Введение.

В наше время человек пытается сделать свою жизнь проще и комфортнее, создавая новые изобретения. Люди хотят улучшить свою среду обитания. А что если задаться вопросом: что нужно человеку для комфортного существования? Как сделать жизнь людей комфортнее и удобнее? Конечно, для этого нам понадобится поручить контроль среды обитания человека системе, которую называют «Умный дом». Такая система способна контролировать почти всё в Вашем доме, что является очень удобным. И тут мы задумались: «А почему бы не создать свою систему умного дома», так и родилась идея нашего проекта.

**Цель**: создание системы «Умный дом», которая сделает жизнь комфортнее.

**Задачи**:

1. Выбрать контроллер на базе которого будет работать наша система.
2. Разработать проект системы, предусмотрев его применение в реальной жизни.
3. Выяснить, что необходимо для создания системы.
4. Научиться программировать контроллер и управлять передачей данных с него.
5. Сделать управление системой разносторонним и удобным.

Что же нужно для комфортного существования человека?

Мы выяснили, что пользователям будет очень удобно, если система «Умный дом» будет выполнять следующие функции:

* Климат-контроль – поддержание постоянной температуры в доме.
* Авто полив растений – не даст комнатным цветам засохнуть.
* Автоматическое управление светом – вам больше не придётся пользоваться выключателями
* Функции защиты дома:
  + Сигнализация от воров.
  + Противопожарная сигнализация (контроль утечки газа).

Управление Умным домом будет осуществляться с помощью ИК пульта (как от телевизора) и с мобильного телефона на платформе Android. Все данные система будет передавать на компьютер (или сервер).

**§1. Выбор аппаратной составляющей**

Прежде чем начать работу над проектом нужно подобрать комплектующие для его реализации, начнём с того, какую плату (контроллер) мы выбрали.

Существует множество контроллеров и программируемых плат для решения нашей задачи, однако мы решили выбрать контроллер Arduino Mega 2560.

Почему именно Arduino? Мы выбрали именно плату Arduino Mega, потому что она имеет прекрасные характеристики и идеально подходит для нашего проекта "Умный дом". У платы есть огромное количество аналоговых и цифровых контактов, а так же четыре порта последовательной передачи данных, что позволяет создавать масштабные проекты с большим количеством датчиков и разными каналами передачи данных.

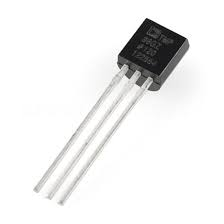
*Основные характеристики платы*

|  |  |
| --- | --- |
| Микроконтроллер | ATmega2560 |
| Рабочее напряжение | 5В |
| Входное напряжение (рекомендуемое) | 7-12В |
| Входное напряжение (предельное) | 6-20В |
| Цифровые Входы/Выходы | 54 (14 из которых работают в режиме ШИМ) |
| Аналоговые входы | 16 |
| Постоянный ток через вход/выход | 40 mA |
| Постоянный ток для вывода 3.3 В | 50 mA |
| Флеш-память | 256 KB (8 КB используются для загрузчика) |
| ОЗУ | 8 KB |
| Энергонезависимая память | 4 KB |
| Тактовая частота | 16 MHz |

Также плата легка в программировании, скетчи (программы) пишутся на основе C или C++. Микроконтроллер ATmega2560 имеет: 256 Кб флеш-памяти для хранения кода программы.

Для реализации системы Умного дома, нам понадобятся следующие комплектующие.

Для определения температуры в доме мы будем использовать

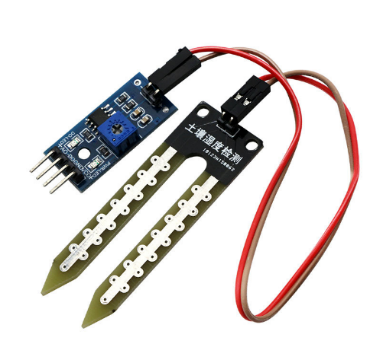
* Датчик температуры LM35

Характеристики

* *Напряжение питания*: 2.7-5.5В
* *Погрешность измерений*: 2 градуса
* *Измеряемая температура*: от 10°C до 125°C
* *Потребляемый ток*: 50мкА

Для определения влажности почвы растений применим

* Датчик влажности FC-28-B (Гигрометр)

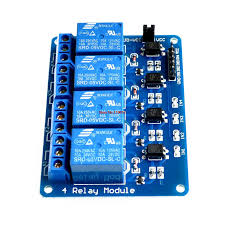
Характеристики

* *Выходной сигнал*: 0 - 4,2В
* *Ток*: 35мА
* *Размеры*: 90х58х8 мм
* 4-х проводной интерфейс:

1. VCC: 3.3-5В
2. GND: "Земля" ("-")
3. DO: логический (цифровой) выход с компаратора ("0" при влажной почве и "1" при сухой)
4. AO: аналоговый выход, показывающий степень влажности, считывается при помощи микроконтроллера

Чтобы включать и выключать сторонние приборы пригодится

* Релейный модуль

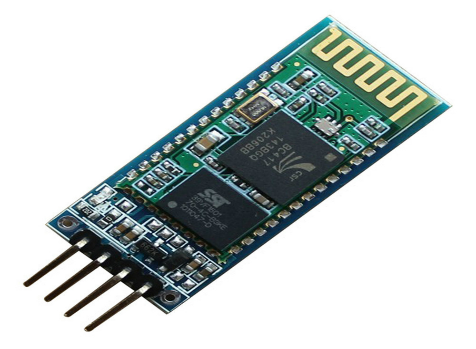


Характеристики

* *Максимальный ток*: 10А
* *Рассчитано на постоянное напряжения тока до* 30В *и перемене до* 220В
* *С помощью этих реле можно управлять различными бытовыми приборами рассчитанными на напряжение* 220В

Для передачи данных между системой «Умный дом» и мобильными устройствами необходим

* Bluetooth модуль

Характеристики

* *Напряжение питани*:3,3–6 В
* *Максимальное входное напряжение логической единицы*:5 В
* *Выходное напряжение логической единицы*:3,3 В
* *Максимальный ток потребления*:45 мА
* *Скорость передачи данных*:1200–1382400 бод
* *Дальность связи при прямой видимости*:30 м (Без учета преград)

При создании противопожарной сигнализации никак не обойтись без этого датчика

* Датчик газа MQ2

Характеристики

* Напряжение питания: 5 В
* Потребляемый ток: 160 мА

Диапазон измерений

* Пропан: 0,2 – 5 промилле
* Бутан: 0,3 – 5 промилле
* Метан: 5 – 20 промилле
* Водород: 0,3 – 5 промилле
* Пары спиртов: 0,1 – 2 промилле

Для отображения температуры применим

* **4-х разрядный цифровой дисплей**



Характеристики

* Grove совместимый интерфейс
* 4 цифры красного свечения
* 8 регулируемых уровней яркости
* Рабочее напряжения: 3.3 ~ 5.5 В
* TM1637 драйвер
* Размер 42 \* 24 мм.

Охрану вашего дома и управление светом обеспечат

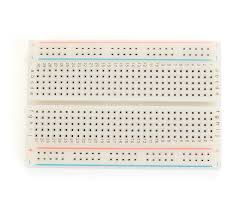
* Ультразвуковые датчики расстояния HC-SR04

Характеристики

* Напряжение питания: 5V DC
* Ток покоя: <2mA
* Эффективный угол: <15°
* Диапазон измерения расстояния: 2–400 см
* Разрешение: 0.3 см

Чтобы скомпоновать все устройства в одно целое понадобится

* Breadboard RKP-BB-400



Характеристика

* Рабочая температура: от -20°C до + 80°C
* Количество соединений для контактов: ≥ 10000
* Диапазон захвата контактов: от 0,4 мм до 0,7 мм
* Сопротивление контакта: 100mΩ (Max)
* Сопротивление изолятора: 1000mΩ (Min.)

Подобрав нужные комплектующие, опишем, как будет функционировать каждый модуль Умного дома и вся система в целом, а также как будет происходить обмен данными между платой и внешними устройствами.

**§2. Описание работы систем**

**Климат-контроль**

Функция поддержания постоянной температуры будет работать следующим образом:

* собираются данные с датчика температуры LM35, подключённому к аналоговому входу А0;
* значения выводятся на дисплей TM1637(led модуль), подключённому к контактам 22 и 24;
* если полученные значения больше нужной температуры (как она устанавливается пояснено далее, изначально нужная температура равна 20°С), то срабатывает реле (подключенное к цифровому контакту 52), на которое замкнут кондиционер или иной охлаждающий прибор;
* если значения датчика меньше нужной температуры, то реле размыкается и выключается кондиционер;
* требуемая температура регулируется с помощью кнопки (на контакте 53) и потенциометра (переменного резистора на контакте А0): при нажатии кнопки значение нужной температуры принимает значение показания потенциометра; требуемая температура отображается на дисплее;
* работа всего модуля управляется через ИК пульт или/и телефон: в Android приложении, для включения функции достаточно нажать на переключатель «Климат-контроль», на пульте ­­­– кнопку 1;
* также существует возможность включения кондиционера вручную[[1]](#footnote-1), для это в приложении вы должны нажать на кнопку «Кондиционер», а на пульте – на кнопку «Menu».

Схема модуля «Климат-контроль» см. Приложение 1

**Автополив растений**

Функция автоматического полива растений работает следующим образом:

* собираются данные с датчика влажность почвы FC-28, подключённого к цифровому контакту 51;
* если датчик срабатывает (земля сухая), то включаем реле насоса полива (контакт 45) на 10 секунд, после чего снова проверяем значения датчика. Если датчик выдаёт значение 0 (земля достаточно влажная), то выключаем реле насоса;
* работа всего модуля управляется с помощью Android приложения и ИК пульта, для включения функции автополива в приложении достаточно нажать переключатель с названием «Автополив», а на пульте – нажать на кнопку 2;
* также существует возможность включения насоса вручную[[2]](#footnote-2), для этого в приложении вы должны нажать на кнопку «Насос полива», а на пульте – на кнопку «Info». В случае ручного включения насос будет работать 10 секунд, после чего выключится.

Схема модуля «Автополив» см. Приложение 2

**Автоматическое управление светом**

Опишем принцип работы автоматического управления светом:

* Два ультразвуковых дальномера (подключённые к контактам 8,9 и 6,7) устанавливаются у входа в комнату (см. Приложение 5, схема установки датчиков), будем условно считать, датчик находящийся ближе к входу первым, а оставшийся датчик вторым.
* В исходном коде задаётся расстояние проёма для входа в комнату (в котором стоят датчики), датчики двигать нельзя.
* Будем считать количество людей в комнате следующим образом: если показания первого датчика меньше, чем расстояние до стены, то значит человек начал входить, как только показания второго датчика тоже уменьшатся, людей в комнате стало на одного больше; если же происходит обратная ситуация: сначала уменьшаются значения на втором датчике, а затем на первом, то это обозначает что человек выходит, в комнате стало на одного меньше.
* Таким образом, если в комнате есть хотя бы один человек включаем свет, иначе если в комнате никого нет - выключаем. Управление светом осуществляется через модуль реле (контакт 43).
* Работа модуля управляется с помощью ИК пульта или/и телефона: в Android приложении, для включения функции достаточно нажать на переключатель «Управление светом», на пульте ­­­– кнопку 3.
* Также предусмотрена возможность управления светом вручную[[3]](#footnote-3), для этого в приложении для телефона необходимо нажать на кнопку «Свет», а на пульте – на кнопку «EPG»

Схема модуля автоматического управления светом см. Приложение 6

**Противопожарная сигнализация[[4]](#footnote-4)**

Принцип работы функции противопожарной сигнализации.

* Собираются данные с датчиков газа и температуры (датчик газа и дыма MQ2 подключен к контакту А7).
* Если данные хотя бы одного датчика превышают допустимое значение (450 для MQ2 и 50°С для датчика температуры), то немедленно передаётся сигнал на компьютер: символ «F». После отправки сигнала модуль выключается.
* Тем временем программа, которая постоянно включена на ПК осуществляет мониторинг порта и как только получит сигнал отправляет SMS с помощью сервиса sms\_api на мобильный телефон пользователя.
* Так же существует возможность установки GPRS модуля для аппаратной отправки SMS по мобильной связи без подключения к ПК, однако наш способ используется, как более эффективный и экономный.
* Работа всего модуля управляется с помощью Android приложения и ИК пульта, для включения противопожарной сигнализации в приложении достаточно нажать переключатель с названием «Противопожарная сигнализация», а на пульте – нажать на кнопку 5.

Схема модуля «Противопожарная сигнализация»см. Приложение 3

**Сигнализация от воров[[5]](#footnote-5)**

Принцип работы функции сигнализации от воров.

* Сигнализация работает на основе тех же датчиков, что и автоматическое управление светом.
* Если хотя бы один из датчиков срабатывает, как это описано ранее, то устанавливается задержка на 30 секунд, на случай если в поле зрения датчика попал сам пользователь. За это время он должен успеть подойти к системе и нажать на выключатель сигнализации, или выключить функцию через телефон (см. далее).
* Если ничего не было сделано и 30 секунд прошли, то посылается сигнал на компьютер: символ «A». После отправки сигнала модуль отключается.
* Тем временем программа, которая постоянно включена на ПК осуществляет мониторинг порта и как только получит сигнал отправляет SMS с помощью сервиса sms\_api на мобильный телефон пользователя.
* Так же существует возможность установки GPRS модуля для аппаратной отправки SMS по мобильной связи без подключения к ПК, однако наш способ используется, как более эффективный и экономный.
* Работа всего модуля управляется с помощью Android приложения и ИК пульта, для включения сигнализации от воров в приложении достаточно нажать переключатель с названием «Сигнализация от воров» (при отключении сигнализации будет отключена тревога, если таковая была), а на пульте – нажать на кнопку 4.

Схема модуля «Сигнализация от воров» см. Приложение 4

**Система связи**

Как вы уже поняли, наша система использует три вида связи: связь с компьютером, связь с мобильными устройствами через Bluetooth и связь с инфракрасным пультом через ИК приёмник. Рассмотрим принцип работы их всех.

**Связь с ПК**

Для связи с персональным компьютером или сервером используется стандартное подключение через стандартный USB порт. Подключение необходимо нам только для работы функций сигнализаций. При этом на ПК (сервере) должно быть установлено приложение, написанное нами на языке программирования Java специально для отправки SMS сообщений. Данное приложение постоянно осуществляет мониторинг порта, к которому подключена наша система и при виде сигнала посылает SMS. Программа также включает в себя библиотеку sms\_api для отправки сообщений, код программы смотрите в приложениях.

**Связь с ИК пультом**

Для передачи данных с пульта на нужно подключить к Arduino ИК приёмник, он позволит нам принимать сигнал с Ик передатчика, установленного на пульте. Стоит отметить, что при получении данных с пульта, эти же данные отправляются на Android (зачем это нужно, описано далее).

**Связь с мобильными устройствами**

Для управления системой с телефона или другого мобильного устройства нам понадобится подключить Bluetooth модуль к Arduino, таким образом мы сможем передавать данные с Android на Arduino. Для удобства работы пользователя мы создали собственное Android приложение на языке Java, с удобным интерфейсом. Интерфейс программы показан на фотографии.

Приложение может как передавать, так и принимать данные. Таким образом, осуществляется связь между передачей данных через пульт и через телефон: приложение «знает» включен ли модуль или нет (хотя его могли включить не только через Android, но и через пульт). Нет никакой разницы *как* включит пользователь одну из функций, типы передачи данных связаны между собой. Полный код программы для мобильных устройств смотрите в описании.

**Заключение**

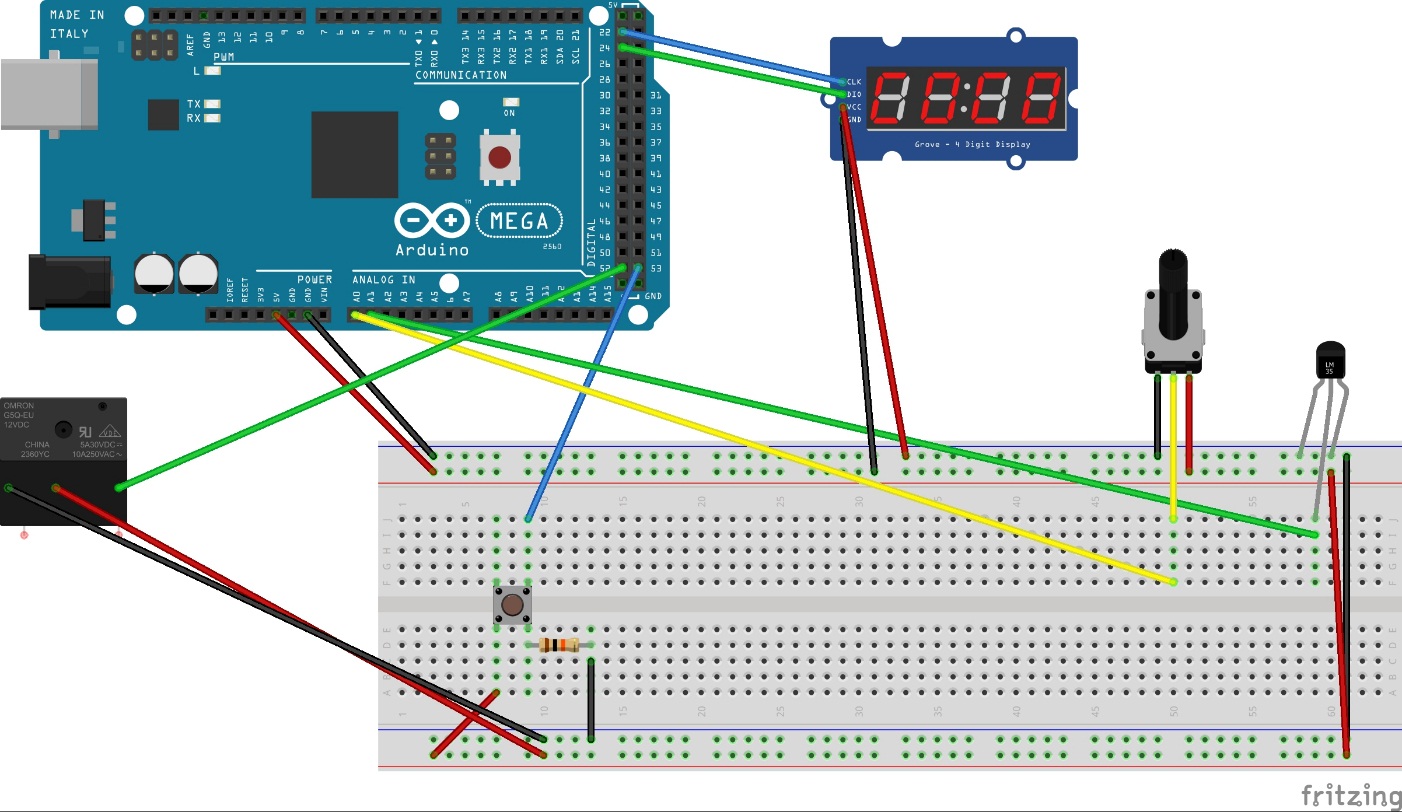
Исходя из проделанной работы можно сделать вывод, что цель и задачи были выполнены. Программа имеет практическое применение. Например,

представительский особняк успешного бизнесмена демонстрирует респектабельность и уверенность в себе хозяина дома, за счет быстрой и слаженной работы всех систем.

**Приложения**

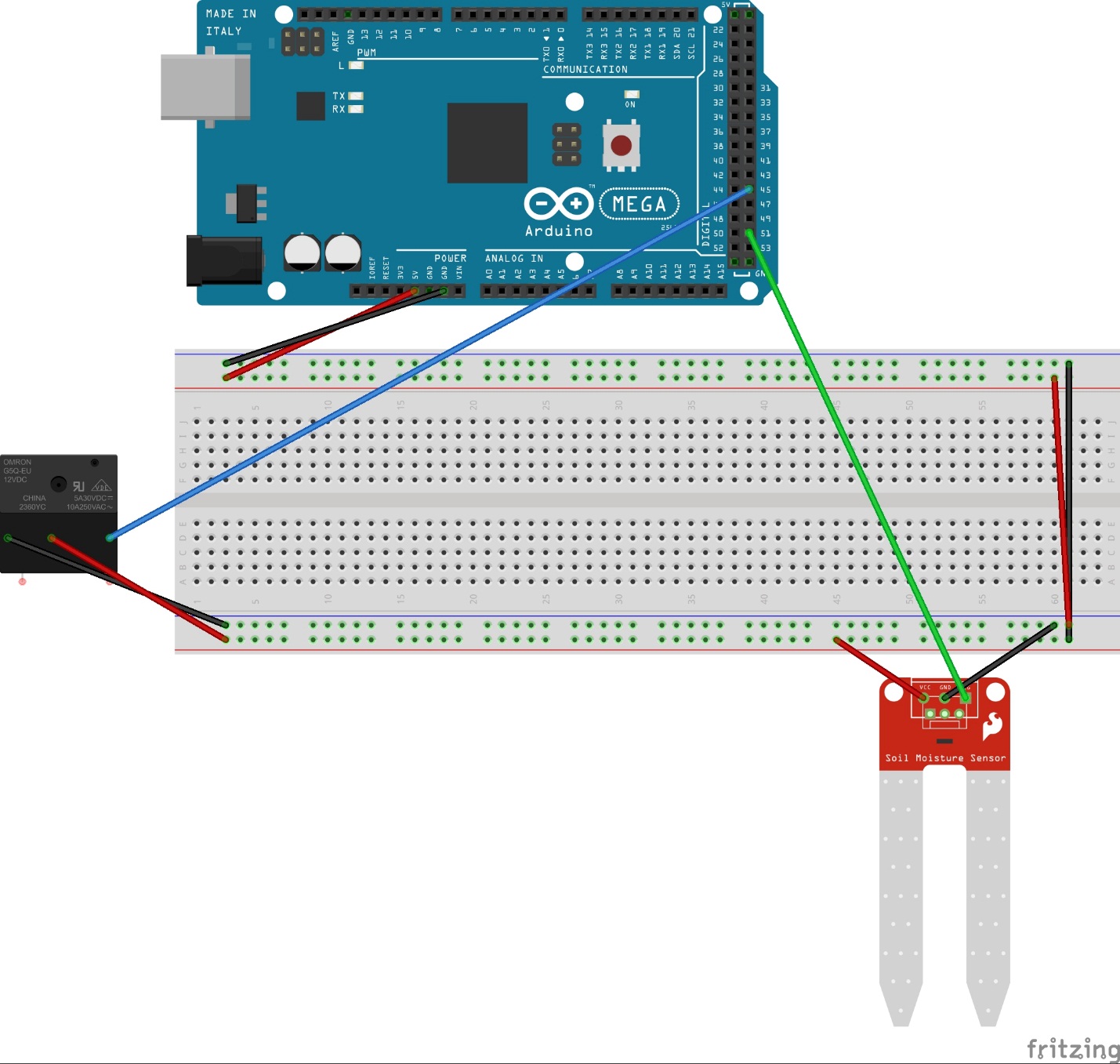
Приложение 1

**Схема модуля «Климат-контроль»**



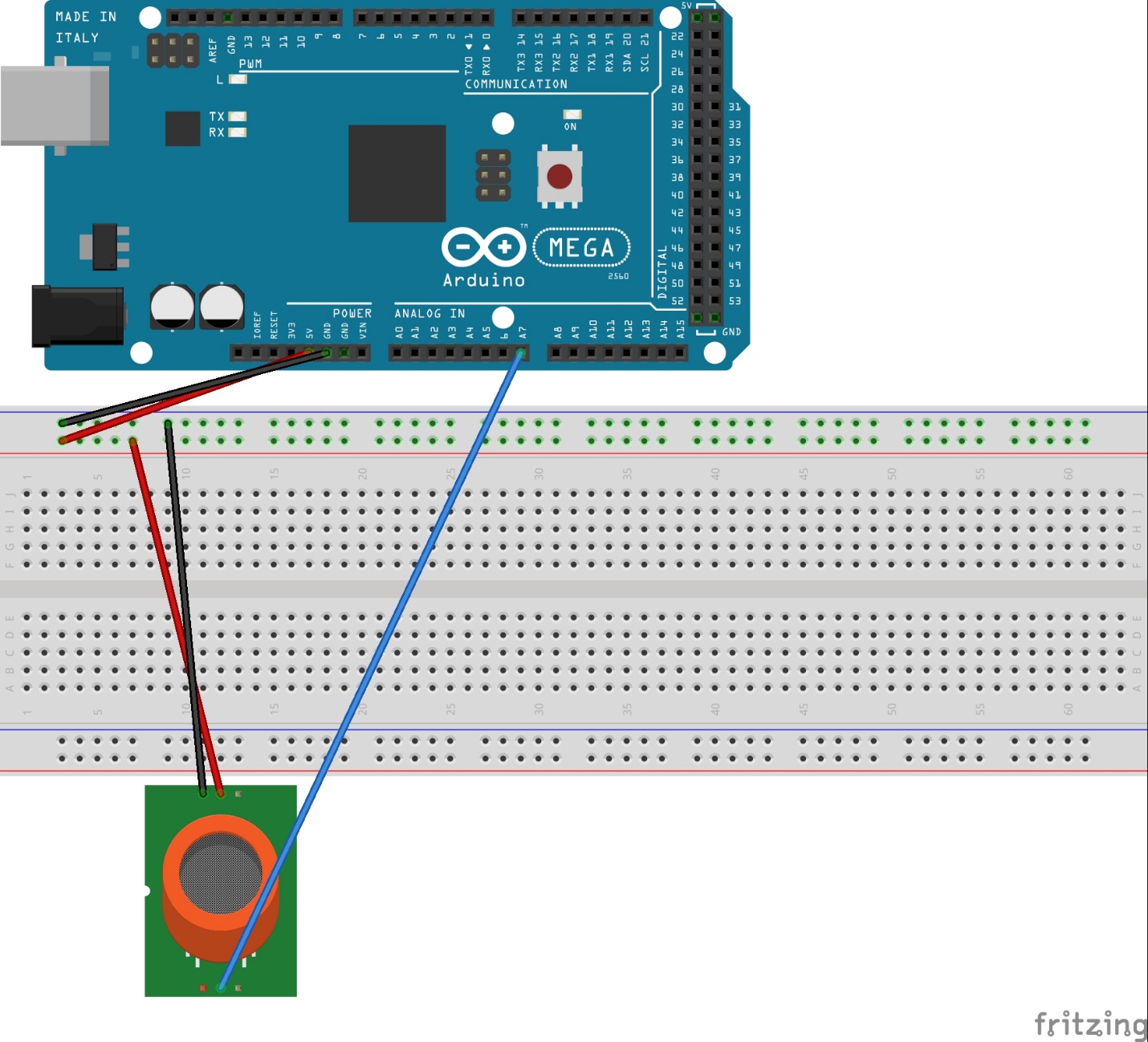
Приложение 2

**Схема модуля «Автополив»**



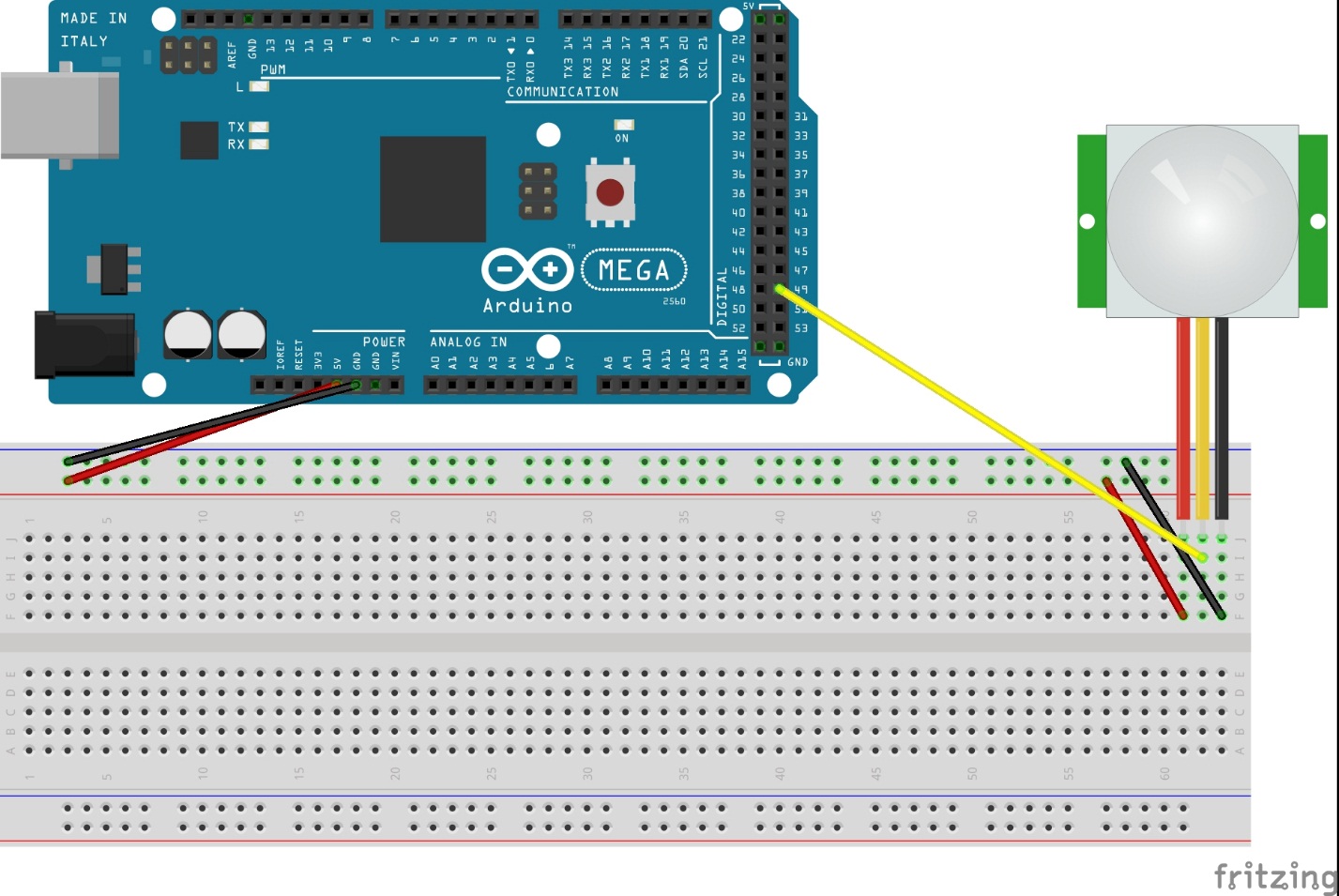
Приложение 3

**Схема модуля «Противопожарная сигнализация»**



Приложение 4

**Схема модуля «Сигнализация от воров»**



Полный код скетча Smart\_home для Arduino

#include <TM1637.h>

#include <IRremote.h>

const int but=53; //кнопка

const int rele\_tempr =52;//реле кондиционера

const int tempr = A1; //датчик температуры

const int pm =A0; //потенциометр

const int RECV\_PIN = 2; //ИК приёмник

const int dd =49; //датчик движения

const int dd\_offer = 48; //выключатель тревоги

const int dd\_led =47; //светодиод сигнализации от воров

const int tempr\_led =50;//светодиод модуля климат контроля

const int MQ2= A7 ;//датчик дыма

const int MQ\_led= 46;//светодиод противопожарной сигнализации

const int grow\_sensor = 51;//датчик влажности почвы

const int grow\_rele = 45;//реле насоса

const int grow\_led = 44;//светодиод системы автополива

TM1637 tm1637(22,24); //подключаем LED модуль(дисплей)

IRrecv irrecv(RECV\_PIN);

decode\_results results;

float real\_tempr, need\_tempr=22;

boolean is\_tempr=false,is\_dd=false,is\_MQ=false, off=false, is\_cond=false, is\_grow=false, is\_nasos=false, is\_grow\_work=false, is\_tempr\_work=false;//переменны вкл/выкл модулей

void setup() {

tm1637.init();// инициализация библиотеки «TM1637.h»

tm1637.set(5);//установка яркости

analogReference(INTERNAL1V1);//понижаем напряжение на аналоговых входах до 1.1V

//определяем типы контактов

pinMode(but,INPUT);

pinMode(dd\_led,OUTPUT);

irrecv.enableIRIn(); // Start the receiver

pinMode(dd,INPUT);

pinMode(grow\_sensor,INPUT);

pinMode(rele\_tempr,OUTPUT);

pinMode(grow\_rele,OUTPUT);

digitalWrite(rele\_tempr,HIGH);

digitalWrite(grow\_rele,HIGH);

pinMode(dd\_offer,INPUT);

pinMode(MQ2, INPUT);

pinMode(MQ\_led,OUTPUT);

pinMode(tempr\_led,OUTPUT);

Serial1.begin(9600);

Serial.begin(9600);

}

void loop() {

//основной цикл

if (Serial1.available() > 0) { //приём данных с телефона

//Считываем пришедший байт

byte incomingByte = Serial1.read();

Serial.print(incomingByte);

if (incomingByte==48){is\_tempr=false;}//0

else if (incomingByte==49){is\_tempr=true;}//1

else if (incomingByte==97){is\_dd=false;digitalWrite(dd\_led,LOW); } //a

else if (incomingByte==65){is\_dd=true;digitalWrite(dd\_led,HIGH);} //A

else if (incomingByte==102){is\_MQ=false;}

else if (incomingByte==70){is\_MQ=true;}

else if (incomingByte==53){is\_grow=false;}

else if (incomingByte==52){is\_grow=true;}

else if (incomingByte==50){is\_cond=true;digitalWrite(rele\_tempr,LOW);}

else if (incomingByte==51){is\_cond=false;digitalWrite(rele\_tempr,HIGH);}

else if (incomingByte==55){is\_nasos=false;}

else if (incomingByte==54){is\_nasos=true;}

}

if (irrecv.decode(&results)) { //приём данных с пульта

switch ( results.value ) {

case 0x237807F:

if(is\_tempr==false){is\_tempr=true;Serial1.print(1);}

else {is\_tempr=false;Serial1.print(0);}

break;

case 0x23720DF:

if(is\_dd==false){is\_dd=true;Serial1.print("A");}

else {is\_dd=false;Serial1.print("a");}

break;

case 0x23702FD:

if (is\_tempr==false)

{if(is\_cond==false)

{

digitalWrite(rele\_tempr,LOW);is\_cond=true;Serial1.print(2);

}

else if(is\_cond==true)

{

digitalWrite(rele\_tempr,HIGH);is\_cond=false;Serial1.print(3);

}

}

break;

case 0x237A05F:

if(is\_MQ==true){is\_MQ=false;Serial1.print("f");}else{is\_MQ=true;Serial1.print("F");}break;

case 0x23740BF:

if(is\_grow==true){is\_grow=false;Serial1.print(5);}else {is\_grow=true;is\_nasos=false;Serial1.print(4);}break;

case 0x23758A7:

if(is\_grow==false){is\_nasos=true;Serial1.write(6);} break;

}

irrecv.resume(); // Receive the next value

}

if (is\_tempr==true){ //климат контроль

digitalWrite(tempr\_led,HIGH);

while (digitalRead(but)==HIGH)

{

need\_tempr = analogRead(pm)/70 + 15;

tm1637.display(2,need\_tempr/10);

tm1637.display(3,(int)need\_tempr % 10);

}

Serial.print("need t:");

Serial.println(need\_tempr);

real\_tempr = analogRead(tempr)/ 12.1;

Serial.print("real t:");

Serial.println(real\_tempr);

tm1637.display(2,real\_tempr/10);

tm1637.display(3,(int)real\_tempr % 10);

if(real\_tempr > need\_tempr)

{

digitalWrite(rele\_tempr,LOW);

}

else if(real\_tempr <= need\_tempr){digitalWrite(rele\_tempr,HIGH);}

}else{digitalWrite(tempr\_led,LOW);

tm1637.clearDisplay();

if (is\_cond==false){digitalWrite(rele\_tempr,HIGH);}

}

if(is\_dd==true){ //Сигнализация от воров

digitalWrite(dd\_led,HIGH);

if (digitalRead(dd)==HIGH)

{delay(30000);

if (Serial1.available() > 0)

{

byte incomingByte = Serial1.read();

Serial.print(incomingByte);

if (incomingByte==51){is\_dd=false; digitalWrite(dd\_led,LOW); }

}

if(digitalRead(dd\_offer)==LOW && is\_dd==true){

Serial.print("A");

is\_dd=false;

}

else if(digitalRead(dd\_offer)==HIGH)

{

is\_dd=false;

}

}

}else{digitalWrite(dd\_led,LOW);}

if(is\_MQ==true)//Противопожарная сигнализация

{ digitalWrite(MQ\_led,HIGH);

Serial.print("smoke: ");

Serial.println(analogRead(MQ2));

if(analogRead(MQ2)>450 ||(analogRead(tempr)/12.1)>50)

{Serial.print("F"); is\_MQ=false;Serial1.print("f");

}

}else{digitalWrite(MQ\_led,LOW);}

if(is\_nasos==true && is\_grow==false) //насос

{ Serial.println("is\_nasos");

digitalWrite(grow\_rele,LOW);

delay(10000);

digitalWrite(grow\_rele,HIGH);

is\_nasos=false;

}else{digitalWrite(grow\_rele,HIGH);}

if(is\_grow==true) //автополив

{ digitalWrite(grow\_led,HIGH);

Serial.print("dry:");

is\_grow\_work=digitalRead(grow\_sensor);

Serial.println(is\_grow\_work);

if(is\_grow\_work==true)

{digitalWrite(grow\_rele,LOW);

delay(10000);

digitalWrite(grow\_rele,HIGH);

}else{digitalWrite(grow\_rele,HIGH);}

}else{digitalWrite(grow\_led,LOW);}

}

Полный код программы для Android

MainActivity.java

**package** com.example.coder\_face.smarthome;  
  
**import** java.io.IOException;  
**import** java.io.InputStream;  
**import** java.io.OutputStream;  
**import** java.util.UUID;  
**import** android.os.Bundle;  
**import** android.os.Handler;  
**import** android.app.Activity;  
**import** android.util.Log;  
**import** android.view.View;  
**import** android.view.View.OnClickListener;  
**import** android.widget.Switch;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
**import** android.bluetooth.\*;  
**import** android.content.Intent;  
**import** android.widget.ToggleButton;  
  
**public class** MainActivity **extends** Activity {  
 **private static final int *REQUEST\_ENABLE\_BT*** = 1;  
 **final int ArduinoData** = 1;  
 **final** String **LOG\_TAG** = **"myLogs"**;  
 **private** BluetoothAdapter **btAdapter** = **null**;  
 **private** BluetoothSocket **btSocket** = **null**;  
 **private static** String *MacAddress* = **"20:16:06:12:01:81"**; *// MAC-адрес БТ модуля* **private static final** UUID ***MY\_UUID*** = UUID.*fromString*(**"00001101-0000-1000-8000-00805F9B34FB"**);  
 **private** ConnectedThred **MyThred** = **null**;  
 **public** TextView **mytext**;  
 Switch **tempr\_swich**,**grow\_swich**;  
 ToggleButton **thief\_button**,**conditioner\_button**,**smoke\_button**,**nasos\_button**;  
 Handler **h**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
  
 **btAdapter** = BluetoothAdapter.*getDefaultAdapter*();  
 **mytext** = (TextView) findViewById(R.id.***textView7***);  
  
 **if** (**btAdapter** != **null**){  
 **if** (**btAdapter**.isEnabled()){  
 **mytext**.setText(**"Bluetooth включен. Все отлично."**);  
 }**else** {  
 Intent enableBtIntent = **new** Intent(BluetoothAdapter.***ACTION\_REQUEST\_ENABLE***);  
 startActivityForResult(enableBtIntent, ***REQUEST\_ENABLE\_BT***);  
 }  
  
 }**else** {  
 MyError(**"Fatal Error"**, **"Bluetooth ОТСУТСТВУЕТ"**);  
 }  
  
 **tempr\_swich** = (Switch) findViewById(R.id.***switch3***);  
 **grow\_swich** = (Switch) findViewById(R.id.***switch1***);  
 **thief\_button** = (ToggleButton) findViewById(R.id.***toggleButton4***);  
 **smoke\_button** = (ToggleButton) findViewById(R.id.***toggleButton5***);  
 **conditioner\_button** = (ToggleButton) findViewById(R.id.***toggleButton2***);  
 **nasos\_button** = (ToggleButton) findViewById(R.id.***toggleButton***);  
  
 **tempr\_swich**.setOnClickListener(**new** OnClickListener() {  
 **public void** onClick(View v) {  
 **if** (**tempr\_swich**.isChecked()) {  
 **MyThred**.sendData(**"1"**);  
  
 **conditioner\_button**.setVisibility(View.***INVISIBLE***);  
 } **else** {  
 **MyThred**.sendData(**"0"**);  
 **conditioner\_button**.setVisibility(View.***VISIBLE***);  
  
 }  
 }  
 });  
  
  
  
 **thief\_button**.setOnClickListener(**new** OnClickListener() {  
 **public void** onClick(View v) {  
 **if** (**thief\_button**.isChecked()) {  
 **MyThred**.sendData(**"A"**);  
  
 *// mytext.setText("Отправлены данные: 1");* } **else** {  
 **MyThred**.sendData(**"a"**);  
 }  
 }  
 });  
  
 **smoke\_button**.setOnClickListener(**new** OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **if**(**smoke\_button**.isChecked())  
 {  
 **MyThred**.sendData(**"F"**);  
 }**else** {  
 **MyThred**.sendData(**"f"**);  
 }  
 }  
  
 });  
  
 **grow\_swich**.setOnClickListener(**new** OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **if** (**grow\_swich**.isChecked()) {  
 **MyThred**.sendData(**"4"**);  
 **nasos\_button**.setVisibility(View.***INVISIBLE***);  
 }  
 **else** {  
 **MyThred**.sendData(**"5"**);  
 **nasos\_button**.setVisibility(View.***VISIBLE***);  
 }  
  
 }  
 });  
 **conditioner\_button**.setOnClickListener(**new** OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **if** (**conditioner\_button**.isChecked()) {  
 **MyThred**.sendData(**"2"**);  
  
  
 } **else** {  
 **MyThred**.sendData(**"3"**);  
  
 }  
  
 }  
 });  
  
 **nasos\_button**.setOnClickListener(**new** OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **if**(**nasos\_button**.isChecked())  
 {  
 **MyThred**.sendData(**"6"**);  
 }**else** {  
 **MyThred**.sendData(**"7"**);  
 }  
 }  
 });  
  
 **h** = **new** Handler() {  
 **public void** handleMessage(android.os.Message msg) {  
 **switch** (msg.**what**) {  
 **case ArduinoData**:  
 **byte**[] readBuf = (**byte**[]) msg.**obj**;  
 String strIncom = **new** String(readBuf, 0, msg.**arg1**);  
 **mytext**.setText(**"Данные от Arduino: "** + strIncom);  
 **if**(strIncom.equals(**"0"**))  
 {  
 **tempr\_swich**.setChecked(**false**);  
 **conditioner\_button**.setVisibility(View.***VISIBLE***);  
 }  
 **if**(strIncom.equals(**"1"**))  
 {  
 **tempr\_swich**.setChecked(**true**);  
 **conditioner\_button**.setVisibility(View.***INVISIBLE***);  
 }  
 **if**(strIncom.equals(**"A"**))  
 {  
 **thief\_button**.setChecked(**true**);  
 }  
 **if**(strIncom.equals(**"a"**))  
 {  
 **thief\_button**.setChecked(**false**);  
 }  
 **if**(strIncom.equals(**"F"**))  
 {  
 **smoke\_button**.setChecked(**true**);  
 }  
 **if**(strIncom.equals(**"f"**))  
 {  
 **smoke\_button**.setChecked(**false**);  
 }  
 **if**(strIncom.equals(**"4"**))  
 {  
 **grow\_swich**.setChecked(**true**);  
 **nasos\_button**.setVisibility(View.***INVISIBLE***);  
 }  
 **if**(strIncom.equals(**"5"**))  
 {  
 **grow\_swich**.setChecked(**false**);  
 **nasos\_button**.setVisibility(View.***VISIBLE***);  
 }  
 **if**(strIncom.equals(**"6"**))  
 {  
 **nasos\_button**.setChecked(**true**);  
 }  
 **if**(strIncom.equals(**"7"**))  
 {  
 **nasos\_button**.setChecked(**false**);  
 }  
 **if**(strIncom.equals(**"2"**))  
 {  
 **conditioner\_button**.setChecked(**true**);  
 }  
 **if**(strIncom.equals(**"3"**))  
 {  
 **conditioner\_button**.setChecked(**false**);  
 }  
  
  
 **break**;  
 }  
 };  
 };  
  
 }  
  
 @Override  
 **public void** onResume() {  
 **super**.onResume();  
  
 BluetoothDevice device = **btAdapter**.getRemoteDevice(*MacAddress*);  
 Log.*d*(**LOG\_TAG**, **"\*\*\*Получили удаленный Device\*\*\*"**+device.getName());  
  
 **try** {  
 **btSocket** = device.createRfcommSocketToServiceRecord(***MY\_UUID***);  
 Log.*d*(**LOG\_TAG**, **"...Создали сокет..."**);  
 } **catch** (IOException e) {  
 MyError(**"Fatal Error"**, **"В onResume() Не могу создать сокет: "** + e.getMessage() + **"."**);  
 }  
  
 **btAdapter**.cancelDiscovery();  
 Log.*d*(**LOG\_TAG**, **"\*\*\*Отменили поиск других устройств\*\*\*"**);  
  
 Log.*d*(**LOG\_TAG**, **"\*\*\*Соединяемся...\*\*\*"**);  
 **try** {  
 **btSocket**.connect();  
 Log.*d*(**LOG\_TAG**, **"\*\*\*Соединение успешно установлено\*\*\*"**);  
 } **catch** (IOException e) {  
 **try** {  
 **btSocket**.close();  
 } **catch** (IOException e2) {  
 MyError(**"Fatal Error"**, **"В onResume() не могу закрыть сокет"** + e2.getMessage() + **"."**);  
 }  
 }  
  
 **MyThred** = **new** ConnectedThred(**btSocket**);  
 **MyThred**.start();  
 }  
  
 @Override  
 **public void** onPause() {  
 **super**.onPause();  
  
 Log.*d*(**LOG\_TAG**, **"...In onPause()..."**);  
  
 **if** (**MyThred**.status\_OutStrem() != **null**) {  
 **MyThred**.cancel();  
 }  
  
 **try** {  
 **btSocket**.close();  
 } **catch** (IOException e2) {  
 MyError(**"Fatal Error"**, **"В onPause() Не могу закрыть сокет"** + e2.getMessage() + **"."**);  
 }  
 }  
  
 **private void** MyError(String title, String message){  
 Toast.*makeText*(getBaseContext(), title + **" - "** + message, Toast.***LENGTH\_LONG***).show();  
 finish();  
 }  
  
  
 *//Отдельный поток для передачи данных* **private class** ConnectedThred **extends** Thread{  
 **private final** BluetoothSocket **copyBtSocket**;  
 **private final** OutputStream **OutStrem**;  
 **private final** InputStream **InStrem**;  
  
 **public** ConnectedThred(BluetoothSocket socket){  
 **copyBtSocket** = socket;  
 OutputStream tmpOut = **null**;  
 InputStream tmpIn = **null**;  
 **try**{  
 tmpOut = socket.getOutputStream();  
 tmpIn = socket.getInputStream();  
 } **catch** (IOException e){}  
  
 **OutStrem** = tmpOut;  
 **InStrem** = tmpIn;  
 }  
  
 **public void** run()  
 {  
 **byte**[] buffer = **new byte**[1024];  
 **int** bytes;  
  
 **while**(**true**){  
 **try**{  
 bytes = **InStrem**.read(buffer);  
 **h**.obtainMessage(**ArduinoData**, bytes, -1, buffer).sendToTarget();  
 }**catch**(IOException e){**break**;}  
  
 }  
  
 }  
  
 **public void** sendData(String message) {  
 **byte**[] msgBuffer = message.getBytes();  
 Log.*d*(**LOG\_TAG**, **"\*\*\*Отправляем данные: "** + message + **"\*\*\*"** );  
  
 **try** {  
 **OutStrem**.write(msgBuffer);  
 } **catch** (IOException e) {}  
 }  
  
 **public void** cancel(){  
 **try** {  
 **copyBtSocket**.close();  
 }**catch**(IOException e){}  
 }  
  
 **public** Object status\_OutStrem(){  
 **if** (**OutStrem** == **null**){**return null**;  
 }**else**{**return OutStrem**;}  
 }  
 }  
}

activity\_main.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:paddingBottom="@dimen/activity\_vertical\_margin"  
 android:paddingLeft="@dimen/activity\_horizontal\_margin"  
 android:paddingRight="@dimen/activity\_horizontal\_margin"  
 android:paddingTop="@dimen/activity\_vertical\_margin"  
 tools:context="com.example.coder\_face.smarthome.MainActivity"**>  
  
 <**Switch  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Автополив"  
 android:id="@+id/switch1"  
 android:checked="false"  
 android:layout\_below="@+id/switch3"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true"** />  
  
 <**Switch  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Климат-контроль"  
 android:id="@+id/switch3"  
 android:checked="false"  
 android:clickable="false"** />  
  
 <**ToggleButton  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/toggleButton2"  
 android:checked="false"  
 android:textOff="Кондеционер выключен"  
 android:textOn="Кондеционер включен"  
 android:layout\_alignTop="@+id/toggleButton"  
 android:layout\_alignParentRight="true"  
 android:layout\_alignParentEnd="true"  
 android:layout\_toRightOf="@+id/switch3"  
 android:layout\_toEndOf="@+id/switch3"** />  
  
 <**TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:textAppearance="?android:attr/textAppearanceLarge"  
 android:text="Сигнализации:"  
 android:id="@+id/textView6"  
 android:layout\_below="@+id/toggleButton2"  
 android:layout\_centerHorizontal="true"  
 android:layout\_marginTop="25dp"** />  
  
 <**ToggleButton  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="New ToggleButton"  
 android:id="@+id/toggleButton4"  
 android:layout\_below="@+id/textView6"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true"  
 android:checked="false"  
 android:layout\_toLeftOf="@+id/toggleButton2"  
 android:layout\_toStartOf="@+id/toggleButton2"  
 android:textOff="Сигнализация от воров выключена"  
 android:textOn="Сигнализация от воров включена"** />  
  
 <**ToggleButton  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="New ToggleButton"  
 android:id="@+id/toggleButton5"  
 android:layout\_alignTop="@+id/toggleButton4"  
 android:layout\_toRightOf="@+id/toggleButton4"  
 android:layout\_toEndOf="@+id/toggleButton4"  
 android:layout\_alignRight="@+id/toggleButton2"  
 android:layout\_alignEnd="@+id/toggleButton2"  
 android:textOff="Противопожарная сигнализация выключена"  
 android:textOn="Противопожарная сигнализация включена"** />  
  
 <**TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:textAppearance="?android:attr/textAppearanceLarge"  
 android:id="@+id/textView7"  
 android:layout\_alignRight="@+id/toggleButton5"  
 android:layout\_alignEnd="@+id/toggleButton5"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true"  
 android:layout\_alignParentBottom="true"  
 android:layout\_below="@+id/toggleButton5"  
 android:text="Данные от Arduino:"** />  
  
 <**ToggleButton  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="New ToggleButton"  
 android:id="@+id/toggleButton"  
 android:textOff="Насос выключен"  
 android:textOn="Насос включен"  
 android:layout\_below="@+id/switch1"  
 android:layout\_toLeftOf="@+id/toggleButton2"  
 android:layout\_toStartOf="@+id/toggleButton2"** />  
  
</**RelativeLayout**>

AndroidManifest.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 package="com.example.coder\_face.smarthome"**>  
 <**uses-permission android:name="android.permission.BLUETOOTH"**/>  
 <**uses-permission android:name="android.permission.BLUETOOTH\_ADMIN"**/>  
 <**application  
 android:allowBackup="true"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:supportsRtl="true"  
 android:theme="@style/AppTheme"**>  
 <**activity android:name=".MainActivity"**>  
 <**intent-filter**>  
 <**action android:name="android.intent.action.MAIN"** />  
  
 <**category android:name="android.intent.category.LAUNCHER"** />  
 </**intent-filter**>  
 </**activity**>  
 </**application**>  
  
</**manifest**>

Программа на ПК для отправки SMS

Региональная научно-практическая конференция по информатике, ИКТ и робототехнике.

1. Кондиционер можно включить/выключить вручную, только когда отключена функция климат-контроля. [↑](#footnote-ref-1)
2. Вручную насос можно включить/выключить, только если отключена функция автополива. [↑](#footnote-ref-2)
3. Вручную свет можно включить/выключить, только если отключена функция автоматического управления светом [↑](#footnote-ref-3)
4. Для работы модуля необходим интернет и подключение системы к ПК или серверу. [↑](#footnote-ref-4)
5. Для работы модуля необходим интернет и подключение системы к ПК или серверу. [↑](#footnote-ref-5)