МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Тольяттинский государственный университет»

Институт машиностроения
(наименование института полностью)

Кафедра «Промышленная электроника»
(наименование)

11.03.04 Электроника и наноэлектроника
(код и наименование направления подготовки/ специальности)

Электроника и робототехника
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему Модуль акселератора для электрического гоночного болида

Обучающийся А.Э. Дробченко
(Инициалы Фамилия) (личная подпись)

Руководитель к.т.н., доцент, М.В. Позднов
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Консультант к.ф.н., доцент, М.М. Бажутина
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2022

Аннотация

Тема бакалаврской выпускной квалификационной работы – модуль акселератора для электрического гоночного болида. Объём работы 67 стр., 69 рисунков, 7 таблиц, 22 источника и 1 приложение.

Объектом выпускной квалификационной работы является устройство для подключения педали акселератора к блоку управления двигателем автомобиля, имеющее несколько режимов работы для универсальности.

Целью работы является сокращение сроков проектирования гоночного болида класса Formula Student за счёт унификации узлов, а также сокращение количества проводов, идущих от передней части автомобиля до модуля управления, который как правило находится в задней части шасси, за счёт использования CAN шины, состоящей из экранированной витой пары.

В процессе решения поставленной цели определены и выполнены следующие задачи:

- анализ доступных на рынке универсальных электронных педалей акселератора;
- определение параметров устройства;
- конструкторское проектирование устройства;
- разработка схемы электрической принципиальной;
- разработка печатного узла;
- разработка корпуса устройства;
- экспериментальное исследование работы устройства.

Выпускная квалификационная работа состоит из пяти разделов, в которых последовательно решены обозначенные выше задачи.

В первом разделе формулируется актуальность, цель и задачи проекта, определяются критерии для разработки устройства, а также проводится анализ рынка на наличие аналогов.

Во втором разделе описывается конструкторское проектирование устройства, варианты разработки устройства и его структура.

В третьем разделе описывается программный код и алгоритм работы устройства.

В четвертым разделе разрабатывается печатная плата, корпус и сопутствующие элементы.

В пятом разделе описывается экспериментальное исследование работы устройства

В процессе выполнения работы использовалось следующее программное обеспечение:

- Visual studio code – программная среда для программирования микроконтроллера;
- КОМПАС-3D V16 – создание чертежей;
- Easy EDA – проектирование и разводка печатной платы
- Ultimaker CURA – подготовка к печати корпуса устройства;
- Tinker CAD – проектирование корпуса устройства;
- Adobe Photoshop – изготовление наклейки на печатную плату.

Областью применения данной ВКР является использование спроектированного устройства в гоночных болидах класса Formula Student как с двигателем внутреннего сгорания, так и на электрической тяге.

Abstract

The title of the bachelor's thesis is «Accelerator module for an electric racing car». The work consists of a 76-page explanatory note including 69 figures, 7 tables, a list of 22 references, 1 appendix and the graphic part on 6 A1 sheets.

The object of the research is a device for connecting the accelerator pedal to the vehicle engine control unit, which has several modes of operation for versatility.

The aim of research is to reduce the design time for a Formula Student racing car by unifying the nodes, as well as reducing the number of wires running from the front of the car to the control module, which is usually located in the rear of the chassis, through the use of a CAN bus consisting of a shielded twisted wire.

In the process of achieving the aim, the following objectives were identified and completed:

− analysis of universal electronic accelerator pedals available on the market;

− determination of device parameters;

− engineering design of the device;

− development of electrical circuit diagram;

− development of a printed circuit;

− development of the device housing shell;

− experimental study of the operation of the device.

It can be concluded that all the tasks being accomplished.

The bachelor's thesis consists of five chapters, in which the above objectives were sequentially performed.

The first part formulates the relevance, purpose and objectives of the project, defines the criteria for the development of the device, and analyzes the market for analogues.

The second part describes the design of the device, options for developing the device and its structure.

The third part describes the program code and algorithm of the device.

In the fourth part, the printed circuit board, package and related elements are developed.

The fifth part describes an experimental study of the operation of the device.

The following software was used during the work:

- Visual studio code – software environment for microcontroller programming;
- KOMPAS-3D V16 – creation of drawings;
- Easy EDA - PCB design and layout;
- Ultimaker CURA – preparing the device case for printing;
- Tinker CAD – design of the device housing shell;
- Adobe Photoshop - making a paper sticker for a printed circuit board.

Brief conclusions on the results of the bachelor's thesis: the use of the designed device in racing cars of the Formula Student class, both with an internal combustion engine and with electric motors.

Содержание

Введение

Исторически всегда существовала механическая связь между педалью акселератора и дроссельной заслонкой, будь то трос или тяги и рычаги. Сегодня же их заменили сложные электронные модули управления, датчики и приводы. Эта система называется «Drive-by-Wire».

«Система «Drive-by-Wire» не является новой концепцией, поскольку она была представлена BMW на их 7-й серии ещё в 1988 году. Система, используемая BMW, называется EML (electronic throttle management) (немецкий термин для электронного управления дроссельной заслонкой). Сегодня система нашла применение и в других транспортных средствах, в том числе и бюджетных моделях.» [21]

Есть несколько причин, по которым электронный привод дроссельной заслонки предпочтительнее обычного троса:

- бортовые электронные системы автомобиля способны контролировать всю работу двигателя, в том числе количество поступающего воздуха;
- оптимизация подачи воздуха гарантирует, что вредные выбросы выхлопных газов будут сведены к абсолютному минимуму, а управляемость будет сохранена независимо от обстоятельств;
- соединение электронного привода дроссельной заслонки с системами адаптивного круиз-контроля, контроля тяги, контроля скорости холостого хода и контроля устойчивости автомобиля также позволяет добиться более точного управления мотором, в том числе повышая уровень безопасности и комфорта.

Также использование электронной педали акселератора имеет преимущества перед обычным кабельным вариантом за счёт следующих факторов:

- устранение механического элемента – троса дроссельной заслонки, и замена его быстродействующей электроникой уменьшает количество

движущихся частей (и связанный с этим износ) и, следовательно, система требует минимальной регулировки и обслуживания;

- большая точность данных улучшает управляемость автомобиля, что, в свою очередь, обеспечивает лучшую реакцию, экономичность и безопасность.

К тому же, всё более распространёнными становятся электромобили, в конструкции которых невозможно использовать классическую педаль акселератора с тросовым приводом. Следуя общемировым тенденциям, команды, принимающие участие в студенческих инженерных соревнованиях Formula Student, так же начинают разрабатывать гоночные болиды на электрической тяге, но при этом многие не отказываются от параллельной постройки болида с двигателем внутреннего сгорания. Таким образом командам требуется универсальное решение, отвечающее всем требованиям для всех типов болидов.

1 Постановка задачи

1.1 Формулирование актуальности, цели и задач проекта

Целью работы является сокращение сроков проектирования гоночного болида класса Formula Student за счёт унификации крупных узлов, а также сокращение количества проводов, идущих от передней части автомобиля до модуля управления, который как правило находится в задней части, за счёт использования CAN шины.

В процессе решения поставленной цели определены и выполнены следующие задачи:

- анализ доступных на рынке универсальных электронных педалей акселератора;
- определение параметров устройства;
- конструкторское проектирование устройства;
- разработка схемы электрической принципиальной;
- разработка печатного узла;
- разработка корпуса устройства;
- экспериментальное исследование работы устройства.

1.2 Поиск и анализ технических параметров аналогичных устройств

Проведя анализ доступных решений поставленной цели в сети Интернет, составлена следующая сравнительная таблица, с помощью которой можно наглядно сравнить устройства по следующим критериям:

- возможность работы по CAN шине;
- возможность работы по аналоговой шине (от 0 В до 12 В);

− количество датчиков вращения (согласно регламенту соревнований Formula Student (пункт T 11.8.5 и T 11.8.6) [2] на оси вращения должны быть установлены 2 датчика);

− стоимость решения.

Для составления сравнительной таблицы возьмем следующие три педали от различных производителей:

− 0-5V Pedal Throttle for Electric Vehicle / Car (рисунок 1) [12];

− HS-010-635 (рисунок 2) [6];

− Модуль педальный ВАЗ Лада 1118,2190 электронная E-GAS (рисунок 3) [1];

− GM OEM Pedal Position Sensor CAN Bus (рисунок 4) [9].

Рисунок 1 – 0-5V Pedal Throttle for Electric Vehicle / Car

Данная педаль представляет собой законченное решение, питающееся от бортовой сети автомобиля (12 В), и выдающее сигнал в диапазоне от 0 В до 5 В в зависимости от угла нажатия. Недостатком решения является

крепление педали непосредственно на вал потенциометра с одной стороны, что запрещено регламентом соревнований, пункт T 11.8.5 и T 11.8.6 [2].

Рисунок 2 – HS-010-635

Педаль HS-010-635 имеет более компактные габариты, что является немаловажной характеристикой в компактном гоночном болиде. Питание – так же от бортовой сети автомобиля (12В). Сигнал – от 0 В до 12 В в зависимости от угла нажатия. Конструкция имеет такой же недостаток, как и у предыдущего образца.

Рисунок 3 – Модуль педальный ВАЗ Лада 1118,2190 электронная, E-GAS

Управление заслонкой дросселя через электронную педаль осуществляется с помощью двух датчиков, фиксирующих изменения в положении педали акселератора (датчики ДППА). Это потенциометрические резисторы с питанием в 3,3 В от контроллера. ДППА с приводом педали связаны механически при помощи пружин. Выходное напряжение датчиков зависит от силы нажатия на педаль акселератора. Если она находится в состоянии покоя, сигнал первого датчика должен составлять 0,31 В – 0,56 В, а сигнал второго – 0,15 В - 0,28 В. Когда педаль нажата, сигналы соответственно усиливаются до 1,9 В и 0,95 В. Недостатком изделия является необходимость использования микроконтроллера для преобразования сигнала в CAN сообщение, а также для использования с мотоциклетными блоками управления, используемыми в болидах класса Formula Student, в

виду необходимости преобразования напряжения в стандартный диапазон 0 В – 12 В.

Рисунок 4 – GM OEM Pedal Position Sensor CAN Bus

Модуль GM OEM Pedal Position Sensor CAN Bus представляет собой универсальный датчик положения педали для подключения к CAN сети автомобиля марки General Motors. Преимуществом модуля является компактность, а недостатками – отсутствие аналогового выхода.

Таблица 1 – Сравнительная таблица устройств

Название	Количество датчиков, шт.	Возможность работы по CAN шине	Возможность работы по аналоговой шине (0 В - 12 В)	Стоимость, руб.
0-5V Pedal Throttle for Electric Vehicle	1	-	-	5780
HS-010-635	1	-	+	2750
Модуль педальный ВАЗ Лада 1118,2190	1	-	-	1237
GM OEM Pedal Position Sensor CAN Bus	1	+	-	1414

Проанализировав исходные данные в таблице 1, сделан вывод о том, что в основном устройства с подобным функционалом имеют:

– высокую стоимость;

– отсутствие встроенной обработки данных;

– наличие только одного датчика вращения;

– отсутствие CAN трансмиттера практически во всех предлагаемых на рынке моделях.

Таким образом можно сделать вывод, что разрабатываемый проект целесообразен и выделяется среди подобных устройств уникальным функционалом и оптимальной стоимостью.

1.3 Критерии для разработки

Перед началом проектирования необходимо понять, каким образом происходит управление двигателем внутреннего сгорания или электромотором.

В двигателях внутреннего сгорания тросовые системы управления полностью вытеснены электроприводами. Они присоединены к дроссельной заслонке и управляются сигналом с электронного блока управления (ЭБУ). ЭБУ как правило посылает логический сигнал в виде уровня напряжения от 0 В до 12 В. Однако современные могут оснащаться более энергоэффективной электроникой, и верхнее напряжение понизится до 5 В или даже 3 В.

Электродвигатели управляются инверторами, к которым также подключаются педаль или курок акселератора. Инвертор, также как и ЭБУ, ожидает логический уровень напряжения на своём входе, и таким образом, выходной сигнал устройства подходит для обоих типов систем управления.

Исходя из вышеизложенного и задания на выполнение выпускной квалификационной работы, определены следующие критерии для разработки:

- источник питания: бортовая сеть 12 В;

- подключение педали с двумя датчиками вращения;
- постоянный мониторинг состояния датчиков для предотвращения аварии, а именно если разница показаний датчиков составит более 10%, то выдать ошибку и прекратить работу;
- способ соединения с главным блоком управления: CAN шина или аналоговое подключение на выбор;
- возможность изменения конфигурации устройства без программатора с помощью элементов, расположенных на плате;
- высокоскоростной отказоустойчивый контроллер;
- аппаратная поддержка CAN протокола и соответствие стандарту автомобильной промышленности SAE J1939 [13];
- изготовление платы без использования модулей для уменьшения размера и повышения надёжности соединений.

Вывод по разделу

После постановки цели выпускной квалификационной работы и определения задач для её выполнения, исследованы способы и принципы управления скоростью как двигателей внутреннего сгорания, так и электродвигателей. Далее проведён анализ существующих решений на рынке, их параметры сведены таблицу и на её основе определены критерии для разработки и основная концепция устройства.

2 Конструкторское проектирование устройства

2.1 Варианты разработки устройства и его структура

Для выбора микроконтроллера проведено сравнение, показанное в таблице 2, такого популярного решения, как Arduino Pro Micro, уже используемого в болиде eScorpion Togliatti Racing Team микроконтроллера Teensy 4.0, и несколько менее популярного, но тем не менее достаточно распространённого чипа STM32.

Таблица 2 – Сравнение микроконтроллеров

Название	I2C	Аналоговые порты, шт.	CAN протокол	Возможность приобрести чип	Стоимость, руб.
Arduino Pro Micro	+	4	-	+ ATMEGA 32U4	1000
Teensy 4.0	+	14	+	- IMXRT1062DVL6	1900
STM32f103	+	10	+	+	250

В качестве главного вычислительного устройства решено использовать микроконтроллер из семейства STM32 производителя STMicroelectronics, а именно STM32F103C6T6A (рисунок 5). Данный микроконтроллер производится в нескольких корпусах (таблица 3).

Таблица 3 – Доступные корпуса микроконтроллера STM32F103

Серия	STM32F103Cx	STM32F103Rx	STM32F103Vx
Корпус	LQFP48	LQFP64	LQFP100

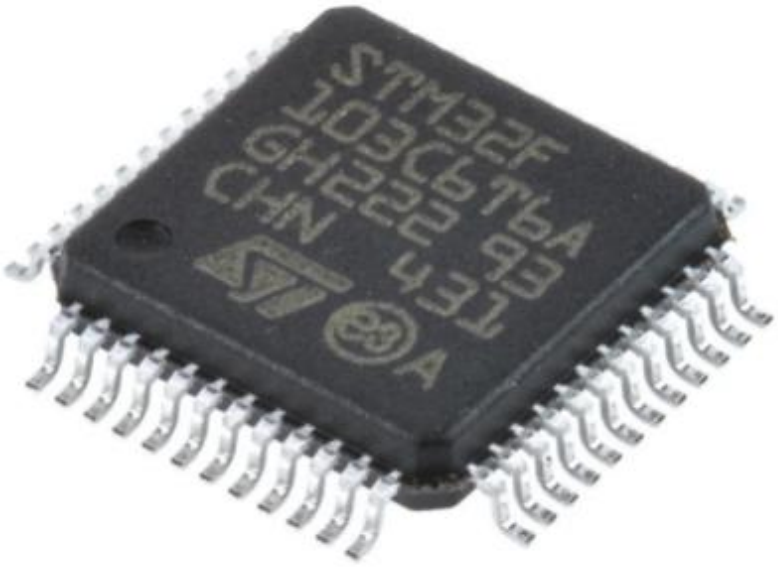

Рисунок 5 – Микроконтроллер STM32F103C6T6A

Данный микроконтроллер «обладает двумя встроенными аналогово-цифровым преобразователями с 12 битным разрешением, поддерживает протоколы CAN 2.0, I2C, и работает в широком диапазоне температур (от -40 до +105 °C)» [16]. В процессе разработки устройства удобно использовать плату с установленным микроконтроллером STM32 – BluePill [17]. Она позволяет заниматься отладкой программы на макетной плате.

Исходя из задания, для выполнения требования регламента Formula Student «требуется сравнение сигналов с двух потенциометров» [2]. Это сравнение можно проводить как программно, так и с помощью специальных микросхем. Например – ADS1115 (рисунок 6) от компании Texas Instruments, имеющая разрешение 16 бит, 4 аналоговых входа, умеющих переключаться в режим компаратора по парно. Эта микросхема производится в двух видах корпусов, и для выполнения работы выбран более крупный VSSOP для облегчения пайки экспериментального образца. «Для связи с микроконтроллером на борту имеется протокол I2C» [4].

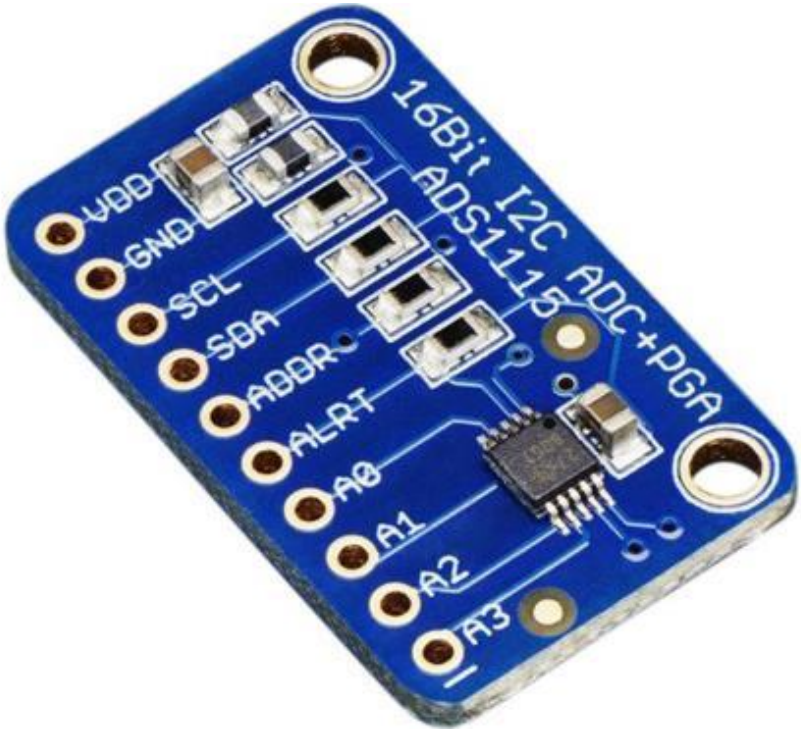

Рисунок 6 – Модуль для Arduino - ADS1115

Исходя из опыта, полученного при проектировании болида eScorpion, принято решение использовать хорошо себя зарекомендовавший CAN трансмиттер TJA1051 (рисунок 7) от компании NXP. TJA1051 это высокоскоростной приёмопередатчик CAN, обеспечивающий связь между микроконтроллером, поддерживающим Controller Area Network (CAN), и физической двухпроводной шиной CAN. «Он может напрямую подключаться к микроконтроллерам с напряжения питания от 3 В до 5 В. TJA1051 реализует физический уровень CAN в соответствии со стандартами ISO 11898-2:2016 и SAE J2284-1 – SAE J2284-5» [19].

Рисунок 7 – Модуль TJA1051

Так как педаль акселератора будет установлена на борту болида, питание производится через встроенный в аккумуляторную батарею высококачественный DC/DC преобразователь и целесообразно отказаться от фильтрации помех по шине питания. Для преобразования бортового напряжения 12 В в 3.3 В предполагается использование микросхемы AMS1117 от Texas Instruments (рисунок 8) [10].

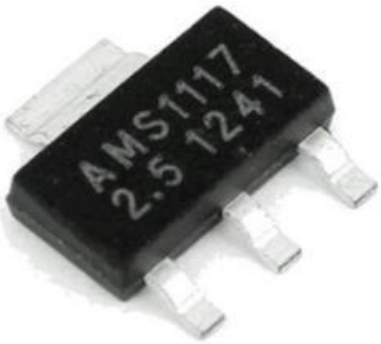

Рисунок 8 – AMS1117 на 2.5В

Таким образом, структуру устройства можно представить в виде схемы на рисунке 9.

Рисунок 9 – Структурная схема устройства

На приведённой схеме отображены все подобранные элементы, а также показана связь между ними.

2.2 Разработка электронной части устройства

Разработка электронной части устройства велась в программе EasyEDA. Далее описаны все схемотехнические решения, использованные в прототипе изготавливаемой педали акселератора для гоночного болида.

На рисунке 10 показан светодиод, информирующий о наличии питания и конденсаторы для сглаживания помех для питания микросхемы АЦП ADS1115.

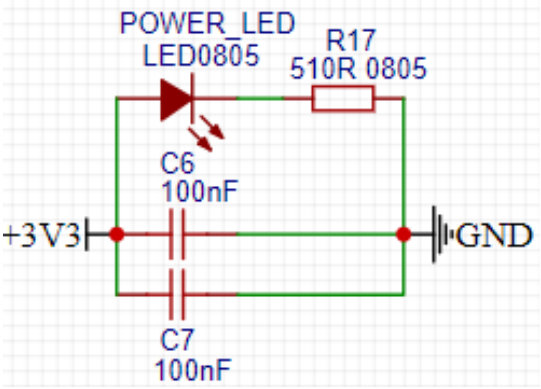

Рисунок 10 – Светодиод питания и конденсаторы в цепи питания микроконтроллера

На рисунке 11 изображена микросхема U4 CAN трансмиттера TJA1051. Если требуется наличие сопротивления 120 Ом на CAN шине, то перемычка J_CAN запаивается.

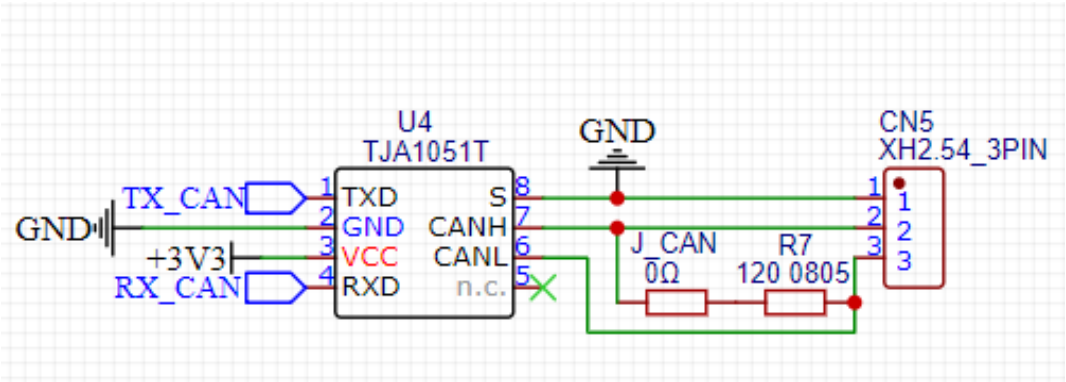

Рисунок 11 – Модуль TJA1051

Джамперы BOOT1 и BOOT0 (рисунок 12) предназначены для выбора одного из трёх вариантов запуска:

- запуск программы из памяти пользователя (нормальный режим);
- запуск загрузчика для программирования микроконтроллера;
- режим для отладки – данные не сохраняются в памяти микроконтроллера.

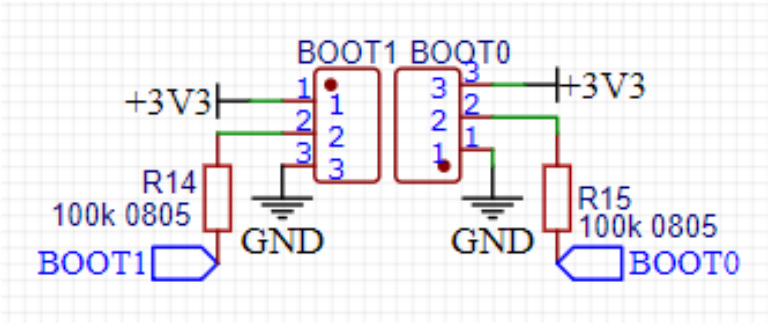

Рисунок 12 – Джамперы для включения режима программирования

Аналогово-цифровой преобразователь ADS1115, показанный на рисунке 13, подключён согласно документации производителя. Линии передачи данных I2C подтянуты с помощью резисторов R13 и R19 к питанию 3 В. Делители напряжения, например с верхним плечом R11 и с нижним R12 предназначены для понижения напряжения с 12 В датчика положения педали, до 6В, максимального напряжения с которым работает ADS1115.

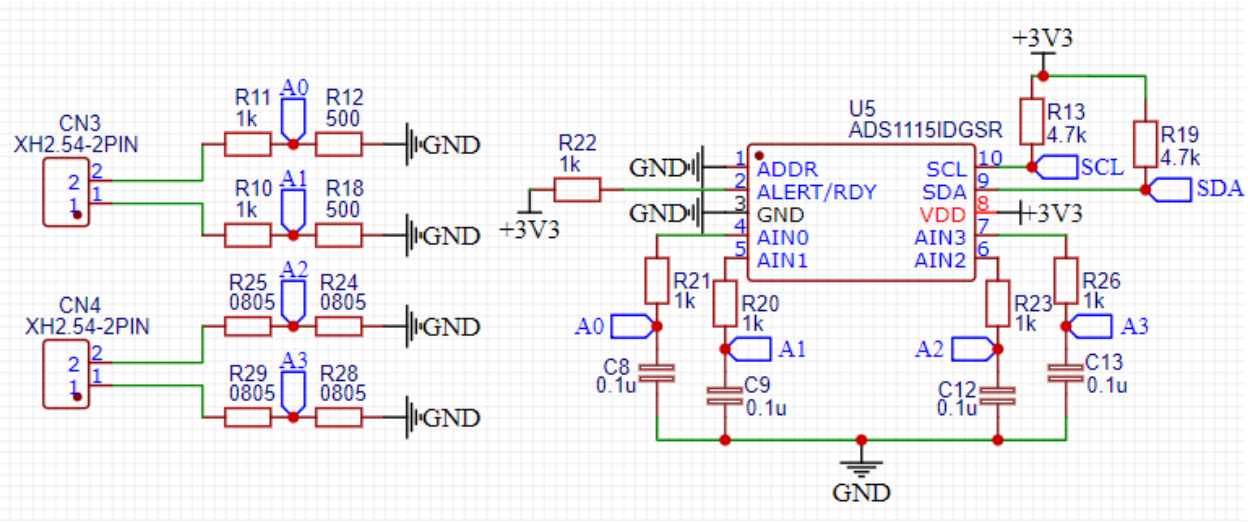

Рисунок 13 – Модуль АЦП ADS1115

Для повышения частоты работы микроконтроллера с 8 МГц до 32МГц использовались внешние пассивные тактовые генераторы (рисунок 14).

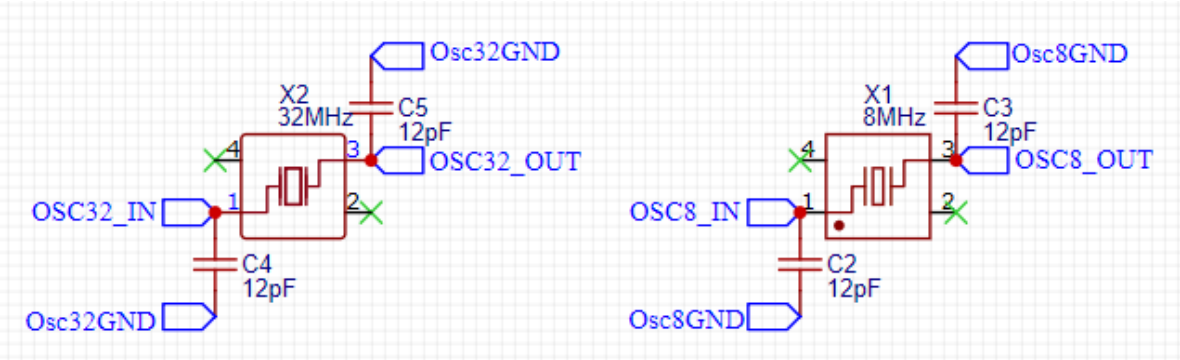

Рисунок 14 – Внешние кварцы для тактирования STM32

Разъём ISP (in-circuit serial programming) предназначен для возможности программирования уже припаянного на плату микроконтроллера STM32 (рисунок 15).

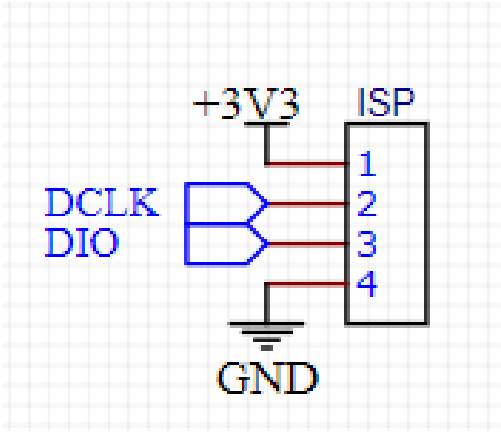

Рисунок 15 – Разъём для программирования программатором ST-Link

Для выбора режима работы Analog / CAN предназначена перемычка J_MODE, показанная на рисунке 16. В случае если она запаяна, на вход микроконтроллера STM32 приходит 3 В, иначе он подтягивается к земле. Таким образом реализовано логическое управление программой (1/0).

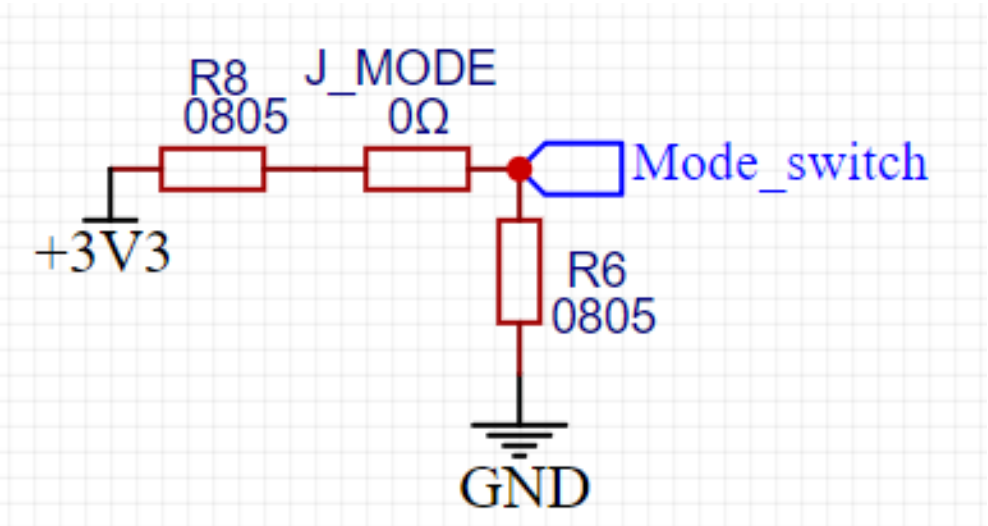

Рисунок 16 – Перемычка выбора режима работы

Далее на рисунке 17 показан микроконтроллер STM32F103C6T6, а также сигнальные светодиоды LED1, LED2 и LED3.

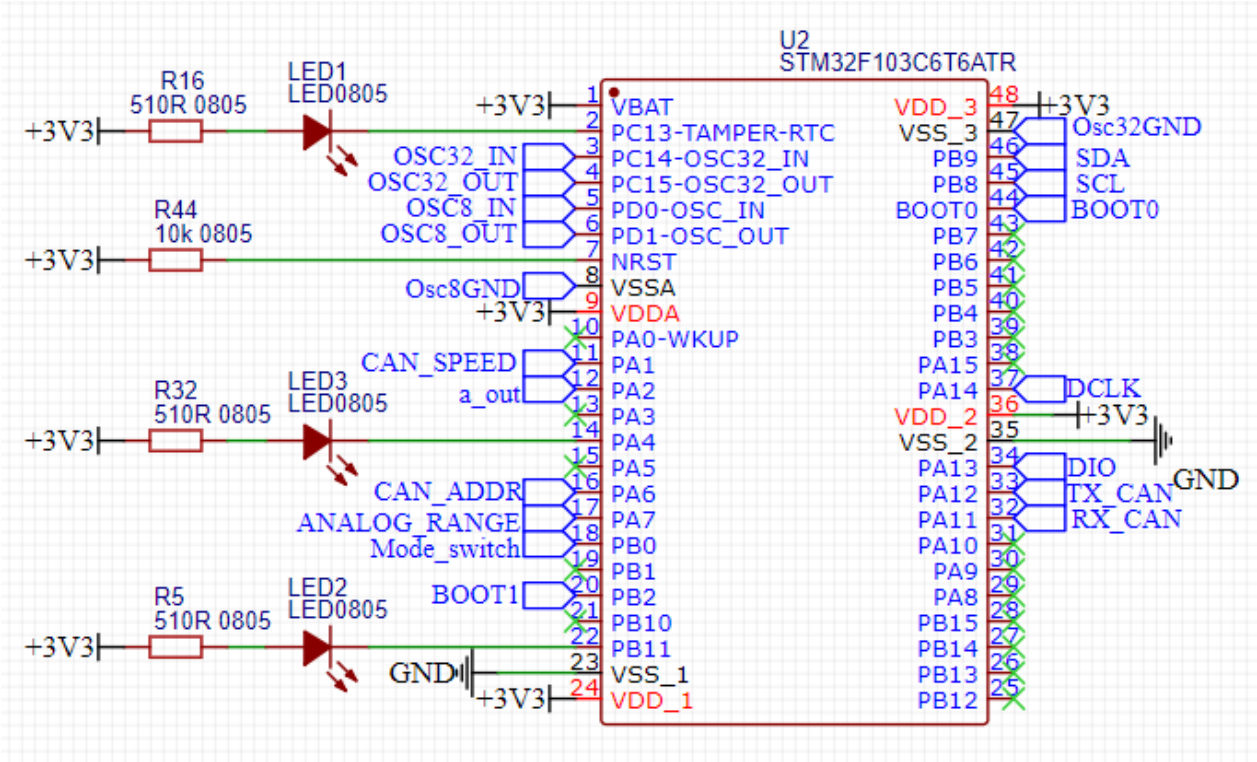

Рисунок 17 – Микроконтроллер STM32

Для согласования логических уровней используются транзисторы BSS138 (рисунок 18). В случае если выбран режим 3 В, то напряжение приходит на коннектор CN1 с выхода микроконтроллера через защитный диод D4. Если же выбран режим 12 В, то выходом микроконтроллера управляется транзистор Q1, который регулирует выходное напряжение от 12 В до 0 В. В режиме 5 В добавляется делитель напряжения (резисторы R34, R43) для получения 5 В.

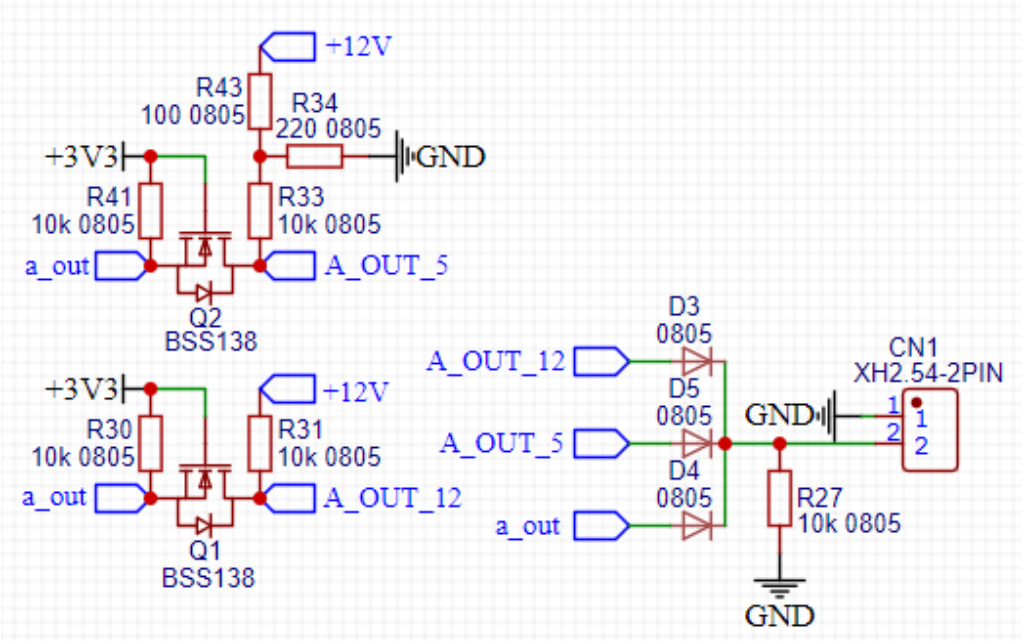

Рисунок 18 – Транзисторы в цепи аналогового выхода

Конфигурация программы осуществляется с помощью запаиваемых перемычек в делителях напряжения, показанных на рисунке 19.

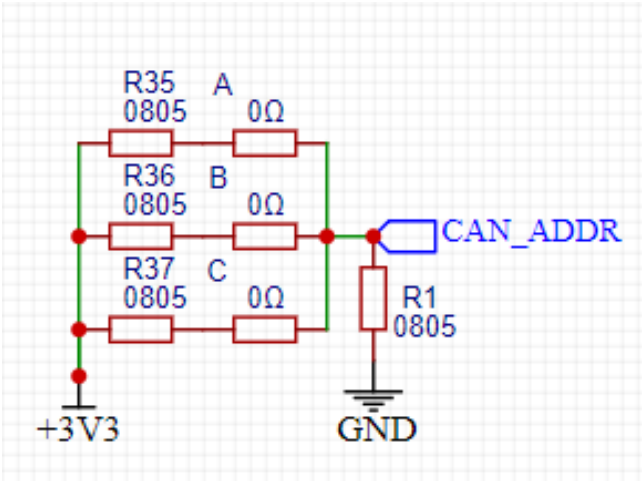

Рисунок 19 – Делитель напряжения с перемычками для программирования адреса CAN сообщения

.

В случае отсутствия перемычек, вход микроконтроллера подтягивается к земле, образуя логический 0.

2.3 Разработка схемы питания

На плате устройства будут расположены следующие микросхемы: микроконтроллер STM32F103C6T6A, CAN трансмиттер TJA 1051 и аналогово-цифровой преобразователь ADS 1115. Все эти элементы требуют напряжение 3 В для своего питания. Входное напряжение равняется 12 В, и можно принять, что оно лишено влияющих на работоспособность помех в виду наличия качественного DC/DC преобразователя на борту гоночного болида.

Для выбора решения по преобразованию напряжения 12 В в 3 В необходимо посчитать суммарное потребление всего устройства. Для этого составим таблицу 4 с перечнем элементов и их потреблением.

Таблица 4 – Токи потребления компонентами устройства

Название компонента	Потребляемый ток, мА
STM32F103C6T6A	150
TJA 1051	110
ADS 1115	100
Итого	360

Проанализировав данные в таблице, решено использовать линейный преобразователь AMS1117 с фиксированным выходным напряжением 3 В. Согласно технической документации, «максимальный ток данной микросхемы составляет 800 мА, а максимальное входное напряжение 15 В» [10], что соответствует заданию. На рисунке 20 представлена схема подключения преобразователя в устройстве. Диоды D1 и D2 обеспечивают защиту от неверного подключения питания. Конденсаторы C11 и C10 предназначены для сглаживания пульсаций.

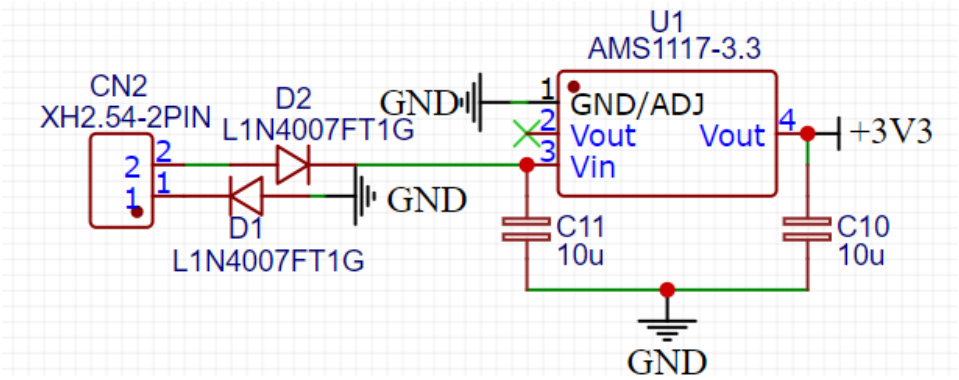

Рисунок 20 – Схема питания устройства от бортовой сети болида 12В

Вывод по разделу

В данном разделе проанализированы доступные для покупки на рынке решения на предмет соответствия техническим требованиям поставленной задачи. В результате анализа, стала очевидной необходимость разработки устройства, имеющего возможность подключения к блокам управления, работающих как с аналоговыми педалями, так и с подключаемыми по CAN шине, в виду отсутствия подобных универсальных решений. Так же анализ подтвердил актуальность работы.

Для решения задачи выбран современный и технологичный микроконтроллер STM32, а также надёжная в использовании и простая в программировании периферия в виде CAN трансмиттера TJA1051 и АЦП ADS1115.

3 Разработка алгоритма и управляющей программы

Платы Arduino и среда разработки Arduino IDE произвели революцию в том, как любители и профессионалы с минимальной подготовкой могут создавать электронные устройства дома. Однако со временем Arduino IDE может начать сковывать, особенно если ранее программировать в более полнофункциональных средах. Хорошей новостью является то, что есть другие варианты, например редактор кода Visual Studio Code в сочетании с расширением PlatformIO. Как и Arduino IDE, Visual Studio Code (VSC) работает на Windows, Mac и Linux и является бесплатным для использования. Хотя технически это не программное обеспечение с открытым исходным кодом, Microsoft утверждает, что оно «построено на основе открытого исходного кода» [20]. Таким образом, есть возможность использовать его как в личных, так и в коммерческих целях. Чтобы заставить VSC работать с Arduino (и множеством других плат), также понадобится плагин PlatformIO (PIO). PIO представляет собой «экосистему нового поколения для разработки встраиваемых систем» [3] и сама по себе имеет открытый исходный код.

В отличие от Arduino IDE, Visual Studio Code может автоматически определить COM-порт, к которому подключён микроконтроллер, что существенно облегчает отладку. Так же к достоинствам можно отнести подсветку синтаксиса, авто дополнение вводимого кода, и простую работу с библиотеками.

Для программирования Arduino обычно достаточно кабеля USB, так как на плате расположена микросхема FT232RL [8], изображённая на рисунке 21, позволяющая подключать микроконтроллер ATMEGA к компьютеру без дополнительных адаптеров или программаторов.

Рисунок 21 – Плата Arduino Duemilanove и FT232RL

Микроконтроллер STM32 не имеет возможности программирования через USB, а добавление FT232RL на плату нецелесообразно, так как устройство будет программироваться при производстве и в случае выявления ошибок в процессе эксплуатации. Таким образом необходимо использовать внешний программатор, например ST-LINK/V2, показанный на рисунке 22.

«ST-LINK/V2 это внутрисхемный отладчик и программатор для микроконтроллеров STM8 и STM32» [15]. «Стандарт однопроводного интерфейса (SWIM) и интерфейсы JTAG/Serial (SWD) используются для связи с любым микроконтроллером STM8 или STM32» [14].

Функционал ST-LINK/V2:

- питание 5 В от разъёма USB;
- полная совместимость, со стандартом USB 2.0;
- кабель USB Type-A — Mini-B в комплекте;

- поддержка прикладного напряжения от 1,65 В до 5,5 В;
- поддержка JTAG;
- поддержка прямого обновления прошивки (DFU);
- светодиод состояния информирует о связи с ПК.

Рисунок 22 – Программатор ST Link V2

Далее, на рисунке 23 показана схема подключения программатора ST Link V2 к микроконтроллеру STM32. Для программирования достаточно 4 проводов, питание на микроконтроллер подавать не требуется. В процессе программирования индикаторный светодиод моргает, и по окончании загрузки программы светится зелёным.

Прежде чем приступать к написанию кода, необходимо составить блок-схему алгоритма (БСА) и определить функционал программы. БСА – популярный формат описания алгоритмов или процессов, в котором каждый шаг отображается в виде блока определённой формы, при этом каждая форма означает своё действие, а линии между блоками показывают направление последовательности. Блок-схема алгоритма показана на рисунках 24 и 25.

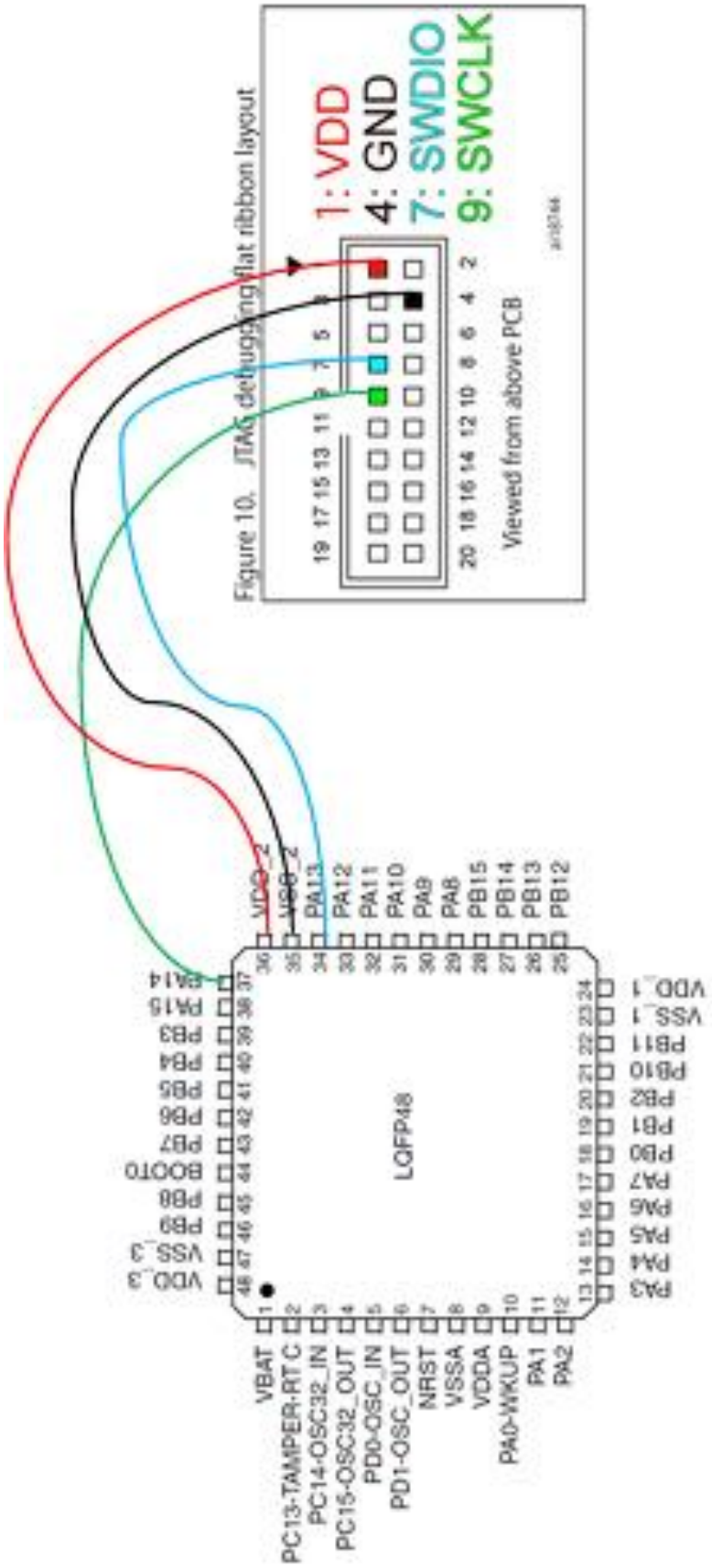

Рисунок 23 – Схема подключения программатора ST Link V2 к STM32

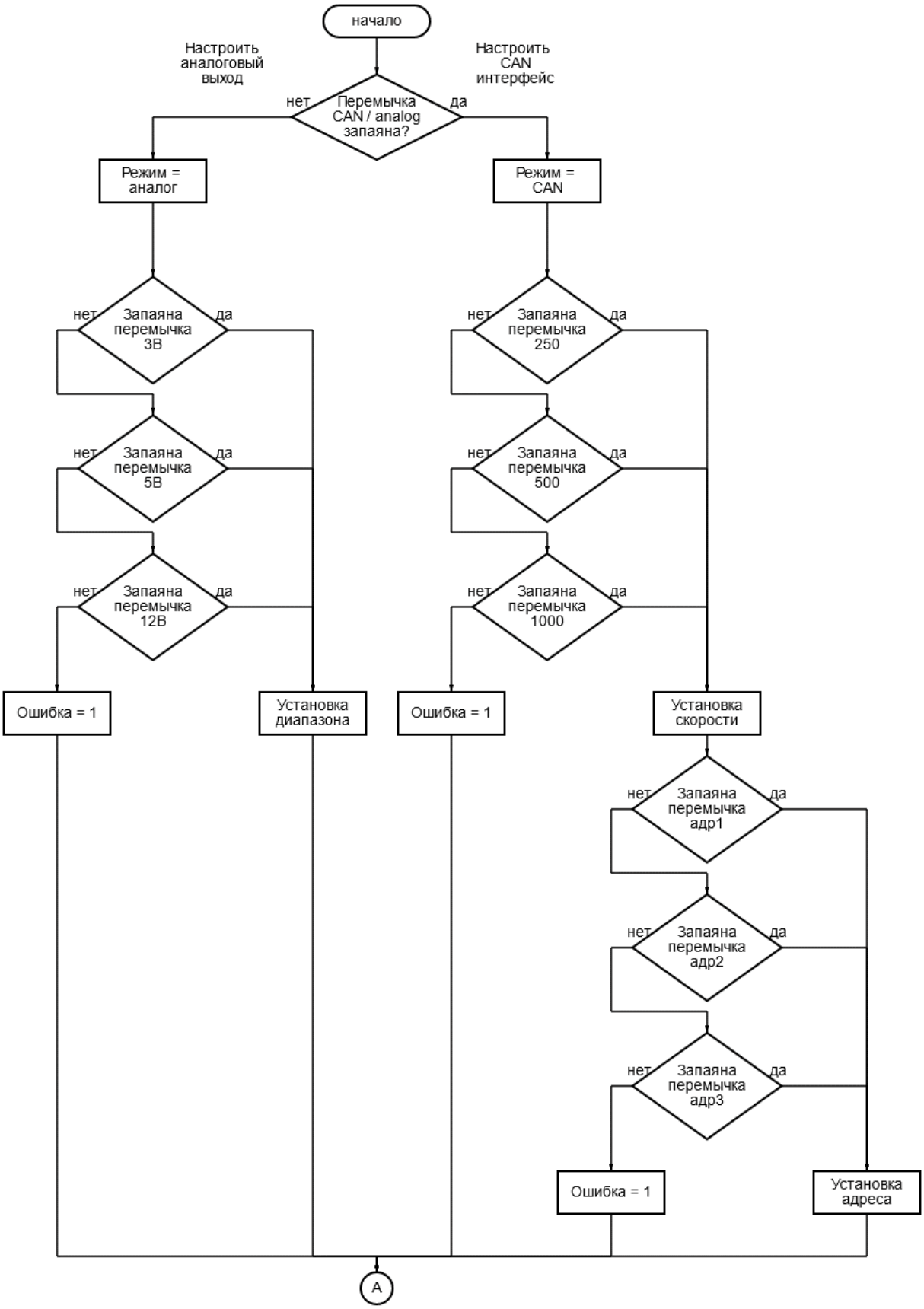

Рисунок 24 – Блок схема алгоритма

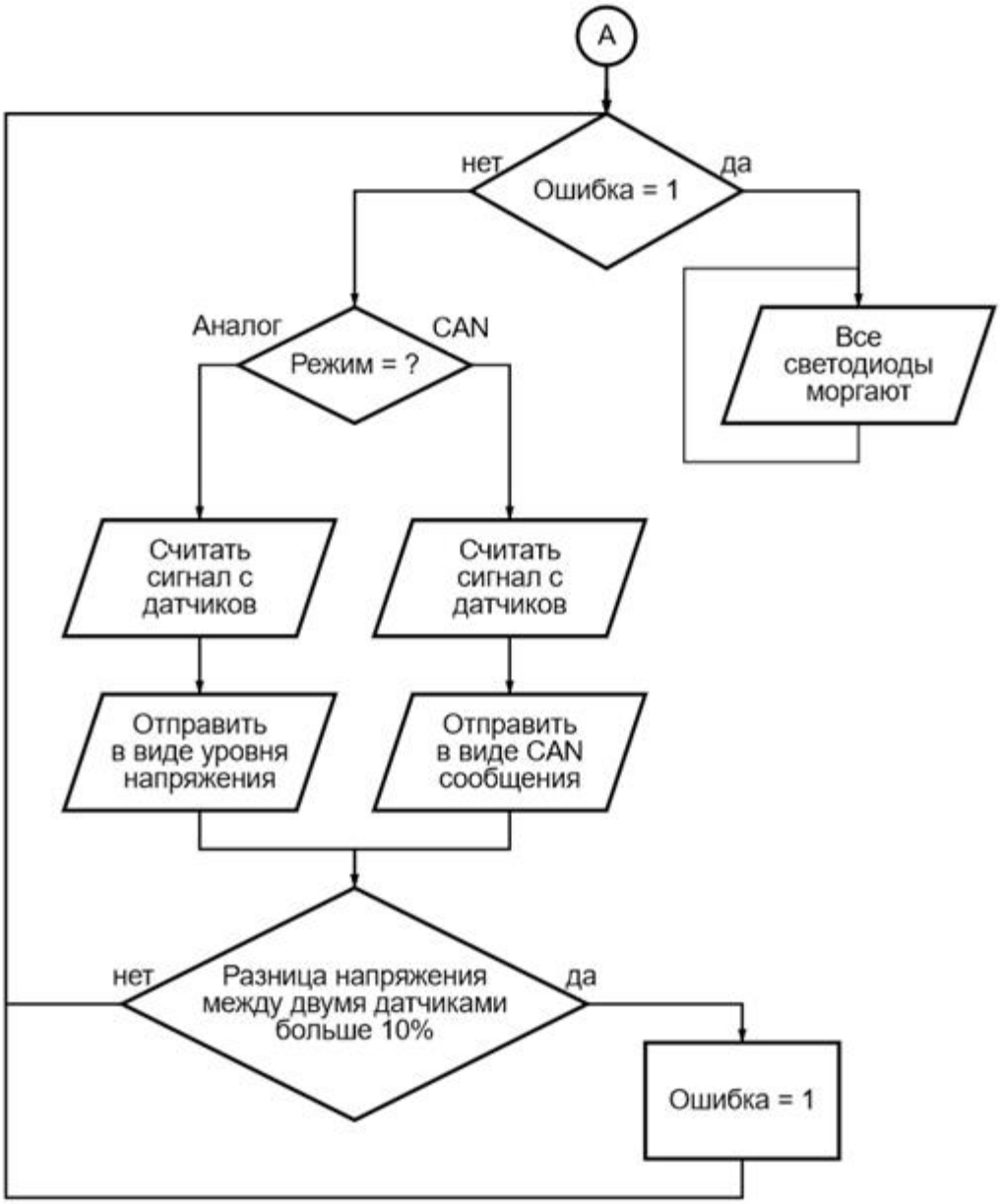

Рисунок 25 – Блок схема алгоритма, продолжение

Составив блок-схему, можно приступать к написанию кода программы будущего устройства. Язык программирования – C++. Далее на рисунке 26 показан код инициализации переменных и библиотек для работы с АЦП и CAN.

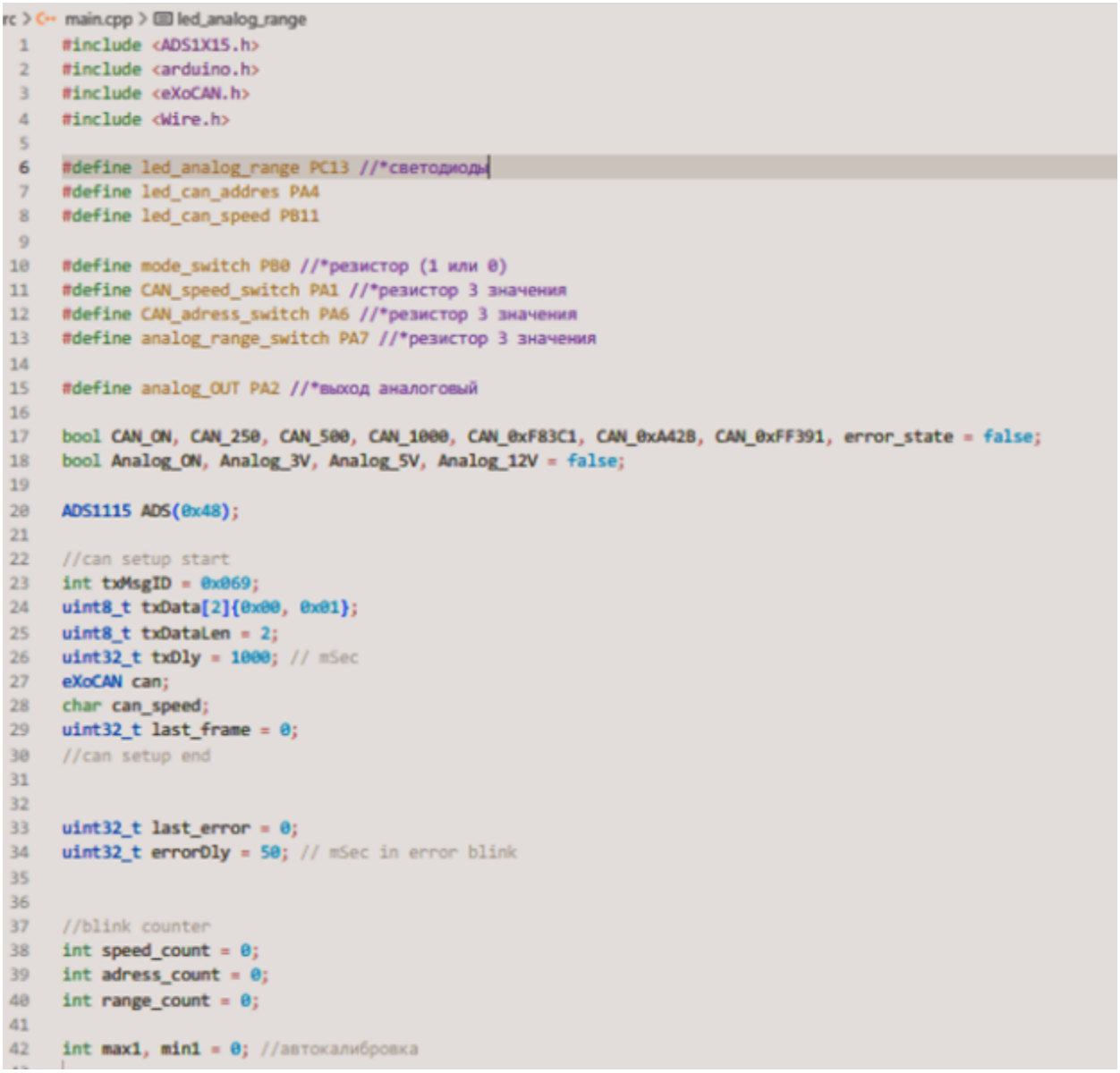

Рисунок 26 – Инициализация переменных

После первичной инициализации разово выполняется код в функции setup(); (рисунок 27). В ней производится настройка последовательного порта, определение входов и выходов микроконтроллера, а также определение режима работы в зависимости от того, запаяна перемычка или нет. Если запаяна, то далее производится настройка CAN подключения, если нет – то настройка АЦП.

```
void setup() {
  Serial.begin(9600);
  pinMode(led_can_speed, OUTPUT);// светодиод
  pinMode(led_can_addres, OUTPUT);
  pinMode(led_analog_range, OUTPUT);
  pinMode(analog_OUT, OUTPUT);
  if (digitalRead(mode_switch) == true){ //если запаяно то CAN
    set_can();
  } else {
    set_analog();
    set_adc();
  }
  blink();
}
```

Рисунок 27 – Функция setup

Настройка АЦП ADS1115 показана на рисунках 28 и 29, в функциях set_analog(); и set_adc();. В них определяется диапазон выходного напряжения в зависимости от установленной перемычки и настраивается библиотека ADS1115.

```
void set_adc() {
  ADS.begin();
  ADS.setGain(0);      // 6.144 volt
  ADS.setDataRate(7);  // fast
  ADS.setMode(0);      // continuous mode
  ADS.readADC(0);      // first read to trigger
  ADS.readADC(1);
}
```

Рисунок 28 – Функция set_adc

```
void set_analog() {
  Analog_ON = true;
  //digitalWrite(led_Analog_ON, HIGH);
  switch (analogRead(analog_range_switch)) {
  case 128:
    Analog_3V = true;
    range_count = 1;
    break;
  case 256:
    Analog_5V = true;
    range_count = 2;
    break;
  case 512:
    Analog_12V = true;
    range_count = 3;
    break;
  default:
    error_state = true;
    break;
  }
}
```

Рисунок 29 – Функция set_analog

Если же перемычка не запаяна, то начинается инициализация CAN. В функции set_can(); (рисунок 30) определяется скорость протокола (250 Кбит/сек, 500 Кбит/сек или 1 Мбит/сек) и адрес сообщения (0xF83C1, 0xFF391 или 0xA42B). Адрес выбирается для исключения конфликта с другими устройствами в CAN сети.

После настройки в функции setup(); единоразово выполняется функция blink();, код на рисунке 31. Она предназначена для информирования с помощью светодиодов о настроенном диапазоне аналогового выхода или о скорости и адресе CAN шины.

Если же в процессе инициализации выявится ошибка, например будут запаяны две перемычки адреса, то начнёт выполняться код в функции error(); (рисунок 32). Светодиоды будут бесконечно моргать, дальнейшая программа выполняться не будет.

```
void set_can() {
  CAN_ON = true;
  switch (analogRead(CAN_speed_switch)) {
  case 128:
    CAN_250 = true;
    speed_count = 1;
    can_speed = BR250K;
    break;
  case 256:
    CAN_500 = true;
    speed_count = 2;
    can_speed = BR500K;
    break;
  case 512:
    CAN_1000 = true;
    speed_count = 3;
    can_speed = BR1M;
    break;
  default:
    error_state = true;
    break;
  }
  switch (analogRead(CAN_adress_switch)) {
  case 128:
    CAN_0xF83C1 = true;
    adress_count = 1;
    txMsgID = 0xF83C1;
    break;
  case 256:
    CAN_0xA42B = true;
    adress_count = 2;
    txMsgID = 0xA42B;
    break;
  case 512:
    CAN_0xFF391 = true;
    adress_count = 3;
    txMsgID = 0xFF391;
    break;
  default:
    error_state = true;
    break;
  }
  if (!error_state){
    can.begin(STD_ID_LEN, can_speed, PORTA_11_12_WIRE_PULLUP);
  }
}
```

Рисунок 30 – Функция set_can

```
void blink(){
  for (int i = 1; i < speed_count; i++) {
    digitalWrite(led_can_speed, true);
    delay(500);
    digitalWrite(led_can_speed, false);
    delay(500);
  }
  for (int i = 1; i < adress_count; i++) {
    digitalWrite(led_can_addres, true);
    delay(500);
    digitalWrite(led_can_addres, false);
    delay(500);
  }
  for (int i = 1; i < range_count; i++) {
    digitalWrite(led_analog_range, true);
    delay(500);
    digitalWrite(led_analog_range, false);
    delay(500);
  }
}
```

Рисунок 31 – Функция blink

```
void error() {
  Serial.println("Error");
  if (millis() - errorDly > last_error){ // tx every txDly
    last_error = millis();
    digitalWrite(led_can_speed, !digitalRead(led_can_speed));
    digitalWrite(led_can_addres, !digitalRead(led_can_addres));
    digitalWrite(led_analog_range, !digitalRead(led_analog_range));
  }
}
```

Рисунок 32 – Функция error

Далее в коде программы находится функция loop();, код на рисунке 33. Эта функция после своего выполнения начинает работу снова, и так до отключения питания. В зависимости от выбранного режима, выполняется считывание сигнала с датчиков (рисунок 34), их сравнение (рисунок 35) и

либо отправка CAN сообщения (рисунок 36), либо установка уровня напряжения в зависимости от выбранного диапазона (рисунок 37).

Если в процессе работы вдруг выявится ошибка в перемычках, то выполнение остановится. Так же остановка работы произойдёт в случае выявления разницы между показаниями датчиков более чем на 10 процентов.

```
void loop() {
  if (error_state) {
    error();
  } else {
    if (CAN_ON) { //can
      send_can(calculate(get_voltage_first(), get_voltage_second()));
    }
    if (Analog_ON) { // analog
      send_analog(calculate(get_voltage_first(), get_voltage_second()));
    }
  }
}
```

Рисунок 33 – Функция loop

```
int get_voltage_first(void) {
  Serial.print(" VOLTAGE_AIN0="); Serial.print(ADS.readADC(0)); Serial.println("V");
    return ADS.readADC(0);
}


int get_voltage_second(void) {
  Serial.print(" VOLTAGE_AIN1="); Serial.print(ADS.readADC(1)); Serial.println("V");
    return ADS.readADC(1);
}
```

Рисунок 34 – Функции get_voltage_first и get_voltage_second

```
int calculate(int left, int right) {
  int ThrottleAct = left - 305 - fabs((double)(right - 305)) - 25 ;
  ThrottleAct = map(ThrottleAct, 0, 305, 0, 1023);
  if (fabs((double)left - right) > 102) {
    error_state = true;
    return error_state;
  }
  if (ThrottleAct > max1) { //*автокалибровка диапазона после запуска
    max1 = ThrottleAct; }
  else if (ThrottleAct < min1) {
    min1 = ThrottleAct; }
  int ThrottleCalculated = map(ThrottleAct, min1, max1, -125, 1200);
  int range = constrain(ThrottleCalculated, 0, 1023);
    return range;
}
```

Рисунок 35 – Функция calculate

```
void send_can(int Value) {
  Serial.println("send can frame");
  txData[0] = 1;
  txData[1] = 1;
  if (millis() - txDly > last_frame){// tx every txDly
    last_frame = millis();
    can.transmit(txMsgID, txData, txDataLen);
  }
  Serial.println("send can frame");
}
```

Рисунок 36 – Функция send_can

```
void send_analog(int Value) {
  Serial.println("send analog");
  analogWrite(analog_OUT, Value);
}
```

Рисунок 37 – Функция send_analog

После компиляции программы, произведена её загрузка в микроконтроллер. В результате израсходовано 90 % памяти ПЗУ, и 11 % памяти ОЗУ, что показано на рисунке 38. Для безошибочной работы свободной ОЗУ достаточно. Конечно, можно оптимизировать код для сокращения занимаемой памяти ПЗУ, например используя вместо analogWrite прямую запись в регистры, однако разрабатываемое устройство не предполагает в дальнейшем расширение функционала, и поэтому оптимизация кода для освобождения памяти не требуется. Полный код программы приведён в приложении А.

Рисунок 38 – Успешная компиляция программы

Вывод по разделу

В данном разделе проанализированы существующие и подходящие для выполнения задания выпускной квалификационной работы среды разработки для плат с микроконтроллерами STM32 и ATMEGA. После выбора среды разработки, изучены способы программирования микроконтроллеров как с помощью внешних программаторов, так и используя внутрисхемные решения, такие как FT232RL.

После подготовительных работ составлена блок-схема алгоритма и определён функционал программы. Далее последовательно написаны все программные функции будущего устройства и, наконец, готовая программа загружена в память микроконтроллера.

4 Разработка печатной платы

Макетные платы отлично подходят для прототипирования схем, но они не так хороши для реального использования создаваемых устройств. Воздействие вибрации может устроить серьёзную проблему случайно отсоединившимся проводом. Лучший способ избежать этого – изготовить печатную плату. Печатная плата представляет собой текстолитовую пластину с медными дорожками, соединяющими компоненты, припаянные непосредственно к плате.

Существует несколько материалов, из которых изготавливают печатные платы. Широкое распространение получили гетинакс и стеклотекстолит. Гетинакс представляет из себя пластик, имеющий слоёную структуру. В качестве слоёв используется бумага, а для пропитки – резольные смолы. Стеклотекстолит имеет схожую структуру, только вместо бумаги используются листы стеклоткани. Стеклотекстолитовые платы обладают большей прочностью, имеют более широкий температурный диапазон и изготовление таких плат возможно заказать на производстве, для того чтобы исключить возможность ошибки при изготовлении платы с помощью фоторезиста в домашних условиях.

В данной работе разработан прототип печатной платы устройства (рисунки 39–42). Для этого использовалась среда проектирования EasyEDA, «работающая как в виде отдельного Windows приложения, так и в веб-браузере» [7]. Так же в программе имеется простой интерфейс для создания Gerber файла. Этот файл содержит в себе координаты всех элементов и дорожек, а также границы платы.

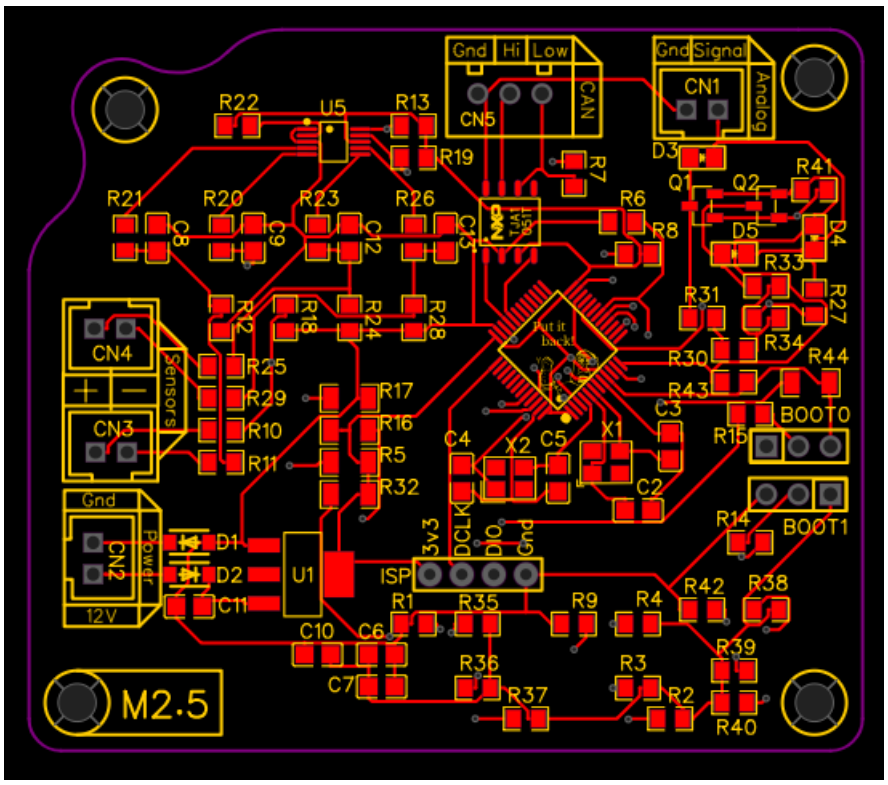

Рисунок 39 – Верхний слой печатной платы

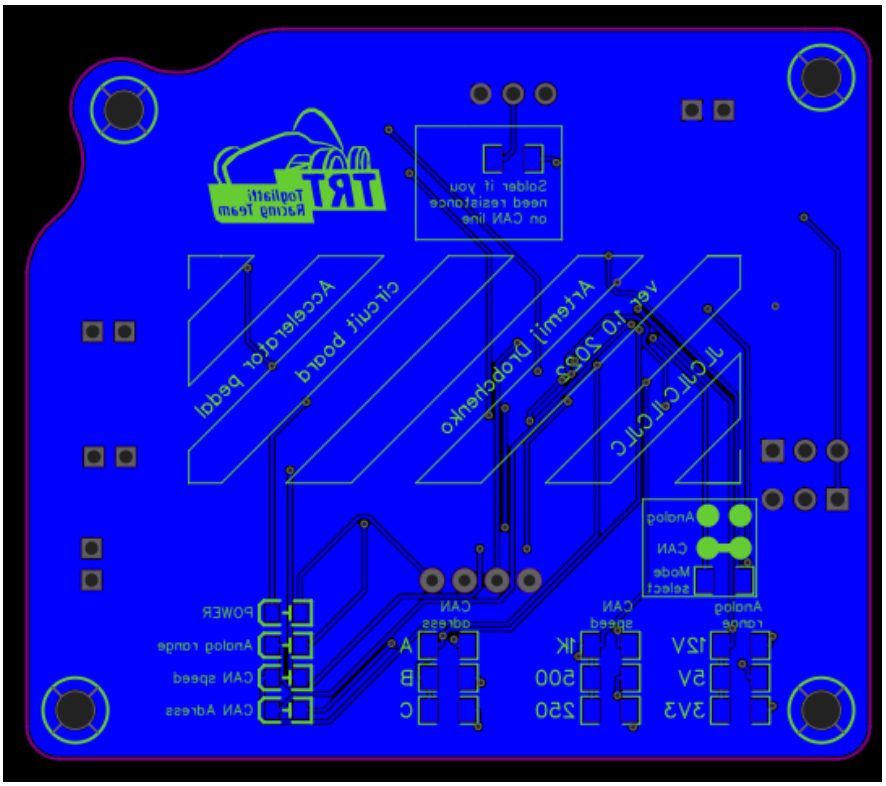

Рисунок 40 – Проектирование печатной платы

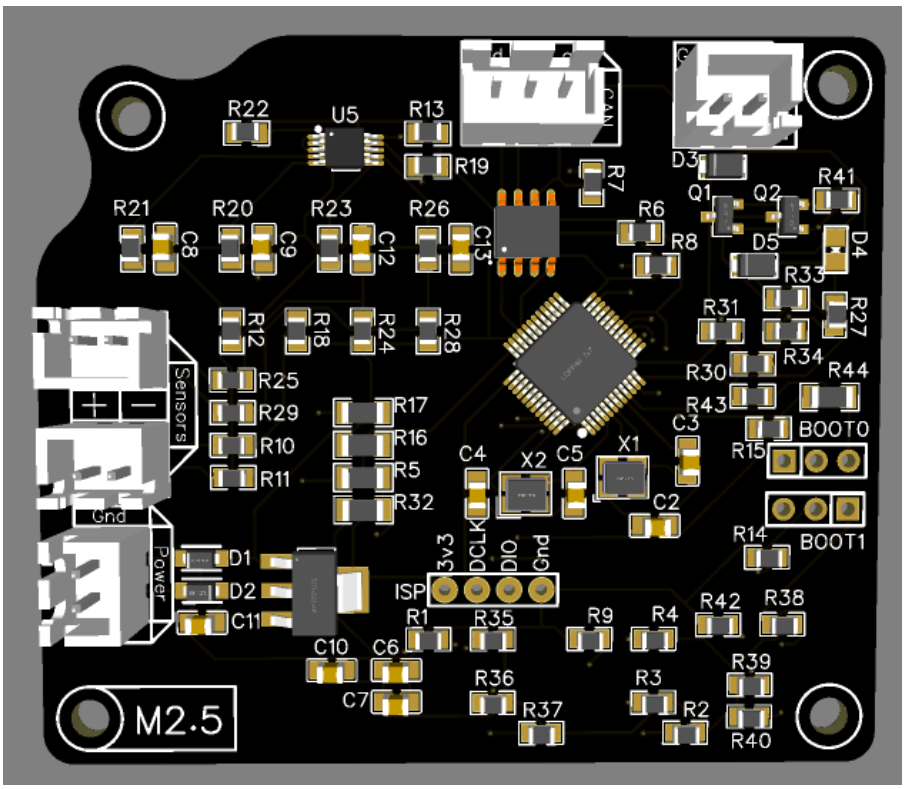

Рисунок 41 – 3D модель печатной платы, передняя сторона

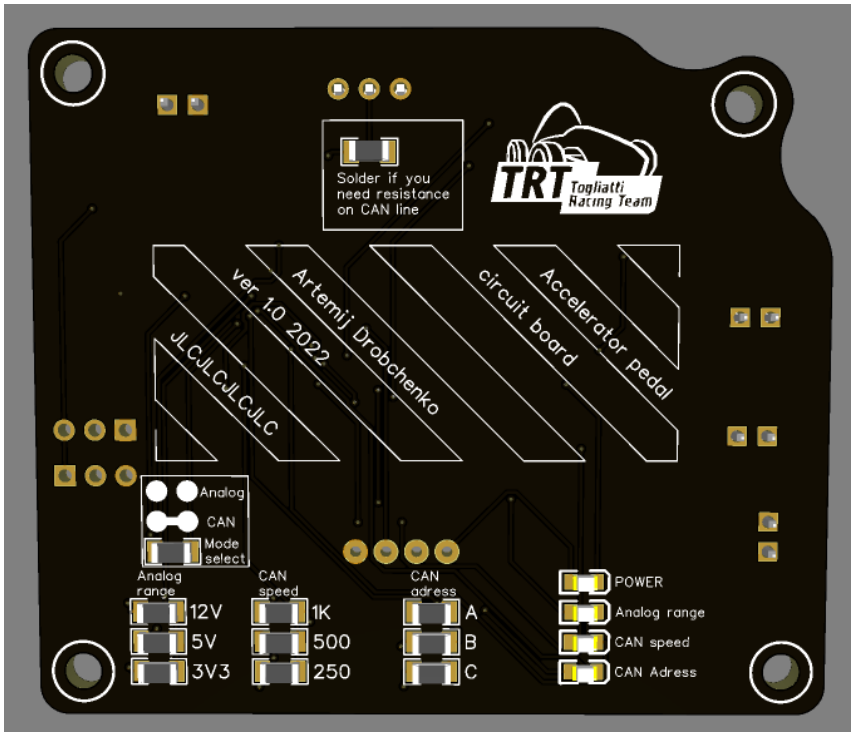

Рисунок 42 – 3D модель печатной платы**,** обратная сторона

В углах печатной платы расположены крепёжные отверстия под винт с резьбой M2.5. Для подключения внешних устройств использовались разъёмы JST XH (рисунок 43) [22]. Они выбраны из-за компактности и наличия ключа, не позволяющего перепутать полярность подключения.

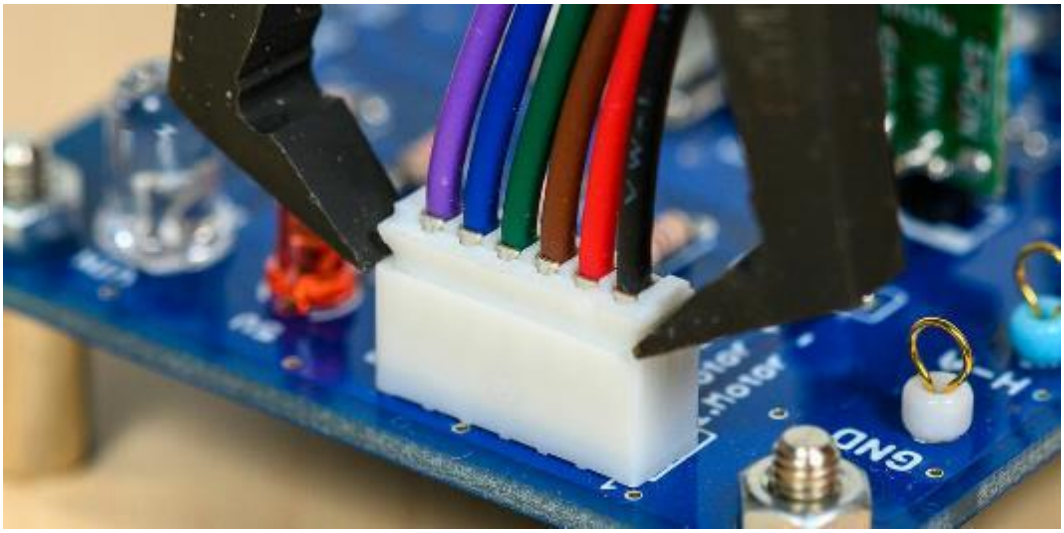

Рисунок 43 – разъём JST XH на плате

На обратной стороне платы, рядом с запаиваемыми перемычками и светодиодами предусмотрено место под наклейку с информацией (рисунок 44). На ней размещено название устройства и CAN адреса. В дальнейшем это позволяет изменять программу и, не переделывая шелкографию печатной платы указывать новые адреса или режимы работы на этом стикере (рисунок 45). Размер области под наклейку – 43 мм x 17,5 мм. Для изготовления использован Adobe Photoshop (рисунок 46).

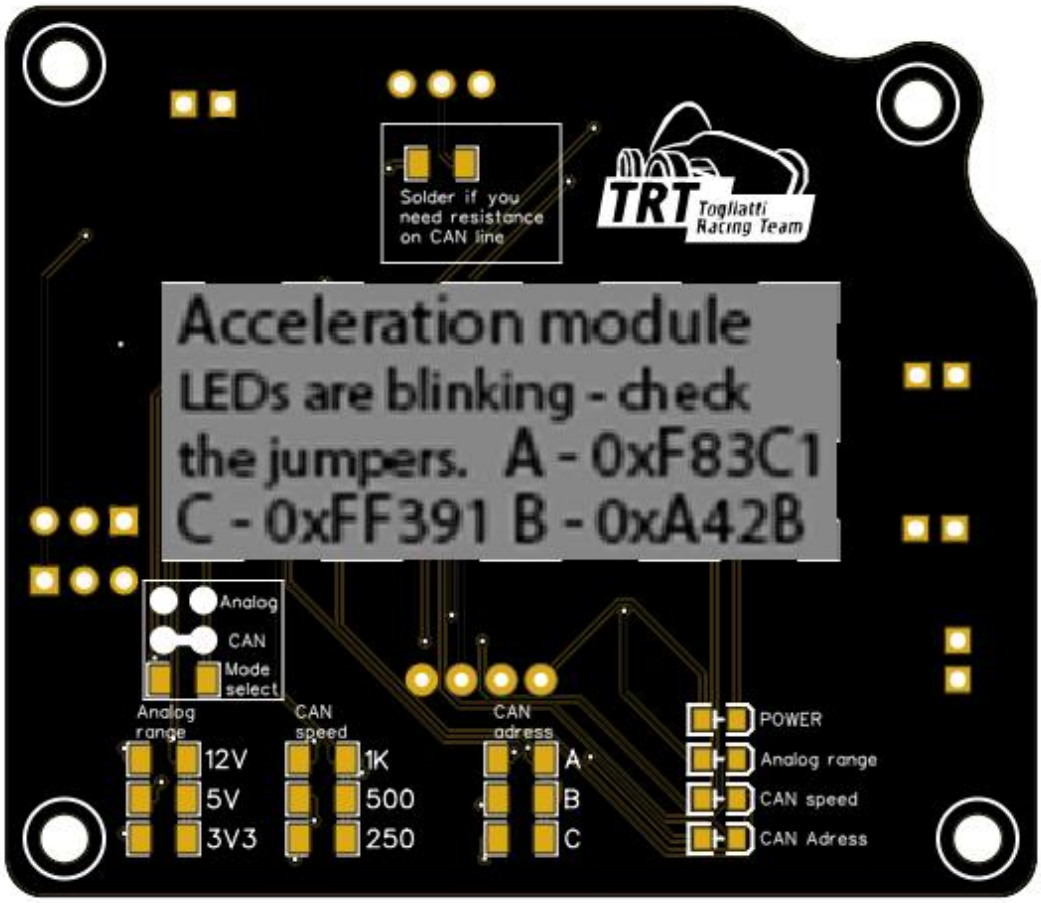

Рисунок 44 – Плата с наклейкой

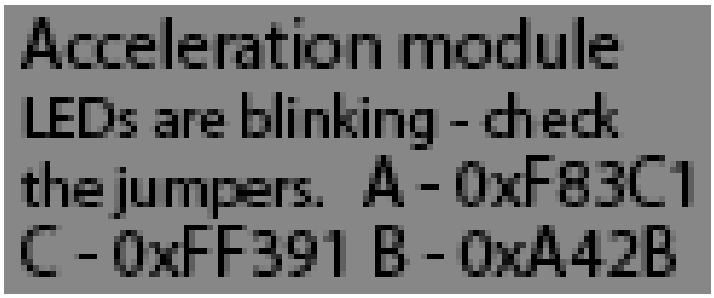

Рисунок 45 – Наклейка на плату

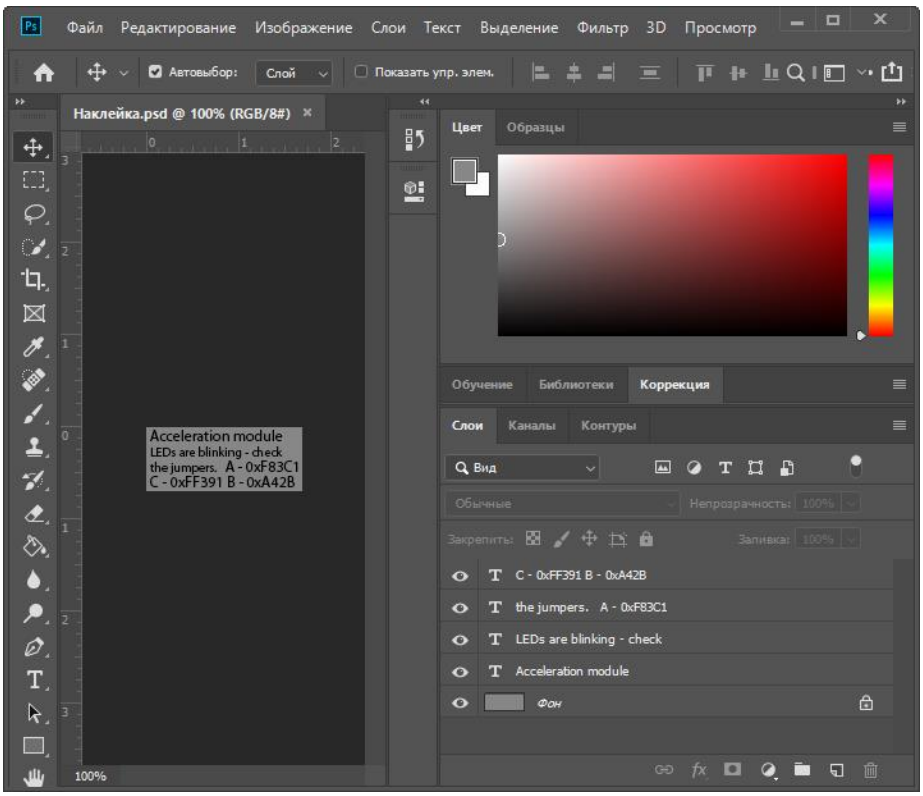

Рисунок 46 – Интерфейс программы Adobe Photoshop

Для конфигурирования устройства предусмотрены запаиваемые перемычки. На плате расположены: перемычка подключения сопротивления на CAN шине (рисунок 47), перемычка выбора режима работы CAN/Аналоговый (рисунок 48), перемычки выбора скорости и адреса CAN, а также выбор диапазона аналогового выхода (рисунок 49).

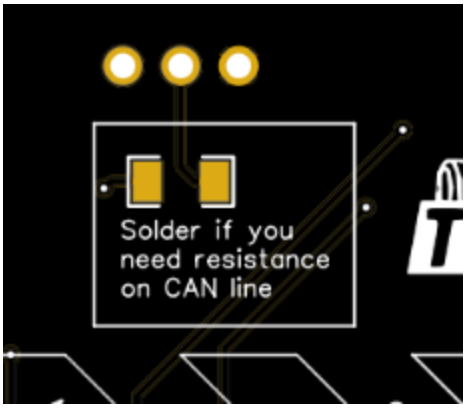

Рисунок 47 – Перемычка для подключения сопротивления на CAN шине

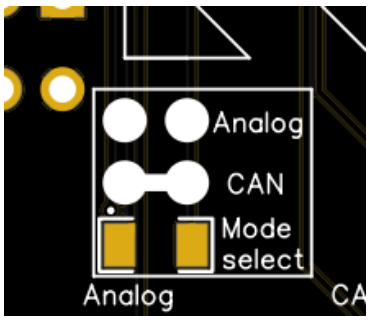

Рисунок 48 – Перемычка выбора режима работы устройства

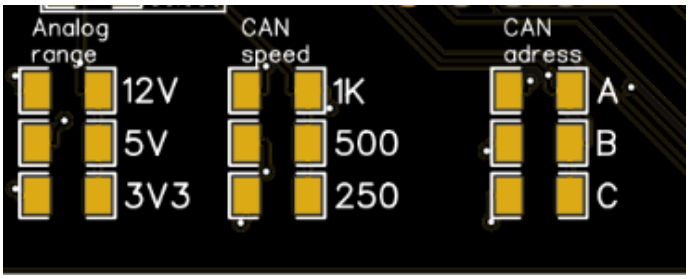

Рисунок 49 – Перемычки для конфигурирования устройства

Так же на оборотной стороне платы присутствуют индикаторные светодиоды (рисунок 50). Их функционал описан в таблице 5.

Таблица 5 – Режим работы светодиодов

Название	Нормальный режим работы	Режим инициализации после подачи питания	Ошибка
POWER	Всегда включён		
Analog range	Всегда выключен	Если режим аналоговый, то: 1 включение – 3 В, 2 включения – 5 В, 3 включения – 12 В	Моргают с одинаковой периодичностью
CAN speed		Если режим CAN, то: 1 включение – 1 Мбит/с, 2 включения – 500 кбит/с, 3 включения – 250 кбит/с	
CAN address		Если режим CAN, то: 1 включение – адрес A, 2 включения – адрес B, 3 включения – адрес C	

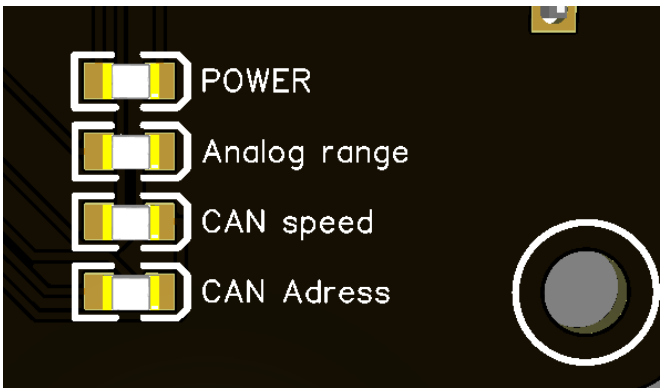

Рисунок 50 – Информационные светодиоды

В дальнейшем производство плат заказано на JLCPCB.com. «JLCPCB имеет собственные заводы с полностью автоматизированными производственными линиями, обеспечивающие высокое качество и точность изготовления. Продукция сертифицирована по ISO 9001:2015, ISO 14001:2015 и IPC-6012E» [11].

4.1 Разработка корпуса устройства

В качестве способа изготовления корпуса выбрана 3D печать в виду дешевизны изготовления штучного изделия, а также из-за наличия доступа к оборудованию и низкой стоимости печати.

Существует несколько популярных видов пластика для 3D печати (рисунки 51–53), рассмотрим их основные свойства, достоинства и недостатки и определим наиболее оптимальный материал для изготовления корпуса устройства. В таблице 6 представлены рассматриваемые материалы.

Таблица 6 – Сравнение материалов для 3D печати

Название	Температура плавления прутка, ℃	Температура подогрева стола, ℃	Преимущества	Недостатки
PETG	200 - 220	50 - 60	Простота печати, прочность	Требовательность к температурному режиму печати
ABS	220 - 250	80 - 100	Высокая прочность, термостойкость	Токсичность при печати, необходимость наличия закрытой камеры для печати, сильная усадка
PLA	190 - 210	60	Простота печати, экологичность, отсутствие усадки	Низкая температура плавления

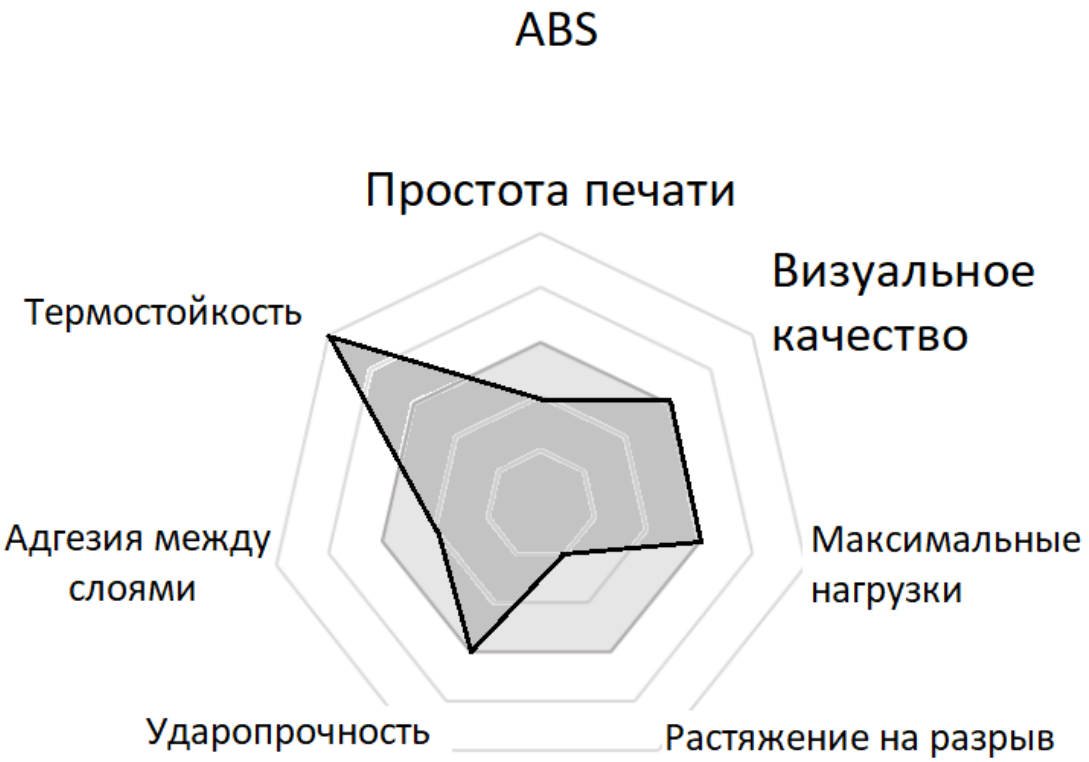

Рисунок 51 – Диаграмма характеристик ABS пластика

Рисунок 52 – Диаграмма характеристик PETG пластика

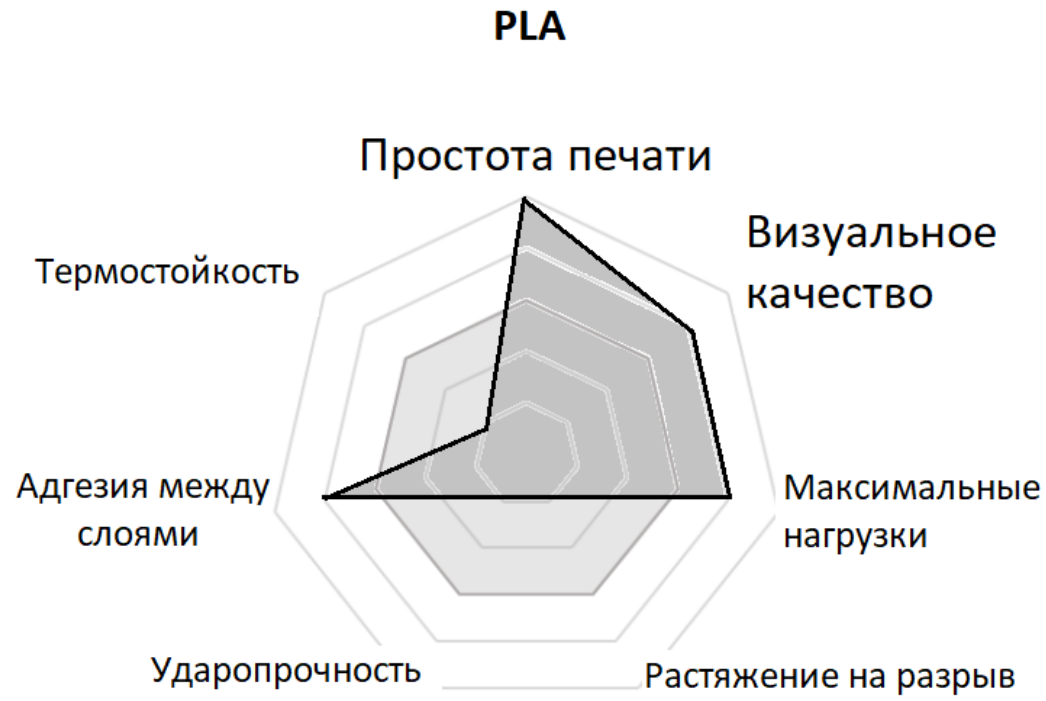

Рисунок 53 – Диаграмма характеристик PLA пластика

Таким образом, для изготовления корпуса устройства решено использовать пластик PLA в виду простоты печати в домашних условиях и экологичности. Так же стоит отметить, что температурные условия эксплуатации ниже, чем температура плавления, так что корпус сохранит свою первозданную форму.

Корпус устройства спроектирован в среде Tinker CAD. «Tinker CAD – это веб приложение, включающее в себя редактор, симулятор электронных схем и трёхмерную среду проектирования» [18] (рисунок 54).

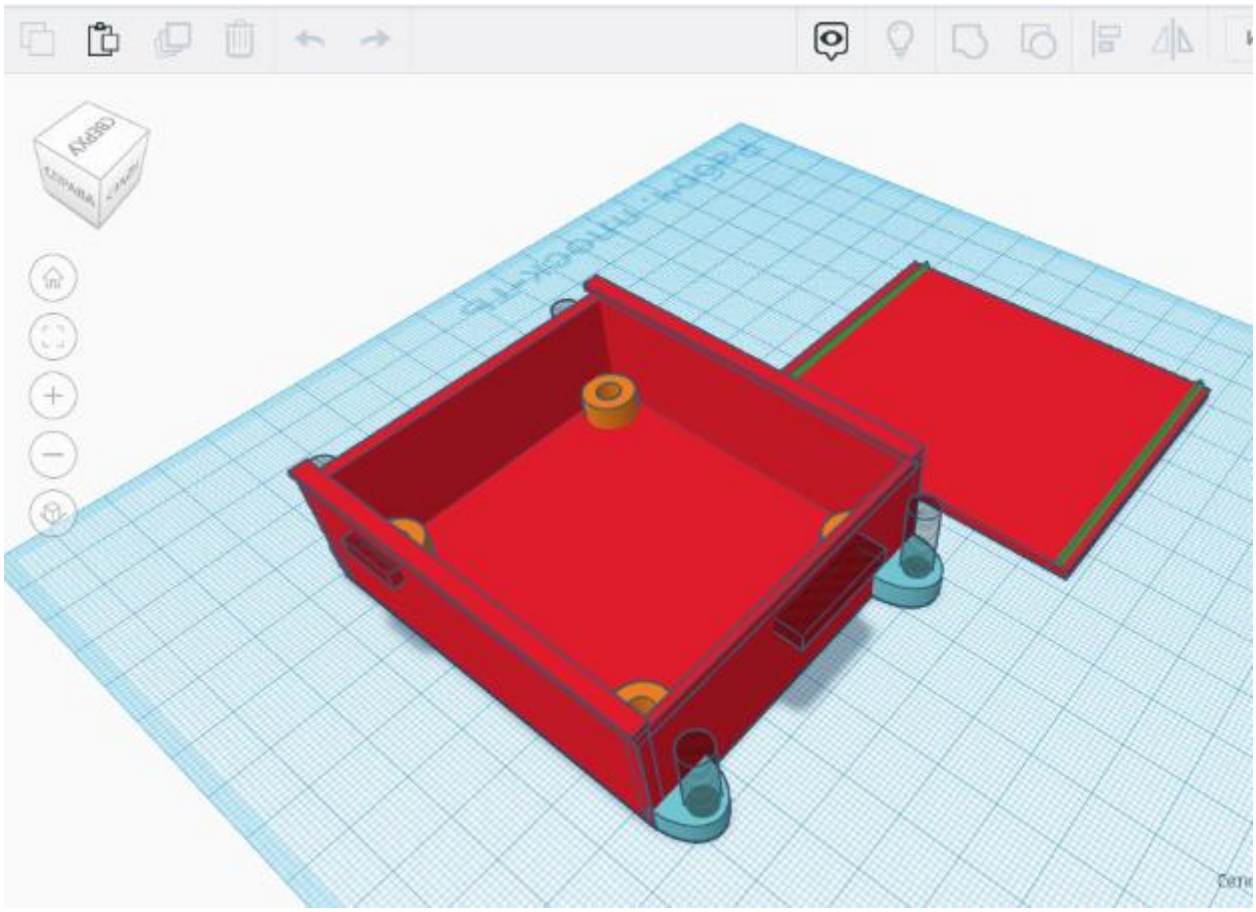

Рисунок 54 – Сервис для 3D моделирования Tinker CAD

Для фиксации платы в корпусе используются резьбовые вставки в корпус (рисунки 55, 56). Они представляют из себя латунный цилиндр с насечками на наружной поверхности и с резьбой на внутренней. Для их установки необходимо воспользоваться паяльником – установить вставку над

отверстием, предназначенным для неё, затем держа паяльник перпендикулярно корпусу нагреть вставку, и тем самым расплавить пластик. После того, как вставка займёт своё место, необходимо дать пластику остыть и только после этого производить установку платы в корпус.

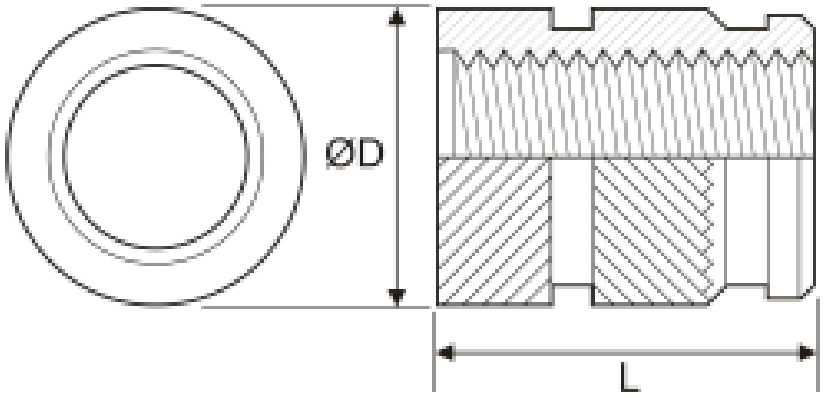

Рисунок 55 – Чертёж резьбовой вставки

Рисунок 56 – Латунные резьбовые вставки

Для подготовки файла 3D модели корпуса к печати требуется воспользоваться специальной программой – слайсером. Программное обеспечение Ultimaker Cura, показано на рисунке 57, от производителей 3D принтеров отлично справляется с этой задачей и к тому же распространяется бесплатно.

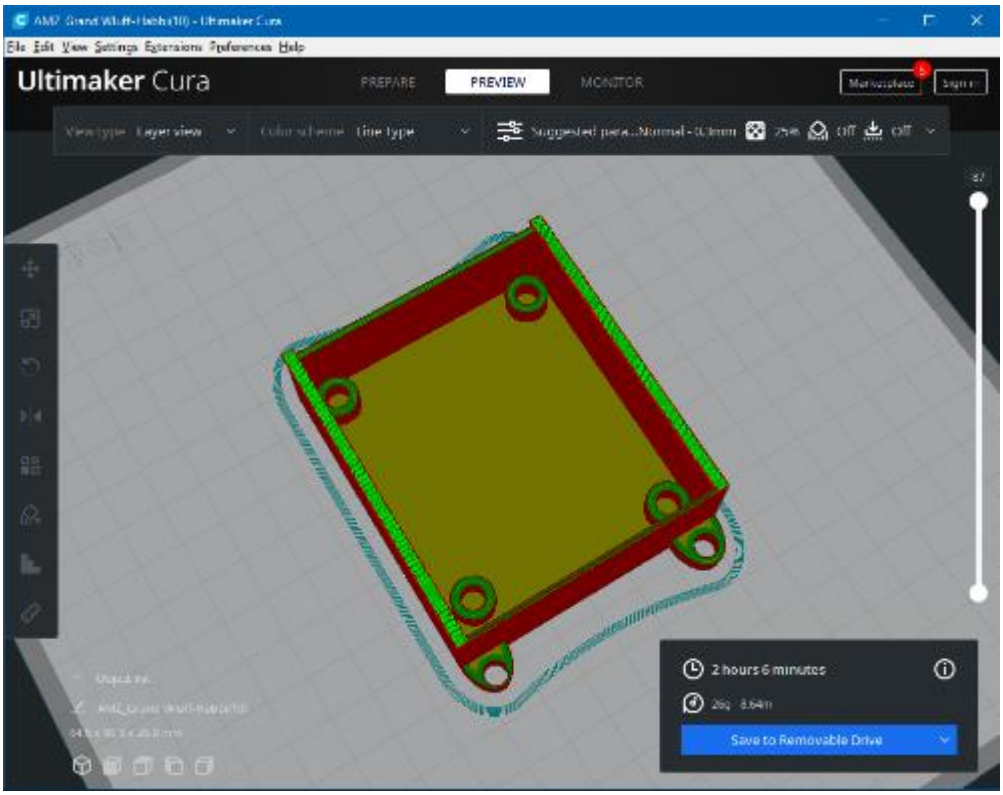

Рисунок 57 – Окно программы Ultimaker Cura для 3D печати

Вывод по разделу

В качестве основы устройства принято решение использовать печатную плату, которую планируется заказать на производстве в Китае. Подобные платы выполнены в строгом соответствии с проектом и количество погрешностей и отклонений настолько мало, что ими можно пренебречь. Для изготовления корпуса решено использовать технологию 3D печати с помощью PLA пластика. Для крепления платы в корпусе предусмотрены специальные резьбовые вставки.

5 Экспериментальные исследования работы устройства

Для проверки работоспособности, устройство собрано на беспаечной макетной плате из модулей, содержащих описанные микросхемы (рисунок 58).

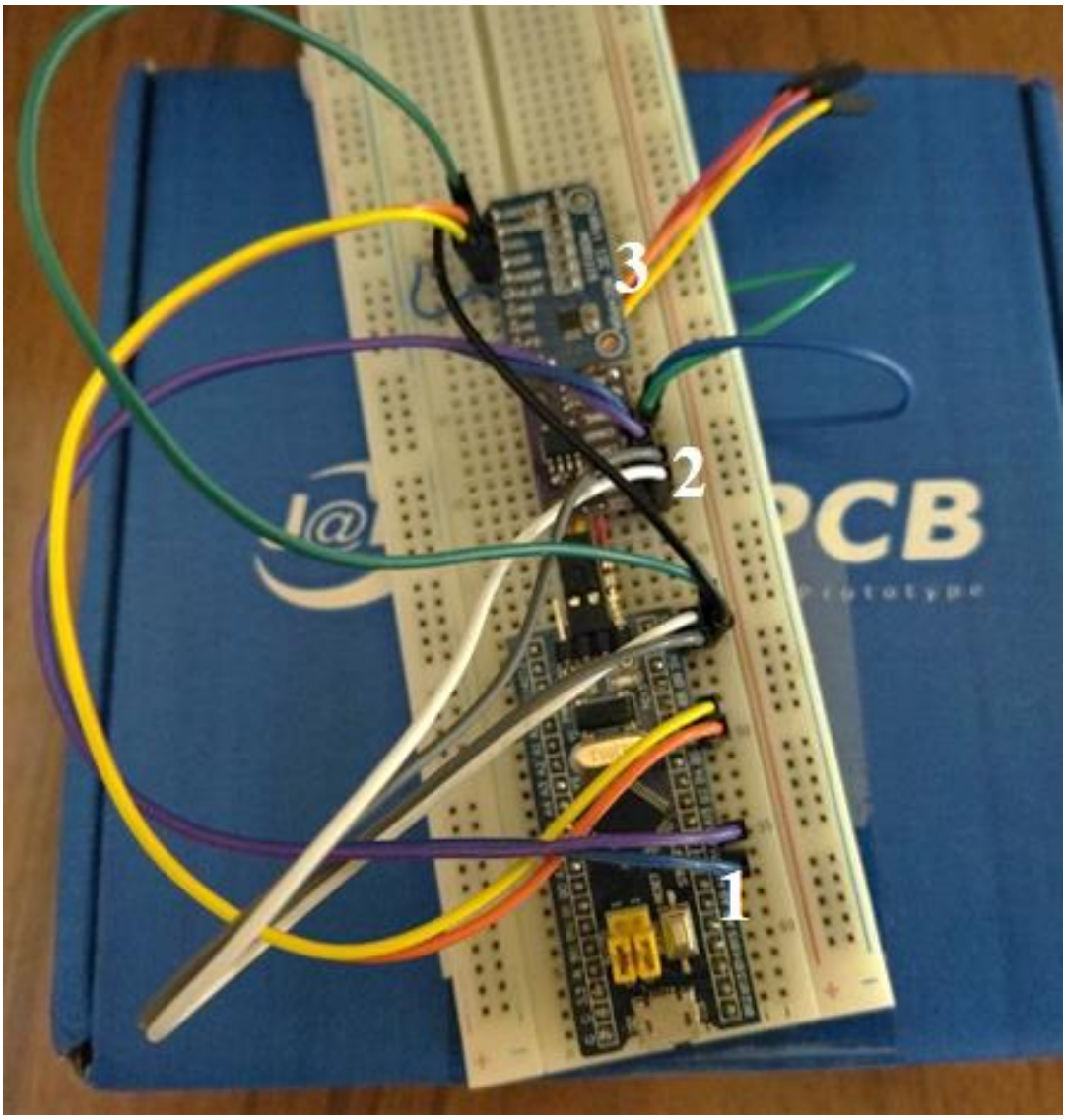

Рисунок 58 – Фотография прототипа устройства, 1 – STM32, 2 – TJA1051, 3 – ADS1115

После сборки произведена прошивка микроконтроллера STM32 с помощью программатора ST-Link. Правильность подключения устройства к компьютеру через программатор проверена загрузкой программы Blink.

После компиляции и прошивки, светодиод на плате blue pill начал моргать с заданной последовательностью. Удостоверившись в правильности сборки, в микроконтроллер загружена написанная программа для работы с датчиками педали.

Для проверки работы, к устройству подключены два датчика вращения. Для получения информации о выводном сигнале, к компьютеру подключён USB CAN adapter производства компании ORION, изображённый на рисунке 59.

Рисунок 59 – USB CAN adapter для проверки работы устройства

После подключения проверен режим ошибки – программно установлены две взаимоисключающие перемычки и подан сигнал с датчиков вращения с разницей больше 10%. В обоих случаях устройство прекращало работу и начинало сигнализировать об ошибке с помощью встроенных светодиодов и прекращало передачу сигнала.

Аналоговый выходной уровень проверялся с помощью осциллографа Hantek DSO-5062B. Выбор режима осуществлялся изменением переменной в программном коде. Результаты эксперимента занесены в таблицу 7.

Таблица 7 – Результаты экспериментального запуска устройства

Уровень нажатия педали, %	Программный уровень	Аналоговый уровень 3 В, В	Аналоговый уровень 5 В, В	Аналоговый уровень 12 В, В	CAN сообщение
0	0	0	0	0	0x000
10	102	0,321	0,511	1,2	0x66
20	205	0,614	1,044	2,4	0xCD
30	307	0,912	1,521	3,6	0x133
40	410	1,253	2,132	4,8	0x19A
50	512	1,584	2,501	6	0x200
60	614	1,861	3,030	7,2	0x266
70	727	2,231	3,500	8,4	0x2D7
80	830	2,522	4,100	9,6	0x33E
90	942	2,831	4,506	10,8	0x3AE
100	1024	3,300	5,090	12	0x3FF

По полученным данным можно построить график и увидеть погрешность в различных режимах. Для наглядности, на рисунке 60 значения при режиме работы 3 В умножены на 300, 5 В на 200, 12 В на 100.

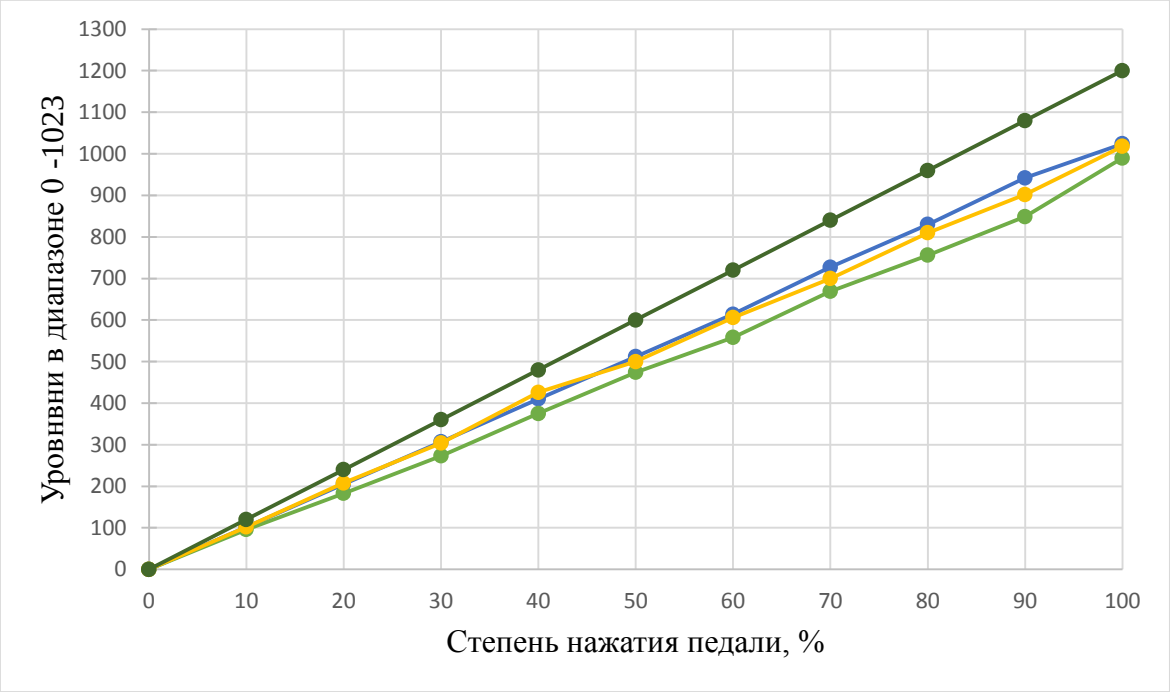

Рисунок 60 – График соответствия аналоговых выходных уровней и эталона, темно зелёный – 12В, светло зелёный – 5В, жёлтый – 3В, синий – программный уровень (эталон)

После экспериментального подтверждения работоспособности решения, производство печатной платы было заказано в компании JLCPCB, расположенной в Китае. Получившиеся платы до сборки показаны на рисунках 61 и 62.

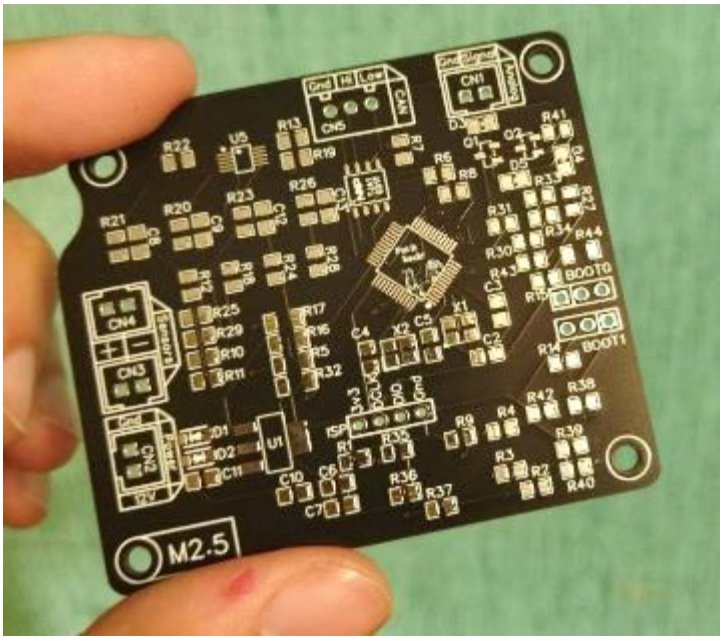

Рисунок 61 – Изготовленная плата, лицевая сторона

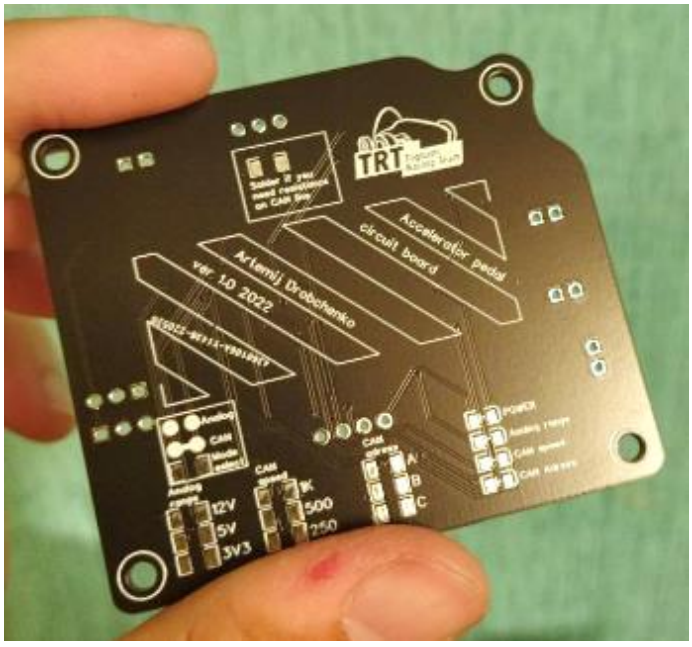

Рисунок 62 – Изготовленная плата, обратная сторона

Далее произведена сборка платы с использованием паяльника SH-72 и жала BC2 (рисунки 63 и 64). Использовался припой ПОС-61 и паяльная паста. Для промывки платы после пайки использовался изопропиловый спирт.

Рисунок 63 – Паяльник SH72

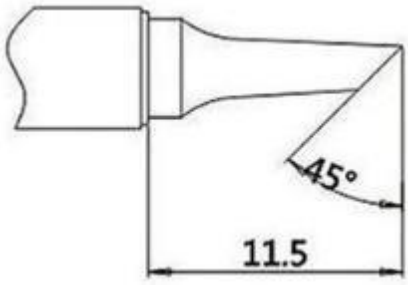

Рисунок 64 – Схематичное изображение формы жала BC2

На рисунке 65 показана лицевая сторона собранной платы, а на рисунке 66 – обратная. Далее распечатан корпус устройства на 3D принтере Anycubic Mega Zero 2.0 [5]. Процесс печати показан на рисунках 67 и 68. После печати, в корпус установлены резьбовые вставки, печатная плата зафиксирована с помощью винтов под шестигранный ключ. Собранное устройство показано на рисунке 69.

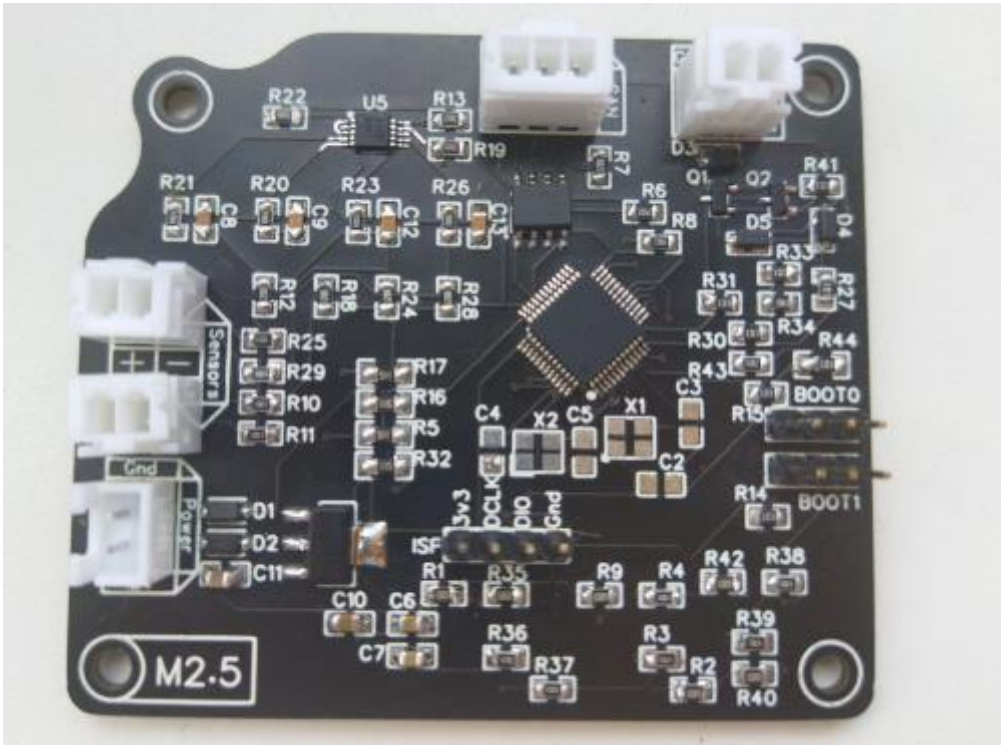

Рисунок 65 – Лицевая сторона платы

Рисунок 66 – Обратная сторона платы

Рисунок 67 – Процесс печати корпуса

Рисунок 68 – Экран состояния принтера в процессе печати

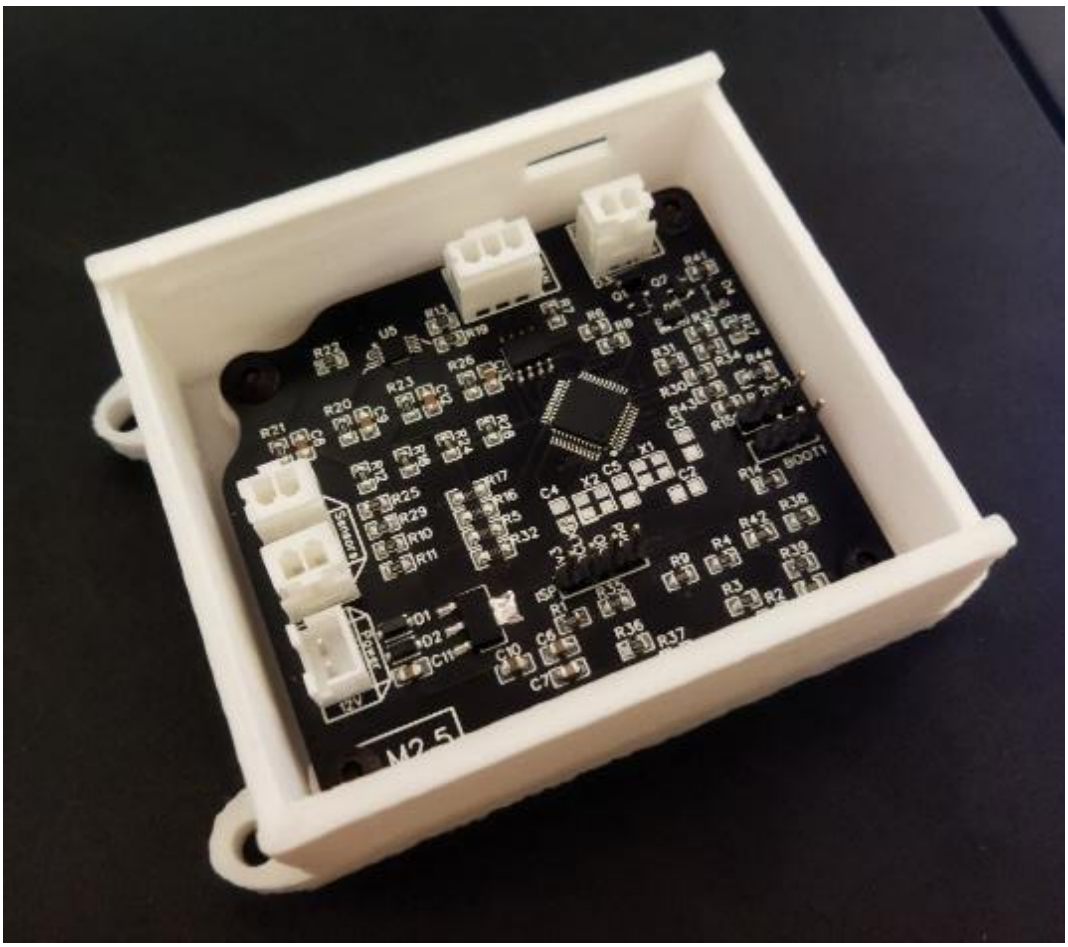

Рисунок 69 – Собранное устройство

Вывод по разделу

В результате экспериментального исследования работы устройства определено, что отклонение от эталонного значения крайне мало и им можно пренебречь. Так же, экспериментально проверены все режимы работы устройства, в том числе режим ошибки по входным значениям и по неправильной конфигурации.

Собранная плата установлена в напечатанный корпус, устройство после сборки протестировано. Модуль акселератора для электрического гоночного болида полностью отвечает поставленным требованиям и целям выпускной квалификационной работы и является устройством, полностью готовым к установке на автомобили.

Заключение

В ходе выполнения выпускной квалификационной работы разработан модуль акселератора для электрического гоночного болида.

На начальном этапе определены цель ВКР, а также задачи для её выполнения. Далее исследованы способы и принципы управления скоростью автомобилей как с двигателями внутреннего сгорания, так и электродвигателями.

Далее проанализированы доступные для покупки на рынке решения на предмет соответствия техническим требованиям поставленной задачи. В результате анализа выявлена очевидная необходимость разработки устройства, имеющего возможность подключения к блокам управления, работающих как с аналоговыми педалями, так и с подключаемыми по CAN шине, в виду отсутствия подобных универсальных решений. Также в результате анализа подтверждена актуальность работы.

Для решения поставленных задач выбран современный и технологичный микроконтроллер STM32, а также надёжная в использовании и простая в использовании периферия в виде CAN трансмиттера TJA1051 и АЦП ADS1115.

На следующем этапе проанализированы среды разработки для плат с микроконтроллером STM32. После выбора среды разработки, изучены способы программирования микроконтроллеров с помощью внешних или внутрисхемные программаторов.

После составлена блок-схема алгоритма и определён функционал программы. Далее, последовательно написаны все программные функции будущего устройства и, наконец, готовая программа загружена в память микроконтроллера.

В качестве основы устройства принято решение использовать печатную плату, которую планируется заказать на производстве в Китае. Подобные платы выполнены в строгом соответствии с проектом и

количество погрешностей и отклонений настолько мало, что ими можно пренебречь.

Для изготовления корпуса использована технология 3D печати с помощью PLA пластика. Для крепления платы в корпусе предусмотрены специальные резьбовые вставки.

В результате экспериментального исследования работы устройства определено, что отклонение от эталонного значения крайне мало и им можно пренебречь. Также, экспериментально проверены все режимы работы устройства, в том числе режим ошибки по входным значениям и по неправильной конфигурации.

Разработку модуля акселератора для электрического гоночного болида можно считать завершённой, поскольку модуль отвечает всем выдвинутым требованиям и является полностью готовым устройством.

Список используемых источников

1. Модуль педальный ВАЗ Лада 1118,2190 электронная E-GAS ОАО АВТОВАЗ 11183-1108500-01: АВТОВАЗ. Продажа оптом и в розницу. [Электронный ресурс] URL: https://autopiter.ru/goods/11183110850001/avtovaz/id105056005 (дата обращения 06.02.2022)

2. Регламент соревнований Formula Student : FSG [Электронный ресурс] URL: https://www.formulastudent.de/fileadmin/user_upload/all/2022/rules/FS-Rules_2022_v1.0.pdf (дата обращения 05.11.2021).

3. A professional collaborative platform for embedded development · PlatformIO | platformio.org [Электронный ресурс] URL: https://platformio.org/ (дата обращения 18.03.2022).

4. ADS1115 data sheet, product information and support | TI.com [Электронный ресурс] URL: https://www.ti.com/product/ADS1115 (дата обращения 01.04.2022).

5. Anycubic Mega Zero 2.0 FDM 3D Printer – ANYCUBIC 3D Printing | anycubic.com [Электронный ресурс] URL: https://www.anycubic.com/products/mega-zero-2-0-fdm-3d-printer (дата обращения 11.04.2022).

6. Auto Accelerator Pedal Sensor Throttle Pedal Accelerator For Electric Car - Buy Accelerator For Electric Car, Throttle Pedal, Auto Accelerator Pedal Sensor Product on : Alibaba.com [Электронный ресурс] URL: https://www.alibaba.com/product-detail/auto-accelerator-pedal-sensor-throttle-pedal_1600152478651.html (дата обращения 12.01.2022).

7. EasyEDA - Online PCB design & circuit simulator | easyeda.com [Электронный ресурс] URL: https://easyeda.com/ (дата обращения 22.04.2022).

8. FT232RL - FTDI | ftdichip.com [Электронный ресурс] URL: https://ftdichip.com/products/ft232rl/ (дата обращения 22.04.2022).

9. GM OEM Pedal Position Sensor CAN Bus : Three Pedals [Электронный ресурс] URL: https://threepedals.com/products/electrical/pedal-switch-harness/pedal-position/gm-oem-pedal-position-sensor-can-bus/ (дата обращения 06.02.2022).

10. LM1117 SNOS412O – FEBRUARY 2000 – REVISED JUNE 2020 Product data sheet : Texas Instruments [Электронный ресурс] URL: https://www.ti.com/lit/ds/symlink/lm1117.pdf (дата обращения 01.04.2022).

11. PCB Prototype & PCB Fabrication Manufacturer - JLCPCB | JLCPCB.com [Электронный ресурс] URL: https://jlcpcb.com/ (дата обращения 17.05.2022).

12. Pedal Throttle For Electric Vehicle / Car - Buy Throttle For Electric Motor, Pedal Throttle, Pedal Throttle For Electric Vehicle Product on : Alibaba.com [Электронный ресурс] URL: https://www.alibaba.com/product-detail/0-5V-Pedal-Throttle-for-Electric_60767699765.html (дата обращения 26.12.2021).

13. SAE J1939 – Wikipedia | wikipedia.org [Электронный ресурс] URL: https://en.wikipedia.org/wiki/SAE_J1939 (дата обращения 03.04.2022).

14. ST-LINK/V2 - ST-LINK/V2 in-circuit debugger/programmer for STM8 and STM32 - STMicroelectronics | st.com [Электронный ресурс] URL: https://www.st.com/en/development-tools/st-link-v2.html (дата обращения 03.04.2022).

15. ST-LINK/V2 in-circuit debugger/programmer for STM8 and STM32 microcontrollers – data brief | st.com [Электронный ресурс] URL: https://www.st.com/resource/en/data_brief/st-link-slsh-v2.pdf (дата обращения 12.05.2022).

16. STM32F103C6T6 Microcontroller: 72MHz, 48-LQFP, Pinout and Datasheet : Utmel electronic [Электронный ресурс] URL: https://www.utmel.com/components/stm32f103c6t6-microcontroller-72mhz-48-lqfp-pinout-and-datasheet?id=1355 (дата обращения 01.04.2022).

17. STM32F103C8T6 Blue Pill Pinout, Peripherals, Programming and Features : Microcontrollerslab [Электронный ресурс] URL: https://microcontrollerslab.com/stm32f103c8t6-blue-pill-pinout-peripherals-programming-features/#STM32F103C8T6_Blue_Pill_Pinout (дата обращения 15.03.2022).

18. Tinkercad - From mind to design in minutes | tinkercad.com [Электронный ресурс] URL: https://www.tinkercad.com/ (дата обращения 01.02.2022).

19. TJA1051 High-speed CAN transceiverRev. 9 – 28 November 2017, Product data sheet : NXP® Semiconductors Official Site [Электронный ресурс] URL: https://www.nxp.com/docs/en/data-sheet/TJA1051.pdf (дата обращения 13.03.2022).

20. Visual Studio Code - Code Editing. Redefined | visualstudio.com [Электронный ресурс] URL: https://code.visualstudio.com/ (дата обращения 04.04.2022).

21. What Is Drive-By-Wire Technology? | lifewire.com [Электронный ресурс] URL: https://www.lifewire.com/what-is-drive-by-wire-534825 (дата обращения 01.02.2022).

22. XH Connector | JST Sales America | jst.com [Электронный ресурс] URL: https://www.jst.com/products/crimp-style-connectors-wire-to-board-type/xh-connector/ (дата обращения 28.04.2022).

Приложение А

Листинг кода программы

```
#include <ADS1X15.h>
#include <arduino.h>
#include <eXoCAN.h>
#include <Wire.h>

#define led_analog_range PC13 //*светодиоды
#define led_can_addres PA4
#define led_can_speed PB11

#define mode_switch PB0 //*резистор (1 или 0)
#define CAN_speed_switch PA1 //*резистор 3 значения
#define CAN_adress_switch PA6 //*резистор 3 значения
#define analog_range_switch PA7 //*резистор 3 значения

#define analog_OUT PA2 //*выход аналоговый

bool CAN_ON, CAN_250, CAN_500, CAN_1000, CAN_0xF83C1,
CAN_0xA42B, CAN_0xFF391, error_state = false;
bool Analog_ON, Analog_3V, Analog_5V, Analog_12V = false;

ADS1115 ADS(0x48);

//can setup start
int txMsgID = 0x069;
uint8_t txData[2]{0x00, 0x01};
uint8_t txDataLen = 2;
uint32_t txDly = 1000; // mSec
```

```
eXoCAN can;
char can_speed;
uint32_t last_frame = 0;
//can setup end


uint32_t last_error = 0;
uint32_t errorDly = 50; // mSec in error blink


//blink counter
int speed_count = 0;
int adress_count = 0;
int range_count = 0;

int max1, min1 = 0; //автокалибровка

void set_can() {
  CAN_ON = true;
  switch (analogRead(CAN_speed_switch)) {
  case 128:
    CAN_250 = true;
    speed_count = 1;
    can_speed = BR250K;
    break;
  case 256:
    CAN_500 = true;
    speed_count = 2;
    can_speed = BR500K;
```

```
  break;
case 512:
  CAN_1000 = true;
  speed_count = 3;
  can_speed = BR1M;
  break;
default:
  error_state = true;
  break;
}
switch (analogRead(CAN_adress_switch)) {
case 128:
  CAN_0xF83C1 = true;
  adress_count = 1;
  txMsgID = 0xF83C1;
  break;
case 256:
  CAN_0xA42B = true;
  adress_count = 2;
  txMsgID = 0xA42B;
  break;
case 512:
  CAN_0xFF391 = true;
  adress_count = 3;
  txMsgID = 0xFF391;
  break;
default:
  error_state = true;
  break;
```

```
 }
  if (!error_state){
    can.begin(STD_ID_LEN, can_speed, PORTA_11_12_WIRE_PULLUP);
  }
}


void send_can(int Value) {
  Serial.println("send can frame");
  txData[0] = 1;
  txData[1] = 1;
  if (millis() - txDly > last_frame){// tx every txDly
    last_frame = millis();
    can.transmit(txMsgID, txData, txDataLen);
  }
  Serial.println("send can frame");
}


void set_analog() {
  Analog_ON = true;
  //digitalWrite(led_Analog_ON, HIGH);
  switch (analogRead(analog_range_switch)) {
  case 128:
    Analog_3V = true;
    range_count = 1;
    break;
  case 256:
    Analog_5V = true;
```

```
  range_count = 2;
  break;
  case 512:
    Analog_12V = true;
    range_count = 3;
    break;
  default:
    error_state = true;
    break;
  }
}


void send_analog(int Value) {
  Serial.println("send analog");
  analogWrite(analog_OUT, Value);
}


void set_adc() {
  ADS.begin();
  ADS.setGain(0);      // 6.144 volt
  ADS.setDataRate(7);  // fast
  ADS.setMode(0);      // continuous mode
  ADS.readADC(0);      // first read to trigger
  ADS.readADC(1);
}
```

```
int get_voltage_first1(void) {
  Serial.print("          VOLTAGE_AIN0=");          Serial.print(ADS.readADC(0));
Serial.println("V");
    return ADS.readADC(0);
}


int get_voltage_first(void) {
  Serial.print("          VOLTAGE_AIN0=");          Serial.print(ADS.readADC(0));
Serial.println("V");
    return ADS.readADC(0);
}


int get_voltage_second(void) {
  Serial.print("          VOLTAGE_AIN1=");          Serial.print(ADS.readADC(1));
Serial.println("V");
    return ADS.readADC(1);
}


void error() {
  Serial.println("Error");
  if (millis() - errorDly > last_error){ // tx every txDly
    last_error = millis();
    digitalWrite(led_can_speed, !digitalRead(led_can_speed));
    digitalWrite(led_can_addres, !digitalRead(led_can_addres));
    digitalWrite(led_analog_range, !digitalRead(led_analog_range));
```

```
  }
}


void blink(){
  for (int i = 1; i < speed_count; i++) {
    digitalWrite(led_can_speed, true);
    delay(500);
    digitalWrite(led_can_speed, false);
    delay(500);
  }
  for (int i = 1; i < adress_count; i++) {
    digitalWrite(led_can_addres, true);
    delay(500);
    digitalWrite(led_can_addres, false);
    delay(500);
  }
  for (int i = 1; i < range_count; i++) {
    digitalWrite(led_analog_range, true);
    delay(500);
    digitalWrite(led_analog_range, false);
    delay(500);
  }
}


void setup() {
  Serial.begin(9600);
  pinMode(led_can_speed, OUTPUT);// светодиод
```

```
 pinMode(led_can_addres, OUTPUT);
  pinMode(led_analog_range, OUTPUT);
  pinMode(analog_OUT, OUTPUT);
  if (digitalRead(mode_switch) == true){ //если запаяно то CAN, иначе ANALOG
    set_can();
  } else {
    set_analog();
    set_adc();}
  blink();}


int calculate(int left, int right) {
  int ThrottleAct = left - 305 - fabs((double)(right - 305)) - 25 ; //* 305 offset to zero, -25 offset to compensate diff
  ThrottleAct = map(ThrottleAct, 0, 305, 0, 1023);
  if (fabs((double)left - right) > 102) {
    error_state = true;
    return error_state;
  }
  if (ThrottleAct > max1) { //*автокалибровка диапазона после запуска
    max1 = ThrottleAct; }
  else if (ThrottleAct < min1) {
    min1 = ThrottleAct; }
  int ThrottleCalculated = map(ThrottleAct, min1, max1, -125, 1200);
  int range = constrain(ThrottleCalculated, 0, 1023);
    return range;
}
```

```
void loop() {
  if (error_state) {
    error();
  } else {
    if (CAN_ON) { //can
      send_can(calculate(get_voltage_first(), get_voltage_second()));
    }
    if (Analog_ON) { // analog
      send_analog(calculate(get_voltage_first(), get_voltage_second()));
    }
  }
}
```