Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное
образовательное учреждение высшего образования
«Южно-Уральский государственный университет» (НИУ)
Высшая школа электроники и компьютерных наук
Кафедра «Инфокоммуникационные технологии»

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой
_____ Даровских С.Н
“____” _____ 2020 год

«Умный дом» на основе программно-аппаратной платформы Arduino IDE»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
ЮУрГУ – Д 11.03.02.2020.219.00 ПЗ ВКР

Руководитель работы:
Пискорский Д.С._____
“____” _____ 2020 год
Автор работы:
студент группы КЭ-411
Комлева Л.Т. _____
“____” _____ 2020 год
Нормоконтролёр:
Спицына В.Д. _____
“____” _____ 2020 год

Челябинск
2020

РЕФЕРАТ

Комлева Л.Т. «Умный дом» на основе программно-аппаратной платформы ArduinoIDE. - Челябинск: ЮУрГУ, КЭ-411; 2020, 80с., библиогр. список 22 наим., 24 рис., 2плаката

Целью данной выпускной квалификационной работы является разработка системы «Умный Дом» на основе программно-аппаратной платформы ArduinoIDE. В процессе выполнения будет рассмотрена концепция понятия «Умный Дом», будут разработаны структурная схема системы, алгоритмы общей программы и подпрограмм, подобрана элементная база, написаны тексты для программной среды ArduinoIDE.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Умный дом (англ. Smart House) - жилой дом современного типа, организованный для удобного и комфортабельного проживания людей при помощи новых современных высокотехнологичных устройств.

Само понятие «умный дом» сформировалось Институтом интеллектуального здания в Вашингтоне (округ Колумбия) еще в 1970-х годах: «Здание, обеспечивающее продуктивное и эффективное использование рабочего пространства.»

Принцип «Системы интеллектуального управления зданием» предполагает абсолютно новый подход в организации жизнеобеспечения строения, в котором за счет комплекса программно-аппаратных средств значительно возрастает эффективность функционирования и надежность управления всех систем и исполнительных устройств здания. [1]

В современном мире с каждым годом все больше и больше увеличивается тенденция к автоматизации повседневных задач. Ведь автоматизация окружающей среды делает нашу жизнь благоустроенной, удобной и во многом упрощает её.

«Умный Дом» это система, которая должна уметь распознавать конкретные ситуации, происходящие в здании, и подходящим образом на них реагировать: одна из систем может управлять поведением других по заранее выработанным алгоритмам. Основной особенностью интеллектуального здания является объединение отдельных подсистем в один единый управляемый комплекс.

Огромным преимуществом и свойством "Умного дома" отличающим его от других способов организации жизненного пространства является то, что это наиболее прогрессивная концепция взаимодействия человека с жилым пространством, когда человек одной командой задает желаемую обстановку, а уже автоматика в соответствии с внешними и внутренними условиями задает и отслеживает режимы работы всех инженерных систем и электроприборов. В доме, оборудованном системой "Умный дом" достаточно одним нажатием на смартфоне выбрать один из сценариев. Дом сам настроит работу всех систем в соответствии с

пожеланием потребностей человека, временем суток, положением человека в доме, погодой, внешней освещённостью и т. д. для обеспечения комфортного состояния внутри дома. [2]

Такая технология помогает не только экономить время, но и позволяют не зависеть от местонахождения. Например, в настоящее время человек всегда может быть уверенным в сохранности своего имущества в свое отсутствие, или может не вставать с дивана, чтобы включить или выключить свет. Также находясь на работе, он может регулировать температуру и открывать окна в доме. В этом всем помогают автоматизированные системы. Дополнительной привлекательностью является безопасность и надежность.

Технология «Умный дом» является современным инструментом повышения комфорта и уровня жизни, так как часть процессов происходит автоматически, а остальной частью можно управлять удаленно, что делает ее актуальной для изучения и совершенствования.

Одно из наиболее перспективных направлений развития технологий – это Интернет вещей, а Умный дом – его особенно приоритетная сфера. Неудивительно, что с каждым годом во всем мире интерес к интеллектуальным системам только растет – актуальность Умного дома давно не требует подтверждений. [3]

1 Аналитический раздел

1.1 Анализ технического задания и возможных способов реализации поставленной задачи

Согласно техническому заданию на выпускную квалификационную работу необходимо разработать систему «Умный Дом». Система предназначена для автоматизированной работы домашних устройств, которые выполняют свои определенные задачи без участия человека.

Основные возможности, которые должны быть реализованы в соответствии с ТЗ:

1) обеспечение безопасности путем идентификации пользователя с помощью электронного ключа (RFID метки) или по отпечатку пальца в совокупности с работой автоматической охранной сигнализацией не допустит несанкционированный вход чужих людей в дом;

2) автоматическое и дистанционное управление освещением позволит не только включать свет по необходимости, но и сэкономит средства и электрическую энергию;

3) автоматическое и дистанционное управление климатом, с помощью согласованной работы систем отопления, кондиционирования и открывания/закрывания окон позволят пользователю создать комфортные условия в доме;

4) управление бытовыми приборами со смартфона осуществляется дистанционно, что является большим удобством;

5) «умная кормушка» для домашних питомцев, разработана с автоматической выдачи порциии контролем времени кормления;

В соответствии с ТЗ система должна быть реализована на базе программно-аппаратной платформы ArduinoIDE, основными достоинствами которой являются:

– возможность автономной работы, обусловленная наличием собственного контроллера;

– широкие возможности по настройке работы системы (пользователь сам пишет программу, в которой могут быть предусмотрены сценарии любой сложности);

– простота процесса загрузки программы в контроллер: программатор для этого не требуется, достаточно иметь USB-кабель.

– доступная стоимость компонентов, обусловленная отсутствием у того или иного производителя монопольных прав (архитектура является открытой).

Целью данной работы является разработка системы управления “Умный-дом” на базе программно-аппаратной платформы Arduino IDE. Разрабатываемая система управления должна реализовывать следующие функции:

– идентификация пользователя с помощью электронного ключа (RFID метки) или по отпечатку пальца;

– охранная сигнализация: звуковой сигнал и информирование владельца при проникновении в дом (открытии двери);

– ручное и автоматическое регулирование уровня освещенности;

– ручное и автоматическое регулирование температуры;

– дистанционное управление бытовыми приборами;

– управление «Умной кормушкой».

Для достижения поставленной цели необходимо решить следующие задачи:

– изучить соответствующую теме литературу;

– выбрать место для реализации проекта;

– провести поиск и анализ существующих решений, выявить их достоинства и недостатки;

– разработать структурную схему системы;

– подобрать соответствующие компоненты: контроллер, сенсоры и вспомогательное оборудование;

– разработать алгоритмы управления и программы для микроконтроллера;

При выборе программных средств, для разработки программы «Умный Дом» на аппаратной платформе Arduino необходимо учитывать возможности описания структуры данных, определение модулей программы и связи между ними, оценки развитости аппарата структур и типов данных.

Учет этих возможностей позволит сделать систему легкодоступной для использования, позволит предупредить возникновение логических ошибок, обеспечить надежность программного обеспечения и его модифицируемость.

Реализация идеи и разработка системы «Умный дом» будет создана в небольшом жилом помещении – в двухэтажном доме площадью 114 кв. м^2, в котором на первом этаже расположено 3 комнаты: зал, ванная комната, и спальня. А на втором этаже 1 комната: гостиная. На рисунке 1 изображена схема планировки первого и второго этажей.

Рисунок 1 – Планировка первого и второго этажей дома

В данной работе необходимо использовать большое количество датчиков и управлять значительным числом устройств. Поэтому в качестве основного узла управления выбрана программно-аппаратная платформа ArduinoMega 2560 выполненная на базе контроллера ATmega 2560.

Структурная схема «Умного Дома»будет состоять из:

– центрального контроллера ArduinoMega 2560;

– комплекса управления освещением, в который входят: система внутренних датчиков света, система внешних датчиков света, блок управления световыми приборами, система датчиков движения света, и блок сервоприводов управления шторами;

– комплекса управления климатом, в который входят: система кондиционирования, система обогревателей, система внутренних температурных датчиков, система внешних температурных датчиков, и блок сервоприводов для управления окнами;

– комплекса обеспечения безопасности, в который входит: считыватель RFID меток, сканер отпечатка пальца, инфракрасный датчик, датчик открывания двери;

– комплекса управления бытовыми приборами;

– блока управления «Умной кормушки»;

– дисплея;

– Bluetooth модуля.

На рисунке 2 изображена структурная схема системы.

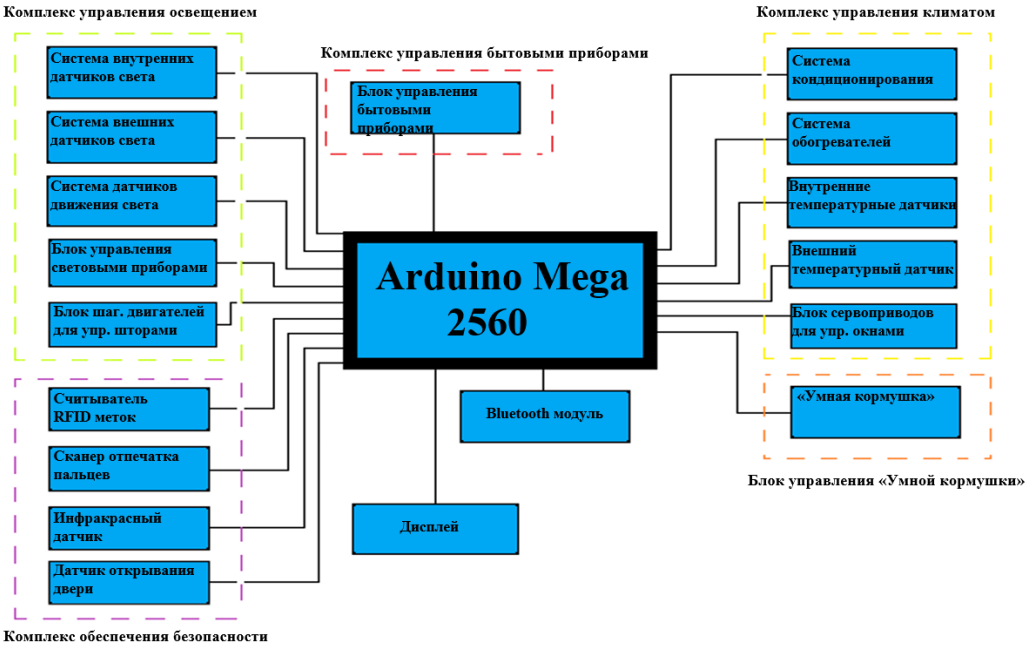

Рисунок 2 – Структурная схема системы

2 Выбор элементной базы

Для реализации данного проекта в качестве центрального контроллера был выбран ArduinoMega 2560. Мы выбрали именно этот контроллер, потому чтоArduino Megaимеет много входов/выходов, а это очень важно для нашей системы, так как мы будем управлять множеством устройств. Также, для этого контроллера несложно написать «язык общения», что является огромным преимуществом.

Плата ArduinoMega 2560выполнена на базе контроллера ATmega 2560. Она имеет 54 цифровых входа/выходов (14 из которых могут использоваться как выходы ШИМ), 16 аналоговых входов,4 последовательных порта UART, кварцевый генератор 16 МГц, USB коннектор, разъем питания, разъем ICSP и кнопка перезагрузки. Для работы необходимо подключить платформу к компьютеру посредством кабеля USB или подать питание при помощи адаптера AC/DC, или аккумуляторной батареей. Далее представлена последовательность подключений к плате:

– 8, 14, 6 - номера аналоговых пинов, к которым подключены датчики освещенности;

– 9, 15, 7 - номера аналоговых пинов, к которым подключены датчики температуры;

– 12, 31, 24 - номера пинов, к которым подключены светодиоды, демонстрирующие работу освещения;

– 29, 30, 26 - номера пинов, к которым подключены светодиоды, демонстрирующие работу обогревателей;

– 27, 34, 13 - номера пинов, к которым подключены светодиоды, демонстрирующие работу кондиционеров;

– (45, 50, 44, 51),(47, 48, 46, 49),(41, 38, 40, 39),(37, 42, 36, 43) - номера пинов, к которым подключены шаговые двигатели (по 4 на каждую штору);

– 4, 9, 8, 6, 10, - номера пинов, к которым подключены сервоприводы (первые 4 для окон, последний для кормушки);

– UART - порт для подключения дисплея;

– UART1 - порт для bluetooth модуля;

– UART2 - порт для подключения считывателя электронных ключей;

– UART3 - сканера отпечатков пальцев; [4]

В данном проекте, контроллер ArduinoMega2560 будет являться мозгом системы, так как ему стоит управлять следующими функциями (рисунок 3):

– предоставление команды на следующие устройства: сервоприводы для открывания/закрывания окон, шаговые двигатели для поднятия/опускания штор, включение/отключение систем отопления и кондиционирования, включение/отключение освещения;

– отображение принятых данных на сенсорном дисплее, передача их на смартфон;

– прием данных настроек и команд, задаваемых пользователем в специальном приложении на смартфоне;

– совместная обработка данных для автоматического поддержания заданных пользователем параметров;

– сбор данных о текущих параметрах и состоянии дома, путем периодического опроса датчиков освещенности, температуры, электронного ключа, сканера отпечатков пальцев, датчика движения и открытия двери;

– управление функциями «Умной кормушки».

На рисунке 3 показана плата управления системы. [5]

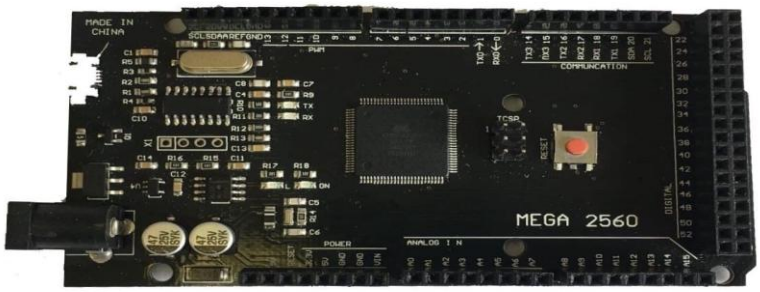

Рисунок 3 – Плата ArduinoMega 2560

2.1 Обеспечение безопасности

Путем идентификации пользователя с помощью электронного ключа (RFID метки) или по отпечатку пальца и автоматической охранной сигнализации обеспечивается безопасность нашего дома.

Первый барьер безопасности – это обеспечение надежной защиты жилища. Поэтому, одной из главных составляющих системы «Умный дом» является охранная сигнализация. Она предупреждает о несанкционированном доступе, а также способна принимать участие в предотвращении преступления против имущества и хозяев дома.

В состав системы обеспечения безопасности путем идентификации пользователя с помощью электронного ключа (RFID метки) или по отпечатку пальца и автоматической охранной сигнализации в помещении входят такие элементы как: считыватель RFID меток, сканер отпечатков пальцев, инфракрасный датчик, датчик открывания двери.

В некоторых проектах в целях обеспечения безопасности и допуска нужных пользователей может быть аутентификация по отпечаткам пальцев или с помощью электронного ключа.

Сканеры отпечатков пальцев — это ещё один вид современной защиты своего жилища, часть системы умного дома. В данной работе, мы будем использовать сканер отпечатка пальца FPM10A. Оптический сканер отпечатка пальца FPM10A построен на процессоре ARM Cortex M 32-bit — Synochip AS608 (FPM10A).

Этот сканер поддерживает алгоритмы шифрования данных, создает базу отпечатков во внутренней памяти, производит сравнение по шаблону. Сканер может управляться компьютером – такой вариант позволяет не только сканировать отпечатки, записывать их в память и сравнивать с имеющимся, но и просматривать снимки отпечатков. Также он может хорошо работает с Arduino, что будет реализовано в нашем проекте. Этот вариант удобен тем, что позволяет использовать сканер в автономных, независимых от компьютера устройствах

Характеристики:

– напряжение питания: 3.6 … 6.0 В;
– потребляемый ток: 120 мА (140 мА макс.);
– время обработки изображения отпечатка: менее 1,0 секунды;
– сенсор: оптический;
– размер сенсора: 14 мм х 18 мм;
– размер сигнатуры: 256 байт;
– размер шаблона: 512 байт;
– емкость: 300 ячеек;
– уровни безопасности (1-5);
– интерфейс: UART TTL;
– скорость передачи данных: 9600, 19200, 28800, 38400, 57600 (по умолчанию 57600);
– рабочая температура: -20 С …+50 С;
– допустимый уровень влажности: 40… 85 % RH;
– габаритные размеры: 45 х 26 х 19 мм;
– вес: 15 грамм;

Модуль имеет 6 проводов, два из которых не нужно подключать. TX и RX это линии передачи и приема данных UART соответственно, которые можно подключить ко 2-му и 3-му цифровому выводу платы Arduino соответственно.

Для работы с модулем сканера отпечатков пальцев необходимо добавить в Arduino IDE специальную библиотеку. Затем открыть Arduino IDE и можно зарегистрировать новый отпечаток пальца. В базе может храниться 127 различных отпечатков пальцев. [6]

Преимущества биометрического контроля доступа по отпечаткам пальцев - легкость в использовании, удобство и надежность, высокая достоверность и низкая стоимость устройств, сканирующих изображение отпечатка пальца. Сканер отпечатка пальца изображен на рисунке 4.

Рисунок 4 – Сканер отпечатка пальца

Ниже приведен текст программы:

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#define OLED_RESET 4
Adafruit_SSD1306 display(OLED_RESET);

#include <Adafruit_Fingerprint.h>
#include <SoftwareSerial.h>
SoftwareSerial mySerial(2, 3);

Adafruit_Fingerprint finger = Adafruit_Fingerprint(&mySerial);
int fingerprintID = 0;
String IDname;

void setup(){
```

```
  Serial.begin(9600);
  finger.begin(57600);

  if (finger.verifyPassword()) {
    Serial.println("Found fingerprint sensor!");
  }
  else {
    Serial.println("Did not find fingerprint sensor :(");
    while (1) { delay(1); }
  }

  Wire.begin();
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  displayMainScreen();
 }

 void loop(){
  displayMainScreen();
  fingerprintID = getFingerprintIDez();
  delay(50);
  if(fingerprintID == 1 || fingerprintID == 3 || fingerprintID == 4 || fingerprintID
== 5){
    IDname = "Sara";
    displayUserGreeting(IDname);
  }
  else if(fingerprintID == 2){
    IDname = "Rui";
    displayUserGreeting(IDname);
  }
 }
```

```
int getFingerprintIDez() {
  uint8_t p = finger.getImage();
  if (p != FINGERPRINT_OK)  return -1;

  p = finger.image2Tz();
  if (p != FINGERPRINT_OK)  return -1;

  p = finger.fingerFastSearch();
  if (p != FINGERPRINT_OK)  return -1;

  Serial.print("Found ID #");
  Serial.print(finger.fingerID);
  Serial.print(" with confidence of ");
  Serial.println(finger.confidence);
  return finger.fingerID;
}

void displayMainScreen(){
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(7,5);
  display.println("Waiting fingerprint");
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(52,20);
  display.println("...");
  display.display();
  delay(2000);
```

```
}

void displayUserGreeting(String Name){
  display.clearDisplay();
  display.setTextColor(WHITE);
  display.setTextSize(2);
  display.setCursor(0,0);
  display.print("Hello");
  display.setCursor(0,15);
  display.print(Name);
  display.display();
  delay(5000);
  fingerprintID = 0;
}
```

Для чтения RFID для чтения RFID меток бесконтактный считыватель – модуль RDM6300, который пользуется большой популярностью в связи с низкой ценой, наличием интерфейса UART TTL и неплохими характеристиками

Модуль RDM6300.RFID (Radio Frequency IDentification) представляет собой радиочастотную идентификацию, посредством которой можно осуществлять обмен данными между считывателем (сканером) и удобной меткой, которую можно носить с собой. Дальность действия такой связи может достигать 150 мм. Но это не все преимущества RFID устройств. Также, можно отметить, что они отличаются малым временем отклика (до 100 мс).

Модуль RDM6300 соответствует всему вышеописанному, также в комплекте к нему идет антенна (отвечает непосредственно за контакт с RFID устройствами). Он позволяет считывать идентификатор метки в виде массива данных, а затем, с помощью логических операторов, выполнять определенные действия. [7]

Считыватель можно установить скрыто в стену или в корпус для настенной установки. С помощью модуля RDM6300можно организовать свою собствен-

ную систему контроля доступа, что является большим преимуществом. Считыватель RFID меток изображен на рисунке 5.

Техническими характеристиками «RDM6300» являются:

а) рабочая частота 125 кГц;

б) битрейт 9600 (TTL уровень RS232);

в) интерфейс Wiegand26 или TTL уровень RS232;

г) потребляемый ток, мА: <50;

д) дальность считывания, мм: 20…50;

е) напряжение питания постоянного тока, В: 5 В;

ж) диапазон рабочих температур, оС: -10…+70;

з) размеры платы, мм: 38.5 х 19 х 9;

и) размеры антенны, мм: 45 х 35 х 3;

Подключение “RDM6300”:

Разъем P1:

- “TX” – передача данных – можно подключить к D2 пину Ардуино;
- “RX” – прием данных – не используется, не применяется;
- “GND” – общий провод;
- “+5V” – “+5 В” постоянного тока;

Разъем P2:

- ANT1 – подключение антенны;
- ANT2 – подключение антенны;

Разъем P3:

- LED;
- +5V (DC) – “+5” В постоянного тока;
- GND – общий; [8]

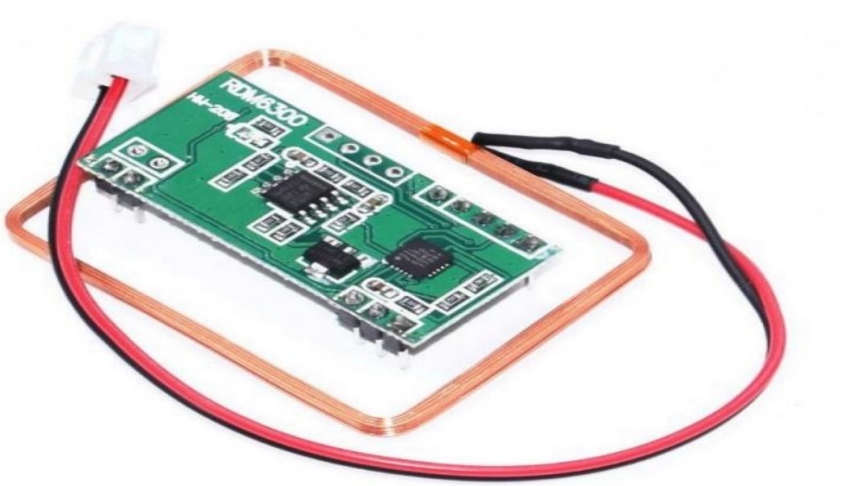

Рисунок 5 –Считыватель RFID меток RDM6300

Текст программы приведен ниже

```
// Arduino test sketch for http://arduino-ua.com/prod259-
125Khz_RFID_modyl_RDM6300
#include <SoftwareSerial.h>
#define STX 2
#define ETX 3

SoftwareSerial softSerial(10, 11); // recommended pins for RX on Mega: 10, 11, 12...
int rx_counter;
byte rx_data[14]; // 1+10+2+1

void setup() {
  rx_counter = 0; // init counter
  Serial.begin(9600);
  softSerial.begin(9600);
}

void loop() {
  if (softSerial.available() > 0) {
```

```
    rx_data[rx_counter] = softSerial.read();
    if (rx_counter == 0 && rx_data[0] != STX) {
      Serial.println("Out of sync!"); // do not increment rx_counter
    } else {
      rx_counter++;
    }
    if (rx_counter >= 14) {
      rx_counter = 0; // reset counter
      if (rx_data[0] == STX && rx_data[13] == ETX) { // packet starts with STX and
ends with ETX
        byte calc_checksum = 0;
        for (int i = 0; i < 6; i++) { // data with checksum
          calc_checksum ^= ascii_char_to_num(rx_data[i*2+1]) * 16 +
ascii_char_to_num(rx_data[i*2+2]);
        }
        if (calc_checksum == 0) {
          Serial.print("ID: ");
          for (int i = 1; i <= 10; i++) {
            Serial.write(rx_data[i]);
          }
          Serial.println();
        } else {
          Serial.println("Checksum ERROR!");
        }
      } else {
          Serial.println("Incorrect packet!");
      }
    }
  }
}
```

```
// convert a single hex character to its byte value using ASCII table (see
https://ru.wikipedia.org/wiki/ASCII)
byte ascii_char_to_num(byte a) {
  a -= '0'; // 0..9
  if (a > 9) a -= 7; // A..F
  return a;
}
```

Также мы будем использовать инфракрасный датчик движения HC-SR505. ИК-датчик движения HC-SR505 — инфракрасный датчик движения,который способен определять движение тёплых объектов (излучающих инфракрасные волны) в зоне своей чувствительности. Он подключается к цифровому входу микросхемы и отличается высокой точностью и надежностью. Принцип работы HC-SR505 основан на изменении характеристик поляризации под воздействием инфракрасного излучения. При появлении движущего объекта на сенсоре появляется электрическое поле и происходит срабатывание. Отключается устройство только спустя 8 секунд после удаления объекта с ИК-излучением.Малые габариты, низкая стоимость, простота эксплуатации и отсутствие сложностей в подключении - главные достоинства этого датчика. Данный датчик изображен на рисунке 6.

Технические характеристики:

- напряжение питания: 4,5В …20В;
- ток потребления: <60Ua;
- напряжение на выходе: 3.3V TTL логика;
- дистанция обнаружения: 3 м;
- угол детектирования: 80°…100;
- длительность импульса при обнаружении: 8 с;
- рабочая температура: -20…+80 °C;
- диаметр линзы: 10 мм;

– габаритные размеры платы: 23x10x10 мм;

Подключение мини датчика движения.

1) “-” – общий;

2) OUT – выход;

3) “+” – “+” питания;

Для подключения модуля датчика HC-SR501 будем использовать внешние прерывания на вход 18 платы Arduino Mega. Это прерывание int5.Загрузим на плату Arduino Mega скетч вывода счетчика срабатываний датчика HC-SR501 и вывода в последовательный порт Arduino. [9]

Рисунок 6 – Инфракрасный датчик HC-SR501

Ниже приведен текст программы:

```
int led = 13 ;// назначение пина для светодиода
int pirPin = 10; // назначение пина для мини ИК датчика
int value ;// переменная для хранения положения датчика
void setup ()
{
  Serial.begin(9600);
  pinMode (led, OUTPUT) ;// пин светодиода работает как выход
  pinMode (pirPin, INPUT) ; // пин датчика работает как вход
}
```

```
void loop ()
{
  value = digitalRead (pirPin) ;// чтение значения с датчика
  if (value == HIGH) // когда с ИК сенсора появляется высокий уровень, светодиод загорается
  {
    digitalWrite (led, HIGH);
    Serial.println("movement");
  }
  else
  {
    digitalWrite (led, LOW);
    Serial.println("no movement");
  }
  delay(1000);
}
```

Не мало важным элементом в этом комплексе является датчик открытия двери (дверной датчик), фиксирующий открытие и закрытие двери. Датчики открытия широко распространены как один из основных элементов домашней автоматизации и охранных систем, наряду с датчиками движения. В моем проекте этот датчик будет представлен в виде магнита и геркона.

В системах автоматики и защиты широко применяются различные датчики. Они работают на разных физических принципах. Например, хорошо известны датчики движения, срабатывающие дистанционно. Но в некоторых случаях необходим контроль событий на малых расстояниях. Например, если на несанкционированное открывание контролируются окна и двери помещений. Для этих целей идеально подходят датчики – герконы.

В закрытом состоянии магнит максимально приближен к геркону, который поэтому замкнут. При открывании магнит удаляется от датчика, цепь размыкает-

ся, и сигнализация срабатывает. Герметичный контакт, которым по сути является этот элемент, получил свое название от сокращения описывающего его словосочетания. [10]

Для работы с датчиком магнитного поля необходимо один из его выводов подключить с минусу источника питания, второй к выходу контроллера. Датчик установлен в дверном проеме, а магнит закреплен в соответствующем месте к двери. Выход контроллера, подключенный к данному датчику должен быть настроен таким образом, чтобы при отсутствии внешних воздействий с него считывалось значение логической единицы (необходимо подключить его через внутренний подтягивающий резистор к источнику напряжения 5 В). Датчик является нормально закрытым, поэтому при отсутствии магнитного поля не будет пропускать ток и напряжение на выходе контроллера будет соответствовать логической единице. При появлении внешнего магнитного поля датчик начинает проводить ток и напряжение на выходе становится равно нулю, при этом с контроллер считывает с выхода логический ноль. На рисунке 7 показан датчик движения. [11]

Рисунок 7 – Датчик движения

Ниже приведен текст программы:

```
#define  gerconPin  2 // задаем имя для порта 2
#define  ledPin  13     // задаем имя для порта 13
byte  gercon;
void setup() {
  Serial.begin(9600);
```

```
  pinMode(gerconPin, INPUT);
  pinMode(ledPin, OUTPUT);
}
void loop() {
  gercon = digitalRead(gerconPin);        // считываем данные с датчика
  Serial.print("Reed Switch Sensor - ");  // выводим данные на монитор
  Serial.println(gercon);
  if (gercon == HIGH)
    digitalWrite(ledPin, HIGH);
  else
    digitalWrite(ledPin, LOW);
}
```

2.2 Автоматическое управление освещением

Тенденция нашего времени – это путь к энергосбережению, поэтому я склоняюсь к такому мнению, что управление электрическим освещением является тривиальной функцией системы управления «Умный Дом». Простыми словами можно сказать, что в умном доме она служит для повышения уровня комфорта при включении света.

Основными функциямикомплекса управления уровнем освещения в нашем проекте–это способностьавтоматически и дистанционно включать/выключать световые приборы, а также поднимать/опускать шторы.

В состав комплекса управления освещения в помещении входят системы и блоки, такие как: система внутренних датчиков света, система внешних датчиков света, блок управления световыми приборами, система датчиков движения света.

В нашем двухэтажном доме площадью 114 кв. м^2 имеется 4 комнаты. На первом этаже 3 комнаты и на втором этаже 1 комната. По условию технического задания нужно разработать систему управления освещением во всем помещении.

Для реализации этой задачи нужно установить в каждой комнате по световому прибору. Во всем доме у нас 5 окон (3 окна на первом этаже и 2 окна на втором). Для повышения уровня комфортного освещения – в системе предусмотрена такая функция как поднимать/опускать шторы. Шторы есть на всех окнах, их 5 штук.

Так как, для выбора светильников никаких критериев нет, то сразу переходим к реализации функции автоматического и дистанционного освещения с помощью световых приборов. Для этого требуются датчики освещенности внутренние и внешние. В качестве датчиков внутренних и внешних мы выберем фоторезисторы и датчики интенсивности света.

Фоторезистор, как следует из названия, имеет прямое отношение к резисторам, которые часто встречаются практически в любых электронных схемах. Основной характеристикой обычного резистора является величина его сопротивления. От него зависят напряжение и ток, с помощью резистора мы выставляем нужные режимы работы других компонентов. Как правило, значение сопротивления у резистора в одних и тех же условиях эксплуатации практически не меняется.

В отличие от обычного резистора, фоторезистор может менять свое сопротивление в зависимости от уровня окружающего освещения. Это означает, что в электронной схеме будут постоянно меняться параметры, в первую очередь нас интересует напряжение, падающее на фоторезисторе. Фиксируя эти изменения напряжения на аналоговых пинах Arduino, мы можем менять логику работы схемы, создавая тем самым адаптирующиеся под вешние условия устройства.

Самый популярный и доступный фоторезистор – это модели (клоны) изделий производителя VT. Мы будем использовать VT93N2 — 48...500кОм (48K – освещенный, 100K – в темноте). На рисунке 8 изображен фоторезистор. [12]

Рисунок 8 – Фоторезистор VT93N2

Также, для функции управления освещением нужен датчик интенсивности света. Датчик интенсивности света используется для измерения интенсивности, падающего на его чувствительный элемент, света.

Для использования датчика нужно собрать на его основе макет: подключить питание, подключить к контроллеру, поместить датчик в нужное место. Потом нужно записать на контроллер специальную программу, которая позволяет работать с датчиком. После этого можно начинать работу. На корпусе платы расположено отверстие для закрепления датчика болтом или шурупом на поверхности. Принцип работы чувствительного элемента основан на фотогальваническом эффекте. На плате есть красный светодиод PWR-LED, который горит, когда на датчик подается питание. На плате датчика есть зеленый светодиод DO-LED, который горит, когда на цифровом выходе низкий уровень сигнала.

Управление датчиком осуществляется или от Arduino контроллера, или от другого управляющего микропроцессорного устройства с помощью специальных программ. На плате расположен переменный резистор, который используется для настройки порога срабатывания датчика чувствительности. Датчик интенсивности света имеет один интерфейс для подключения к питанию и микроконтроллеру. Штыревой разъем имеет 4 вывода. Обозначение выводов: GND (общий контакт), AO (аналоговый выход), DO (цифровой выход), VCC (напряжение питания). Выходным DO является цифровой сигнал. Высокий уровень сигнала означает наличие света, низкий - отсутствие света.

Выходным AO является аналоговый сигнал, который обратно пропорциональный интенсивности света окружающей среды. Питание датчика осуществляется или от Arduino контроллера, или от другого управляющего микропроцессорного устройства, или внешнего источника питания (блока питания, батареи). Напряжение питания датчика 3,3…5В. Датчик интенсивности света изображен на рисунке 9.

Характеристики[/]:

1) возможность регулировки чувствительности датчика: с помощью переменного резистора;

2) датчик собран на микросхеме: LM393;

3) светодиодная индикация питания;

4) штыревой интерфейс имеет 4 вывода: GND, AO, DO, VCC;

5) напряжение питания: 3,3… 5 В;

6) габариты модуля: 53,1 х 11,3 х 13,8 мм;

7) вес модуля: 3 г. [13]

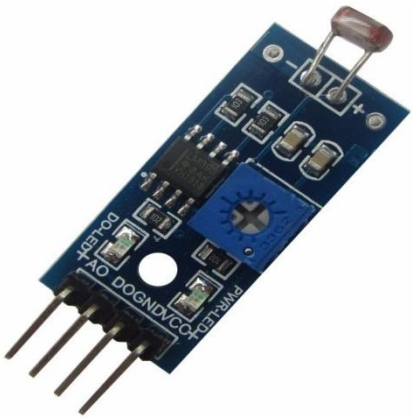

Рисунок 9 –Датчик интенсивности света

Главные преимущества вышеперечисленных двух элементов это простота и доступность. Прямое изменение сопротивления в зависимости от попадающего на неё света позволяет упростить электрическую схему подключения. Сам фото-

резистор очень дешев, входит в состав многочисленных наборов и конструкторов Arduino.

Проекты с применением датчика освещенности на базе фоторезистора достаточно просты и эффектны. Подключение фоторезистора осуществляется по схеме делителя напряжения с дополнительным сопротивлением. Датчик подключается к аналоговому порту для измерения различных значений уровня освещенности или к цифровому, если нам важен лишь факт наступления темноты. В тексте программы мы просто считываем данные с аналогового (или цифрового) порта и принимаем решение, как реагировать на изменения. Данные устройства будут эксплуатироваться как внешние и внутренние датчики освещенности. Ниже приведены тексты программ для совместной работы фоторезистора и датчика интенсивности:

Определяем уровень сигнала с аналогового пина.

Сравниваем уровень с пороговым значением. Максимально значение будет соответствовать темноте, минимальное – максимальной освещенности. Пороговое значение выберем равное 300.

Если уровень меньше порогового – темно, нужно включать светодиод.

Иначе – выключаем светодиод.

```
#define PIN_LED 13
#define PIN_PHOTO_SENSOR A0
void setup() {
  Serial.begin(9600);
  pinMode(PIN_LED, OUTPUT);
}
void loop() {
  int val = analogRead(PIN_PHOTO_SENSOR);
  Serial.println(val);
  if (val < 300) {
    digitalWrite(PIN_LED, LOW);
  } else {
```

```
    digitalWrite(PIN_LED, HIGH);
  }
}
```

Яркость лампочки будем менять через ШИМ, посылая с помощью analogWrite() на пин со светодиодом значения от 0 до 255.

Для преобразования цифрового значения уровня освещения от датчика освещенности (от 0 до 1023) в диапазон ШИМ яркости светодиода (от 0 до 255) будем использовать функцию map().

```
#define PIN_LED 10
#define PIN_PHOTO_SENSOR A0
void setup() {
  Serial.begin(9600);
  pinMode(PIN_LED, OUTPUT);
}
void loop() {
  int val = analogRead(PIN_PHOTO_SENSOR);
  Serial.println(val);
  int ledPower = map(val, 0, 1023, 0, 255); // Преобразуем полученное значение в уровень PWM-сигнала. Чем меньше значение освещенности, тем меньше мощности мы должны подавать на светодиод через ШИМ.
  analogWrite(PIN_LED, ledPower); // Меняем яркость
```

В случае другого способа подключения, при котором сигнал с аналогового порта пропорционален степени освещенности, надо будет дополнительно «обратить» значение, вычитая его из максимального:

```
int val = 1023 – analogRead(PIN_PHOTO_RESISTOR);
```

Для открывания и закрывания штор понадобится шаговые двигатель 28BYJ-48 с драйвером ULN2003. В функции управления открыванием и закрыванием штор выбран именно этот элемент, потому что данный шаговый двигатель

очень миниатюрен и отличается низкой стоимостью, также он хорошо работает на Arduino.

28BYJ-48 — это маленький 5 вольтовый шаговый моторчик с редуктором. Передаточное число редуктора у него примерно 64:1, что позволяет получить вполне достойный крутящий момент для моторчика такого размера и скорость порядка 15 об/мин. Шаговый двигатель может точно перемещаться на минимально возможный угол, называемый шагом. Для практических задач можно считать, что шаговый мотор немного похож на сервопривод. Можно задать ему повернуться в некоторое положение и можно рассчитывать получить достаточно стабильные результаты в нескольких повторных экспериментах. Обычно, сервоприводы ограничены углом поворота в диапазоне от 0 до 180°, шаговый же двигатель может вращаться непрерывно, подобно двигателю постоянного тока.

Преимуществом шаговых двигателей является то, что можно достичь гораздо большей степени контроля над движением. К недостатку шаговых двигателей можно отнести несколько более сложное управление, чем в случаях с сервоприводами или моторами постоянного тока.

Характеристика шагового двигателя 28BYJ-48:

а) тип мотора: униполярный шаговый двигатель;

б) число фаз: 4;

в) подключение: 5-выводов (к контроллеру двигателя);

г) рабочее напряжение: 5…12 вольт;

д) частота: 100 Гц;

е) сопротивление изоляции:> 10 МОм (500 В);

ж) шум: <35 дБ (120 Гц, без нагрузки, 10 см);

з) режим шага: рекомендован полушаговый режим (8-шаговая управляющая сигнальная последовательность);

и) угол шага:

1) полушаговый режим: 8-шаговая управляющая сигнальная последовательность. 5.625 градусов на шаг, 64 шага на оборот внутреннего вала мотора;

2) режим полного шага: 4-шаговая управляющая сигнальная последовательность. 11.25º/шаг, 32 шага на оборот внутреннего вала двигателя;

к) передаточное отношение редуктора: 64:1 ≈ 4076 шагов на полный оборот;

л) подключение к контроллеру: ULN2003 A (синий), B (розовый), C (желтый), D (Оранжевый), E (красный, средний вывод обмоток);

м) вес: 30г.

Ниже приведен текст программы:

```
#include<AccelStepper.h>
 #define HALFSTEP 8
 // Определение пинов для управления двигателем
#define motorPin1  3 // IN1 на 1-м драйвере ULN2003
#define motorPin2  4 // IN2 на 1-м драйвере ULN2003
#define motorPin3  5 // IN3 на 1-м драйвере ULN2003
#define motorPin4  6 // IN4 на 1-м драйвере ULN2003
 // Инициализируемся с последовательностью выводов IN1-IN3-IN2-IN4
// для использования AccelStepper с 28BYJ-48
AccelStepper stepper1(HALFSTEP, motorPin1, motorPin3, motorPin2, motorPin4);
 void setup(){
  stepper1.setMaxSpeed(1000.0);
  stepper1.setAcceleration(100.0);
  stepper1.setSpeed(200);
  stepper1.moveTo(20000);
}
 void loop(){
```

```
 // Изменяем направление, если шаговый достигает заданного положения
  if(stepper1.distanceToGo()==0)
    stepper1.moveTo(-stepper1.currentPosition());
  stepper1.run();
}
```

Плата драйвера шагового двигателя на базе микросхемы ULN2003, представляющей собой массив транзисторов, включенных по схеме Дарлингтона, позволяет достаточно просто управлять мотором 28BYJ-48, используя микроконтроллер. В нашем случае, в качестве управляющего микроконтроллера мы выберем плату Arduino Mega с микроконтроллером ATmega2560. Помимо самой микросхемы ULN2003AN, на плате имеется пяти контактный разъем для подключения к шаговому двигателю и четыре светодиода, показывающих, какая из обмоток запитана в текущий момент времени.

Плата управления на базе ULN2003

Также сбоку расположен джампер (два вывода под четырьмя резисторами), установка которого позволяет подавать питание на шаговый двигатель. Подавать питание мотора от 5 В Arduino не рекомендуется, так как мотор может потреблять ток, превышающий возможности Arduino. Лучше использовать внешний 5-12 В источник питания, выдающий ток не менее 1 А. Четыре управляющих входа помечены как IN1-IN4 и должны быть подключены к четырем цифровым выводам Arduino. Требуется подключить выводы IN1, IN2, IN3 и IN4 к пинам 3, 4, 5 и 6 ArduinoMega. Положительный контакт источника питания необходимо подключить к выводу, помеченному как «+», а землю источника питания к выводу «-» на плате контроллера. [14] Если для питания Arduino и мотора используются различные источники питания, то необходимо объединить выводы «земля» источников вместе. Изображение шагового двигателя предоставлено на рисунке 10.

Рисунок 10 – Шаговый двигатель

Датчики движения света. В качестве датчика движения света мы будем использовать термистор. Термистор изменяет свое сопротивление при изменении температуры. Для определения температуры используется следующая схема подключения. Температура определяется путем считывания напряжения с данного входа контроллера и последующего его преобразования. Ниже приведены технические характеристики термистора:

- сопротивление при 25 °C: 10K ±1%;
- B25/50: 3950 ±1%;
- диапазон измеряемых температур от -55°C …125°C;
- диаметр: 3.5 мм / 0.13 дюйма;
- длина: 18 дюймов / 45 см; [15]

Термистор одним выводом подключен к источнику напряжения 5 В, другим к аналоговому входу контроллера и через резистор к минусу питания. Термистор изображен на рисунке 11.

Рисунок 11 – Термисторы

Ниже приведен текст программы:

```
// значение 'другого' резистора
#define SERIESRESISTOR 10000
// к какому пину подключается термистор
#define THERMISTORPIN A0
void setup(void) {
Serial.begin(9600);
}
void loop(void) {
float reading;
reading = analogRead(THERMISTORPIN);
Serial.print("Analog reading ");
Serial.println(reading);
// преобразуем полученные значения в сопротивление
reading = (1023 / reading) - 1;
reading = SERIESRESISTOR / reading;
Serial.print("Thermistor resistance ");
Serial.println(reading);
delay(1000);
}
```

2.3 Управление климатом

В данном проекте в «Умном Доме» предусмотрена такая функция как управление климатом. Задача климат-контроля — одна из самых распространенных задач, решаемых в современном доме. В нашем проекте суть этой задачи сводится к согласованию работы системы отопления и кондиционирования и открывания окон. При грамотном ее решении, управление климатом в доме становиться простым, удобным и надежным. Контроллер снимает показания температуры и сравнивает их с заданной пользователем. Если температура внутри комнаты выше заданной, то система принимает решение понизить температуру. Если же температура становится ниже заданной, то система должным образом повышает температуру в помещении.

В состав комплекса управления климатом в помещении входят такие блоки как: система кондиционирования, система обогревателей, блок сервоприводов для открытия/закрытия окон, внешние и внутренние датчики температуры.

В роли внутреннего и внешнего температурного датчика выбран DS18B20. DS18B20 – это цифровой температурный датчик, обладающий множеством полезных функций, это целый микроконтроллер, который может хранить значение измерений, сигнализировать о выходе температуры за установленные границы (сами границы мы можем устанавливать и менять), менять точность измерений, способ взаимодействия с контроллером и многое другое. Сам температурный датчик изготовлен в очень небольшом корпусе, и что является огромным преимуществом - в защитном исполнении.

Микросхема имеет три выхода, из которых для данных используется только один, два остальных – это земля и питание. Число проводов можно сократить до двух, если использовать схему с паразитным питанием и соединить Vdd с землей. К одному проводу с данными можно подключить сразу несколько датчиков DS18B20 и в плате Arduino будет задействован всего один пин.

Особенности цифрового датчика DS18B20:

1) погрешность измерения не больше 0,5 С (для температур от -10С…85С), что позволяет точно определить значение температуры;

2) температурный диапазон измерений лежит в пределах от -55 …+125 °С;

3) датчик питается напряжением от 3,3 … 5 В;

4) можно программно задать максимальную разрешающую способность до 0,0625С, наибольшее разрешение 12 бит;

5) можно подключить сразу до 127 датчиков к одной линии связи;

6) информация передается по протоколу 1-Wire;

7) для присоединения к микроконтроллеру нужны только 3 провода;

Датчик DS18B20 является цифровым. Цифровые датчики передают значение измеряемой температуры в виде определенного двоичного кода, который поступает на цифровые или аналоговые пины Arduino и затем декодируется. Коды могут быть самыми разными, ds18b20 работает по протоколу данных 1-Wire. Обмен информацией в 1-Wire происходит благодаря следующим операциям: инициализация, запись данных,чтение данных. [16] На рисунке 12 показан температурный датчик в корпусе.

Рисунок 12 – Температурный датчик в корпусе

Ниже приведен текст программы:

```
#include <OneWire.h>
/*
* Описание взаимодействия с цифровым датчиком ds18b20
* Подключение ds18b20 к ардуино через пин 8
*/
OneWire ds(8); // Создаем объект OneWire для шины 1-Wire, с помощью которого будет осуществляться работа с датчиком
void setup(){
  Serial.begin(9600);
}
void loop(){
  // Определяем температуру от датчика DS18b20
  byte data[2]; // Место для значения температуры
    ds.reset(); // Начинаем взаимодействие со сброса всех предыдущих команд и параметров
  ds.write(0xCC); // Даем датчику DS18b20 команду пропустить поиск по адресу. В нашем случае только одно устрйоство
  ds.write(0x44); // Даем датчику DS18b20 команду измерить температуру. Само значение температуры мы еще не получаем - датчик его положит во внутреннюю память
    delay(1000); // Микросхема измеряет температуру, а мы ждем.
    ds.reset(); // Теперь готовимся получить значение измеренной температуры
  ds.write(0xCC);
  ds.write(0xBE); // Просим передать нам значение регистров со значением температуры
  // Получаем и считываем ответ
  data[0] = ds.read(); // Читаем младший байт значения температуры
  data[1] = ds.read(); // А теперь старший
  // Формируем итоговое значение:
  //    - сперва "склеиваем" значение,
```

```
//     - затем умножаем его на коэффициент, соответсвующий разрешающей способности (для 12 бит по умолчанию - это 0,0625)
  float temperature =  ((data[1] << 8) | data[0]) * 0.0625;

// Выводим полученное значение температуры в монитор порта
  Serial.println(temperature);
```

Для осуществления функции открытия/закрытия окон выбран сервопривод TowerPro MG995 Metal gear Servo 180°

Электронный (цифровой) сервопривод Tower Pro MG995 (MG 995, MG-995) Digi Hi-Speed - цифровая высокоскоростная сервомашинка с большим усилием. Идеально подходит для установки где необходимо большое усилие. [17]

Характеристики:

1) вес: 55 гр.;

2) размер: 40,7 х 19,7 х 42,9 мм;

3) скорость: 0,20 сек/60 гр. (4,8V), 0,16 сек/60 гр. (6V);

4) угол поворота: 120 градусов;

5) усиление: 8,5 кг/см (4.8V), 10 кг/см (6V);

6) рабочее напряжение: 4.8V… 7.2V;

7) рабочая температура: 0 C… 55 C;

8) разъем: JR (FitsJRandFutaba); [18]

Шестеренки из металла.Датчики подключены к следующим контактам Arduino:

1) PIN 2 - датчик внутреннего вида;

2) PIN 4 - датчик, расположенный в проеме окна;

3) PIN 6 - датчик, выходящий наружу;

4) PIN 10 - провод управления серводвигателями;

5) PIN 13 - светодиод, который горит, когда окно находится в движении;

На рисунке 13 изображен сервопривод.

Рисунок 13 – Сервопривод

Ниже приведен текст программы:

```
#include <Servo.h>
const int insidePin = 2;     // sensor inside
const int betweenPin = 4;     // sensor between
const int outsidePin = 6;      // sensor outside
const int ledPin =  13;       // the number of the LED pin
Servo servo1;  // create servo object to control a servo, a maximum of eight servo objects can be created
int pos1 = 0;    // variable to store the servo1 position
int windowState = HIGH;
int insideState = HIGH;
int outsideState = HIGH;
int betweenState = HIGH;
int helpState = HIGH;
void setup()
{
  servo1.attach(10);  // attaches the servo1 on pin 10 to the servo object
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
  // initialize the sensors pins as an input:
  pinMode(insidePin, INPUT);
  pinMode(outsidePin, INPUT);
```

```
  pinMode(betweenPin, INPUT);
  servo1.write(180);            // tell servo to go to position window closed, set state low
  windowState = LOW;
}
void loop()
{
  // read the state of the sensors:
  // First INSIDE sensor
  delay(500);
  helpState = insideState;
  insideState = digitalRead(insidePin);
  if (insideState != helpState)
  {
    delay(1000);
    insideState = digitalRead(insidePin);
    if (insideState != helpState)
    {
      helpState = insideState;
    }
    else
    {
      insideState = helpState; // state has to be at least 10ms in changed -state to be changed
    }
  }
  //   **********************
  // then OUTSIDE sensor
  delay(500);
  helpState = outsideState; // store current state
  outsideState = digitalRead(outsidePin);
```

```
 if (outsideState != helpState)
 {
   delay(1000);
   outsideState = digitalRead(outsidePin);
   if (outsideState != helpState)
   {
     helpState = outsideState;
   }
   else
   {
     outsideState = helpState; // state has to be at least 1s in changed -state to be changed NOT changed
   }
 }
//    **********************
 // lastly BETWEEN sensor
 delay(500);
 helpState = betweenState; // store current state
 betweenState = digitalRead(betweenPin);
 if (betweenState != helpState)
 {
   delay(1000);
   betweenState = digitalRead(betweenPin);
   if (betweenState != helpState)
   {
     helpState = betweenState;
   }
   else
   {
```

```
    betweenState = helpState; // state has to be at least 1s in changed -state to be changed
    }
  }
  // check if the cat wants inside or outside. If it is so, the state of one sensor is LOW:
  if (((insideState == LOW) or (outsideState == LOW) or (betweenState == LOW)) and (windowState == LOW))
    // cat at either windowsill or between windows and window closed
   // open sesame!
  {
    for(pos1 = 180; pos1>=1; pos1-=1)      // goes from 180 degrees to 0 degrees
    {
      servo1.write(pos1);              // tell servo to go to position in variable 'pos'
      delay(15);                       // waits 15ms for the servo to reach each position
    }
    windowState = HIGH;
    digitalWrite(ledPin, HIGH);     // turn LED on
    delay (10000);                  // wait at least 10 secs
  }
  // maybe the cat is doing the things cats do elsewhere
  // if so, we may as well close the window...
  if ((insideState == HIGH) and (outsideState == HIGH) and (betweenState == HIGH) and (windowState == HIGH))
  {
    // close it will!
    for(pos1 = 0; pos1 < 180; pos1 += 1)   // goes from 0 degrees to 180 degrees
    {                                  // in steps of 1 degree
      servo1.write(pos1);              // tell servo to go to position in variable 'pos'
      delay(150);                      // waits 150ms for the servo to reach each position
    }
```

```
    windowState = LOW;            // no delay here, be prepared to open immediately
    digitalWrite(ledPin, LOW);    // turn LED off
  }
}
```

Управление системой кондиционирования и система обогревания будет осуществляться посредством управления электромагнитного реле SRD-05VDC-SL-C.

Характеристики

1) габариты: 19,1x15.3x15.5 мм;

2) коммутируемая нагрузка (активная): 240В/7А(AC) 28В/7А(DC);

3) коммутируемая нагрузка (индуктивная): 120В/3А(AC) 28В/3А(DC);

4) контакты 1 x DT - переключающие (1C);

5) номинальное напряжение катушки 5 В;

6) номинальный ток катушки 71,4 мА;

7) макс. коммутируемое напряжение (AC) 240 В;

8) макс. коммутируемый ток 7 А;

Реле фирмы SONGLE SRD-05VDC. Данное реле управляется напряжением 5V и способно коммутировать до 10A 30V DC и 10A 250V AC.

Реле имеет две раздельные цепи: цепь управления, представленная контактами (A1, A2) и управляемая цепь, контакты (1, 2, 3). Цепи никак не связаны между собой. Между контактами A1 и A2 установлен металлический сердечник, при протекании тока по которому к нему притягивается подвижный якорь(2). Контакты же 1 и 3 неподвижны. Стоит отметить что якорь подпружинен и пока мы не пропустим ток через сердечник, якорь будет удерживается прижатым к контакту 3. При подаче тока, как уже говорилось, сердечник превращается в электромагнит и притягивается к контакту 1. При обесточивании пружина снова возвращает якорь к контакту.

Подключение:

При включении контроллера выводы находятся в высокоомном состоянии, транзистор не открыт. Так как у нас транзистор p-n-p типа, то для его открытия

нужно подать на базу минус. Для этого используем функцию digitalWrite (pin, LOW); Теперь транзистор открыт и через управляющую цепь течет ток и реле срабатывает. Для отключения реле следует закрыть транзистор, подав на базу плюс, вызвав функцию digitalWrite (pin, HIGH)

Модуль имеет 4 вывода для подключения к Ардуино:

1) VCC: "+" питания;

2) IN или S: Вывод входного сигнала;

3) GND: "-" питания;

4) VCC на + 5 вольт на Ардуино. GND на любой из GND пинов--- ардуино;

Подключение устройств к модулю реле:

У модуля есть еще 3 контакта для подключения устройств:

1) NO — нормально разомкнутый (Normally Open);

2) NC — нормально замкнутый (Normally Closed);

3) COM — общий (Common);

К контактам NC и NO подключаются устройства, общий COM подключается к + питания. Когда реле выключено, общий контакт «COM» (common) будет подключен к нормально замкнутому контакту «NC» (Normally Closed). Когда же реле сработает «общий» контакт COM соединится с «нормально разомкнутым» контактом «NO» (Normally Open).

Механизм работы достаточно прост, и он заключается в следующем: чтобы ток пошел, нужно подать на пин логический ноль. Тогда реле будет разомкнуто, и произойдет включение кондиционера или батареи. Если подать логическую единицу на пин, то тогда реле будет замкнуто и произойдет выключение определенного прибора. [19]

На рисунке 14 показан электромагнитный реле SRD-05VDC-SL-C.

Рисунок 14 – Реле SRD-05VDC-SL-C

2.4 Дистанционное управление бытовыми приборами

Тренд последних лет в системе «Умный дом» — это управление бытовыми приборами. В нашем проекте будет разработана система управления, позволяющая дистанционно, с помощью приложения на смартфоне управлять бытовыми приборами.

В данном проекте управление бытовыми приборами будет осуществляться посредством «умной розетки» и реле SRD-05VDC-SL-C, которое использовалось в работе с кондиционерами и батареями.

С помощью розетки и реле SRD-05VDC-SL-C можно будет включать и отключать бытовой прибор.

Механизм работы точно такой же, как и в комплексе с управлением кондиционеров и батарей: он заключается в том, чтобы ток пошел, нужно подать на пин логический ноль. Тогда реле будет разомкнуто, и произойдет включение бытового прибора. А чтобы выключить бытовой прибор, то нужно подать на пин логическую единицу, тогда реле будет замкнута, ток перестанет идти и произойдет выключение прибора. [20]

Ниже приведен текст программы:

```
#include <SoftwareSerial.h>
SoftwareSerial BT(8, 9);         // Контакты подключения bluetooth модуля. 8 для TXD, 9 для RXD
int relayPin = 7;
```

```
char c = "";                 // Переменная для считывания символа из последовательного порта
String str = "";             // Переменная для формирования целой строки
void setup() {
  BT.begin(9600);            // Инициализация bluetooth последовательного порта
  Serial.begin(9600);        // Инициализация последовательного порта
  Serial.println("Test for Arduino");
  BT.println("Test for Phone");
  pinMode(rgb[0], OUTPUT);
  pinMode(rgb[1], OUTPUT);
  pinMode(rgb[2], OUTPUT);
  pinMode(relayPin, OUTPUT);
}
void loop(){
  if (BT.available()){       // На последовательном порту bluetooth есть что инициализировать
    delay(50);               // Задержка между считыванием символов
    c = BT.read();           // Считываем код символа в переменную типа int
    if (isDigit(c)) {        // Если считана цифра
      str += (char)c;        // Конвертируем байт в символ и присоединяем к строке
    }
        if (c == '\n') {     // Если обнаружен символ конца строки
      light = constrain(str.toInt(), 0, 1 );  // Ограничиваем значение для вывода на контакт
      digitalWrite(relayPin, light);
      Serial.println(light);                  // Выводим строку в монитор порта как цифру
```

```
                str = "";                    // Очищаем строку перед следующим вво-
дом
              }
            }
          }
```

2.5 «Умная кормушка» для домашних питомцев

Умная кормушка - это устройство, состоящее из лотка, в который засыпается еда для домашних питомцев. Благодаря автоматической кормушке у питомцев всегда будет корм в достаточном количестве.

«Умной кормушкой» это устройство будет называться тогда, когда оно сможет функционировать автоматически, а, чтобы она функционировала автоматически для этого нам требуется сервопривод Micro Servo Towerpro SG90, что уже использовался ранее в комплексах данного проекта.

Логика работы механизма проста: в крышке банки делается дырка типа сектор (центр крышки не вырезается), такой же сектор вырезается из пластмассы. Внутрь крышки прикрепляется серводвигатель. На ось двигателя насаживается сначала крышка, а с внешней стороны вырезанный сектор. Так, при повороте оси серводвигателя вырезанный сектор смещается относительно дырки в крышке банки.С помощьюсервопривода Micro Servo Towerpro SG90 будет осуществляться выдача порции в кормушку. Кормушка будет запрограммирована таким образом, что выдача порций будет осуществляться каждый 6 часов.

Подключение будет осуществляться следующим образом:

Для того, чтобы устройство заработало, требуется к плате Arduino Mega подключить сервопривод Micro Servo Towerpro SG90 на 10 вход платы. Также требуется написать тексты программы и загрузить их в библиотеку Arduino.

Ниже приведен текст программы:

```
#include <Servo.h>
#define servoPin 9
```

```
Servo myservo;
void setup() //процедура setup
{
myservo.attach(servoPin); //привязываем привод к порту 9
}
void loop()
{
myservo.write(0); //ставим вал под 0 градусов
delay(300); //ждем 0.3 секунды
myservo.write(160); //ставим вал под 160 градусов
delay(21600); //ждем 6 часов
}
```

2.6 Вспомогательные элементы в системе

2.6.1 Дисплей

Дисплеи Nextion это модули, оснащённые мощным 32 разрядным микроконтроллером, контроллером сенсорного экрана, флеш-памятью, часами реального времени и разъемами: SD-карты, шины UART, и выводов GPIO. Модули дисплеев способны самостоятельно обрабатывать поступающую информацию (касание элементов экрана и команды поступающие по шине UART), управлять элементами экрана и управлять внешними устройствами (отправлять команды по шине UART, управлять выводами GPIO).

Программа Nextion Editor для работы с дисплеями Nextion позволяет создать как интерфейс пользователя, так и прописать алгоритм поведения дисплея. Для проверки работы написанного кода нет необходимости загружать данные в дисплей, так как в программе имеется встроенный эмулятор, который отображает не только поведение элементов интерфейса, но и получает/отображает принимаемые/возвращаемые данные по шине UART.

Дисплей Nextion имеет простой интерфейс, он прост в управлении и в нем нет никаких лишних функций. Также, помимо интерфейса, можно отметить, что

сам дисплей можно приобрести за низкую стоимость. Дисплей изображен на рисунке 15.

Для подключения дисплея к Arduino можно воспользоваться как аппаратной шиной UATR (№ выводов TX и RX Arduino указаны на плате), так и программной шиной UART (№ выводов TX и RX Arduino назначаются в скетче). Вывод TX дисплея подключается к выводу RX Arduino, вывод RX дисплея подключается к выводу TX Arduino.

Для подключения дисплея к компьютеру нужен адаптер USB-UART. Вывод TX дисплея подключается к выводу RX адаптера, вывод RX дисплея подключается к выводу TX адаптера. Вместо отдельного адаптера USB-UART, можно использовать контроллер Arduino, о чем рассказано в статье Wiki - Используем Arduino как USB - UART преобразователь.

К выводам JPIO дисплея можно подключать устройства для управления ими, или получения данных с этих устройств.

Характеристики:

1) тип экрана: Сенсорный;

2) разрешение: 480 x 272;

3) интерфейс дисплея: Последовательный;

4) совместим с RaspberryPi (A+, B+ 2, 3) и Arduino;

5) RGB 65K true to life colours;

6) экран TFT с интегрированным резистивным сенсорным экраном с 4 проводами;

7) легкое подключение разъёма на 4 провода к TTLSerialHost;

8) 16M Флэш-памяти для пользовательского кода программы и данных;

9) карта памяти microSD для перепрошивки;

10) видимый экран:95.04mm(L)×53.86mm(W);

11) адаптивная яркость:0~230, интервал регулирования составляет 1%;

12) потребление энергии: 5V 250mA;

13) размер: 120(L)×74(W)×6.2(H);

14) вес: 93.8 гр;

Рисунок 15 – Дисплей Nextion

Ниже приведен текст программы:

```
#include <SoftwareSerial.h> // Подключаем библиотеку SoftwareSerial для работы с программным UART
const uint8_t pinRX = 4; // Определяем константу хранящую номер вывода Arduino RX программного UART, подключается к выводу TX дисплея
const uint8_t pinTX = 5; // Определяем константу хранящую номер вывода Arduino TX программного UART, подключается к выводу RX дисплея
const uint8_t pinVD1 = 6; // Определяем константу хранящую номер вывода Arduino к которому подключён первый светодиод
const uint8_t pinVD2 = 7; // Определяем константу хранящую номер вывода Arduino к которому подключён второй светодиод
const uint8_t pinR = A0; // Определяем константу хранящую номер вывода Arduino к которому подключён потенциометр
SoftwareSerial softSerial(pinRX,pinTX); // Объявляем объект softSerial с указанием выводов RX и TX по которым общаемся с дисплеем
//
void setup(){ //
```

```
//  Подготовка:                                                    //
   softSerial.begin(9600);                                        // Инициируем передачу данных по программному UART на скорости 9600 бит/сек
   pinMode(pinVD1, OUTPUT);                                       // Конфигурируем вывод pinVD1 как выход
   pinMode(pinVD2, OUTPUT);                                       // Конфигурируем вывод pinVD2 как выход
//  Устанавливаем состояние первого светодиода:
//
   softSerial.print((String) "print h0.val"+char(255)+char(255)+char(255));
// Отправляем команду дисплею: print h0.val заканчивая её тремя байтами 0xFF
   while(!softSerial.available()){}                               // Ждём ответа. Дисплей должен вернуть состояние слайдера h0, отправив 4 байта данных
   analogWrite(pinVD1, softSerial.read());        delay(10);      // Устанавливаем на выводе pinVD1 сигнал ШИМ с коэффициентом заполнения равным значению первого принятого байта ответа дисплея
   while(softSerial.available()){softSerial.read(); delay(10);}   // Читаем остальные 3 байта ответа ничего с ними не делая, так как установленный диапазон значений слайдера от 0 до 255 уместился в первом байте данных
//  Устанавливаем состояние второго светодиода:
//
   softSerial.print((String) "print bt0.val"+char(255)+char(255)+char(255));
// Отправляем команду дисплею: «print bt0.val» заканчивая её тремя байтами 0xFF
   while(!softSerial.available()){}                               // Ждём ответа. Дисплей должен вернуть состояние кнопки bt0, отправив 4 байта данных, где 1 байт равен 0x01 или 0x00, а остальные 3 равны 0x00
   digitalWrite(pinVD2, softSerial.read());       delay(10);      // Устанавливаем на выводе pinVD2 состояние в соответствии с первым принятым байтом ответа дисплея
   while(softSerial.available()){softSerial.read(); delay(10);}   // Читаем остальные 3 байта ответа ничего с ними не делая, так как состояние кнопки было передано в первом байте
}                                                                 //
                                                                  //
void loop(){                                                      //
   if(softSerial.available()>0){                                  // Если есть данные принятые от дисплея, то ...
```

```
      String str;                                                    // Объявляем
строку для получения этих данных
      while(softSerial.available()){str+=char(softSerial.read()); delay(10);}
// Читаем принятые от дисплея данные побайтно в строку str
      for(int i=0; i<str.length(); i++){                             // Прохо-
димся по каждому символу строки str
        if(memcmp(&str[i],"h0" ,2)==0){i+=5; analogWrite (pinVD1, str[i-
3]);}else        // Если в строке str начиная с символа i находится текст "h0",
значит после него следует 4 символа (байта) данных слайдера, первый из
который следует указать как ШИМ для вывода pinVD1
        if(memcmp(&str[i],"ON" ,2)==0){i+=1; digitalWrite(pinVD2, HIGH);
}else        // Если в строке str начиная с символа i находится текст "ON",
значит кнопка дисплея была включена, устанавливаем высокий уровень на
выводе pinVD2
        if(memcmp(&str[i],"OFF",3)==0){i+=2; digitalWrite(pinVD2, LOW);
}           // Если в строке str начиная с символа i находится текст "OFF",
значит кнопка дисплея была выключена, устанавливаем низкий уровень на
выводе pinVD2
      }                                                              //
    }                                                                //
    if(millis()%500<=5){delay(5);                                    // Сле-
дующее действие выполняем каждые пол секунды

softSerial.print((String)"t0.txt=\""+analogRead(pinR)+"\""+char(255)+char(255)
+char(255)); // Отправляем команду дисплею: «t0.txt="текст"» заканчивая её
тремя байтами 0xFF. Эта команда устанавливает значение для текстового
поля t0
    }                                                                // В качестве текста
передаётся значение в аналогового входа pinR считанное функцией
analogRead
}
```

В наш проект нужен Bluetooth модуль.Нередко в проектах возникает необходимость дистанционного управления или передачи данных с телефона или другого устройства. Одним из самых популярных и удобных способов является обмен данных через Bluetooth. Для связи платы Ардуино и компьютера используется интерфейс UART (Serial).

Самыми популярными модулями являются устройства на основе чипа BC417. Эта серия называется HC. Мы выбрали Bluetooth модуль HC-06.

Характеристики Bluetooth модуля:

1) Bluetooth V2.0 + EDR 3 Мбит;

2) радиус действия - 10 метров;

3) чувствительность -80dBm;

4) UART интерфейс с программируемой скоростью обмена;

5) низкое напряжение питания 1,8 до 3,6 В;

6) скорость по умолчанию: 38400, Data bits: 8, Stop bit: 1, Parity: нет.;

7) заводской пин код "0000";

8) полный набор AT команд;

Bluetooth модуль HC-06 предназначен для беспроводного соединения контроллеров Arduino, STM8, STM32 с другими устройствами, таких как телефон, смартфон, планшет, PC, ПК, Android и другие совместимые Bluetooth устройства. Данный Bluetooth модуль изображен на рисунке 16. Подключение Bluetooth HC-06 к Ардуино производится по UART к выводам RX и TX. Пин RX на ардуино подключается к TDX, TX – к RDX, GND – к GND, 5V – к VCC.

У Bluetooth модуля, есть несколько преимуществ, перед стандартными дополнениями под другие МК:

1) пользователю нет необходимости изучать технологию протокола блютуз, чтобы написать софт или начать использовать уже готовые библиотеки;

2) простота использования в целом. Не нужно будет паять отдельную плату под распределение мощностей, просто нужно подсоединить устройство к уже готовому МК через пины;

3) обширный выбор библиотек. Так как Ардуино имеет низкий порог вхождения, под все его модули можно найти большое количество библиотек, разного назначения; [22]

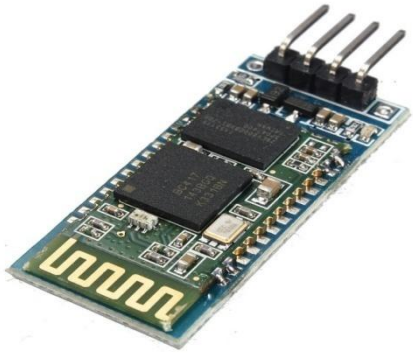

Рисунок 16 –Bluetooth модуль

Ниже приведен текст программы:

```
char incomingByte; // входящие данные
int LED = 12; // LED подключен к 12 пину
void setup() {
Serial.begin(9600); // инициализация порта
pinMode(LED, OUTPUT);
Serial.println("Press 1 to LED ON or 0 to LED OFF...");
}
void loop() {
if (Serial.available() > 0) { //если пришли данные
incomingByte = Serial.read(); // считываем байт
if(incomingByte == '0') {
digitalWrite(LED, LOW); // если 1, то выключаем LED
Serial.println("LED OFF. Press 1 to LED ON!"); // и выводим обратно сообщение
}
if(incomingByte == '1') {
digitalWrite(LED, HIGH); // если 0, то включаем LED
Serial.println("LED ON. Press 0 to LED OFF!");
}
```

}

}

Мы подобрали устройства, датчики и вспомогательное оборудование для системы «Умный Дом». При выборе мы учли возможности этих элементов, оценили связь между ними, тем самым предотвратили появление логических ошибок, а также сделали систему удобной в использовании и обеспечили ее надежность.

На этом, выбор элементной базы завершен. После того, как мы выбрали все подходящие устройства, можно написать алгоритмы общей программы и остальных подпрограмм.

Выводы по разделу два

……
……..

3 Алгоритмы

После того, как мы сделали подбор элементной базы, стоит разработать для системы «Умного Дома» алгоритмы, по которым этасистема будет запускаться и работать.

3.1 Алгоритм общей программы

«Алгоритм общей программы» будет являться является основой, так как он представляет совокупность всех функционирующих подпрограмм и других операций.

Начало алгоритма общей программы начинается с инициализации данных. После этого действия происходит настройка и подключение последовательных портов. На следующем этапе выполняется настройка и запуск таймера. Далее следует отсчет 200 мс, происходит опрос датчика и если отсчет пройден, то происходит вызов подпрограммы включения/выключения сигнализации. А если отсчет 200 мс не пройден, то снова происходит отсчет времени, чтобы по истечении 200 мс перейти к вызову подпрограммы включения/выключения сигнализации. После вызова подпрограммы включения/отключения сигнализации следует вызов подпрограммы управления светом. Далее вызов подпрограммы управления климатом. После следует вызов подпрограммы отображения данных на дисплее. Завершающим действием будет вызов подпрограммы приема данных от пользователя и отправки данных. После этого, следует конец цикла. На рисунке 17 изображен алгоритм общей программы.

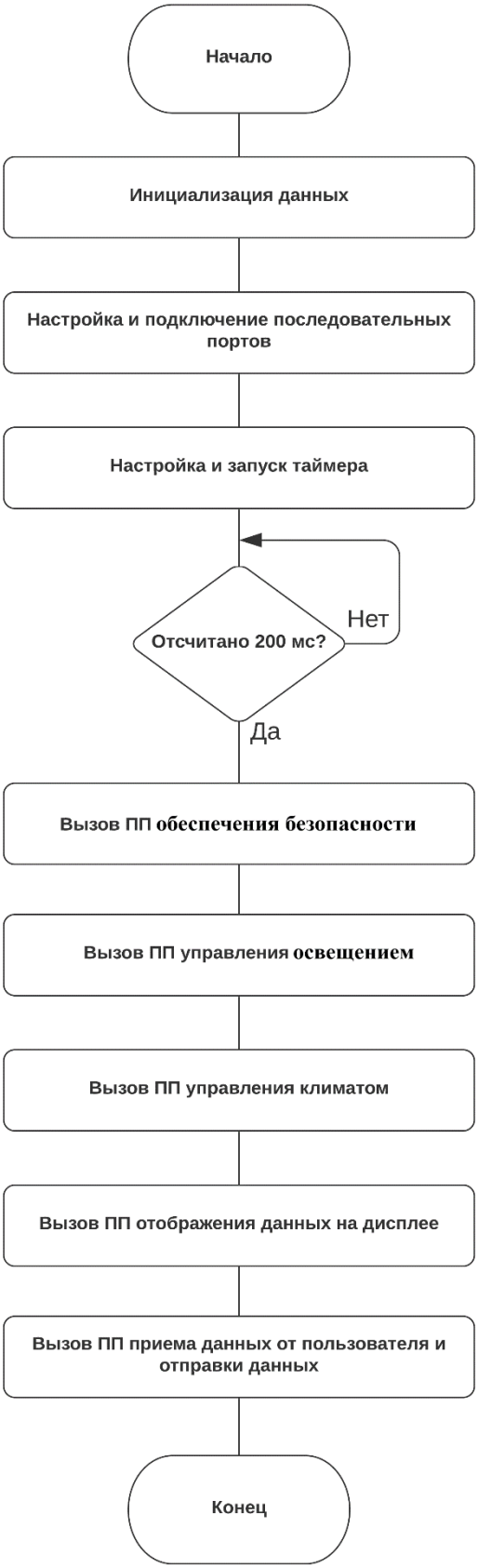

Рисунок 17 – Алгоритм общей программы

Комплекс обеспечения безопасности будет представлять собой два алгоритма подпрограмм: подпрограмму контроля доступа и подпрограмму включения/выключения сигнализации.

Подпрограмма контроля доступа в помещение.

Данная подпрограмма представляет собой функционирование нескольких подсистем и устанавливает контроль доступа пользователя в помещение, поэтому попасть в дом может только тот пользователь, чей отпечаток будет сохранен в базе данных сканера или тот, у кого будет электронный ключ.

Сканер отпечатка пальца —ScanFdata.

Сканер RFIDметки (эл. ключ) —ScanRFdata.

База данных (эл. ключей) —ScanFuser1…ScanFuser2.

База данных (эл. ключей)— ScanRFID1…ScanRFID2.

Начало алгоритма начинается с опроса сканера отпечатка пальца. Далее после опроса программа переходит к условию – совпадает ли отпечаток пальца с отпечатком из базы данных. В случае если отпечаток пальца обнаружен и совпадает с отпечатком из БД, то условие выполняется и цикл закончен. В случае невыполнения этого условия происходит опрос сканера RFID метки (электронного ключа). Если данный электронный ключ не обнаружен базой данных, то ничего не происходит и цикл заканчивается. А если электронный ключ обнаружен, то происходит очередной опрос датчиков – совпадают ли обнаруженный электронный ключ с данными из базы данных? Если это условие невыполнимо, то происходит конец цикла. А если выполнимо, то система снова опрашивает датчики – включена ли сигнализация (State_A=1)? Если сигнализация включена, то выполняется команда о выключении сигнализации (State_A=0) и конец цикла. А если сигнализация не включена и так как данная подсистема функционирует совместно с другими подсистемами, то поступает команда включить сигнализацию (State_A=1), закрыть окна (State_W=0), выключить свет (State_L=0), выключить приборы (State_P=0), включить датчики движения (State_M=1) и конец цикла. На рисунке 18 изображена подпрограмма контроля доступа.

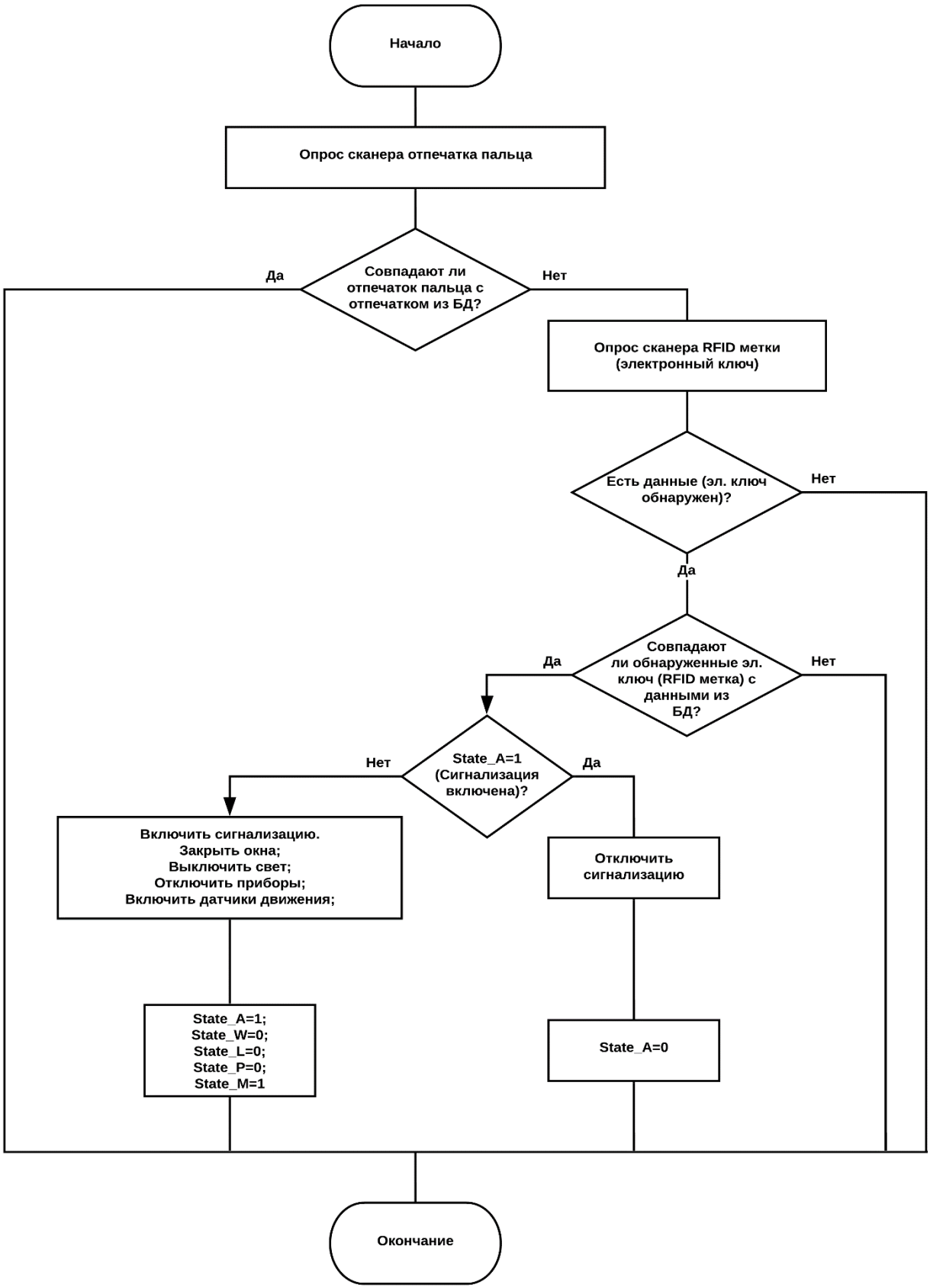

Рисунок 18 – ПП контроля доступа

Подпрограмма включения/выключения сигнализации

В данной подпрограмме ключевыми элементами будут являться датчик движения и датчик открывания двери.

Датчики движения —Motsens. Есть движение «1», нет движения «0».

Датчик открытия двери —DoorSens. Дверь открыта «1», дверь закрыта «0».

Начало алгоритма начинается с опроса датчиков. Далее, после опроса датчиков программа переходит к условию включена ли сигнализация на данный момент. Если сигнализация выключена (State_S=0), то цикл закончен. Если же сигнализация включена (State_S=1), то происходит опрос датчика движения. Если датчик заметил какое-либо движение (MotSens=1), то подается сигнал тревоги и цикл на этом заканчивается. А если движения датчиком не обнаружены, то происходит опрос датчика открывания двери. Если же дверь открыта (DoorSens=1), то подается сигнал тревоги и заканчивается цикл. Если же дверь закрыта, то без каких-либо действий происходит конец цикла. На рисунке 19 представлен алгоритм подпрограммы включения-выключение сигнализации.

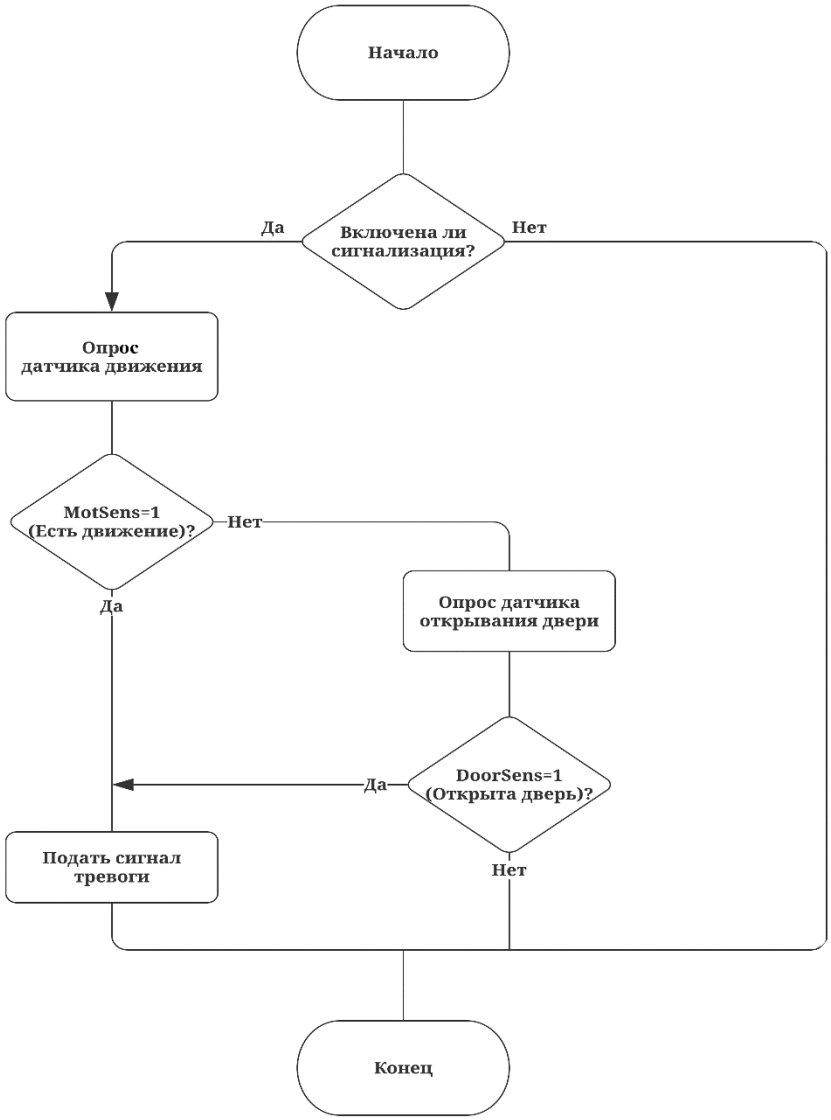

Рисунок 19– ПП включения-выключения сигнализации

3.1.1 Описание подпрограммы управления освещением

Начало алгоритма начинается с расшифровки всех вышеперечисленных параметров. Программа начинает проводить опрос всех датчиков данного алгоритма, чтобы узнать в каком состоянии они находятся в данный момент. В этой подпрограмме пользователь сам задает комфортную для него освещенность в помещении, а также пользователь вправе управлять этой системой автоматически. Для изменения освещения будут задействованы такие команды как включить/выключить свет и поднять/опустить шторы.

Пользователь задает в своем доме комфортную для него освещенность, которая должна поддерживаться системой, а также пользователь создает заданный допуск «Δ». Это небольшой интервал изменения освещенности, при котором датчики не будут срабатывать, если заданное освещение станет чуть больше или меньше.

Если текущая освещенность в комнате с заданным допуском «Δ» соответствует заданной пользователем освещенности, то в помещении, как и прежде без изменений будет поддерживаться такая же освещенность, до тех пор, пока какие-нибудь факторы не повлияют на ее изменение (Условие 1).

Lул —освещенность на улице

Lк — освещенность в комнате

Блок управления освещением: State_L. Свет включен «1», свет выключен «0».

Блок управления шторами: State_J. Шторы подняты «1», шторы опущены «0».

Пользовательские настройки: Lз — заданная освещенность. «Δ»— интервал нерегулирования (допуск).

Allow_J — автоматическое управление шторами разрешено «1», не разрешено «0».

В случае, если это условие не выполняется, то происходит опрос датчиков – больше ли уровень освещения в комнате Lк уровня заданного освещения Lз с заданным допуском «Δ»?(Условие 2). Если это условие выполняется, значит, нужно уменьшать уровень освещения в комнате. Это значит, что нужно снова опросить датчики освещенности, а конкретно узнать превышает ли уровень освещенности на улице Lул уровень освещенности в комнате Lк, открыты ли шторы (State_J=1) и может ли пользователь автоматически ими управлять (Allow_J=1). Если эти условия выполняются, значит система автоматически принимает решение опустить шторы (State_J=0), чтобы понизить уровень освещения в помещении. Если же, последнее условие не выполнимо, то происходит еще один опрос датчиков, при котором система узнает- включено ли освещение (State_L=1) или

нет. Если освещение выключено, то происходит конец цикла. Если освещение не включено-система принимает решение выключить свет (State_L=0) и только тогда происходит конец цикла.

Сейчас, возвращаемся к условию 2 где есть другая ветвь. Снова происходит опрос датчиков – больше ли уровень освещенности в комнате Lк заданного уровня освещенности с заданным допуском «Δ»? Так как это условие может быть не выполняется, значит, в помещении темно и следует увеличить освещенность. Далее происходит очередной опрос датчиков, при котором системе нужно узнать превышает ли уровень освещенности Lул уровень освещения в помещении Lк, закрыты ли шторы (State_J=0) и можно ли автоматически ими управлять (Allow_J=1). Если эти условия выполнимы, то система принимает решение поднять шторы (State_J=1) и происходит конец цикла. Если же, это условие не выполнимо, тогда происходит опрос датчиков освещения, так как они должны быть в выключенном положении. Если же, они выключены, то произойдет включение датчиков света (State_L=1) и конец цикла. Если освещение было выключено, то без каких-либо действий сразу происходит конец цикла. Подпрограмма управления освещением изображена на рисунке 20.

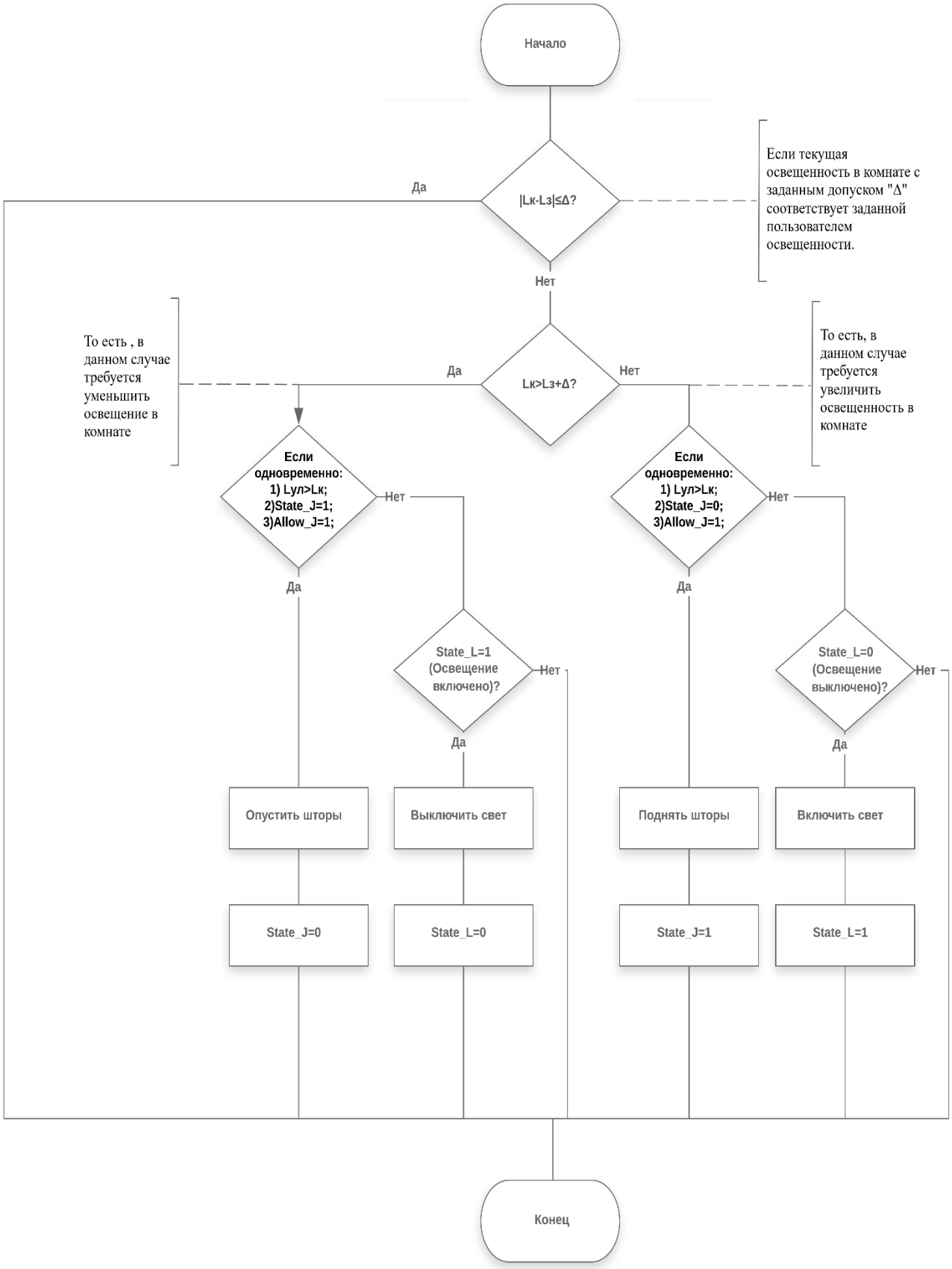

Рисунок 20 – ПП управления освещением

3.2 Алгоритм подпрограммы управления климатом

3.2.1 Описание подпрограммы управления климатом

Данный алгоритм подпрограммы управления климатом немного похож на алгоритм подпрограммы управления освещением.

Начало алгоритма начинается с расшифровки всех вышеперечисленных параметров. Программа начинает проводить опрос всех датчиков данного алгоритма, чтобы узнать в каком состоянии они находятся в данный момент.

В этой подпрограмме пользователь сам задает комфортную для него температуру в помещении, а также пользователь вправе управлять этой системой автоматически. Для изменения температурой будут задействованы такие команды как включить/выключить вентилятор, включить/выключить батареи, открыть/закрыть окна.

Пользователь задает в своем доме комфортную для него температуру, которая должна поддерживаться системой, а также пользователь создает заданный допуск «Δ». Это небольшой интервал изменения температуры, при котором датчики не будут срабатывать, если заданная температура станет чуть больше или меньше.

Если текущая температура в комнате с заданным допуском «Δ» соответствует заданной пользователем температуре, то в помещении, как и прежде без изменений будет поддерживаться такая же температура, до тех пор, пока какие-нибудь факторы не повлияют на ее изменение (Условие 1).

В случае, если это условие не выполняется, то происходит опрос датчиков – больше ли температура в комнате Тк заданной температуры Тз с заданным допуском «Δ»? (Условие 2). Если это условие выполняется, значит, требуется уменьшить температуру в помещении. Дальше происходит опрос датчиков – больше ли температура на улице Тул температуры комнатной Тк? (Условие 3) Если это условие выполнимо, система принимает решение провести следующий опрос датчиков – включены ли батареи (State_B=1)? В случае, если батареи включены, то система дает команду отключить батареи (State_B=0) и закончить данный

небольшой цикл и перейти к следующему действию, а если батареи не были включены, то сразу же переходим к следующему действию. После того как последнее действие выполнилось происходит опрос датчиков – открыты ли окна (State_W=1)? Если окна открыты, то система должна принять решение закрыть их (State_W=0) и перейти к следующему действию, а если окна не открыты – то сразу переходим к следующему действию. По алгоритму требуется снова провести опрос датчиков – включен ли кондиционер (State_C=1)? Если не включен, то происходит включение кондиционера (State_C=1), а если включен, то происходит сразу конец цикла этой ветви.

Возвращаемся к условию 3, где Тул больше Тк и это условие не выполняется. В этом случае происходит опрос датчиков – включены ли батареи (State_B=1)? Если батареи выключены, то никаких действий не происходит, а если они включены, то их нужно отключить (State_B=0) и после этого открыть окна (State_W=1). После происходит конец цикла.

Возвращаемся к условию 2. В опросе датчиков может быть такой исход, что температура в комнате Тк меньше заданной температуры Тз с заданным допуском «Δ» и в этом случает требуется увеличить температуру. Снова проходит опрос датчиков больше ли температура уличная Тул температуры комнатной Тк? (Условие 4) Если все выполнимо, то система выключает кондиционер (State_C=0) и открывает окна (State_W=1). А если то условие невыполнимо, то система сама выключает кондиционер (State_C=0) и проводит опрос датчиков – открыты ли окна? Если нет, то сразу происходит конец этого маленького цикла, а если окна открыты, то окна закрываются (State_W=0). После происходит включение батарей (State_B=1). Конец цикла этой ветви. На рисунке 21 показан алгоритм подпрограммы управления климатом.

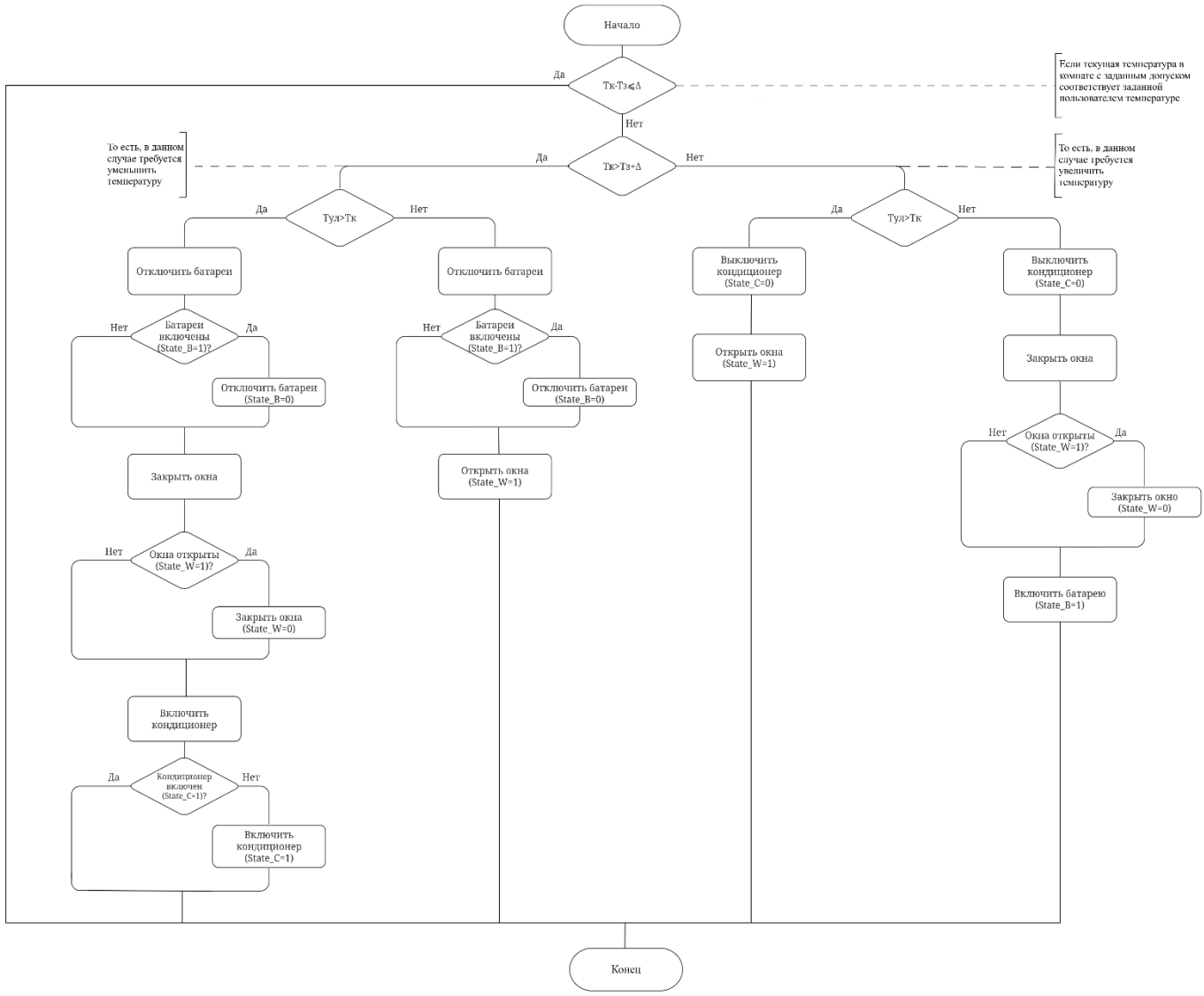

Рисунок 21 – ПП управления климатом

После составления алгоритмов общей программы и алгоритмов подпрограмм требуется сделать привязку к мобильному приложению.

4 Мобильное приложение

В качестве приложения для управления проектом, мы возьмем готовое решение – мобильное приложение SHome.

SHome – это приложение, с помощью которого можно управлять проектом на Arduino с использованием смартфона.

В настоящее время, трудно представить свою жизнь без смартфона, ведь с помощью него мы можем выполнять множество функций. Поэтому я склоняюсь к такому мнению, что очень удобно использовать смартфон в качестве пульта дистанционного управления для проектов домашней автоматизации.

Так как смартфон всегда под рукой, мы можем комфортно управлять температурой, светом, бытовыми приборами, и кормушкой. Идея этого проекта заключается в том, чтобы управлять с помощью смартфона электрическими устройствами. Для отправки команд в Arduino мы будем использовать модуль Bluetooth HC-06. Принцип действия довольно прост, смартфон отправляет команды через Bluetooth, а Arduino принимает их, и в соответствии с полученной командой активирует или деактивирует реле.

4.1 Шаг 1: "Привязка" bluetooth модуля

Первое, что нам нужно сделать, это “привязать” Bluetooth модуль, который будет использоваться в проекте. Это делается только один раз, и модуль будет сохранен в памяти смартфона, поэтому нам не придется каждый раз “привязывать” Bluetooth модуль. Чтобы “связать” Bluetooth модуль со смартфоном, мы должны подключить питание к Bluetooth модулю. Можно просто подключить GND bluetooth модуля к GND arduino, а вывод +5 v bluetooth модуля к выводу 5v arduino. После требуется подключить Arduino к компьютеру. Если все в порядке, индикатор модуля Bluetooth должен быстро мигать.

Чтобы связать модуль со смартфоном, нужно зайти в настройки смартфона в разделе «Подключения / Bluetooth» и активируйте Bluetooth (если он деактивирован). Когда он активирован, мы нажимаем кнопку «Поиск». Когда смартфон найдет модуль, нужно нажать имя модуля Bluetooth и начинается процесс соединения, через несколько секунд появится сообщение, что модуль Bluetooth уже подключен. Теперь Bluetooth модуль находится в памяти смартфона.

4.2 Шаг 2: Выбрать Bluetooth модуль в приложении

Теперь нужно выбрать в приложении модуль, который мы собираемся использовать. Поскольку Bluetooth модуль уже подключен, то он будет в списке подключенных устройств Bluetooth. Сначала требуется активировать Bluetooth, чтобы иметь возможность выбрать Bluetooth модуль из списка, затем на главном экране приложения выбираем «Настройки», и в меню, которое появится на экране выберем «Выбрать Bluetooth модуль». Мы увидим список всех устройств, связанных со смартфоном. Нам нужно выбрать Bluetooth модуль, который мы собираемся использовать. После выбора он будет сохранен в памяти и нам не нужно выбирать его снова, только если мы хотим использовать другой Bluetooth модуль.

4.3 Шаг 3: Создайте список устройств для управления

Теперь можно создать список электрических устройств, которые мы собираемся включить / выключить. На главном экране мы нажимаем желтую кнопку, и открывается другой экран, где мы можем создать список устройств. На открывшемся экране нажмите кнопку «Новый элемент» (это кнопка с тремя линиями и символом плюса «+»).

Нажатие на кнопку «Новый элемент» открывает два поля для заполнения. Первое поле – это имя, с помощью которого мы будем распознавать устройство, которое мы собираемся включить / выключить. Второе поле – это значение, которое будет отправлено в arduino, когда мы нажмем кнопку «включить / выклю-

чить», Каждая новая кнопка должна иметь уникальное значение Это означает, например, что если мы создадим кнопку со значением «A» мы не сможем присвоить то же значение для других кнопок которые мы собираемся создать. Далее мы увидим список значений в тексте программ Arduino для каждого из пинов (их можно изменить на свое усмотрение).

Теперь, после создания устройств, у нас есть список устройств для управления, если мы нажимаем на устройство в списке появляются две кнопки, одна – кнопка для возврата к списку устройств, а другая – кнопка которая отправляет значение в arduino (значение, которое мы написали, когда создали устройство).

Например, если мы создаем устройство «Кормушка 1» со значением «A», нажимая имя «Кормушка 1» в списке, появятся две кнопки, одна большая и одна маленькая, при нажатии большой кнопки будет отправлено значение «A», которое мы присвоили элементу «Кормушка 1». Arduino получит значение «A», и таким образом мы можем написать код в arduino для выполнения некоторой функции, когда он получит это значение «A». Вот почему каждое устройство, которое будет создано должно иметь свою собственную букву или символ, чтобы Arduino мог отличить одно устройство от другого.

"A"---------- включить / выключить пин 2 на Arduino

"B"---------- включить / выключить пин 3 на Arduino

"C"---------- включить / выключить пин 4 на Arduino

"D"---------- включить / выключить пин 5 на Arduino

"E"---------- включить/ выключить пин 6 на Arduino

"F"---------- включить / выключить пин 7 на Arduino

"G"---------- включить / выключить пин 8 на Arduino

"H"---------- включить / выключить пин 9 на Arduino

"I"---------- включить / выключить пин 10 на Arduino

5 Расположение датчиков

Устройства, которые понадобятся для комплекса обеспечения безопасности – считыватель RFID меток, сканер отпечатков пальцев, инфракрасный датчик, датчик открывания двери.

Данные элементы, такие как считыватель RFID меток и сканер отпечатка пальца будут установлены у входной двери в стене, так как пользователю будет удобно аутентифицироваться и осуществить вход в помещение.

Инфракрасный датчик будет установлен около входной двери.

Датчик открытия двери представляет с собой магнит с герконом. Эта установка будет внутри двери.

Устройства, которые понадобятся для комплекса управления освещением:

Внутренние датчики света, внешний датчик света, шаговые двигатели для открытия/закрытия штор. В доме установлено 5 окон, 7 световых приборов.

Внутренние датчики света требуется установить в каждой комнате, а комнат в нашем проекте 4. Также следует установить фоторезистор с датчиком интенсивности света. В студии будет 2 датчика интенсивности света и 2 фоторезистора. Устанавливаться эти устройства будут в стене. Несмотря на то что это датчики света-эти датчики нельзя ставить в места, направленные на окна, кондиционеры или монтировать их на улице в сильно продуваемых местах. Устанавливать надо на высоте от 2.3 до 2.7 м. Датчик света должен быть установлен на достаточно расстоянии от окна и от приборов освещения. Термисторы, которые представляют собой датчики движения света будут устанавливаться тоже в каждой комнате. В студии будет 2 термистора.

Внешний датчик света также представляет собой фоторезистор с датчиком интенсивности света. Датчик освещенности должен устанавливаться на открытой местности. Где он не может быть затенен деревом, строением или другими объектами. Иначе это может привести к его ложной работе. Устанавливается в стене недалеко от входной двери.

Шаговый двигатель для открывания окон устанавливается в «рулонах» жалюзи, для удобного управления шторами. В проекте 5 окон и на всех этих окнах есть шторы, следовательно, нужно 5 шаговых двигателей.

Устройства, которые понадобятся для комплекса управления климатом: внутренние датчики температуры, внешний датчик температуры, блок сервоприводов для управления окнами.

В доме 5 окон, 4 радиатора, 3 кондиционера.

Внутренние датчики температуры будут установлены в каждой комнате. В большинстве случаев советуют устанавливать датчик температуры на высоте 1,5 м от пола, подальше от окон, дверей, радиаторов отопления и от источников освещения, для того, чтобы показания были правильными.

Внешние датчик температуры следует установить на улице недалеко от входной двери.

Система кондиционирования состоит из 3 кондиционеров, и также, как и система обогревателей будет состоять из 3 радиаторов.

Блок сервоприводов для управления открытия/закрытия окон будет устанавливаться в оконной раме, для удобного управления окнами.

Устройства, которые нужны для комплекса управления бытовыми приборами.

Производится установка «умной розетки» на кухне. Основным элементом в этом устройстве будет являться реле.

Кормушка. Установлена на кухне. Основным элементом в «Умной кормушке» будет являться сервопривод и инфракрасный датчик. Кормушка находится в студии.

Обозначение датчиков на планировке:

1) светло-зеленый цвет датчика – фоторезистор;

2) темно-зеленый цвет датчика – датчик интенсивности света;

3) фиолетовый цвет датчика – термистор;

4) желтый цвет датчика – шаговый двигатель;

5) красный цвет датчика – температурные датчики;

6) оранжевый цвет датчика – розетка;

7) синий цвет датчика – сервоприводы;

8) розовый цвет датчика – инфракрасный датчик;

9) бордовый цвет датчика – магнит с герконом;

10) коричневый цвет датчика – RFID сканер и RDM модуль;

На рисунке 22 изображена планировка 1-го и 2-го этажей с установленными датчиками.

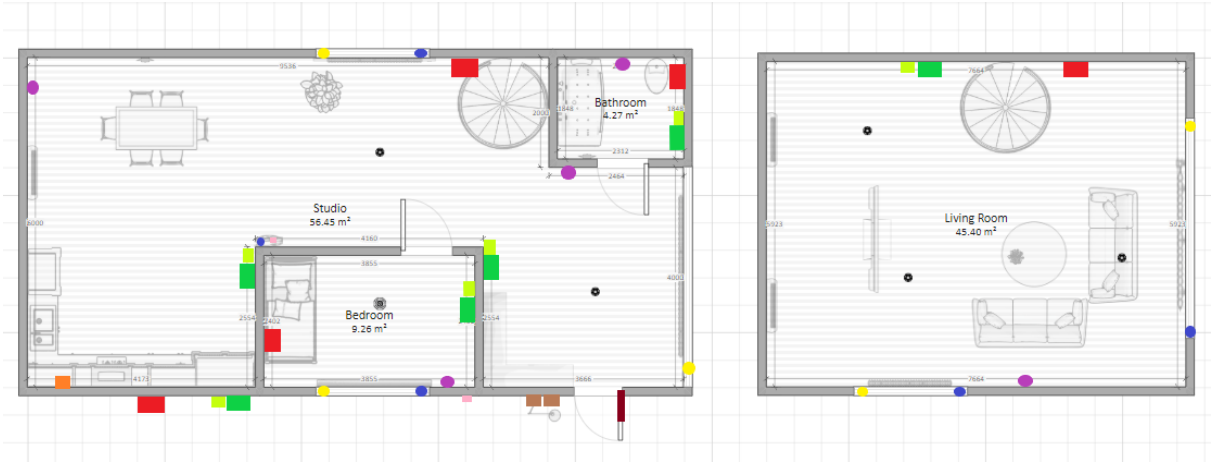

Рисунок 22– Планировка 1-го и 2-го этажей с установленными датчиками

Вывод по разделу 5

ЗАКЛЮЧЕНИЕ

В ходе выполнения дипломной работы, получены следующие результаты.

1. Анализ современных систем автоматизации зданий, результатыкоторых позволили обосновать выбор инструментальных средствразработки программного обеспечения.

2. Разработана архитектура, включающая программные и аппаратныекомпоненты, где главной отличительной особенностью является гибкость иудобство в использовании.

3. Разработан программно-аппаратный комплекс умного дома, которыйкоторый реализует предложенную архитектуру.

4. Разработаны алгоритмы функционирования системы управления исистемы контроля, позволяющие удаленно в реальном времениконтролировать и управлять программно-аппаратным комплексом.

5. Результаты тестирования показали, что разработанная системаудовлетворяет требованиям технического задания.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1) «Умный» дом XXI века. А.И. Дементьев – 100 с.

2) «Умный дом» своими руками. Строим интеллектуальную цифровую систему в своей квартире. Е.В. Тесля – 360 с.

3) Концепция и положения умного дома https://studbooks.net/2347880/tehnika/suschestvuyuschee_polozhenie_umnogo_doma

4) Плата Arduino https://ru.wikipedia.org/wiki/Arduino#Arduino/

5) Плата Arduino Mega 2560 http://arduino.ru/Hardware/ArduinoBoardMega2560/

6) Работа на Arduino сканера отпечатков пальцев https://infostart.ru/public/889318/

7) Подключение RFID модуля RDM630 к Arduino Mega 2560 https://geekelectronics.org/arduino/podklyuchenie-rfid-modulya-rdm630-k-arduino-mega-2560.html/

8) Подключение RFID модуля RDM630 к Arduino Mega 2560 https://geekelectronics.org/arduino/podklyuchenie-rfid-modulya-rdm630-k-arduino-mega-2560.html/

9) Инфракрасный датчик https://arduino.ua/prod193-ik-datchik-dvijeniya-dlya-arduino-hc-sr501

10) Герконовый датчик https://domelectrik.ru/oborudovanie/datchik/gerkony

11) Герконовый датчик — принцип работы, характеристики и применение http://oknoportal.com/datchiki-otkrytiya-dveri/

12) Подключение фоторезистора к Arduino и работа с датчиком освещенности https://arduinomaster.ru/datchiki-arduino/photorezistor-arduino-datchik-sveta/

13) Датчик интенсивности света https://freedelivery.com.ua/arduino-100/datchiki-130/datchik-intensivnosti-sveta-yarkosti-modul-arduino-1484.html

14) Шаговый двигатель http://robotosha.ru/arduino/stepper-motor-28byj-uln2003-arduino.html

15) Термисторы http://arduino-diy.com/arduino-thermistor

16) Описание температурного датчика DS18B20 https://silines.ru/datchik-ds18b20-opisanie/

17) Сервопривод http://moshobby.ru/magazin-2/product/towerpro-mg995-metal-gear-servo-1

18) Установка сервопривода на окна https://gadgetpage.ru/instrukcii/3266-otkrytie-i-zakrytie-okna-s-pomoschju-arduino.html/

19) Реле https://itelectronics.biz/p603202987-rele-srd-05vdc.html

20) Розетка https://habr.com/ru/post/395461/

21) Дисплей https://wiki.iarduino.ru/page/Nextion/

22) Bluetooth модуль https://duino.ru/bluetooth-modul-hc-06.html

23) Приложение https://neco-desarrollo.es/shome-2

24) СТО ЮУрГУ 04–2008 Стандарт организации. Курсовое и дипломное проектирование. Общие требования к содержанию и оформлению/ состави-тели: Т.И. Парубочая, Н.В. Сырейщикова, В.И. Гузеев, Л.В. Винокурова. – Челя-бинск: Изд-во ЮУрГУ, 2008. – 56 с.

№	Формат	Обозначение	Наименование	Колич.
			Текстовые документы.	
1	А4	ЮУрГУ-Д.11.03.02.18.ХХХ.00 ПЗ	Пояснительная записка	80
			Графические материалы.	
2	А1	ЮУрГУ-Д.11.03.02.20.219.00 Д2	Структурная схема	1
3	А1	ЮУрГУ-Д.11.03.02.20.219.01 Д2	Алгоритмы	1

Изм.	Лист	№ докум.	Подпись	Дата	ЮУрГУ-Д.11.03.02.2020.219.00 Д4		
Разраб.	Комлева Л.Т.				«Умный дом» на основе программно-аппаратной платформы Arduino IDE	Лит.	Лист	Листов
Провер.	Пискорский Д.С.					Д	78	80
Н. Контр.	Спицына В.Д.					ЮУрГУ		
Утв.	Даровских С.Н.							