СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc116863761)

[1 ОБЗОР ЛИТЕРАТУРЫ 4](#_Toc116863762)

[1.1 Состав устройства 4](#_Toc116863763)

[1.2 Микроконтроллеры 4](#_Toc116863764)

[1.3 Датчики освещенности 5](#_Toc116863765)

[1.4 Датчики температуры 5](#_Toc116863766)

[1.5 Датчики влажности и атмосферного давления 6](#_Toc116863767)

[1.6 Модули часов реального времени 6](#_Toc116863768)

[1.7 Светодиоды для индикации 7](#_Toc116863769)

[1.8 Устройства управления внешней нагрузкой 7](#_Toc116863770)

[1.9 Модуль информирования пользователя 8](#_Toc116863771)

[2 РАЗРАБОТКА СТРУКТУРЫ УСТРОЙСТВА 9](#_Toc116863772)

[2.1 Постановка задачи 9](#_Toc116863773)

[2.2 Определение компонентов структуры устройства 9](#_Toc116863774)

[2.3 Взаимодействие компонентов устройства 10](#_Toc116863775)

[3 ОБОСНОВАНИЕ ВЫБОРА УЗЛОВ, ЭЛЕМЕНТОВ ФУНКЦИОНАЛЬНОЙ СХЕМЫ УСТРОЙСТВА 11](#_Toc116863776)

[3.1 Обоснование выбора микроконтроллеров 11](#_Toc116863777)

[3.2 Обоснование выбора датчика горючих газов 11](#_Toc116863778)

[3.3 Обоснование выбора датчика освещенности 11](#_Toc116863779)

[3.4 Обоснование выбора ультразвукового датчика расстояния 11](#_Toc116863780)

[3.5 Обоснование выбора модулей радиопередачи 12](#_Toc116863781)

[3.6 Обоснование выбора драйвера моторов 12](#_Toc116863782)

[3.7 Обоснование выбора мотор-редукторов 12](#_Toc116863783)

[3.8 Обоснование выбора модуля воспроизведения звука 13](#_Toc116863784)

[4 РАЗРАБОТКА ПРИНЦИПИАЛЬНОЙ ЭЛЕКТРИЧЕСКОЙ СХЕМЫ УСТРОЙСТВА 14](#_Toc116863785)

[4.1 Расчёт мощности элементов схемы 14](#_Toc116863786)

[4.2 Расчёт нагрузки светодиодов 15](#_Toc116863787)

[4.3 Микроконтроллеры 16](#_Toc116863788)

[4.4 Датчик горючих газов 16](#_Toc116863789)

[4.5 Датчик освещенности 16](#_Toc116863790)

[4.6 Ультразвуковой датчик расстояния 17](#_Toc116863791)

[4.7 Модуль радиопередачи 17](#_Toc116863792)

[4.8 Драйвер моторов 17](#_Toc116863793)

[4.9 Мотор-редукторы 17](#_Toc116863794)

[4.10 Пьезодинамик 17](#_Toc116863795)

[4.11 Джойстики 18](#_Toc116863796)

[5 РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 19](#_Toc116863797)

[5.1 Требования к разработке программного обеспечения 19](#_Toc116863798)

[5.2 Блок-схема алгоритма 19](#_Toc116863799)

[5.3 Исходный код программы для устройства управления 20](#_Toc116863800)

[5.4 Исходный код программы для передвижного устройства 20](#_Toc116863801)

[ЗАКЛЮЧЕНИЕ 22](#_Toc116863802)

[ПРИЛОЖЕНИЕ А 25](#_Toc116863803)

[ПРИЛОЖЕНИЕ Б 26](#_Toc116863804)

[ПРИЛОЖЕНИЕ В 27](#_Toc116863805)

[ПРИЛОЖЕНИЕ Г 28](#_Toc116863806)

[ПРИЛОЖЕНИЕ Д 29](#_Toc116863807)

[ПРИЛОЖЕНИЕ Е 34](#_Toc116863808)

[ПРИЛОЖЕНИЕ Ж 35](#_Toc116863809)

# ВВЕДЕНИЕ

Вставить тут

# ОБЗОР ЛИТЕРАТУРЫ

## Состав устройства

Проектируемое устройство должно решать поставленные задачи по мониторингу за параметрами окружающей среды в рамках тепличного комбината и при необходимости доведения эти параметров до состояния нормы. Данный мониторинг включает в себя отслеживание температуры и влажности, атмосферного давления и уровня освещенности, а также информирование пользователя об измеряемых параметрах. Для решения вышеперечисленных задач проектируемое устройство должно включать в себя:

- микроконтроллер

- датчик освещенности

- датчик температуры

- датчик влажности и атмосферного давления

- модуль часов реального времени

- светодиоды для индикации

- модуль управления внешней нагрузкой

- модуль информирования пользователя

## Микроконтроллеры

Существует огромное разнообразие плат с разными микроконтроллерами. Они могут различаться по исполнению, мощности, энергопотреблению, наличию периферии.

При выборе микроконтроллера для текущего проекты были рассмотрены отладочные платы на Arduino и STM (см таблицу 1.1)

Таблица 1.1 — Сравнение микроконтроллеров

|  |  |  |  |
| --- | --- | --- | --- |
| **Параметры сравнения** | **Esp - 01** | **Arduino Nano** | **STM32F4DISCOVERY** |
| Микроконтроллер | Xtensa L106 | ATmega328p | STM32F407VGT6 |
| Входное напряжение | 5 – 20 В | 6 – 20 В | 3- 5 В |
| Флэш-память | 1024 Кб | 32 Кб | 1024 Кб |
| ОЗУ | 96 Кб | 2 Кб | 192 Кб |
| Тактовая частота | 80 МГц | 16 МГц | 168 МГц |
| Разрядность | 32 бит | 8 бит | 32 бит |
| Цифровые  входы/выходы | 11 шт | 14 шт | 82 шт |
| Аналоговые  входы/выходы | 1 шт | 6 шт | 16 шт |
| Выходное напряжение | 3.3В | 3.3В, 5 В | 3.3В |

Продолжение таблицы 1.1

|  |  |  |  |
| --- | --- | --- | --- |
| Рабочая температура | от -25 до +85 ºС | от -25 до +85 ºС | от -40 до +85 ºС |
| Встроенный видеочип | нету | нету | нету |
| Размеры | 24 мм × 515 мм | 45 мм × 18 мм | 97 мм × 66 мм |

Для получения более подробной информации о рассмотренных микроконтроллерах использовались источники [ Ссылки на источики].

## Датчики освещенности

К наиболее популярным можно отнести такие модели датчиков освещенности, как датчик освещенности CJMCU-TEMT6000, цифровой датчик освещенности GY-302 и фоторезистивный датчик освещенности XD-80 на чипе **LM393**. В таблице 1.3 приведены их сравнительные характеристики.

Таблица 1.3 — Сравнение датчиков освещенности

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Параметры сравнения** | **Цифровой датчик освещенности GY-302** | **Фоторезистивный датчик**  **освещенности KY-018** | **Датчик освещенности CJMCU-TEMT6000** | |
| Угол чувствительности | 180 | 180 | 180 | |
| Напряжение питания | 3.3 – 5 В | 3.3 – 5 В | 3 – 5 В |
| Рабочая температура | от -5 до +70 ºС | от -40 до +85 ºС | от -25 до +85 ºС |
| Потребляемый ток | до 17 мА | до 130 мА | до 20 мкА |
| Измеряемые значения | 65536 градаций | 65536 градаций | 65536 градаций |
| Тип выходного канала | Цифровой, интерфейс I2C | аналоговый | аналоговый |

Для получения характеристик о данных датчиках использовалась источники [11, 12, 13 ссылки].

## Датчики температуры

Датчик температуры используется для определения температуры окружающей среды (см. таблицу 1.3). К популярным можно отнести датчик DHT22, DHT11, ds18b20, передающие данные по цифровому интерфейсу 1-WIRE.

Таблица 1.3 – Сравнение датчиков температуры

|  |  |  |  |
| --- | --- | --- | --- |
| **Параметр** | **DS18B20** | **DHT11** | **DHT22** |
| Напряжение питания | 3–5.5 В | 3-5 | 3-5 |
| Точность, % | 0,5 | 2 | 1 |
| Температурный диапазон | -55–+125 ºС | 0-50 ºС | -40–+80 ºС |
| Потребляемый ток, | 9 мА | <2.5 мА | <2.5 мА |
| Измерение влажности воздуха | нет | да | да |

Для получения характеристик о данных датчиках использовалась источники [11, 12, 13].

## Датчики влажности и атмосферного давления

Наиболее распространенными датчиками такого типа являются модели BME280, HTS221 и LPS25. При этом BME280 может измерять и давление, и влажность одновременно, HTS221 – только влажность, LPS25 – только давление. Все они измеряют температуру. Информация о них приведена в таблице 1.4

Таблица 1.4 — Сравнение датчиков

|  |  |  |  |
| --- | --- | --- | --- |
| **Параметры сравнения** | **BME280** | **HTS221** | **LPS25** |
| Напряжение питания | 3.3 В | 1.7– 3.6 В | 1.7 – 3.6 В |
| Потребляемый ток | до 2 мкА | до 1 мкА | до 2 мкА |
| Рабочая температура | от -40 до +85 ºС | от -40 до +85 ºС | от -15 до +70 ºС |
| Точность измерения | 0.01 гПа, 0.01 ºС, 3% | 1 °С, 0,1% | 1 гПа и 2°С |

Для получения точной информации о данных датчиках использовалась информация из источников [].

## Модули часов реального времени

Наиболее распространенными моделями часов являются DS1302, DS1307, DS3231. Они основаны на подключаемом модуле RTC (часы реального времени).

Сравнение приведено в таблице 1.5

Таблица 1.5 — Сравнение модулей RTC

|  |  |  |  |
| --- | --- | --- | --- |
| **Параметры сравнения** | **DS1307** | **DS1302** | **DS3231** |
| Частота | 1 Гц, 4.096 кГц, 8.192 кГц, 32.768 кГц | 32.768 кГц | Два выхода – первый на 32.768 кГц, второй – программируемый от 1 Гц до 8.192 кГц |
| Напряжение питания | 5 В | 5 В | 5 В |
| Потребляемый ток | до 14 мА | до 28 мА | до 6 мА |
| Рабочая температура | от -40 до +85 ºС | от -20 до +85 ºС | от -40 до +85 ºС |
| Протокол передачи | I2C | I2C, SPI | I2C |
| Точность | 2,5 секунды в сутки | 5 секунд в сутки | до 2 сек |

Для получения информации о данных модулях использовались источники [18, 19].

## Светодиоды для индикации

Наиболее популярными решениями адресных светодиодов являются решения на базе чипов WS2812, WS2813 и WS2811.

Результаты сравнения представлены в таблице 1.6.

Таблица 1.6 — Сравнение чипов адресных светодиодов

|  |  |  |  |
| --- | --- | --- | --- |
| **Параметры сравнения** | **WS2812** | **WS2813** | **WS2811** |
| Напряжение питания логики | 3.5 – 5.3 В | 3.5 – 5.3 В | 12 -24 В |
| Светодиодов на чип | 1 | 1 | 3 |
| Потребляемый ток  светодиодов | до 12 мА | до 15 мА | до 20 мА |
| Количество информационных входов | 1 | 2 (дублирующий) | 1 |

Для получения информации о данных светодиодах использовался источник [20, 21, 22].

## Устройства управления внешней нагрузкой

Классическим способом управления мощной внешней нагрузкой является использование механических реле. Сравнение модулей реле приведено в таблице 1.7.

Таблица 1.7 — Сравнение модулей реле

|  |  |  |
| --- | --- | --- |
| **Параметры сравнения** | **Relay Shield v3.0** | **Relay MOD 250-10** |
| Управляющее напряжение | 5 В | 5 В |
| Коммутация переменного напряжения | Нет | Да |
| Максимальный ток нагрузки | До 5А | До 10 А |
| Количество коммутируемых каналов | 4 | 4 |

Для получения информации о модулях реле использовался источник [23].

## Модуль информирования пользователя

Для информирования пользователя о состояниях датчиков были рассмотрены следующие модели экранов:

Результаты сравнения экранов приведены в таблице 1.8.

Таблица 1.8 — Сравнение экранов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Характеристика** | **LCD 1602** | **LCD 2004** | **OLED 12864** | **LCD 12864** |
| Технология | Жидкокристаллический | Жидкокристаллический | OLED | Жидкокристаллический |
| Разрешение | 16 символов  2 стоки | 20 символов  4 строки | 128x64 | 128x64 |
| Питание | 3.3 – 5В | 3.3 – 5В | 3.3 - 5В | 5В |
| Интерфейс подключения | Параллельный | I2C, параллельный | I2C | Параллельный, SPI |
| Подсветка | Да / нет | Да / нет | Да | Да |
| Регулировка яркости | Нет | Нет | Нет | Нет |
| Адресуемость отдельного пикселя | Нет | Да | Да | Да |
| Размер, мм | 69.5\*14.5 | 98 \* 60 | 28 \* 28 | 93 \* 70 |

Для получения информации о данных экранах использовались источники [24, 25].

# РАЗРАБОТКА СТРУКТУРЫ УСТРОЙСТВА

## 2.1 Постановка задачи

Перед тем как составить структуру разрабатываемого устройства, необходимо выделить функции, которые будет выполнять устройство.

Необходимо разработать микропроцессорное устройство контроля окружающей среды в рамках тепличного комбината. Функции, выполняемые данным устройством:

- контроль внешней температуры воздух;

- контроль влажности воздуха в помещении;

- контроль атмосферного давление;

- отслеживание уровня освещенности;

- автоматическое управление подключенной нагрузкой в зависимости от текущих значений датчиков

-  информирование пользователя о текущем состоянии окружающей среды.

- обеспечение возможности отправки получаемых значений по сети WiFi.

## 2.2 Определение компонентов структуры устройства

Компоненты структуры устройства выбираются исходя из функций, определенных в постановке задачи. Проанализировав выделенные функции, были определены следующие компоненты (см. приложение А), представленные ниже.

1)  Модуль взаимодействия с пользователем – экран, который отображает информацию об окружающей среде, а также поворотный энкодер, с помощью которого осуществляется взаимодействие с интерфейсом.

2) Модуль питания — стабилизатор напряжения и источник питания схемы.

3) Модуль RTC – модуль часов реального времени.

4) Модуль радиопередачи — передатчик, который передает информацию по Wifi.

5) Датчик влажности и атмосферного давление – датчик, считывающий информацию об указанных параметрах.

6) Датчик температуры – датчик, получающий температуру окружающей среды

7) Датчик освещенности — датчик, считывающий информацию о степени освещенности рабочего пространства устройства.

8) Модуль индикации — светодиод, выполняющий функцию индикации состояния устройства.

9) Модуль управления нагрузкой – контроль включения внешней сетевой нагрузки.

10) Микроконтроллер – анализирует информацию и вырабатывает управляющие сигналы.

2.3 Взаимодействие компонентов устройства

Микроконтроллер производит периодические запрос на считывание данных с датчиков, проверяет полученные данные на выход за пределы допустимых значений параметров.

Контроллер сравнивает допустимые значения и полученные значения с датчиков освещенности, температуры, влажности, давления и, при превышении, подает сигнал на модуль оповещения, который включает звуковой сигнал.

При обнаружении отклонений от нормы и/или сигнала с модуля управления микроконтроллер отдает сигнал модулю индикации для светового оповещения и, в зависимости от настроек, включает или отключает внешнюю нагрузку модуль управления нагрузкой.

Одновременно с этим микроконтроллер выводит собираемые данные на экран и получаем сигналы управления от энкодера, с которым взаимодействует пользователь.

Дополнительно микроконтроллер производит проверку входного буфера UART. При наличии в нём данных, выполняется обработка и сохранение либо отправка необходимых данных

В зависимости от режима, микроконтроллер может с заданной периодичностью сохранять текущие значения окружающей среды себе в флэш-память либо передавать их на радиопередатчик.

# ОБОСНОВАНИЕ ВЫБОРА УЗЛОВ, ЭЛЕМЕНТОВ ФУНКЦИОНАЛЬНОЙ СХЕМЫ УСТРОЙСТВА

## Обоснование выбора микроконтроллеров

Микроконтроллер является главным модулем разрабатываемого устройства, потому что берет на себя главные функции: управление всеми внешне подключёнными модулями, а также производить обработку получаемых с датчиков данных.

В таблице 1.1 были продемонстрированы ключевые параметры устройств. Микроконтроллеры STM32F407 и Esp-01 опережают Arduino Nano по многим параметрам, но было принято решение о достаточности параметров Arduino Nano для выполнения поставленных задач. Также данный микроконтроллер обладает большим количеством пользовательских библиотек для различных сценариев использования данного устройства.

## Обоснование выбора датчика освещенности

Рассмотрев сравнительные характеристики датчиков освещенности, представленных в таблице 1.2, было принято решение выбрать аналоговый датчик на основе фоторезистора. Данный датчик прост в своем устройстве и легкодоступен. Минусом его использования является отсутствие аппаратного гистерезиса.

## Обоснование выбора датчика температуры

В таблице 1.3 обзора литературы приведены сравнения наиболее распространенных датчиков температуры. Все они цифровые и работают по протоколу 1-Wire. Самым точным из них является ds18b20. Он также существует в выносном герметичном исполнении, что и обусловило выбор данного датчика.

## Обоснование выбора датчика влажности и атмосферного давления

После сравнения представленных датчиков видно, что BME280 – единственный, который может проводить измерения температуры, влажности и атмосферного давления одновременно. Также он достаточно точный и недорогой, что обусловило выбор в его пользу.

## Обоснование выбора часов реального времени

Сравнение модулей RTC, представленных в таблице 1.5 обзора литературы показало, что они все имеют схожие параметры. Поэтому модуль на основе DS3231 был выбран как наиболее точный и энергоэффективный.

## Обоснование выбора светодиода

Из представленных адресных светодиодов был выбран вариант на чипе WS2812, т.к они существуют в маленьких типоразмерах. Также причиной выбора в его пользу было то, что напряжение его питания согласуется с напряжением питания устройства.

## Обоснование выбора устройства управления внешней нагрузкой

В качестве устройства управления нагрузкой было выбран модуль недорогих механических реле, расчитанных на коммутацию достаточно маломощной нагрузки. Был выбран вариант в исполнении, включающем 2 блока реле.

## Обоснование выбора модуля информирования пользователя

После анализа альтернатив, в качестве экрана для использования была выбрана именно модель LCD2004, т.к она способна отображать 4 символьных строки по 20 символов. Вместе с достаточно большими размерами и низкой ценой, данный экран становится идеальным выбором для применения в проекте

Опционально пользователь может информироваться по сети Wifi. Для этого в проекте использован контроллер Esp-01, характеристики которого были рассмотрены в разделе 1. Плюсом его использования является дешевизна и наличие возможности подключения к Wifi сетям.

# РАЗРАБОТКА ПРИНЦИПИАЛЬНОЙ ЭЛЕКТРИЧЕСКОЙ СХЕМЫ УСТРОЙСТВА

## Расчёт мощности элементов схемы

Потребляемая мощность разрабатываемого устройства равна сумме мощностей, потребляемых его элементами. Расчет мощности элементов схемы устройства управления и самого устройства представлены в таблице 4.1 и 4.2 соответственно.

Таблица 4.1 – Расчет мощности элементов схемы устройства управления

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Блок | U, В | I, мА | Кол-во | P, мВт |
| Микроконтроллер Arduino UNO R3 | 5 | 22 | 1 | 110 |
| Модуль радиопередачи MX-05V | 5 | 28 | 1 | 140 |
| Джойстик | 5 | 8 | 2 | 80 |
| Светодиод | 5 | 20 | 1 | 100 |
| Суммарная мощность, мВт | | | | 430 |

В реализованной схеме используются микроконтроллер Arduino UNO R3, модуль радиопередачи MX-05V, 2 джойстика и светодиод.

Таким образом потребляемая мощность будет равна:

Р = 5 ∙ 22 + 5 ∙ 28 + 5 ∙ 8 ∙ 2 + 5 ∙ 20 = 430 мВт.

Учитывая поправочный коэффициент в 20%, максимальная потребляемая мощность составит 516 мВт.

Рассчитаем потребляемый ток:

Таблица 4.2 – Расчет мощности элементов схемы устройства

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Блок | U, В | I, мА | Кол-во | P, мВт |
| Микроконтроллер Arduino UNO R3 | 5 | 22 | 1 | 110 |
| Модуль радиоприема XD-RF-5V | 5 | 6 | 1 | 30 |
| Датчик горючих газов MQ-5 | 5 | 150 | 1 | 750 |
| Датчик освещенности LM393 | 5 | 15 | 1 | 75 |
| Датчик расстояния HC-SR04 | 5 | 15 | 1 | 75 |
| Драйвер моторов L298N | 5 | 36 | 1 | 180 |
| Мотор-редуктор 1:48 3-8V | 5 | 600 | 4 | 12 000 |

Продолжение таблицы 4.2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Пьезодинамик | 5 | 10 | 1 | 50 |
| Светодиод | 5 | 20 | 4 | 400 |
| Суммарная мощность, мВт | | | | 13670 |

В реализованной схеме используются микроконтроллер Arduino UNO R3, модуль радиоприема XD-RF-5V, датчик горючих газов MQ-5, датчик освещенности LM393, датчик расстояния HC-SR04, драйвер моторов L298N, 4 мотор-редуктора 1:48 3-8V, пьезодинамик и 4 светодиода.

Таким образом потребляемая мощность будет равна:

Р = 5 ∙ 22 + 5 ∙ 6 + 5 ∙ 150 + 5 ∙ 15 + 5 ∙ 15 + 5 ∙ 36 + 5 ∙ 600 ∙ 4 + 5 ∙ 10 + 5 ∙ 20 ∙ 4 = 13670 мВт.

Учитывая поправочный коэффициент в 20%, максимальная потребляемая мощность составит 16404 мВт.

Рассчитаем потребляемый ток:

## Расчёт нагрузки светодиодов

В данном курсовом проекте используется пять светодиодов различных цветов, подключенные к соответствующим пинам микроконтроллера (см. рисунок 4.2).



Рисунок 4.2 — Схема подключения светодиодов

Для ограничения тока светодиода используется резистор номиналом, рассчитываемым по следующей формуле:

где *U*П – напряжения питания, *U*Д – напряжение, падающее на светодиоде, *I*ПР – прямой ток светодиода.

В проекте используются светодиоды белого, желтого и зеленого цветов, со следующими параметрами: *I*ПР = 20 мА. *U*Д = 1 В.

Получаем:



Следовательно, для того, чтобы светодиод не перегорел он должен быть подключён через резистор с сопротивлением не менее 200 Ом, однако если взять слишком большое сопротивление, то светодиод будет гореть очень слабо, поэтому в данном проекте светодиоды подключаются через резисторы сопротивлением 220 Ом.

## 4.3 Микроконтроллеры

Информация о выбранном микроконтроллере Arduino UNO представлена в пункте 3.1 раздела 3.

Микроконтроллер соединен со всеми модулями схемы через аналоговые или цифровые входы и выходы.

В схеме с устройством управления к аналоговым входам A1 и A2 подключены джойстики, а на цифровой выход D12 подключен радиопередатчик.

В схеме передвижного устройства модуль освещенности подключен к аналоговому входу A1, датчик газов к аналоговому входу A0 и цифровому входу D2, датчик расстояния подключен к цифровым входам D9 и D8, радиоприемник к цифровому входу D11, пьезодинамик к цифровому выходу D10. Драйвер моторов подключен к Arduino через цифровые выходы D4 – D7, а светодиоды через цифровые выходы D12 и D13. Данный микроконтроллер на обоих схемах питается от напряжения 5 B.

## 4.4 Датчик горючих газов

Информация о выбранном датчике горючих газов MQ-5 представлена в пункте 3.2 раздела 3. Данный датчик подключен как к аналоговому входу микроконтроллера A0, так и к цифровому D2, это обусловлено тем, что через цифровой вход передается информация о присутствии газа, а через аналоговый о его концентрации. Питается датчик от напряжения 5 В.

## 4.5 Датчик освещенности

Информация о выбранном датчике освещенности LM393 представлена в пункте 3.3 раздела 3. Данный датчик в схеме питается от напряжения 5 В. К аналоговому входу A1 микроконтроллера подключен выход датчика A0, через который поступает информация о степени освещенности.

## 4.6 Ультразвуковой датчик расстояния

Информация о выбранном ультразвуковом датчике расстояния HC-SR04 представлена в пункте 3.4 раздела 3. Данный датчик подключается к контроллеру с помощью двух своих выходов Echo и Trig, в данной схеме они подключены к цифровым входам микроконтроллера D8 и D9. Питается датчик от напряжения 5 В.

## 4.7 Модуль радиопередачи

Информация о выбранных модулях радиопередачи представлена в пункте 3.5 раздела 3.

На устройстве управления используется радиопередатчик FS1000A/XD-FST, который питается от напряжения 5 В. Также на модуле есть вход Data, он подключен к цифровому выходу D12 микроконтроллера, через него осуществляется передача данных с Arduino на модуль.

На передвижном устройстве установлен радиоприемник XD-RF-5V, который также питается от напряжения 5 В. На данном модуле есть единственный выход Data, который подключен к цифровому входу D11 микроконтроллер. Через данный выход микроконтроллер получает информацию с радиоприемника.

Оба модуля подключены к антенне для улучшения радиосигнала.

## 4.8 Драйвер моторов

Информация о выбранном драйвере моторов L298N представлена в пункте 3.6 раздела 3. На вход VCC подается напряжение 5 В. Входы IN1, IN2, IN3, IN4 подключаются к микроконтроллеру, который выставляет направление вращения моторов. Драйвер анализирует полученные значения и запускает моторы через выходы OUT1, OUT2, OUT3 и OUT4.

## 4.9 Мотор-редукторы

Информация о выбранных мотор-редукторах представлена в пункте 3.7 раздела 3. Все они связаны с микроконтроллером через драйвер двигателей L298N. При поступлении питания на соответствующие входы моторов они начинают вращаться в заданную сторону.

## 4.10 Пьезодинамик

Для получения звукового сигнала используется пьезодинамик, который подключается к цифровому выходу Arduino D10. Микроконтроллер включает и отключает питание на данном динамике из-за чего включается и отключается звук.

## 4.11 Джойстики

В качестве управления в данном курсовом проекте используются два джойстика подключенных в схеме устройства управления к аналоговым входам Arduino A1 и A2. На вход VCC подается напряжение 5 В.

# РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

## Требования к разработке программного обеспечения

Разработанное микропроцессорное устройство собирает информацию о трех параметрах: концентрация горючих газов в среде, уровень освещенности и расстояние до ближайшего объекта.

Устройство работает следующим образом. При включении пульта дистанционного управления на нем загорается светодиод, который сигнализирует о том, что питание подключено. Когда пользователь меняет направление на джойстиках, микроконтроллер анализирует полученную информацию и отправляет код состояния на радиоприемник устройства через радиопередатчик, установленный на пульте. Затем полученная информация анализируется на устройстве и в зависимости от полученного кода состояния запускаются или приостанавливаются двигатели.

В течении всего времени устройство анализирует показания с датчиков. При получении информации о высокой концентрации горючих газов в среде включается пьезодинамик, который начинает издавать звук. Если в рабочей зоне недостаточно освещения, тогда включаются два светодиода, закрепленных на передней стороне устройства. Когда поступает информация о том, что перед устройством находится какой-то объект, то движение вперед останавливается и загораются светодиоды, установленные по бокам устройства. Если устройству не поступает никакая из вышеперечисленных информаций, то оно работает в обычном режиме без индикации.

## Блок-схема алгоритма

Блок-схема — это схематичное представление процесса, системы или компьютерного алгоритма. Блок-схемы часто применяются в разных сферах деятельности, чтобы документировать, изучать, планировать, совершенствовать и объяснять сложные процессы с помощью простых логичных диаграмм.

Рассмотрим блок-схему алгоритма программного обеспечения данного курсового проекта, представленную в приложении Г.

На первом листе приложения представлена блок схема алгоритма самого устройства. Блоки 2 – 3 представляют собой подготовку программы для дальнейшей работы (инициализация переменных и определение модулей, подключенных к микроконтроллеру). Ключевыми являются блоки 5 – 29, которые реализуют саму логику программы в бесконечном цикле. В блоках 5, 9, 15 происходит получение данных с датчиков освещенности, расстояния и газа соответственно. В блоках 6 – 8, 11 – 14, 16 – 18 данная информация анализируется и, при необходимости, включается соответствующая индикация. В 20 блоке проверяется наличие входящего сообщения от радиопередатчика, полученного в 19 блоке. Если данные присутствуют, то, в зависимости от полученного кода (блоки 21 – 29) происходит движение, заданное пользователем.

На втором листе приложения представлена блок схема пульта дистанционного управления устройством. Блоки 2 – 3 аналогичны блокам устройства. Блоки 5 – 6 реализуют получение информации о действиях пользователя, а блоки 7 – 15 анализируют полученные данные и выставляют соответствующий код на радиопередатчик. В блоке 16 информация отправляется устройству.

## Исходный код программы для устройства управления

Пульт дистанционного управления считывает показания с двухосевых джойстиков, а затем преобразует данные для быстрой и удобной отправки по радиоканалу. Исходный код программного обеспечение под данное устройство можно найти в приложении Д (строки 1 – 53).

Функция void setup() (строки 22 – 27) необходима для начальной настройки контроллера, здесь задаются входные и выходные пины, а так же другие настройки. В данном случае vw\_setup(2000) (строка 23) настраивает наш передатчик, а pinMode(pinX, INPUT) и pinMode(pinY, INPUT) (строки 25 и 26 соответственно) задают пины джойстика, как входные значения на плату.

Функция void loop() (строки 29 – 48) является главной, циклической функцией и работает на протяжении всей работы микроконтроллера. Здесь задается основная логика работы микроконтроллера. На начальном этапе работы функции обнуляются входные значения с джойстиков, а также очищается строка для отправки информации на радиоприёмник. Затем считываются новые данные с джойстиков, на основе полученных значений формируется новое сообщение, которое хранит номер операции для взаимодействия с устройством, подключенному к приёмнику. После этого сообщение конвертируется в массив char и передается на радиопередатчик для отправки.

Функция void send (char \*message) (строки 50 – 53) получает в качестве параметра массив символов, которые необходимо отправить на радиоприёмник. Полученный массив отправляется и затем происходит ожидание полной отправки сообщения.

## Исходный код программы для передвижного устройства

Данное устройство является основным в разрабатываемом курсовом проекте и здесь реализована большая часть логики. Исходный код программного обеспечение можно найти в приложении Д (строки 58 – 235).

Функция void setup() (строки 97 – 115) выполняет такие же задачи, как и в устройстве управления. В строках 98 – 99 настраивается радиоприемник, затем в строках 101 – 114 задаются входные и выходные пины устройства.

Функция void loop() (строки 117 – 124) является главной, здесь вызываются все методы для выполнения поставленных задач устройства. Она также работает на протяжении всей работы микроконтроллера.

Функция void checkIlumination() (строки 127 – 136) реализует логику взаимодействия с датчиком освещенности, здесь происходит считывание информации (строка 128) с последующей проверкой минимального допустимого значения. Строки 131 и 134 включают и выключают собственное освещение устройства соответственно.

Функция void checkDistanceToObject() (строки 138 – 159) реализует логику взаимодействия с датчиком расстояния. В строках 140 – 145 посылается два ультразвуковых сигнала равные 2 и 10 миллисекунд для получения информации о расстоянии до ближайшего объекта. В строке 149 полученное значение переводится в сантиметры, после чего (строки 151 – 158) сравнивается с минимально допустимым и, в зависимости от результата включаются и выключаются боковые светодиоды устройства (строки 153 и 157 соответственно), а также происходит остановка устройства, если значение довольно мало.

Функция void checkOfGas() (строки 161 – 171) реализует логику взаимодействия с датчиком горючих газов. Изначально происходит считывание информации с данного датчика (строки 162 – 163), а затем проверка допустимых значений (строки 165 – 169), если значение выше допустимого, то вызывается метод turnOnSound(), в ином случае метод turnOffSound() о которых будет сказано ниже.

Функция void turnOnSound() (строки 173 – 175) включает пьезодинамик, а функция void turnOffSound() (строки 177 – 179) его выключает.

Функция void selectStateOfMotors(char state) (строки 181 – 198) получает в качестве параметра состояние движения, которое отправил пользователь. В теле метода происходит сравнение со всеми возможными состояниями и, в зависимости от результатов сравнения, вызываются методы для работы с моторами.

Функции void mooveForward() (строки 200 – 205) , void mooveBack() (строки 207 – 212), void turnLeft() (строки 214 – 220) , void turnRight() (строки 222 – 228) и void stand() (строки 230 – 235) реализуют логику взаимодействия с моторами: движение прямо, назад, влево, вправо и бездействие соответственно.

# ЗАКЛЮЧЕНИЕ

В результате работы над данным курсовым проектом было разработано работоспособное микропроцессорное радиоуправляемое передвижное устройство со своим программным обеспечением. Устройство отслеживает показания концентрации горючих газов в среде, уровень освещенности и расстояние до ближайшего объекта. Помимо этого, осуществляется анализ полученных значений и соответствующая индикация показателей. Данный проект был спроектирован в соответствии с поставленными задачами, весь функционал был реализован в полном объеме.

Разработанное микропроцессорное устройство обладает следующими достоинствами: относительно низкая стоимость, простота реализации и сборки. Однако существенным недостатком является необходимость в написании собственного программного обеспечения для взаимодействия со всеми подключенными датчиками и анализа полученных данных.

В дальнейшем планируется усовершенствование данного курсового проекта. Одним из таких улучшений является оптимизация алгоритма анализа полученных данных, улучшение питания, а также создание более дружественного интерфейса.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

[1]. Вычислительные машины, системы и сети: дипломное проектирование (методическое пособие) [Электронный ресурс] : Минск БГУИР 2019. – Электронные данные. – Режим доступа: <https://www.bsuir.by/m/12_100229_1_136308.pdf>

[2]. Документация Arduino [Электронный ресурс]. – Электронные данные. – Режим доступа: <https://docs.arduino.cc/>

[3]. Геддес, М. 25 крутых проектов с Arduino / М. Геддес ; [пер. с англ. М. А. Райтмана]. ­– Москва : Эксмо, 2019. – 272 c.

[4]. Arduino UNO [Электронный ресурс]. – Электронные данные. – Режим доступа: http://arduino.ru/Hardware/ArduinoBoardUno – Дата доступа: 11.09.2021

[5]. Raspberry PI 2 Model B — второе поколение Raspberry Pi [Электронный ресурс]. – Электронные данные. – Режим доступа: https://micro-pi.ru/raspberry-pi-2-model-b-rpi-bcm2836-bcm2837/ – Дата доступа: 12.09.2021

[6]. OLIMEXINO-STN32 development board [Электронный ресурс]. – Электронные данные. – Режим доступа: https://data.electronshik.ru/pdf/pdf/o/olimexino-stm32.pdf – Дата доступа: 12.09.2021

[7]. Статья – Датчики [Электронный ресурс] – Электронные данные. – Режим доступа: <http://www.electrolibrary.info/subscribe/sub_16_datchiki.htm> – Дата доступа: 18.09.2021

[8]. MQ-5 [Электронный ресурс]. – Электронные данные. – Режим доступа: [https://files.seeedstudio.com/wiki/Grove-Gas\_Sensor-MQ5/res/MQ-5.pdf](https://files.seeedstudio.com/wiki/Grove-Gas_Sensor-MQ5/res/MQ-5.pdf%20) – Дата доступа: 22.09.2021

[9]. MQ-2 new [Электронный ресурс]. – Электронные данные. – Режим доступа: <https://www.pololu.com/file/0J309/MQ2.pdf> – Дата доступа: 22.09.2021

[10]. Microsoft Word - MQ-9 New [Электронный ресурс]. – Электронные данные. – Режим доступа: https://files.amperka.ru/datasheets/MQ-9.pdf – Дата доступа: 22.09.2021

[11]. Подключение фоторезистора к ардуино и работа с датчиком освещенности [Электронный ресурс]. – Электронные данные. – Режим доступа: https://arduinomaster.ru/datchiki-arduino/photorezistor-arduino-datchik-sveta/ – Дата доступа: 24.09.2021

[12]. TEMP6000 — аналоговый датчик освещенности [Электронный ресурс]. – Электронные данные. – Режим доступа: https://voltiq.ru/shop/temt6000-light-sensor/ – Дата доступа: 25.09.2021

[13]. GY-302 BH1750 модуль освещения, Light Sensor для Arduino [Электронный ресурс]. – Электронные данные. – Режим доступа: http://www.avrobot.ru/product\_info.php?products\_id=4033 – Дата доступа: 25.09.2021

[14]. Ultrasonic Sensor. HC-SR04 Datasheet [Электронный ресурс]. – Электронные данные. – Режим доступа: https://datasheetspdf.com/pdf/1380136/ETC/HC-SR04/1 – Дата доступа: 28.09.2021

[15]. Detection Sensor. DYP-ME007 Datasheet [Электронный ресурс]. – Электронные данные. – Режим доступа: https://datasheetspdf.com/pdf/1251016/ETC/DYP-ME007/1 – Дата доступа: 28.09.2021

[16]. Ультразвуковой датчик расстояния, температуры и освещенности [Электронный ресурс]. – Электронные данные. – Режим доступа: https://robot-kit.ru/3086/ – Дата доступа: 28.09.2021

[17]. NRF24L01 Datasheet (PDF) – List of Unclassified Manufacturers [Электронный ресурс]. – Электронные данные. – Режим доступа: <https://pdf1.alldatasheet.com/datasheet-pdf/view/1243924/ETC1/NRF24L01.html> – Дата доступа: 01.10.2021

[18]. Беспроводной передатчик на 433 МГц [Электронный ресурс]. – Электронные данные. – Режим доступа: https://3d-diy.ru/product/besprovodnoj-peredatchik-na-433-mgc – Дата доступа: 01.10.2021

[19]. Беспроводной приемник на 433 МГц [Электронный ресурс]. – Электронные данные. – Режим доступа: https://3d-diy.ru/product/besprovodnoj-priemnik-na-433-mgc – Дата доступа: 01.10.2021

[20]. Драйвер двигателя L298N [Электронный ресурс]. – Электронные данные. – Режим доступа: https://3d-diy.ru/wiki/arduino-moduli/drayver-dvigatelya-l298n/ – Дата доступа: 02.10.2021

[21]. Плата расширения L298P Motor Shield (для Arduino) [Электронный ресурс]. – Электронные данные. – Режим доступа: https://radioprog.ru/shop/merch/41 – Дата доступа: 02.10.2021

[22]. Обзор драйвера мотора MX1508 [Электронный ресурс]. – Электронные данные. – Режим доступа: https://robotchip.ru/obzor-drayvera-motora-mx1508/ – Дата доступа: 02.10.2021

[23]. Мотор постоянного тока с редуктором 1:48 [Электронный ресурс]. – Электронные данные. – Режим доступа: https://3d-diy.ru/wiki/arduino-mechanics/motor-postoyannogo-toka-reduktorom-1-48/ – Дата доступа: 03.10.2021

[24]. Пищалка – пьезодинамик Arduino [Электронный ресурс]. – Электронные данные. – Режим доступа: https://arduinomaster.ru/uroki-arduino/pishhalka-pezodinamik-arduino/ – Дата доступа: 03.10.2021

[25]. Microsoft Word – MP3 – TF – 16P V1.0 [Электронный ресурс]. – Электронные данные. – Режим доступа: https://picaxe.com/docs/spe033.pdf – Дата доступа: 03.10.2021

# ПРИЛОЖЕНИЕ А

(обязательное)

**Схема структурная**

# ПРИЛОЖЕНИЕ Б

(обязательное)

**Схема функциональная**

# ПРИЛОЖЕНИЕ В

(обязательное)

**Схема принципиальная**

# ПРИЛОЖЕНИЕ Г

(обязательное)

**Схема программы**

# ПРИЛОЖЕНИЕ Д

(обязательное)

**Листинг кода**

001. /\*

002. Скетч для передатчика

003. Подключите передатчик к 12 контакту Arduino

004. \*/

005.

006. #include <VirtualWire.h>

007.

008. #define pinX A2 // ось X джойстика

009. #define pinY A1 // ось Y джойстика

010.

011. const int MAX\_AXIS\_VALUE = 700;

012. const int MIN\_AXIS\_VALUE = 200;

013.

014. char FORWARD = '1';

015. char BACK = '2';

016. char RIGHT = '3';

017. char LEFT = '4';

018. char STOP = '5'

019.

020. char outputMessage[1];

021.

022. void setup() {

023. vw\_setup(2000);

024.

025. pinMode(pinX, INPUT);

026. pinMode(pinY, INPUT);

027. }

028.

029. void loop() {

030. int X = analogRead(pinX);

031. int Y = analogRead(pinY);

032. String output;

033.

034. if (X > MAX\_AXIS\_VALUE) {

035. output = FORWARD;

036. } else if (X < MIN\_AXIS\_VALUE) {

037. output = BACK;

038. } else if (Y > MAX\_AXIS\_VALUE) {

039. output = RIGHT;

040. } else if (Y < MIN\_AXIS\_VALUE) {

041. output = LEFT;

042. } else {

043. output = STOP;

044. }

045.

046. output.toCharArray(outputMessage, 2);

047. send(outputMessage);

048. }

049.

050. void send (char \*message) {

051. vw\_send((uint8\_t \*)message, strlen(message)); //отправка сообщения

052. vw\_wait\_tx(); // Ожидание полной отправки сообщения

053. }

054.

055.

056.

057. /\*

058. Скетч для приемника

059. Подключите приемник к 11 контакту Arduino

060. \*/

061.

062. #include <VirtualWire.h>

063. #include <NewPing.h>

064.

065. #define motor\_IN1 4

066. #define motor\_IN2 5

067. #define motor\_IN3 6

068. #define motor\_IN4 7

069.

070. #define headlights 13

071. #define side\_lights 12

072.

073. #define light\_sensor\_D0 A1

074.

075. #define trigPin 8

076. #define echoPin 9

077.

078. #define MQ5\_digitalSignal 2

079. #define MQ5\_analogSignal A0

080. #define soundPin 10

081.

082. byte message[VW\_MAX\_MESSAGE\_LEN]; // Массив входящих сообщений

083. byte messageLength = VW\_MAX\_MESSAGE\_LEN; // Размер массива входящих сообщений

084.

085. int stops;

086. boolean isGas; //переменная для хранения значения о присутствии газа

087. int gasValue = 0; //переменная для хранения количества газа

088.

089. int PERMISSIBLE\_GAS\_VALUE = 300;

090. int MIN\_DISTANCE\_BEFORE\_STOP = 12;

091.

092. char BACK = '1';

093. char FORWARD = '2';

094. char LEFT = '3';

095. char RIGHT = '4';

096.

097. void setup() {

098. vw\_setup(2000);

099. vw\_rx\_start(); // Активировать процесс приемника.

100.

101. pinMode(motor\_IN1, OUTPUT);

102. pinMode(motor\_IN2, OUTPUT);

103. pinMode(motor\_IN3, OUTPUT);

104. pinMode(motor\_IN4, OUTPUT);

105.

106. pinMode (light\_sensor\_D0, INPUT);

107. pinMode(headlights, OUTPUT);

108.

109. pinMode(trigPin, OUTPUT);

110. pinMode(echoPin, INPUT);

111. pinMode(side\_lights, OUTPUT);

112.

113. pinMode(MQ5\_digitalSignal, INPUT);

114. pinMode(soundPin, OUTPUT);

115. }

116.

117. void loop() {

118. checkIllumination();

119. checkDistanceToObject();

120. checkOfGas();

121.

122. if (vw\_get\_message(message, &messageLength)) {

123. selectStateOfMotors(message[0]);

124. }

125. }

126.

127. void checkIllumination() {

128. int xD0 = digitalRead (light\_sensor\_D0);

129.

130. if (xD0 == HIGH) {

131. digitalWrite (headlights, HIGH);

132. }

133. else {

134. digitalWrite (headlights, LOW);

135. }

136. }

137.

138. void checkDistanceToObject() {

139. int duration, cm;

140. digitalWrite(trigPin, LOW);

141. delayMicroseconds(2);

142.

143. digitalWrite(trigPin, HIGH);

144. delayMicroseconds(10);

145. digitalWrite(trigPin, LOW);

146.

147. duration = pulseIn(echoPin, HIGH);

148.

149. cm = duration / 58; // вычисляем расстояние в сантиметрах

150.

151. if (cm < MIN\_DISTANCE\_BEFORE\_STOP && cm > 0) {

152. stops = 1;

153. digitalWrite (side\_lights, HIGH);

154.

155. } else {

156. stops = 0;

157. digitalWrite (side\_lights, LOW);

158. }

159. }

160.

161. void checkOfGas() {

162. isGas = digitalRead(MQ5\_digitalSignal); //считываем значение о присутствии газа

163. gasValue = analogRead(MQ5\_analogSignal); // и о его количестве

164.

165. if (gasValue < PERMISSIBLE\_GAS\_VALUE) {

166. turnOffSound();

167. }

168. else {

169. turnOnSound();

170. }

171. }

172.

173. void turnOnSound() {

174. digitalWrite(soundPin, 50); // включаем пьезодинамик

175. }

176.

177. void turnOffSound() {

178. digitalWrite(soundPin, 0); // отключаем пьезодинамик

179. }

180.

181. void selectStateOfMotors(char state) {

182.

183. if (state == BACK) {

184. mooveBack();

185.

186. } else if (state == FORWARD && stops == 0) {

187. mooveForward();

188.

189. } else if (state == LEFT) {

190. turnLeft();

191.

192. } else if (state == RIGHT) {

193. turnRight();

194.

195. } else {

196. stand();

197. }

198. }

199.

200. void mooveForward() {

201. digitalWrite(motor\_IN1, LOW);

202. digitalWrite(motor\_IN2, HIGH);

203. digitalWrite(motor\_IN3, LOW);

204. digitalWrite(motor\_IN4, HIGH);

205. }

206.

207. void mooveBack() {

208. digitalWrite(motor\_IN1, HIGH);

209. digitalWrite(motor\_IN2, LOW);

210. digitalWrite(motor\_IN3, HIGH);

211. digitalWrite(motor\_IN4, LOW);

212. }

213.

214. void turnLeft() {

215. digitalWrite(motor\_IN1, LOW);

216. digitalWrite(motor\_IN2, HIGH);

217.

218. digitalWrite(motor\_IN3, HIGH);

219. digitalWrite(motor\_IN4, LOW);

220. }

221.

222. void turnRight() {

223. digitalWrite(motor\_IN1, HIGH);

224. digitalWrite(motor\_IN2, LOW);

225.

226. digitalWrite(motor\_IN3, LOW);

227. digitalWrite(motor\_IN4, HIGH);

228. }

229.

230. void stand() {

231. digitalWrite(motor\_IN1, LOW);

232. digitalWrite(motor\_IN2, LOW);

233. digitalWrite(motor\_IN3, LOW);

234. digitalWrite(motor\_IN4, LOW);

235. }

# ПРИЛОЖЕНИЕ Е

(обязательное)

**Перечень элементов**

# ПРИЛОЖЕНИЕ Ж

(обязательное)

**Ведомость документов**