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

Институт энергетики и электротехники
(институт)

Кафедра «Промышленная электроника»

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

Промышленная электроника
направленность (профиль)

БАКАЛАВРСКАЯ РАБОТА

на тему Многофункциональное устройство контроля доступа в помещение на микроконтроллерном управлении

Студент	Н.С. Семыкин	
	(И.О. Фамилия)	(личная подпись)
Руководитель	А.В. Прядилов	
	(И.О. Фамилия)	(личная подпись)
Консультант	О.Н. Брега	
	(И.О. Фамилия)	(личная подпись)

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

«_____»_____20_____г.

Тольятти 2017

Аннотация

Тема данной дипломной работы – «Многофункциональное устройство контроля доступа в помещение на микроконтроллерном управлении», в рамках которой были рассмотрены основные принципы разработки и создания системы, позволяющей контролировать доступ к закрытым помещениям.

Объектом исследования является макет устройства контроля доступа в помещение на микроконтроллерном управлении.

Цель работы заключается в изготовлении действующего макета устройства контроля доступа на микроконтроллерном управлении.

Для того чтобы решить поставленную цель работы, были сформулированы основные задачи:

1. Обзор существующих решений
2. Разработка схемы устройства
3. Выбор необходимых компонентов
4. Задание алгоритма работы
5. Создание программы работы устройства
6. Изготовление макета устройства
7. Отладка программы и макета

Работа состоит из 67 страниц, включающих в себя 36 рисунков, 1 таблицу, 2 приложения. Ключевые термины работы: микроконтроллер, электронный замок, радиочастотная идентификация. Для решения поставленных задач использован 21 источник литературы.

Для создания наглядных схем подключения различных элементов использовались программные решения Autodesk Circuits и Fritzing.

Результатом дипломной работы является действующий макет устройства контроля доступа в помещение на микроконтроллерном управлении.

Областью применения данной работы являются системы контроля доступа в помещение.

Abstract

The topic of the graduation work is “Multifunctional device on microcontroller’s operation for access control in premises.” The work presents the main principles of development and creation of a system allowing control access to closed premises.

The object of the work is a model of the access control device on the microcontroller’s operation for the premises.

The aim of the work is to produce an operating model of the access control device on the microcontroller’s operation.

In order to solve the stated aim of the work, the main tasks were formulated:

1. Review of existing solutions;
2. Development of the device diagram;
3. Selection of the required components;
4. Setting of the operation algorithm;
5. Creation of a program of the device;
6. Device modeling;
7. Debugging of the program and the model.

The work consists of 67 pages, including 36 figures, 1 table, and 2 appendixes. 21 references were used for review.

Keywords: microcontroller, electronic lock, radio frequency identification.

Autodesk Circuits and Fritzing software solutions were used to create visual diagrams for connecting various elements.

The result of the work is the operating model of the access control device on the microcontroller’s operation for the premises.

The scope of this work is the access control system for the premises.

Оглавление

Введение

Электронный замок — специальное электронное устройство, необходимое для того, чтобы предотвратить доступ в закрытое помещение посторонних лиц, или ограничить выход из помещения. Система контроля принимает решение о разрешении на доступ в помещение на основе сигналов от различных устройств: считывателей магнитных карт, штрих-кодов, датчиков контактной памяти, биометрических датчиков, наборной клавиатуры, считывателей магнитных карт, дистанционного управления и других всевозможных датчиков. В большинстве случаев электронный замок является частью сложной системы контроля доступа в помещение. В качестве механизмов, препятствующих в доступе в помещение, используются электромеханические и электромагнитные запорные устройства. Использование микроконтроллера позволит упростить основные манипуляции с системой контроля доступа и самим электронным замком.

В рамках данной работы предполагается разработать, создать и отладить небольшую действующую модель устройства контроля доступа на микроконтроллерном управлении. Такая модель позволит практически продемонстрировать работу электронного замка и сопутствующих ему элементов.

1 Состояние вопроса

1.1Формулирование актуальности, цели и задач проекта

В настоящее время электронные системы контроля доступа в помещение получают все большее распространение в жизни огромного количества людей. Электронные замки используются там, где запрещено находиться посторонним, например, в складских помещениях предприятий, в подсобных помещениях в магазинах и крупных супермаркетах. Несмотря на это, электронные системы контроля доступа являются достаточно дорогим оборудование для обычного человека. Но что если существуют решения, которые бы позволили широкому кругу населения использовать электронные замки в различных целях.

Исходя из вышесказанного, разработка многофункционального устройства контроля доступа в помещение на микроконтроллерном управлении является актуальной задачей.

Целью данной работы является: изготовление действующего макета устройства контроля доступа на микроконтроллерном управлении.

Для достижения этой цели поставлены и выполнены следующие **задачи**:

1. Обзор существующих решений
2. Разработка схемы устройства
3. Выбор необходимых компонентов
4. Задание алгоритма работы
5. Создание программы работы устройства
6. Изготовление макета устройства
7. Отладка программы и макета

1.2Анализ исходных данных и известных решений

На сегодняшний день присутствует несколько вариаций замков, которым не требуется ключ в привычном для нас понимании для того, чтобы открыть или закрыть дверь. Есть несколько типов замков – с магнитным ключом, представленным в виде брелка или карточки, похожей на банковскую, или с кодовой комбинацией, которые устанавливаются прямо на дверь, речь о которых пойдет ниже.

Современные кодовые замки можно классифицировать по нескольким признакам: по типу установки, способу управления механизмом запирания двери, возможности изменять кодовую комбинацию.

По типу кодовые замки подразделяются на навесные и врезные замки. Навесные замки используется в основном для сараев, гаражей, складских помещений, и в других подобных случаях, но для жилых домов и квартир лучше использовать врезные кодовые замки (рисунок 1.1), которые надежней навесных из-за того, что рабочие механизмы такого замка располагаются внутри дверного полотна, защищая его от возможности разбора и взлома извне. Навесные замки можно использовать в качестве дополнительно защиты.

Рисунок 1.1 – Замок врезного типа

По принципу управления запирающим механизмом замки можно разделить на механические, пример показан на рисунке 1.2, и электронные. В настоящее время механические замки начинают терять свою привлекательность, хотя нельзя с уверенностью говорить, что их уже начинают сильно вытеснять электронные модели, ведь время механики в нашей стране будет еще долго длиться, и трудно точно сказать, когда электроника начнет доминировать сфере замков. Несмотря на это, механические замки имеют свои недостатки, например, они быстрее выходят из строя, вследствие постоянного их использования, также серьезной проблемой является то, что к данным замкам достаточно легко подобрать нужную комбинацию для открытия, по сравнению с другими видами.

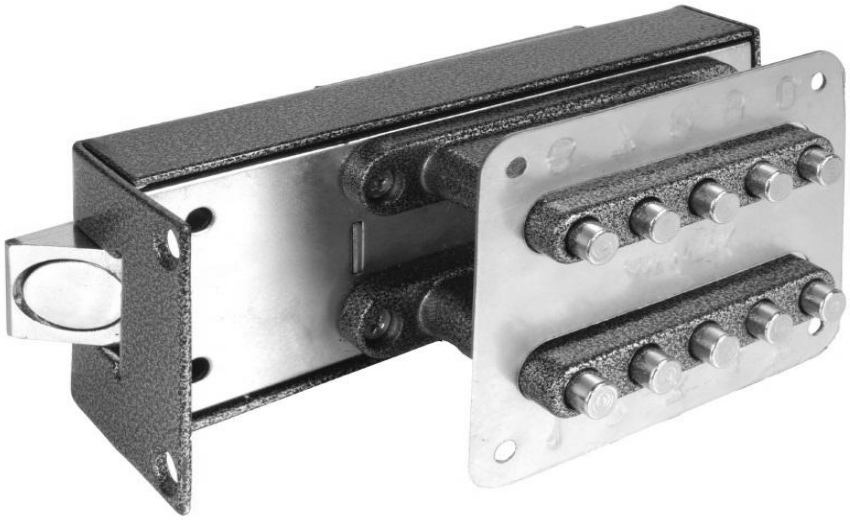

Рисунок 1.2 – Механический кодовый замок

Механические замки имеют кнопочный кодовый механизм или поворотный. На практике кнопочные замки показывают себя хуже всего. Постоянные нажатия на кнопки стирают их верхний слой, и они начинают западать, что дает возможность любому человеку легко подобрать верную комбинацию. Поворотный механизм, в сравнении с кнопочным имеет неоспоримое преимущество – определить, как и сколько раз и в какую сторону нужно повернуть ручку не получиться, несмотря на то, насколько затерты надписи или как часто этим замком пользовались. Существуют

также электронные кодовые замки, которые заслуживают отдельного внимания.

Электронные кодовые замки имеют одно существенное преимущество перед механическими – блок управления может находиться в удалении от основного устройства. Его можно установить в любом месте – это основа принципа замка – невидимки. Когда нет возможности видеть то, с чем придется иметь дело, становится проблематично вскрыть такое устройство. Кроме того, современный кодовый замок – это высокотехнологичное устройство, управляемое при помощи микропроцессора, а это миллионы различных комбинаций. Все электронные кодовые замки можно разделить на следующие категории:

1. Кнопочные кодовые замки с электронным управлением. Они получили наибольшее распространение, но в то же время это уязвимая группа запирающих устройств. Причина, как и в случае механических замков, та же – стирающиеся и западающие кнопки. В основном их устанавливают там, где не требуется хранить особо ценные предметы. Их используют для предотвращения проникновения в помещение, например в подъезды, складские и подсобные помещения в небольших предприятиях или магазинах. В большинстве случаев пульт управления и сам замок изготавливаются одним блоком, но имеются модели с раздельным оснащением. С развитием технологий кнопочные кодовые замки получили свое логическое продолжение в виде замков с сенсорной панелью. Преимущества перед кнопками здесь очевидны, но есть серьезная проблема с тем, что на марких сенсорных экранах при нажатии остаются отпечатки пальцев, которые становятся заметны под определенным углом обзора. Для избегания взлома комбинации по отпечаткам, требуются различные ухищрения, на которые идут производители замков с сенсорной панелью. Одним из способов сокрытия комбинации является ввод дополнительных ненужных символов вначале или в конце основного кода.

2. Кодовые замки с магнитным носителем комбинации. Это очень надежные замки, вскрыть которые можно, если заполучить сам магнитный носитель. В этой роли могут выступать: пластиковая карточка, небольшой брелок, пульт дистанционного управления, передающий код приемнику радиосигналом или сигналом в инфракрасном спектре, легко перехватываемый и декодируемый.
3. Комбинированные замки (рисунок 1.3). Данный вид замков наиболее распространен на сегодняшний день. Попасть внутрь помещения можно только при использовании последовательно нескольких различных открывающих устройств, например, кодовой комбинации и пластиковой карты, что делает комбинированные замки очень надежной системой. Их особенностью является то, что разблокировать дверь невозможность только при помощи одного ключа, потребуется пройти все последовательные действия, необходимые для открытия, а также то, что ключи от других производителей будут игнорироваться при попытке их использования.

Рисунок 1.3 – Электронный кодовый замок с сенсорной панелью ввода производства компании Samsung

Устройство электронного замка состоит четырех частей:

1. Запирающий механизм. Чтобы он открылся или закрылся, на него подают короткий электрический импульс. Когда кодовая комбинация совпадет с заданной, замок откроется.
2. Считыватель кода или пульт управления. Это считывающее устройство – электроники управления в нем нет. С помощью этого устройства вводится код и направляется в блок управления.
3. Блок управления. Он распоряжается вопросом, подавать или не подавать импульс для срабатывания запирающего механизма. В основном замки закрываются автоматически как, в принципе, и любой другой механический захлопывающийся замок.
4. Источник бесперебойного питания (ИБП). Наличие ИБП в таких замках обусловлено тем, что во время отключения электроэнергии попасть в дом будет затруднительно, ведь при отсутствии электричества замки автоматически блокируются. С его помощью кодовый электронный замок проработает автономно несколько суток. Обычно ИБП располагается в тайном месте, там же, где и блок управления.

В сети Интернет также можно найти различные схемотехнические решения для реализации электронных кодовых замков, например, как на рисунке 1.4. Данная схема представляет собой устройство, состоящее из: двух КМОП микросхем 561ЛА7 и одной 561ЛЕ5, транзисторы VT1-VT3 – КТ361, или КТ3107, транзистор VT4 –КТ315, транзистор VT5 – КТ815. Вторичная обмотка трансформатора Т1 рассчитана на 12 вольт. Трансформатор Т1 выбирается достаточной мощности, обеспечивающей срабатывание исполнительного устройства, диоды VD3-VD7 выбираются выпрямительные FR107, обеспечивающие достаточный ток нагрузки исполнительного устройства. Диоды VD8-VD20 – маломощные импульсные КД521. В качестве аккумуляторной батареи используется малогабаритная

щелочная батарея, используемая в источниках бесперебойного питания. Набор кода осуществляется при помощи кнопочной панели SA1.

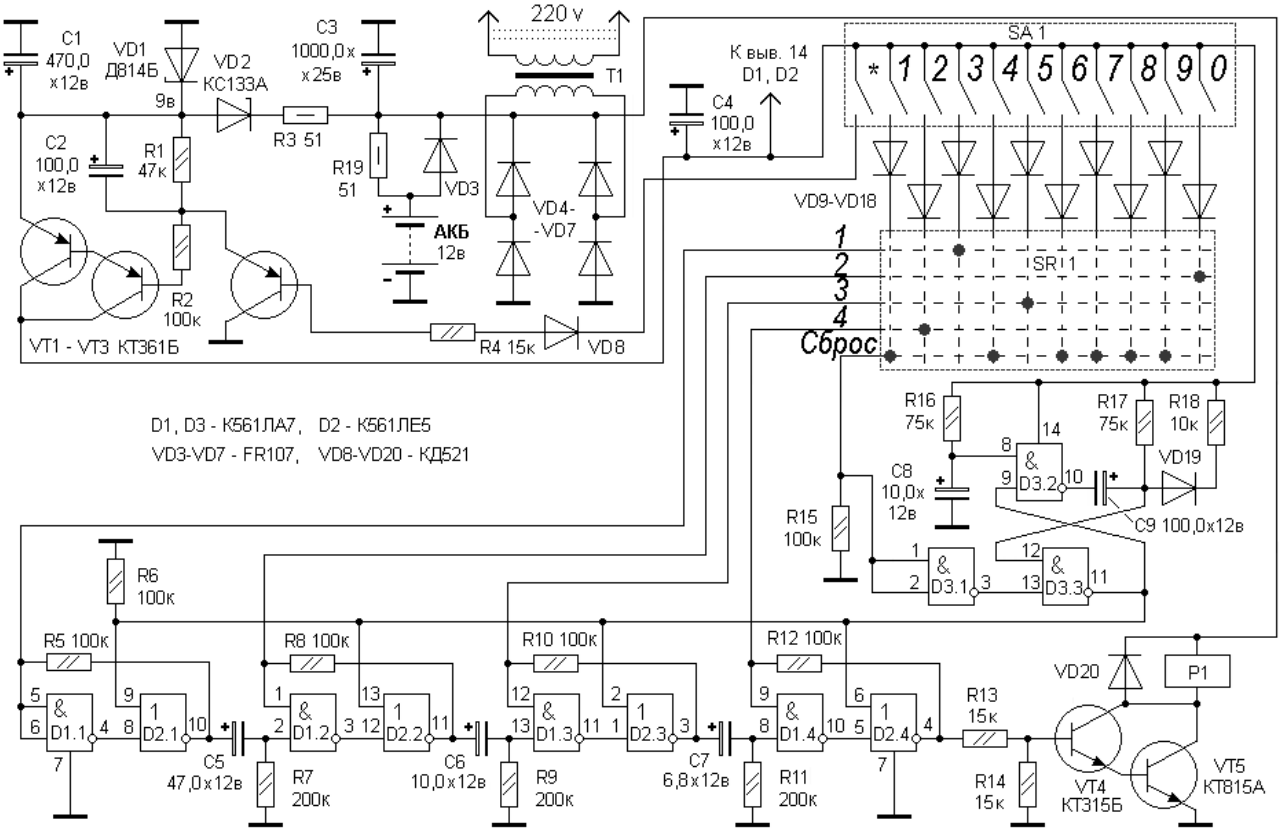

Рисунок 1.4 – Схема электрическая принципиальная электронного кодового замка

Заявленными преимуществами данной схемы по сравнению с другими схемотехническими решениями, доступными в сети Интернет являются: высокая защищенность от взлома обманными кнопками, при нажатии на которые замок блокируется на некоторое время, кнопочная панель располагается отдельно от основного устройства, благодаря чему взлом при помощи измерительной аппаратуры становится невозможен, питание от аккумулятора на 12 вольт, или от сети переменного тока 220 вольт.

Несмотря на это, представленная выше схема кодового замка имеет свои недостатки как электронный замок: чувствительность к электромагнитным помехам, тяжелым погодным условиям, требование к бесперебойной подаче питания. Но главным фактором является то, что для

создания подобных схемотехнических решений требуются глубокие познания в схемотехнике и в микросхемной логике, нужен точный подбор элементов для правильной работы, поэтому из-за всего вышеперечисленного достаточно тяжело рассматривать этот тип замка в качестве учебного ознакомления. Поэтому в качестве основы для этой работы был выбран электронный замок на микроконтроллерном управлении.

Для того чтобы реализовать устройство контроля доступа на микроконтроллерном управлении, в качестве исходных данных были выбраны способы управления при помощи кодовой комбинации, реализуемой при помощи кнопок, а также радиочастотная идентификация, реализуемая с помощью специально разработанных для работы с микроконтроллерами модулями радиочастотной идентификации. Рассмотрим более подробно основные принципы организации радиочастотной идентификации.

RFID – радиочастотная идентификация является фундаментальной и недорогой технологией, осуществляющей беспроводную передачу данных. Раньше эта технология не так часто использовалась в индустрии в связи с отсутствием стандартизации в производственных компаниях. RFID – технологии эффективней и надежней, по сравнению с другими. С RFID – технологией беспроводная автоматическая идентификация принимает очень специфичный вид: объект, местоположение, или индивид маркируются уникальным идентификационный кодом, содержащимся в RFID - метке, которая каким-то образом прикреплена или вдавлена в объект. Радиочастотная идентификация представляет собой не отдельный продукт, а полноценную систему. Типичная RFID - система включает в себя три базовых элемента: RFID-метку (транспондер), считыватель (трансивер, запросчик) и серверную программу (базу данных), которая запрашивает поддержку компьютерной сети. Программное обеспечение используется для управления, контроля, оперирования, обработки, ведения учета различных пользователей. Цифровая система блокировки дверей осуществляется и

управляется при помощи RFID – считывателя, который проводит проверку и аутентификацию пользователя и автоматически открывает дверь. Он также сохраняет данные о регистрации пользователя. Очень важно провести аутентификацию пользователя до открытия охраняемого помещения и радиочастотная идентификация позволяет сделать это. Система позволяет пользователю регистрироваться в быстрых, безопасных удобных условиях. Система блокировки дверей открывает их только тогда, когда пользователь поместит свою метку на считыватель, а данные о пользователе сравнятся с теми, что хранятся в базе данных. Радио частотная идентификация контролирует открытие и закрытие двери. В зависимости от источника электрической энергии, метки RFID классифицируются как активные или пассивные. Активные метки используют батарею для питания и передачи информации метки по запросу считыватель. Однако эти теги очень дороги и редко используются. С другой стороны, пассивные метки получают энергию от считывателя для питания их схемы. Эти метки очень рентабельны и, следовательно, большинство приложений используют их. Главными преимуществами пассивной технологии являются низкая стоимость и небольшие габариты. Пассивная метка RFID передает информацию считывателю, когда она попадает в электромагнитное поле, генерируемое считывателем. Явление основано на законе электромагнитной индукции Фарадея. Ток, протекающий через катушку считывателя, создает магнитное поле, которое соединяется с катушкой транспондера, тем самым создавая ток в катушке транспондера. Затем катушка транспондера изменяет этот ток, изменяя нагрузку на свою антенну. Это изменение фактически является модулированным сигналом, который принимается катушкой считывателя посредством взаимной индукции между катушками. Катушка считывателя декодирует этот сигнал и передает его на компьютер для дальнейшей обработки. Дополнительное подсоединение антенны позволяет уменьшить размеры устройства.

Существует несколько типов классификации меток. Широко известная и распространенная разделяет метки на чиповые (рисунок 1.5) и бесчиповые (безмикросхемные), изображенные на рисунке 1.6.

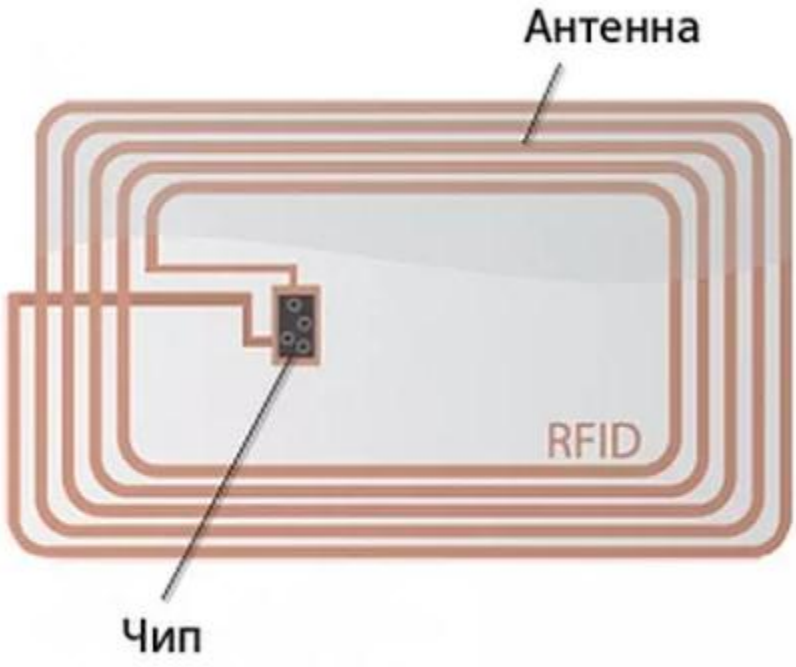

Рисунок 1.5 – Чиповая RFID-метка

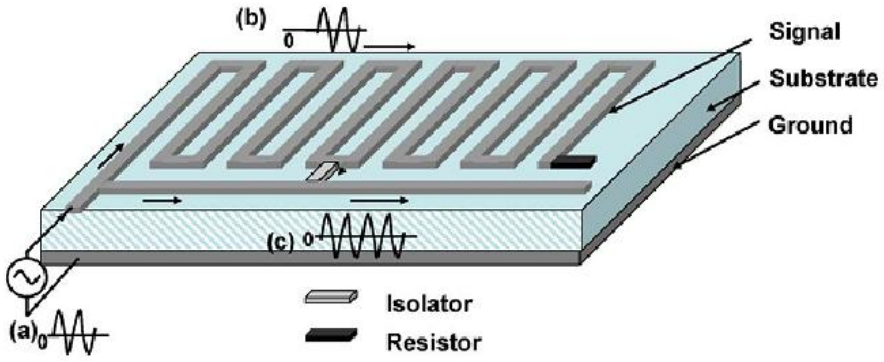

Рисунок 1.6 – Бесчиповая RFID-метка

Чиповые метки содержат интегральную микросхему – чип, а бесчиповые – нет. Вторая классификация разделяет типы меток на пассивные, полуактивные и активные. В состав пассивных меток не входит встроенный источник питания и активный передатчик; полуактивные метки содержат элемент питания, но не имеют активного передатчика; активные метки содержат оба элемента. В еще одной классификации метки

подразделяются на только считываемые (read only) и считываемые/записывающие (read/write). Только считываемые метки получают свой идентификационный код на производстве. Память данного типа меток или только читаемая (ROM) или однократно программируемая и многократно читаемая (WORM), т.е. в ходе работы с WORM памятью, идентификационный код данного типа меток можно перезаписать один раз. Считываемые/записывающие метки могут многократно перепрограммироваться в процессе эксплуатации, и на них можно записать дополнительную информацию, а не только серийный номер или код.

По рабочей частоте метки делятся на метки низкочастотного диапазона LF (125—134 кГц), высокочастотного диапазона HF (13,56 МГц), и сверхвысокочастотного диапазона UHF (860—960 МГц).

2 Проектный раздел

2.1 Разработка электрической схемы и выбор необходимых компонентов

В качестве основы, которая будет контролировать всю работу устройства, был выбран микроконтроллер серии Arduino UNO R3, лицевая часть которого представлена на рисунке 2.1.

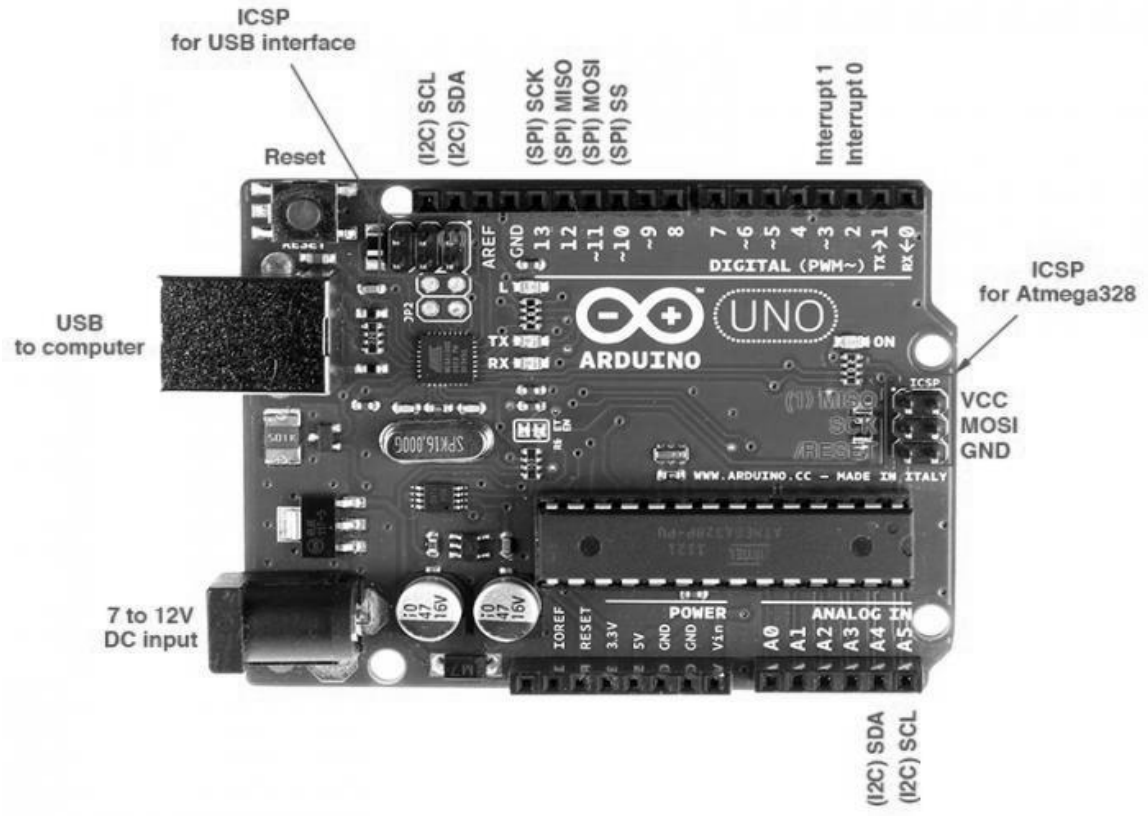

Рисунок 2.1 – Лицевая сторона Arduino Uno R3

Arduino Uno R3 - это устройство на основе микроконтроллера ATmega328. В его состав входит: 14 цифровых входов/выходов под номерами 0-13 (из них 6 могут использоваться в качестве ШИМ-выходов, на плате обозначены со знаком “~”), 6 аналоговых входов A0-A5, кварцевый резонатор на 16 МГц, разъем USB, разъем питания, разъем для внутрисхемного программирования (ICSP) и кнопка сброса. Для начала работы нужно подать питание от AC/DC-адаптера или батарейки, или через

USB-кабель от персонального компьютера. Порт AREF задает опорное напряжение аналоговых входов. Порт IOREF позволяет платам расширения подстраиваться под рабочее напряжение Arduino. Он необходим для совместимости плат расширения как с 5 вольт (В) Arduino на базе микроконтроллеров AVR, так и с 3.3В платами Arduino Due.

Основные характеристики Arduino Uno R3 представлены ниже.

Рабочее напряжение питания 5В, рекомендуемое напряжение питания от 7 до 12В, предельное напряжение питания в диапазоне 6-20В. 14 цифровых входа/выхода, 6 аналоговых входов, максимальный ток одного вывода равен 40 мА, максимальный ток вывода 3.3V равен 50 мА. У микроконтроллера присутствует Flash-память на 32 килобайт (КБ) (ATmega328), используемая при создании программ, из которых 0.5 КБ используются загрузчиком, а также электрически стираемая энергонезависимая перезаписываемая память EEPROM на 1 КБ. Тактовая частота кварцевого резонатора 16 МГц.

Для реализации радиочастотной идентификации был выбран RFID-модуль RC522. Его характеристики представлены ниже.

Напряжение питания 3,3 В, потребляемый ток не более 30 мА, рабочая полоса частот 13,55–13,57 МГц, считывается на расстоянии 0–25 мм, физический размер считывателя 40 х 60 мм, рабочая температура от 20 до 80С°.

Сопровождаемые карты: классы S50, S70, Ultralight, Pro, DESFire; типы Mifare S50, Mifare S70, Mifare UltraLight, Mifare Pro, Mifare DESfire. Скорость передачи информации 106, 212, 424, 848 кбит/с. Шифрование Security Features Mifare classic™(термин Mifare может только компания NXP Semiconductors, а также компании, имеющие лицензию от NXP на производство чипов). Метки MiFare Classic работают на высокочастотных радиоволнах, в частности на частоте 13,56 МГц. Это та же частота, на которой работают устройства с поддержкой Near Field Communication (NFC). В RFID-метках отсутствует микропроцессор и защищенный элемент,

способный к аутентификации. RFID-метки MiFare были введены NXP Semiconductors в 1995 году, и с тех пор было продано более миллиарда меток во всем мире. Действуя в качестве систем контроля доступа и электронных кошельков, метки привлекли внимание исследовательских групп, которые провели многочисленные исследования, касающиеся безопасности, которые предлагают метки. MiFare Classic реализуют собственный криптографический алгоритм под названием CRYPTO-1. Это потоковый шифр с 48-битным секретным ключом, который используется для обеспечения конфиденциальности данных и взаимной аутентификации между меткой и считывателем.

Внешняя схема устройства для считывания RFID-меток представлена на рисунке 2.2.

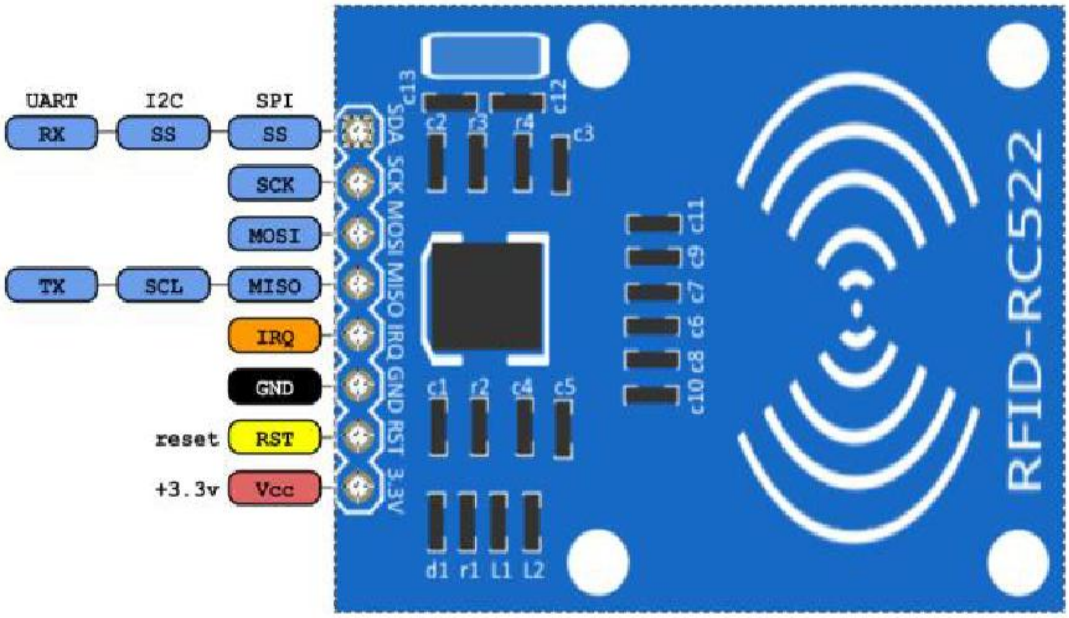

Рисунок 2.2 – Считыватель RC522

Контакты и сигналы RFID RC522:

- VCC — Питание 3.3V;
- RST — Reset. Линия сброса, вход;
- GND — Ground. Земля;
- MISO — Master Input Slave Output — данные от ведомого ведущему, выход SPI;

- MOSI — Master Output Slave Input — данные от ведущего к ведомому, вход SPI;
- SCK — Serial Clock — тактовый сигнал, вход SPI;
- SDA — Slave Select — выбор ведомого, вход SPI;
- IRQ — линия прерываний, выход.

Считыватель поддерживает интерфейсы SPI, UART и I2C через которые происходит обмен данными с другими приборами. На плате модуля RFID RC522 установкой логических уровней на специальных выводах микросхемы выбран интерфейс SPI. С одним Arduino могут работать несколько приборов, подключенных к шине SPI.

Подключение модуля RC533 к Arduino Uno производится соответственно таблице 2.1. Для нормальной работы представленного выше модуля выход IRQ не подключается к Arduino.

Таблица 2.1 – Подключение RC522 к Arduino Uno

MFRC522	Arduino Uno
RST	9
SDA	10
MOSI	11
MISO	12
SCK	13
3.3V	3.3V
GND	GND

В комплекте с данным модулем входит белая пластиковая карта Mifare Classic 1K или метка в виде брелка, изображенная на рисунке 2.3.

Рисунок 2.3 – RFID-метка Mifare 1K

Внутри нее находятся антенна и микросхема Mifare S50, содержащая память и радиочасть. Размер памяти 1 килобайт, тип EEPROM. Она разделена на 16 секторов, состоящих из 4 разделов. В каждом разделе три информационных части и одна для ключей. Внутри одной части есть 16 байт памяти. Срок хранения данных 10 лет, количество циклов перезаписи 100000.

Уникальность карточки Mifare обеспечивается присвоением изготовителем номера, используемого в качестве идентификационного кода. Для защиты данных в микросхеме карты использовано аппаратное шифрование. Во время работы данные с карточки поступают на считыватель только после взаимной идентификации кода, записанного в сектор памяти карточки и хранящегося в считывателе.

Для работы в среде разработки Arduino можно воспользоваться различными сторонними библиотеками, разработанными для того, чтобы существенно упростить работу со считывателем RC522. Для записи и чтения с карты необходимо знать ее уникальный номер, необходимый для работы системы радиочастотной идентификации. Для этого нужно загрузить программу из списка примеров библиотеки RFID под названием «CardInfo», подключить RC522 к Arduino, и запустить программу в среде разработки Arduino IDE. При нахождении рабочей метки в зоне действия RFID – считывателя на мониторе порта появится информация о карте, представленная на рисунке 2.4.

```
Card found
Cardnumber:
Dec: 60, 121, 172, 213, 60
Hex: 3C, 79, AC, D5, 3C
```

Рисунок 2.4 – Идентификационные номера RFID – меток

Программа выводит ряд чисел: 60, 121, 172, 213, 60. Необходимо записать их в обратном порядке. Первое число исключается (контрольная сумма), которое сначала было последним, а оставшиеся числа переводятся в шестнадцатеричный код. Затем они записываются в том же порядке, но без пробелов. Полученное большое число необходимо перевести в десятичный код, вследствие чего получится идентификационный номер карты. С его помощью уже можно проводить различные манипуляции и составлять различные программы, например системы контроля доступа в помещение.

Для того, чтобы вводить кодовую комбинацию, можно воспользоваться специально сконструированной для работы с микроконтроллерами матричной клавиатурой, состоящей из 16 кнопок, расположенных в 4 рядах и 4 столбцах, внутренняя схема которой изображена на рисунке 2.5.

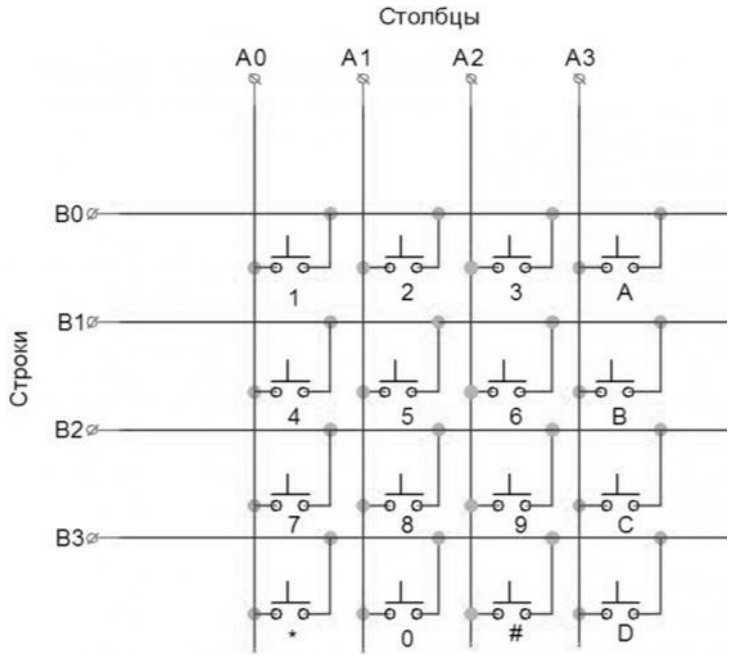

Рисунок 2.5 – Схема матричной клавиатуры 4x4

Матричные клавиатуры для микроконтроллеров достаточно разнообразны в своем построении. Кроме 16-кнопочных клавиатур существуют решения с 12 или с 4 кнопками, с мембранной подложкой или с простыми кнопками. Для решения поставленных задач, воспользуемся типовым решением в виде матричной клавиатуры из 16 кнопок, виды которой представлены на рисунке 2.6.

Для того чтобы подключить матричную клавиатуру к Arduino, от платы выведено 8 контактов, которые подключаются через соединительные провода к цифровым входам микроконтроллера.

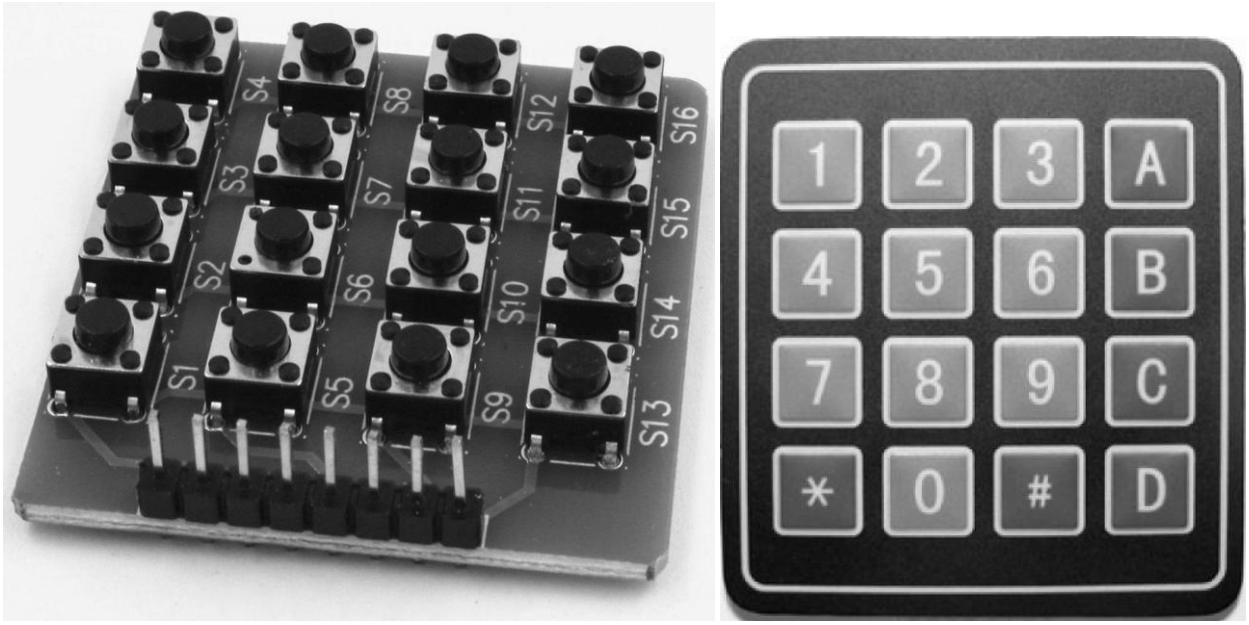

Рисунок 2.6 – Кнопочная и мембранная клавиатуры 4x4

В устройстве многих электронных замков может присутствовать элемент, воспроизводящий звук. Для этих целей подойдет пьезокерамический излучатель звука (пьезодинамик), который может воспроизвести звук на основе пьезоэлектрического эффекта. Пьезодинамик, изображенный на рисунке 2.7, состоит из металлической пластины, с нанесенной на ней пьезоэлектрической керамики, имеющей токопроводящее напыление. Пластина и напыление являются контактами пьезоизлучателя, полярность которых — плюс и минус. Если к контактам приложить

напряжение, под действием обратного пьезоэлектрического эффекта излучатель начнет воспроизводить звук, а если механически воздействовать на пьезоэлемент, то на его контактах появится напряжение. Чтобы подключить пьезодинамик к микроконтроллеру, контакт, обозначенный знаком «+» подключается к любому цифровому входу, а минусовой контакт к выходу GND.

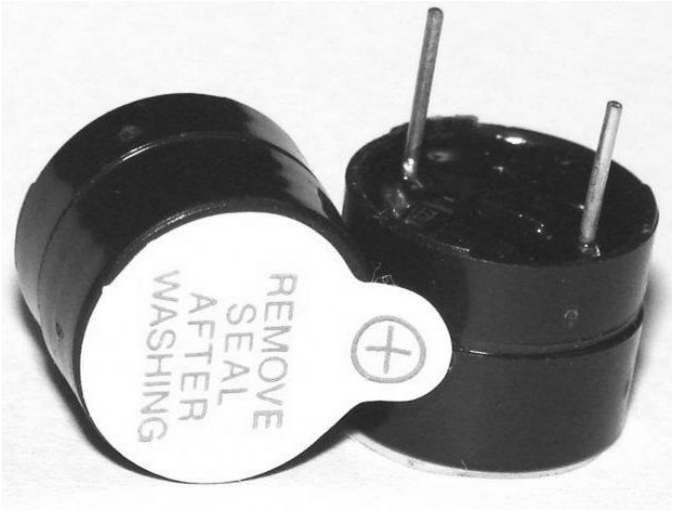

Рисунок 2.7 – Простой пьезокерамический излучатель

Удобным способом отображения различной информации, необходимой во время работы устройства контроля доступа, является жидкокристаллический дисплей. Выберем самый простой в управлении символьный дисплей LCD1602 на основе контроллера HD44870, изображенный на рисунке 2.8. На этом рисунке также изображена плата последовательного I2C-интерфейса на основе микросхемы PCF8574AT, с помощью которой можно подключать дисплей к микроконтроллеру через 4 провода. Подключение происходит очень просто: SDA и SCL входы подключаются к входам SDA и SCL микроконтроллера, VCC и GND соответственно к +5В и GND микроконтроллера.

Рисунок 2.8 – Дисплей LCD1602 и плата подключения I2C-интерфейса

Для наглядности используем модуль RGB-светодиода KY-016, представленный на рисунке 2.9. В этом модуле выводы, отвечающие за передачу цвета, уже подключены через резисторы номиналом 220 Ом, поэтому нет необходимости в отдельных резисторах, чтобы защитить светодиод от выхода из строя. Выводы R, G, B соединяются с цифровыми входами микроконтроллера, а вывод “ - “ к входу GND.

Рисунок 2.9 – RGB-светодиод KY-016

Для коммутации различных приборов используется специальный одноканальный модуль реле для микроконтроллеров, изображен на рисунке 2.10, а принципиальная схема устройства – на рисунке 2.11.

Рисунок 2.10 – Схематичное изображение реле (вид сверху)

В состав реле входят: резисторы номиналом 1 кОм (R1, R2), подтягивающий резистор R3 на 10 кОм, p-n-p транзистор(VT1), обратный диод(VD2) и, реле (К1). VD1 (красный светодиод) – индикация подачи питания на модуль, загорание VD3 (зеленый светодиод) свидетельствует о замыкании реле. Контакты реле показаны на рисунке 2.12.

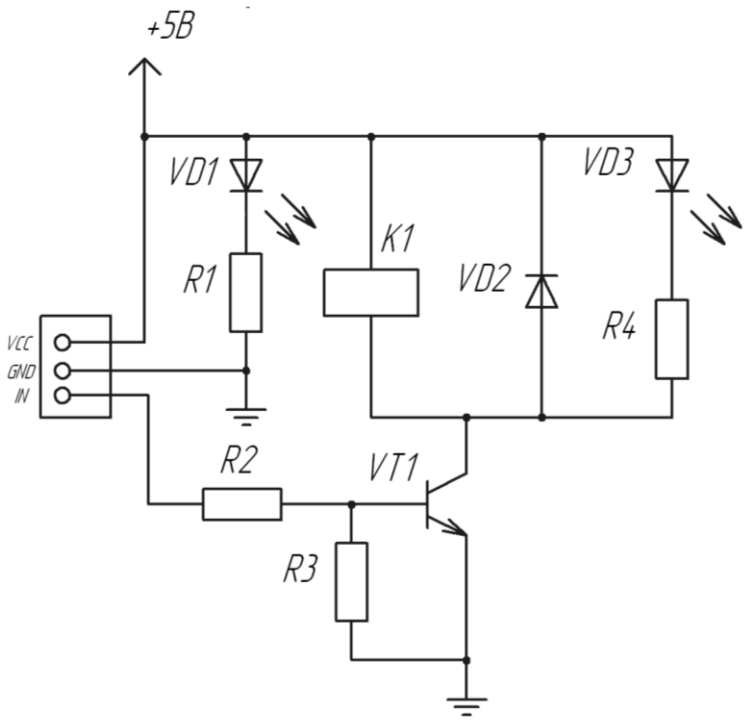

Рисунок 2.11 – Принципиальная схема модуля реле

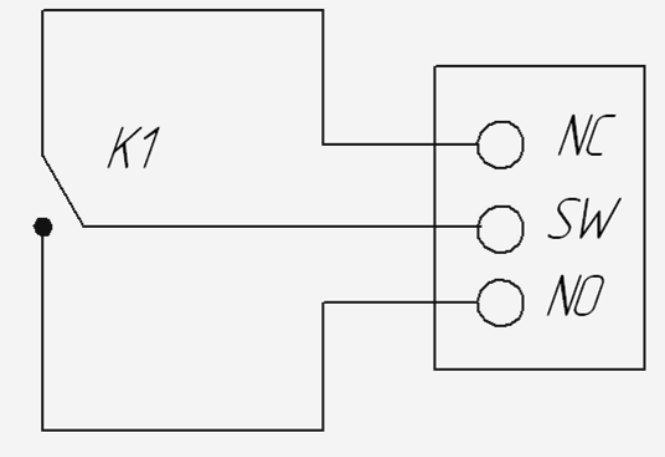

Рисунок 2.12 – Контакты одноканального реле

При включении выводы находятся в высокоомном состоянии, транзистор не открыт. Так как транзистор p-n-p типа, то для его открытия нужно подать на базу минус. Для этого нужно использовать функцию digitalWrite(pin, LOW). Реле срабатывает, когда транзистор открыт и через управляющую цепь течет ток. Для отключения реле следует закрыть транзистор, подав на базу плюс при помощи функции digitalWrite(pin, HIGH). Контакты реле NC – нормально замкнутый, SW – контакт переключения, NO – нормально разомкнутый.

2.2Разработка конструкции устройства

Для представления конструкции устройства контроля доступа воспользуемся очень удобной программой для визуализации всех этапов работы с Arduino. При помощи программы Autodesk Circuits с электронного ресурса circuits.io, распространяемой на бесплатной основе и не требующей установки на персональный компьютер, можно моделировать различные решения, которые будут показаны в удобной для пользователя форме макета или схемы подключения.

Промоделируем модули с помощью Autodesk Circuits, используемые совместно с микроконтроллером Arduino, и выведем их схемы подключения к Arduino Uno R3, показанные на рисунках 2.13 – 2.16.

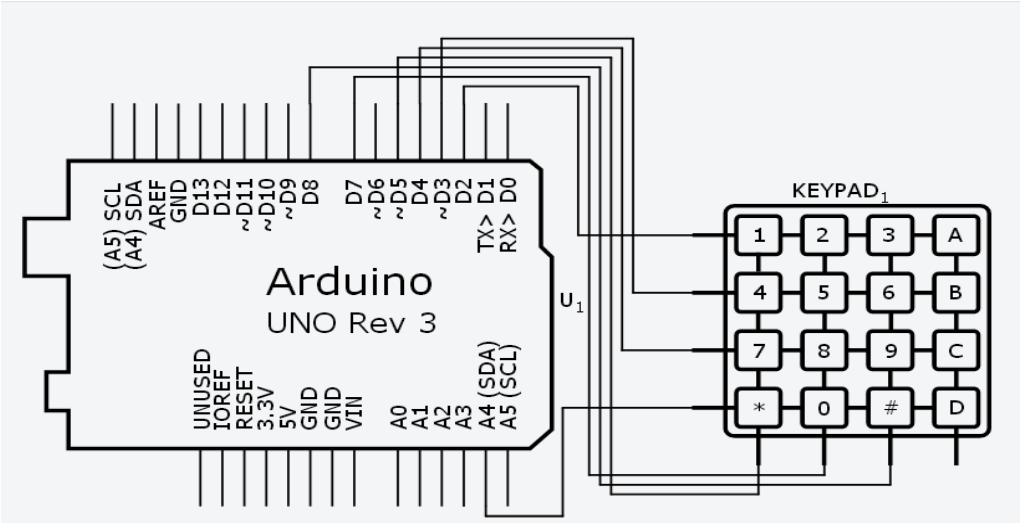

Рисунок 2.13 – Схема подключения матричной клавиатуры

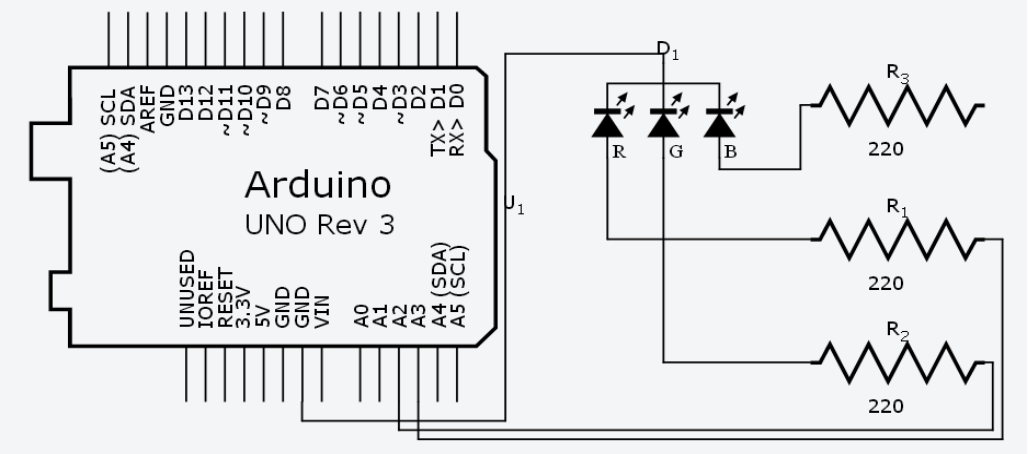

Рисунок 2.14 – Схема подключения RGB – светодиода

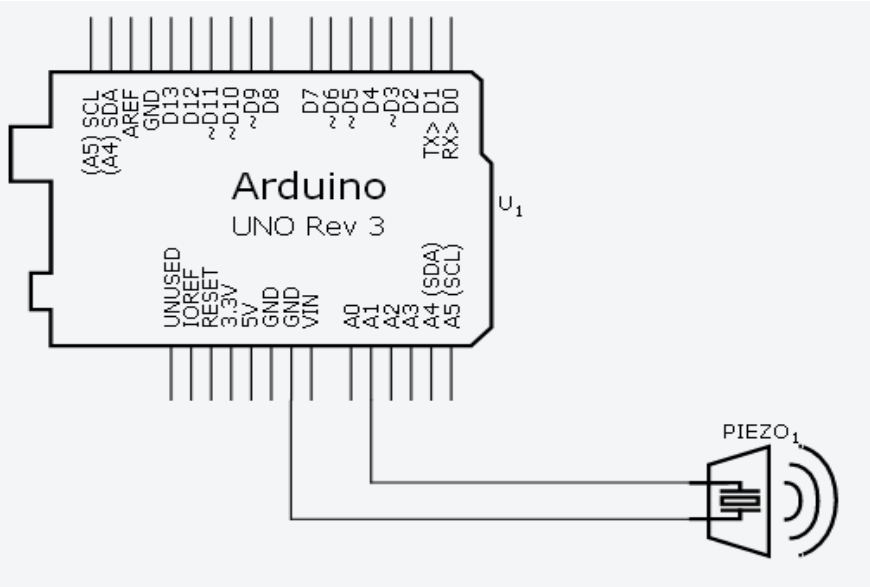

Рисунок 2.15 – Схема подключение пьезодинамика

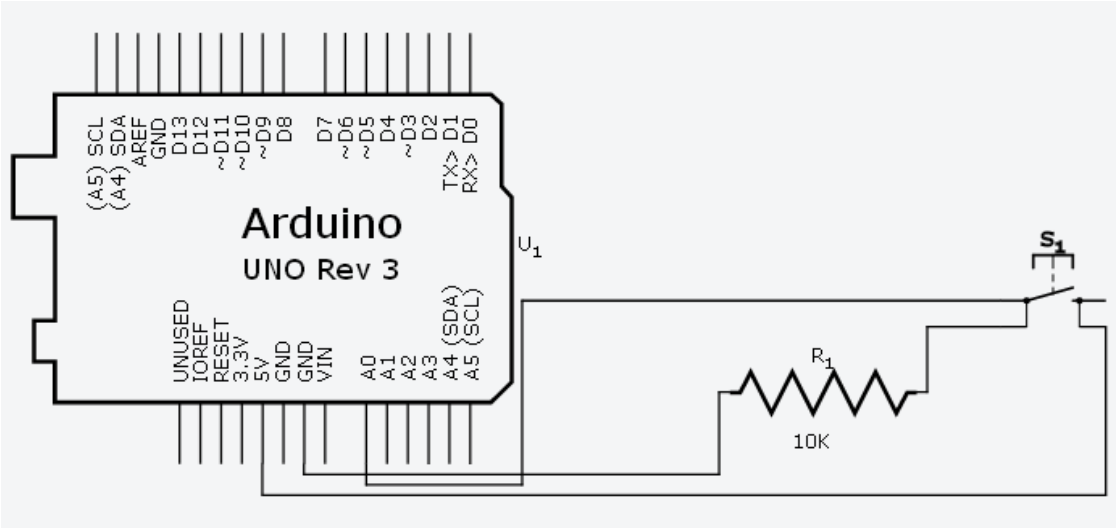

Рисунок 2.16 – Схема подключения кнопки сброса EEPROM

Несмотря на все преимущества представленной выше программы, у нее есть существенный недостаток – сильно ограниченное количество элементов, с которыми можно работать. Поэтому дальнейшее моделирование системы проводилось в программной среде разработки Fritzing, специально разработанной для моделирования схем Arduino. Схемы подключений, созданные в программной среде Fritzing, изображены на рисунках 2.17 – 2.19.

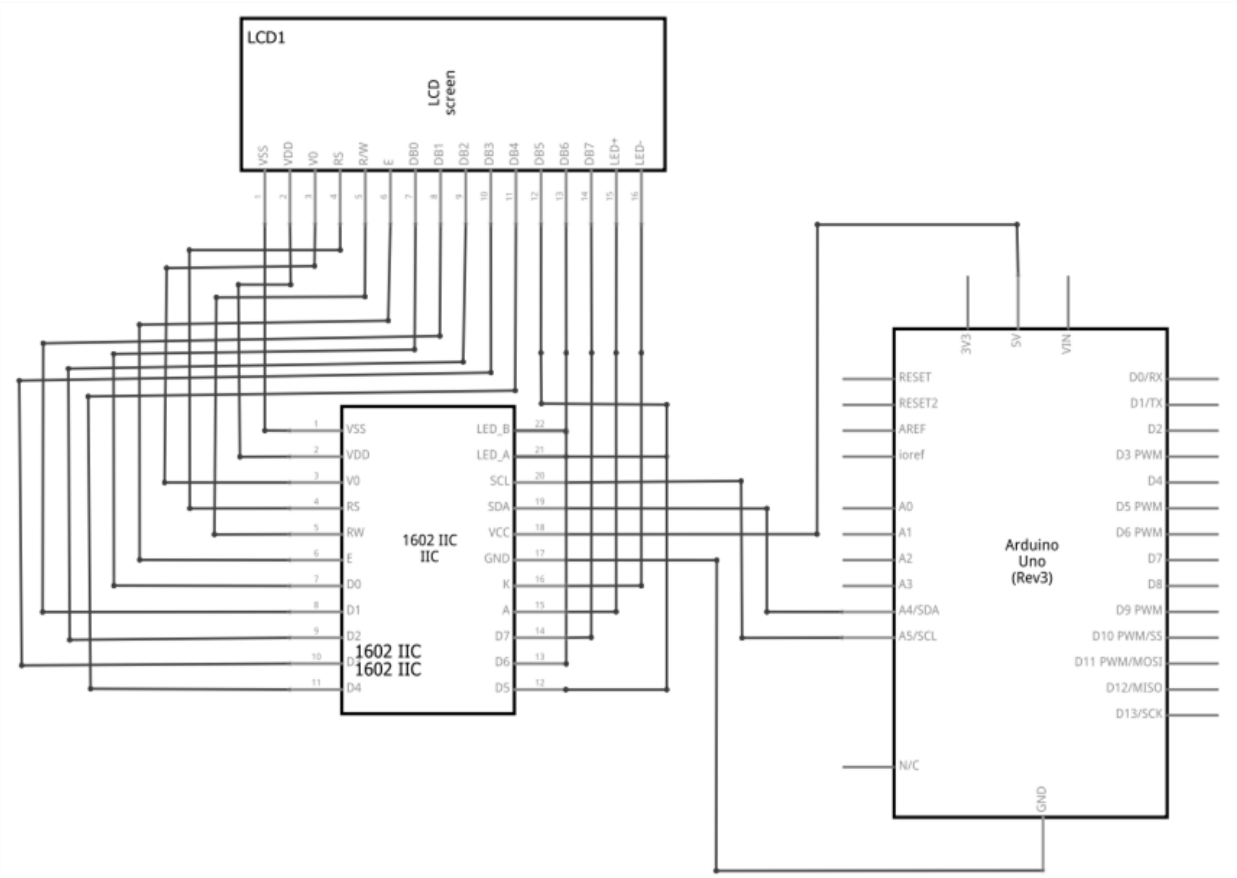

Рисунок 2.17 – Схема подключение дисплея LCD1602 при помощи интерфейса I2C

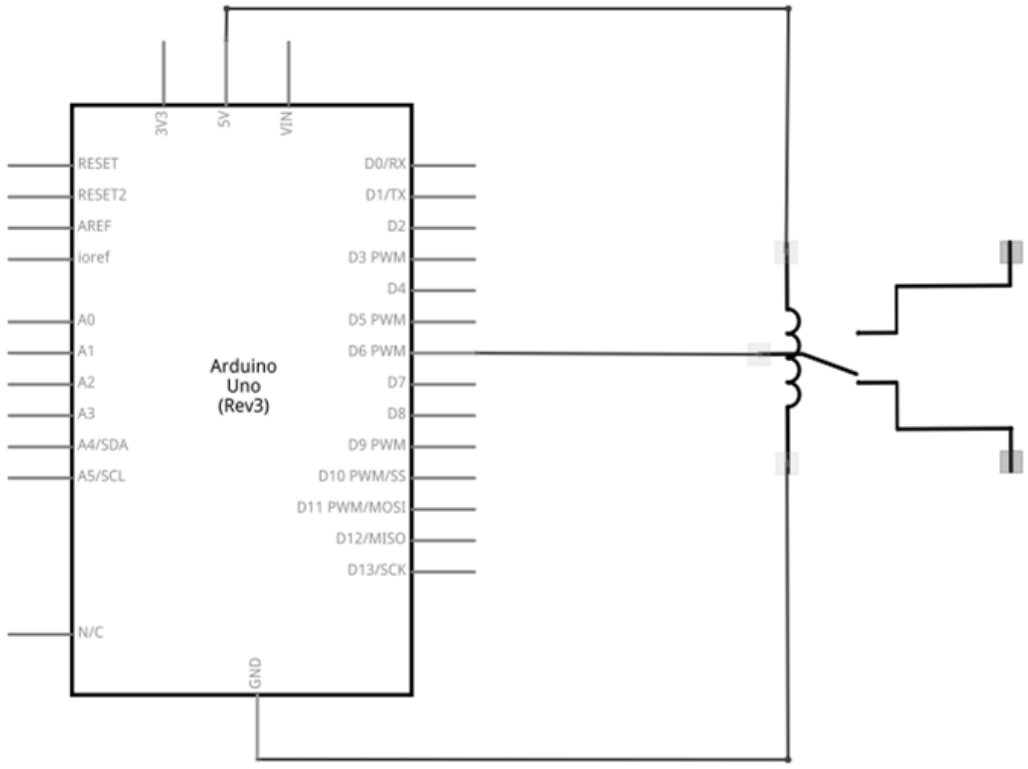

Рисунок 2.18 – Схема подключения одноканального реле

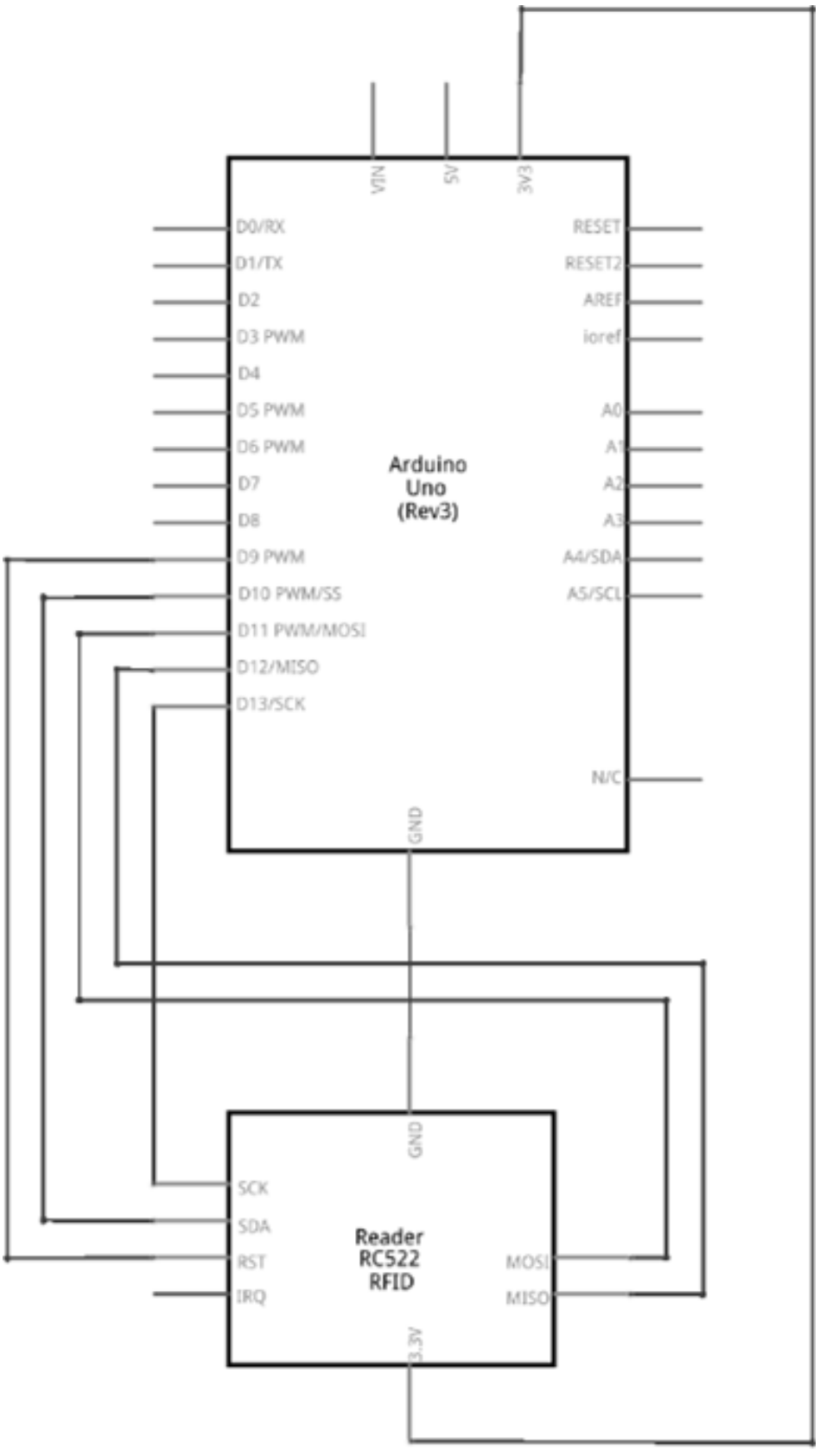

Рисунок 2.19 – Схема подключения RFID-считывателя RC522

В итоге совместили все модули на одной схеме подключения, представленной на рисунке 2.20.

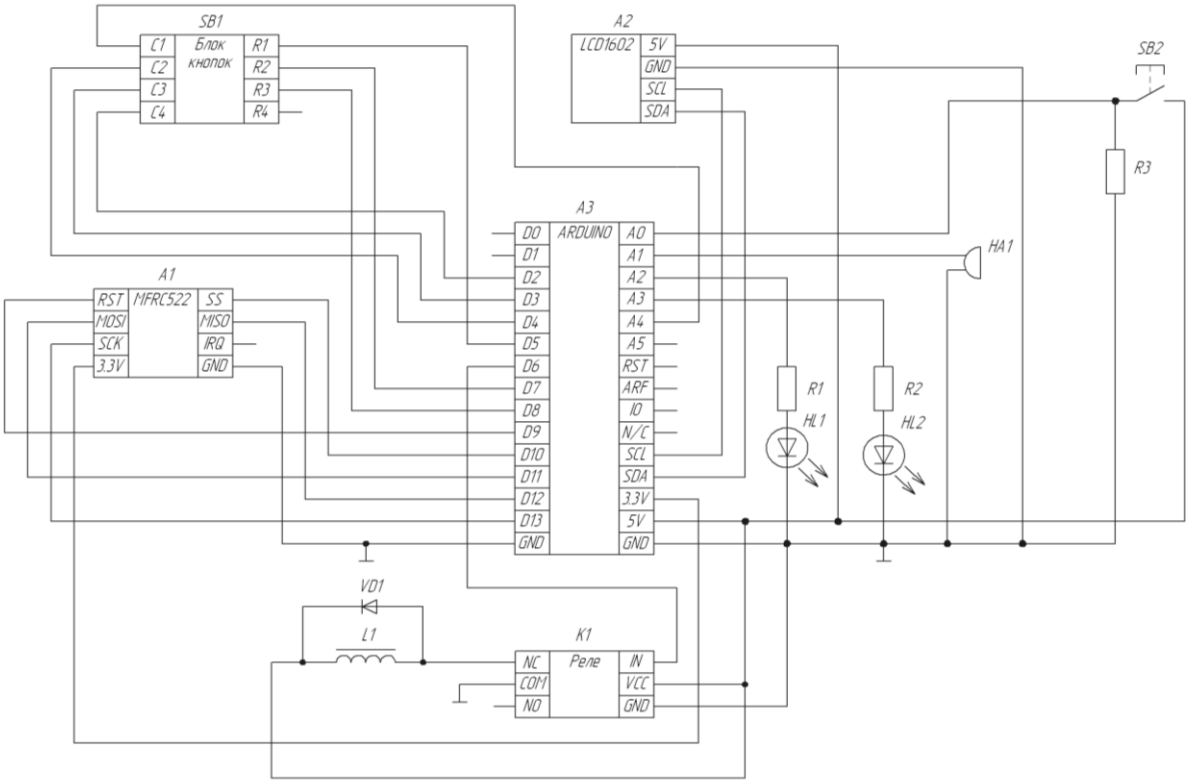

Рисунок 2.20 – Схема подключения всех необходимых компонентов для устройства контроля доступа к микроконтроллеру

2.3Разработка программной части устройства

Для того чтобы начать разрабатывать программу необходимо сначала задаться базовым алгоритмом работы устройства, необходимым для понимания процессов исполнения различных ситуаций (рисунок 2.19).

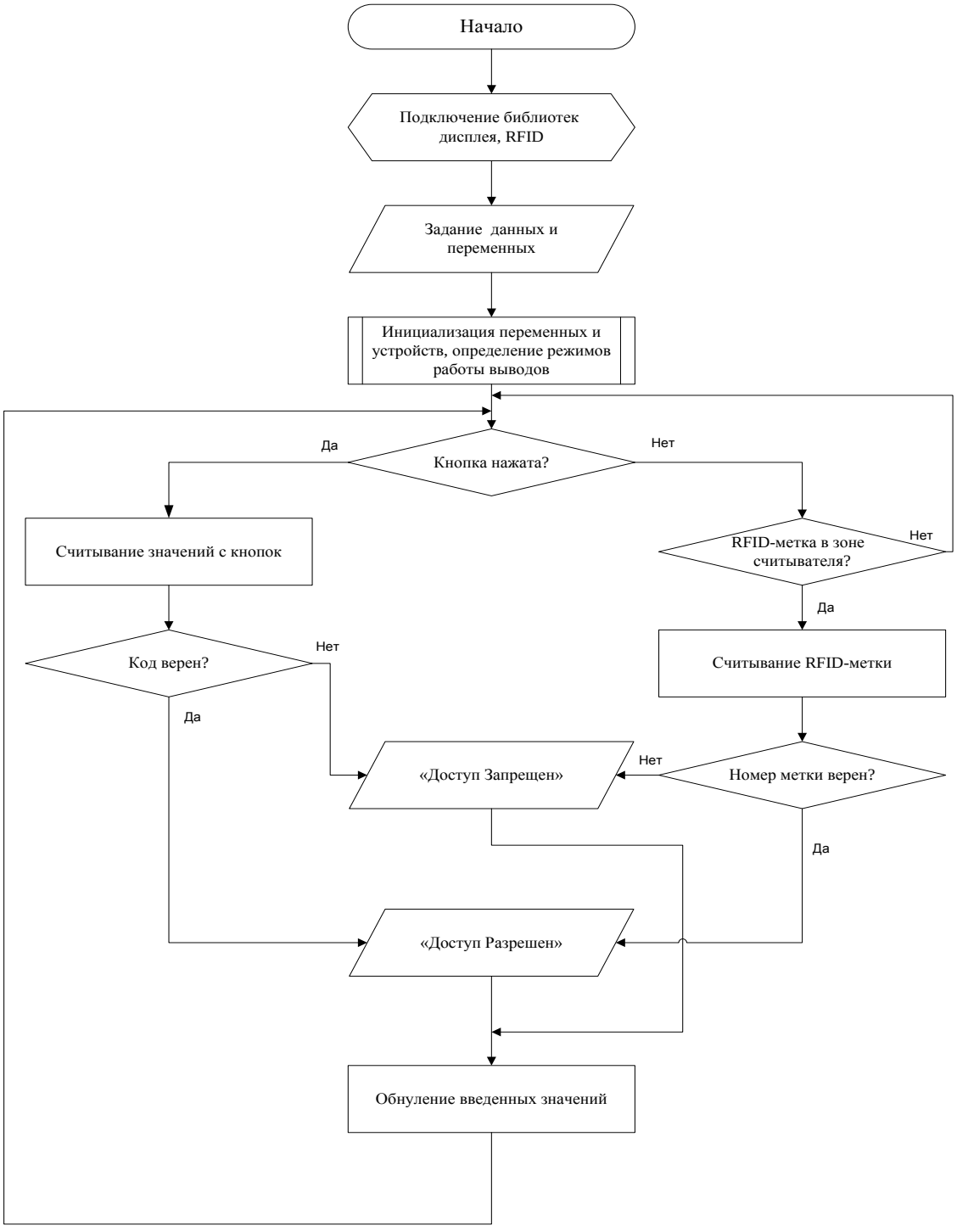

Рисунок 2.19 – Базовый алгоритм работы устройства контроля доступа.

Для написания программы воспользуемся специально разработанной средой разработки Arduino IDE, которая распространяется в свободном доступе в сети Интернет и которую можно найти на официальном сайте разработчика www.arduino.cc. В итоге, после длительного процесса разработки путем проб и ошибок была составлена программа, с помощью которой можно реализовать основной функционал работы электронного замка на основе Arduino Uno R3. Для правильной работы программы необходимо заранее установить несколько сторонних библиотек либо через программу «Arduino» - «Скетч» - «Подключить библиотеку» - «Управлять библиотеками» - «Поиск», либо через ресурсы сети Интернет, установив ее по пути «Arduino» - «Скетч» - «Подключить библиотеку» - «Добавить .ZIP библиотеку». Необходимые сторонние библиотеки для работы с программой: «Keypad», «Password», «MFRC522», «LCD_1602_RUS», «Bounce2». Программа, реализующая потенциал устройства контроля доступа в помещение приведена в приложении А.

3 Практическая реализация проекта

3.1Изготовление системы

Для подключения различных модулей к микроконтроллеру Arduino существует специальную макетную плату, на которой удобно располагать различные элементы схемы, а также соединять их проводами между собой и микроконтроллером. В конечном итоге, получилась схема, представленная на рисунке 3.1. Все элементы соединены с микроконтроллером согласно принципиальным схемам, которые были разработаны в специализированных программах моделирования Autodesk Circuits и Fritzing.

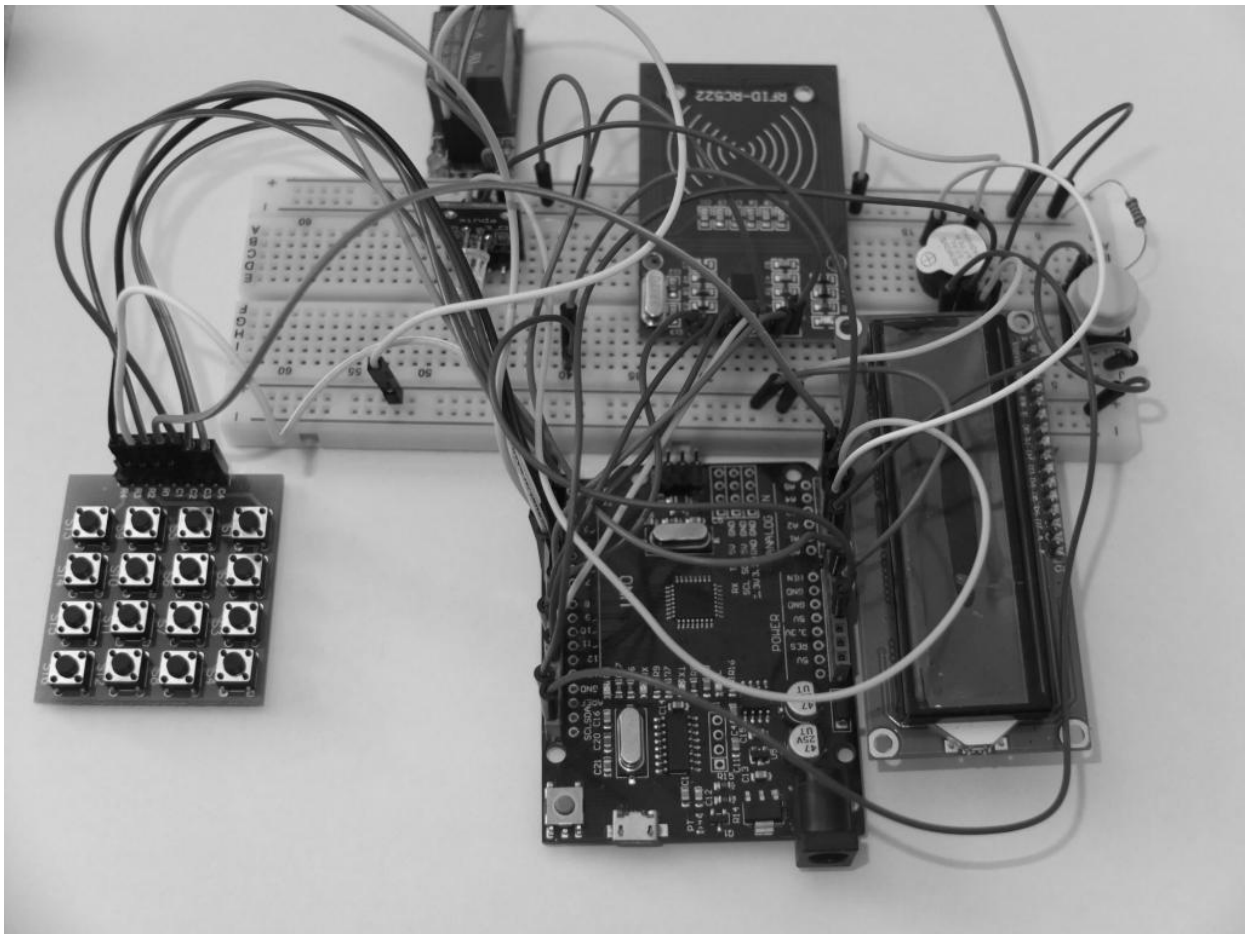

Рисунок 3.1 – Полная схема устройства контроля доступа

Рассмотрим подключение RFID-считывателя RC522 крупным планом, представленным на рисунке 3.2, и светодиода, изображенного на рисунке 3.3.

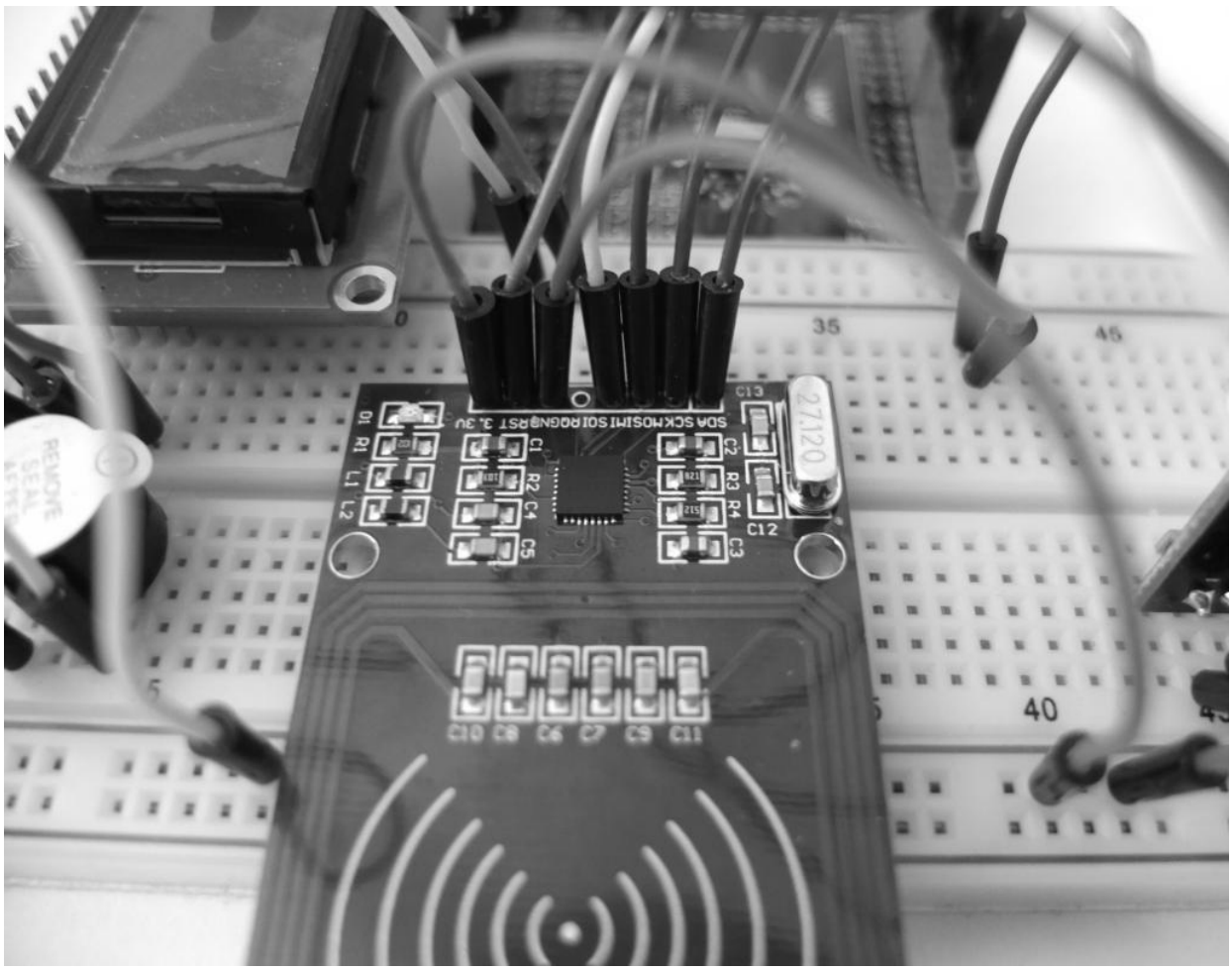

Рисунок 3.2 – Соединение RC522 с микроконтроллером

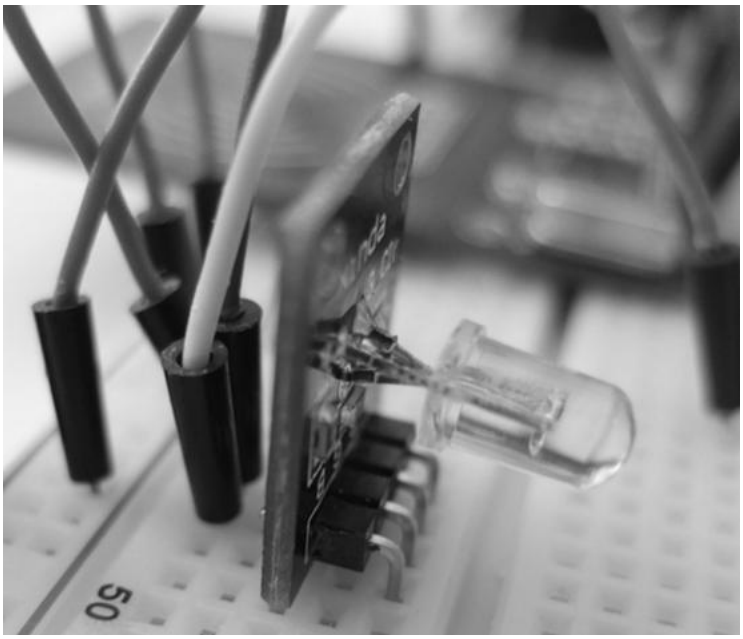

Рисунок 3.3 – Соединение RGB-светодиода с микроконтроллером

3.2Проверка и отладка программной части устройства

В ходе практической работы и создания модели электронного устройства контроля доступа, возникли некоторые проблемы, а также новые идеи по модернизации программной части. В частности, была произведена замена рабочей библиотеки RFID с Rfid.h на MFRC522, у которой функционал превосходил предшествующую библиотеку. Затем была добавлена отдельная функция squeaker(), с помощью которой было проще и удобней настраивать пьезодинамик. На плате микроконтроллера Arduino UNO R3 присутствуют два дополнительных входа SDA и SCL, позволяющие работать по протоколу I2C с различными устройствами, в нашем случае, с дисплеем LCD1602. Изменения произошли и с информацией, которая выводилась на дисплей. Изначально все символы, выводимые во время работы устройства, были записаны латинскими буквами. Решить этот вопрос помогла сторонняя библиотека LCD_1602_RUS.h, позволяющая использовать кириллицу. В конечном итоге, самым главным решением было добавление программной работы с энергонезависимой перезаписываемой памятью EEPROM микроконтроллера, вследствие чего идентификационные номера RFID-меток, а затем и кодовая комбинация, вводимая при помощи матричной клавиатуры. Они стали записываться в эту энергонезависимую память и при работе программы выводиться в последовательный порт персонального компьютера, а не задаваться в самой программе, как было ранее. Вместе с EEPROM была добавлена специальная кнопка сброса памяти микроконтроллера, которая заменяла бы все данные в EEPROM на нулевые значения, кроме значений, используемых в пароле.

Теперь следует проверить основные этапы работы устройства контроля доступа. На рисунке 3.4 показана информация, которая выводится в последовательный порт компьютера. Информацию с последовательного порта можно отобразить на экране монитора, зайдя в среду разработки Arduino IDE, в строке «Инструменты» - «Порт» указать номер COM порта, к

которому подключен микроконтроллер, а затем в строке «Инструменты» - «Монитор порта», что позволит вывести отладочную информацию на экран монитора.

```
Start

KEYS COUNT: 1
------------------------------
KEY: 0 | 60 121 172 213
------------------------------

PASSWORD: 1204
------------------------------
```

Рисунок 3.4 – Отладочная информация, указывающая количество доступных системе ключей, их идентификационный номер, а также пароль

При запуске программы с чистой EEPROM памятью, или же когда она была очищена при помощи специальной кнопки сброса, на экране отобразится информация, представленная на рисунке 3.5. В этом случае контакты реле будут замкнуты, таким образом моделируя открытое состояние, программа запросит приложить RFID-ключ к считывателю, который впоследствии и станет мастер-ключом.

```
Memory cleaning is completed


Start

The master key is not in memory. The first presentation to the key will be the master!
```

Рисунок 3.5 – Отладочная информация, указывающая, что в памяти микроконтроллера отсутствуют RFID-метки.

При установке мастер-ключа, программа запишет его идентификационный номер и укажет на его принадлежность (рисунок 3.6).

```
UID: 60 121 172 213
master key is created
```

Рисунок 3.6 – Создание мастер-ключа

Мастер-ключ позволяет вносить в память микроконтроллера новые ключи при его удержании у считывателя, а также выходить из этого режима программирования за то же время удержания. В это время реле замыкает свои контакты, и не размыкает до тех пор, пока не произойдет выход из режима программирования новых ключей (рисунок 3.7). Весь этот этап сопровождается звуковыми сигналами и сообщенияеми на дисплее.

```
UID: 60 121 172 213
MASTER PROGRAMMING MODE ON

UID: 101 229 204 101
add key in eeprom

KEYS COUNT: 2
------------------------------
KEY: 0 | 60 121 172 213
KEY: 1 | 101 229 204 101
------------------------------

UID: 60 121 172 213
MASTER PROGRAMMING MODE OFF
```

Рисунок 3.7 – Режим программирования

Ввод верной кодовой комбинации или одного из записанных в памяти ключей сопровождаются характерными звуковыми сигналами, информации на дисплее (рисунок 3.8), а также замыканием контактов реле и срабатывании зеленого светодиода в течение пяти секунд, после система приходит в начальное положение. Неправильный код или неизвестный ключ заставляют систему реагировать отрицательно: устройство становится недоступным на короткий промежуток времени, воспроизводится характерный звуковой сигнал, на экране дисплея появляется сообщение

«Доступ запрещен», после этого система возвращается к исходной позиции.

Рисунок 3.8 – Информация на дисплее при верном решении

Кодовая комбинация «0000» позволяет входить в режим изменения пароля. После этого дается три попытки на ввод предыдущей комбинации, иначе выводится сообщение, изображенное на рисунке 3.9.

```
Pass 7777
Attention!_3xWrong_Pass for change pas - cancel change pass
Access denied!
```

Рисунок 3.9 – Отладочное сообщение на попытку ввода неправильного пароля в третий раз.

При смене пароля также недопустимым является комбинация «0000», а также прежний пароль. В этом случае на дисплее отображается сообщение «Старый пароль», но программа позволяет ввести новый. Когда новая комбинация успешно введена, загорается зеленый светодиод, затем система переходит в начальное состояние.

Инструкция по работе с устройством контроля доступа в помещение изложена в приложении Б.

Заключение

В данной выпускной квалификационной работе осуществлена разработка, создание рабочей программы, изготовление, отладка и действующего макета устройства контроля доступа в помещение на микроконтроллерном управлении.

Устройство моделирует основные функции контролирования за различными датчиками и устройствами, входящими в периферию электронной системы.

Макет электронного замка позволяет смоделировать основные функции и задачи по контролю доступа в различные помещения при помощи ввода кодовой комбинации через матричную клавиатуру, или радиочастотной метки, используемой в качестве ключ-карты. Модуль реле и подключенный к его контактам электромагнит симулируют открытое или закрытое состояния двери в помещение. При утере мастер-ключа предусмотрен сброс памяти микроконтроллера с целью указания нового мастер-ключа, используемого для ввода новых карт доступа.

Вся отладочная информация будет поступать на компьютер только при подключении к нему микроконтроллера через USB-интерфейс. Таким образом, устройство может работать автономно через подключенный сетевой блок питания, но в этом случае становится недоступной отладочная информация.

Основной целью данной работы было создание макета устройства контроля доступа на микроконтроллерном управлении. Полученная система позволяет смоделировать основные функции, доступные аналогичным механизмам без микроконтроллеров.

Список используемой литературы

1. Шарфельд Т. Системы RFID низкой стоимости / Пер. с англ. / Под ред. Корнеева С.В. М.: 2006.- С.9-11. [Scharfeld T.A. An Analysis of the Fundamental Constrains on Low Cost Passive Radio Frequency Identification System Design, 2001].
2. RFID Based Security and Access Control System / U.Farooq, Mahmood ul Hasan, M.Amar et al. // IACSIT International Journal of Engineering and Technology, Vol. 6, No.4, P.309-314, August 2014. [Электронный ресурс]. URL: https://www.researchgate.net/publication/275685766_RFID_Based_Security_and_Access_Control_System (дата обращения: 4.04.2017).
3. A Digital Security System with Door Lock System Using RFID Technology / G.K Verma, P.Tripathi // International Journal of Computer Applications, Volume 5 – No.11, P.6-8, August 2010. [Электронный ресурс]. URL: https://www.researchgate.net/publication/45602075_A_Digital_Security_System_with_Door_Lock_System_Using_RFID_Technology (дата обращения: 4.04.2017).
4. A Review on Chipless RFID Tag Design / A.Hashemi, A.H.Sarhaddi, H.Emami // Majlesi Journal of Electrical Engineering, vol.7, No.2, P.68-75, June 2013. [Электронный ресурс].URL:https://www.researchgate.net/publication/260190506_A_Review_on_Chipless_RFID_Tag_Design (дата обращения: 4.04.2017).
5. Кодовые замки на двери: виды и их особенности. [Электронный ресурс]. URL: http://dveridoma.net/kodovye-zamki-na-dveri/ (дата обращения: 7.05.2017).
6. Электронный кодовый замок. [Электронный ресурс]. URL: http://www.meanders.ru/kodovij_zamok.shtml (дата обращения: 7.05.2017).
7. Arduino UNO R3: схема, инструкция. [Электронный ресурс]. URL: https://www.syl.ru/article/203717/new_arduino-uno-r-shema-instruktsiya (дата

обращения: 6.04.2017).

8. Arduino Uno R3. [Электронный ресурс]. URL: http://radiodetalki.narod.ru/pribory/Arduino_Uno_R3.pdf (дата обращения: 21.04.2017).
9. RFIDRC522. [Электронный ресурс]. URL: https://arduino-kit.ru/userfiles/image/RFIDRC522.pdf (дата обращения: 19.04.2017).
10. Electronic Wallet and Access Control Solution Based on RFID MiFare Cards / Stefan-Victor Lefter // Journal of Mobile, Embedded and Distributed Systems, vol.5, no.1, P29-35, 2013. [Электронный ресурс]. URL: http://www.jmeds.eu/index.php/jmeds/article/view/Electronic_Wallet_and_Access_Control_Solution_Based_on%20%20_RFID_MiFare_Cards (дата обращения: 4.04.2017).
11. Как хранить данные в Arduino. [Электронный ресурс]. URL: http://soltau.ru/index.php/arduino/item/378-kak-khranit-dannye-v-arduino (дата обращения: 11.05.2017).
12. Форум Калининграда - Кодовый замок на Ардуино. [Электронный ресурс]. URL: http://forklg.ru/viewtopic.php?t=830 (дата обращения: 15.04.2017).
13. Подключение матричной клавиатуры к Arduino. [Электронный ресурс]. URL: http://robots4life.ru/arduino-keypad (дата обращения: 19.04.2017).
14. Подключение пьезоизлучателя к Ардуино. [Электронный ресурс]. URL: http://роботехника18.рф/включение-пьезопищалки-на-ардуино/ (дата обращения: 22.04.2017).
15. Подключение LCD 1602 по I2C интерфейсу. [Электронный ресурс]. URL: http://radiolaba.ru/microcotrollers/podklyuchenie-lcd-1602-po-i2c-interfeysu.html (дата обращения: 26.04.2017).
16. Подключение RGB светодиода к Ардуино. [Электронный ресурс]. URL: http://роботехника18.рф/подключение-rgb-светодиода-к-ардуино/ (дата обращения: 24.04.2017).

17. Реле модуль подключение к Arduino. [Электронный ресурс]. URL: http://zelectro.cc/relayModule (дата обращения: 6.05.2017).
18. Arduino transistor relay. [Электронный ресурс]. URL: http://www.asirunningshoes.com/arduino/arduino-transistor-relay (дата обращения: 6.05.2017).
19. Autodesk Circuits. [Электронный ресурс]. URL: https://circuits.io (дата обращения: 3.05.2017).
20. Fritzing. [Электронный ресурс]. URL: http://fritzing.org/home/ (дата обращения: 3.05.2017).
21. Аппаратная платформа Arduino. [Электронный ресурс]. URL: http://arduino.ru (дата обращения: 3.04.2017).

Приложение А

Листинг программы устройства контроля доступа в помещение

Инициализация библиотек, необходимых для упрощения работы с различными модулями: avr/wdt.h – стандартная библиотека для управления сторожевыми (watchdogs) таймерами, при переполнении которых произойдет программный сброс микроконтроллера; Wire.h – стандартная библиотека для работы с I2C/TWI-устройствами; LCD_1602_RUS.h – сторонняя библиотека, позволяющая выводить на экран дисплея русские символы; SPI.h – стандартная библиотека для работы с устройствами, поддерживающими SPI протокол; MFRC522.h – сторонняя библиотека для работы с RFID-модулем RC522; Bounce2.h – сторонняя библиотека для программного устранения дребезга кнопок; EEPROM.h – стандартная библиотека работы с энергонезависимой перезаписываемой памятью EEPROM у Arduino; Password.h, Keypad.h – сторонние библиотеки, реализующие возможности матричных клавиатур ввода данных.

Начало программы:

```
#include <avr/wdt.h>
#include <Wire.h>
#include "LCD_1602_RUS.h"
#include <SPI.h>
#include <MFRC522.h>
#include <Bounce2.h>
#include <EEPROM.h>
#include <Password.h>
#include <Keypad.h>
LCD_1602_RUS lcd(0x27, 16, 2); // инициализация дисплея, 16 столбцов, 2 строки

//Определение основных пинов, к которым подключаются различные модули:
#define PIN_RESET 14 // кнопка для сброса EEPROM
#define PIN_RELAY 6 // подключение реле
#define PIN_TONE 15 // пьезодинамик
```

```
#define PIN_RST 9 // RFID RST
#define PIN_SS 10 // RFID SS
#define RED_LED 17 // красный светодиод
#define GREEN_LED 16 //зеленый светодиод

//Инициализация RFID-считывателя:
MFRC522 mfrc522(PIN_SS, PIN_RST);

//Переменные, необходимые для работы со списком RFID-ключей:
byte **keyss;
byte keys_count = EEPROM.read(0);

//Переменные необходимые для режима программирования RFID-меток:
byte modeProgTime = 5; // Количество секунд удержания мастер ключа для входа или выхода из режима программирования
bool mode = false;
byte modeClean = 0;
unsigned long modeTimer = 0;
unsigned long resetTimer = 0;

//Управление замком:
unsigned long openTimer = 0;

//Защита кнопок от дребезга:
Bounce key_reset = Bounce();
Bounce key_open = Bounce();

//Программный reset:
void(* resetFunc) (void) = 0;

//Функция звукового оповещения. Принимает параметры: количество
//звуковых сигналов, частота в герцах, продолжительность звука, пауза в
//милисекундах (не обязательно):

void squeaker(byte count, unsigned int Hz, unsigned int duration, unsigned int sleep = 0)
{
 for(int i=0; i<count; i++) {
 tone(PIN_TONE, Hz, duration);
```

```
 if(sleep > 0) delay(sleep);
 }
}

//Функция для считывания EEPROM и составления списка RFID-ключей.
//Первый байт в памяти содержит количество ключей. UID ключа содержит 4
//байта. Максимум можно записать 254 ключа (255-1 из-за того, что в
//EEPROM записывается кодовая комбинация из 4 символов):

void keysRead() {
 //Выводим количество ключей:
 Serial.print(F("KEYS COUNT: "));
 Serial.println(keys_count);
 int eb = 4; // Запись количества доступных ключей     производится в
 keyss = (byte**)malloc(sizeof(byte*)*keys_count);
 // Читаем список ключей из EEPROM:
 Serial.println(F("-----------------------------"));
 for(byte i=0; i<keys_count; i++) {
 Serial.print(F("KEY: "));Serial.print(i);Serial.print(" | ");
 keyss[i] = (byte*)malloc(sizeof(byte)*4);
 for(byte b=0; b<4; b++) {
 keyss[i][b] = EEPROM.read(++eb);
 Serial.print(keyss[i][b]);
 if(b < 3) Serial.print(F(" "));
 }
 Serial.println();
 }
 Serial.println(F("-----------------------------"));
 Serial.println();
}

//Функция вывода пароля, записанного с 1 по 5 ячейку памяти, т.к
//переменные записываются в типе char, то нужно перевести код ASCII в
//десятичный. Для цифр от 0 до 9 это можно сделать просто вычитанием из
//полученного результата числа 48:
void passRead() {
Serial.print(F("PASSWORD: "));
 Serial.print(EEPROM.read(1)-48);
 Serial.print(EEPROM.read(2)-48);
```

```
 Serial.print(EEPROM.read(3)-48);
 Serial.print(EEPROM.read(4)-48);
 Serial.println();
 Serial.println(F("----------------------------"));
 Serial.println(); }

//Функция выводит UID ключа и, при необходимости, сопроводительное
//сообщение:
 void uidPrint(String text = "") {
 Serial.print(F("UID: "));
 for(byte i=0; i<mfrc522.uid.size; i++) {
 Serial.print(mfrc522.uid.uidByte[i]);
 if(i < mfrc522.uid.size - 1) Serial.print(F(" ")); }
 Serial.println();
 if(text.length() != 0) Serial.println(text + "\r\n");
}

int presses=0; // количество нажатий
const byte ROWS = 4; // количество строк
const byte COLS = 4; // количество столбцов

// Определение символов матричной клавиатуры:
char keys[ROWS][COLS] =
{
 {'1','2','3','A'},
 {'4','5','6','B'},
 {'7','8','9','C'},
 {'*','0','#','D'}
};

//Цифровые входы, к которым подключается матричная клавиатура:
byte rowPins[ROWS] = {2,3,4,18};
byte colPins[COLS] = {5,7,8,19};

//Инициализация матричной клавиатуры:
Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS
);
String pass; //переменная, указывающая код по умолчанию
String summ; //переменная, указывающая уже введенный код
```

```
int wrong=0; //количество ошибочных вводов кода (для блокировки)
int shetch=1; //количество набранных символов для смены пароля
int change=0; //флаг проверки кода для смены пароля
int dochange=0; //флаг производится смена пароля

//Функция, запускающаяся только в начале работы микроконтроллера или
//при аппаратном сбросе:
void setup()
{
 // Настраиваем сторожевой таймер
 wdt_disable();
 //delay(8000);
 wdt_enable(WDTO_8S);

//Запись исходного пароля в энергонезависимую память, необходимо только
//при первоначальной прошивке, чтобы задать пароль:

//EEPROM.write(1, '1');
//EEPROM.write(2, '2');
//EEPROM.write(3, '0');
//EEPROM.write(4, '4');

//Запись символов в формате char:
 pass = "";
 pass = pass+char(EEPROM.read(1));
 pass = pass+char(EEPROM.read(2));
 pass = pass+char(EEPROM.read(3));
 pass = pass+char(EEPROM.read(4));

//Эти строки только для отладки, выводится в COM-порт, к которому
//подключен микроконтроллер:
 Serial.println(char(EEPROM.read(1))); // первый знак пароля в
 Serial.println(char(EEPROM.read(2))); // второй знак пароля
 Serial.println(char(EEPROM.read(3))); // третий знак пароля
 Serial.println(char(EEPROM.read(4))); // четвертый знак пароля
 Serial.print("Pass "); Serial.println(pass); //весь пароль


//Инициализация используемых входов:
```

```
//Реле:
 pinMode(PIN_RELAY, OUTPUT);
 digitalWrite(PIN_RELAY, HIGH);

//Кнопка для сброса памяти:
 pinMode(PIN_RESET,INPUT_PULLUP);
 key_reset.attach(PIN_RESET);
 key_reset.interval(5);

//Инициализация консоли последовательного вывода данных на экран:
 Serial.begin(9600);
 while (!Serial);
 Serial.println(F("Start\r\n"));

//Инициализация основных модулей:
 lcd.init();
 lcd.backlight();
 SPI.begin();
 mfrc522.PCD_Init();
 pinMode(RED_LED, OUTPUT);
 pinMode(GREEN_LED, OUTPUT);
 digitalWrite(RED_LED, HIGH);
 digitalWrite(GREEN_LED, LOW);

 lcd.setCursor(4,0);
 lcd.print(L"ОЖИДАНИЕ");
 lcd.setCursor(4,1);
 lcd.print(L"ДЕЙСТВИЯ");

//Считываем количество ключей, значение должно быть =>1 т.к первый ключ
//это мастер-ключ. В случае его потери, сбросить EEPROM и создать новый
//мастер-ключ:

 if(keys_count > 0 and keys_count < 255) {
 keysRead();
 passRead();
 }
 else {
```

```
 keys_count = 0; //Определение нового мастер-ключа
 Serial.println(F("The master key is not in memory. The first presentation to the
key will be the master!\r\n"));
 digitalWrite(PIN_RELAY, LOW);
 lcd.clear();
 lcd.setCursor(4,0);
 lcd.print(L"УКАЖИТЕ");
 lcd.setCursor(2,1);
 lcd.print(L"МАСТЕР-КЛЮЧ");
 }
}

//Функция цикла программы:
void loop()
{
 // Сбрасываем сторожевой таймер микроконтроллера:
 wdt_reset();

 if(resetTimer > millis()+10000) resetTimer = 0;
 if(openTimer > millis()+10000) openTimer = 0;
 char key = keypad.getKey(); //задаем функцию для работы с кнопками
клавиатуры
 if (key) // если нажата кнопка клавиатуры
{
 Serial.println(key);
 squeaker(1, 2000, 100);
 presses=presses+1; //увеличиваем на единицу счет количества символов
 summ=summ+key;
 Serial.print("Pass "); Serial.println(summ);

//Вывод на дисплей знаков «*» при нажатии на клавиатуру:
if(presses == 1 or shetch == 1)
{
lcd.clear();
lcd.setCursor(1,0);
lcd.print(" < PIN >");
lcd.setCursor(0,1);
lcd.print("*_");
}
```

```
if(presses == 2 or shetch == 2)
{
lcd.clear();
lcd.setCursor(1,0);
lcd.print(" < PIN >");
lcd.setCursor(0,1);
lcd.print("**_");
}
if(presses == 3 or shetch == 3)
{
lcd.clear();
lcd.setCursor(1,0);
lcd.print(" < PIN >");
lcd.setCursor(0,1);
lcd.print("***_");
}
if(presses == 4 or shetch == 4)
{
lcd.clear();
lcd.setCursor(1,0);
lcd.print(" < PIN >");
lcd.setCursor(0,1);
lcd.print("****");
}

 switch (dochange) //ветвление на ввод нового пароля (case 1) и проверку (case 0)
 {
 case 0: //блок для обычной работы - проверка правильности ввода пароля

//В данном случае кнопки # и * приводят к сбросу количества нажатий:
 if (key=='#')
 {
 summ="";
 presses=0;
 Serial.println("# for RESET");
 squeaker(1, 500, 100);
 };
```

```
 if (key=='*')
 {
 summ="";
 presses=0;
 Serial.println("* for ENTER");
 squeaker(1, 500, 100);
 };

// Если правильный пароль и не было запроса на его смену:
 if (summ==pass && change==0) {
 Serial.println("PASS OK");
 summ="";
 presses=0;
 wrong=0;
 openTimer = millis()/1000;
 squeaker(2, 3500, 200, 100);
 digitalWrite(PIN_RELAY, LOW);
 allow();
 };

//Если нажата комбинация для смены пароля:
 if (summ=="0000") {
 Serial.println("Change pass go test");
 summ="";
 presses=0;
 wrong=0;
 change=1;
 squeaker(3, 700, 150);
 lcd.clear();
 lcd.setCursor(2,0);
 lcd.print(L"СМЕНА ПАРОЛЯ");
 };

//Если правильный пароль и был запрос на смену кода
 if (summ==pass && change==1){
 Serial.println("Pass ok go change pass");
 summ="";
 presses=0;
 wrong=0;
```

```
 dochange=1;
 key = keypad.getKey();
 squeaker(4, 1000, 50);
 lcd.clear();
 lcd.setCursor(0,0);
 lcd.print(L"СМЕНА РАЗРЕШЕНА");
 };

//При смене пароля - если введено полное количество знаков пароля, и он
//ошибочный, формируется звуковой сигнал:
 if (wrong==0 && presses==4 && change==1) {
 summ="";
 presses=0;
 wrong=wrong+1;
 Serial.println("Wrong_Pass");
 squeaker(4, 500, 50);
 lcd.clear();
 lcd.setCursor(0,0);
 lcd.print(L"НЕВЕРНЫЙ ПАРОЛЬ");
 };

//При смене пароля - если два раза ошибочный код, формируем звуковые
//сигналы
 if (wrong==1 && presses==4 && change==1) {
 summ="";
 presses=0;
 wrong=wrong+1;
 Serial.println("Attention!_2xWrong_Pass for change pas");
 squeaker(4, 500, 50);
 lcd.clear();
 lcd.setCursor(0,0);
 lcd.print(L"НЕВЕРНЫЙ ПАРОЛЬ");
 lcd.setCursor(3,1);
 lcd.print(L"ВТОРОЙ РАЗ");
 };

//При смене пароля - если три раза ошибочный код, выходим из режима
//смены пароля:
 if (wrong==2 && presses==4 && change==1){
```

```
 summ="";
 presses=0;
 wrong=0;
 change=0;
 Serial.println("Attention!_3xWrong_Pass for change pas - cancel change pass");
 squeaker(1, 500, 1000);
 denied();
 };

//Если введено полное количество знаков пароля, но он ошибочный:
 if (presses==4 && change==0){
 summ="";
 presses=0;
 wrong=wrong+1;
 Serial.println("Wrong_Pass");
 squeaker(1, 500, 300);
 lcd.clear();
 lcd.setCursor(0,0);
 lcd.print(L"НЕВЕРНЫЙ ПАРОЛЬ");
 delay(1000);
 wait();
 };

//Если два раза ошибочный пароль:
 if (wrong==2 && presses==0 && change==0) {
 Serial.println("Attention!_2xWrong_Pass");
 squeaker(1, 500, 500);
 lcd.clear();
 lcd.setCursor(0,0);
 lcd.print(L"НЕВЕРНЫЙ ПАРОЛЬ");
 lcd.setCursor(3,1);
 lcd.print(L"ВТОРОЙ РАЗ");
 delay(1000);
 wait();
 };

//Если три раза ошибочный пароль:
 if (wrong==3 && presses==0 && change==0) {
 summ="";
```

```
 presses=0;
 wrong=0;
 Serial.println("Attention!_3xWrong_Pass");
 squeaker(1, 500, 1000);
 denied();
 };
 break;

//Вторая часть блока для смены пароля, начало блока для смены пароля и его
//записи в энергонезависимую память:
 case 1:
 if (key=='#') //если введен символ # сбрасываем код
 {
 shetch=1;
 summ="";
 Serial.println("# is not an option, reset");
 squeaker(1, 500, 100);
 }
 else if (key=='*') //если введен символ * сбрасываем код
 {
 shetch=1;
 summ="";
 Serial.println("* is not an option, reset");
 squeaker(1, 500, 100);
 }
 else if (shetch==1 && (key)) //Меняем 1-ый символ пароля
 {
 Serial.print("NewPass_symbol_one "); Serial.println(key);
 squeaker(1, 2000, 100);
 shetch=2; //увеличиваем на единицу счет количества символов нового пароля
 EEPROM.write(1, key);

 Serial.println(char(EEPROM.read(1))); //проговариваем первый знак пароля в
порт
 }

 else if (shetch==2 && (key)) // Меняем 2-ой символ пароля
 {
 Serial.print("NewPass_symbol_two "); Serial.println(key);
```

```
 EEPROM.write(2, key);
 squeaker(1, 2000, 100);
 shetch=3; //увеличиваем на единицу счет количества символов нового пароля

 Serial.println(char(EEPROM.read(2))); //проговариваем второй знак пароля в
порт
 }

 else if (shetch==3 && (key)) // Меняем 3-ий символ пароля
 {
 Serial.print("NewPass_symbol_three "); Serial.println(key);
 EEPROM.write(3, key);
 squeaker(1, 2000, 100);
 shetch=4; //увеличиваем на единицу счет количества символов нового пароля

 Serial.println(char(EEPROM.read(3))); //проговариваем третий знак пароля в
порт
 }

 else if (shetch==4 && (key)) // Меняем 4-ый символ пароля
{
 Serial.print("NewPass_symbol_four "); Serial.println(key);
 EEPROM.write(4, key);
 squeaker(1, 2000, 100);

 Serial.println(char(EEPROM.read(4))); // проговариваем четвертый знак
пароля в порт

 String passnew = ""; // вводим переменную, содержащую новый введенный
пароль
 passnew = passnew+char(EEPROM.read(1));
 passnew = passnew+char(EEPROM.read(2));
 passnew = passnew+char(EEPROM.read(3));
 passnew = passnew+char(EEPROM.read(4));
 passRead();

 if (passnew==pass) // если новый пароль равен старому
 {
 shetch=1; // запрашиваем другой новый пароль
```

```
 summ="";
 Serial.println("NewPass equal old pass, Reset");
 squeaker(5, 600, 100);
 lcd.clear();
 lcd.setCursor(1,0);
 lcd.print(L"СТАРЫЙ ПАРОЛЬ");
 }
 else if (passnew=="0000") // если новый пароль равен комбинации для смены пароля
 {
 shetch=1; // запрашиваем другой новый пароль
 summ="";
 Serial.println("NewPass equal 0000, Reset");
 squeaker(5, 600, 100);
 lcd.clear();
 lcd.setCursor(1,0);
 lcd.print(L"СТАРЫЙ ПАРОЛЬ");
 }
else {

//Присваиваем паролю значения из энергонезависимой памяти:
 pass = "";
 pass = pass+char(EEPROM.read(1));
 pass = pass+char(EEPROM.read(2));
 pass = pass+char(EEPROM.read(3));
 pass = pass+char(EEPROM.read(4));

//Вывод в порт пароля для отладки:
 Serial.println("Pass read test: ");
 Serial.println(char(EEPROM.read(1))); //проговариваем первый знак пароля в порт
 Serial.println(char(EEPROM.read(2))); //проговариваем второй знак пароля в порт
 Serial.println(char(EEPROM.read(3))); //проговариваем третй знак пароля в порт
 Serial.println(char(EEPROM.read(4))); //проговариваем четвертый знак пароля в порт
 Serial.print("Pass "); Serial.println(pass); //проговариваем пароль в порт
```

```
//Выходим из цикла "case 1":
 dochange=0;
 change=0;
 presses=0;
 shetch=1;
 summ="";
 squeaker(5, 900, 100);
 lcd.clear();
 lcd.setCursor(2,0);
 lcd.print(L"НОВЫЙ ПАРОЛЬ");
 digitalWrite(GREEN_LED, HIGH);
 digitalWrite(RED_LED, LOW);
 delay(3000);
 digitalWrite(GREEN_LED, LOW);
 digitalWrite(RED_LED, HIGH);
 wait();
 break;
 }
 };
 //конец блока для смены пароля и его записи в энергонезависимую память
 }
}

//Очистка памяти:
 key_reset.update();
 if(key_reset.read() == HIGH) {
 if(resetTimer == 0) resetTimer = millis();
 else {
 if((millis()-resetTimer)/1000 > 5) {
 Serial.println(F("Launched memory cleaning"));
 squeaker(4, 1600, 300, 200);
 wdt_disable();
 for(int i=5; i<=EEPROM.length(); i++) {
 EEPROM.write(i, 0);
 if(!(i%50)) Serial.println(F("#")); else Serial.print(F("#"));
 }
 Serial.println(F("\r\nMemory cleaning is completed\r\n"));
 delay(1000);
 resetFunc();
```

```
 }
 }
 }
 else if(resetTimer != 0) resetTimer = 0;

//Автоматическое закрытие двери через 5 секунд:
 if(openTimer != 0) {
 if(millis()/1000 - openTimer > 5) {
 openTimer = 0;
 digitalWrite(PIN_RELAY, HIGH);
 digitalWrite(GREEN_LED, LOW);
 digitalWrite(RED_LED, HIGH);
 wait();
 Serial.println("* closed lock\r\n");
 }
 }

//Если ключ отсутствует или не читается, не выполняем дальнейший код:
 if(!mfrc522.PICC_IsNewCardPresent()) {

//Очистка таймера входа в режим программирования, в случае если
//считыватель свободен:
 if(modeTimer != 0) {
 if(++modeClean > 5) modeTimer = modeClean = 0;
 }
 return;
 }
 if(!mfrc522.PICC_ReadCardSerial()) return;

//Останавливаем режим очистки:
 modeClean = 0;

//Создание мастер-ключа:
 if(keys_count == 0) {
 for(byte i=0; i<4; i++) EEPROM.write(i+5, mfrc522.uid.uidByte[i]);
 EEPROM.write(0, keys_count = 1);
 uidPrint(F("master key is created"));
 digitalWrite(PIN_RELAY, HIGH);
 keysRead();
```

```
 squeaker(8, 1200, 100, 100);
 delay(2000);
 return;
 }

//Проверка ключа на соответствие:
 bool access = false;
 bool master = false;
 for(byte i=0; i<keys_count; i++) {
 for(byte b=0; b<4; b++) {
 if(keyss[i][b] != mfrc522.uid.uidByte[b]) break;
 if(b == 3) {
 access = true;
 if(i == 0) master = true;
 // Останавливаем проверку
 i = keys_count;
 }
 }
 }

//Контроль доступа:
 if(access and !mode and !master)  {
 // Доступ разрешен
 openTimer = millis()/1000;
 digitalWrite(PIN_RELAY, LOW);
 squeaker(2, 3500, 200, 200);
 allow();
 }

 else if(!access and !mode and !master) {
 // Доступ запрещен
 squeaker(1, 500, 1000);
 denied();
 }


//Режим программирования - запись ключа:

 if(access and mode and !master) {
```

```
 // Попытка записи существующего ключа
 uidPrint(F("error: key already exists in eeprom"));
 squeaker(2, 500, 300);
 lcd.clear();
 lcd.setCursor(1,0);
 lcd.print(L"КАРТА ИЗВЕСТНА");
 delay(2000);
 wait();
 }
 else if(!access and mode and !master) {
 // Записываем новый ключ
 // Максимум 255 ключей (с учетом первого байта)
 if(keys_count < 255) {
 for(byte i=0; i<4; i++) EEPROM.write(5 + keys_count*4 + i,
mfrc522.uid.uidByte[i]);
 EEPROM.write(0, ++keys_count);
 uidPrint(F("add key in eeprom"));
 keysRead();
 lcd.clear();
 lcd.setCursor(4,0);
 lcd.print(L"ПРИЛОЖИТЕ");
 lcd.setCursor(2,0);
 lcd.print(L"НОВУЮ КАРТУ");
 squeaker(2, 2200, 200, 200);
 }
 else // нет памяти для записи
 {
 uidPrint(F("error: not enough memory for recording key!"));
 squeaker(2, 500, 300);
 }
 delay(2000);
 wait();
 }

//Работа с мастер-ключом:
 else if(access and master) {
 // Мастер ключ в обычном режиме
 if(modeTimer == 0) {
 modeTimer = millis()/1000;
```

```
 if(!mode) {
 openTimer = millis()/1000;
 digitalWrite(PIN_RELAY, LOW);
 // Сигнал о наличии мастер ключа в обычном режиме
 uidPrint(F("MASTER KEY"));
 squeaker(2, 3200, 200, 200);
 allow();
 }
 }
 else
 {
 if(millis()/1000 - modeTimer > modeProgTime and modeTimer != 0)
 {
 modeTimer = 0;
 if((mode = !mode) == true)
 {

//Вход в режим программирования:
 digitalWrite(PIN_RELAY, LOW);
 uidPrint(F("MASTER PROGRAMMING MODE ON"));
 squeaker(4, 1200, 200, 200);
 }
 else {
 // Выход из режима программирования
 digitalWrite(PIN_RELAY, HIGH);
 uidPrint(F("MASTER PROGRAMMING MODE OFF"));
 squeaker(4, 2200, 200, 200);
 }
 }
 delay(5000);
 wait();
 }
 // Мастер ключ удерживается у считывателя на 5 секунд
 }
}

//Функция, срабатывающая при верном пароле/ключе:
void allow()
{
```

```
 Serial.println("Access accept!"); //доступ получен
 digitalWrite(GREEN_LED, HIGH);
 digitalWrite(RED_LED, LOW);
 lcd.clear();
 lcd.setCursor(0,0);
 lcd.print(L"ДОСТУП РАЗРЕШЕН");
 summ="";
 presses = 0;
 delay(5000);
 digitalWrite(GREEN_LED, LOW);
 digitalWrite(RED_LED, HIGH);
 wait();
}

//Функция, срабатывающая при неправильном пароле/ключе:
void denied()
{
 Serial.println("Access denied!"); //доступ закрыт
 digitalWrite(RED_LED, HIGH);
 lcd.clear();
 lcd.setCursor(0,0);
 lcd.print(L"ДОСТУП ЗАПРЕЩЕН");
 summ="";
 presses = 0;
 delay(5000);
 wait();
}
//Функция режима ожидания:
void wait()
{
 lcd.clear();
 lcd.setCursor(4,0);
 lcd.print(L"ОЖИДАНИЕ");
 lcd.setCursor(4,1);
 lcd.print(L"ДЕЙСТВИЯ");
}
```

Приложение Б

Инструкция по работе с устройством контроля доступа в помещение

Для начала работы с устройством контроля доступа (электронным замком), необходимо подключить микроконтроллер Arduino UNO R3 к персональному компьютеру через USB-кабель для того, чтобы наблюдать отладочную информацию, поступающую от устройства. При подключении питания загорятся красные светодиоды у реле и RGB-светодиода, а на экране дисплея LCD1602 появится сообщение – «Ожидание действия». Если EEPROM память микроконтроллера очищена и не содержит никаких данных, кроме 0, устройство попросит ввести первоначальный мастер-ключ, на дисплее отобразится сообщение «Введите мастер ключ», контакты реле будут разомкнуты – загорится зеленый светодиод реле. Для установки первого мастер-ключа необходимо поднести RFID-метку к считывателю RC522. После этого система выдаст отладочную информацию об идентификационном номере ключа в последовательный порт компьютера, контакты реле замкнутся, зеленый светодиод реле погаснет, несколько раз прозвенит пьезодинамик, на дисплее будет отображаться сообщение «Ожидание действия» до тех пор, пока не будет совершено какое-либо действие со стороны пользователя. Для ввода пароля необходимо использовать матричную клавиатуру. При нажатии на любую кнопку пьезодинамик будет выдавать звуковой сигнал, а на дисплее отобразится введенное значение в виде «*». Если ввести неправильный пароль, то на экране дисплея отобразится сообщение – «Неверный пароль». Если пароль введен неправильно три раза, прозвучит характерный звуковой сигнал и на дисплее появится сообщение «Доступ запрещен», и в течение пяти секунд микроконтроллер не будет реагировать на любые действия. Если ввести правильную кодовую комбинацию, загорится зеленый светодиод, контакты реле разомкнутся, на дисплее появится сообщение «Доступ разрешен». Если ввести комбинацию «0000», станет возможным смена пароля. Для этого

необходимо за три попытки ввести прежнюю комбинацию, а затем новый пароль, иначе устройство выйдет из режима смены пароля и не будет доступно в течение пяти секунд.

Если к считывателю RC522 поднести мастер-ключ, то система сработает аналогично случаю с верным паролем. При удержании мастер-ключа у считывателя в течение определенного отрезка времени станет доступным режим программирования новых ключей, контакты реле будут разомкнуты, моделируя открытое состояние двери. В режиме программирования можно добавлять новые ключи. После этого мастер-ключ удерживается еще определенный промежуток времени, за которым следует выход устройства из режима программирования. На неизвестные ключи система срабатывает аналогично вводу неправильного пароля три раза.

Для сброса всех известных ключей необходимо удерживать специальную кнопку сброса в течение пяти секунд. После этого память микроконтроллера будет очищена, за исключением кодовой комбинации, устройство потребует установить первый мастер-ключ. Сброс необходим в том случае, если утерян прежний мастер-ключ. Пароль и все известные ключи отображаются через последовательный порт подключения к компьютеру при использовании функции «Монитор порта» в программной среде Arduino IDE.