Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей
Кафедра электронных вычислительных машин
Дисциплина: Структурная и функциональная организация ЭВМ

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к курсовому проекту
на тему
РЕАЛИЗАЦИЯ МОДЕЛИ АВТОМОБИЛЯ, УПРАВЛЯЕМОЙ ПРИ ПОМОЩИ МОБИЛЬНОГО ПРИЛОЖЕНИЯ

БГУИР КП 1-40 02 01 401 ПЗ

Студент: М. Ю. Авдеенко

Руководитель: И. С. Тарасюк

МИНСК 2023

Учреждение образования
«Белорусский государственный университет информатики
и радиоэлектроники»

Факультет компьютерных систем и сетей

УТВЕРЖДАЮ
Заведующий кафедрой

_____
(подпись)
_____2023г.

ЗАДАНИЕ
по курсовой работе

Студенту *Авдеенко Максиму Юрьевичу*

1. Тема проекта *Реализация модели автомобиля, управляемой при помощи мобильного приложения*

2. Срок сдачи студентом законченного проекта *20 мая 2023 г*

3. Исходные данные к проекту *Микроконтроллер Arduino Nano, Беспроводной Bluetooth-приемопередатчик HC-06, модуль привода генератора MX-150, Электродвигатель постоянного тока с пластиковым колесом.*

4. Содержание расчетно-пояснительной записки (перечень вопросов, которые подлежат разработке) *Введение. 1. Обзор литературы. 2. Разработка структурной схемы. 3. Обоснование функциональной схемы. 4. Разработка принципиальной схемы. 5. Разработка програмного обеспечения. Заключение. Список использованных источников. Приложения.*

5. Перечень графического материала (с точным обозначением обязательных чертежей и графиков) *1. Схема электрическая структурная. 2. Схема электрическая функциональная. 3. Схема электрическая принципиальная.*

6. Консультант по проекту *Тарасюк И. С.*

7. Дата выдачи задания *3 февраля 2023 г.*

8. Календарный график работы над проектом на весь период проектирования (с обозначением сроков выполнения и трудоемкости отдельных этапов):

разделы 1,2 к 1 марта 2023 г. – 20 %;

раздел 3,4 к 15 марта 2023 г. – 50 %;

раздел 5 к 1 апреля 2023 г. – 80 %;

оформление пояснительной записки и графического материала к 29 апреля 2023 г – 100 %

Защита курсового проекта с 4 мая 2023 г. по 15 мая 2023 г

РУКОВОДИТЕЛЬ _____ И. С. Тарасюк
(подпись)

Задание принял к исполнению _____ М. Ю. Авдеенко
(дата и подпись студента)

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Современные технологии позволяют нам контролировать различные устройства и системы с помощью мобильных устройств. Одним из примеров является машина на Bluetooth управлении, которая может быть управляема с помощью смартфона или планшета. Однако, чтобы понять, как это работает, необходимо иметь представление о структурной и функциональной организации электронной вычислительной машины.

ЭВМ состоит из множества компонентов, включая процессор, память, устройства ввода и вывода, а также различные интерфейсы и контроллеры. Каждый компонент выполняет определенную функцию, и все они работают вместе для обработки данных и выполнения задач.

Для управления машиной на Bluetooth управлении необходимо иметь устройство, которое будет выполнять функции контроллера. Это может быть, как сам смартфон или планшет, так и специальное устройство, связанное с машиной. Когда пользователь управляет машиной через приложение на смартфоне, данные передаются через Bluetooth-интерфейс, который подключен к контроллеру машины. Контроллер интерпретирует команды и управляет ее движением.

Современные машины на Bluetooth управлении часто имеют множество функций, таких как управление скоростью, направлением движения, освещением и даже возможностью записывать видео. Эти устройства также часто имеют различные датчики, такие как гироскопы и акселерометры, которые могут помочь в управлении и стабилизации аппарата.

Кроме того, машины на Bluetooth управлении могут использоваться в различных областях, включая развлечения, образование и даже промышленность. Они могут быть использованы для обучения программированию и робототехнике, а также для тестирования и отладки систем управления на предприятиях.

В целом, понимание структурной и функциональной организации ЭВМ может помочь лучше понять, как устройства на Bluetooth управлении работают и как они могут быть улучшены и оптимизированы в будущем.

1 ОБЗОР ЛИТЕРАТУРЫ

1.1 Микроконтроллер

Для реализации данного курсового проекта, необходимо выбрать микроконтроллер, для возможности получения, обработки значений, а также вывод сигналов на устройство управления через Bluetooth-модуль. На данный момент существует две крупнейшие фирмы разрабатывающие микроконтроллеры, доступные для покупки и сборки небольших устройств: Raspberry и Arduino.

Arduino является очень распространенной среди начинающих пользователей. Такую распространенность они получили в связи с простотой освоения и большим количеством учебного материала. Данные платы часто используются для создания простых схем различного назначения. В данном курсовом проекте в качестве микроконтроллера была выбрана Arduino Nano (рис. 1.1). Платформа Nano, построенная на микроконтроллере ATmega328 (Arduino Nano 3.0) или ATmega168 (Arduino Nano 2.x), имеет небольшие размеры и может использоваться в лабораторных работах. Она имеет схожую с Arduino Duemilanove функциональность, однако отличается сборкой. Отличие заключается в отсутствии силового разъема постоянного тока и работе через кабель Mini-B USB. Nano разработана и продается компанией Gravitech.

Язык программирования Arduino называется Arduino C и представляет собой язык C++ с фреймворком Wiring, он имеет некоторые отличия по части написания кода, который компилируется и собирается с помощью avr-gcc, с особенностями, облегчающими написание работающей программы — имеется набор библиотек, включающий в себя функции и объекты. При компиляции программы IDE создает временный файл с расширением *.cpp.

Arduino Nano может получать питание через подключение Mini-B USB, или от нерегулируемого 6-20 В (вывод 30), или регулируемого 5 В (вывод 27), внешнего источника питания. Автоматически выбирается источник с самым высоким напряжением.

На платформе Arduino Nano установлено несколько устройств для осуществления связи с компьютером, другими устройствами Arduino или микроконтроллерами. ATmega168 и ATmega328 поддерживают последовательный интерфейс UART TTL (5 В), осуществляемый выводами 0 (RX) и 1 (TX). Установленная на плате микросхема FTDI FT232RL направляет данный интерфейс через USB, а драйверы FTDI (включены в программу Arduino) предоставляют виртуальный COM порт программе на компьютере. Мониторинг последовательной шины (Serial Monitor) программы Arduino позволяет посылать и получать текстовые данные при подключении к платформе. Светодиоды RX и TX на платформе будут мигать при передаче данных через микросхему FTDI или USB подключение (но не при использовании последовательной передачи через выводы 0 и 1) [1].

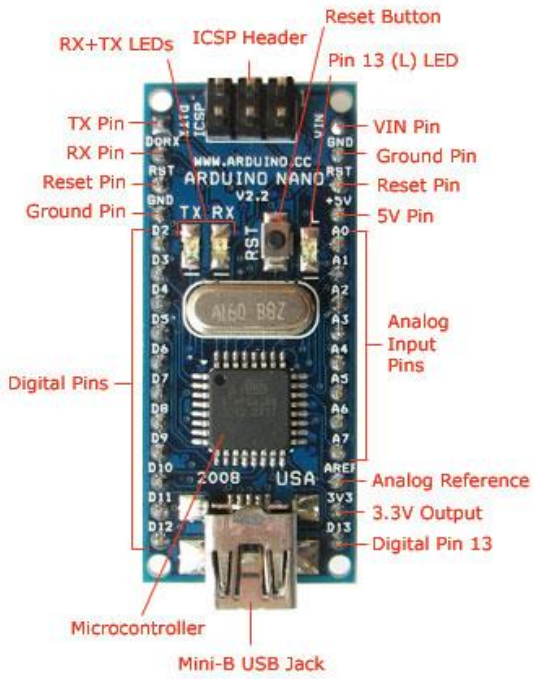

Рисунок 1.1 – Микроконтроллер Arduino Nano

1.2 Bluetooth-приемопередатчик

На данный момент на рынке присутствует множество различных Bluetooth приемопередатчиков. Некоторые имеют более высокую цену, некоторые более низкую. Среди огромного разнообразия необходимо выбрать тот, который будет наилучшим образом подходить для данного устройства. Рассмотрим несколько из них.

Начнем с BK3231 SPP-C.

Bluetooth-модуль SPP-C предназначен для интеллектуальной беспроводной передачи данных и создания следующих версий V2.1+. Спецификация EDR Bluetooth. Модуль поддерживает интерфейс UART и поддерживает последовательный протокол Bluetooth SPP, низкая стоимость, небольшой размер, низкое энергопотребление, преимущества чувствительности отправки и получения, всего несколько внешних компонентов смогут реализовать свои мощные функции.

Этот модуль в основном используется для беспроводной передачи данных в полевых условиях на короткие расстояния. И ПК может быть легко подключаемое устройство Bluetooth, оно также может обмениваться данными между двумя модулями. Избегать громоздкие кабельные соединения, могут напрямую заменить последовательный кабель [2].

Теперь рассмотрим Bluetooth BT-06.

Управление Bluetooth-модулем осуществляется с устройства master Bluetooth (телефон, PDA, ноутбук и т.п.). Данные передаются по стандарту UART, который вшит в большинстве встроенных систем (этот порт имеется почти во всех микроконтроллерах или легко организуется программно). Также

возможно управление с помощью АТ-команды. Bluetooth-модуль с антенной имеет один 6 пиновый интерфейс.

Питание Bluetooth-модуля SPP-C осуществляется либо от микроконтроллера, либо от внешнего источника питания (блока питания, батареи). Напряжение питания модуля 3,3 В ток потребления 30...40 мА. Модуль установлен на подложку, на которой расположен стабилизатор напряжения с 5 на 3,3 В. Поэтому можно подсоединять напряжение питания 3,6 – 6 В от внешнего источника или 5 В от Arduino. Линии Bluetooth-модуля SPP-C могут работать и с TTL логикой (5 В), что позволяет подсоединять его UART к Arduino [3].

Теперь рассмотрим Bluetooth-приемопередатчик HC-06.

Модуль связи Bluetooth V2.0 + EDR для применения в приборах на основе микроконтроллеров и прочих устройствах, оснащенных интерфейсом UART или RS232. Комплект поставки состоит из модуля HC-06 Bluetooth, смонтированного на плате, имеющей 4 штыревых контакта и кабеля длиной 20 см. Применение модуля позволяет связать двунаправленным мостом UART–Bluetooth–UART систему, изготовленную из компонентов Arduino, устройства на базе микроконтроллера и второй прибор, имеющий Bluetooth master в стандарте SPP. Модуль работает в режиме slave. Работа в режиме master возможна только при перезаписи внутренней основной программы, при этом наименование HC-06 теряет смысл. Организация связи происходит по двум схемам. Контроллер-контроллер – для связи двух приборов на основе микроконтроллеров или компьютер-контроллер – используя специально написанную программу и размещенную на андроиде, ноутбуке или персональном компьютере. При использовании модуля инженеру не требуется вникать в особенности Bluetooth.

При физическом соединении модуля с контроллером или ПК обмен информацией между ними происходит через проводной интерфейс UART. Контроллер воспринимает модуль как ответное устройство UART, а за организацию связи по радио отвечает электроника модуля. Также “на борту” модуля интерфейс SPI.

Конструктивно представляет собой две небольшие платы одна поверх другой. Размещены: миниатюрная антенна из дорожки на верхнем слое печатной платы в виде змейки, микросхема BC417 серии BlueCore4-Ext фирмы Cambridge Silicon Radio, микросхема флэш-памяти ES29LV800DB-70WGI производства Excel Semiconductor объемом 8 Мбит (1 МБ), хранящая программу и настройки. Микросхема BC417 поддерживает спецификацию Bluetooth V2.0 и АТ-команды, может работать в режиме Master или Slave с различной скоростью обмена данными. На более крупной плате расположен стабилизатор напряжения 3,3 В. Интерфейс UART и линии питания выведены на четыре штыревых контакта. Предназначен для бытового и коммерческого применения [4]. Компонент представлен на рисунке 1.2.

Рисунок 1.2 – Bluetooth-приемопередатчик HC-06

1.3 Драйверы моторов

Драйвера моторов являются неотъемлемой частью при проектировании передвижных устройств. Благодаря драйверам регулируется питание, поступающее на моторы, скорость и направление их вращения.

Для сравнения были выбраны модули L298N, L298P и более простой и новый модуль MX1508, спроектированный на базе модуля L298N. Результаты сравнения представлены в таблице 1.1.

Таблица 1.1 – Сравнение драйверов моторов

Параметры сравнения	L298N	L298P	MX-1508
Напряжение питания логики	5В	5В	5В
Потребляемый ток логики	До 36 мА	До 36 мА	До 36 мА
Напряжение питания приводов	5 – 35 В	5 – 25 В	2 – 10 В
Потребляемый ток приводов	До 3 А	До 2 А	До 2 А
Рабочая температура	от -25 до +135 ºС	от -25 до +130 ºС	от -40 до +80 ºС
Размер	43.5 × 43.2 × 29.4 мм	58 × 56 × 19 мм	24.7 × 21 × 5 мм

Для получения информации о данных драйверах использовалась источники [5, 6, 7].

1.4 Электродвигатель постоянного тока с пластиковым колесиком

Существует множество видов мотор-редукторов, которые отличаются по форме и характеристикам. По форме они разделяются на 4 типа: прямой одноосевой, прямой двухосевой, угловой одноосевой и угловой двухосевой.

Для сборки проекта будут использоваться прямые двухосевые мотор-редукторы с передаточным числом 1:48, которых на рынке представлено только два. Их сравнительные характеристики приведены в таблице 1.2.

Таблица 1.2 – Сравнение двухосевых моторов-редукторов

Параметры сравнения	Мотор-редуктор 1:48 3-8V	Мотор-редуктор 1:48 6V
Напряжение питания	3 – 8 В	6 В
Потребляемый ток	до 700 мА	до 670 мА
Передаточное число	1:48	1:48
Скорость вращения без нагрузки	до 200 об/мин	до 180 об/мин
Крутящий момент	2 кг/см	2 кг/см

Для получения информации о драйверах использовался источник [8].

Исходя из вышеперечисленного анализа, были выбраны моторы, которые имеют средние характеристики между двумя описанными. Их детальные характеристики представлены ниже.

Номинальное напряжение: 3 ~ 6В. Непрерывный ток без нагрузки: 150 мА +/- 10%. Мин. рабочая скорость (3 В): 90 +/- 10% об/мин. Мин. рабочая скорость (6 В): 200 +/- 10% об/мин. Крутящий момент: 0,15 нм ~ 0,60 нм. Крутящий момент, крутящий момент (6V): 0,8 kg. Передаточное отношение: 1:48. Размеры корпуса: 70x22x18 мм. Длина провода: 200 мм 28 AWG. Вес продукта: 29 г. Характеристики колеса: диаметр колеса: 65 мм, толщина колеса: 28 мм.

2 РАЗРАБОТКА СТРУКТУРНОЙ СХЕМЫ

2.1 Постановка задачи

Для того, чтобы составить структуру разрабатываемого устройства, необходимо выделить функции, которые будет выполнять устройство, затем определить компоненты и связь между ними исходя из данных функций. Результаты можно посмотреть на структурной схеме, представленной в приложении А.

В рамках данного курсового проекта необходимо разработать передвижное устройство, управляемое через Bluetooth. Исходя из этого, были выделены следующие функции, которые должно выполнять данное устройство:

– передвижение по суши и изменение направления движения путем регулирования тяговых усилений колес;

– управление с помощью мобильного телефона посредством Bluetooth-сигнала.

2.2 Определение компонентов структуры устройства

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

1) Микроконтроллер — ключевой компонент всей схемы. Выполняет функцию обработки поступающей информации и выдает управляющие сигналы.

2) Блок питания — источник питания схемы.

3) Модуль Bluetooth-приема — приемник, который получает информацию с мобильного телефона.

4) Модуль передвижения — моторы-редукторы и драйвер моторов для данных редукторов, которые реализуют передвижение по суше.

5) Модуль управления — мобильная программа, управляющая направлением и движением устройства.

2.3 Взаимодействие компонентов устройства

Устройство последовательно считывает информацию с Bluetooth-модуля, затем эти данные передаются на контроллер, который их анализирует.

Пользователь дистанционной взаимодействует с устройством посредством мобильного телефона или любого другого устройства, имеющего Bluetooth соединение и программу для управления. При перемещении соответствующих джойстиков в сторону, в которую будет осуществляться движение, телефон считывает выставленное направление и передает информацию на приемник. После чего контроллер, установленный на

устройстве, анализирует полученную информацию и, при определенных значениях, отправляет сигнал драйверу моторов, который запускает вращение двигателей в нужную сторону.

Модуль питания взаимодействует со всеми элементами схемы напрямую или через контроллер, благодаря ему осуществляется питание всех необходимых элементов.

3 ОБОСНОВАНИЕ ФУНКЦИОНАЛЬНОЙ СХЕМЫ

Это основной раздел пояснительной записки, дающий ключ к пониманию работы проектируемого устройства, а также информацию об обработке цифровых и аналоговых сигналов согласно назначению устройства.

Функциональная схема представлена в приложении Б.

3.1 Обоснование выбора микроконтроллера

В качестве управляющей платформы для данного проекта используется платформа Arduino Nano, построенная на микроконтроллере ATmega328. Данные платы часто используются для создания простых схем различного назначения. Краткие характеристики приведены в таблице 3.1.

Таблица 3.1 – Краткие характеристики Arduino Nano.

Микроконтроллер	Atmel ATmega328
Рабочее напряжение (логический уровень)	5 В
Входное напряжение (рекомендуемое)	7 – 12 В
Входное напряжение (предельное)	6 – 20 В
Цифровые входы/выходы	14
Аналоговые входы	8
Постоянный ток через вход/выход	40 мА
Флеш-память	32 Кб (2 кБ используется для загрузчика)
ОЗУ	2 кБ
EEPROM	1 кБ
Тактовая частота	16 МГц
Размеры	1,85 см x 4,2 см

Плата Arduino Nano может получать питание через подключение Mini-B USB, или от нерегулируемого 6 – 20 В (вывод 30), или регулируемого 5 В (вывод 27), внешнего источника питания. Автоматически выбирается источник с самым высоким напряжением.

Микроконтроллер ATmega328 имеет 32 кБ флеш-памяти для хранения кода программы, 2 кБ из которых используются для хранения загрузчика. ATmega328 имеет 2 кБ ОЗУ и 1 кБ EEPROM.

Входы и выходы платы Arduino Nano приведены на рис. 3.1. Каждый из 14 цифровых выводов Nano, используя функции pinMode(), digitalWrite(), digitalRead(), может настраиваться как вход или выход. Выводы работают при напряжении 5 В. Каждый вывод имеет нагрузочный резистор (стандартно отключен) 20 – 50 кОм и может пропускать до 40 мА.

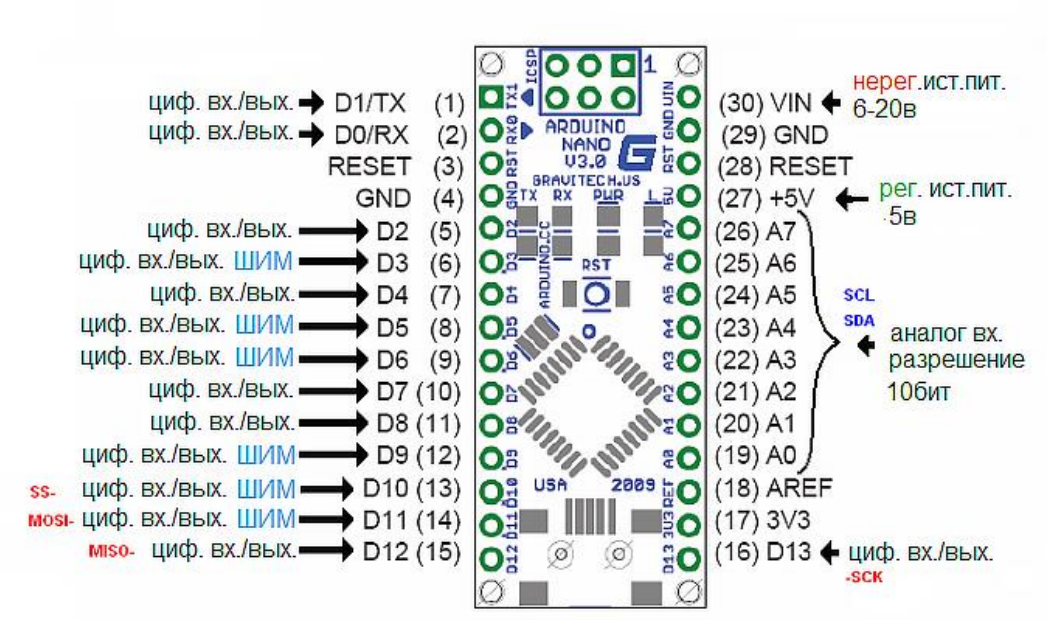

Рисунок 3.1 – Входы и выходы Arduino Nano

На платформе Arduino Nano установлено несколько устройств для осуществления связи с компьютером, другими устройствами Arduino или микроконтроллерами. ATmega328 поддерживают последовательный интерфейс UART TTL (5 В), осуществляемый выводами 0 (RX) и 1 (TX). Установленная на плате микросхема FTDI FT232RL направляет данный интерфейс через USB, а драйверы FTDI (включены в программу Arduino) предоставляют виртуальный COM-порт программе на компьютере. Также ATmega328 поддерживают интерфейсы I2C (TWI) и SPI. Для удобства использования I2C в Arduino включена библиотека Wire.

Платформа программируется посредством ПО Arduino или с помощью расширения Visual Micro для Microsoft Visual Studio. Микроконтроллер ATmega328 поставляется с записанным загрузчиком, облегчающим запись новых программ без использования внешних программаторов. Связь осуществляется оригинальным протоколом STK500. Также есть возможность не использовать загрузчик и запрограммировать микроконтроллер через выводы блока ICSP (внутрисхемное программирование).

3.2 Обоснование выбора модулей приема

После анализа модулей приема, был сделан вывод, что модуль HC-06 наилучшим образом подходит для данного устройства. Плюсами данного устройства являются: отсутствие необходимости вникать в полное устройство технологии Bluetooth, большая дальность связи (до 20 метров), а также высокая скорость приема и передачи данных. Данное устройство поддерживает двунаправленную передачу данных, однако в данном курсовом проекте это не будет использоваться, так как передача всегда будет

происходить с мобильного устройства на приемник. Невысокая стоимость также является преимуществом данного устройства.

3.3 Обоснование выбора драйвера моторов

В таблице 1.1 обзора литературы приведены сравнения драйверов моторов. В данном курсовом проекте будет использована модель MX-1508.

Основной чип модуля это микросхема MX1508, состоящая из двух H-мост (H-Bridge), один для выхода A, второй для выхода B, каждый канал рассчитан на 1,5 A с пиком 2,5 A. H-мост широко используется в электронике и служит для изменения вращения двигателем, схема H-моста содержит четыре транзистора (ключа) с двигателем в центре, образуя H-подобную компоновку. Принцип работы прост, при одновременном закрытие двух отдельных транзистора изменяется полярность напряжения, приложенного к двигателю. Это позволяет изменять направление вращения двигателя.

Данный модуль привода двигателя очень подходит для умных автомобилей с батарейным питанием, игрушечных автомобилей, роботов и т.д. Напряжение питания 2В~10В, может управлять двумя двигателями постоянного тока или двухфазным шаговым двигателем A 4-line, может реализовать функцию положительное и отрицательное вращение и регулировка скорости, каждый ток может достигать 1.5A непрерывного тока, пиковый ток может достигать 2.5 A, тепловая защита и автоматическое восстановление. Данный компонент представлен на рисунке 3.2.

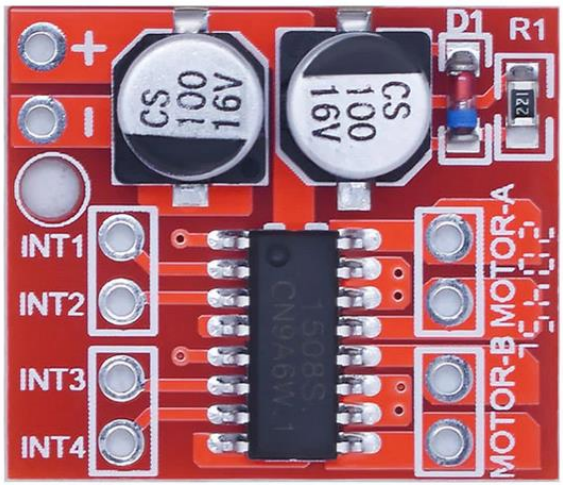

Рисунок 3.2 – Модуль привода генератора MX-1508

3.4 Обоснование выбора мотор-редукторов

В данном устройстве будут использованы прямые двухосевые мотор-редукторы с передаточным числом 1:48. Полные характеристики данных компонентов приведены в пункте 1.4 обзора литературы. Данные мотор-редукторы 1:48 3-6 В наилучшим образом подходят, так как имеют

подходящий диапазон напряжения питания и высокую скорость вращения. Компонент представлен на рисунке 1.4.

Рисунок 1.4 – Электродвигатель постоянного тока с пластиковым колесиком

4 РАЗРАБОТКА ПРИНЦИПИАЛЬНОЙ СХЕМЫ

Принципиальная схема является наиболее полной электрической схемой изделия, на которой изображают все электрические элементы и устройства, необходимые для осуществления и контроля в изделии заданных электрических процессов, все связи между ними, а также элементы подключения (разъемы, зажимы), которыми заканчиваются входные и выходные цепи.

Полученная принципиальная схема представлена в приложении В.

4.1 Расчет мощности элементов схемы

Потребляемая мощность разрабатываемого устройства равна сумме мощностей, потребляемых его элементами. Расчет мощности элементов схемы устройства представлен в таблице 4.1.

Таблица 4.1 – Расчет мощности элементов схемы устройства

Блок	U, В	I, мА	Кол-во	P, мВт
Микроконтроллер Arduino Nano	5	40	1	200
Bluetooth-приемопередатчик HC-06	5	40	1	200
Драйвер моторов MX-1508	5	36	1	180
Мотор-редуктор 1:48 3-8 В	5	150	4	3000

В данной схеме используется микроконтроллер Arduino Nano, Bluetooth-приемопередатчик HC-06, драйвер моторов MX-1508 и 4 мотора-редуктора 1:48 3-8 В.

Таким образом потребляемая мощность будет равна:

$P = 5 \cdot 40 + 3.3 \cdot 40 + 5 \cdot 36 + 5 \cdot 150 \cdot 4 = 3580$ мВт.

Учитывая максимальный поправочный коэффициент в 20%, максимальная потребляемая мощность составит 4296 мВт.

Рассчитаем потребляемый ток:

$$I = \frac{P}{U} = \frac{4.296}{5} = 0.8592 \approx 0.86 \text{ А}$$

4.2 Микроконтроллер

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

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

В схеме передвижного устройства Bluetooth-приемопередатчик подключен к цифровым входам D12 и D13, драйвер моторов подключен к Arduino через цифровые входы D11, D10 и D3, D2. Данный микроконтроллер питается от напряжения 5В.

4.3 Модуль приема

Информация о выбранном модуле приема представлена в пункте 3.2 раздела 3.

На устройстве используется приемопередатчик HC-06, который питается от напряжения 5В. Контакт RXD подключен к цифровому входу D13 микроконтроллера, TXD подключен к D12, на VCC подается плюс питания, а GND соединяется с минусом.

Контакт RXD предназначен для приема последовательных данных со скоростью 9600 бит/с и имеет рабочую логику 3.3 В.

Контакт TXD предназначен для передачи последовательных данных со скоростью 9600 бит/с и имеет рабочую логику 3.3 В.

4.4 Драйвер моторов

Информация о выбранном драйвере моторов MX-1508 представлена в пункте 3.3 раздела 3.

На вход плюс и минус подается напряжение 5В. Входы IN1, IN2, IN3, IN4 подключаются к контактам микроконтроллера D2, D3, D10, D11 соответственно, который выставляет направление вращения моторов. Драйвер анализирует полученные значения и запускает моторы через выходы OUT1, OUT2, OUT3 и OUT4.

4.5 Мотор-редукторы

Информация о выбранных мотор-редукторах представлена в пункте 3.4 раздела 3.

Все моторы связаны с микроконтроллером через драйвер двигателей MX-1508. Поскольку драйвер предназначен только для двух моторов, то было принято решение соединить их параллельно. Левые моторы подключаются к выходам MOTOR-B, правые к выходам MOTOR-A. При поступлении питания на соответствующие входы моторов они начинают вращаться в заданную сторону.

5 РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

5.1 Требования к разработке программного обеспечения

При разработке программного обеспечения следует учитывать следующие факторы:

– сигналы должны поступать на машину с мобильного телефона по Bluetooth;

– в зависимости от поданного сигнала, машина должна ехать в соответствующем направлении;

– все приходящие сигналы должны обрабатывать на микроконтроллере Arduino Nano и отправлять на драйвер.

Устройство работает следующим образом. При подаче питания на машину, загораются светодиоды Arduino и начинают моргать красный и синий светодиоды Bluetooth-модуля. Это означает, что необходимо подключится по Bluetooth к машине через приложение. Запуская приложение на мобильном телефоне в первый раз, сначала необходимо в настройках Bluetooth телефона добавить устройство. При добавлении будет запрошен стандартный пароль 1234, который необходимо ввести, чтобы устройство добавилось. В дальнейшем, при подключении к машине, этого не требуется делать, так как программа запомнила устройство. Когда пользователь меняет направление на джойстиках, программа анализирует полученную информацию и отправляет команду на машину через Bluetooth-передатчик телефона. Затем полученная информация анализируется на устройстве и в зависимости от полученного кода состояния запускаются или приостанавливаются двигатели.

5.2 Исходный код программы передвижного устройства

В данном блоке детально рассмотрим функционирование программного обеспечения.

Для выполнения задачи, поставленной в ходе реализации проекта, будет произведен подробный обзор архитектуры реализуемого программного обеспечения, будут рассмотрены основные классы, их строение, связи между классами и другие зависимости.

Так как в разработке приложения используется объектно-ориентированная парадигма программирования, все модули данного программного обеспечения представлены различными классами, логически объединенными в зависимости от выполняемой функции.

5.2.1 Класс `Motor`

Данный класс является основным в программе и в нем сосредоточена основная логика для управления машиной. Рассмотрим подробно методы, представленные в данном классе:

– Motor(M_driverType type, int8_t param1, int8_t param2, int8_t param3, int8_t param4) – данный конструктор создает объект класса для драйвера, в зависимости от того, какой подключен. В данном случае используется мотор на два выхода. Также в данном конструкторе инициализируются сигналы, которые будут отправляться на контакты для движения машины;

– void setSpeed(int16_t duty) – данный метод устанавливает скорость вращения колес. 0-255 для 8 бит и 0-1023 для 10 бит;

– void setMode(GM_workMode mode) – данный метод устанавливает следующие режимы работы мотора: FORWARD – вперед, BACKWARD – назад, STOP – остановить, BRAKE – активный тормоз, AUTO – подчиняется setSpeed(-255..255);

– void setDirection(bool direction) – данный метод устанавливает направление вращения колес. Могут иметь два типа: NORM – обычное, REVERSE – обратное;

– void setMinDuty(int duty) – данный метод устанавливает минимальную скважность;

– void setResolution(byte bit) – данный метод устанавливает разрешение ШИМ в битах;

– void setDeadtime(uint16_t deadtime) – данный метод устанавливает минимальное время после каждого события, в течение которого система не может зарегистрировать другое событие;

– void setLevel(int8_t level) – данный метод устанавливает уровень драйвера. По умолчанию HIGH;

– void smoothTick(int16_t duty) – данный метод отвечает за плавное изменение к указанной скорости (значению ШИМ);

– void setSmoothSpeed(uint8_t speed) – данный метод устанавливает быстроту изменения скорости;

– int getState() – данный метод возвращает состояние движения машины. -1 при вращении BACKWARD, 1 при FORWARD, 0 при остановке и торможении;

– void set8bitMode() – данный метод устанавливает выход в 8 бит;

– void set10bitMode() – данный метод устанавливает выход в 10 бит;

– protected void setPins(bool a, bool b, int c) – данный метод позволяет подать на два цифровых сигнала и один аналоговый питание 5В;

– void run(GM_workMode mode, int16_t duty = 0) – данный метод предназначен для подачи сигнала мотору. Он проверяет параметр dadetime, значение которого, если больше 0, то происходит установка нуля на контакты. А далее, в зависимости от выбранного направления (FORWARD,

BACKWARD, BRAKE, STOP), устанавливаются соответствующие сигналы на пины.

Данный класс включает следующие поля:

– int16_t _dutyS – данное поле отвечает за хранение значения скорости движения, установленной стиком;

– int _minDuty – данное поле отвечает за хранение значения минимальной скорости, при которой мотор должен начать крутиться;

– int state – данное поле отвечает за хранение значения состояния движения машины. -1 при вращении BACKWARD, 1 при FORWARD, 0 при остановке и торможении;

– int8_t _digA – данное поле отвечает за хранение значения цифрового сигнала, который будет подан на выход OUTPUT;

– int8_t _digB – данное поле отвечает за хранение значения цифрового сигнала, который будет подан на выход OUTPUT;

– int8_t _pwmC – данное поле отвечает за хранение значения аналогового сигнала, который будет подан на выход OUTPUT;

– bool _direction – данное поле отвечает за хранение значения инвертирования управления. false – управление не инвертировано, true – инвертировано.

– int8_t _level – данное поле отвечает за хранение значения уровня напряжения, на пин. Может принимать значение HIGH или LOW.

– int _maxDuty – данное поле отвечает за хранение значения максимальной скорости вращения колес.

– M_workMode _mode – данное поле отвечает за хранение значения режима, в котором находится машина. Возможны варианты FORWARD или BACKWARD;

– M_workMode _lastMode – данное поле отвечает за хранение значения режима, в котором находилась машина. Служит для предотвращения установки одного и того же режима два раза подряд. Может быть FORWARD или BACKWARD;

– M_driverType _type – данное поле отвечает за хранение значения используемого драйвера;

– uint16_t _deadtime – данное поле отвечает за хранение значения минимального времени после каждого события, в течение которого система не может зарегистрировать другое событие;

– uint8_t _speed – данное поле отвечает за хранение значения скорости вращения колес;

– uint32_t _tmr – данное поле отвечает за хранение значения количества секунд с момента начала выполнения текущей программы на плате Arduino;

– float _k – данное поле отвечает за хранение значения минимальной скважности.

5.2.2 Метод `setup`

Данный метод предназначен для инициализации переменных, определения режимов работы выводов, запуска используемых библиотек. Загрузив программу, Arduino дает коду возможность поучаствовать в инициализации системы. Для этого мы должны указать микроконтроллеру команды, которые он выполнит в момент загрузки и потом забудет про них (т.е. эти команды выполнятся только один раз при старте системы). И именно с этой целью в программе выделяется блок, в котором будут храниться эти команды.

В данном методе происходит:

– инициализация порта вывода;

– инициализация Bluetooth-модуля;

– установка режима работы правых и левых моторов;

– установка минимальной скорости, при которой мотор должен начать крутиться для правых и левых моторов;

– установка направления для правых и левых моторов (`RIGHT_MOTOR_DIRECTION` и `LEFT_MOTOR_DIRECTION` соответственно).

5.2.3 Метод `loop`

Данный метод является основным в программе и устанавливает точку входа в программу. Начав выполнение программы, Arduino выполняет ряд операций по инициализации и настройке среды окружения и только затем приступает к выполнению самого кода, который содержится в скетче. Таким образом, микроконтроллер избавляет от необходимости помнить все детали архитектуры микропроцессора.

В данном методе происходит:

– расчет сигнала по x;

– расчет сигнала по y;

– считывание сигнала для правого мотора в `dutyR`;

– считывание сигнала для левого мотора в `dutyL`;

– ограничение значений для правых моторов `dutyR` < 255;

– ограничение значений для левых моторов `dutyL < 255`;

– прокрутка правых моторов;

– прокрутка левых моторов.

5.2.4 Метод `parsing`

Данный метод является не менее важным, чем описанный выше. Он читает информацию из Bluetooth-модуля и обрабатывает ее для дальнейшей отправки на микроконтроллер. У Bluetooth-модуля имеется буфер, в который информация записывается. Информация в нем представлена в виде строки. Чтение данных из данного буфера происходит в символьную переменную, из которой уже с помощью библиотечного метода `string_convert.toInt()` получается итоговое значение.

В данном методе происходит:

– проверка буфера на наличие данных;
– чтение данных из буфера;
– принятие пакета `dataX`;
– принятие пакета `dataY`;
– конвертация пакета в переменную;
– окончание парсинга и очистка переменных пакета.
Исходный код программы представлен в приложении Г.

5.3 Исходный код программы мобильного приложения

В данном блоке подробно рассмотрим функционирование мобильного программного обеспечения.

Для выполнения поставленной задачи в рамках проекта необходимо проанализировать архитектуру программного обеспечения, которое будет реализовано. Для этого будут изучены основные классы, их структура, взаимосвязи и зависимости друг от друга. При разработке приложения использовалась объектно-ориентированная парадигма программирования, поэтому все компоненты программного обеспечения представлены различными классами, которые логически объединены в соответствии с выполняемыми функциями. Стоит отметить, что данное приложение было разработано под платформу Android на языке программирования Kotlin.

5.3.1 Класс `MainActivity`

Данный класс `MainActivity` является главной активностью мобильного приложения, которая представляет собой первый экран, который пользователь видит при запуске приложения. Этот класс наследуется от класса `AppCompatActivity`, который предоставляет ряд базовых функций для работы с пользовательским интерфейсом.

В данном классе определена приватная переменная `navController` типа `NavController`. Для работы с навигацией между фрагментами, которые составляют экраны приложения, в данном классе используется `NavController`.

В методе `onCreate`, который вызывается при создании экземпляра класса `MainActivity`, устанавливается макет экрана, который определяется с помощью метода `setContentView`. Затем находится `NavHostFragment` по его ID и извлекается `NavController`, который будет использоваться для переходов между фрагментами.

5.3.2 Класс `JoystickFragment`

Данный класс `JoystickFragment` представляет собой фрагмент мобильного приложения, который содержит в себе виджет джойстика, отображение текущих координат джойстика, статус соединения с Bluetooth-

устройством и кнопку перехода на экран настроек приложения. Поля данного класса имеют следующий вид:

– `private lateinit var joystickView` – данное поле представляет собой экземпляр класса `JoystickView`, который отвечает за отрисовку и работу с джойстиком;

– `private lateinit var coordinatesTextView` – данное поле представляет собой экземпляр класса `TextView`, который отображает текущие координаты джойстика;

– `private lateinit var btStatusTextView` – данное поле представляет собой экземпляр класса `TextView`, который отображает статус соединения с Bluetooth-устройством;

– `private lateinit var bluetoothChecker` – данное поле представляет собой экземпляр класса `BluetoothChecker`, который проверяет наличие и доступность Bluetooth на устройстве.

Далее будут более подробно рассмотрены методы данного класса:

– `override fun onCreateView` – в данном методе фрагмента устанавливается макет, в котором находятся все необходимые элементы для работы с джойстиком, отображения координат и статуса Bluetooth-соединения. Также создается экземпляр класса `BluetoothChecker`, который проверяет доступность Bluetooth на устройстве и ищет доступные Bluetooth-устройства для подключения. Устанавливается обработчик для перемещения джойстика `setOnMoveListener`. Если устройство подключено, то отправляются текущие координаты джойстика с определенной задержкой. Также меняется статус соединения Bluetooth в соответствии с наличием подключения;

– `override fun onDestroy` – данный метод отвечает за остановку проверки доступности Bluetooth, чтобы исключить затраты на использование лишних ресурсов;

– `private fun sendCoordinates` – данный метод отправляет текущие координаты джойстика на Bluetooth-устройство с помощью потока вывода `BluetoothConstant.outputStream`, если устройство подключено.

5.3.3 Класс `JoystickView`

Данный класс представляет пользовательский виджет в Android, который является визуальным джойстиком для управления направлением движения и силой.

Конструктор класса `JoystickView` принимает три параметра: `context`, `attrs` и `defStyleAttr`. Параметры `context` и `attrs` используются для создания объекта `View`, а параметр `defStyleAttr` используется для определения стиля внешнего вида виджета. Поля данного класса имеют следующий вид:

– `private val DEFAULT_RADIUS` – данное поле представляет собой значение радиуса круга по умолчанию;

– `private val DEFAULT_COLOR` – данное поле представляет собой значение цвета круга по умолчанию;

– `private val DEFAULT_THUMB_RADIUS` – данное поле представляет собой значение радиуса ползунка по умолчанию;

– `private val DEFAULT_THUMB_COLOR` – данное поле представляет собой значение цвета ползунка по умолчанию;

– `private val circlePaint` – данное поле представляет собой объект `Paint`, который используется для рисования круга;

– `private val thumbPaint` – данное поле представляет собой объект `Paint`, который используется для рисования ползунка;

– `private val radius` – данное поле представляет собой радиус круга;

– `private val center` – данное поле представляет собой координаты центра круга;

– `private val thumbRadius` – данное поле представляет собой радиус ползунка;

– `private val thumbPosition` – данное поле представляет собой координаты ползунка

– `private val listener` – данное поле отвечает за реагирования компонента на событие, происходящие в другом компоненте. В данном классе `listener` определен как переменная, которая хранит лямбда-выражение `(angle: Float, strength: Float) -> Unit`. Когда пользователь перемещает палец по экрану, слушатель получает угол и силу движения джойстика, и выполняет определенное действие, которое определено во внешнем коде приложения.

Методы данного класса выглядят следующим образом:

– `init` – данный блок класса используется для настройки обработки событий касания, включая расчет расстояния и угла от центра круга до точки касания, а также вызов метода слушателя при перемещении и отпускании касания;

– `override fun onMeasure` – данный метод используется для определения желаемого и фактического размера виджета. Изначально высчитывается конкретный размер, исходя из радиуса. Далее считается актуальный размер по характеристикам. Финальный размер определяется, как наименьший из актуальных;

– `override fun onDraw` – данный метод отвечает за отрисовку круга и ползунка;

– `fun setOnMoveListener` – данный метод устанавливает слушателя на перемещение маркера на джойстике. Метод принимает функцию `(angle: Float, strength: Float) -> Unit`, которая вызывается каждый раз, когда позиция маркера изменяется, и передает два аргумента: угол

в радианах между центром джойстика и текущей позицией маркера, и силу перемещения в диапазоне от 0 до 1, где 1 - максимальная дистанция, которую можно пройти на джойстике.

5.3.4 Класс `BluetoothChecker`

Данный класс предназначен для проверки состояния Bluetooth-соединения. Поля данного класса имеют следующий вид:

– `private var isChecking` – данное поле указывает, выполняется ли в данный момент проверка соединения;

– `private val checkIntervalMillis` – данное поле является константой, которая указывает интервал между проверками соединения в миллисекундах;

– `private var job` – данное поле является объектом типа `Job`, который используется для запуска и остановки фоновой проверки соединения;

– `private const val TAG` – константа, которая используется для журналирования ошибок в `closeConnections()`.

Методы данного класса имеют следующий вид:

– `fun start` – данный метод запускает фоновую проверку соединения. Он устанавливает значение `isChecking` на `true`, и запускает цикл проверки, который выполняется до тех пор, пока `isChecking` равен `true`. Если во время проверки обнаружено, что Bluetooth соединение отсутствует, метод вызывает `closeConnections()` для закрытия всех соединений и завершает проверку;

– `fun stop` – данный метод останавливает фоновую проверку соединения. Он устанавливает значение `isChecking` на `false` и отменяет `job`, чтобы остановить цикл проверки;

– `private suspend fun isBluetoothConnected` – данный метод проверяет, подключено ли устройство по Bluetooth. Он отправляет сигнал через выходной поток, чтобы узнать, доступен ли он. Если сигнал успешно отправлен, метод возвращает `true`, в противном случае – `false`;

– `private suspend fun closeConnections` – данный метод закрывает все соединения по Bluetooth, связанные с устройством. Он закрывает выходной поток, входной поток, Bluetooth-соединение и устанавливает значение `connectedDevice` в `null`. Если во время закрытия происходят ошибки, они записываются в журнал `Log`.

5.3.5 Класс `SettingsFragment`

Данный класс представляет фрагмент настроек, который позволяет пользователю выбрать устройство Bluetooth для подключения. Данный класс имеет несколько полей. Поля представляют собой следующий вид:

– `private lateinit var bluetoothAdapter` – данное поле используется для управления Bluetooth-функциональностью устройства,

такой как запрос списка связанных устройств и установление соединения с удаленным устройством по его адресу MAC;

– `private const val TAG` – константа, которая используется для журналирования в классе данного приложения.

Методы данного класса описаны ниже:

– `override fun onCreateView` – данный метод вызывается при создании пользовательского интерфейса фрагмента. В этом методе фрагмент создает и настраивает свой пользовательский интерфейс, а также выполняет определенные действия, такие как запрос разрешений, доступ к Bluetooth-адаптеру, создание списка устройств и установка обработчика щелчка на элементах списка;

– `private fun connectToBluetoothDevice` – данный метод выполняет подключение к выбранному Bluetooth-устройству с помощью создания соединения через Bluetooth-адаптер, создания Bluetooth-сокета и сохранения потоков ввода/вывода для обмена данными между устройствами.

5.3.6 Объект `BluetoothConstant`

Данный синглтон предоставляет глобальные переменные для хранения состояния Bluetooth-соединения и используется для передачи данных между различными фрагментами приложения. Он имеет следующие поля:

– `var outputStream` – поток вывода данных Bluetooth-соединения;

– `var inputStream` – поток ввода данных Bluetooth-соединения;

– `var connectedDevice` – объект Bluetooth-устройства, к которому установлено соединение;

– `var bluetoothSocket` – Bluetooth-сокет, который представляет установленное соединение.

ЗАКЛЮЧЕНИЕ

В результате работы над данным курсовым проектом было разработано работоспособное микропроцессорное передвижное устройство на Bluetooth-управлении со своим программным обеспечением. 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 компонентами.

В дальнейшем планируется усовершенствование данного курсового проекта. Одним из таких улучшений является добавление FPV-камеры для получения изображения в реальном времени.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Arduino Nano [Электронный ресурс]. – Режим доступа: https://arduino.ru/Hardware/ArduinoBoardNano

[2] BK3231 SPP-C [Электронный ресурс]. – Режим доступа: https://roboticafacil.es/datasheets/SPP.pdf

[3] Bluetooth BT-06 [Электронный ресурс]. – Режим доступа: https://www.alldatasheet.com/view.jsp?Searchword=BT06

[4] Bluetooth HC-06 [Электронный ресурс]. – Режим доступа: https://www.olimex.com/Products/Components/RF/BLUETOOTH-SERIAL-HC-06/resources/hc06.pdf

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

[6] Плата расширения L298P [Электронный ресурс]. – Режим доступа: https://radioprog.ru/shop/merch/41

[7] Драйвер мотора MX1508 [Электронный ресурс]. – Режим доступа: https://robotchip.ru/obzor-drayvera-motora-mx1508/

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

ПРИЛОЖЕНИЕ А
(обязательное)

Схема структурная

ПРИЛОЖЕНИЕ Б
(обязательное)

Схема функциональная

ПРИЛОЖЕНИЕ В
(обязательное)

Схема принципиальная

ПРИЛОЖЕНИЕ Г
(обязательное)

Код программы

ПРИЛОЖЕНИЕ Д
(обязательное)

Перечень элементов

ПРИЛОЖЕНИЕ Е
(обязательное)

Ведомость документов