МАОУ "Усть-Кубинский центр образования"

Умный курятник

Ученик 10 класса:

Сакулин Иван Михайлович

Руководитель:

Андреев Николай Николаевич

с. Устье, 2023

**Оглавление**

[Вступление 3](#__RefHeading___1)

[Теоретическая часть 3](#__RefHeading___28)

[1.1. Выбор сред и инструментов разработки. 3](#__RefHeading___14)

[1.2. Arduino. ArduinoIDE. Atmega328p. ESP8266. 3](#__RefHeading___11)

[1.3. Контроль версий. 3](#__RefHeading___16)

[1.4. Android-приложение 4](#__RefHeading___17)

[1.5 work 4](#__RefHeading___39)

[Практическая часть 4](#__RefHeading___18)

[1. Начало. 4](#__RefHeading___5)

[2. Разработка первоначальной печатной платы. 4](#__RefHeading___29)

[3. Создание первоначального приложения и локального сервера на esp8266. 4](#__RefHeading___30)

[4. Первый функционал 4](#__RefHeading___31)

[5. Приложение 4](#__RefHeading___34)

[6. Наполнение 5](#__RefHeading___37)

[7. work 5](#__RefHeading___38)

[Заключение 5](#__RefHeading___6)

[Список литературы 5](#__RefHeading___7)

[Приложения 6](#__RefHeading___8)

# Вступление

Актуальность проекта:

Цель проекта:

Задачи:

План работы:

# Теоретическая часть

## 1.1. Выбор сред и инструментов разработки.

Очень важные части в реализации проекта: подобрать достаточно удобные инструменты и расходные материалы, с помощью которых можно реализовать необходимые функции, проработать скелет всей системы, /\*...\*/ .

В процесе работы я смог прийти к следующей основе:

/\*Картинку сюды: Андроид-приложение/аппаратный экран и энкодер – esp8266 – atmega328 – приводы, датчики.\*/

## 1.2. Arduino. ArduinoIDE. Atmega328p. ESP8266.

Arduino — это небольшая управляющая плата с собственным процессором и памятью. В процессор Ардуино можно загрузить программу (скетч), которая будет управлять устройствами по заданному алгоритму, в т.ч. используя датчики.

Скетчи удобно разрабатывать в Arduino IDE на языке C++. Это самая мощная среда разработки, использовать что-то другое намного сложнее.

Для макетирования и сборки первых прототипов такая плата подходит идеально, что не всегда скажешь о реализации: на них много светодиодов, стабилизаторов и прочего "обвеса", который может не использоваться, но всё равно будет потреблять энергию. Поэтому в серьёзных проектах лучше использовать чип AVR, который имеет в себе процессор и память, самый популярный из них atmega328p – его я использую, только с необходимым "обвесом".

В проекте atmega328p занимается только частью управления курятником и обработкой датчиков.

Связь основного контроллера с пользователем с целью настройки и проверки происходит посредством более мощного чипа с намного меньшим количеством пинов (выводы и вводы для внешнего воздействия), большим потрелением энергии, но с wi-fi и большей скоростью работы – esp8266. Этот чип не из семейства AVR, но также поддерживается в Arduino IDE.

## 1.3. Контроль версий.

Проект получился весьма объёмный по количеству кода и прочих файлов, поэтому я прибег к GitHub – это удобный инструмент для контроля версий, с его помощью можно без лишних усилий откатить проект, если вдруг что-то важное или неизвестное перестанет работать.

## 1.4. Android-приложение

Приложения для телефона я делал с помощью Android Studio на языке kotlin. Большая часть времени ушла на создание внутреннего сервиса запросов. Во включенном состоянии он параллельно основному потоку опрашивает контроллер и комуницирует с ним и пользователем через графический интерфейс.

## 1.5 work

EasyEDA, привода

# Практическая часть

## 1. Начало.

В практической части я описал выполнение каждого пункта плана.

Началось всё, конечно, с поиска проблемы. После долгих раздумий, я пришёл к выводу: неплохо бы было сделать свой курятник чуть более автономным и попытаться сделать стабильную версию, чтобы в будущем предложить её соседям и родственникам. Я здраво оценивал свои силы и осознавал, что на это времени не хватит, поэтому в рамках проекта ограничил свою цель до размеров своего курятника.

## 2. Разработка первоначальной печатной платы.

Первое что я сделал – развёл плату в EasyEDA, опираясь на те части, которые должны присутствовать в проекте. Заказал её на JLCPCB. А в процессе разработки отмечал все недостатки и ошибки в ней.

/\* В конечном итоге я сделал новую более совершенную разводку платы и заказал её. \*/

## 3. Создание первоначального приложения и локального сервера на esp8266.

Я реализовал простейший интерфейс с парой кнопок и нестабильный поток в приложении. На esp8266 основательно написал следующее: wi-fi точка доступа, сервер, универсальная структура обработчиков http-запросов, serial-соединение с основным чипом, который на этом этапе умел только мигать лампочкой по команде, debug инструменты.

## 4. Первый функционал

Приводы и механизмы – самое сложное, поэтому для начала я подключил к контроллеру модуль реального времени (RTC DS3231 mini) и датчик температуры, влажности и даления (BME280). Затем – прописал передачу этих значений на телефон.

## 5. Приложение

Чтобы написать хорошее, стабильное приложение мне пришлось переписывать всю структуру несколько раз, пока я не добился идеального результата: activity – то, с чего всё начинается и запускается, service для общения с контроллером и несколько fragment 'ов – страниц с функционалом.

## 6. Наполнение

Дальшейшие дейстия - создание реальных устройств, таких как поилка, кормушка с приводами, свет, обогрев, и их включение в созданную основу.

## 7. work

# Заключение

Текст

# Список литературы

Текст

# Приложения

Ничего пока нет