтроллерных плат для начинающих. Она имеет простую структуру, легко программируется и не требует большого количества дополнительных компонентов;

- низкая стоимость - Arduino Nano имеет низкую стоимость по сравнению с многими другими микроконтроллерными платами, что делает его доступным для большинства людей, желающих начать работу с микроконтроллерами;

- широкая поддержка - Arduino Nano имеет огромное сообщество пользователей, которые делятся опытом и знаниями, что позволяет новичкам быстро получить помощь и поддержку;

- многофункциональность - Arduino Nano может использоваться для широкого спектра проектов, позволяя работать с цифровым и аналоговым сигналами;

- низкое энергопотребление - Arduino Nano имеет низкое энергопотребление, что делает его идеальным для проектов, которые требуют длительной автономной работы на батарейках или других источниках питания;

Arduino Nano является простой в использовании, доступной и многофункциональной платформой, которая может быть использована в широком спектре проектов. Она обладает множеством преимуществ, которые

делают ее хорошим выбором для платформы разработки программно-аппаратного комплекса электронного замка. С особенностями данной платформой можно ознакомиться в источнике [4].

Arduino Nano поддерживает несколько Bluetooth модулей, которые можно использовать для обмена информацией посредством последовательного порта. Некоторые из популярных модулей Bluetooth, которые можно использовать с Arduino Nano, приведены далее:

- bluetooth-модуль HC-05 – это один из самых популярных Bluetooth-модулей для Arduino Nano. Он поддерживает протоколы Bluetooth 2.0 и Bluetooth 4.0 и позволяет обмениваться данными между устройствами на расстоянии до 10 метров;

- bluetooth-модуль HC-06 – это более простой Bluetooth-модуль, который также поддерживает протоколы Bluetooth 2.0 и Bluetooth 4.0. Он обладает меньшими возможностями, чем HC-05, но также может использоваться для создания беспроводных проектов с Arduino Nano;

- bluetooth-модуль HM-10 – этот Bluetooth-модуль поддерживает Bluetooth 4.0 и имеет низкое энергопотребление. Он может использоваться для создания беспроводных проектов с Arduino Nano, которые работают на батарейках или других источниках питания;

- bluetooth-модуль BLE Nano – этот Bluetooth-модуль поддерживает Bluetooth 4.0 и обладает малыми размерами. Он может использоваться для создания небольших, беспроводных проектов с Arduino Nano;

- bluetooth-модуль HC-08 – это Bluetooth-модуль с низким энергопотреблением, который поддерживает Bluetooth 4.0. Он может использоваться для создания беспроводных проектов с Arduino Nano, которые работают на батарейках или других источниках питания;

Выбор конкретного модуля Bluetooth для Arduino Nano зависит от требований проекта, доступности и надежности датчика. Bluetooth-модуль HC-04 является простым и дешевым модулем, что выделяет его на фоне аналогов. Он использует протокол Bluetooth 2.0 и может обмениваться

данными на расстоянии до 10 метров. Одним из преимуществ HC-04 является его простота использования. Он не имеет первоначальной настройки параметров перед началом работы и не требует дополнительных компонентов для работы с Arduino Nano. HC-04 также может быть хорошим выбором для проектов, которые не требуют передачи большого объема данных или более высокой скорости передачи данных, которую могут обеспечить более современные модули Bluetooth. Однако, если проект требует более быстрой передачи данных, большей дальности обмена или большего объема передаваемых данных, то более продвинутые модули Bluetooth, такие как HC-05, HC-06, HM-10 и BLE Nano, могут быть более подходящими выборами, но исходя их эргономики, доступности и необходимого функционала был выбран модуль HC-04. С особенностями внешних модулей можно ознакомиться в источниках [10] и [12].

После выбора логических компонентов для передачи данных и обработки команд, приходящих от пользователя, следует выбрать замок, обладающий возможностью отпирающего открытия с помощью подачи на него кратковременного электрического сигнала, так как в логической работе программы предусмотрена подача краткосрочного напряжения с порта отладочной платы на реле, открывающее подачу импульса на замок. Рассмотрим возможные альтернативы:

- lbp85.2 – врезной электромеханический замок, обладающий стальной конструкцией и штифтовым цилиндровым механизмом секретности. Имеет 2 метода доступа к помещению: физический ключ и электрический импульс. Номинальное напряжение 12 В;

- st-el-01 – электромеханический замок, который можно открывать с физического ключа и кратковременного электрического импульса. Имеет аналогичную с Falcon Eye FE-1269i стальную конструкцию с цилиндровым механизмом. Номинальное напряжение 12 В;

- cisa 11610-60-3-0000-C5 – врезной электромеханический замок, обладающий стальным корпусом с цилиндровым механизмом, имеющий

также два режима доступа с помощью физического ключа и подачей отпирающего сигнала. Номинальное напряжение 12 В;

Замки в данном классе обладают схожими характеристиками, позволяя выбрать более доступное решение, не пренебрегая уровнем защиты комплекса. Замок Falcon Eye-FE2369 имеет несколько преимуществ, которые могут быть важны сравнении с альтернативными замками:

- высокий уровень безопасности - замок обладает стальным корпусом и механизмом, что является высокой степенью защиты от взлома, в том числе от механических и электронных методов взлома;

- простота использования – замок легко устанавливается и подключается посредством двух электромагнитных контактов;

- долговечность – замок выполнен из качественных материалов и имеет высокую степень износостойкости, что гарантирует его долговечность и надежность в эксплуатации;

- доступность – замок имеет официальную сертификацию в Российской Федерации и распространяется по рекомендованной розничной цене, составляющий минимальную финансовую планку входа в аналогичные защитные конструкции;

Замок Falcon Eye-FE2369 имеет дополнительный метод авторизации в виде подачи кратковременного импульса, что делает его привлекательным выбором для взаимодействия с ним через краткосрочную отправку отпирающего сигнала с отладочной платы Arduino Nano. Технические характеристики замка приведены в таблице 3.

Таблица 3 – Технические характеристики Falcon Eye FE-1269i

Рабочее напряжение	12 В
Потребляемый ток	3-4 А
Время подачи отпирающего сигнала	Не более 3 сек.
Рабочая температура	- 40 … 50 град. С
Вес	1,5 кг

2.2 Анализ и выбор АКБ

Для питания аппаратного комплекса, а именно микроконтроллера с замком, необходим питающий элемент – АКБ. Основные разновидности АКБ, используемые в промышленности: литий-ионный и свинцовый аккумуляторы. С принципом работы АКБ можно ознакомиться в источнике [14]. Литий-ионный аккумулятор (Li-Ion) – это тип электрохимической батареи, который использует литий в качестве одного из элементов, взаимодействующих в процессе зарядки и разрядки. Принцип работы литий-ионного аккумулятора основан на переносе литиевых ионов между электродами, которые находятся в электролите. В аккумуляторе есть два электрода: катод и анод. Катод, обычно, изготавливается из окисленного лития (LiCoO2), а анод - из углерода (графита). При зарядке аккумулятора, литий-ионы движутся из катода в анод, где они встраиваются в структуру графитового электрода. В этот момент, происходит обратная реакция на катоде, где ионы лития соединяются с катионами металлического кобальта. При разрядке аккумулятора, происходит обратный процесс - литий-ионы переходят из анода в катод, выделяя электрический ток, который можно использовать для питания устройства. Ключевой особенностью литий-ионных аккумуляторов является высокая энергетическая плотность, то есть они могут хранить больше энергии на единицу массы, чем другие типы аккумуляторов. Кроме того, литий-ионные аккумуляторы имеют высокий КПД и могут быть перезаряжены множество раз, что делает их идеальным выбором для большинства мобильных устройств. Литий-ионный аккумулятор показан на рисунке 7.

Рисунок 7 – Литий-ионный аккумулятор

Следует рассмотреть преимущества и недостатки каждого типа батареи, основываясь на их физико-химических свойствах.

Литий-ионные и свинцовые батареи - это два разных типа аккумуляторов, которые отличаются своими химическими свойствами и конструкцией. Перечислим различия между ними:

- химический состав - свинцовые батареи используют свинец и кислоту для создания электрической энергии, а литий-ионные батареи содержат литий и другие химические соединения для создания энергии;

- емкость - литий-ионные батареи обычно имеют более высокую энергетическую плотность (количество энергии, хранимой в единице объема или массы), что позволяет им длительнее работать на одном заряде. Свинцовые батареи имеют более низкую энергетическую плотность, что означает, что они не могут работать так долго без подзарядки;

- вес - литий-ионные батареи в среднем легче, чем свинцовые батареи, при том же уровне емкости;

- надежность - литий-ионные батареи имеют более высокий уровень надежности и меньше подвержены эффекту "памяти" (т.е. потере емкости при многократном перезаряде), чем свинцовые батареи;

- стоимость - литий-ионные батареи могут быть более дорогими, чем свинцовые батареи, но за счет своей долговечности и более высокой энергетической плотности, они могут быть более экономичны в долгосрочной перспективе;

- экологические аспекты - литий-ионные батареи обычно считаются более экологичными, чем свинцовые батареи, потому что они содержат меньше токсичных материалов;

Свинцовый аккумулятор является типом электрохимической батареи, которая хранит электрическую энергию в форме химической энергии. Он состоит из нескольких ячеек, каждая из которых содержит пластины из свинцового сплава (свинец с добавками антимония и кальция), погруженные в электролит. Принцип работы свинцового аккумулятора основан на электролизе воды. Когда аккумулятор заряжен, электрический ток поступает через аккумулятор, приводя к электролизу воды в электролите. Это происходит благодаря наличию кислоты в электролите, которая разлагается на ионы водорода и отрицательные ионы антимония. Положительные пластины в ячейке становятся сульфатом свинца, PbSO4, в то время как отрицательные пластины становятся сульфатом свинца, PbO2. В процессе разряда аккумулятора, обратная реакция происходит, и химическая энергия хранится в виде электрической энергии. Таким образом, когда аккумулятор используется для питания устройства, он выдает электрический ток, питающий устройство. Когда заряд в аккумуляторе исчерпывается, его можно перезарядить путем подключения его к источнику электрической энергии, что позволяет провести обратную реакцию и восстановить свинцовые пластины в их исходное состояние. Свинцовый аккумулятор показан на рисунке 8.

Рисунок 8 - Свинцовый аккумулятор

Свинцовые батареи имеют несколько преимуществ для применения в качестве источника питания для устройства:

- низкая стоимость - свинцовые батареи являются одними из самых дешевых типов аккумуляторов, что делает их доступными для широкого круга потребителей;

- хорошая работа при низкой температуре - свинцовые батареи могут сохранять свою емкость и работать даже при низких температурах, что делает их идеальными для использования в холодных условиях;

- высокая надежность - свинцовые батареи имеют высокую надежность и долговечность, и могут работать в течение длительного времени без необходимости замены;

- необходимость в специальной зарядке - свинцовые батареи не требуют специальных зарядных устройств и могут быть заряжены стандартным зарядным устройством;

- габариты - свинцовые батареи доступны в различных размерах и формах, что позволяет им использоваться в широком диапазоне приложений;

- экологическая безопасность - свинцовые батареи могут быть утилизированы безопасно и переработаны, что делает их более экологически безопасными, чем некоторые другие типы аккумуляторов;

Свинцовые батареи могут быть хорошим выбором для использования в схемах, где требуется высокая надежность и долговечность при низкой стоимости, при этом разрабатываемое устройство имеет статическое расположение, что позволяет использовать свинцовую батарею в вертикальном положении без нужды крепить элемент из-за возможной утечки электролита в случае расположения батареи в состоянии отличном от вертикального, однако современные свинцовые батареи предусматривают различное расположение.

Свойства разряда свинцовой батареи зависят от ее конструкции, состава электродов, температуры и других факторов. Рассмотрим наглядно процесс разряда свинцовой батареи при разных значениях тока на графике зависимости напряжения к времени. График зависимости напряжения от времени при разных температурных условиях показан на рисунке 9.

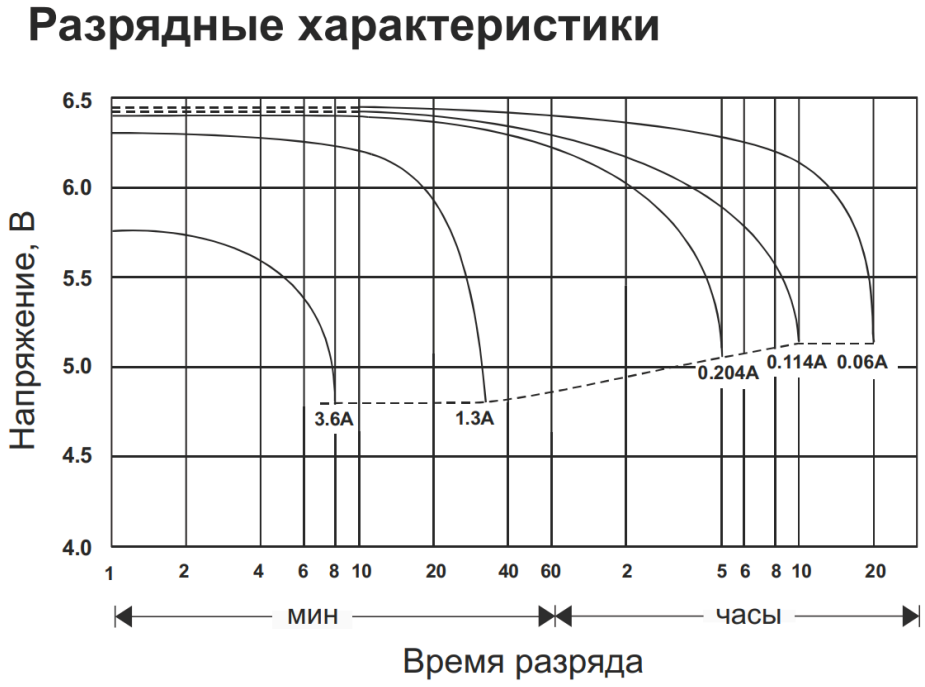

Рисунок 9 – График разряда свинцовой батареи

В качестве аккумуляторной батареи выбран аккумулятор свинцовый GS1.3-6, технические характеристики которого включают напряжение 6 В и емкость 1,3 А*ч. Аккумуляторная батарея рассчитана на использование в охранно-пожарных системах, так как уникальная конструктивная разработка предусматривает отсутствие протечек электролита, за счет чего гарантирована стабильная работа батареи в любом положении. Высокая мощность выходного тока обеспечивает стабильную работу устройства. Изображение аккумулятора показано на рисунке 10.

Рисунок 10 - Аккумулятор свинцовый GS1.3-6

Для электрического питания отладочной платы Arduino Nano необходима подача напряжения 5 В, в случае подачи стабилизируемого тока, однако также принято рассматривать диапазон значений от 6 В до 20 В, в случае нестабилизируемого электрического питания от внешнего источника. Следовательно, данный питающий элемент подходит для обеспечения качественного электрического питания, в случае стабилизированного заряда, подводимого к отладочной плате Arduino Nano.

2.3 Схемы обслуживания АКБ

Значением напряжения подзаряда батареи является 6,4 В при средней температуре 25 град. С, следовательно необходимо понизить подаваемое от

блока питания напряжение 12 В, подключенного к сетевому напряжению 220 В и питающего разрабатываемый комплекс. Для достижения автономной работы замка следует разработать схему обслуживания для свинцового АКБ, питающего готовое схемотехническое решение. Плата обслуживания АКБ включает два каскада, выполняющие следующие задачи:

- контроль тока, подаваемого от нагрузки;

- контроль и понижение напряжения, подаваемого от нагрузки до значения 6,4В;

- контроль разряда свинцового АКБ;

Рассмотрим первый каскад схемы обслуживания АКБ. Разрабатываемый каскад приведен на рисунке 11.

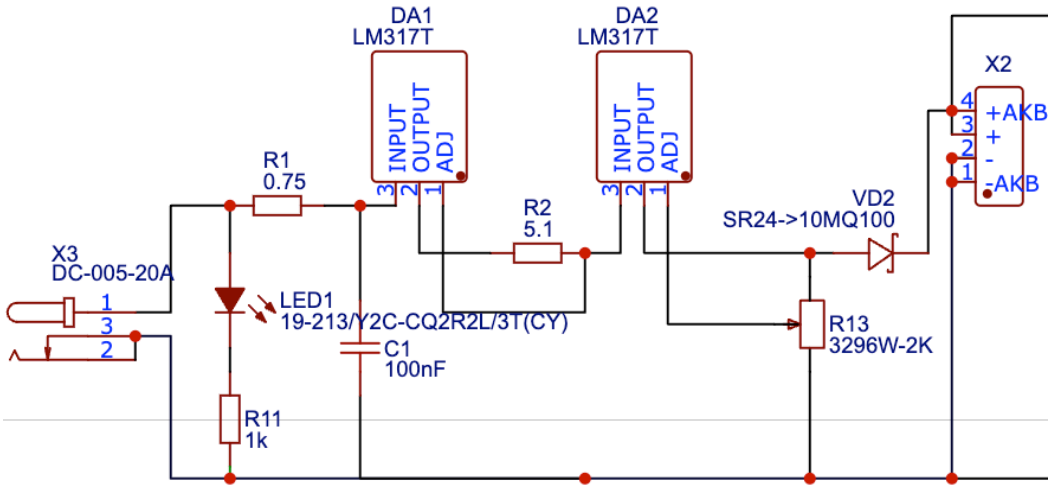

Рисунок 11 – Каскад ограничения тока и напряжения на нагрузке

Каскад включает в себя два регулируемых стабилизатора напряжения LM317T. Данный элемент микроэлектроники является регулируемым стабилизатором напряжения с несколькими преимуществами, он позволяет легко регулировать выходное напряжение, имеет низкий уровень шума и защиту от перегрева, высокий коэффициент подавления помех на входе, а также прост в использовании и надежен. Это делает LM317T очень гибким и удобным решением для использования в различных проектах. Для более подробного ознакомления с данным элементом микроэлеткроники можно

ознакомиться с источником [23]. Характеристики LM317T приведены в таблице 4.

Таблица 4 – Характеристики стабилизатора напряжения LM317T.

Полярность	Положительная
Тип выхода	регулируемый
Количество выходов	1
Выходное напряжение, В	1.2…37
Максимальный ток нагрузки, А	1.5
Максимальное входное напряжение, В	40
Рабочая температура,°С	0…+125
Корпус	to-220sg

В первом каскаде разрабатываемой схемы обслуживания АКБ на выходе первого стабилизатора напряжения присутствует резистор ограничения тока короткого замыкания, что реализует еще одну базовую конструкцию, включающую один резистор на выходе стабилизатора, где выходной ток Iвых рассчитывается по формуле

$$I\text{вых} = \frac{1{,}2}{R0}, \quad (1)$$

где R0 – сопротивление ограничивающего резистора, Ом.

Взято значение тока 0,2 А, так как величина тока не должна превышать величину максимального тока питания аккумулятора 0,36 В. Исходя из формулы (1) можно преобразовать формулу для нахождения сопротивление резистора R0

$$R0 = \frac{1{,}2}{I\text{вых}}, \quad (2)$$

где Iвых – выходной ток, А.

Возможно произвести расчет при известном ожидаемом выходном токе, применяя формулу (2)

$$R0 = \frac{1{,}2}{0{,}2} = 6 \text{ Ом}. \quad (3)$$

Максимально близким подходящим значением сопротивления реального резистора является 5,1 Ом, следует рассчитать значение тока при таком сопротивление токоограничивающего резистора

$$I\text{вых} = \frac{1{,}2}{5{,}1 \text{ Ом}} = 0{,}23 \text{ А}. \quad (4)$$

Ток удовлетворяет заданным условиям при меньшем токе максимального тока питания аккумулятора 0,36 А.

Второй стабилизатор напряжения включен в схему в режиме понижения и стабилизации напряжения. Схема понижения напряжения со стабилизатором напряжения LM317T предусматривает использование двух резисторов, значения сопротивления которых определяет выходное напряжение, и два конденсатора на входе и выходе схемы. Реализация схемы ограничения напряжения с LM317T показана на рисунке 12.

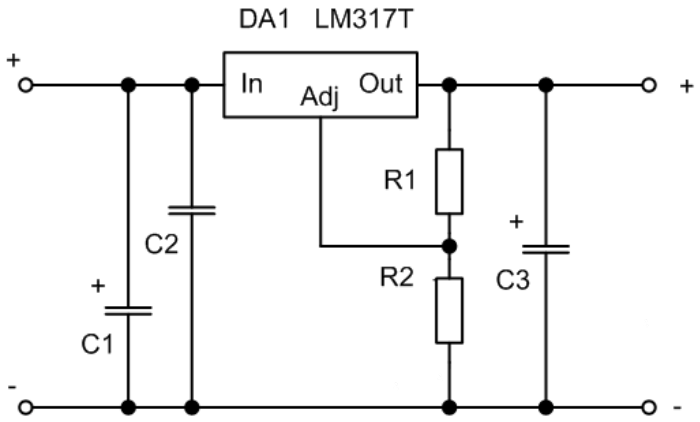

Рисунок 12 – Реализация каскада с стабилизатором напряжения LM317T

Следует обратить внимание на опорное напряжение и ток, вытекающий из вывода подстройки. Каждый экземпляр микросхемы стабилизатора имеет свою уникальную величину опорного напряжения, которая варьируется в диапазоне от 1,2 до 1,3 В. В среднем значение этого напряжения составляет 1,25 В. Опорное напряжение представляет собой целевое значение, которое микросхема стабилизатора стремится поддерживать на резисторе R1. Если резистор R2 замкнуть, то на выходе схемы будет поддерживаться напряжение величиной 1,25 В. Однако, если на R2 будет возникать падение напряжения, то это приведет к увеличению выходного напряжения. Таким образом, значение 1,25 В на R1 складывается с падением напряжения на R2, образуя выходное напряжение. Падение напряжения на R2 играет роль дополнительного вклада в формирование выходного напряжения

микросхемы стабилизатора. Чем больше будет это падение напряжения, тем выше будет выходное напряжение.

Ток, вытекающий из вывода подстройки, является неизбежным паразитным эффектом. Производители обычно заявляют, что этот ток в среднем составляет 50 мкА, с максимальным значением до 100 мкА. Однако, в реальных условиях он может достигать даже 500 мкА. Из-за этого необходимо применять делитель напряжения R1-R2, чтобы обеспечить стабильное выходное напряжение. Для этого требуется пропускать через делитель ток около 5 мА. Это означает, что сопротивление R1 не может превышать 240 Ом. Следует отметить, что именно такое значение сопротивления рекомендуется в схемах включения, представленных в документации.

Для расчета сопротивления резистора R2 следует руководствоваться следующей формулой

$$R2 = R1 \times \left(\frac{U_{\text{вых}}}{U_{\text{оп}}} - 1\right), \quad (5)$$

где R1 – сопротивление резистора R1, Ом;

Uвых – выходное напряжение, В;

Uоп – опорное напряжение, В.

Однако также возможно использование одного подстрочного резистора R2 на выводе подстройки стабилизатора напряжения без включения резистора R1, так как лишь результирующее напряжение зависит от размера падения напряжения на резисторе R2. В методических данных [20] указано рекомендуемое значение сопротивления данного резистора для напряжения питания 12 В, что соответствует номинальному напряжению блока питания. Рекомендуемое значение сопротивления R2 – 2 кОм.

Для исключения разряда аккумулятора при отсутствии питания на входную цепь (стабилизаторы и источник питания) на выходе второго стабилизатора напряжения предусмотрен диод Шоттки SR24, отличный от обычного диода меньшим падением напряжения при его прямом смещении, имеющий рабочий температурный диапазон -55°С – 150°, удовлетворяющий

заданным условиям. Когда на диод Шоттки подается прямое напряжение, электроны могут легко переходить через барьерный слой с металлической стороны на сторону полупроводника P-типа. При этом возникает ток, направленный от металлического контакта к полупроводнику P-типа. В этом режиме диод Шоттки работает как прямой диод с малым падением напряжения на переходе. С описанием работы данного диода можно ознакомиться в методических данных [13]. Технические параметры SR24 предоставлены в таблице 5.

Таблица 5 – Характеристики диода Шоттки SR24

Материал полупроводника	кремний
Кол-во диодов в корпусе	1
Конфигурация диода	Одиночный
Максимальное обратное напряжение (В)	40
Максимальный (средний) прямой ток на диод, (А)	2
Максимальное прямое напряжение ,В	0.5
Обратный ток утечки, мкА	400
Рабочая температура, °С	-65…+125
Корпус	do-214aa/smb
Вес, г	0.26

Рассмотрим второй каскад схемы, принципиальная схема которого показана на рисунке 13.

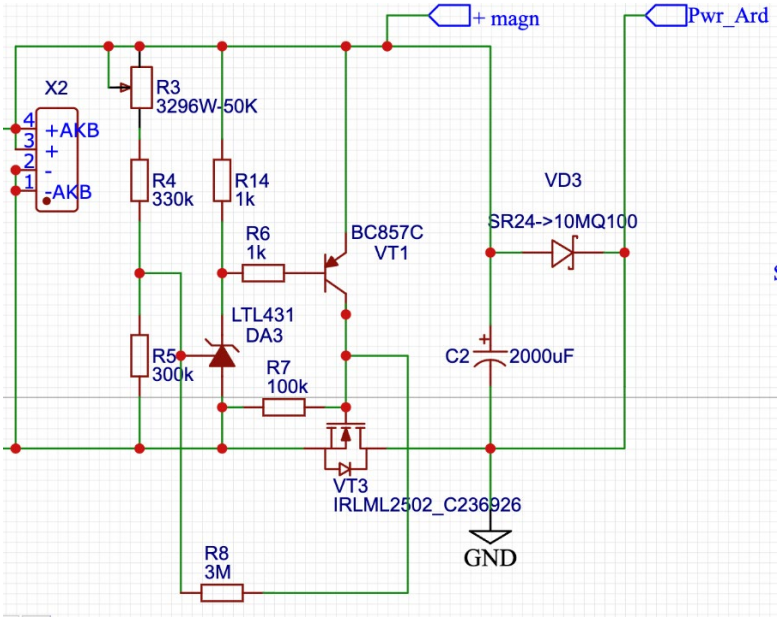

Рисунок 13 - Каскад отслеживания на нагрузке

Каскад выполняет функцию контроля разряда свинцового АКБ, реализуемый на основе шунтирующего регулятора напряжения TL431 и полевого транзистора IML2502. С данными элементами микроэлектроники можно ознакомиться в методических источниках [16] и [22].

TL431 – это прецизионный программируемый шунтирующий регулятор напряжения, использующийся для стабилизации напряжения в электронных устройствах и имеющий несколько важных особенностей. TL431 является программируемым устройством, что означает, что пользователь может легко настроить его на нужное выходное напряжение. Он также имеет встроенную защиту от перегрузки и короткого замыкания, что делает его надежным и безопасным в использовании.

TL431 может использоваться, как самостоятельное устройство, или в сочетании с другими электронными компонентами для регулирования напряжения в электронных схемах. Исходя из этого можно сделать вывод, что TL431 является надежным и универсальным регулятором напряжения, который может использоваться в схемотехнических решениях разных направлений. Характеристики TL431 приведены в таблице 6.

Таблица 6 – Характеристики TL431

Бренд	Бренд Техас Инструментс
Описание/Функция	РЕГУЛИРУЕМЫЙ ПРЕЦИЗИОННЫЙ ШУНТОВЫЙ РЕГУЛЯТОР ЗЕНЕРА
Заводская упаковка	1800 шт.
Высота	5,2 мм (макс.)
Начальная точность	2%
Длина	5,2 мм (макс.)
Производитель	Техас Инструментс
Максимальная рабочая температура	+85 C
Максимальное выходное напряжение	36 В

Второй включенный в схему транзистор является транзистором полевым IML2502, используемый для переключения логических состояний схемы. MOSFET - самый распространенный тип полевого транзистора. Он имеет три основные области: исток (Source), сток (Drain) и затвор (Gate). Управление током между истоком и стоком осуществляется с помощью напряжения на затворе. Технические параметры данного транзистора приведены в таблице 7.

Таблица 7 – Характеристики IML2505

Структура	n-канал
Максимальное напряжение сток-исток Uси,В	20
Максимальный ток сток-исток при 25 С Iси макс..А	4.2
Максимальное напряжение затвор-исток Uзи макс.,В	12
Сопротивление канала в открытом состоянии Rси вкл. (Max) при Id, Rds (on)	0.045 Ом/4.2А, 4.5В
Максимальная рассеиваемая мощность Pси макс..Вт	1.25
Крутизна характеристики, S	5.8
Корпус	Micro-3/SOT-23-3
Пороговое напряжение на затворе	1.2
Вес, г	0.05

Второй каскад имеет триггерную характеристику, соответственно может находится в двух рабочих состояниях. Пороги срабатывания устройства показаны на графике зависимости входного к выходному напряжениям, представленного на рисунке 14. Для более подробного ознакомления с принципом работы триггерной характеристики можно ознакомиться с методической информацией [15].

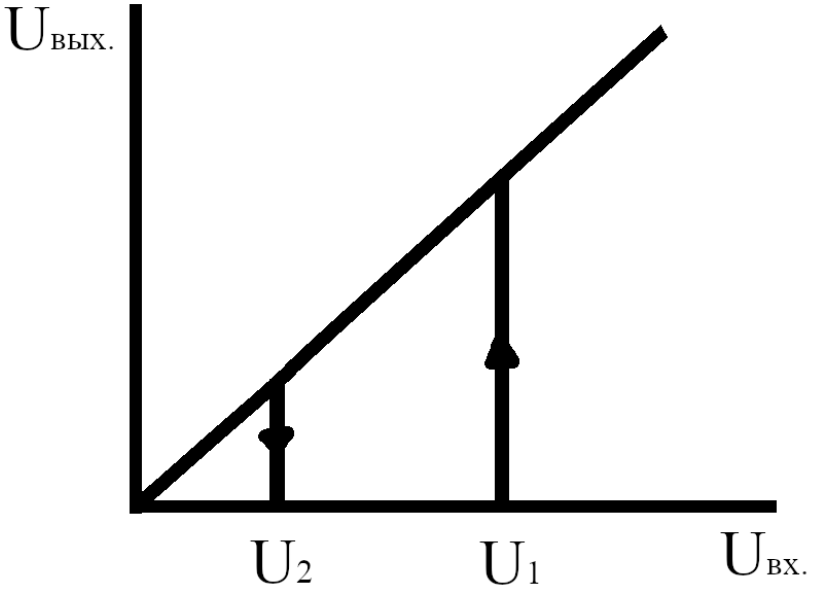

Рисунок 14 – График зависимости входного и выходного напряжений

Подключение АКБ к нагрузке обеспечивается при превышении напряжения U1, отключение при понижении до значения U2. Пока оба транзистора в контуре отключены, напряжение на входе определяется делителем, включающим сопротивления резисторов R3, R4, R5, R7 и R8. Рассчитаем общее сопротивление для контура, включающего значения сопротивлений резисторов R5, R7 и R8 согласно формуле

$$R = R5 \times \frac{R8+R7}{R5+R7+R8}, \quad (6)$$

где R5 – значение резистора R5, Ом;

R7 – значение резистора R7, Ом;

R8 – значение резистора R8, Ом.

Исходя из этого можно рассчитать результирующее сопротивление данного участка при известных величинах сопротивлений резисторов контура по формуле (6)

$$R = 300\ \text{кОм} \times \frac{3000\ \text{кОм}+100\ \text{кОм}}{300\ \text{кОм}+100\ \text{кОм}+100\ \text{кОм}} = 273{,}5\ \text{кОм}. \quad (7)$$

Значение порогового напряжения U1 рассчитывается исходя из заданного контура согласно формуле

$$U1 = \frac{2{,}5}{R} \times (R + R3 + R4), \quad (8)$$

где R - общее сопротивление резисторов R5, R7, R8, Ом;

R3 –сопротивление резистора R3, Ом;

R4 – сопротивление резистора R4, Ом.

Следовательно, можно рассчитать величину этого напряжения, применяя формулу (8), зная величины сопротивлений резисторов в контуре

$$U1 = \frac{380\ \text{кОм} + 273{,}5}{273{,}5} \times 2{,}5 = 5{,}9\ \text{В}\,. \qquad (9)$$

При открытии ключей цепь делителя на управляющем входе TL431 меняется, соответственно пороговое значения напряжения U2 рассчитывается исходя из нового контура, но прежде рассчитаем новое сопротивление контура, включающего значения сопротивлений резисторов R3, R4 и R8

$$R = R3 + R4 \times \frac{R8}{R3 + R4 + R8}\,, \qquad (10)$$

где R3 – сопротивление резистора R3, Ом;

R4 – сопротивление резистора R4, Ом;

R8 – сопротивление резистора R8, Ом.

Соответственно можно произвести расчет сопротивления данного контура, включающего вышеприведенные резисторы согласно формуле (10)

$$R = 380\ \text{кОм} \times \frac{3000\ \text{кОм}}{380\ \text{кОм} + 3000\ \text{кОм}} = 337\ \text{кОм}. \qquad (11)$$

Пороговое значения напряжения U2 рассчитывается исходя из нового контура согласно формуле

$$U2 = \frac{2{,}5}{R5} \times (R + R5), \qquad (12)$$

где R – общее сопротивление резисторов R3, R4 и R8 Ом;

R5 – сопротивление резистора R5, Ом.

Соответственно можно рассчитать величину этого напряжения, применяя формулу (12)

$$U2 = \frac{2{,}5}{300\ \text{кОм}} \times (337\ \text{кОм} + 300\ \text{кОм}) = 5{,}3\ \text{В}. \qquad (13)$$

Второй каскад обеспечивает следящий режим управления нагрузкой, подключаемой к АКБ, исходя из рассчитанных порогов срабатывания переключения режимов нагрузки для аккумуляторной батареи.

Общий каскад обеспечивает стабильную работу с заданными функциональными характеристиками и является простым схемотехническим решением из-за множества возможных вариантов компонентных элементов на рынке и отсутствия сложных схемотехнических решений для разработки дополнительной конструкторской документации.

2.4 Разработка электронной схемы и выбор элементов устройства

Для разработки электронной схемы выбран онлайн редактор принципиальных схем EasyEDA. EasyEDA - это бесплатный онлайн-инструмент для проектирования электрических схем и печатных плат. Он предоставляет удобный интерфейс для создания схем и PCB-дизайна, а также имеет встроенную библиотеку компонентов и инструменты для симуляции и проверки корректности использования элементов в проекте. EasyEDA также позволяет обеспечить совместную удаленную работу над проектами и экспортирование готовых файлов в различных форматах. Для начала работы с инструментом EasyEDA следует обратиться к электронному ресурсу [18].

Общий электрический контур разделен на 3 функциональных каскада:

- каскад обслуживания АКБ;
- каскад логического взаимодействия;
- каскад реле срабатывания;

Электрическая принципиальная схема для обслуживания АКБ, предназначенная для понижения и стабилизации значений тока и напряжения нагрузки, а также осуществляющая ждущий режим для заряда аккумуляторной батареи, реализованная на триггерной характеристике, показана на рисунке 15.

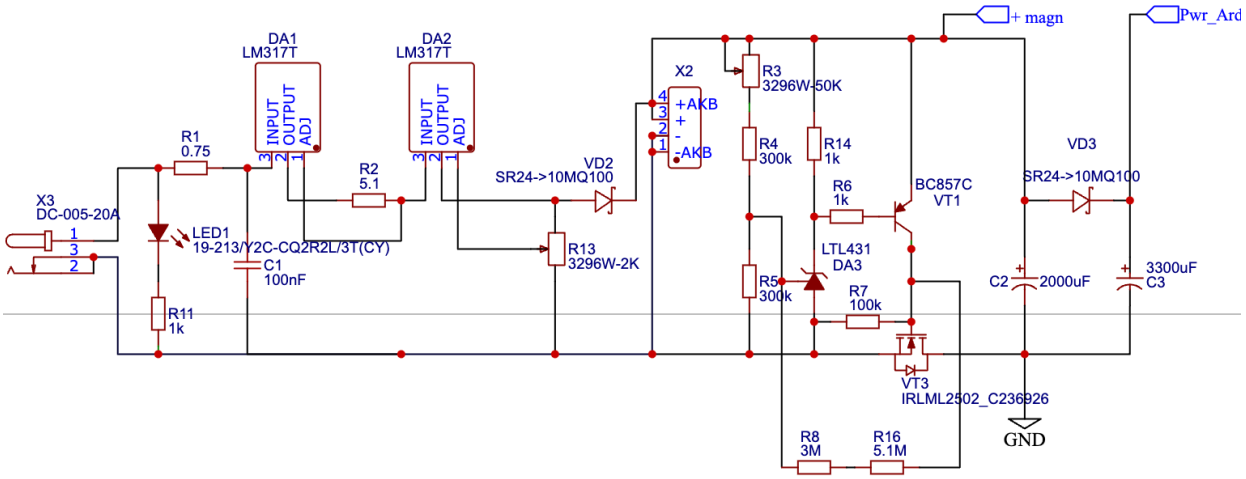

Рисунок 15 – Каскад обслуживания АКБ

Для описания взаимодействия микроконтроллера с электрическими компонентами создан каскад логического взаимодействия, включающий в себя микроконтроллер с подключенными к нему цепью питания, реле и Bluetooth модулем. Приведенный каскад показан на рисунке 16.

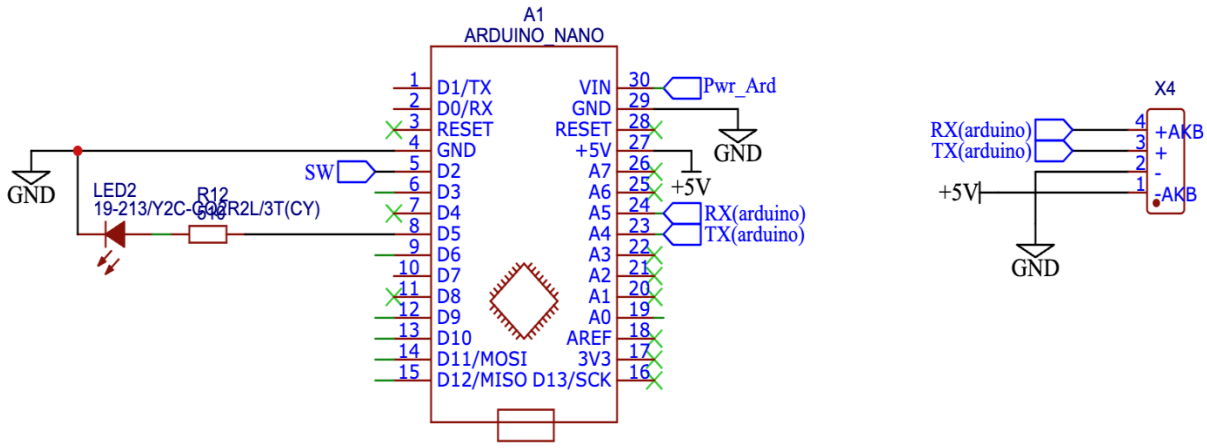

Рисунок 16 – Каскад логического взаимодействия

В качестве реализации открытия замка выбрана подача кратковременного дискретного сигнала с порта отладочной платы на реле, через краткосрочную подачу импульса, полевой транзистор обеспечивает смену логических состояний схемы, подавая краткосрочный импульс на реле срабатывания. Диод включен в схему для защиты от явления самоиндукции, в результате которого возможно пробитие транзистора, что может сказаться на работе всего комплекса. Данный каскад приведен на рисунке 17.

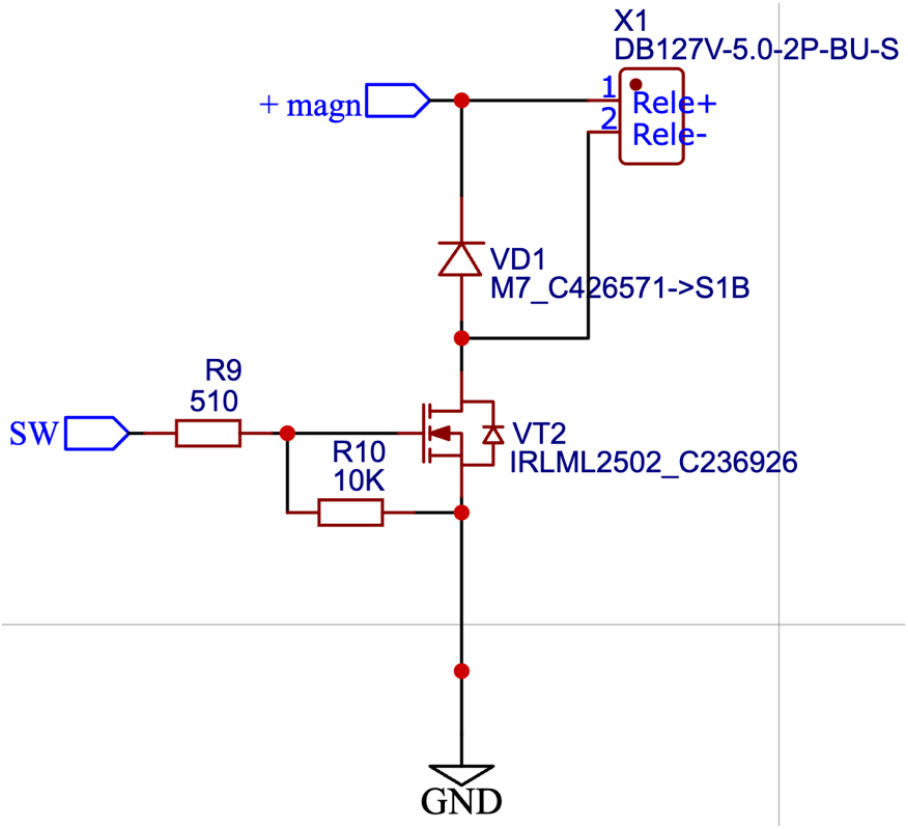

Рисунок 17 – Каскад реле срабатывания

Каскады, рассмотренные в данном контексте, представляют собой группы электронных компонентов, которые связаны между собой с использованием электрических соединений. Эти соединения достигаются путем физического соединения компонентов с помощью креплений, разъемов и пайки.

Компоненты, входящие в каскады, могут включать различные полупроводниковые элементы, такие как транзисторы, диоды, резисторы и конденсаторы, а также другие активные и пассивные элементы. Эти компоненты связываются между собой электрическими проводниками, которые обеспечивают поток электрического тока и сигналов между компонентами.

Структурная схема аппаратного комплекса приведена на рисунке 18.

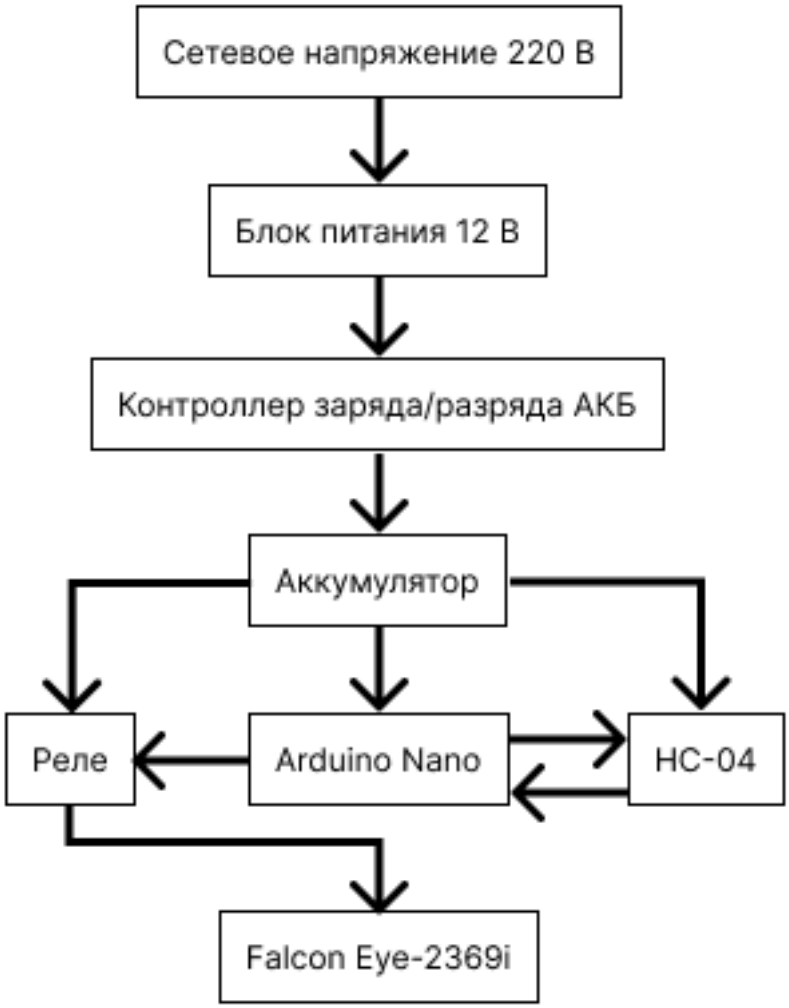

Рисунок 18 – Структурная схема аппаратного комплекса

Указаны электрические связи элементов аппаратного комплекса исходя из режимов работы master или slave.

3 Программирование устройства

Программирование Bluetooth замка включает в себя несколько шагов:

- разработка программы для микроконтроллера Arduino nano. Программа для микроконтроллера должна обрабатывать команды от Android-приложения через Bluetooth-модуль и выполнять определенные действия исходя из типа переданной ему команды;

- добавление Bluetooth-поддержки в Android-приложение. Для подключения к Bluetooth-модулю в приложении необходимо добавить поддержку Bluetooth с помощью внешних библиотек;.

- создание интерфейса пользователя. Необходимо создать пользовательский интерфейс, который будет предоставлять возможность подключения к Bluetooth-модулю и взаимодействовать с ним. Хороший интерфейс сочетает в себе эргономику и функциональность;

- обработка действий пользователя. Необходимо создать интуитивно понятное логическое взаимодействие элементов в программе, при этом не пренебрегая безопасностью системы в целом.

После выполнения всех приведенных выше шагов, пользователь должен иметь возможность подключиться к Bluetooth-модулю замка, и отправлять команды на его открытие и запрашивать смену пароля.

3.1 Аппаратная часть

Аппаратная часть разрабатываемого комплекса состоит из отладочной платы Arduino Nano. Отладочная плата Arduino - это открытая платформа для разработки и программирования устройств, основанная на языке программирования C++ и наборе аппаратных и программных инструментов и позволяющая легко разрабатывать программные решения для работы с ней из-за простоты адаптированного для новичков языка программирования и возможности легкого подключения внешних электронных устройств и

элементов с помощью вмонтированных в отладочную плату интерфейсов – портов.

3.1.1 Описание функций работы аппаратной части

Аппаратная часть разрабатываемого комплекса отвечает следующим требованиям функционала:

- установление соединения с клиентской частью, реализуемое посредством беспроводной технологии Bluetooth. Скорость передачи данных в технологии Bluetooth зависит от версии технологии и может составлять от 1 до 2,1 Мбит/с. Скорость передачи данных Bluetooth ниже, чем в Wi-Fi или Ethernet, но достаточно быстра для передачи музыки, фотографий и других файлов малого размера. Bluetooth имеет два основных режима работы: режим обнаружения и режим передачи данных. Режим обнаружения используется для поиска других устройств Bluetooth в радиусе действия, в то время как режим передачи данных используется для передачи данных между устройствами. В данном функциональном примере рассматривается режим обнаружения и устанавливания соединения;

- получение данных от клиентской части, реализуемое посредством беспроводной технологии Bluetooth в режиме передачи данных. С помощью модуля Bluetooth HC-05 для расширения функционала микроконтроллера Arduino Nano, данные, получаемые с аппаратной части передаются в порт Serial, представляющий собой аппаратный последовательный порт, который может использоваться для передачи данных между микроконтроллером и другими устройствами;

- валидация полученной информации, реализуемая с помощью логических функций, реализующих посимвольную проверку на сходство данных, написанных на языке C++. Хорошая проверка данных включает в себя несколько критериев оценки, выдавая логическое отрицание в случае наличия хотя бы одного отклонения в проверке;

3.1.2 Описание программы и алгоритма аппаратной части

Исходя из вышеописанного функционала, разработана программа для соответствия всем функциональным и логическим запросам. Полный программный код для логической работы отладочной платы Arduino показан в приложении Б. Подробно изучить принципы рабочего цикла программного кода можно в методическом источниках [20] и [21].

Основой любого языка программирования служат переменные – минимальные сущности, необходимые для хранения, передачи или изменения хранящейся в них информации. Переменные в языке Arduino имеют разные типы, которые необходимо указывать при первоначальной инициализации переменных, так как язык программирования Arduino построен на базе языка C++, являющийся строго типизированным языком, не позволяющим реализовывать переменные без указания их типа данных в связи с возможными проблемами, связанными с взаимодействием между разными типами данных. Типы переменных и их характеристики для языка Arduino показаны в таблице 8.

Таблица 8 – Типы данных языка C++

Тип	Занимаемый размер (байт)	Минимальное значение	Максимальное значение
boolean	1	false	true
byte	1	0	255
char	1	-128	127
int, short	2	-32768	32767
unsigned int	2	0	65535
long	4	-2147483648	2147483647
unsigned long	4	0	4294967295
float, double	4	-3.4028235E+38	3.4028235E+38

В актуальной версии программы инициализированы переменные rxPin и txPin, которые хранят значения аналоговых портов A4 и A5 для взаимодействия с Bluetooth модулем по режиму работы протокола обмена информацией по последовательному каналу в режиме трансмиттера (отправителя) или ресивера (получателя). Реализована переменная baudRate, хранящая в себе целочисленную переменную 9600, представляющая собой скорость передачи данных в бит/с через последовательный порт. Также реализованы 3 строковых переменные pass, masterPass, receivedString типа string для хранения пароля, мастер-пароля и получаемого с последовательного порта пароля. Инициализация переменных показана на листинге 1.

Листинг 1

```
#include <EEPROM.h>
#include <SoftwareSerial.h>
//порты передачи данных через Bluetooth модуль
const int rxPin = A4;
const int txPin = A5;
const int baudRate = 9600;
//пин перезагрузки Bluetooth модуля
const int resetPin = 4;
//логические переменные (пароль, мастер пароль, полученная
строка и пин подачи импульса на замок)
//String pass = "123456";
//String masterPass ="1234567890";
String receivedString = "";
int openPin = 2;
SoftwareSerial hc04(rxPin, txPin);
```

Стоит отметить импортируемые в начале программного кода внешние библиотеки, расширяющие функциональные возможности взаимодействия с микроконтроллером посредством внедрения уникального функционала работы портов отладочной платы и общей логики работы микроконтроллера. В актуальном коде импортированы библиотеки #include <SoftwareSerial.h> и #include <EEPROM.h>. Первая библиотека расширяет возможности обмена информацией через tx и rx протокол с помощью цифровых и аналоговых портов на отладочной плате Arduino Nano, изначально это могут делать

специально предназначенные порты RX и TX. Вторая библиотека позволяет работать с данными в энергонезависимой памяти EEPROM микроконтроллеров Arduino. Данный вид памяти позволяет сохранять информацию, не переживая за ее сохранность после многочисленных перезагрузок устройства, разработчиками микроконтроллера заявлена стабильная работа при 100000 перезаписывании памяти.

Программа для микроконтроллеров Arduino, написанная на языке Arduino всегда должна включать в себя основные функциональные блоки: setup и loop, представляющие собой основные выполняемые блоки программы. Метод setup() представляет собой активность, выполняющуюся единовременно после начала работы микроконтроллера с загруженной программой. Согласно современным паттернам проектирования программы принято включать в этот функциональный блок инициализацию переменных, назначения портов и выполнение первоначального функционала программы для инициализации стабильной работы программы. Блок setup() показан на листинге 2.

Листинг 2

```
void setup() {
  //подача сигнала HIGH в течение 1 секунды на пин перезагрузки
Bleutooth адаптера
  digitalWrite(resetPin, HIGH);
  delay(1000);
  digitalWrite(resetPin, LOW);
  hc04.begin(baudRate);
  Serial.begin(baudRate);
  //clearEEPROM();
  //changePass(pass);
  Serial.println("Password written to EEPROM: " + readPass());
  //changeMasterPass(masterPass);
  Serial.println("Master password written to EEPROM: " +
readMasterPass());
  pinMode(openPin, OUTPUT);
  Serial.println("Ready to receive string values from HC-
04...");
}
```

В актуальной версии программы реализована изначальная перезагрузка Bluetooth модуля для обеспечения стабильной работы модуля. В блок setup() включены назначение типа вывода OUTPUT для переменной openPin, представляющей собой второй порт на отладочной плате для подачи с него импульса на замок. Метод реализуется встроенным в базовый синтаксис языка Arduino методом pinMode(openPin, OUTPUT), позволяющим изначально настроить назначение порта на режим трансмиттера – отдачи данных, в случае, если порт принимает информацию, необходимо заменить OUTPUT на INPUT для смены назначения порта, как принимающей стороны. Также установлен режим скорости работы последовательного порта методом hc04.begin(baudRate) для Bluetooth модуля и Serial.begin(baudRate) для Serial порта микроконтроллера. Также для информирования пользователя и готовности работы микроконтроллера в последовательное окно выводится информация методом Serial.println(), который позволяет с установленным значением скорости обмена данными выводить информацию, окончание ln означает перенос строки при вызове нового метода.

Рассмотрим основной блок кода loop(), представляющий из себя обязательную структурную и функциональную часть любой программы для микроконтроллеров, представляющий из себя цикл, не имеющий точки выхода из него, соответственно работа программы представляет собой нескончаемый цикл с прописанным разработчиком условием выхода из него. Основной функциональный блок кода с внешними методами, используемыми в нем, показаны на листинге 3.

Листинг 3

```
void loop() {
  if (hc04.available() > 0) {
    receivedString = hc04.readString();
    Serial.print("Received string: ");
    Serial.println(receivedString);
    /*Условие сверки мастер пароля из полученного пароля,
    в случае удачи меняется значение пароля в EEPROM*/
    if(compareMasterPass(readMasterPass(), receivedString)) {
      changePass(receivedString.substring(10, 16));
```

```
      Serial.println("Password changed: " + readPass());
    }
    /*Условие сверки пароля из полученного пароля,
    в случае успеха открытие замка путем подачи импульса
    на замок длительностью 50 мс.*/
    else if(comparePass(readPass(), receivedString)){
      digitalWrite(openPin, HIGH);
      delay(50);
      digitalWrite(openPin, LOW);
      Serial.println("Unlocked");
    }
    /*в случае получения неверного пароля ардуино уходит в паузу
на 10 секунд*/
    else {
      Serial.println("Do not even try to to something!");
      delay(10000);
    }
  }
}
//метод полной очистки EEPROM, используется при начале работы
программы
void clearEEPROM() {
  for (int i = 0; i < EEPROM.length(); i++) EEPROM.update(i, 0);
}
//метод смены мастер-пароля в EEPROM, его использование на
данный момент упразднено в связи с логической бессмысленностью
void changeMasterPass(String pass) {
  char charBuf[pass.length() + 1];
  pass.toCharArray(charBuf, pass.length() + 1);
  for (int i = 0; i < 10; i++) {
    EEPROM.write(i, pass[i]);
  }
}
//метод смены пароля в EEPROM, используется при удачной сверке
мастер-пароля из приходящей строки
void changePass(String pass) {
  char charBuf[pass.length() + 1];
  pass.toCharArray(charBuf, pass.length() + 1);
  for (int i = 0; i < 6; i++) {
    EEPROM.write(10 + i, pass[i]);
  }
}
//метод чтения значения пароля из EEPROM, используется при
сверке с приходящим паролем
String readPass() {
  char charBuf[6 + 1];
  for (int i = 0;i < 6; i++) {
    charBuf[i] = EEPROM.read(10 + i);
  }
  charBuf[6] = '\0';
  return String(charBuf);
}
```

```
//метод сверки мастер пароля, используется при сверке с
приходящим паролем
String readMasterPass() {
  char charBuf[10 + 1];
  for (int i = 0;i < 10; i++) {
    charBuf[i] = EEPROM.read(i);
  }
  charBuf[10] = '\0';
  return String(charBuf);
}
//метод сверки мастер-паролей, используется для смены пароля в
EEPROM
bool compareMasterPass(String pass, String masterPass) {
  for (int i = 0; i < 10; i++) {
    if ( pass[i] != masterPass[i]) {
      return false;
    }
  }
  return true;
}
//метод сверки паролей, используется для открытия замка
bool comparePass(String pass, String secondPass) {
  for (int i = 0; i < 6; i++) {
    if ( pass[i] != secondPass[i] ) {
      return false;
    }
  }
  return true;
}
```

Большинство времени программа работает в режиме ожидания получения какой либо информации с помощью условия if (hc04.available() > 0), вхождением в которое является появление 1 и более символов на Bluetooth Serial канале. Внутри условия имеются два условия:

- условие сверки пароля – для открытия замка приходящая информация сверяется посимвольно со значением пароля в энергонезависимой памяти микроконтроллера методом comparePass(), который возвращает логическую единицу в случае удачной посимвольной проверки двух входящих в него аргументов – значений приходящего и хранимого паролей. С помощью импортированной библиотеки EEPROM.h можно легко взаимодействовать с элементами энергонезависимой памяти с помощью методов EEPROM.read() и EEPROM.write() для чтения и записи информации. При чем следует обратить внимание, что информация записывается и читается посимвольно с

помощью индекса или адреса каждой ячейки энергонезависимой памяти, пример хранения строковой переменной со значением “hello” в памяти EEPROM показан на рисунке 19.

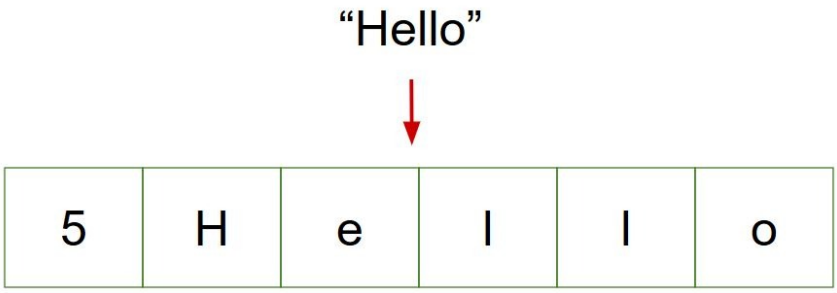

Рисунок 19 – Хранение информации в EEPROM

Реализованы запись и чтение информации в памяти EEPROM, при чем следует обратить внимание, значения пароля и мастер-пароля записываются друг за другом и читаются так же, первые 10 индексов занимаются мастер-паролем, следующие 6 символов занимаются паролем. Соответственно для чтения значения пароля из EEPROM следует считать значения индексов 10, 11, 12, 13, 14 и 15. Подробнее с принципом работы памяти EEPROM можно ознакомиться в источнике [7];

- условие сверки мастер-пароля – для смены передаваемого пароля задействован мастер-пароль, представляющий из себя строковое значение из 10 символов. Сверка двух входящего и хранимого значений мастер-паролей обеспечивается методом compareMasterPass с чтением первых 10 символов из энергонезависимой памяти с помощью метода readMasterPass() и циклом чтения, включающим в себя 10 итераций посимвольной записи значения мастер-пароля из приходящей строки, в случае удачной сверки следующие 6 символов приходящей строки посимвольно записываются в память EEPROM вместо старого значения мастер-пароля методом changePass();

- условие – исключение – в случае несоответствия входящей величины двум условиям, в целях защиты, предусмотрен режим паузы длительностью в 1 минуту;

Реализована функциональная часть аппаратной части комплекса программно-аппаратного комплекса. В программном коде преобладает функциональное программирование, так как для хранения и передачи информации между компонентами программы используются прописанные методы работы с информацией для лаконичности основного функционального блока кода. Блок схема программы для отладочной платы Arduino Nano приведена в приложении А.

3.2 Клиентская часть

Клиентская часть разрабатываемого корпуса состоит из мобильного устройства с операционной системой Android. Устройства на операционной системе Android являются доступным решением для разработки программных решений, так как существует множество языков программирования для разработки мобильных приложения для операционной системы Android, в данном программном коде используется язык программирования Java из-за наличия у студента знаний функционального программирования с паттернами ООП в данном языке программирования. Также Java является вторым по популярности языком программирования для разработки мобильных приложений и имеет прямую интеграцию с программными инструментами JetBrains и программой Android Studio, позволяющую собирать и экспортировать готовые программные решения.

3.2.1 Описание функций работы клиентской части

Клиентская часть разрабатываемого комплекса отвечает следующим требованиям функционала:

- установление соединения с аппаратной частью, реализуемое посредством беспроводной технологии Bluetooth. Устройством – обнаружителем является мобильное устройство с операционной системой Android, отвечающее всем правилам современных Bluetooth модулей. Аппаратная часть принимает запрос на соединение от устройства – обнаружителя, чем является клиентская часть;

- отправка данных аппаратной части, реализуемое посредством беспроводной технологии Bluetooth в режиме передачи данных. С помощью встроенного в устройство – передатчик Bluetooth модуль осуществляется передача данных;

- изменение пароля, передаваемого к аппаратной части. Данный функционал реализуется посредством валидации “мастер пароля”, отправляемого к аппаратному комплексу для его последующей валидации и сохранении нового пароля в случае удачной проверки.;

- разрыв соединения с аппаратным комплексом, реализуемое с помощью пользовательского интерфейса;

Паттерны разработки программного кода взяты исходя из информации, приведенной в источнике [9].

3.2.2 Описание программы и алгоритма клиентской части

Программа для клиентской части, а именно устройства с операционной системой Android разрабатывается в IDE Android Studio. Android Studio - это интегрированная среда разработки (IDE), предназначенная для разработки мобильных приложений на операционной системе Android. Android Studio разработана компанией Google на основе IntelliJ IDEA. Она предоставляет разработчикам широкий набор инструментов для создания, отладки и тестирования приложений для Android, включая редактор кода, визуальный макет-редактор, инструменты для управления проектом, средства отладки,

эмуляторы устройств и многое другое. Для подробного изучения принципов работы Android программы можно ознакомиться с информацией, приведенной в методических источниках [11], [17] и [8].

Android Studio имеет обширную документацию и сообщество, которое поддерживает разработчиков, помогая им решать проблемы и получать новые знания и навыки. Также Android Studio интегрирована с сервисами Google, такими как Google Play Store, Google Analytics, Firebase и другими, что облегчает разработку и тестирование приложений. Разработка ведется на языке программирования Java, с которым можно ознакомитсья подробнее в источнике [9].

Проект Android Studio на языке Java состоит из нескольких компонентов:

- файлы исходного кода, содержащие исходный код на языке Java. Они разбиты на пакеты и классы, каждый из которых выполняет определенную функцию в приложении;

- xml-файлы макетов, определяющие пользовательский интерфейс (UI) приложения. Они содержат описание элементов интерфейса, таких как кнопки, текстовые поля и изображения, а также информацию о том, как они должны быть расположены и выглядеть на экране;

- ресурсы, содержащие данные, необходимые для приложения, такие как изображения, звуки, строки, цвета и т.д. Они хранятся в различных папках внутри проекта;

- библиотеки, которые можно использовать в приложении для выполнения определенных задач, таких как работа с сетью или базами данных. Библиотеки могут быть включены в проект как в виде исходного кода, так и в виде готовых JAR-файлов;

- файлы конфигурации, которые содержат информацию о настройках приложения, таких как версия SDK, API-ключи и настройки сборки. Они используются для настройки проекта и сборки приложения;

- файлы Gradle, которые используются для управления зависимостями и сборки приложения. Они содержат информацию о библиотеках, используемых в проекте, а также инструкции для сборки приложения в определенный формат, такой как APK-файл;

Все эти компоненты совместно образуют проект Android Studio на языке Java и позволяют разработчикам создавать высококачественные приложения для платформы Android.

В Android активность (Activity) представляет собой экран приложения, на котором пользователь может взаимодействовать с интерфейсом. Каждая активность может быть запущена и управляема самостоятельно. Она обычно используется для представления конкретного пользовательского интерфейса или взаимодействия с пользователем. Активность в Android представляется классом, который содержит логику для создания и управления пользовательским интерфейсом. Активность может включать в себя элементы управления, такие как кнопки, текстовые поля, списки и другие виджеты, которые пользователь может использовать для взаимодействия с приложением. Каждая активность должна быть зарегистрирована в файле манифеста приложения, который определяет ее функциональность и доступность. При запуске приложения Android определяет, какая активность должна быть показана первой, и вызывает ее метод onCreate().

Взаимодействие между активностями может быть выполнено с помощью намерений (Intents), которые позволяют приложению передавать данные между различными компонентами и запускать другие активности.

В Android Studio можно создавать и настраивать активности с помощью визуального редактора интерфейса или путем написания кода на языке Java. Каждая активность имеет свой жизненный цикл, который состоит из ряда состояний, таких как создание, запуск, приостановка, возобновление и уничтожение. Эти состояния могут быть использованы для управления процессом создания и управления пользовательским интерфейсом. Более подробно о активностях можно узнать в источнике [21].

Актуальное Android приложение имеет 3 активности:

- MainActivity - это основная активность в Android - приложении, которая запускается при старте приложения и обычно является точкой входа в приложение. Она может содержать элементы управления, такие как кнопки, текстовые поля и другие виджеты, которые пользователь может использовать для взаимодействия с приложением. MainActivity обычно содержит логику для создания и управления главным пользовательским интерфейсом приложения. Она может вызывать другие активности и фрагменты в зависимости от потребностей приложения. Например, если приложение содержит несколько различных экранов, то MainActivity может вызывать другие активности для отображения этих экранов. MainActivity имеет свой жизненный цикл, который состоит из ряда состояний, таких как создание, запуск, приостановка, возобновление и уничтожение. Каждое состояние может быть использовано для управления процессом создания и управления пользовательским интерфейсом. Например, если приложение должно сохранять состояние приложения при изменении ориентации экрана, то можно использовать метод onSaveInstanceState() в MainActivity для сохранения состояния.

В Android Studio MainActivity создается по умолчанию при создании нового проекта. Он может быть настроен и изменен в соответствии с требованиями приложения. Исходный программный код активности MainActivity представлен в приложении В. Приведенный код является главной активностью, с которой начинается рабочий цикл всего приложения для устройств Android, написанный на языке Java с использованием расширяющей библиотеки Android. Рассмотрим методы, переменные и программный код.

Метод onCreate() является переопределенным методом из класса AppCompatActivity и выполняется при создании активности. В методе происходит установка разметки активности с помощью метода

setContentView(R.layout.activity_main), где activity_main - это XML-файл с разметкой пользовательского интерфейса.

Инициализируются переменные btnPaired и devicelist, связанные с кнопкой и списком устройств в пользовательском интерфейсе соответственно.

Вызывается экземпляр класса BluetoothAdapter с помощью метода BluetoothAdapter.getDefaultAdapter(). Если Bluetooth-адаптер недоступен (myBluetooth==null), выводится Toast-сообщение о недоступности устройства Bluetooth, и активность завершается с помощью метода finish(). Если Bluetooth-адаптер выключен (!myBluetooth.isEnabled()), открывается диалоговое окно запроса на включение Bluetooth с помощью интента BluetoothAdapter.ACTION_REQUEST_ENABLE, и результат запроса обрабатывается в методе onActivityResult(), указанном соответствующим requestCode.

Устанавливается обработчик события клика на кнопку btnPaired, который вызывает метод pairedDevicesList(). Метод pairedDevicesList() выполняет поиск и вывод списка спаренных с исходным устройством других Bluetooth-устройств. Показывается множество спаренных устройств с помощью метода myBluetooth.getBondedDevices(). Если количество спаренных устройств больше нуля, каждое устройство добавляется в список list в формате "Имя устройства \n Адрес устройства". Если спаренных устройств не найдено, выводится Toast-сообщение о их отсутствии. Создается экземпляр ArrayAdapter, связанный с ListView devicelist, с использованием списка list в качестве источника данных. Устанавливается обработчик щелчков по элементам списка с помощью метода devicelist.setOnItemClickListener(myListClickListener).

Анонимный класс AdapterView.OnItemClickListener реализует интерфейс AdapterView.OnItemClickListener. При щелчке на элементе списка извлекается информация о названии и адресе устройства, сохраненная в

строке info. Адрес устройства извлекается из строки info с помощью метода info.substring(info.length()-17).

Инструмент для построения графических имплементаций в активности позволяет легко построить отображение графических элементов в окне активности без необходимости создавать элементы собственноручно, можно выбрать готовые логические конструкции из множества предустановленных в базовый Android SDK. Также важно привязать значения кнопок и текстовых полей к триггерам срабатывания определенных методов для возможности работы с интерфейсом. Интерфейс активности показан на рисунке 20.

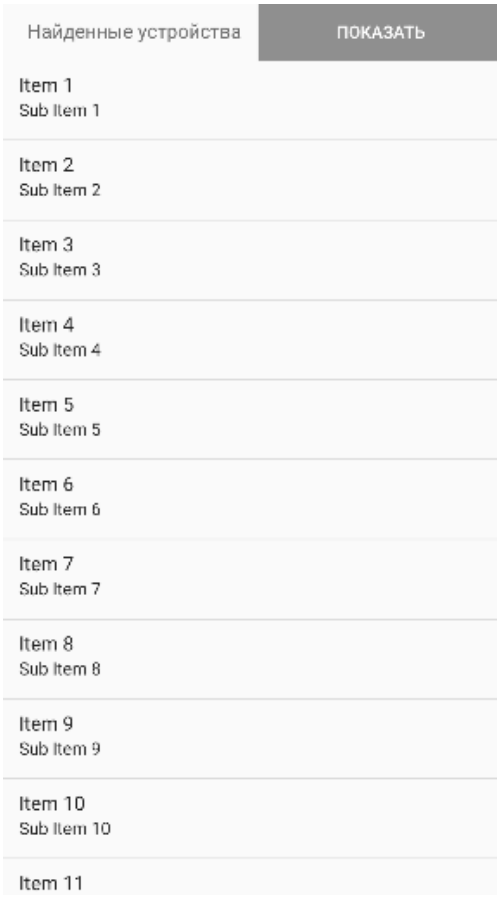

Рисунок 20 – Графический интерфейс MainActivity

- PassControl - это вторичная активность в Android - приложении, которая запускается после подключения к выбранному устройству в MainActivity для последующего выбора пункта управления меню. Исходный код активности PassControl представлен на листинге 4.

Листинг 4

```
package infoaryan.in.hc05_bluetooth;
import androidx.appcompat.app.AppCompatActivity;
import android.annotation.SuppressLint;
import android.app.ProgressDialog;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import java.io.IOException;
import java.util.UUID;

public class PassControl extends AppCompatActivity {
    Button btnSend, btnDis, btnChange;
    String address = null;
    TextView lumn;
    private ProgressDialog progress;
    BluetoothAdapter myBluetooth = null;
    static BluetoothSocket btSocket = null;
    static String pass="heLlo1";
    private boolean isBtConnected = false;
    static final UUID myUUID = UUID.fromString("00001101-0000-
1000-8000-00805F9B34FB");

    @SuppressLint("MissingInflatedId")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_led_control2);
        Intent intent = getIntent();
        address =
intent.getStringExtra(MainActivity.EXTRA_ADDRESS);
        btnSend =  findViewById(R.id.button2);
        btnDis = findViewById(R.id.button4);
        btnChange = findViewById(R.id.button3);

        new PassControl.ConnectBT().execute();

        btnSend.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick (View v) {
                sendSignal(pass);
            }
        });
```

```
        btnChange.setOnClickListener(new View.OnClickListener()
{
            @Override
            public void onClick(View v) {
                openNextActivity(v);
            }
        });
        btnDis.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick (View v) {
                Disconnect();
            }
        });
    }

    private void sendSignal ( String number ) {
        if ( btSocket != null ) {
            try {

btSocket.getOutputStream().write(number.toString().getBytes());
            } catch (IOException e) {
                msg("Error");
            }
        }
    }

    private void openNextActivity(View v) {
        Intent intent = new Intent(this, SetPass.class);
        startActivity(intent);
    }
    private void Disconnect () {
        if ( btSocket!=null ) {
            try {
                btSocket.close();
            } catch(IOException e) {
                msg("Error");
            }
        }

        finish();
    }

    private void msg (String s) {
        Toast.makeText(getApplicationContext(), s,
Toast.LENGTH_LONG).show();
    }

    private class ConnectBT extends AsyncTask<Void, Void, Void>
{
        private boolean ConnectSuccess = true;

        @Override
        protected  void onPreExecute () {
```

```
            progress = ProgressDialog.show(PassControl.this,
"Соединение...", "Пожалуйста, подождите!");
        }

        @Override
        protected Void doInBackground (Void... devices) {
            try {
                if ( btSocket==null || !isBtConnected ) {
                    myBluetooth =
BluetoothAdapter.getDefaultAdapter();
                    BluetoothDevice dispositivo =
myBluetooth.getRemoteDevice(address);
                    btSocket =
dispositivo.createInsecureRfcommSocketToServiceRecord(myUUID);

BluetoothAdapter.getDefaultAdapter().cancelDiscovery();
                    btSocket.connect();
                }
            } catch (IOException e) {
                ConnectSuccess = false;
            }

            return null;
        }

        @Override
        protected void onPostExecute (Void result) {
            super.onPostExecute(result);

Andrew Gipsy, [28.05.2023 14:16]
if (!ConnectSuccess) {
                msg("Не удалось установить соединение.");
                finish();
            } else {
                msg("Соединение установлено");
                isBtConnected = true;
            }

            progress.dismiss();
        }
    }
}
```

Приведенный код представляет активность приложения на устройствах Android для управления Bluetooth-подключенным устройством. Рассмотрим элементы программного кода.

Импорты:

- импорт класса AppCompatActivity из пакета androidx.appcompat.app для создания активности;

- импорт класса ProgressDialog из пакета android.app для отображения диалогового окна прогресса;

- импорт класса BluetoothAdapter из пакета android.bluetooth для работы с Bluetooth адаптером;

- импорт класса BluetoothDevice из пакета android.bluetooth для работы с Bluetooth устройствами;

- импорт класса BluetoothSocket из пакета android.bluetooth для работы с Bluetooth сокетом;

- импорт класса Intent из пакета android.content для перехода между активностями;

- импорт класса AsyncTask из пакета android.os для выполнения задач в фоновом режиме;

- импорт класса Bundle из пакета android.os для передачи данных между компонентами Android;

Объявление переменных:

- объявление кнопок btnSend, btnDis и btnChange;

- переменная address для хранения адреса Bluetooth устройства;

- объявление текстового поля lumn;

- прогресс-диалог ProgressDialog для отображения процесса соединения;

- переменная myBluetooth для работы с Bluetooth адаптером;

- статическая переменная BluetoothSocket для Bluetooth сокета;

- статическая переменная pass для хранения пароля;

- флаг isBtConnected, указывающий на состояние Bluetooth соединения;

- статическая переменная UUID для хранения уникального идентификатора устройства для Bluetooth соединения;

Метод onCreate() вызываемый при создании активности, устанавливает разметку активности с помощью метода setContentView(R.layout.activity_led_control2), где activity_led_control2 - это XML-файл разметки активности, получает адрес Bluetooth устройства из

предыдущей активности, связывает переменные с элементами пользовательского интерфейса, создает экземпляр класса ConnectBT и запускает его с помощью метода execute(), назначает обработчики событий кнопок btnSend, btnChange и btnDis. Метод sendSignal() отправляет сигнал (пароль) на Bluetooth устройство через Bluetooth сокет. Если сокет btSocket не равен null, то отправляет сигнал с помощью метода getOutputStream().write(). Метод openNextActivity() открывает следующую активность (SetPass) с помощью объекта Intent. Метод Disconnect() разрывает Bluetooth соединение путем закрытия сокета btSocket. Метод msg() отображает всплывающее сообщение с помощью класса Toast.

Внутренний класс ConnectBT необходим для установления Bluetooth соединения с удаленным устройством в фоновом режиме, переопределяет методы onPreExecute(), doInBackground() и onPostExecute(). В методе onPreExecute() отображается прогресс-диалог. В методе doInBackground() выполняется установка Bluetooth соединения с удаленным устройством. В методе onPostExecute() проверяется успешность установки соединения и выводится соответствующее сообщение.

С помощью инструмента для построения графических имплементаций построен простейший интерфейс для взаимодействия с логическими конструкциями программы. Также важно привязать значения кнопок и текстовых полей к триггерам срабатывания определенных методов для возможности работы с интерфейсом. Интерфейс активности показан на рисунке 21.

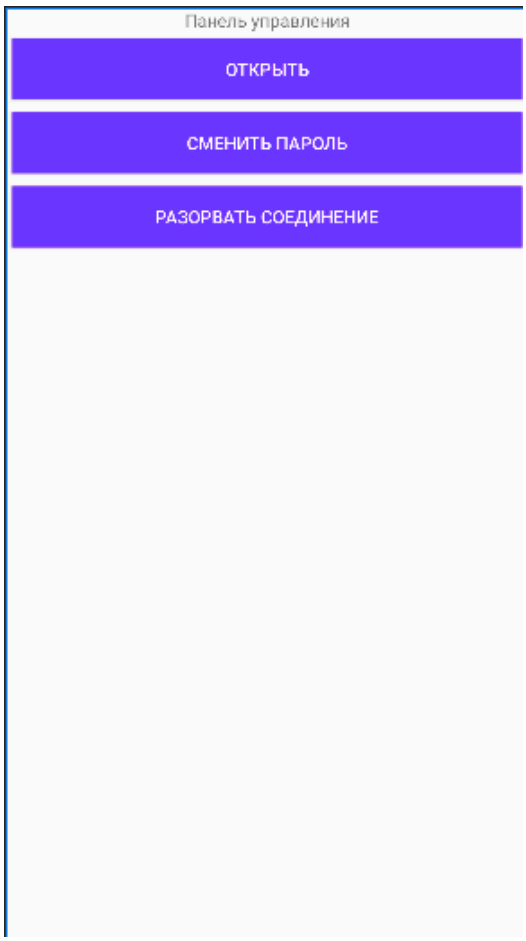

Рисунок 21 – Графический интерфейс PassControl

- SetPass – это вторичная активность в Android – приложении, которая запускается после выбора пользователем пункта меню “Сменить пароль” в активности PassControl и предоставляет функциональный интерфейс для смены пароля доступа к замку. Исходный код активности представлен на листинге 5.

Листинг 5

```
package infoaryan.in.hc05_bluetooth;
import static infoaryan.in.hc05_bluetooth.PassControl.btSocket;

import android.annotation.SuppressLint;
import android.bluetooth.BluetoothAdapter;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
```

```
import androidx.appcompat.app.AppCompatActivity;

import java.io.IOException;
import java.util.UUID;



public class SetPass extends AppCompatActivity {
    String address = null;
    Button sendPass, go_back;
    BluetoothAdapter myBluetooth = null;
    static final UUID myUUID = UUID.fromString("00001101-0000-
1000-8000-00805F9B34FB");

    @SuppressLint("MissingInflatedId")
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.set_pass);
        Intent intent = getIntent();
        final EditText newPass, masterPasss;
        newPass = (EditText) findViewById(R.id.newPass);
        masterPasss = (EditText) findViewById(R.id.masterPass);
        sendPass = findViewById(R.id.sendPass);
        go_back = findViewById(R.id.goBack);

        sendPass.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick (View v) {
                if (masterPasss.getText().toString().length() ==
10 && newPass.getText().toString().length() == 6) {
                    sendSignal(masterPasss +
newPass.getText().toString());
                } else {
                    msg("Недопустимая длина пароля");
                }
            }
        });
        /*sendPass.setOnClickListener(new View.OnClickListener()
{
            @Override
            public void onClick (View v) {
                if (masterPasss.getText().toString().length() ==
4 && newPass.getText().toString().length() == 6) {
                    if
(masterPasss.getText().toString().equals(masterPass)) {
                        sendSignal(masterPass +
newPass.getText().toString());
                        pass = newPass.getText().toString();
                        msg("Пароль изменён на " +
String.valueOf(newPass.getText()) +"!");
                    } else {
                        msg(masterPass + " не " +
String.valueOf(masterPasss.getText()));
```

```
                    }
                } else {
                    msg("Недопустимая длина пароля");
                }
            }
        });*/

        go_back.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick (View v) {
                onBackPressed();
            }
        });
    }

    private void sendSignal ( String number ) {
        if ( btSocket != null ) {
            try {

btSocket.getOutputStream().write(number.toString().getBytes());
            } catch (IOException e) {
                msg("Error");
            }
        }
    }


    private void msg (String s) {
        Toast.makeText(getApplicationContext(), s,
Toast.LENGTH_LONG).show();
    }

}
```

Приведенный код представляет активность приложения на устройствах Android для установки пароля через Bluetooth соединение. Рассмотрим программный код данной активности.

Импорт import static infoaryan.in.hc05_bluetooth.PassControl.btSocket является методом статической переменной btSocket из класса PassControl, что позволяет использовать ее непосредственно в этом классе без префикса PassControl.

Переменная address служит для хранения адреса Bluetooth устройства. Button sendPass, go_back;: Объявление кнопок sendPass и go_back.

Переменная myBluetooth служит для работы с Bluetooth адаптером.

Статическая переменная UUID служит для хранения уникального кода утсройства UUID Bluetooth соединения.

Метод onCreate() вызывается при создании активности, устанавливает разметку активности с помощью метода setContentView(R.layout.set_pass), где set_pass - это XML-файл с пользовательским интерфейсом, получает интент, переданный в эту активность, инициализирует объекты newPass и masterPasss, связанные с EditText в пользовательском интерфейсе, устанавливает обработчики нажатия на кнопки sendPass и go_back.

При нажатии на кнопку sendPass выполняется проверка длины введенных значений masterPasss и newPass. Если длина masterPasss равна 10 и длина newPass равна 6, вызывается метод sendSignal() с комбинированным паролем. Если длины паролей недопустимы, выводится сообщение через метод msg() с сообщением о недопустимой длине паролей, так как в логике работы с паролями заложены фиксированные длины пароля с мастер паролем.

При нажатии на кнопку go_back вызывается метод onBackPressed(), который возвращает пользователя на предыдущую активность.

Метод sendSignal() отправляет сигнал через Bluetooth соединение. Если btSocket не является пустым, вызывается OutputStream для отправки данных в виде байтового массива.

Метод msg() выводит на экран сообщение в виде Toast-уведомления на экране устройства.

С помощью инструмента для построения графических имплементаций построен простейший интерфейс для взаимодействия с логическими конструкциями программы. Также важно привязать значения кнопок и текстовых полей к триггерам срабатывания определенных методов для возможности работы с интерфейсом. Интерфейс активности показан на рисунке 22.

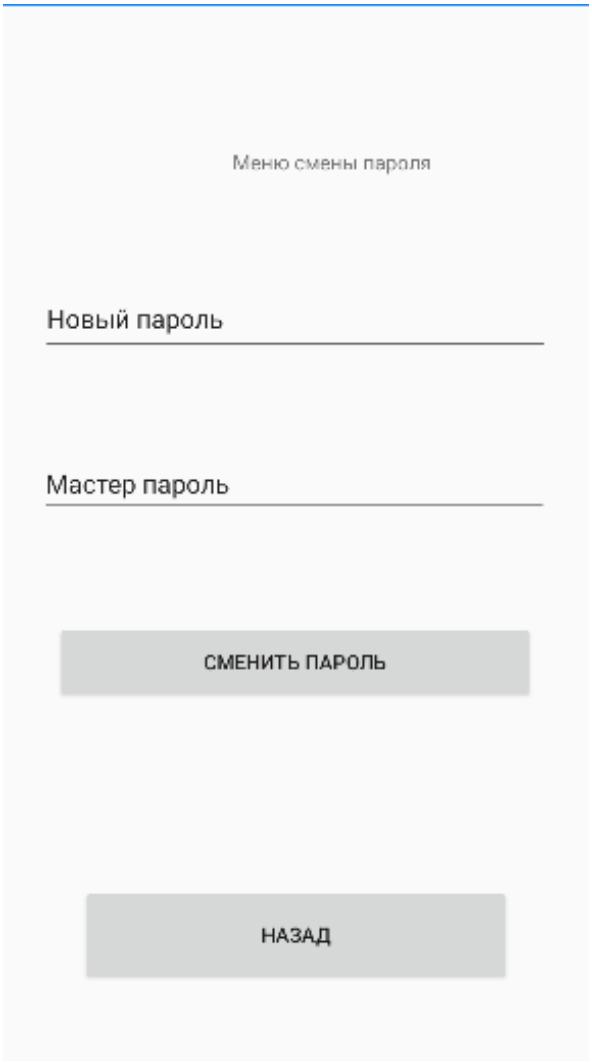

Рисунок 22 – Графический интерфейс SetPass

Реализован рабочий программный комплекс функционального пользовательского интерфейса для работы с замком, а именно реализация открытия замка и смена передаваемого пароля в целях избежания утечки статичного, не меняющегося пароля.

4 Конструкторская часть

Конструкторская часть является одним из важнейших этапов разработки устройства, так как разработчику следует ответственно подойти к материалам и технологии создания каждого конструктивного элемента устройства исходя из особенностей технического решения.

4.1 Разработка печатной платы и узла питания

Разработка печатной платы является важным этапом разработки устройства, так как после построения электрической принципиальной схемы важно правильно ее преобразовать исходя из габаритов и физических свойств материалов печатной платы.

Онлайн сервис EasyEDA предоставляет необходимый функционал для дальнейшего преобразования электрической принципиальной схемы в формат PCB, содержащий информацию о дизайне печатной платы, такую как расположение компонентов, проводников, отверстий и других элементов. Этот формат используются для передачи информации о дизайне печатной платы из программы проектирования PCB в программу производства, которая занимается созданием физической копии печатной платы. С особенностями разработки в EasyEda можно ознакомиться в источнике 18. Таким образом реализован PCB файл разрабатываемого устройства, без необходимости использования дополнительного программного обеспечения. На участке 10x10 удалось разместить две копии схемы, позволяя оптимизировать габаритные размеры. Разработанный участок PCB показан на рисунке 23.

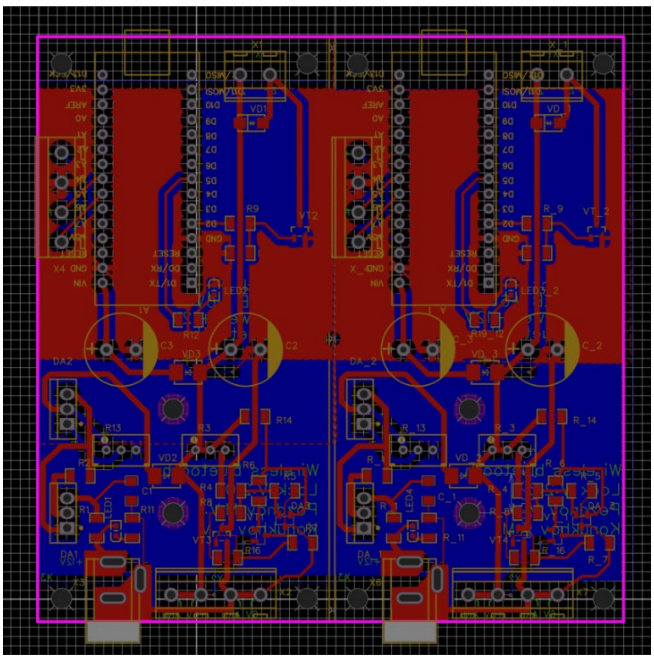

Рисунок 23 - PCB файл разрабатываемой платы

Сервис EasyEDA имеет прямую интеграцию с производственной компанией JLCPCB, основным направлением которой является разработка составных элементов для электроники и микроэлектроники. Сервис EasyEDA реализует автоматизированную загрузку необходимых программных файлов партнеру-производителю, затем следует выбрать характеристики и количество физических копий печатной платы. Печатная плата состоит из двуслойного FR-4, представляющего собой стеклотекстолит, укрепленный эпоксидной смолой, обладающий огнестойкими свойствами. Также есть ряд других конфигурируемых параметров на печатной плате. Ряд выбранных характеристик в конфигураторе печатной платы показан на рисунке 24.

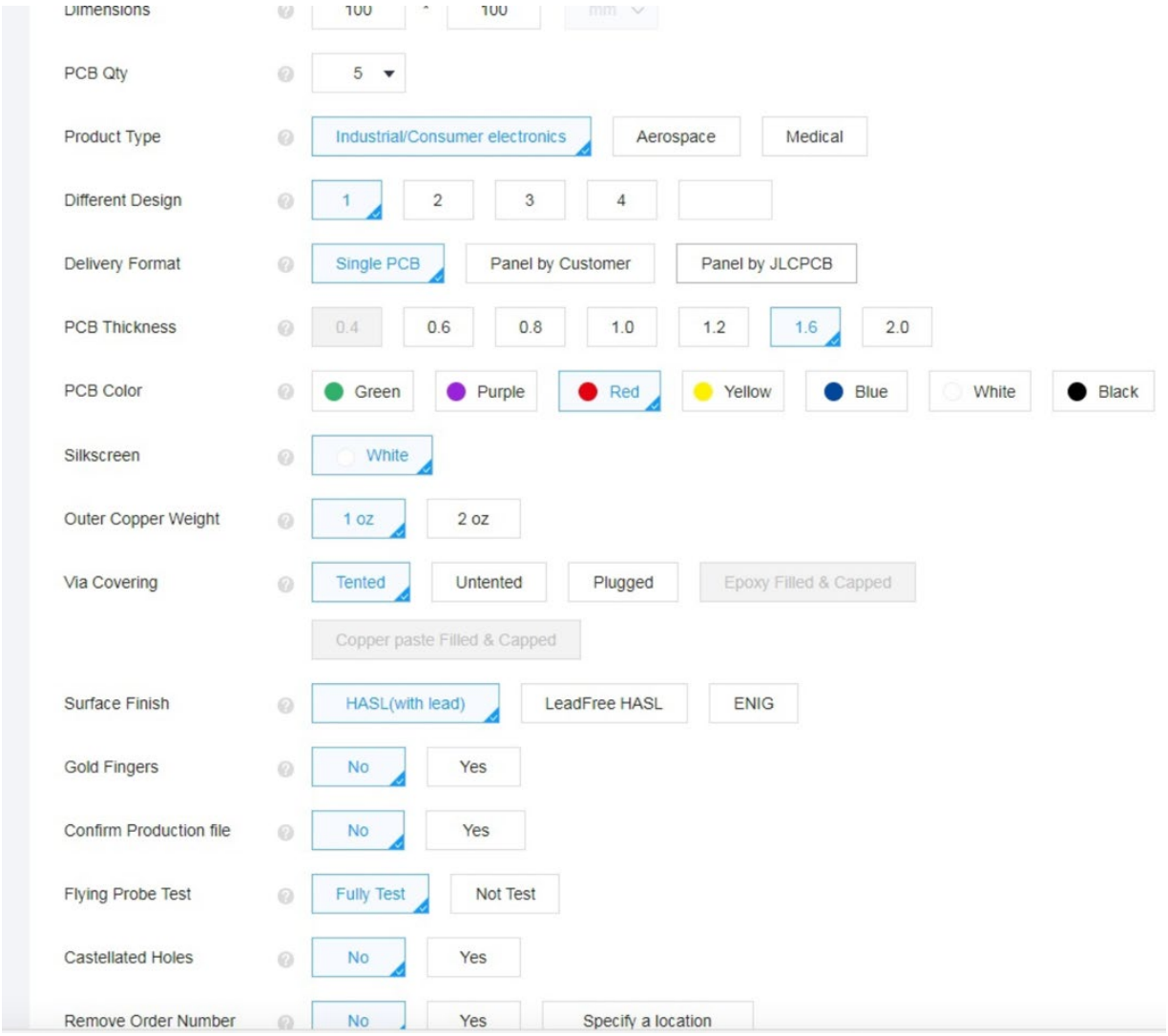

Рисунок 24 – Выбранные характеристики печатной платы

На производство и доставку 5 копий печатных плат затрачено 20 дней, готовых для дальнейшего монтажа микроэлементов на их поверхность. Печатная плата показана на рисунке 25.

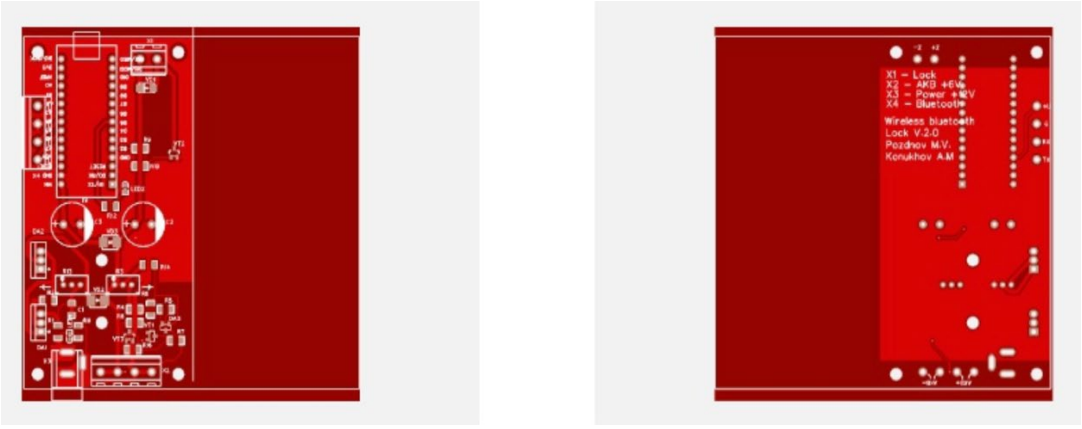

Рисунок 25 – Печатная плата для разрабатываемого устройства

Процесс монтажа микроэлементов на печатную плату включает в себя процесс пайки ножек микроэлементов на металлические пластины, расположенные на печатной плате в соответствии с размещением элементов в программном файле Gerber. После размещения и монтажа элементов микроэлектроники получен рабочий экспериментальный образец. Печатная плата после монтажа элементов приведена на рисунке 26.

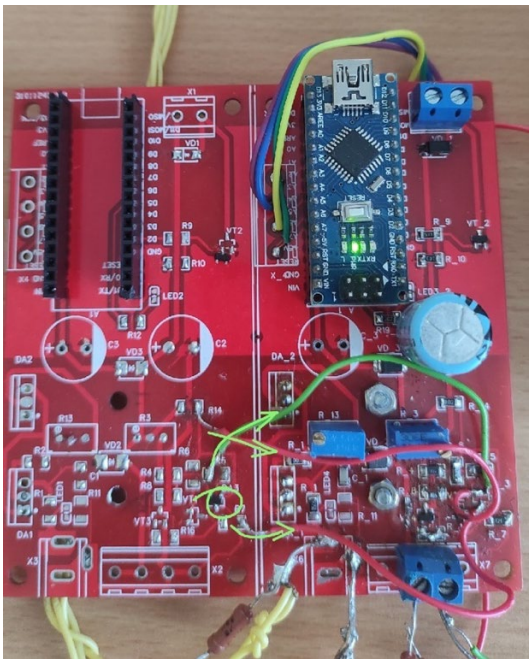

Рисунок 26 – Готовая микросхема

4.2 Выбор корпуса для устройства

При выборе корпуса устройства следует руководствоваться следующими правилами:

- необходимо выбрать корпус, достаточно удовлетворяющий габаритам устройства и сопутствующих элементов схемы;

- выбор материала корпуса зависит от требований к прочности, защите от внешних факторов, электромагнитных помех и других факторов.;

- необходимо убедиться в простоте доступа к элементам устройства, не прибегая к возможному демонтажу всей конструкции;

- корпус должен соответствовать требованиям по удобству использования и удобству размещения на рабочем месте;

- корпус должен соответствовать требованиям по дизайну и эстетике, если это важно для конечного продукта;

- необходимо выбрать корпус, который соответствует бюджету проекта, не ущербом для качества и требований к устройству;

Исходя из этого выбран корпус Gainta G1037, обладающий габаритами 189x113x66,6 мм и позволяющий вместить в себя все элементы схемы с размещением их в участках для возможности последующего монтажа каждого элемнета. Размещение элементов аппаратного комплекса в корпусе показано на рисунке 27.

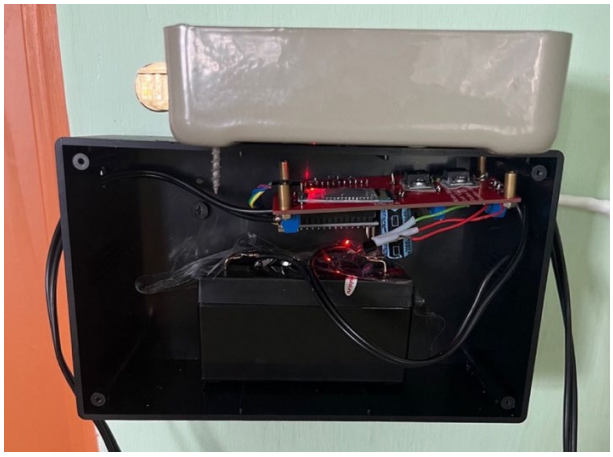

Рисунок 27 – Размещение элементов в корпусе

Конструктивные элементы схемы расположены согласно рекомендациям по конструированию аппаратного комплекса, включающие изолирование отдельных проводящих элементов схемы и расположение строго вертикально элементов, уязвимых к смене своего угла положения.

5 Экспериментальные исследования программно-аппаратного комплекса

Произведен монтаж готового экземпляра разработанного аппаратного комплекса на входную дверь для последующего выявления возможных недостатков в конструкции. Установленный аппаратный комплекс показан на рисунке 28.

Рисунок 28 – Введенный в эксплуатацию аппаратный комплекс

Экспериментальные исследования разрабатываемого программно-аппаратного комплекса являются одним из самых важных этапов разработки, так как на этом этапе выявляются возможные уязвимости, связанные со стабильностью работы и безопасности системы в целом. Исходя из этого приведены следующие этапы тестирования комплекса:

- экспериментальные исследования работы устройства. Этап включает в себя проверку, насколько стабильно устройство работает в собранном виде, питающийся от аккумуляторной батареи АКБ и питания от сети;

- экспериментальные исследования работы программы. Этап включает в себя проверку программной части комплекса на возможные уязвимости

безопасности системы, так как комплекс ориентирован на безопасность физической собственности;

5.1 Экспериментальные исследования работы устройства

После полной сборки устройства с распределением структурных элементов в корпусе и подключения цепи к нагрузке выявлены следующие замечания:

- малые потери тока в проводнике при питании цепи от блока питания 12 В, подключенным к сетевому источнику тока, приводящие к нестабильной работе отладочной платы Arduino Nano с подключенным к нему Bluetooth модулем HC-05. Нестабильность работы проявляется в периодических перезагрузках Bluetooth модуля HC-05, что сказывается на стабильности работы с комплексом в целом. Также отмечен недостаточный импульс срабатывания пружинного механизма открытия замка, что может привести к возможным проблемам с доступом в помещение. Проблема решена включением в цепь питания блока АКБ питания, как основного источника питания цепи;

- моменты перехода аппаратного комплекса в режим сна. Под режимом сна понимается отсутствие каких-либо реакций на приходящие через Bluetooth модуль команды. Проблема решена дополнением в коде в виде периодической перезагрузки отладочной платы с интервалом 1 час;

- нестабильная работа отладочной платы из-за отходящего контакта разъема питания. Нестабильная работа выражается в периодических перезагрузках отладочной платы Arduino Nano при физическом воздействии на разъем питания схемы от сети. Проблема решена повторной пайкой контактов разъема питания схемы;

5.2 Экспериментальные исследования работы программы

Экспериментальные исследования работы программы включают в себя два этапа:

- экспериментальные исследования работы программы для Arduino – Этап выявления возможных угроз безопасности устройства и его стабильности работы с последующей доработкой программного кода;

- экспериментальные исследования работы программы для Android – Этап выявления возможности поведения спам-атак на программно-аппаратный комплекс, управляемый системой Arduino с последующей доработкой программного кода;

5.2.1 Экспериментальные исследования Arduino

В ходе тестирования программного обеспечения для Arduino выявлены следующие недочеты, связанные с безопасностью системы:

- малая длина мастер-пароля, необходимого для смены основного пароля, передаваемого для открытия замка. Изначальная длина мастер-пароля составляла 4 символа, было решено увеличить длину мастер-пароля до 10 символов для увеличения уровня безопасности и противодействия возможным атакам методом подбора возможных паролей;

- задержка после получения неверного пароля. Изначально при получении данных, не удовлетворяющих условиям вхождения в одно из условий выполнения работы программы не было предусмотрено дополнительных действий, что приводило к дальнейшим итерациям цикла, позволяющим злоумышленникам получить доступ к помещению методом подбора пароля. Таким образом при получении неверных значения пароля или мастер-пароля была введена задержка, равная 1 минутам, что увеличивает уровень безопасности от внешних беспроводных атак;

5.2.2 Экспериментальные исследования Android

В ходе тестирования программного обеспечения для устройств на операционной системы Android, необходимых для управления комплексом подл управления логической программы на языке Arduino выявлен следующий недочет, связанные с логикой проверки передаваемого мастер-пароля – Изначально производилась двуступенчатая проверка мастер-пароля, включающая посимвольную проверку в программе Android, затем после передачи в программе Arduino. Это является хорошей практикой безопасной передачи данных, но данная проверка обязует сохранять значение мастер-пароля в переменные обоих частей программного кода, что может вызвать рассинхронизацию значений мастер-пароля при возможных нарушениях в логике работы программы. Принято решение упразднить посимвольную проверку мастер-пароля на стороне логики программы Android и оставить проверку длины мастер пароля, длина которого 10 символов.

Таким образом нет необходимости в создании переменной, хранящей мастер пароль и его проверка выполняется только на стороне отладочной платы Arduino, что исключает возможную рассинхронизацию значений мастер-пароля между двумя сторонами комплекса.

Заключение

В ходе выполнения выпускной квалификационной работы велась разработка программно-аппаратный комплекса по гибкому подходу к управлению проектом Agile, который позволяет вносить изменения в существующий план проекта на протяжении всей разработки и не исключающий смены существующего плана работ и дат их выполенния.

Актуальный разработанный комплекс представляет собой совокупность компонентов, включающих мобильное приложение для устройств на операционной системе Android с встроенным Bluetooth модулем. Приложение функционирует в режиме master и предназначено для управления электромеханическим замком Falcon Eye-2369i. С помощью этого приложения пользователь может осуществлять контроль доступа и управление замком с помощью своего смартфона или планшета. Одной из ключевых особенностей разработки является возможность автономной работы аппаратного комплекса без внешнего контроля. Для этого было разработано и реализовано схемотехническое решение, которое позволяет комплексу функционировать независимо от внешних устройств или систем. Такая автономность обеспечивает надежность и устойчивость работы замка в случае возможных сбоев или отключения связи с внешними устройствами. Разработанный комплекс представляет собой современное решение, обеспечивающее удобство использования, безопасность и возможность автономной работы.

Рабочее аппаратное решение установлено на входной двери 410 аудитории учебного корпуса “Э” для внедрения дополнительного метода доступа к помещению и введено в рабочий режим, позволяя получить доступ к помещению посредством беспроводной технологии Bluetooth или физического ключа в случае возможных неисправностей, так как комплекс находится в активной фазе тестирования. Комплекс активно тестируется на наличие возможных уязвимостей и недочетов.

Список используемых источников

1. Армбристер С., Унмейк К. Микроконтроллеры: архитектура, программирование, интерфейсы. - М.: ДМК Пресс, 2019. - 352 с.
2. Айардуино – купить микроконтроллеры Arduino. : [Электронный ресурс]. URL: https://wiki.iarduino.ru/page/at-komandy-bluetooth-hc-05/AT/ / Команды Bluetooth HC-05 (Дата обращения: 20.05.2023)
3. Все инструменты. Все для дома и дачи. : [Электронный ресурс]. URL: https://www.vseinstrumenti.ru/category/elektromehanicheskie-zamki-169269/ / Электромеханические замки (Дата обращения: 05.06.2023)
4. Джереми Блум. Arduino для самых маленьких. - М.: ДМК Пресс, 2018. 240 с.
5. Джонс Дэвид, Введение в электронику: Основы и принципы. - М.: Издательство "Вильямс", 2018. 400 с.
6. Мазиди М. Р. Архитектура микроконтроллеров PIC. - Санкт-Петербург: Питер, 2017. - 448 с.
7. Маккрейди К., Маккрейди Д. Программирование микроконтроллеров AVR на языке С. - М.: ДМК Пресс, 2018. - 416 с.
8. Марсиканский А. Андроид. Программирование для профессионалов. - СПб.: БХВ-Петербург, 2017. 704 с.
9. Метанит – сайт о программировании. : [Электронный ресурс]. URL: https://metanit.com/java/android/ / Программирование под Андроид на Java (Дата обращения: 20.05.2023)
10. Митропольский Ю. И. Мир электроники. Электроника – практический курс. - М.: Техносфера, 2016. 146 с.
11. Ратансетьянакорн Т., Маккинтайр И. Android. Программирование для профессионалов. - М.: Вильямс, 2019. 960 с.
12. Смит Джон, Курс электроники: Введение в теорию и практику. - М.: Издательство "БХВ-Петербург", 2019. 320 с.

13. СМП. : [Электронный ресурс]. URL: https://www.smd.ru/katalog/poluprovodnikovye_diody_SMD/diody/diody_shottky/sr24/ / Диод Шоттки SR24 (Дата обращения: 05.06.2023)

14. Соколов Александр, Электроника и микроэлектроника: Теория и практика. - М.: Издательство "Бином", 2019. 384 с.

15. Хоровиц Яков, Микроэлектроника: Устройство и применение интегральных схем. - М.: Издательство "Лань", 2020. 480 с.

16. Чен Майкл, Микроэлектроника: Технология, проектирование и применение. - М.: Издательство "Диалектика", 2017. 520 с.

17. Android developers. : [Электронный ресурс]. URL: https://developer.android.com/guide/topics/connectivity/bluetooth/connect-bluetooth-devices / / Connect Bluetooth devices (Дата обращения: 20.05.2023)

18. EasyEDA. Online PCB Design Tool. : [Электронный ресурс]. URL: https://easyeda.com/ / EasyEDA PCB tools (Дата обращения: 05.06.2023)

19. Linuxhint. : [Электронный ресурс]. URL: https://linuxhint.com/top-microcontrollers-2022/ / Top 5 Microcontrollers you should get to know in 2022 (Дата обращения: 20.05.2023)

20. Mazidi, Muhammad Ali, McKinlay, Rolin D., Causey, Danny. PIC Microcontroller and Embedded Systems: Using Assembly and C for PIC18. - Prentice Hall, 2013. 832 с.

21. Monk, Simon. Programming Arduino: Getting Started with Sketches. - McGraw-Hill Education, 2016. 192 с.

22. ONSEMI. : [Электронный ресурс]. URL: https://www.onsemi.com/pdf/datasheet/tl431-d.pdf / Programmable Precision References (Дата обращения: 05.06.2023)

23. ONSEMI. : [Электронный ресурс]. URL: https://www.onsemi.com/pdf/datasheet/lm317t-d.pdf / LM317-d (Дата обращения: 05.06.2023)

24. PCMAG. : [Электронный ресурс]. URL: https://www.pcmag.com/picks/the-best-smart-locks / Best smart locks for 2023 (Дата обращения 05.06.2023)

Приложение А

Блок схема программы Arduino.

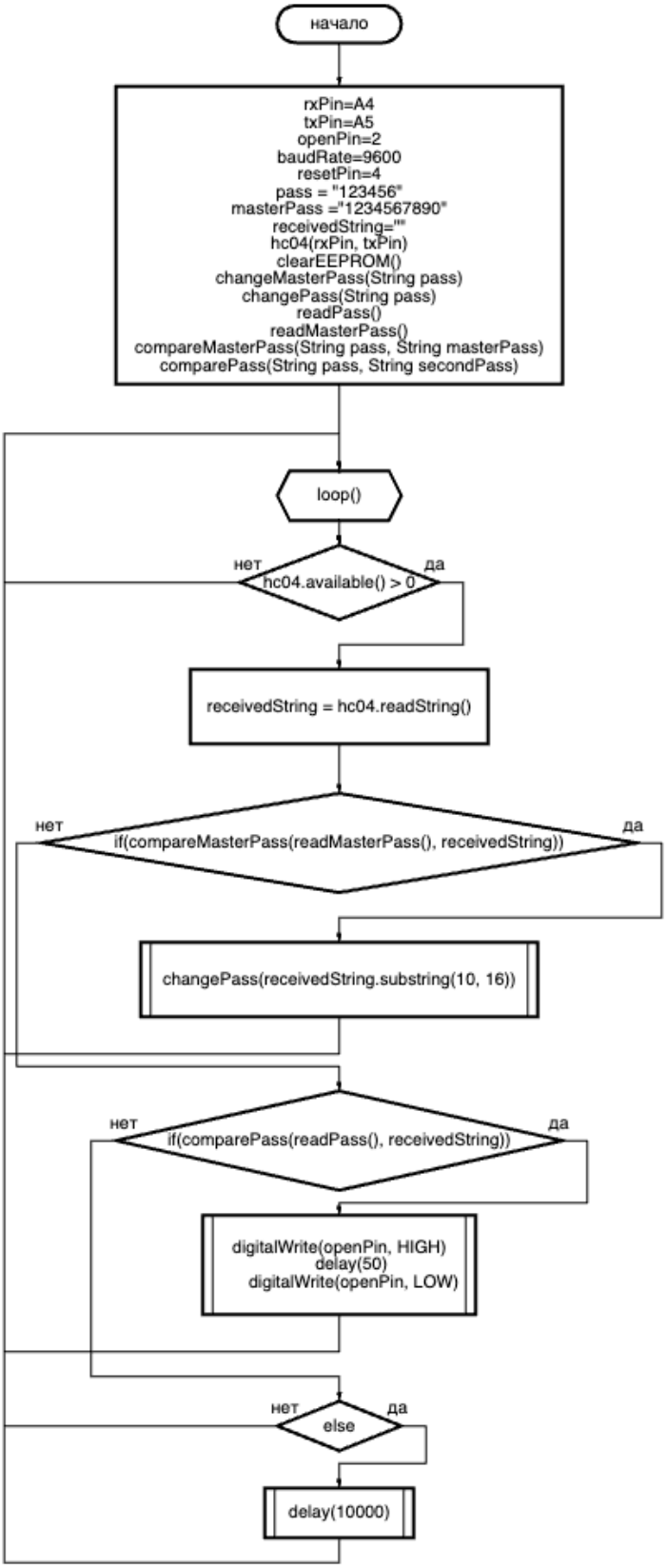

Рисунок А.1 – Блок схема программы для Arduino Nano

Приложение Б

Программный код для Arduino Nano.

```
#include <EEPROM.h>
#include <SoftwareSerial.h>
//порты передачи данных через Bluetooth модуль
const int rxPin = A4;
const int txPin = A5;
const int baudRate = 9600;
const int resetPin = 4;
//логические переменные (пароль, мастер пароль, полученная
строка и пин подачи импульса на замок)
//String pass = "123456";
//String masterPass ="1234567890";
String receivedString = "";
int openPin = 2;
SoftwareSerial hc04(rxPin, txPin);
void setup() {
  //подача сигнала HIGH в течение 1 секунды на пин перезагрузки
Bleutooth адаптера
  digitalWrite(resetPin, HIGH);
  delay(1000);
  digitalWrite(resetPin, LOW);
  hc04.begin(baudRate);
  Serial.begin(baudRate);
  //clearEEPROM();
  //changePass(pass);
  Serial.println("Password written to EEPROM: " + readPass());
  //changeMasterPass(masterPass);
  Serial.println("Master password written to EEPROM: " +
readMasterPass());
  pinMode(openPin, OUTPUT);
  Serial.println("Ready to receive string values from HC-
04...");
}

void loop() {
  if (hc04.available() > 0) {
    receivedString = hc04.readString();
    Serial.print("Received string: ");
    Serial.println(receivedString);
    /*Условие сверки мастер пароля из полученного пароля,
    в случае удачи меняется значение пароля в EEPROM*/
    if(compareMasterPass(readMasterPass(), receivedString)) {
      changePass(receivedString.substring(10, 16));
      Serial.println("Password changed: " + readPass());
    }
    /*Условие сверки пароля из полученного пароля,
    в случае успеха открытие замка путем подачи импульса
    на замок длительностью 50 мс.*/
    else if(comparePass(readPass(), receivedString)){
      digitalWrite(openPin, HIGH);
      delay(50);
```

```
      digitalWrite(openPin, LOW);
      Serial.println("Unlocked");
    }
    /*в случае получения неверного пароля ардуино уходит в паузу
на 10 секунд*/
    else {
      Serial.println("Do not even try to to something!");
      delay(10000);
    }
  }
}
//метод полной очистки EEPROM, используется при начале работы
программы
void clearEEPROM() {
  for (int i = 0; i < EEPROM.length(); i++) EEPROM.update(i, 0);
}
//метод смены мастер-пароля в EEPROM, его использование на
данный момент упразднено в связи с логической бессмысленностью
void changeMasterPass(String pass) {
  char charBuf[pass.length() + 1];
  pass.toCharArray(charBuf, pass.length() + 1);
  for (int i = 0; i < 10; i++) {
    EEPROM.write(i, pass[i]);
  }
}
//метод смены пароля в EEPROM, используется при удачной сверке
мастер-пароля из приходящей строки
void changePass(String pass) {
  char charBuf[pass.length() + 1];
  pass.toCharArray(charBuf, pass.length() + 1);
  for (int i = 0; i < 6; i++) {
    EEPROM.write(10 + i, pass[i]);
  }
}
//метод чтения значения пароля из EEPROM, используется при
сверке с приходящим паролем
String readPass() {
  char charBuf[6 + 1];
  for (int i = 0;i < 6; i++) {
    charBuf[i] = EEPROM.read(10 + i);
  }
  charBuf[6] = '\0';
  return String(charBuf);
}
//метод сверки мастер пароля, используется при сверке с
приходящим паролем
String readMasterPass() {
  char charBuf[10 + 1];
  for (int i = 0;i < 10; i++) {
    charBuf[i] = EEPROM.read(i);
  }
```

```
  charBuf[10] = '\0';
  return String(charBuf);
}
//метод сверки мастер-паролей, используется для смены пароля в
EEPROM
bool compareMasterPass(String pass, String masterPass) {
  for (int i = 0; i < 10; i++) {
    if ( pass[i] != masterPass[i]) {
      return false;
    }
  }
  return true;
}
//метод сверки паролей, используется для открытия замка
bool comparePass(String pass, String secondPass) {
  for (int i = 0; i < 6; i++) {
    if ( pass[i] != secondPass[i] ) {
      return false;
    }
  }
  return true;
}
```

Приложение В

Программный код активности MainActivity.

```
package infoaryan.in.hc05_bluetooth;
import androidx.appcompat.app.AppCompatActivity;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
import java.util.ArrayList;
import java.util.Set;

public class MainActivity extends AppCompatActivity {
    private BluetoothAdapter myBluetooth = null;
    private Set<BluetoothDevice> pairedDevices;
    public static String EXTRA_ADDRESS = "device_address";
    ListView devicelist;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Button btnPaired;

        btnPaired = (Button) findViewById(R.id.button);
        devicelist = (ListView) findViewById(R.id.listView);

        myBluetooth = BluetoothAdapter.getDefaultAdapter();
        if ( myBluetooth==null ) {
                Toast.makeText(getApplicationContext(),
"Bluetooth device not available", Toast.LENGTH_LONG).show();
                finish();
            }
        else if ( !myBluetooth.isEnabled() ) {
                Intent turnBTon = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
                startActivityForResult(turnBTon, 1);
            }

        btnPaired.setOnClickListener(new View.OnClickListener()
{

            @Override

            public void onClick(View v) {
                    pairedDevicesList();
```

```
                                    }
                                });

                            }

                    private void pairedDevicesList () {
                pairedDevices = myBluetooth.getBondedDevices();
                        ArrayList list = new ArrayList();

                        if ( pairedDevices.size() > 0 ) {
                    for ( BluetoothDevice bt : pairedDevices ) {
                        list.add(bt.getName().toString() + "\n" +
                    bt.getAddress().toString());
                                    }
                                } else {
                     Toast.makeText(getApplicationContext(), "No
    Paired Bluetooth Devices Found.", Toast.LENGTH_LONG).show();
                                    }

              final ArrayAdapter adapter = new ArrayAdapter(this,
            android.R.layout.simple_list_item_1, list);
                            devicelist.setAdapter(adapter);

       devicelist.setOnItemClickListener(myListClickListener);
                                    }

                  private AdapterView.OnItemClickListener
    myListClickListener = new AdapterView.OnItemClickListener() {
                                    @Override
                public void onItemClick(AdapterView<?> parent, View
                    view, int position, long id) {
                                String info = ((TextView)
                      view).getText().toString();
                    String address = info.substring(info.length()-
                                17);

                       Intent i = new Intent(MainActivity.this,
                          PassControl.class);
                          i.putExtra(EXTRA_ADDRESS, address);
                                    startActivity(i);
                                        }
                                    };
                                }
```