Федеральное государственное автономное образовательное учреждение
высшего образования
Санкт-Петербургский политехнический университет Петра Великого
Высшая школа автоматизации и робототехники

НАУЧНО-ИССЛЕДОВАТЕЛЬСКАЯ РАБОТА
РАЗРАБОТКА АЛГОРИТМА РАБОТЫ СИСТЕМЫ АВТОМАТИЗИРОВАННОГО ПОДДЕРЖАНИЯ РОСТА РАСТЕНИЙ

по направлению подготовки 15.03.06 «Мехатроника и робототехника»
направленность (профиль) 15.03.06_01 «Проектирование и конструирование мехатронных модулей и механизмов роботов»

Выполнил
студент гр. 3331506/10102 И.О. Иванов

Руководитель
Доцент ВШАиР кф-м.н. М.С. Ананьевский

Санкт-Петербург
2024

Оглавление

1. ВВЕДЕНИЕ

КЛЮЧЕВЫЕ СЛОВА: ARDUINO МИКРОКОНТРОЛЛЕРЫ, ТЕПЛИЦА OMEGAGROW, ARDUINO IDE, ТЕХНОЛОГИЯ IOT.

Тема научно-исследовательской работы: «Разработка алгоритма работы системы автоматизированного поддержания роста растений».

Данная работа посвящена разработке алгоритма работы системы автоматизированного поддержания роста растений на базе Arduino.

Данный набор (теплица OMEGAGROW) предназначен для изучения основных понятий о технологии Интернет-вещей (IoT). Теплицы являются одними из самых популярных объектов, где активно при- меняется технология IoT [1]. Для более конкретного представления о концепции "Интернета вещей" приведем его определение [2].

Интернет вещей (англ. internet of things, IoT) — концепция сети передачи данных между физическими объектами («вещами»), оснащёнными встроенными средствами и технологиями для взаимодействия друг с другом или с внешней средой. Предполагается, что организация таких сетей способна перестроить экономические и общественные процессы, исключить из части действий и операций необходимость участия человека.

В результате данной работы алгоритм должен обеспечивать комфортные условия для роста растений. К ним относятся влажность почвы, интенсивность искусственного освещения и температура окружающего воздуха.

Помимо автоматизации процесса в работе было уделено внимание хранению кода и контролю его целостности. Был создан репозиторий кода, в котором хранятся файлы с расширением .ino (скетчи). Репозиторий имеет определенную структуру для поддержания порядка в файлах.

Файл с кодом, а также репозиторий, хранящий .ino размещен в публичном репозитории Github.com.

(URL: https://github.com/soomrack/M2024/tree/main/ivanov_i_o/srw, актуально на 18.05.2024).

2. ХОД РАБОТЫ

2.1. Постановка задачи

Можно выделить три крупные задачи, которые следует выполнить:

1) Изучить устройство и принцип работы теплицы OMEGAGROW.

Изучить составляющие, входящие в набор умной теплицы. Рассмотреть их прицнип действия и общее устройство теплицы.

2) Решить проблему контроля полива, а также приема и использование данных для датчика света.

Изучить вопрос и придумать решение проблемы контроля полива. Рассмотреть проблему приема данных с датчика освещенности.

3) Составить алгоритм для автоматизации работы теплицы.

Написать программу, загружаемую на контроллер Arduino.

Написанное программное обеспечение загружено на программируемый контроллер Arduino UNO на базе микроконтроллера ATmega328 [4]. Данный контроллер выбран как удобный для прототипирования проектов средних размеров, которые не требуют высокой мобильности. Несомненным плюсом выбора данного микроконтроллера является наличие его исходной документации, как для аппаратной, так и для программной части, так как данный проектявляется «open-source» разработкой. Кроме того, открытый исходный код не влечёт за собой никаких ограничений на применение микроконтроллера в коммерческих проектах, за исключением использования торговых марок, на которые распространяются авторские права компании Arduino (название и логотип). Также высокая популярность платформы Arduino способствует появлению большого количества библиотек, форумов и ресурсов, облегчающих ее использование.

Для работы с программным кодом используется приложение Arduino IDE.

Ход работы состоял из следующих шагов:

1. Изучить компоненты набора теплицы OMEGAGROW;
2. Найти информацию о состовляющих и о наборе в интернет-ресурсах;

3. Установка утилиты Arduino IDE, а также скачивание библиотеки TroykaDHT;
4. Решение проблемы с конвертацией освещенности;
5. Решение проблемы контроля полива почвы;
6. Написание программного кода, загрузка на Arduino UNO и проверка работоспособности;
7. Составление отчета по проделанной работе, а также публичного репозитория для хранения научно-исследовательской работы.

2.2. Изучение компонентов умной теплицы

В исходную систему автоматизированного поддержания роста растений (далее — теплица) включены три независимых системы контроля за условиями окружающей среды:

1) контроль освещенности (модуль фоторезистора и фитодиодная лента);
2) контроль влажности почвы (датчик влажности почвы и система капельного полива);
3) контроль температуры воздуха (датчик температуры и влажности воздуха и совокупность вентилятора с нагревательным элементом).

Исследование схемы питания и элементов управления

Первым этапом работы над теплицей было исследование схемы питания теплицы и возможностей управления исполнительными устройствами.

Главным питающим устройством является блок питания от сети 220В с выходными характеристиками 12В и 8А. Силовая часть теплицы, включающаявсе ее исполнительные устройства, питается напрямую от блока питания. С помощью регулируемого понижающего DC-DC преобразователя на основе микросхемы LM2596HVS [6] входящее напряжение с блока питания понижается и создает источник для питания логических компонентов теплицы, в том числе и программируемого контроллера.

Комплектация набора умной теплицы OMEGAGROW:

— Программируемый контроллер Arduino UNO;

— Понижающий DC-DC преобразователь LM2596HVS;

— Воздушно-водяной насос R385 с емкостью для забора воды;

— Модуль аналогового датчика влажности почвы;

— Система контроля температуры воздуха (вентилятор + нагревательный элемент);

— Модуль датчика освещенности на основе фоторезистора;

— Блок из двух 2-канальных модулей реле SRD-05VDC-SL-C для управления исполнительными устройствами теплицы [7];

— Дополнительный модуль реле аналогичной модели для переподклю- чения питания датчика DHT-21 (модификация для корректной работы, которая будет описана далее);

— Датчик температуры и влажности DHT-21;

— Фитодиодные ленты.

На рисунке 1 представлен общий вид теплицы.

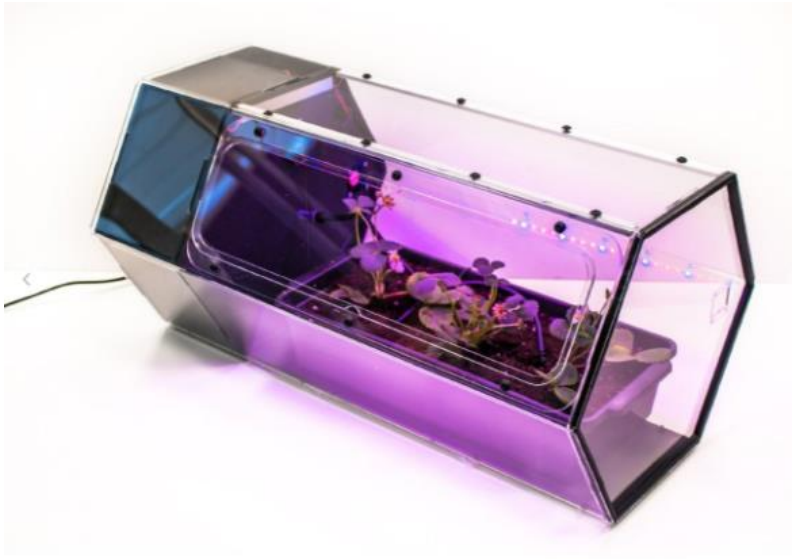


Рисунок 1 - Общий вид теплицы

Возможность управления исполнительными устройствами реализована с помощью блока из четырех электромеханических реле с опторазвязкой, то есть выходной сигнал может принимать только значения «включить» и «вы- ключить».

В дальнейшем процесс работы над каждой из систем контроля будет рассмотрен в отдельности.

Реализация контроля освещенности

На данном этапе работы было необходимо настроить реакцию системы на изменение внешней освещенности. В начала работы был выявлен один недостаток сборки аппаратной части: модуль датчика освещенности, представленный на рисунке 2, соединялся с контроллером через цифровой выход, который имеет лишь два состояния (логические «ноль» и «единица»). Низкий логический уровень соответствует высокому уровню внешней освещенности, высокий логический уровень — низкому. Однако подключение датчика на стороне микроконтроллера осуществляется в аналоговой вход, что подразуме-вает использование аналогового значения во время работы. Данный недостаток не позволяет настраивать пороговый уровень срабатывания датчика программно, это возможно сделать только с помощью подстроечного резистора на плате модуля.

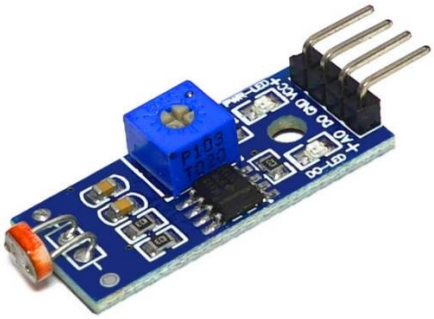


Рисунок 2 - Аналоговый датчик освещенности на фоторезисторе

Исходя из вышеописанных условий, был выбран следующий режим работы освещения: при появлении высокого сигнала на цифровом выходе датчика на реле отправляется высокий уровень, замыкающий разрыв цепи и подающий питание на фитодиодную ленту, пример которой представлен на рисунке 3.

Рисунок 3 - Пример фитодиодной ленты

Реализация контроля влажности почвы

На этой стадии работы необходимо было настроить верный режим полива при понижении влажности почвы до порогового значения. В данной конструкции теплицы стоит отметить выбранный для измерения влажности датчика (рисунок 4). Он основан на емкостном эффекте, поэтому не подвержен износу от коррозии, так как, в отличие от датчиков резистивного типа, не проводит микротоки между электродами.

Рисунок 4 - Датчик влажности почвы

Величина показаний с датчика обратно пропорциональна влажности почвы (сигнал, равный 1023 соответствует нахождение на воздухе, 0 – погружению в воду). Также в систему контроля влажности почвы входит воздушно-водяной насос, представленный на рисунке 5, на основе электродви-гателя постоянного тока R385. Алгоритм работы системы капельного полива был выбран с учетом задержки на распространение воды до датчика.

Рисунок 5 - Воздушно-водяная помпа

Реализация контроля температуры воздуха

Данный этап разработки предполагает контроль за температурой непосредственно внутри рабочего пространства теплицы, где размещаются растения. Управляющая система состоит из комплексного датчика температуры и влажности воздуха DHT-21 [8] (рисунок 7), вентилятора и нагревательного элемента установленного непосредственно на пути воздушного потока, выходящего из вентилятора. Нагревательный элемент состоит из металла с высоким сопротивлением и является простейшим преобразователем электрическойэнергии в тепловую.

Для снятия данных с данной модели датчика требуется подключить библиотеку, реализующую специальный протокол передачи данных [9], благодаря которой происходит обработка 40-битового сообщения от датчика. Выбор данной модели датчика может быть не оправдан, так как младшие модели серии DHTтакже обладают приемлемым диапазоном измерений и точностью.

Рисунок 6 - Датчик модели DHT-21

Рисунок 7 – Вентилятор постоянного тока

Совокупность включенных вентилятора и нагревательного элемента (модель вентилятора, идентичная используемой изображена на рисунке 6) дает возможность нагревать рабочее пространство теплицы. При этом сила воздушного потока является достаточно высокой, чтобы нагревательный элемент не вызывал чрезмерное повышение температуры корпуса. Одиночно включенный вентилятор позволяет понизить температуру в рабочем пространстве теплицы, чтобы избежать перегрева растений.

2.3. Решение основных проблем

Конвертация получаемых значений для влажности почвы

Для удобства и читаемости написанного алгоритма было принято решение использовать отдельную функцию, для перевода получаемых с датчика значений в процентном виде. Данная функция представлена ниже.

```
double soil_humidity_convert(int value)
{
  double persents;
  persents = (double)((1023 - value) * 100 / 1024);
  return persents;
}
```

Реализация контроля влажности (предотвращение залития почвы)

Указанные в блок-схеме временные интервалы подобраны экспериментально. Выбор периода полива основывается на количестве воды, необходимое для увлажнения объема почвы, размещенного в рабочей области теплицы. Выбор периода проверки данных датчика выбран эксперементально, для учета времени распространения воды. Блок-схема алгоритма представлена на рисунке 8.

Рисунок 8 - Алгоритм работы системы капельного полива

Реализация алгоритма управления

Программа управления реализована с помощью среды разработки Arduino IDE [2]. Необходимая библиотека из стандартного менеджера библиотек среды разработки. Было принято решение реализовать каждую из систем контроля в качестве отдельной функции с возможностью ввода таких параметров, как минимальная и максимальная температура воздуха и пороговое значение влажности. При написании программы также необходимо было учитывать ограничение на период между снятиями данных с датчика температуры.

3. ЗАКЛЮЧЕНИЕ

В результате проведения научной работы были исследованы возможности умной теплицы OMEGAGROW. Также были изучены возможности и параметры компонентов, входящих в состав набора. Использованные в ходе выполнения работы решения оправдали себя и привели к приемлимому результату. Был создан алгоритм для работы системы автоматизированного поддержания роста растений.

При необходимости можно улучшить техническую составляющую теплицы. К примеру, можно использовать новые датчики или использовать экран для вывода времени или других данных, необходимых для лучшего понимания работы теплицы. Также были онаружены некоторые проблемы, сильно влияющие на работу с теплицей.

Кабель-менеджмент. Кабель-менеджмент электронной части теплицы не предоставляет возможности доступа к отдельным проводам, так как все провода соединяются в косу в области около клемм питания.

Неудобство присоединения крышки. Крышка теплицы напрямую свя- зана с корпусом теплицы при помощи двух проводов от порта для подключе- ния блока питания. Это делает затруднительным ее отсоединение от остальнойтеплицы, что при ограниченной длине проводов не позволяет удобно располо-жить ее в рабочем пространстве.

Отсутствие свободных гнезд в клеммах. В клеммах, используемых для соединения проводов питания, не предусмотрено свободных гнезд для подключения дополнительных проводов, так как имеющиеся провода питания достаточно плотно занимают место в имеющихся гнездах.

В приложении представлены блок-схема всего алгоритма, а также программный код.

СПИСОК ЛИТЕРАТУРЫ

1. Умная теплица OMEGAGROW — https://omegabot.ru/product/28
2. Интернет вещей (Материал из Википедии) — https://ru.wikipe-dia.org/wiki/Интернет_вещей
3. Умная теплица на базе arduino из подручного материала с регуляторомтемпературы — https://habr.com/ru/post/536666/
4. Arduino UNO R3 — Product Reference Manual: https://docs.arduino.cc/re-sources/datasheets/A000066-datasheet.pdf
5. Arduino IDE 2 Tutorials — https://docs.arduino.cc/software/ide-v2
6. LM2596HV 60V 3A 150kHz Step-Down Voltage Regulator Datasheet — http://amperkot.ru/static/3236/uploads/datasheets/LM2596HV.PDF
7. Даташит для SRD-05VDC-SL-C, реле 1 переключатель 5VDC/10A,125VAC — https://static.chipdip.ru/lib/642/DOC012642672.pdf
8. Temperature and humidity module AM2301 Product Manual —https://smartep.ru/datasheets/sensors/DHT21%20AM2301.pdf
9. Библиотека для Arduino, позволяющая считывать данные датчиков се-рии DHT — https://github.com/amperka/TroykaDHT

Приложение

Приложение 1. Программный код.

```
#define PIN_LIGHT_SENSOR A0
#define PIN_LIGHT 6
#define PIN_HUMIDITY_SENSOR A1
#define PIN_WATER_PUMP 5
#define PIN_DHT_SENSOR 2
#define PIN_VEN_HEAT 4
#define PIN_VEN 7
#define ON 1
#define OFF 0

#include <TroykaDHT.h>
DHT dht_sensor(PIN_DHT_SENSOR, DHT21);

int PROGRAMM_CHECK_TIME = 1;

struct Climate {
  int norm_luminosity;
  double min_air_temp;
  double max_air_temp;
  double min_air_humidity;
  double max_air_humidity;
  int norm_soil_humidity;
  int watering_time;
  int ven_time;
};

struct Sensors {
  int hours;
  int minutes;
  int seconds;
  int luminosity;
  double air_temp;
  double air_humidity;
  double soil_humidity;
};

struct State {
  long long int ven_time;
  long long int watering_time;
  long long int last_watering;
  bool regular_ven;
  bool light;
  bool ven;
  bool pump;
  bool heat;
};
```

```
State state;
Climate clim;
Sensors sens;


void setup() {
  Serial.begin(9600);
  dht_sensor.begin();
  pinMode(PIN_LIGHT, OUTPUT);
  pinMode(PIN_WATER_PUMP, OUTPUT);
  pinMode(PIN_DHT_SENSOR, INPUT);
  pinMode(PIN_VEN_HEAT, OUTPUT);
  pinMode(PIN_VEN, OUTPUT);
}


void plant() {
  clim.norm_luminosity = 700;
  clim.min_air_temp = 18;
  clim.max_air_temp = 30;
  clim.norm_soil_humidity = 75;
  clim.min_air_humidity = 50;
  clim.max_air_humidity = 60;
  clim.watering_time = 60000;
  clim.ven_time = 60000;
}


void set_time() {
  sens.seconds = millis() / 1000;

  if (sens.seconds == 60) {
    sens.minutes += 1;
    sens.seconds = 0;
  }
  if (sens.minutes == 60) {
    sens.hours += 1;
    sens.minutes = 0;
  }
  if (sens.hours == 24) {
    sens.hours = 0;
  }
}


double soil_humidity_convert(int value) {
  double persents;
  persents = (double)((1023 - value) * 100 / 1024);
  return persents;
}
```

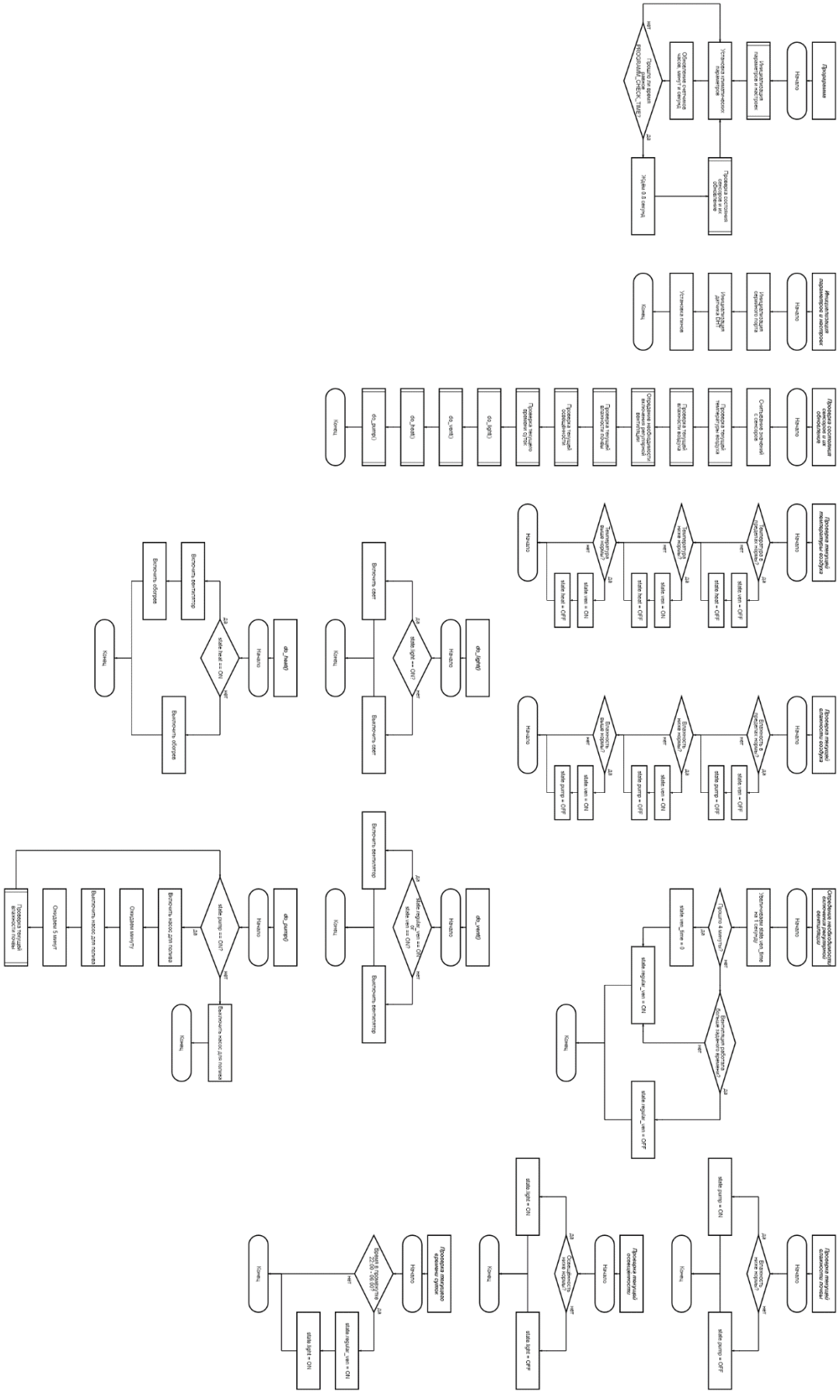
```
void get_sensors() {
  dht_sensor.read();

  sens.soil_humidity = soil_humidity_convert(analogRead(PIN_HUMIDITY_SENSOR));
  sens.luminosity = analogRead(PIN_LIGHT_SENSOR);
  sens.air_temp = dht_sensor.getTemperatureC();
  sens.air_humidity = dht_sensor.getHumidity();
}

void ventilation() {
  state.ven_time += PROGRAMM_CHECK_TIME * 1000;

  if (sens.minutes % 4 == 0) {
    state.regular_ven = ON;
    state.ven_time = 0;
  } else {
    if (state.ven_time > clim.ven_time) {
      state.regular_ven = OFF;
    } else {
      state.regular_ven = ON;
    }
  }
}


void air_temp() {
  if (sens.air_temp >= clim.min_air_temp && sens.air_temp <= clim.max_air_temp) {
    state.ven = OFF;
    state.heat = OFF;
  }

  if (sens.air_temp < clim.min_air_temp) {
    state.ven = ON;
    state.heat = ON;
  }

  if (sens.air_temp > clim.max_air_temp) {
    state.ven = ON;
    state.heat = OFF;
  }
}
```

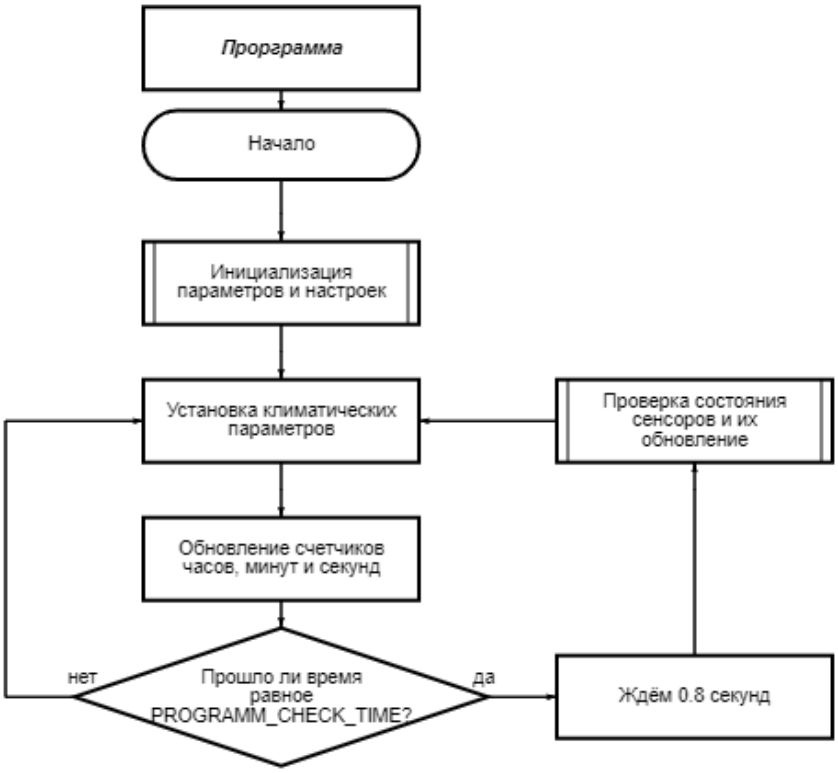
```
void air_humidity()
{
  if (sens.air_humidity >= clim.min_air_humidity && sens.air_humidity <=
clim.max_air_humidity) {
    state.ven = OFF;
    state.pump = OFF;
  }
  if (sens.air_humidity < clim.min_air_humidity) {
    state.ven = OFF;
    state.pump = ON;
  }

  if (sens.air_humidity > clim.max_air_humidity) {
    state.ven = ON;
    state.pump = OFF;
  }
}


void soil_humidity()
{
  if (sens.soil_humidity < clim.norm_soil_humidity) {
    state.pump = ON;
  } else {
    state.pump = OFF;
  }
}


void light()
{
  if (sens.luminosity >= clim.norm_luminosity) {
    state.light = OFF;
  } else {
    state.light = ON;
  }
}


void day_night()
{
  if (sens.hours < 6 || sens.hours > 22){
    state.regular_ven = ON;
    state.light = ON;
  }
}
```

```
void do_light()
{
  digitalWrite(PIN_LIGHT, state.light);
}


void do_heat()
{
  if (state.heat == ON) {
    digitalWrite(PIN_VEN, ON);
    digitalWrite(PIN_VEN_HEAT, ON);
  } else {
    digitalWrite(PIN_VEN_HEAT, OFF);
  }
}


void do_vent()
{
  if (state.regular_ven == ON){
    digitalWrite(PIN_VEN, ON);
  } else if (state.ven == ON) {
    digitalWrite(PIN_VEN, ON);
  } else {
    digitalWrite(PIN_VEN, OFF);
  }
}


void do_pump()
{
  if (state.last_watering > 300000) {

    if (state.pump == ON) {
      digitalWrite(PIN_WATER_PUMP, state.pump);
      state.watering_time += PROGRAMM_CHECK_TIME * 1000;

      if (state.watering_time > clim.watering_time) {
        digitalWrite(PIN_WATER_PUMP, OFF);
        state.watering_time = 0;
        state.last_watering = 0;
      }
    } else {
      digitalWrite(PIN_WATER_PUMP, state.pump);
    }
  }
  state.last_watering += PROGRAMM_CHECK_TIME * 1000;
}
```

```
void periodic_check()
{
  get_sensors();

  air_temp();
  air_humidity();
  ventilation();
  soil_humidity();
  light();

  day_night();

  do_light();
  do_vent();
  do_heat();
  do_pump();
}


void loop()
{
  plant();
  set_time();

  if (sens.seconds % PROGRAMM_CHECK_TIME == 0) {
    delay(800);
    periodic_check();
  }
}
```

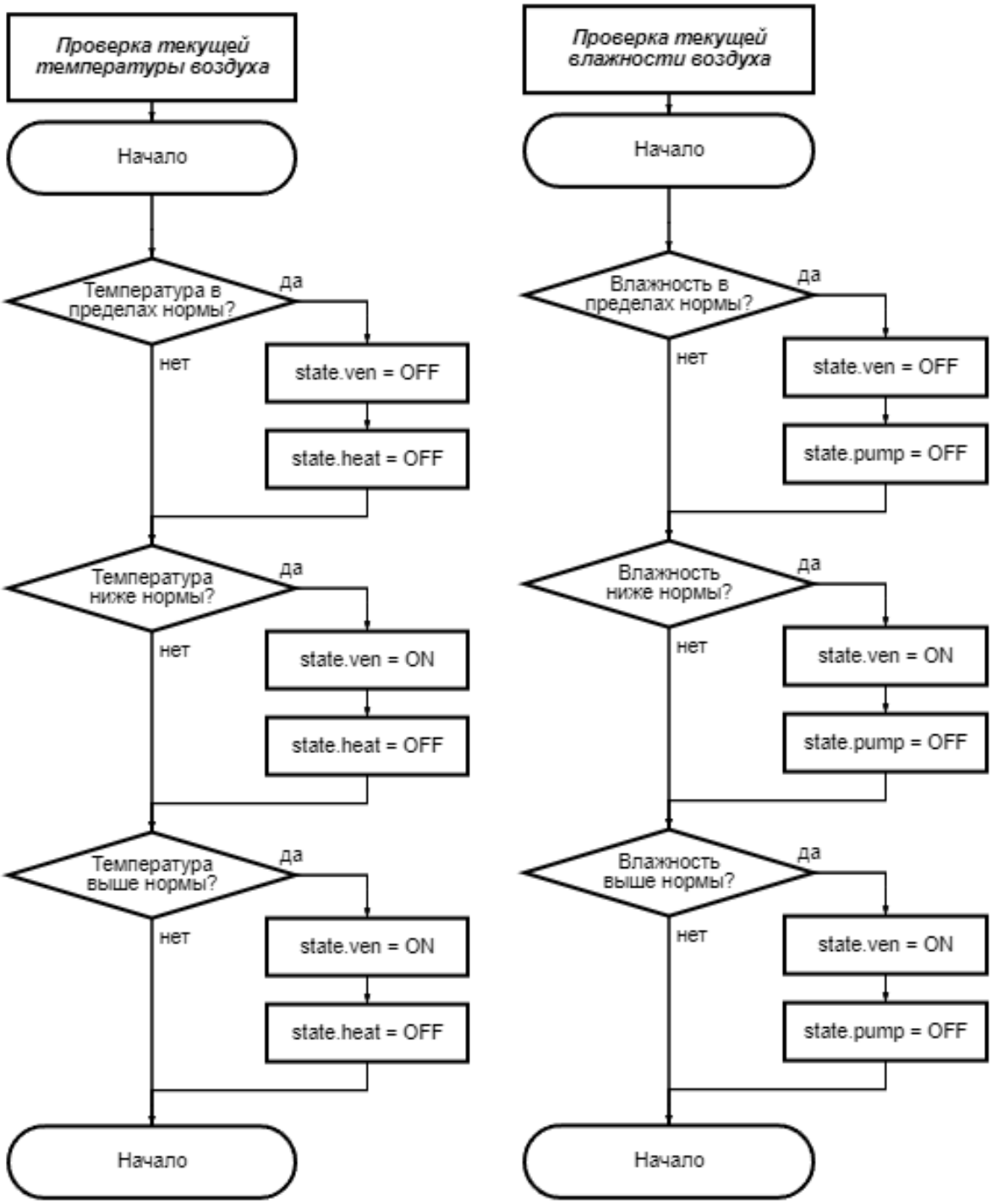
Приложение 2. Блок-схема алгоритма.

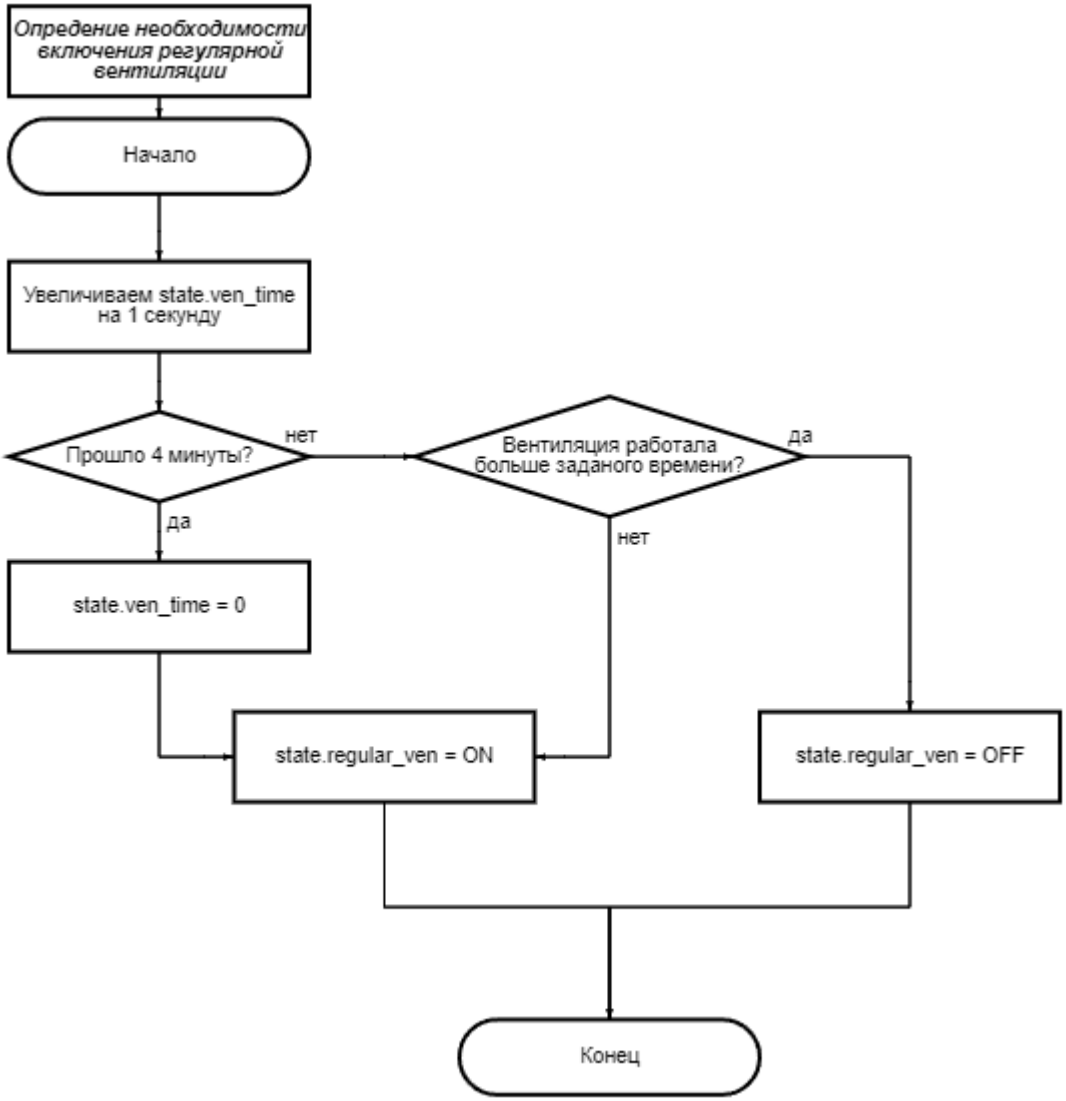



Программа
Начало
Инициализация параметров и настроек
Установка климатических параметров
Обновление счетчиков часов, минут и секунд
Прошло ли время равное PROGRAMM_CHECK_TIME?
нет
да
Ждём 0.8 секунд
Проверка состояния сенсоров и их обновление


Инициализация параметров и настроек
Начало
Инициализация серийного порта
Инициализация датчика DHT
Установка пинов
Конец

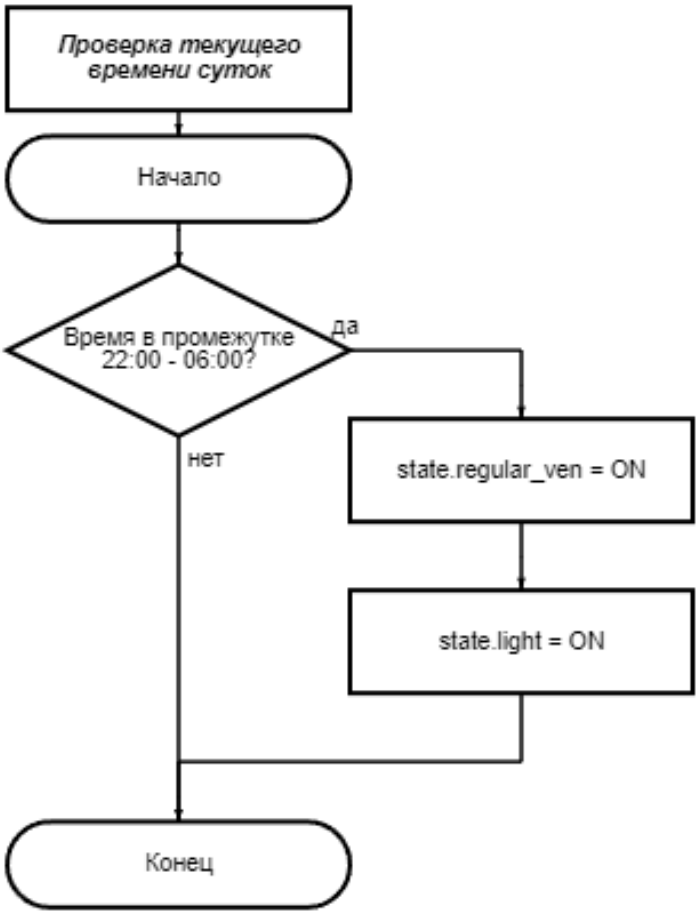

Проверка состояния сенсоров и их обновление
Начало
Считывание значений с сенсоров
Проверка текущей температуры воздуха
Проверка текущей влажности воздуха
Опредение необходимости включения регулярной вентиляции
Проверка текущей влажности почвы
Проверка текущей освещённости
Проверка текущего времени суток
do_light()
do_vent()
do_heat()
do_pump()
Конец

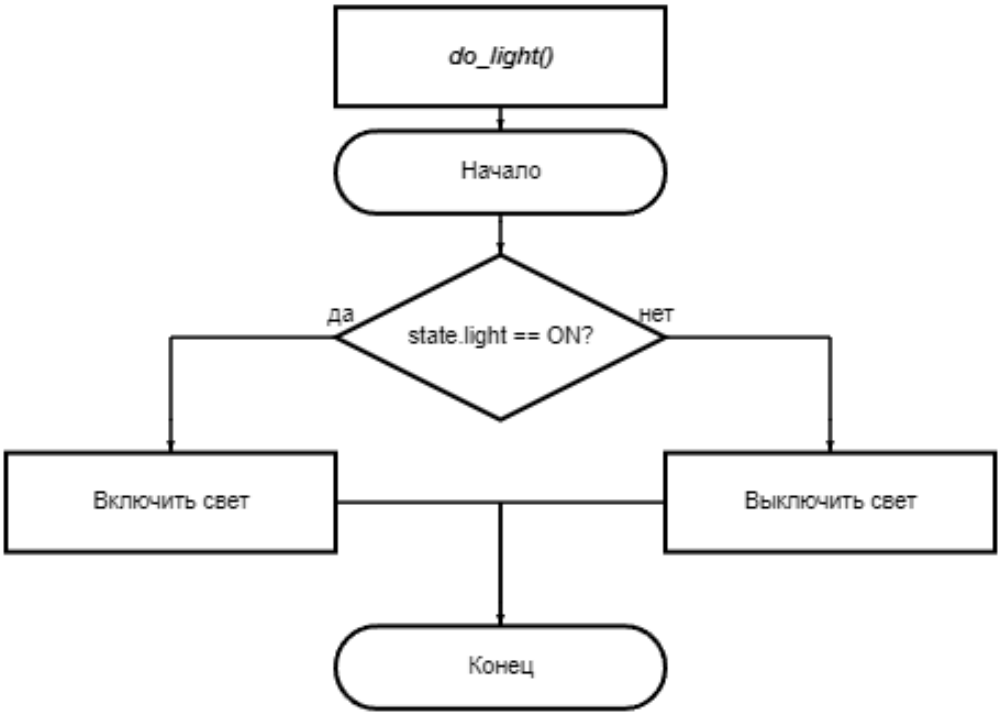

Проверка текущей температуры воздуха
Начало
Температура в пределах нормы?
да
нет
state.ven = OFF
state.heat = OFF
Температура ниже нормы?
да
нет
state.ven = ON
state.heat = OFF
Температура выше нормы?
да
нет
state.ven = ON
state.heat = OFF
Начало
Проверка текущей влажности воздуха
Начало
Влажность в пределах нормы?
да
нет
state.ven = OFF
state.pump = OFF
Влажность ниже нормы?
да
нет
state.ven = ON
state.pump = OFF
Влажность выше нормы?
да
нет
state.ven = ON
state.pump = OFF
Начало

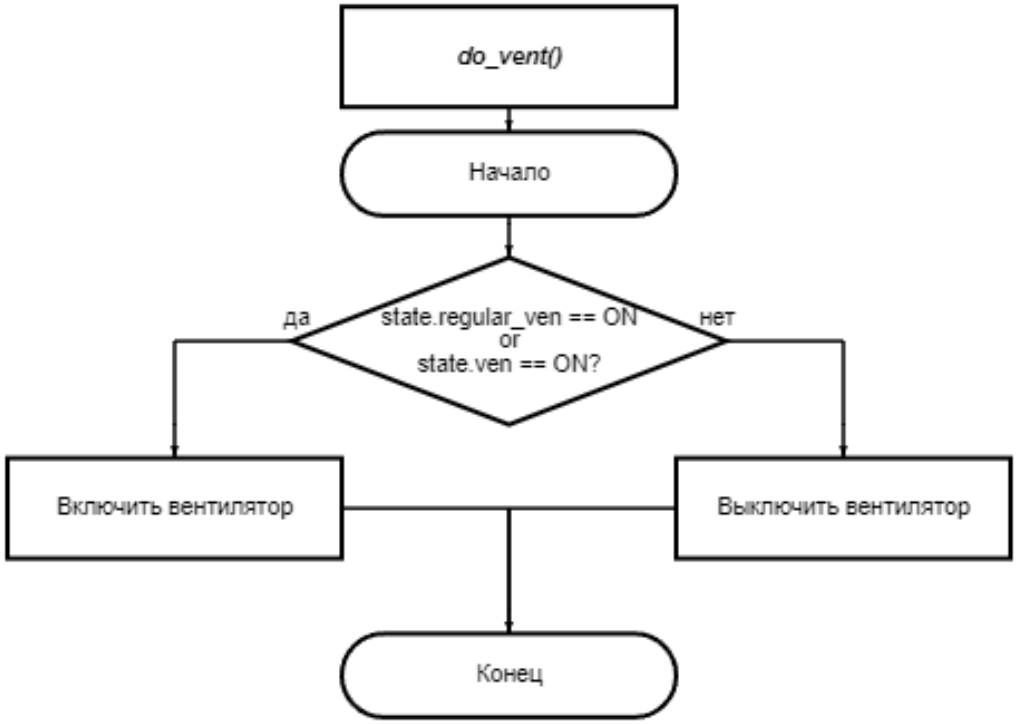

Определение необходимости включения регулярной вентиляции
Начало
Увеличиваем state.ven_time на 1 секунду
Прошло 4 минуты?
нет
да
Вентиляция работала больше заданого времени?
да
нет
state.ven_time = 0
state.regular_ven = ON
state.regular_ven = OFF
Конец

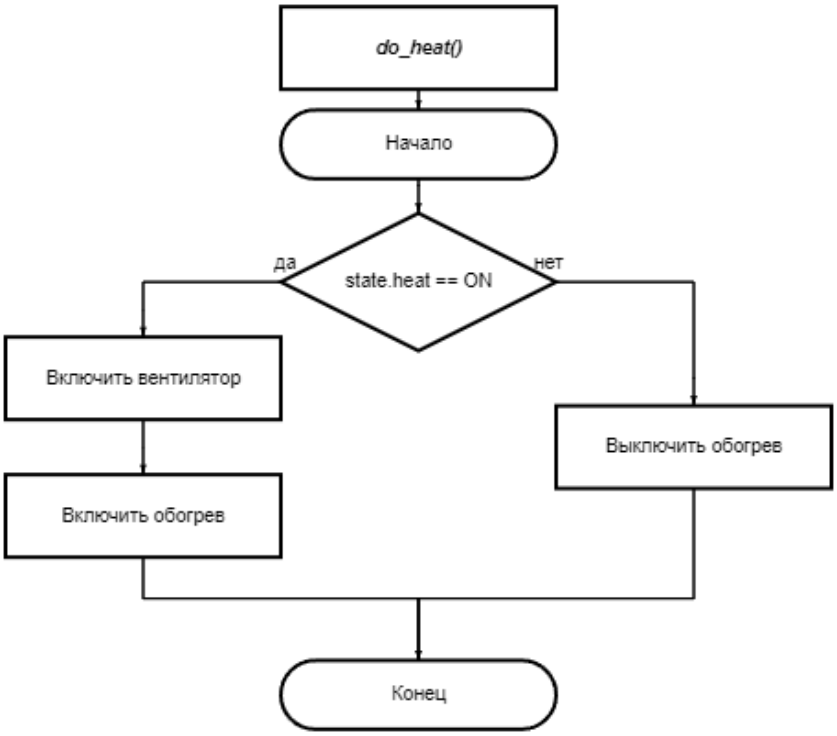

Проверка текущей влажности почвы
Начало
да
Влажность ниже нормы?
нет
state.pump = ON
state.pump = OFF
Конец

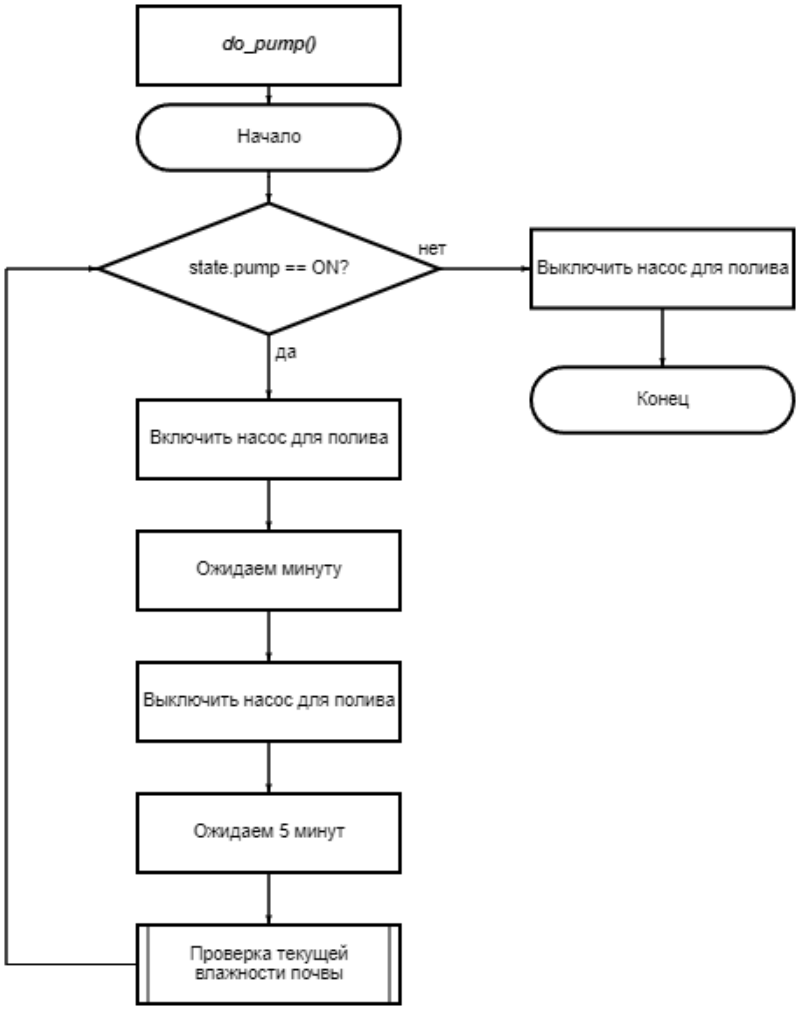

Проверка текущей освещённости
Начало
да
Освещённость ниже нормы?
нет
state.light = ON
state.light = OFF
Конец


Проверка текущего времени суток
Начало
Время в промежутке 22:00 - 06:00?
да
нет
state.regular_ven = ON
state.light = ON
Конец


do_light()
Начало
да
state.light == ON?
нет
Включить свет
Выключить свет
Конец


do_vent()
Начало
да
state.regular_ven == ON
or
state.ven == ON?
нет
Включить вентилятор
Выключить вентилятор
Конец


do_heat()
Начало
да
state.heat == ON
нет
Включить вентилятор
Включить обогрев
Выключить обогрев
Конец


do_pump()
Начало
state.pump == ON?
нет
Выключить насос для полива
Конец
да
Включить насос для полива
Ожидаем минуту
Выключить насос для полива
Ожидаем 5 минут
Проверка текущей влажности почвы