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 Аппаратно-программный комплекс мини-полигона. Система парковки модели автомобиля

Студент(ка) С.В. Терехин _____
(И.О. Фамилия) (личная подпись)

Руководитель А.К. Кудинов
(ученая степень, звание, И.О. Фамилия)

Консультант А.В. Кириллова
(И.О. Фамилия)

Тольятти 2021

Аннотация

Объем бакалаврской работы 60 стр., 42 рисунков, 20 источников, 2 приложений.

В данной бакалаврской работе была создана «автоматизированная парковка», которая имеет тип башня и будет располагаться под землей.

Цель работы: создание автоматизированной парковки, в которую входят следующие функции: вызов автомобиля с любого этажа, система доступа на базе RFID, система пожарной безопасности, защита от наводнений.

Задачами работы являются:

1. Анализ исходных данных и известных решений;
2. Выбор необходимых компонентов;
3. Разработка структурной и принципиальной схем;
4. Разработка алгоритма передвижения и программы;

Работа состоит из четырех глав, в которых решены упомянутые задачи. Для оформления всех диаграмм и чертежей использовался пакет КОМПАС-3D V16.

Для разработки программы, управляющей логикой работы, использовалась среда разработки Arduino IDE.

Областью применения «автоматизированной парковки» могут быть все города и населенные пункты мира, а также торговые центры, больницы, полицейские участки, пожарные службы и др.

Abstract

The title of the graduation work is Hardware and software complex of the testing area. Car model parking system.

The senior thesis consists of an introduction, 3 parts, a conclusion, tables, the list of references including foreign sources and the graphic part on 6 A1 sheets.

The key issue of the graduation project is the development and design of automated parking. We are addressing the problem of streets crowded with cars, which greatly complicates traffic in the lanes and courtyards of residential complexes. We also touch upon the problem of cars safety when leaving vehicles outside in residential neighborhoods which subjects them to early corrosion.

The aim of the work is the development of an automatic security system and the best automated parking.

The graduation work may be divided into several logically connected parts which are analysis of existing solution; justification of the need to automated parking system; selection of equipment and sensors; software development; technological and design solutions; an example of foreign experience in the integration of automatic parking system.

Finally, we present various types of automated parking lots that are used all over the world. Their use made it possible to significantly reduce the number of cars on huge areas, as well as significantly improve the safety of cars.

In conclusion we'd like to stress this work is relevant in solving the problem with a huge number of homeless cars, which clog up the space on the streets, and will also reduce in times the allocation of huge territories for ordinary parking, which is important for the Russian Federation.

Содержание

Введение

В современных реалиях мира почти каждый человек имеет свое транспортное средство в виде автомобиля. Но с каждым годом людей в мире становится все больше соответственно спрос на автомобили тоже растет и как следствие этому растет количество автомобилей на улицах наших городов. Уже сейчас население автолюбителей сталкивается с проблемой парковок и мест для хранения своего автомобиля. На примере нашего города видно, что все улицы полностью забиты автомобилями. Это создает огромный дискомфорт в перемещении. Но есть современное решение и это автоматизированные парковочные места. Автоматизация в современном мире занимает одно из главенствующих мест. Она имеет применение почти во всех видах деятельности. Автоматизированный паркинг не исключение. Если заменить обычные парковочные места на автоматизированные, то это сэкономит огромное количество пространства, которое было захламлено автомобилями, расставленными в хаотичном порядке. Такое решение сократит и количество дорожно-транспортных происшествий на парковочных местах, что сделает паркинг более безопасным. Так же оставлять свой автомобиль в подготовленном для этого месте в разы увеличит срок его эксплуатации, что несравненно лучше, чем оставлять автомобиль на открытом пространстве в любое время года или любую погоду. Данное решение так же поможет сохранить исторические места старой архитектуры, которые являются культурным достоянием.

В России уже используется автоматизированный паркинг, но в малом количестве и в основном в мегаполисах таких как Москва и Санкт Петербург.

1 Состояние вопроса

1.1 Формулирование актуальности, цели и задач проекта

Автоматизирование парковок на данный момент времени являются особо актуальным направлением. Системы, которые разрабатываются в данном направлении, позволяют в скором времени перейти на автоматизированный паркинг большинству стран, а в последствии и всему миру. Что приведет к более рациональному решению по размещению жилых комплексов, торговых центров, больниц и других мест массового скопления людей.

Данное направление активно развивается в странах Европы и Восточной Азии. Так как они обладают огромным индустриальным потенциалом и имеют довольно высокую плотность населения, но и Россия с ее редким расположением городов нуждается в данном решении. Основываясь на нашем родном городе Тольятти, который имеет огромные проблемы с парковочными местами в спальных районах.

Целью работы является разработка системы и средств парковки и парковочного оборудования.

Задачами работы являются:

- Разработка структурной схемы;
- Выбор необходимых компонентов;
- Разработка электрической принципиальной схемы;
- Разработка алгоритма работы;
- Разработка программной части устройства;
- Графическая модель устройства;

1.2 Анализ исходных данных и известных решений

«Как правило, автоматизированная парковка, состоящая из нескольких уровней – это мощная конструкция, выполняемая с использованием железа, бетона, а также стекла. Она оборудуется специальным механизмом, при помощи которого автомобиль поднимается и устанавливается на свободное место, причем все происходит без присутствия человека.

Для того, чтобы разместить такую парковку, достаточно любого незаселенного участка с небольшими размерами. К слову, даже в густонаселенном городе, имеющем точечную застройку, найти подходящий участок не составит значительной проблемы. Обычно, каркас паркинга выполняется металлическим, а во внешней отделке используют качественные и красивые материалы.

В качестве начинки автопарковок применяется комплекс соединенных между собой сканирующих датчиков, а также движения. Также размещается механизм, осуществляющий транспортировку машин, есть лифта и разнообразная компьютерная техника.

Исходя из способа, по которому происходит размещение автомобилей, паркинги могут быть горизонтальными либо вертикальными. Впрочем, для потребителя это отличие большого значения не имеет, так как разница между такими автопарковками заключается лишь в позициях расположения автомобилей.

Автоматизированные парковки, обладающие несколькими уровнями, весьма популярны в странах с большой плотностью населения. Так, если в Германии либо США развернулось активное строительство паркингов, предназначенных именно для автомобилей, то в Японии, а также Корее, наблюдается повышенный спрос даже на стоянки, рассчитанные на

велосипеды.

Существует множество типов АПС и сейчас вы увидите несколько самых распространенных из них.

Первый тип -это «Башни»

Башенная (высотная) парковка предназначена для размещения легковых автомобилей типа седан или джип. Она состоит из центрального вертикального грузового подъемника лифтового типа с манипулятором горизонтального перемещения, а также ячеек хранения автомобилей на парковочных ярусах. Ячейки для хранения автомобилей располагаются, как правило, с двух или четырех сторон от лифтовой шахты. Основание парковки занимает площадь не более, чем уличное размещение 3-5 автомобилей. А благодаря своей компактности и этажности одна типовая башенная парковка позволяет на таком же земельном участке разместить (в зависимости от числа уровней) несколько десятков автомобилей.

Имея низкий уровень шума и вибраций, башенные парковки могут устанавливаться в густо застроенных районах, в местах, где автомобили паркуются или покидают стоянку в ночное время (многоэтажные дома, отели, бизнес — центры, административные здания, больницы, торговые центры и др.). В стесненных условиях исторических центров башенные паркинги могут размещаться в подземном городском пространстве или встраиваться в здания, в т.ч с сохранением архитектурных фасадов. Широкую популярность в Европе получили презентационные (демонстрационные) башенные паркинги, используемые для хранения автомобилей брендовых автодилеров и музеев ретро автомобилей.» [5]

Они идеально подойдут для районов массовой застройки, пример показан на рисунке 1.

Рисунок 1 – башенная парковка для районов массовой застройки

Так же их используют в исторических кварталах, ибо в основном места для парковки в давние времена не выделяли. Пример можно увидеть на рисунке 2.

Рисунок 2 – башенные парковки для исторических кварталов

Некоторые фирмы используют данные парковки для демонстраций новых моделей. Это можно увидеть на рисунке 3.

Рисунок 3 – Демонстрационная башенная парковка

Некоторые крупные застройщики устанавливают такие парковки в своих жилых комплексах премиум класса, как показано на рисунке 4.

Рисунок 4 – Башенная парковка для ЖК премиум класса

«В сумме всего башенная парковка обеспечивает возможность размещения автомобилей на минимальных площадях, при сохранении экологических и других условий комфортного жизненного пространства, позволяет эффективно использовать электроэнергию, совмещает в себе красивый вид и низкий уровень шумов. Увеличение емкости паркинга башенного типа достигается блокированием отдельных башенных парковок (секций), а также использованием и подземного пространства земельного участка.

Второй тип - это парковка типа «Стеллаж»

Стеллажная парковка классического типа является одной из наиболее распространенных многоярусных автоматизированных парковочных систем (АПС). В данном типе стеллажной парковки автомобиль не покидает паллету (поддон) на всех этапах перемещения, в том числе и в ячейке хранения. Один

штабелер (3-х координатный манипулятор) обслуживает подачу/выдачу до 40-50 автомобилей. Этим достигается высокая скорость приема/подачи автомобиля за время, не более 60-90 сек.

Особенности такого типа

Применяются для хранения большинства классов легковых автомобилей (седан, кроссовер, внедорожник).

Возможно увеличение численности хранящихся автомобилей за счет объединения парковочных модулей (блоков).

Идеально подходит для зданий прямоугольной формы в надземном, подземном или комбинированном исполнении.

Классические стеллажные парковки могут иметь по высоте до 10 и более ярусов (уровней). Число парковочных мест вдоль фасада здания практически неограниченно, однако следует учитывать время выдачи автомобиля и противопожарные требования по вместимости машин в одном блоке стеллажной парковки. Присоединение дополнительных блоков стеллажной парковки, возможно при монтаже автономных 3-координатных манипуляторов и оборудовании приемных помещений (боксов).

Применяются в основном там, где крупные жилые кварталы и торговые комплексы, аэропорты, вокзалы, транспортно-пересадочные узлы, спортивные и зрелищные объекты. Подземные стеллажные парковки можно эффективно применять в условиях плотной городской застройки и дефицита свободных площадей земельных участков.

В центральных районах городов стеллажные парковки удачно встраиваются в аварийные (морально устаревшие) здания, которые приобретают новый высоко востребованный парковочный функционал, но при этом сохраняют исторические фасады и архитектурный облик среды.»

[5] Данный тип парковок показан на рисунке 5.

Рисунок 5 – Парковка типа стеллаж

«Третий тип - это парковка типа «Шаттл»

В стеллажной парковке Шаттл используются один или несколько лифтов (по числу приемных помещений, которые доставляют автомобиль к одному из ярусов парковочных стеллажей, где осуществляется его перегрузка на горизонтально двигающийся роботизированный транспортер (шаттл), доставляющий автомобиль на свободное (закрепленное) парковочное место этого яруса стеллажа. Скоординированные действия лифтов и транспортеров напоминают работу челноков в промышленных машинах, что и дало название «Шаттл» данному виду автоматизированных парковок. Один лифт обслуживают несколько шаттлов (2-х координатных

манипуляторов), по числу ярусов стеллажной парковки. Перегрузка автомобиля может осуществляться с использованием паллет (поддонов), или специальных платформ с независимыми тележками. Возможна подача шаттла к разным лифтам.

Особенности

Применяются для хранения легковых автомобилей седан, кроссовер, внедорожник. В приемном помещении автомобили специальными сенсорами и датчиками оцениваются по своим габаритам и весу. Затем система управления выдает сигнал, на какой стеллаж, и в какую парковочную ячейку этот автомобиль направить для постановки.

Обладает большой парковочной емкостью и подходит для средних и крупных объектов жилой и коммерческой недвижимости, а также для городских общественных территорий. Возможно многократное увеличение численности хранящихся автомобилей за счет объединения парковочных модулей (блоков).

Программная оптимизация алгоритмов одновременной работы лифтов и шаттлов позволяет обеспечить время постановки / выдачи (90-120 сек) вне зависимости от количества хранящихся автомобилей.

Наличие нескольких высокоскоростных лифтов и шаттлов дают возможностью одновременного въезда/ выезда с разных уровней надземного и подземного паркования.

Возможно инвариантное обустройство несущего каркаса. При применении самонесущего металлокаркаса в парковках Шаттл применяется (как правило) перемещение автомобиля на перегружаемых поддонах. При монолитном исполнении здания возможно применение роботизированных тележек.

Применяются в основном там, где крупные жилые и торговые комплексы, вокзалы и транспортно-пересадочные узлы, больницы, перехватывающие парковки, объекты коммерческой, деловой и социальной недвижимости. Стеллажные парковки челночного типа эффективно применяют в условиях плотной городской застройки и дефицита свободных площадей земельных участков. При этом большая часть автоматизированного паркинга находится под землей (под городскими магистралями, площадями, рекреационными зонами), а на поверхности размещается только небольшие по габаритам помещения (боксы) для въезда/выезда автомобилей.» [5] Данный тип парковки показан на рисунке 6.

Рисунок 6 – Парковка типа шаттл

И последний тип, который я хотел бы показать – это парковка типа «Оптима»

«В автоматизированной парковке Оптима используются два лифта,

расположенных в конечных точках стеллажей, представляющих собой горизонтальные кассетные системы. Для подачи и приема автомобиля, осуществляется одновременное перемещение всех автомобилей, расположенных на одном уровне хранения (аналогично горизонтальному конвейеру), а также их перемещение между ярусами.

Первый (основной) лифт доставляет автомобиль из приемной камеры к стеллажу, на котором возможна самая быстрая подача свободной ячейки. Перемещение автомобилей между уровнями осуществляет второй (меж ярусный перегрузочный) лифт.

АПС Оптима – является одной из самых современных и высокотехнологичных. В такой парковке обеспечиваются очень высокие показатели коэффициента полезного использования объема помещения, и все автомобили размещаются максимально компактно.

Особенности

Применяются для хранения легковых автомобилей классов седан, кроссовер, джип.

Каждый автономный модуль (блок) парковки обладает средней парковочной емкостью – до 35-45 автомобилей. Возможно увеличение численности хранящихся автомобилей за счет объединения парковочных модулей (блоков).

Программная оптимизация алгоритмов одновременной работы кассетных стеллажей, основного (подающего) лифта и меж ярусного перегрузочного лифта позволяет обеспечить время постановки/выдачи 1,5-2 минуты.

Имеется возможность въезда/ выезда с разных уровней надземного и подземного паркования

Отсутствие необходимости в горизонтальных проемах вдоль стеллажей для работы подъемно-транспортного оборудования позволяет размещать парковку Оптима в помещениях со стесненными условиями.

Применяются в основном там, где жилые комплексы, объекты деловой и коммерческой недвижимости, имеющие средние и малые парковочные потребности.

АПС Оптима, позволяет наиболее эффективно использовать подземное пространство капитальных зданий (неиспользуемые подвалы). В таких случаях только небольшой наземный приемный бокс оборудуется на минимальном участке территории. Уровни хранения в минимальном количестве (2-3) – монтируются в подвале, шириной от 6,0 м. Парковки Оптима удачно встраиваются и в аварийные (морально устаревшие) здания, которые приобретают новый высоко востребованный парковочный функционал, но при этом сохраняют исторические фасады и архитектурный облик среды. Такая недвижимость обычно более доходна и высоко ценится на рынке.» [5] Увидеть данный тип парковки можно взглянув на рисунок 7.

Рисунок 7 – Парковка типа оптима

В исходных данных мы будем использовать микроконтроллер Arduino Mega 2560.

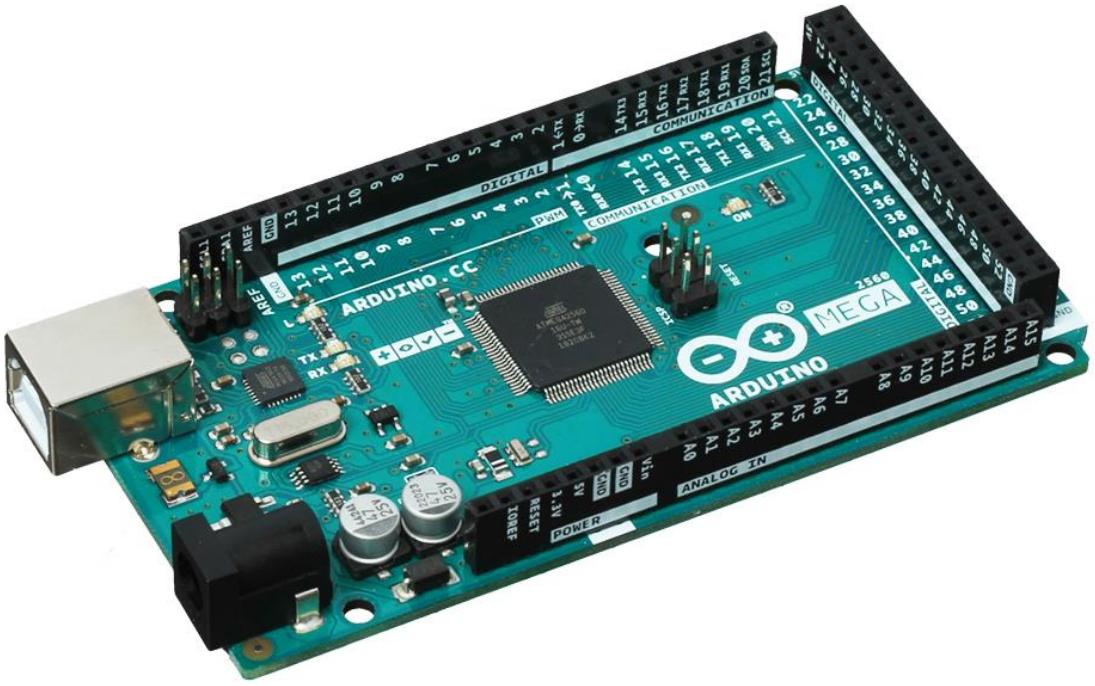

Рисунок 8 – Arduino Mega 2560

Arduino Mega 2560 это флагманская платформа, на которой будет реализован данный проект. Плата рассчитана на комфортное использование так как имеет 54 цифровых входов/выходов, 16 входов аналогового типа, также присутствует usb-разъем для программирования устройства, разъем для внешнего питания и кнопка, которая осуществляет сброс.

Рассмотрим подробнее структуру данного устройства, что показана на рисунке 9.

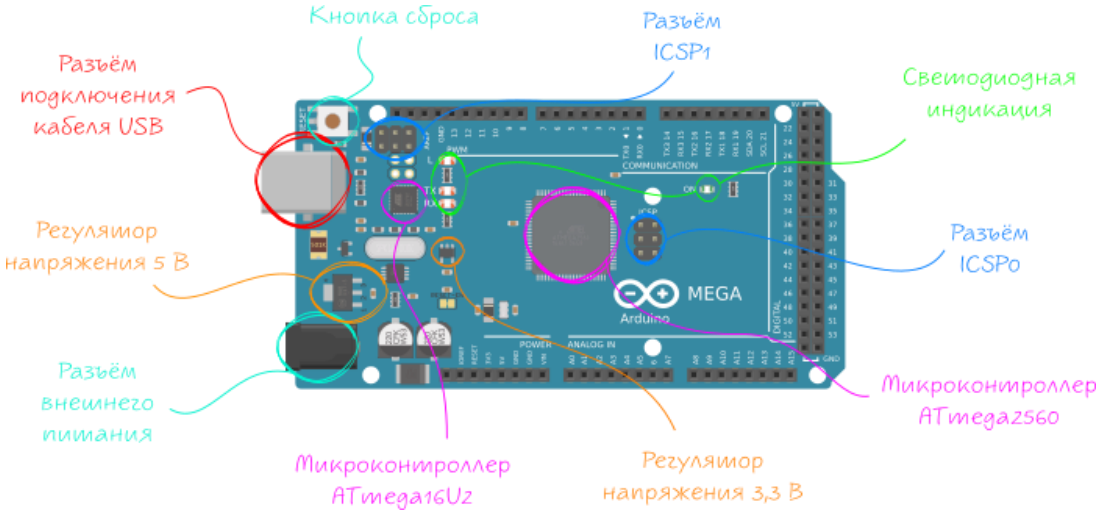

Рисунок 9 – Структура Arduino Mega 2560

Начнем с usb–разъёма, который находится в левой верхней части платформы. Он представляет собой обрамленный металлом разъем для подключения с помощью специального провода к персональному компьютеру для дальнейшего программирования.

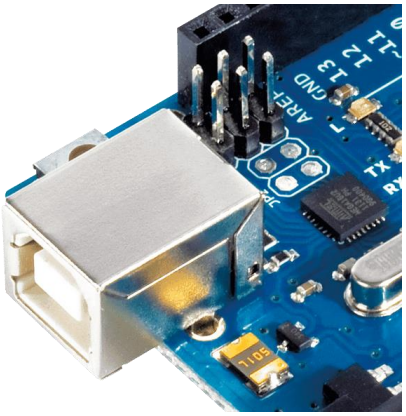

Рисунок 10 – usb-разъем

Далее разберем регуляторы напряжения 5 В и 3,3 В, которые также расположены на данном устройстве и представляют собой две линейные понижающие микросхемы регулирования с выходами в 5 и 3,3 В. Регулятор 5 В, имеет цель обеспечивать питанием сам микроконтроллер, максимальный ток выхода на нем равен 800 мА. В свою очередь, регулятор 3,3 В, имеет выходное напряжение 3,3 В. Он предназначен для питания пина 3v3, и

максимальный ток на выходе будет составлять 150 мА. Оба регулятора представлены на рисунке 9, из которого можно понять о местоположении их на плате и увидеть внешний вид микросхем.

Далее рассмотри сердце нашей платформы, которое представляет собой микроконтроллер 8-битный, что находится в семействе AVR – ATmega2560 и обладает тактовой чистотой в 15 МГц. В самом контроллере мы имеем 256 КБ Flash-памяти, которая необходима для хранения прошивки в целом. Также имеет 8 КБ оперативной памяти SRAM статической памяти с произвольным доступом и 4 КБ памяти, которая является энергозависимой, EEPROM память для хранения данных.

В свою очередь микроконтроллер ATmega16U2 необходим нам для связи микроконтроллера ATmega2560 с USB-портом нашего устройства, чтобы при подключении к персональному компьютеру он определялся как COM-порт и в дальнейшем программировался. Оба микроконтроллера можно увидеть на рисунке 9.

Светодиодная индикация на данном контроллере представляет собой 4 светодиода: RX, TX, L и ON. Рассмотрим каждый из них. Светодиоды RX и TX предназначены для демонстрации того, что контроллер имеет связь с компьютером и о самом совершении обмена данными. При передаче данных с компьютера в контроллер данные светодиоды мигают и тем самым оповещают на о том, что передача данных проходит успешно.

Следующий светодиод L предназначен для 13 пина контроллера и при горении оповещает нас, что на данном пине находится высокий уровень напряжения и тухнет при низком, что помогает пользователю понять идет передача сигнала или нет.

Светодиод ON необходим для оповещения пользователя о том, что на платформу в целом поступает питание и она находится в рабочем состоянии

и готова к эксплуатации. Все светодиоды представлены на рисунке 11.

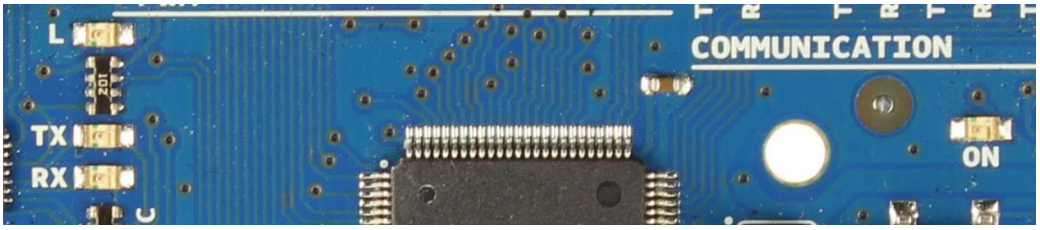

Рисунок 11 – Светодиоды RX, TX, L, ON

Следующим пунктом рассмотрим разъем для внешнего питания контроллера, он распложен ниже USB-разъема, имеет форму цилиндра и представляет собой разъем нефиксированного взаимодействия. Внешнее питание на контроллер для работы составляет от 7 до 8 В. Он представлен на рисунке 12.

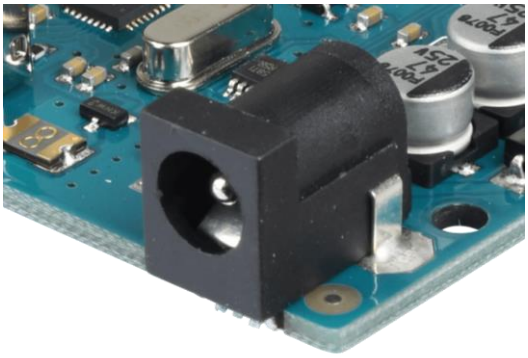

Рисунок 12 – Разъем для внешнего питания

Кнопка сброса представляет собой красную кнопку в правой части контроллера, функционал которой перезапуск системы контроллера. Она подобна кнопке reset на любом персональном компьютере. Она представлена на рисунке 13.

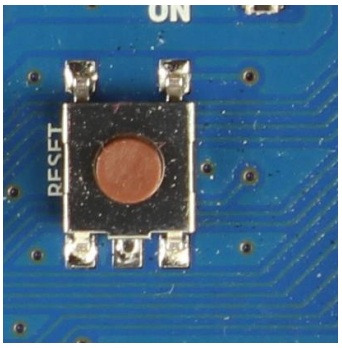

Рисунок 13 – Кнопка сброса RESET

Разъем ICSP необходим для внутрисхемного программирования нашего устройства, то есть самого микроконтроллера ATmega2560. А с применением библиотеки SPI нам будет доступна связь с платами расширения по интерфейсу SPI для выводов. Линии же SPI имеют выход на 6-контактном разъеме и продублированы на пинах цифрового типа: 50(MISO), 51(MOSI), 52(SCK) и 53(SS). Внешний вид данного разъема можно увидеть на рисунке 14.

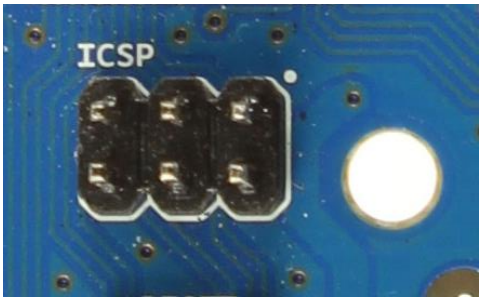

Рисунок 14 – ICSP разъем

Похожий вид имеет разъем ICSP1, но он в отличии от первого предназначен для внутрисхемного программирования уже микроконтроллера ATmega16U2. Он имеет отличное от первого местоположение и маркировку на плате, что показано на рисунке 15.

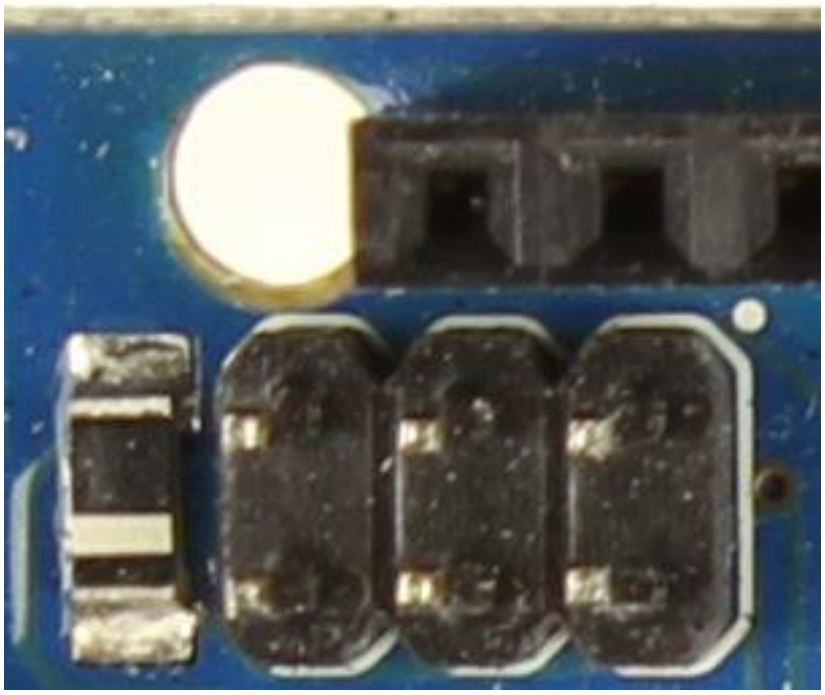

Рисунок 15 – ICSP1 разъем

Теперь поговорим о распиновке данного устройства. Контроллер имеет большое количество пинов, которые расположены по краям контроллера и образуют не замыкающуюся рамку так как в левой части контроллера расположены разъемы питания и передачи данных. Все пины представлены на рисунке 16.

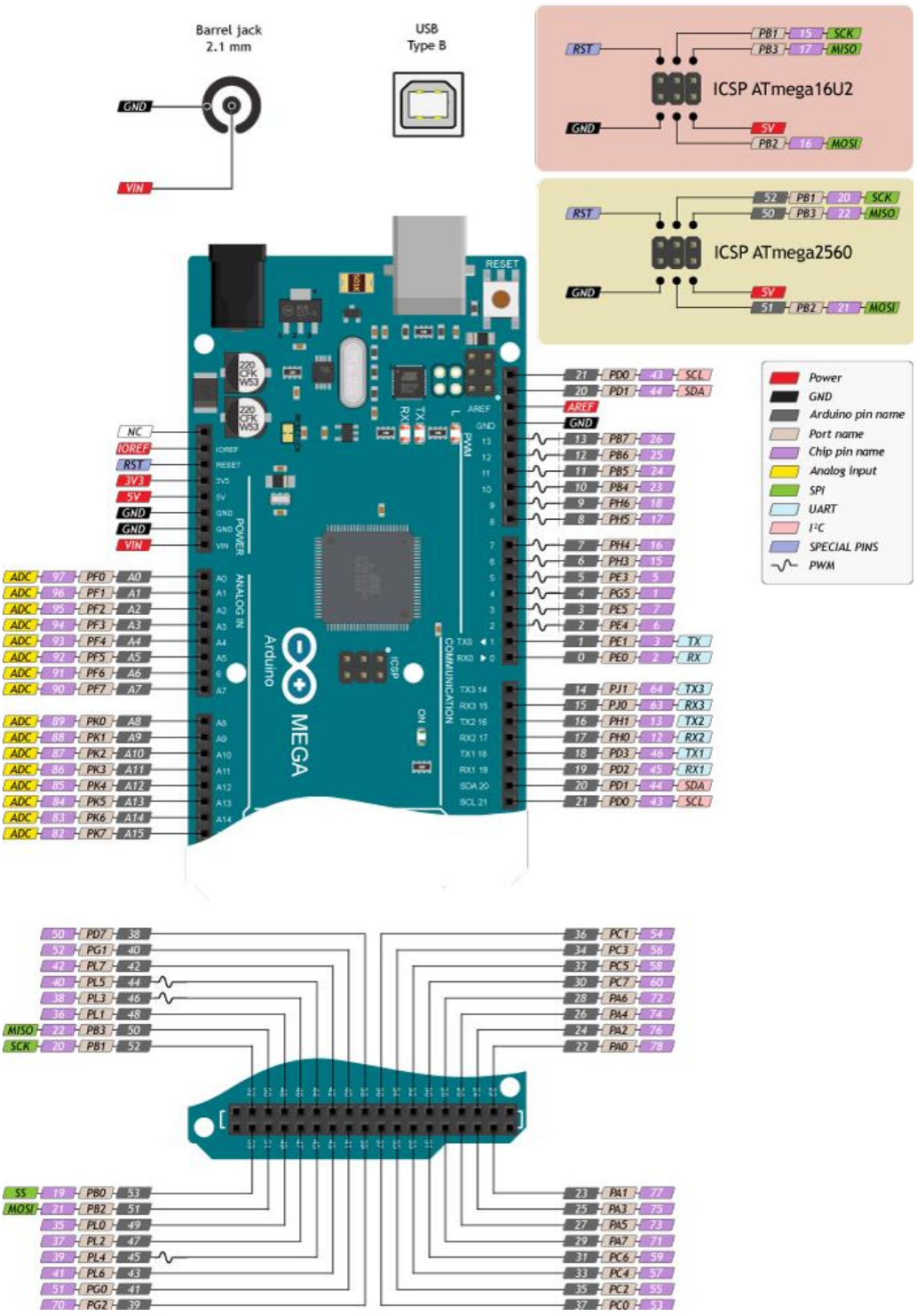

Рисунок 16 – Распиновка ATmega2560

Для начала рассмотрим важнейшие пины, без которых устройство просто не будет работать. Пины питания

- VIN: Входной пин, который предназначен для подключения внешнего источника питания, что должен иметь напряжение строго от 7 до 12 В. В таком случае через контакт можно потреблять напряжение, если устройство будет запитано через данный внешний разъем питания.
- 5V: Этот пин предназначен для ранее описанного регулятора напряжения с выходом 5 В, у которого максимальный выходной ток равен 800 мА.
- 3,3V: Пин ранее описанного регулятора напряжения, который имеет выходное напряжение 3,3 В и максимальный выходной ток 150 мА.
- GND: Данные пины служат для выводов на землю
- IOREF: Данный контакт может предоставить платам расширения информацию о напряжении работы платы микроконтроллера и в зависимости от того какое напряжение требуется, имеет возможность переключения на другой источник питания или задействовать преобразователи уровней.
- AREF: Данный пин необходим для возможного подключения внешнего напряжения, которое будет являться опорным, для АЦП если от него происходят аналоговые измерения, которые задействуются с помощью функции analogReference() с параметрами «EXTERNAL».

Далее рассмотрим порты ввода/вывода

- Цифровые входы/выходы: Это пины с 0 по 53. Логическая единица на таких пинах равна напряжению 5 В, а у логического нуля 0 В соответственно. Максимальный ток выхода на данных пинах составляет 40 мА. К каждому из контактов подключен подтягивающий резистор, который по умолчанию не задействован, но если того требует техническое решение, то их можно включить с помощью программы.

- ШИМ: Это пины со 2 по 13, а также с 44 по 46. Данные порты позволяют нам выводить аналоговый сигнал в виде ШИМ-сигнала, но разрядность ШИМ не меняется и всегда соответствует 8 битам.
- АЦП: Пины с A0 по A16. Данные пины делают возможным преобразование аналогового напряжения в цифровой вид, что необходимо для некоторых проектов. Разрядность данных АЦП не меняется и всегда держит свои 10 бит. Диапазон входного напряжения на таких пинах от 0 до 5 В. Данные пины работают строго с таким напряжением иначе вы рискуете повредить контроллер.
- TWI/I^2C: Это пины 20(SDA) и 21(SCL). Они необходимы для связи с периферийными устройствами при помощи интерфейса I^2C, а для работы данной функции необходимо использовать библиотеку Wire.
- SPI: Это пины 50(MISO), 51(MOSI), 52(SCK), 53(SS). Так же используются для общения с периферийными устройствами, но уже с использованием другой библиотеки, а именно библиотеки SPI.
- UART: Данные пины 0(RX) и 1(TX), 19(RX1) и 18(TX1), 17(RX2) и 16(TX2), 15(RX3) и 14(TX3). Они используются для коммуникации нашей платы и любого персонального компьютера или другого устройства с помощью последовательного интерфейса, а выводы 0(RX) и 1(TX) соединены с выводами от микроконтроллера, о котором мы ранее говорили ATmega16U2, вместе они образуют USB-UART преобразователь, который также предназначен для работы с последовательным интерфейсом, но для него необходима библиотека Serial.

Выводы по разделу

Подведем следующие выводы:

- автоматизация парковочных мест является одной из самых актуальных тем на данный момент времени;
- Это решение актуальное и имеет довольно много готовых решений и проектов, но достаточно крупного масштаба;
- Arduino ATmega2560 является очень удобной платформой для выполнения огромного числа различных проектов, что делает ее наиболее востребованной в сфере автоматизации. Именно поэтому для выпускной квалификационной работы выбрана именно она;
- технические характеристики данной платформы более чем удовлетворяют нашим задумкам для автоматизированной парковки и ее системы безопасности.

2. Аппаратная часть

2.1 Разработка структурной схемы

Структурная схема была разработана на основе технического задания на бакалаврскую работу.

Для выполнения задания потребуется:

- Микроконтроллер, который будет управлять платформой;
- Тактовые кнопки вызова, которые будут вызывать автомобиль;
- Датчик пожара, что будет высылать сигналы о возгорании;
- Датчик протечки, который будет следить за тем, чтобы на парковке не было воды;
- Электродвигатель для передвижения платформы;
- Электродвигатель для открытия крана пожарной системы;
- Электронасос для откачки воды;
- RFID модуль, который будет считывать карты доступа;
- Аккумуляторная батарея для питания платформы.

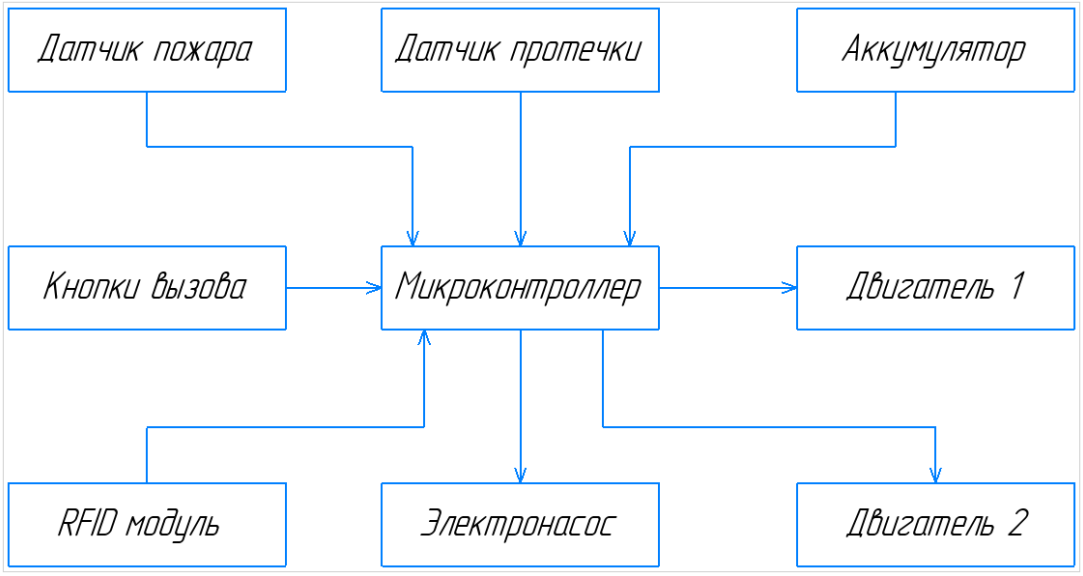

Рисунок 17 – Структурная схема платформы

2.2 Выбор необходимых компонентов

По заданию мы используем контроллер Arduino ATmega2560, но для полного проекта нам потребуется выбрать ряд других компонентов, чтобы объединить с нашим контроллером и на базе ардуино создать автоматизированную парковку с автоматической системой безопасности.

Исходя из нашей структурной схемы нам необходимы такие компоненты как: rfid-модуль, rfid-карты, датчик пламени, датчик уровня жидкости, кран для подачи воды, водяной насос для откачки жидкости, двигатели для лифта, драйверы для двигателей, кнопки.

В первую очередь выберем rfid-модуль и опишем его функциональность и подключение к нашему контроллеру.

Рисунок 18 – rfid-модуль RC522 и rfid-карты

Rfid-модуль RC522 представляет собой плату, которая способна идентифицировать объекты по их уникальному цифровому коду, что является отличным решением с безопасностью доступа. Цифровой код считывается из метки, которая обладает своей памятью, где располагается ее индивидуальный номер. Сам считыватель имеет в себе передатчик и саму антенну, что посылает в эфир электромагнитные сигналы определенной частоты. В свою очередь метки отвечают антенне своими сигналами, в которых содержится информация, то есть индивидуальный код идентификации. Антенна принимает данный сигнал, после чего он придается расшифровке и уже после посылается для дальнейшей обработки. Так же rfid-метки обладают криптографической защитой, что позволяет избежать копирования вашей личной карты или попыток ее подделывания.

Преимущества данной RFID технологии:

- Бесконтактная: Данная технология использует радио связь для получения и передачи данных, что значительно облегчает практическое использование

- Возможность скрытого установления метки: Это позволяет избежать повреждений механизма, ведь считыватель можно убрать в корпус, который сделан из прочных материалов, так как считывание происходит бесконтактно

- Высокая скорость считывания данных: Модуль работает на микросхеме MFRC522, что обеспечивает быстродействие считывания и корректную работу устройства

- Невозможность подделки: Как говорилось ранее данная технология обладает криптографической защитой, что делает его безопасным для эксплуатации.

Модуль RFID RC522 имеет данные технические характеристики:

- Напряжение питания: 3,3 В
- Потребляемый ток: 13-26 мА
- Рабочая частота: 13,56 МГц
- Дальность считывания: от 0 до 60 мм
- Интерфейс: SPI
- Максимальная скорость передачи: 10 Мбит/с
- Размер: 40мм на 60мм.

Интерфейс и назначение выводов данного устройства. Микросхема MFRC522 данного устройства поддерживает такие интерфейсы как SPI, UART, а также I2C, чтобы выбрать интерфейс необходимо установить логические уровни на самом устройстве, а именно на определенных выводах микросхемы. На нашем модуле установлен интерфейс SPI.

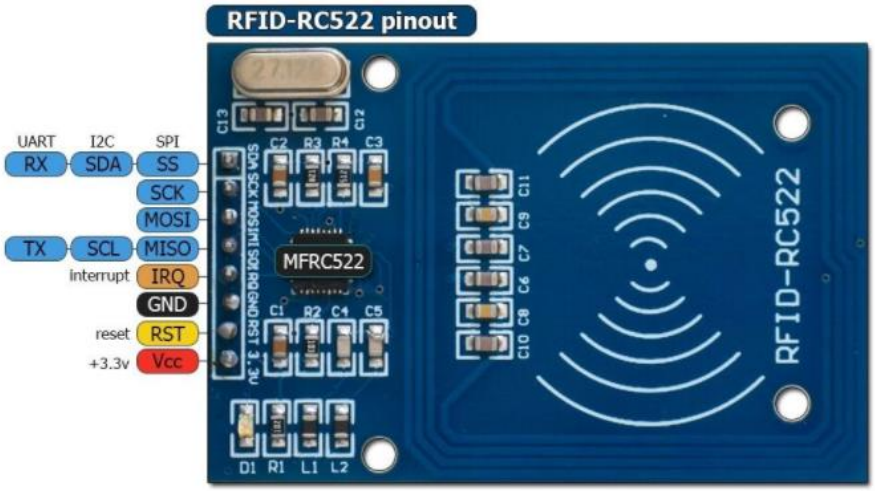

Рисунок 19 – Выводы RFID модуля RC522

Теперь рассмотрим все выводы данной платы с интерфейсом SPI.

- Вывод SDA: Выбор интерфейса ведомого
- Вывод SCK: Сигнал синхронизации нашего устройств
- Вывод MOSI: Передача от master к slave
- Вывод MISO: Работает обратно предыдущему и совершает

передачу от slave к master

- Вывод RST: Работает аналогично такому же выводу на микроконтроллере и имеет функцию сброса
- Вывод IQR: Вывод, отвечающий за прерывания
- Вывод GND: Обеспечивает вывод на землю
- Вывод Vcc: Служит для питания платы, которое составляет 3,3 В.

Следующим компонентом рассмотрим датчик пламени KY-026, который работает по принципу идентификации открытого пламени при помощи инфракрасного приемника. Индикатор включается если обнаруживается открытое пламя, в то же время на выход устройства подается логическая единица, как знак обнаружения угрозы пожара. В обратном случае, если сигнал о пламени не поступает, то на выходе датчика появляется логический ноль, как знак отсутствия угрозы. На основе данного датчика будет построена наша противопожарная система на парковке. Данное устройство выполняется в двух вариантах сборок с 3 выводами (для получения цифрового сигнала) и с 4 выводами (для получения аналогового сигнала). Оба устройства, которые показаны на рисунке 20.

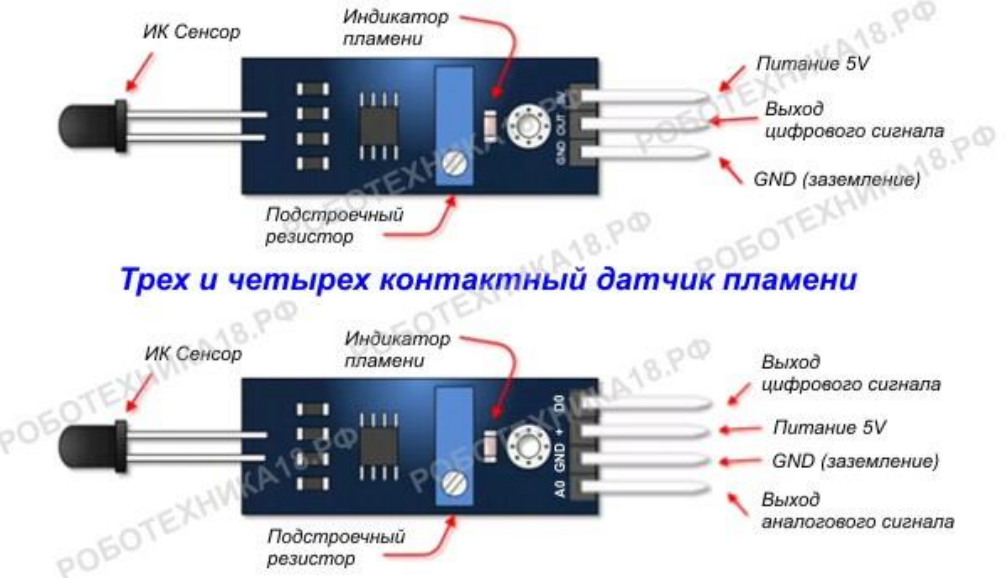

Рисунок 20 – Распиновка датчиков пламени KY-026

Характеристики датчика:

- Данный датчик имеет угол обнаружения пламени равный 60 градусам
- Обнаруживает пламя на расстоянии 1 метра
- Имеет напряжение питания равное 3-5,5 В
- Габариты датчика (длина х ширина) 36 х 16 мм.

Для нашего проекта мы выбираем цифровой датчик с тремя выводами, чтобы на выходе иметь только цифровой сигнал.

Следующим рассмотрим датчик протечки и дождя, который позволит нам определить есть ли вода на дне парковки или же нет, чтобы в случае чего откачать заполненное дно парковки с помощью водяного насоса. Сам модуль данного устройства состоит из двух частей. Во-первых, это сенсорная плата, которая необходима для обнаружения воды. Работает данная плата как переменный резистор, который в случае попадания на него воды замыкается и тем самым информирует с помощью сигнала второе устройство, так как на плате поменялось сопротивление. Во-вторых, у нас имеется сдвоенный компаратор, который превращает сигналы о изменении сопротивления на первом устройстве в аналоговые от 0 до 5 вольт, а затем передает на наш контроллер. Питается такая схема от напряжения в 5 вольт, что позволяет нам спокойно использовать его для нашего проекта. У этого модуля также имеется два выхода, которые представляют собой 2 сигнала (аналоговый и цифровой). На аналоговом выходе данного устройства будет выходить сигнал в диапазоне от 0 до 1023, что будет поступать на наш контроллер. 0 в данном интервале представляет собой ситуацию, в которой датчик практически затоплен, а 1023 будет означать, что наш датчик полностью сух, что говорит об отсутствии влаги или жидкости на дне нашей парковки.

Цифровой же в свою очередь выдает нам сигнал с помощью подачи высокого или же низкого напряжения, который поступает на контроллер, в случае если значение на датчике превысило некий порог, который мы можем настраивать с помощью подстроечного резистора, что позволяет нам настроить какой именно будет максимально возможный уровень жидкости на нашей парковке, чтобы после чего сигнал подавался уже на контроллер, чтобы тот задействовал водяной насос для устранения данной проблемы. Данное устройство можно увидеть на рисунке 21.

Рисунок 21 – Датчик протечки

Данное устройство выбрано в связи с простотой эксплуатации, а также с небольшой стоимостью, но надежным исполнением собственной функции, которая необходима для нашего проекта. Для нашего проекта мы выбираем датчик с компаратором LM393, но также существуют еще компараторы LM293 и LM193, изображение которых представлено на рисунке 22. Выбор определен в силу того, что в основном с данным датчиком используется именно LM393.

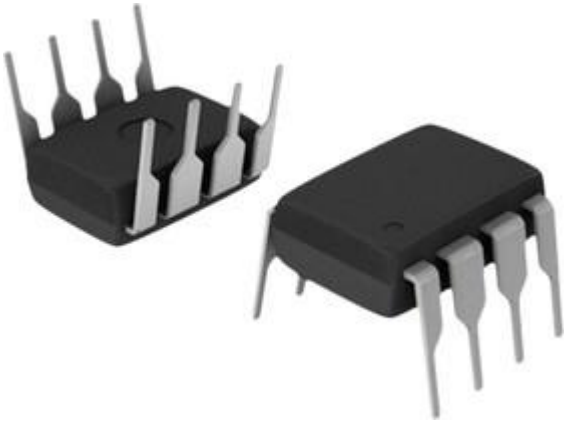

Рисунок 22 – компаратор

Сенсорная плата обнаружения капель идет в комплекте с нашим датчиком, но существуют и другие комплектации, которые отличаются формой и некоторыми параметрами, что можно увидеть на рисунке 23.

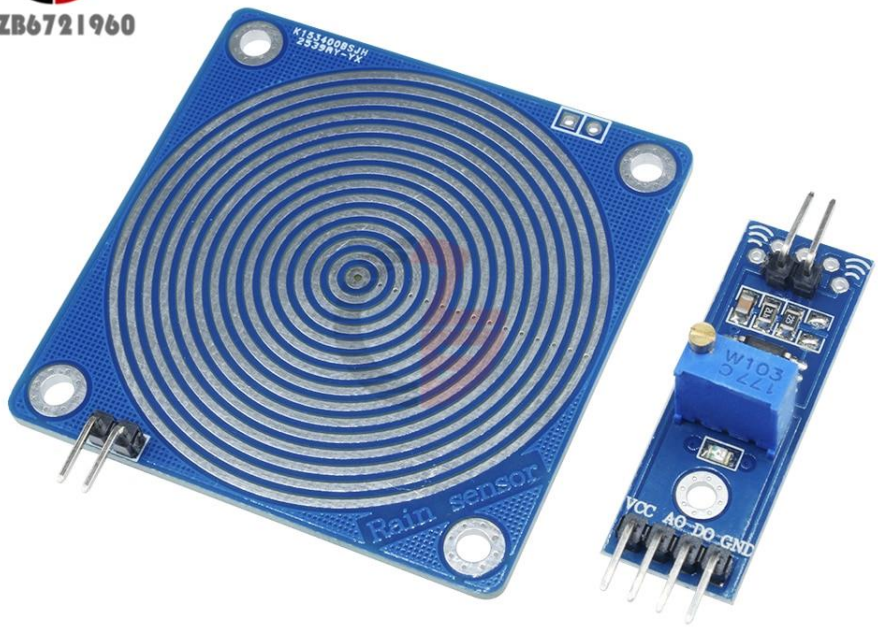

Рисунок 23 – Датчик протечки

Нас интересует первый комплект в связи с ценовым диапазоном.

Далее рассмотрим систему подачи воды в случае пожара на парковке. Для этого нам понадобится двухходовой электрический водяной клапан CVX-15, который будет открываться при сигнале с датчика пламени о пожаре. Изготовлен данный образец из нержавеющей стали, что способствует долгому времени эксплуатации. Имеет рабочее напряжение питания 5 вольт, что полностью соответствует нашим требованиям, а также рабочий ток менее 100 мА, что тоже удовлетворяет нашим требованиям. Устройство представлено на рисунке 23.

Рисунок 23 – двухходовой электрический водяной клапан CVX-15

Рассмотрим водяной насос для откачки воды. Мы будем использовать бесщеточный погружной насос DC 12 В. Данный насос имеет работоспособность, которая позволяет нам выкачать 240 литров в час при этом имея довольно малый рабочий ток 350 мА, что идеально подходит требованиям нашего проекта. Это малоразмерный двигатель, который имеет погружной насос из-за чего данный аппарат будет находится под общей конструкцией и станет ее замыкающим звеном. Работает устройство от питания в 12 вольт, что соответствует проекту. Насос работает тихо по большей части из-за того, что является бесщеточным, а также имеет среднюю

мощность. Работа будет осуществляться по принципу, что на дне парковки скапливается вода и сразу же попадает в отсек, который присоединен к входной трубке насоса, но насос будет бездействовать пока вода не поднимется до уровня нашего датчика протечки, поэтому датчик протечки будет расположен на пару миллиметров выше уровня пола, чтобы на момент срабатывания в насосе уже была вода. Так насос не будет работать в сухом состоянии, что гарантирует отсутствие перегрева и дальнейших повреждений устройства. После того как вода достигнет датчика протечки, он посылает сигнал на контроллер, который в свою очередь подаст сигнал на реле, что подключена к насосу и задействует механизм полностью. После достижения уровня воды ниже датчика протечки, что приведет к подаче сигнала датчиком на контроллер об остановке работы насоса, который в свою очередь не остановится из-за винта и кинетической энергии выкачиваемой воды, но после того как в насос больше не будет поступать вода, он автоматически остановится так как будет отсутствовать сила, что побуждает винт насоса вращаться. Таким образом мы избегаем чрезмерного потребления энергии, а также позволяем насосу иметь возможность плавной остановки, что гарантирует нам долговечность устройства. бесщеточный погружной насос DC 12 В показан на рисунке 24.

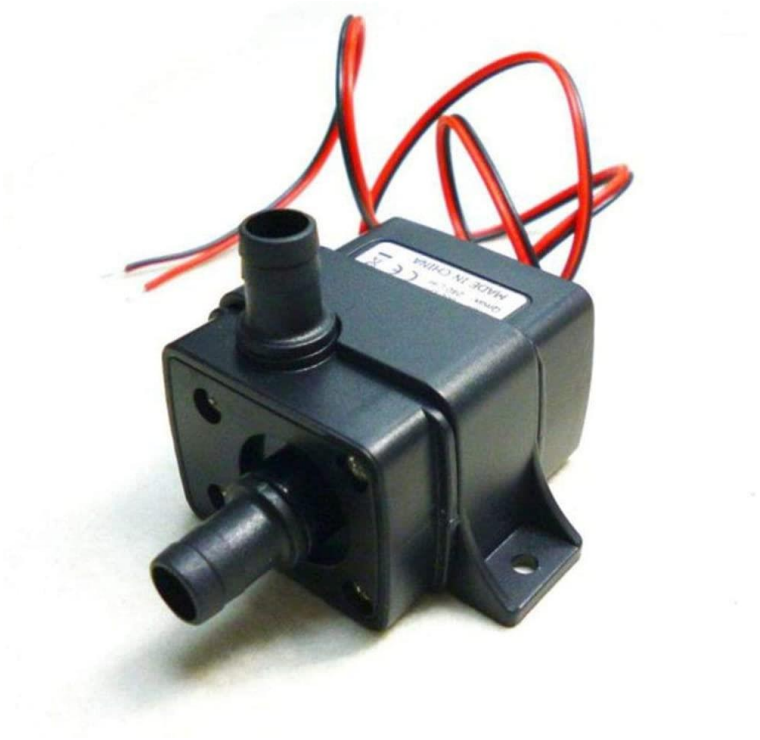

Рисунок 24 – бесщеточный погружной насос DC 12

Характеристики насоса:

- Напряжение питания 12 В
- Рабочий ток 350 мА
- Объем выкачиваемой жидкости 240 литров в час
- Температура жидкости от 0 до 75 градусов

Данное устройство имеет огромное количество аналогов на рынке. Помимо конструкции основными отличительными параметрами являются:

- Напряжение питания
- Рабочий ток
- Объем выкачиваемой жидкости в час
- Рабочая температура

Представим пару аналогов и их характеристики.

Погружной горизонтальный насос DC 12 В, что показан на рисунке 25.

Рисунок 25 – горизонтальный погружной насос

Данный аппарат имеет характеристики:

- Напряжение питания 12 В
- Рабочий ток 220 мА
- Объем выкачиваемой жидкости 600 литров в час
- Температура жидкости от 0 до 50 градусов

Мы остановимся на выборе первого варианта, так как он работает более умеренно, что позволяет избежать перегрева и дальнейшего выхода из строя. Имеет большую термостойкость, а также горизонтальную структуру, что больше подходит для выполнения функции в нашем проекте.

Разберем выбор драйвера для двигателя. Для нашего проекта был выбран драйвер L298N, который зачастую используется для различных многофункциональных проектов, где присутствуют двигатели постоянного тока. Схема данного устройства представляет собой два Н-моста, что позволяет подключить к драйверу биполярный шаговый двигатель, либо два

щеточных двигателя постоянного тока одновременно. Одной из важнейших функций является возможность управления скоростью вращения двигателей, что позволяет пользователю делать плавные переходы в проектах. Для управления на драйвере имеется четыре штыревых контакта и при помощи подачи сигналов на них осуществляется воздействие и корректировки. На рисунке 26 представлен драйвер, который описывался выше.

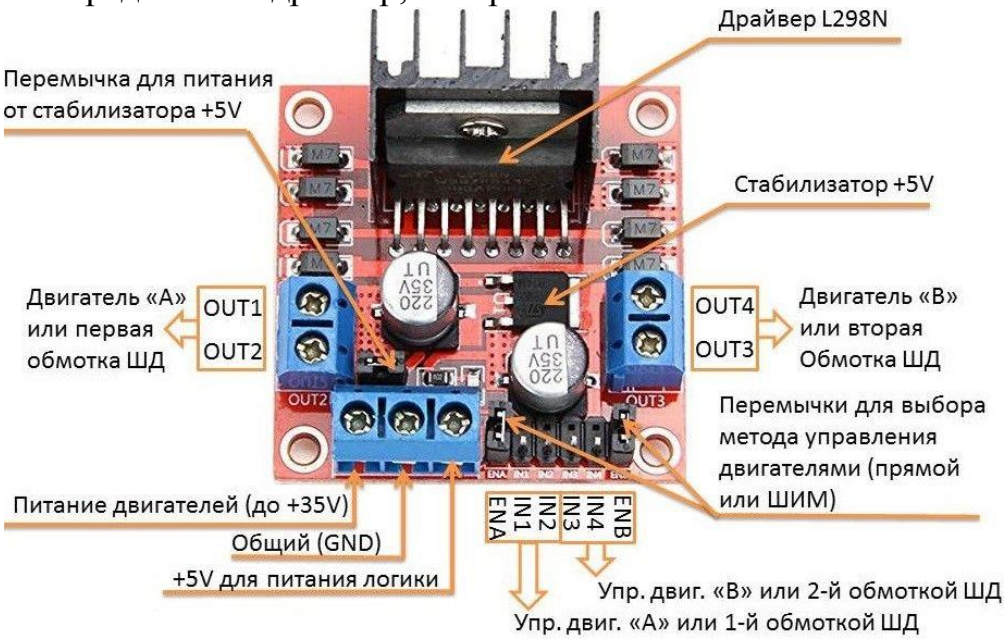

Рисунок 26 – Драйвер L298N

Разберем компоненты, что расположены на плате драйвера.

- OUT1 и OUT2 – это разъемы, на которые можно подключить один щеточный двигатель и управлять его скоростью вращения и направлением, или же подключаем одну из обмоток шагового двигателя OUT3 и OUT4 – разъемы, которые отвечают за те же функции, что OUT1 и OUT2
- VSS – разъем, отвечающий за питание драйвера имеющий уровни от 5 до 35 В
- GND – земля, которую необходимо соединить с таким же выводом на контроллере

- Vs – вход, который используется для принятия питания на логическом элементе +5В. Именно через этот вход питается наша микросхема L298N

- IN1 и IN2 – это контакты, на которые мы подаем управляющие сигналы для регулировки работы щеточного двигателя или для управления одной обмотки шагового двигателя

- IN3 и IN4 – контакты, работающие по тому же принципу, что и предыдущие. Управляют либо щеточным двигателем, либо второй обмоткой шагового двигателя

- ENA и ENB – контакты, созданные для активации, либо деактивации одного из двигателей. Тут работает принцип подачи на контакты либо логической единицы, либо логического нуля, поэтому обычно на них стоят перемычки, чтобы имелась постоянное питание 5В. На рисунке 27 мы можем увидеть электрическую схему данного драйвера, во главе которого стоит микросхема L298N.

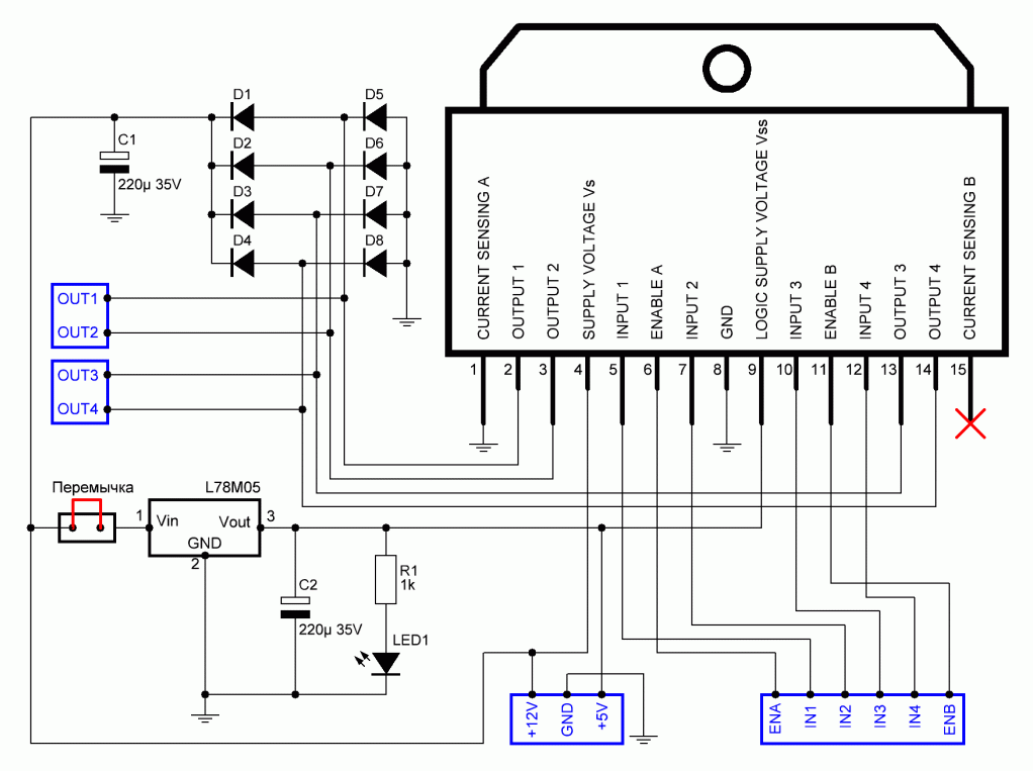

Рисунок 27 – Электрическая схема драйвера L298N

На схеме наглядно представлено наличие двух Н-мостов, которые позволяют нам менять направление вращения двигателей при помощи возможности чередовать открытые и закрытые транзисторные ключи, что можно наглядно увидеть на рисунке 28.

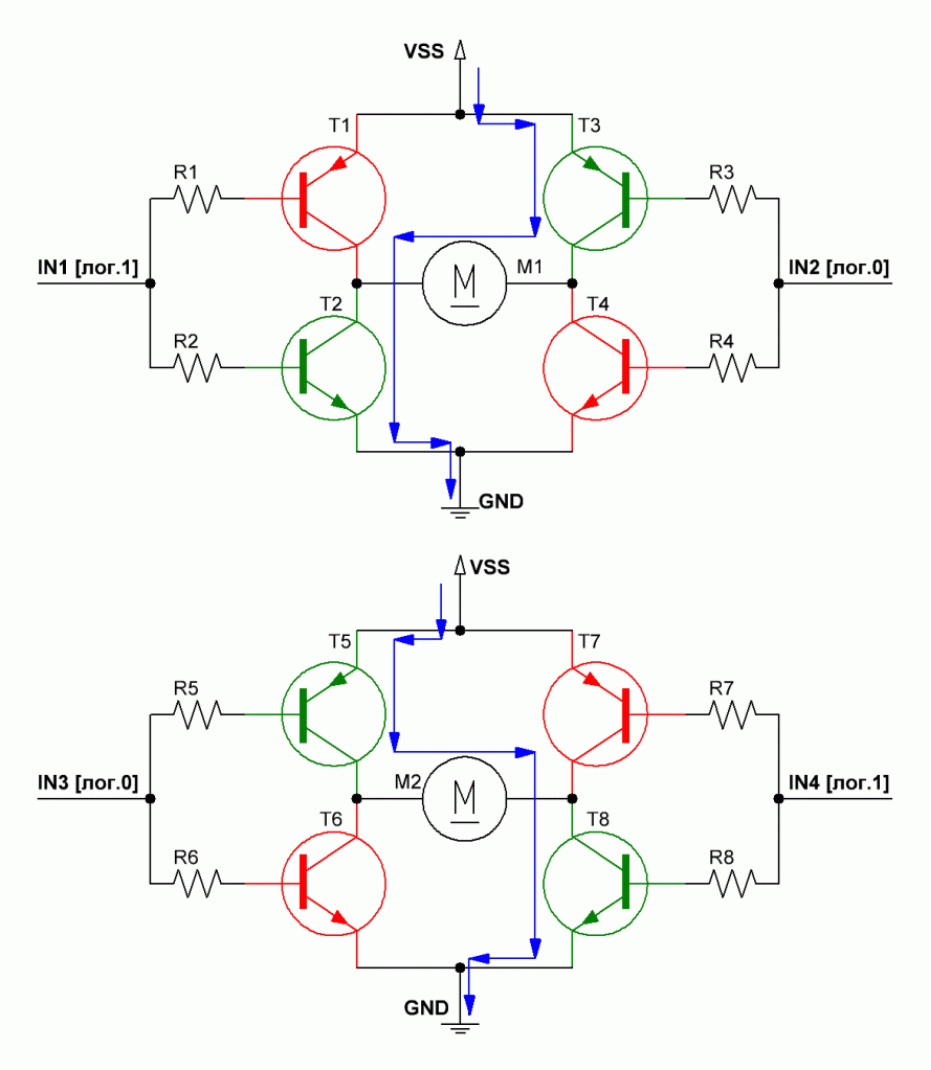

Рисунок 28 – Транзисторные мосты Н-типа

Технические характеристики драйвера:

- Напряжение питания логики составляет 5 В
- Потребляемый логикой ток 36 мА
- Возможное напряжение моторов в диапазоне от 5 до 25 В
- Рабочий ток драйвера составляет 2 А

- Максимальный ток драйвера составляет 3 А
- Пиковая мощность драйвера составляет 20 Вт
- Диапазон температур, в которых драйвер может работать составляет от -25 до +135 градусов Цельсия.

Рассмотрим аналог нашего драйвера. Это будет драйвер моторов двухканальный Pololu на базе микросхемы MC33926, что показан на рисунке 29 и его характеристики.

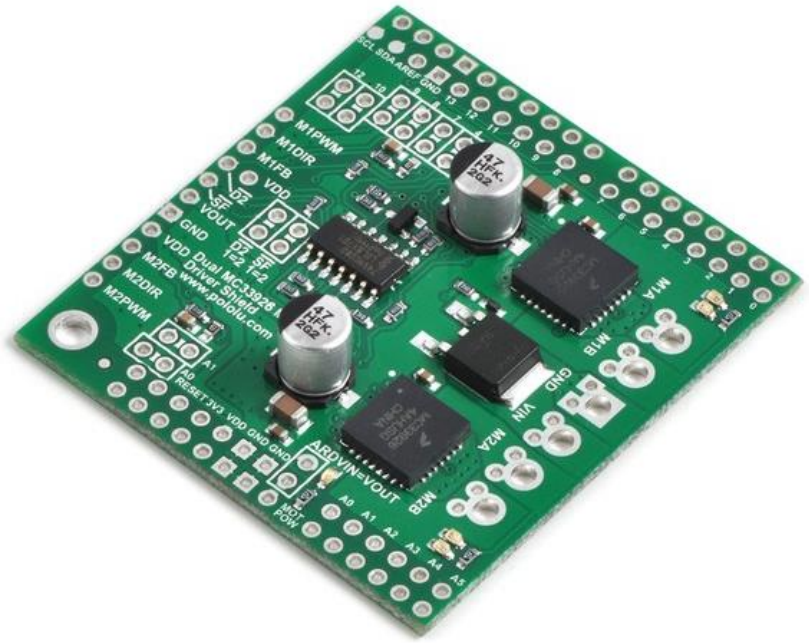

Рисунок 29 – Драйвер Pololu MC33926

Данное устройство более универсально и подходит большому количеству управляющих элементов, также оно более подходит для выполнения масштабных проектов, что явно отражается разницей в ценовом диапазоне первого экземпляра и драйвера Pololu. Мы остановимся на выборе драйвера L298N, ибо мы будем использовать весь его функционал потратив

на это меньшую сумму средств, чем на драйвер pololu.

Переходим к выбору реле. Для нашего проекта подойдет реле, которое обычно находится в комплекте наборов Arduio. В таких комплектах реле обычно подразделяется на два типа:

- Электромагнитное
- Твердотельное

Их разница будет описана ниже. Само по себе реле это микросхема, которая размыкает и замыкает подачу напряжения на управляемый объект. С помощью этого механизма можно реализовать множество функций в различных проектах. Характеризуется устройство такими параметрами как:

- Напряжение или ток подаваемого сигнала для работы механизма
- Напряжение или ток подаваемого сигнала для прекращения работы механизма
- Само время работы процесса срабатывания и отпускания
- Рабочие величины тока и напряжения для схемы
- Внутреннее сопротивление

Теперь рассмотрим, чем же отличается электромагнитное реле от твердотельного. Электромагнитное реле механическим способом размыкает и замыкает сигнал, подаваемый на управляемое устройство с помощью электромагнита. Сам электромагнит представлен в виде проволоки намотанной на катушку из ферромагнетика, а в роли якоря выступает магнитная пластина, которая и будет при подачи определенного сигнала притягивать к себе первую составляющую, что приведет к замыканию и соответственно проходу сигнала и наоборот. Само устройство можно увидеть на рисунке 30.

Рисунок 30 – Электромагнитное реле

Следующим рассмотрим твердотельное реле, которое представляет собой устройство работающее на основе полупроводников. Оно показано на рисунке 31.

Рисунок 31 – Твердотельное реле

Для нашего проекта мы воспользуемся электромагнитным реле в связи с меньшей стоимостью при том же нужном нам функционале.

Также нам понадобятся обычные тактовые кнопки, которые будут исполнять функцию вызова нужного этажа парковки с необходимым автомобилем. Кнопка работает по принципу размыкания и замыкания

участка цепи для подачи управляющего сигнала на управляемый элемент, но путем воздействия на нажатие со стороны пользователя. Изображение кнопки представлено на рисунке 32.

Рисунок 32 – Тактовая кнопка

Следующим шагом выбираем двигатель, который будет поднимать автомобили по принципу работы лифта. Для макета подойдет двигатель TT Motor Operating 1:48, который обладает хорошим показателем оборотов в секунду, а именно 200 оборотов в минуту, что дает хорошую скорость подъема автомобиля. Подойдет данный образец только для миниатюрного макета для мини полигона, так как контроллер такого уровня не может работать с мощными двигателями, а для создания лифта автоматизированной парковки необходимо два мощных двигателя, ибо вес автомобилей превышает несколько тонн и это не считая вес самой конструкции лифта с поддонами. Полные характеристики двигателя:

- Напряжение питания необходимо от 3 до 6 В
- Редуктор 1:48
- С питанием в 6 В, имеет 200 оборотов в минуту
- С питанием в 3 В, имеет 120 оборотов в минуту

Изображение двигателя можно увидеть на рисунке 33.

Рисунок 33 – TT Motor Operating 1:48

Аналогично этому двигателю можно применять и его аналоги, которые обладают другими характеристикам как например ТТ-мотор с редуктором 1:120, который изображен на рисунке 34.

Рисунок 34 – ТТ-мотор с редуктором 1:120

Который обладает характеристиками:

- Напряжение питания необходимо от 3 до 6 В
- Редуктор 1:120
- С питанием в 6 В, имеет 80 оборотов в минуту
- С питанием в 3 В, имеет 50 оборотов в минуту

Что уступает нашему двигателю, поэтому мы остановимся на первом и окончательном варианте.

2.3 Разработка электрической принципиальной схемы

В соответствии с исходными данными, структурной схемой и выбранными компонентами была разработана схема электрическая принципиальная. Схема электрическая соединений представлена на рисунке 35.

Оформление схемы производилось в системе автоматизированного проектирования КОМПАС 3D V16.

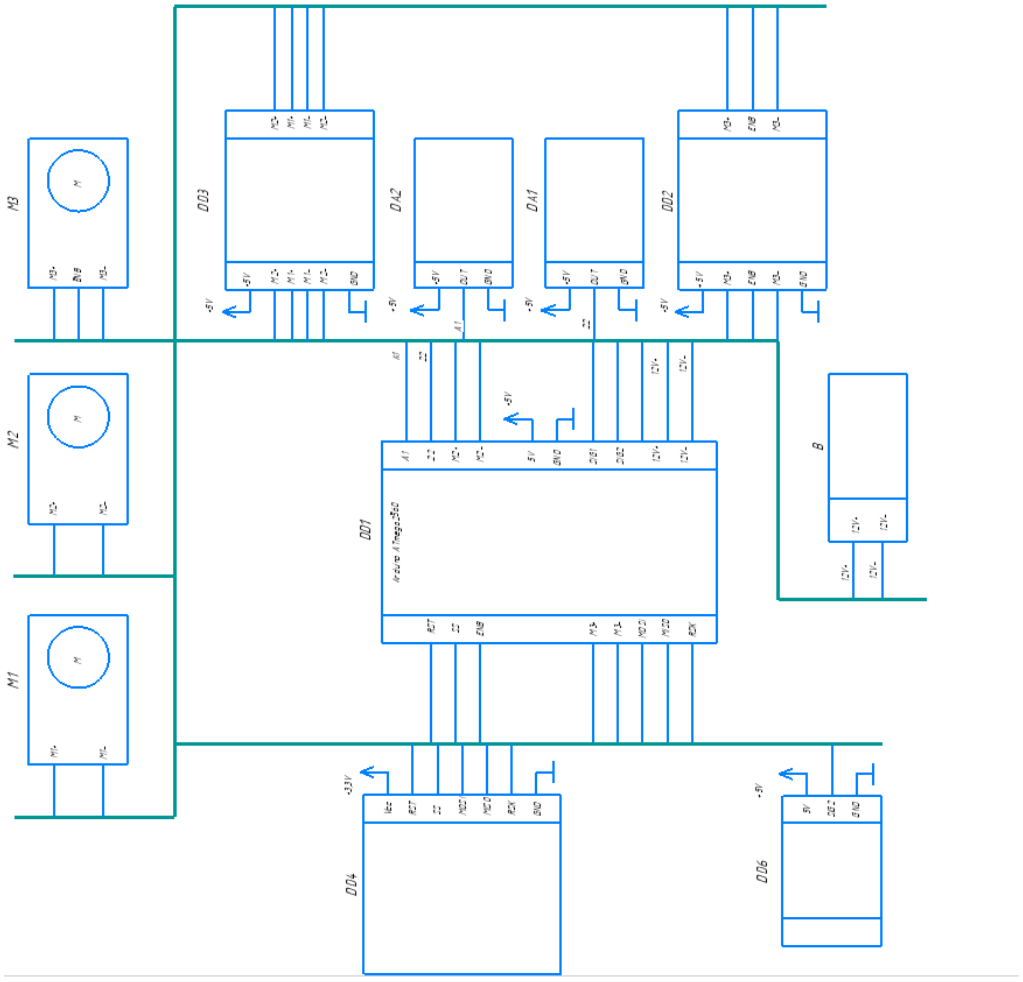

Рисунок 35 – Принципиальная схема автоматизированной парковки

Перечень элементов к схеме представлен в приложении А

Выводы по разделу

Таким образом, в данной главе была создана структурная схема и схема принципиальная электрическая. Так же был изучен принцип работы датчиков пламени и протечки. На основе изученных данных был произведен подбор, сравнение и выбор датчиков для дальнейшего использования в платформе. Была изучена работа RFID модуля и меток. Были рассмотрены твердотельное и электромагнитное реле. Выбор был сделан в пользу компонентов, которые соответствовали нашим требованиям.

3. Программная часть

3.1 Разработка алгоритма работы

По техническому заданию парковка должна поднимать наш автомобиль по вызову определенной кнопки, которую можно будет нажать только после идентификации пользователя при помощи RFID карты с индивидуальным номер, что хранится в ее памяти и занесен в контроллер как разрешенный пользователь. В свою очередь система защиты будет выполнять функции по принципу, что если случается возгорание, то его необходимо моментально потушить, также работает и с наводнениями, если на дне парковки набирается вода, то необходимо ее откачать.

Расположение датчиков пламени и протекания будет на стенках парковки, но на разной высоте. Датчик протечки будет располагаться в паре миллиметрах от дна парковки, а датчик пламени в свою очередь на верхнем уровне, чтобы иметь максимальный охват территории, на которой может начаться возгорание.

На рисунке 36 изображена блок-схема разработанного алгоритма.

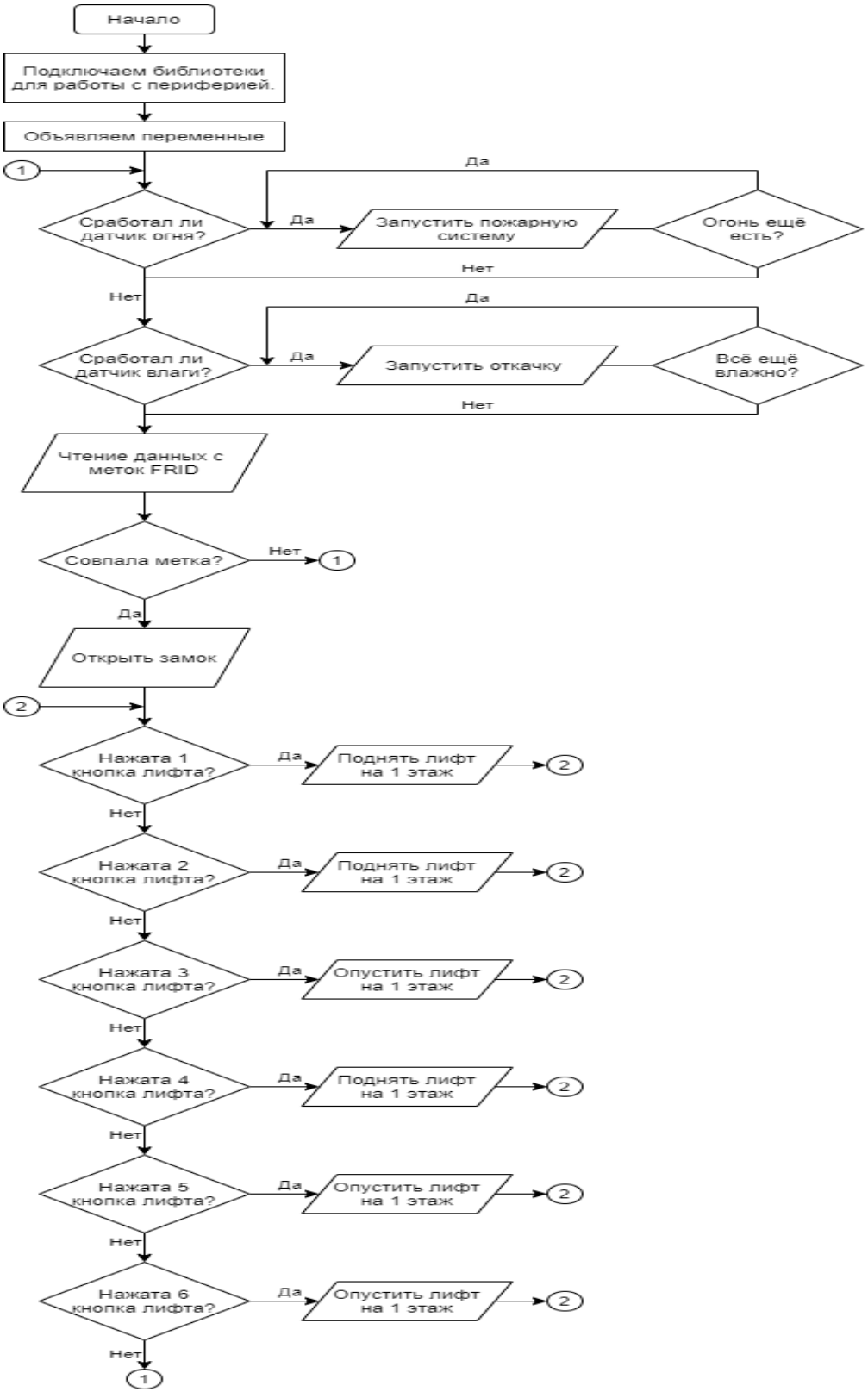

Рисунок 36 – Блок-схема алгоритма работы платформы

3.2 Разработка программной части устройства

Контроллер ATmega2560 подключаем к любому персональному компьютеру, на котором установлен софт разработки в среде ARDUINO. После чего создаем скетч в программе Arduino 1.8.3, и приступаем к созданию кода для нашего проекта. Для начала необходимо подключить все необходимые библиотеки, которые необходимы нам для сокращения работы с определенными функциями, так как они уже прописаны в самих библиотеках, что сокращает затраты времени на написание кода. Делаем это при помощи include, что показано на рисунке 37.

```
// Подключение библиотек
#include <MFRC522.h>
#include <SPI.h>
```

Рисунок 37 – подключение библиотек с помощью include

После чего с помощью воспользуемся директивой define для задания имен нашим константам, чтобы после чего было проще для пользователя и программиста обращаться к ним и видеть, как работает код. Использование директивы define показано на рисунке 38.

```
#define FlamePin  A1       // Пин для датчика огня
#define DryPin 22          // Пин для датчика влажности

#define FlameCraneOpen 23  // Пин открытия крана пожарной системы
#define FlameCraneClose 24 // Пин закрытия крана пожарной системы
```

Рисунок 38 – использование define

После того как мы прописали все основные элементы через define, что позволит нам не запутаться в программе и затратить меньше времени на поиск какой пи и куда мы подключили.

Далее необходимо прописать массив разрешенных uid, что позволит нам сделать ограниченный доступ к парковке благодаря картам. Для одной парковки можно сделать достаточно карт доступа, чтобы при необходимости потребителей выдать несколько экземпляров карт доступа. Пример записи доступных uid можно увидеть на рисунке 39.

```
// Массив разрешенных uid
byte uidok[][4] = {
  {0xE0, 0x2A, 0x87, 0x1B}, // Метка 1
  {0xD9, 0xFA, 0x90, 0x55}  // Метка 2
};
```

Рисунок 39 – Массив разрешенных uid

Теперь можем переходить к подключению систем безопасности, которые работают по принципу, что если датчик улавливает сигнал, то система пожаротушения и откачки воды срабатывают моментально. Прописываем данную функцию через if так как нам не нужно выжидать определенное время, а просто включить и выключить. Пример функции можно увидеть на рисунке 40.

```
// Функция работы сенсоров
void Sensors()
{
  if (digitalRead(FlamePin) == HIGH) // Если срабатывает датчик огня
  {
    // Открываем кран
    digitalWrite(FlameCraneOpen, HIGH);
    digitalWrite(FlameCraneClose, LOW);

    if (digitalRead(FlamePin) == LOW) // Если огня больше нет
    {
      // Закрываем кран
      digitalWrite(FlameCraneOpen, LOW);
      digitalWrite(FlameCraneClose, HIGH);
    }
  }
```

Рисунок 40 – Функция работы датчика пожара

Поиск uid в списке разрешенных выполняется контроллером, по принципу считывания массивов данных с карт и сравнивания их с теми, что

находятся в категории разрешенных. Пример кода представлен на рисунке 41.

```
// поиск считанного uid в списке разрешенных
boolean compare_uid(byte *buffer, byte bufferSize)
{
  int bytes_ok = 0;
  for (int i1 = 0; i1 < 2; i1++) // Проверяем на соотвествие с 2 метками в массиве
  {
    bytes_ok = 0;
    for (byte i2 = 0; i2 < 4; i2++)
    {
      if (buffer[i2] == uidok[i1][i2]) // Если байт соответсвует байту из архива
      {
        bytes_ok = bytes_ok + 1; // Увелчиваем значение переменной на 1
      }
    }
    if (bytes_ok == 4) // Если все 4 байта совпали с байтами в массиве
    {
      return true; // Возвращаем значение true
    }
  }
  return false; // Возвращаем значение false
}
```

Рисунок 41 – Поиск считанного uid в списке разрешенных

Далее рассмотрим функцию подачи автомобиля его владельцу при помощи лифта. Для этого была введена определенная константа расстояния, на которое необходимо поднимать или опускать лифт при вызове автомобиля с определенного этажа. Пример кода можно увидеть на рисунке 42.

```
// Функция поднятия лифта
void ElevatorUp()
{
  Turn = 0; // Обнуляем значение переменной
  for (int i = Turn; i < ElevatorTurn; i++) // Поднимать лифт, пока значене прерывания не достигнет заданной велечины
  {
    analogWrite (MotorSpeed, Speed);
    digitalWrite(MotorForward, HIGH);
    digitalWrite(MotorBackward, LOW);
  }
}

// Функция опускание лифта
void ElevatorDown()
{
  Turn = 0; // Обнуляем значение переменной
  for (int i = Turn; i < ElevatorTurn; i++) // Опускать лифт, пока значене прерывания не достигнет заданной велечины
  {
    analogWrite (MotorSpeed, Speed);
    digitalWrite(MotorForward, LOW);
    digitalWrite(MotorBackward, HIGH);
  }
}
```

Рисунок 42 – Функция подъема лифта

Полный код для ознакомления будет представлен в приложении Б.

Выводы по разделу

Подведем следующие выводы:

- Платы Arduino отлично подходят для довольно небольших проектов, которым необходимо небольшое питание, что идеально подходит для обучения, но большие проекты, которые имеют в составе агрегаты, что требуют больших мощностей просто не подходят для данной платы, так как она элементарно не может выдержать таких нагрузок. Поэтому если говорить о цели существования Arduino, то это однозначно небольшие проекты, но идеально подходящие для обучения и адаптации в мире программирования контроллеров;
- У Arduino есть проблема, что нельзя запускать несколько процессов параллельно, не используя при этом прерывания, что сглаживают этот процесс. Потому что данный контроллер является однопоточным;
- библиотека для программирования очень полезная вещь, что позволяет нам не прописывать множество функций, а использовать уже готовую библиотеку для работы с тем или иным оборудованием.

Заключение

В ходе выполнения бакалаврской работы, была разработана автоматизированная парковка с системой безопасности, которая способна отдавать вам ваш автомобиль с вашего этажа. Система безопасности, которая не позволит никому кроме владельцев доступа как-либо воздействовать на ваш автомобиль, сохранит его и продлит срок эксплуатации, ибо автомобиль находится в закрытом пространстве, защитит его от нахождения в сырости и от пожара-опасных ситуаций. Также мы рассмотрели существующие решения такого вида паркинга во всех странах и разобрали все типы, что нам удалось найти. Огромные парковочные системы, работающие на разных системах и для разных применений. Был произведен подбор, сравнение и выбор необходимых компонентов, таких как: RFID модуль, датчик протечки, датчик пламени, драйвера двигателей, двигатели, реле, насосы и тактовая кнопка. Разработана схема принципиальная электрическая, а также перечень элементов к ней. Разработан алгоритм работы парковки и его программная реализация, в которой была применена специализированная библиотека для удобной работы с микроконтроллером, двигателями и датчиками. Представлена программа и описаны результаты.

Список используемой литературы

1. Амперка [Электронный ресурс]. URL: http://wiki.amperka.ru/продукты:arduino-mega-2560
2. Водяной насос Arduino [Электронный ресурс]. URL: https://www.hwlibre.com/ru/bomba-de-agua-arduino/
3. Драйвер двигателя L298N [Электронный ресурс]. URL: https://3d-diy.ru/wiki/arduino-moduli/drayver-dvigatelya-l298n/
4. Драйвер моторов двухканальный pololu [Электронный ресурс]. URL: https://3d-diy.ru/wiki/arduino-moduli/drayver-dvigatelya-l298n/
5. Парикнги мегаполиса [Электронный ресурс]. URL: https://www.mehparking.ru/avtomatizirovannye-parkovki-parkingi/
6. Справочник языка Ардуино [Электронный ресурс]. URL: http://arduino.ru/Reference
7. ФАЛАБ «робототехника ардуино» [Электронный ресурс]. URL: https://роботехника18.рф
8. AKE [Электронный ресурс]. URL: https://www.akeparking.com
9. Arduino-kit уроки и проекты Arduino [Электронный ресурс]. URL: https://arduino-kit.ru/blogs/blog/uroki-i-proekty
10. ARDUINOMASTER [Электронный ресурс]. URL: https://arduinomaster.ru/datchiki-arduino/datchik-protechki-i-dozhdya-v-arduino-opisanie-shemy-sketchi/
11. Arduino tutorials [Электронный ресурс]. URL: https://www.arduino.cc/en/Tutorial/HomePage
12. ArduinoModules [Электронный ресурс]. URL: https://arduinomodules.info

13. Car and driver [Электронный ресурс]. URL: https://www.caranddriver.com/research/a31995865/automatic-parking-systems/

14. CircuitDigest [Электронный ресурс]. URL: https://circuitdigest.com/arduino-projects

15. ECOSPACE технические решения [Электронный ресурс]. URL: http://ecospacerus.ru/stati/mnogoehtazhnyj-parking/

16. RoboCraft [Электронный ресурс]. URL: http://robocraft.ru/blog/3004.html

17. Stolzer [Электронный ресурс]. URL: https://www.stopa.com/en/parking-systems/157/parking-system-auto-up

18. TADVISER [Электронный ресурс]. URL: https://www.tadviser.ru/index.php/Статья:Умные_парковки

19. wikipedia [Электронный ресурс]. URL: https://en.wikipedia.org/wiki/Automatic_parking

20. 3DiY ардуино и робототехника [Электронный ресурс]. URL: https://3d-diy.ru/wiki/arduino-moduli/rfid-modul-rc522/

Приложение А

Перечень элементов к схеме принципиальной электрической

Поз.	Наименование	Кол.	Примечание
	Датчики		
DA1	Датчик влажности (протечки) LM-393	1	
DA2	Датчик пламени KY-026	1	
M1	Водяной насос	1	
M2	Двух ходовой водяной клапан CVX-15	1	
M3	Мотор TT motor operating 1:48	1	
DD1	Arduino ATmega2560	1	
DD2…DD3	Драйвер L298N	2	
DD4	RC522-RFID-модуль	1	
DD5…DD6	Электромагнитное реле KY-019	1	
B	Аккумуляторная батарея 12V	1	

					21–2110304.53/09.211.ПЭ			
Изм.	Лист	№докум.	Подп.	Дата				
Разраб.		Терехин С.В.			Автоматизированная парковка Перечень элементов	Лит.	Лист	Листов
Пров.		Кудинов А.К.					1	1
Н.контр.		Кудинов А.К.				ТГУ гр. ЭЛб-1702а		
Утв.		Шевцов А. А.						

Приложение Б

Программа автоматизированной мобильной платформы

```
// Подключение библиотек
#include <MFRC522.h>
#include <SPI.h>

#define FlamePin  A1       // Пин для датчика огня
#define DryPin 22          // Пин для датчика влажности

#define FlameCraneOpen 23  // Пин открытия крана пожарной системы
#define FlameCraneClose 24 // Пин закрытия крана пожарной системы

#define DryCraneOpen 25    // Пин открытия крана системы затопления
#define DryCraneClose 26   // Пин закрытия крана системы затопления

#define RELAY_RC522_PIN 27 // Пин для подключения реле
#define RELAY_ON 0         // уровни для включения реле
#define RELAY_OFF 1        // уровни для выключения реле

#define PiezoPin 38 // Пин пьезоэлемента

// константы подключения контактов RST и SS
#define RST_PIN  5
#define SS_PIN  53

// Двигатель лифта
#define MotorSpeed      4  // ENB, Скорость
#define MotorForward    29  // IN1, Вперед
#define MotorBackward   30  // IN2, Назад

#define Speed 255 // Скорость лифта

// кнопки лифта
#define ButtonPin_1 31 // Кнопка 1
#define ButtonPin_2 32 // Кнопка 2
#define ButtonPin_3 33 // Кнопка 3
#define ButtonPin_4 34 // Кнопка 4
#define ButtonPin_5 35 // Кнопка 5
#define ButtonPin_6 36 // Кнопка 6

#define TachPin 0 // Пин энкодера
#define ElevatorTurn 60 // Количество прерываний для лифта

MFRC522 mfrc522(SS_PIN, RST_PIN); // Создание экземпляра объекта MFRC522

// Массив разрешенных uid
byte uidok[][4] = {
```

```
    {0xE0, 0x2A, 0x87, 0x1B}, // Метка 1
     {0xD9, 0xFA, 0x90, 0x55}  // Метка 2
    };

    int Turn = 0;

    void setup() {
     Serial.begin(9600);
     SPI.begin(); // Инициализация SPI
     pinMode(FlamePin, INPUT);  // Инициализация пина датчика огня как вход
     pinMode(DryPin, INPUT);    // Инициализация пина датчика влажности как вход
     pinMode(PiezoPin, OUTPUT); // Инициализация пина пьезоэлемента как выход

     mfrc522.PCD_Init(); // Инициализация MFRC522
     pinMode(RELAY_RC522_PIN, OUTPUT); // Конфигурация выводов реле как
OUTPUT
     digitalWrite(RELAY_RC522_PIN, RELAY_OFF);

     // Инициализация пинов мотора как выход
     pinMode(MotorSpeed, OUTPUT);
     pinMode(MotorForward, OUTPUT);
     pinMode(MotorBackward, OUTPUT);
     // Задание режмов мотора
     digitalWrite(MotorForward, LOW);
     digitalWrite(MotorBackward, LOW);

     // Инициализация пинов кнопки как вход
     pinMode(ButtonPin_1, INPUT);
     pinMode(ButtonPin_2, INPUT);
     pinMode(ButtonPin_3, INPUT);
     pinMode(ButtonPin_4, INPUT);
     pinMode(ButtonPin_5, INPUT);
     pinMode(ButtonPin_6, INPUT);

     pinMode(TachPin, INPUT_PULLUP);
     attachInterrupt(digitalPinToInterrupt(TachPin), ISR_, FALLING);   // Подключаем
прерывания для левого колеса
    }

    // Функция прерывания
    void ISR_()
    {
     Turn++; // Увеличиваем значение на 1
    }

    // Функция работы сенсоров
    void Sensors()
```

```
{
 if (digitalRead(FlamePin) == HIGH) // Если срабатывает датчик огня
 {
   // Открываем кран
   digitalWrite(FlameCraneOpen, HIGH);
   digitalWrite(FlameCraneClose, LOW);

   if (digitalRead(FlamePin) == LOW) // Если огня больше нет
   {
     // Закрываем кран
     digitalWrite(FlameCraneOpen, LOW);
     digitalWrite(FlameCraneClose, HIGH);
   }
 }

 if (digitalRead(DryPin) == LOW) // Если срабатывает датчик влажности
 {
   // Открываем кран
   digitalWrite(DryCraneOpen, HIGH);
   digitalWrite(DryCraneClose, LOW);

   if (digitalRead(DryPin) == HIGH) // Если воды больше нет
   {
     // Закрываем кран
     digitalWrite(DryCraneOpen, LOW);
     digitalWrite(DryCraneClose, HIGH);
   }
 }
 RFID();
}

// поиск считанного uid в списке разрешенных
boolean compare_uid(byte *buffer, byte bufferSize)
{
 int bytes_ok = 0;
 for (int i1 = 0; i1 < 2; i1++) // Проверяем на соотвествие с 2 метками в массиве
 {
   bytes_ok = 0;
   for (byte i2 = 0; i2 < 4; i2++)
   {
     if (buffer[i2] == uidok[i1][i2]) // Если байт соответсвует байту из архива
     {
       bytes_ok = bytes_ok + 1; // Увелчиваем значение переменной на 1
     }
   }
   if (bytes_ok == 4) // Если все 4 байта совпали с байтами в массиве
   {
```

```
      return true; // Возвращаем значение true
        }
       }
       return false; // Возвращаем значение false
      }

      // Функция поднятия лифта
      void ElevatorUp()
      {
       Turn = 0; // Обнуляем значение переменной
       for (int i = Turn; i < ElevatorTurn; i++) // Поднимать лифт, пока значене прерывания не
достигнет заданной велечины
        {
         analogWrite (MotorSpeed, Speed);
         digitalWrite(MotorForward, HIGH);
         digitalWrite(MotorBackward, LOW);
        }
      }

      // Функция опускание лифта
      void ElevatorDown()
      {
       Turn = 0; // Обнуляем значение переменной
       for (int i = Turn; i < ElevatorTurn; i++) // Опускать лифт, пока значене прерывания не
достигнет заданной велечины
        {
         analogWrite (MotorSpeed, Speed);
         digitalWrite(MotorForward, LOW);
         digitalWrite(MotorBackward, HIGH);
        }
      }


      // Функция работы лифтов
      void Elevator()
      {
       if (digitalRead(ButtonPin_1) == HIGH) // Если нажата кнопка ВНИЗ на 1 этаже
       {
        ElevatorUp(); // Лифт поднимается на 1 этаж
       }
       if (digitalRead(ButtonPin_2) == HIGH) // Если нажата кнопка ВВЕРХ на 1 этаже
       {
        ElevatorUp(); // Лифт поднимается на 1 этаж
       }
       if (digitalRead(ButtonPin_3) == HIGH) // Если нажата кнопка ВНИЗ на 1 этаже
       {
        ElevatorDown(); // Лифт опускается на 1 этаж
```

```
 }
  if (digitalRead(ButtonPin_4) == HIGH) // Если нажата кнопка ВВЕРХ на 1 этаже
  {
    ElevatorUp(); // Лифт поднимается на 1 этаж
  }
  if (digitalRead(ButtonPin_5) == HIGH) // Если нажата кнопка ВНИЗ на 1 этаже
  {
    ElevatorDown(); // Лифт опускается на 1 этаж
  }
  if (digitalRead(ButtonPin_6) == HIGH) // Если нажата кнопка ВВЕРХ на 1 этаже
  {
    ElevatorDown(); // Лифт опускается на 1 этаж
  }
}
// Функция работы меток-RFID
void RFID()
{
  if (mfrc522.PICC_IsNewCardPresent())
  {
    // чтение карты
    if ( mfrc522.PICC_ReadCardSerial())
    {
      if (compare_uid(mfrc522.uid.uidByte, mfrc522.uid.size))
      {
        digitalWrite(RELAY_RC522_PIN, RELAY_ON);  // Включить реле
        delay(4000);
        digitalWrite(RELAY_RC522_PIN, RELAY_OFF); // Выключить реле
      }
      else
      {
        tone (PiezoPin, 500); // Включаем пьезоэлемента на 500 Гц
        delay(5000);
      }
    }
  }
  Elevator();
}

void loop()
{
  Sensors();
}
```