Министерство образования Пензенской области

ГБОУ ПО «Академический лицей № 14»

Проект

для участия в VII открытом региональном конкурсе
исследовательских и проектных работ школьников
«Высший пилотаж – Пенза» 2025
на тему:

«Полетный контроллер на базе Arduino для радиоуправляемой модели самолета»

Выполнил:
Карагулян Андрей Дмитриевич

Научный руководитель:
Трофимов Юрий Александрович

Пенза, 2024

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

В современном мире использование новых технологий стало неотъемлемой частью жизни каждого человека. А технологии развиваются с невероятной скоростью, и одним из самых популярных направлений сегодня является создание беспилотных летательных аппаратов (БПЛА). Радиоуправляемые самолеты с полетными контроллерами на базе Arduino представляют собой интересное и перспективное направление для исследований и разработок. В данной работе рассматриваются основные аспекты создания полетного контроллера на базе Arduino для радиоуправляемых моделей самолетов.

Цель работы: создание полетного контроллера на базе Arduino для радиоуправляемой модели самолета, способного обеспечивать стабильное и безопасное управление полетом, а также предоставлять расширенные функции, такие как удержание высоты, автопилот и возможность подключения дополнительных датчиков и модулей.

Задачи работы

1. Изучить основы электроники, программирования на языке Arduino и работы с датчиками.
2. Выбрать и закупить необходимые компоненты (Arduino, датчики, моторы, сервоприводы и т. д.).
3. Разработать программное обеспечение для контроллера, включая обработку сигналов от датчиков и управление моторами.
4. Собрать все компоненты контроллера на печатной плате;
5. Протестировать контроллер в лабораторных условиях, проверить работоспособность и устранить возможные ошибки.
6. Провести летные испытания контроллера на радиоуправляемой модели самолета, скорректировать настройки и алгоритмы работы.
7. Произвести расчет и оценку стоимости полетного контроллера.

Актуальность работы обусловлена следующими факторами.

1. Низкая стоимость комплектующих по сравнению с другими аналогами. Использование Arduino и доступных компонентов позволяет значительно снизить затраты на создание полетного контроллера, делая его доступным для широкого круга любителей и профессионалов.
2. Простота и удобство использования. Arduino имеет открытый исходный код и множество библиотек, упрощающих разработку и настройку полетного контроллера. Это позволяет быстро освоить основы работы с ним и вносить необходимые изменения в программу.
3. Гибкость и расширяемость. Arduino предоставляет возможность подключения различных датчиков, модулей и устройств, что позволяет адаптировать полетный контроллер под конкретные потребности и задачи.
4. Легкость модернизации. Если потребуется внести изменения или дополнения в работу полетного контроллера, это можно сделать без значительных затрат, так как большинство компонентов доступны и недороги.

5. Возможность обучения и развития навыков. Работа над проектом позволит изучить основы электроники, программирования и механики, а также развить навыки решения технических задач и проектирования сложных систем.

Методы исследования: анализ литературы, моделирование, статистико-математический, экспериментальный, наблюдение.

Практическая значимость работы заключается в следующем:

1. Снижение стоимости разработки и производства радиоуправляемых моделей самолетов. Использование полетного контроллера на базе Arduino позволяет значительно сократить затраты на аппаратное обеспечение и разработку программного обеспечения.
2. Улучшение управляемости и стабильности полета радиоуправляемых моделей самолетов. Полетный контроллер обеспечивает более точное управление и стабилизацию полета, что улучшает качество и безопасность полетов.
3. Расширение возможностей радиоуправляемых моделей самолетов. С помощью полетного контроллера на базе Arduino можно добавить различные функции, такие как удержание высоты, автопилот, выполнение маневров и другие.
4. Обучение и развитие навыков в области электроники, программирования и авиамоделирования. Проект позволяет студентам и любителям авиации изучать основы электроники, программирования и управления летательными аппаратами.

Целевая аудитория проекта: любители радиоуправляемых моделей самолетов; школьники, студенты и молодые специалисты, изучающие основы робототехники и беспилотных летательных аппаратов; разработчики и инженеры, работающие в области создания и модернизации беспилотных авиационных систем; предприниматели и инвесторы, заинтересованные в развитии рынка беспилотных авиационных систем и создании новых продуктов и услуг на их основе.

ЧАСТЬ 1. ТЕОРЕТИЧЕСКИЕ ОСНОВЫ СОЗДАНИЯ БПЛА НА ОСНОВЕ ПОЛЕТНЫХ КОНТРОЛЛЕРОВ

1.1. Обзор существующих БПЛА

Существуют следующие основные типы беспилотных летательных аппаратов (БПЛА)[5].

1. Аэростатические БПЛА: аэростаты с оболочкой, заполненной газом или нагретым воздухом, используются для долгосрочного наблюдения, связи и метеорологии.
2. Реактивные БПЛА: управляемые ракеты, которые передвигаются за счет реактивной тяги двигателей.
3. БПЛА самолетного типа с фиксированным крылом: способны летать благодаря подъемной силе, создаваемой аэродинамической формой крыла.
4. БПЛА вертолетного типа: однороторные аппараты с двумя несущими винтами или парой несущего и рулевого винтов.
5. Мультикоптерные (мультироторные) БПЛА: имеют независимые 2 и более несущих винта, уравновешивающих реактивные моменты.
6. Гибридные БПЛА: конвертопланы с поворотными или фиксированными винтами, которые при взлете и посадке работают как подъемные, а при горизонтальном полете — как тянущие.

1.2. Особенности построения современных БПЛА

Беспилотные летательные аппараты отличаются по назначению, но составные части у них схожи [10].

1. Корпус (фюзеляж).
2. Источник энергии (аккумулятор, солнечные батареи, топливные элементы).
3. Двигательная установка (на основе различных типов двигателей: электрический, внутреннего сгорания, воздушно-реактивный).
4. Навигационная система (датчики: гироскоп, акселерометр, альтиметр, ГЛОНАСС/GPS).
5. Система управления (контроллер, бортовой компьютер, автопилот) [6].
6. Система связи (радиоприемник, ретранслятор, радиопередатчик).
7. Съемочная аппаратура (камера, тепловизор).
8. Ряд других приборов, подбираемых под конкретный БПЛА и его задачи.

Особенности построения современных БПЛА включают.

1. Концептуальное проектирование, включающее разработку платформы-носителя агрегатов и конструкции бортового радиоэлектронного оборудования.
2. Важность ранней стадии проектирования, так как ошибки на этом этапе могут привести к серьезным проблемам.
3. Анализ и сравнение лучших образцов БПЛА.

4. Отказ от копирования и заимствования решений у других разработчиков, а вместо этого оценка эффективности решений по ключевым критериям.
5. Разработка собственных уникальных технологий и подходов для достижения превосходства над существующими БПЛА.

1.3. Основные полетные контроллеры для БПЛА

Основным компонентом электронной схемы любого БПЛА является контроллер полета (полетный контроллер – ПК). БПЛА сам по себе никогда не будет лететь ровно, и управлять им без контроллера полета будет невозможно. Существует множество различных систем управления полетом БПЛА – от систем автопилота с поддержкой GPS, управляемых по двусторонней телеметрической связи, до базовых систем стабилизации с использованием радиоуправляемого оборудования любительского уровня [4]. Однако их стоимость может быть достаточно большой и составить значительную часть бюджета.

Поэтому при построении БПЛА на первый план выходит уменьшение стоимости контроллера полета.

Чаще всего в наше время используют бюджетные полетные контроллеры (так называемые стеки). Их задача помогать в управлении БПЛА. Для этого к стекам подключают датчики и сервоприводы для управления, и многое другое.

Обычно полетные контроллеры отличаются только по объему памяти и количеству логических выходов.

Для примера рассмотрим часто используемые полетные контроллеры в БПЛА самолетного и мультироторного типа.

ПК для БПЛА самолетного типа:

- SpeedyBee F405 WING MINI (примерная стоимость – 5000 руб.)
- Ardupilot APM 2.8 (примерная стоимость – 5500 руб.)

ПК для БПЛА мультироторного типа:

- SpeedyBee F405 V3 50A BLS 30x30 (примерная стоимость – 6000 руб.)
- JHEMCU GF30F722-ICM F722 Baro OSD 5V 10V (примерная стоимость – 4000 руб.)

Рассмотрим каждый из представленных полетных контроллеров подробнее.

SpeedyBee F405 WING MINI

Полетный контроллер SpeedyBee F405 WING MINI разработан для управления различными типами беспилотных летательных аппаратов. Оснащенный мощным микроконтроллером STM32F405 с тактовой частотой 168 МГц и 1 МБ флэш-памяти, он обеспечивает быструю и надежную обработку данных (рис. 1).

Контроллер включает в себя интегрированный IMU, состоящий из гироскопа и акселерометра ICM-42688-P, что позволяет точно отслеживать ориентацию и движение аппарата. Дополнительно, барометр SPL006-001 обеспечивает измерение высоты с высокой точностью, что особенно важно для стабильного полета.

Для удобства пользователя предусмотрен экранный чип AT7456E, который позволяет получать визуальную информацию о состоянии системы. Наличие слота для карт памяти microSD позволяет записывать данные полета в режиме Blackbox, что дает возможность анализировать производительность и поведение БПЛА после завершения полета.

Моторчик SpeedyBee для квадрокоптера также поддерживает множество интерфейсов для подключения различных датчиков и устройств. Один I2C-канал может использоваться для подключения магнитометра или цифрового датчика воздушной скорости, а четыре аналоговых входа (ADC) позволяют отслеживать напряжение батареи, ток, RSSI и аналоговую скорость воздуха.

Компактный регулятор для fpv-системы предлагает 12 выходов PWM, включая девять контактных разъемов и дополнительные паяльные площадки. Поддержка приемника ELRS/CRSF через UART1 и вход SBUS через UART2-RX делает систему гибкой и совместимой с современными радиоуправляющими системами.

Индикаторы состояния, представленные тремя светодиодами, предоставляют информацию о текущем статусе контроллера, включая состояние питания. Контроллер поддерживает прошивку FC INAV SpeedyBeeF405WING по умолчанию.

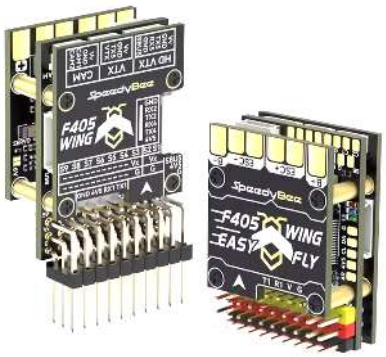


Рисунок 1. Полетный контроллер SpeedyBee F405 WING MINI

Ardupilot APM 2.8

Полетный контроллер ArduPilot Mega является полноценным решением для БПЛА, который позволяет помимо радиоуправляемого дистанционного пилотирования осуществлять автоматическое управление по заранее созданному маршруту, т.е. полет по точкам, а так же обладает возможностью двухсторонней передачей телеметрических данных с борта на наземную станцию (телефон, планшет, ноутбук, DIY) и ведение журнала во встроенную память (рис. 2).

Он основан на автопилоте APM 2.8, разрабатываемым сообществом DIY Drones и базирующийся на open-source проекте, позволяющий превратить любой аппарат в автономное средство и эффективно использовать его не только в развлекательных целях, но и для выполнения профессиональных проектов.

Основные характеристики:

- Акселерометр/гироскоп Invensense 6 DoF MPU-6000

- Высокоточный барометр MS5611-01BA03
- Память для автоматического логирования 4MB Dataflash chip
- Off-board GPS, LEA-6H module with Compass (распаян на плате)
- Чипы Atmel ATMEGA2560 и ATMEGA32U-2 for processing and usb functions respectively

Рисунок 2. Полетный контроллер Ardupilot APM 2.8

SpeedyBee F405 V3 50A BLS 30x30

SpeedyBee F405 V3 BLS 50A 30x30 – это комплект полетного контроллера и регулятора скорости (стек) для квадрокоптеров и других мультироторных систем (рис. 3).

Основные особенности:

- Настройка через Bluetooth: Используйте приложение SpeedyBee на смартфоне для беспроводной настройки полетного контроллера и регулятора скорости.
- Мощный регулятор скорости: 4-в-1 ESC с реальным выходом 50A, поддерживает моторы 6S.
- Встроенный индикатор заряда батареи: 4-уровневый светодиодный индикатор для контроля уровня заряда.
- Поддержка 8 моторов: Подходит для конфигураций X8, Y6 и летающих крыльев.
- Беспроводное изменение направления вращения моторов: Настройка через приложение SpeedyBee.
- 4 комплекта светодиодных выходов с простым переключением режимов.
- Улучшенная защита: TVS-диод и конденсатор 1000 мкФ с низким ESR для защиты от скачков напряжения.

Дополнительные функции:

- Встроенный барометр
- Слот для microSD карты (до 4 ГБ)
- Разъем для DJI Air Unit
- Отдельные BEC 9В 2А и 5В 2А4 UART порта
- Поддержка GPS

Этот стек предлагает широкие возможности настройки и высокую производительность для создания мощных и функциональных мультироторных систем.

Рисунок 3. Полетный контроллер SpeedyBee F405 V3 50A BLS 30x30

JHEMCU GF30F722-ICM F722 Baro OSD 5V 10V

Спецификация

- Производитель : JHEMCU
- Модель: GF30F722-ICM
- Размер: 36x36мм
- Масса: 8.8г
- Стек: 30.5x30.5
- MCU: F722
- IMU: ICM-42688-P
- OSD: да
- Барометр: да
- Blackbox: 16Мб
- Питание: 3-8S
- BEC: 5V 2.5A + 10V 2A
- UART: 6
- I2C: 1
- Контакты Buzzer: да
- USB: Type-C
- LED: да
- Вольтметр: да
- Прошивка: JHEF7DUAL

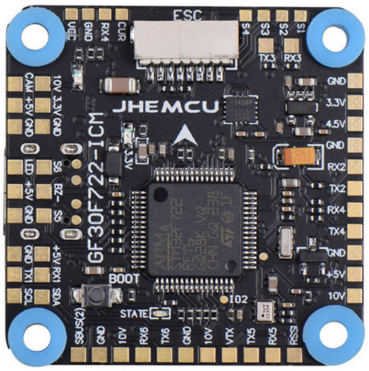


Рисунок 4. Полетный контроллер JHEMCU GF30F722-ICM F722 Baro OSD 5V 10V

1.4. Arduino как универсальное средство для программирования контроллеров

Arduino – это инструмент для проектирования электронных устройств более плотно взаимодействующих с окружающей физической средой, чем стандартные персональные компьютеры, которые фактически не выходят за рамки виртуальности. Данный микроконтроллер применяется для создания электронных устройств с возможностью приема сигналов от различных цифровых и аналоговых датчиков, которые могут быть подключены к нему, и управления различными исполнительными устройствами. Проекты устройств, основанные на Arduino, могут работать самостоятельно или взаимодействовать с программным обеспечением на компьютере [3].

Сфера использования данной платформы на современном этапе практически безгранична. С помощью нее можно спроектировать множество различных систем, которые смогут помочь человеку оптимизировать затраты природных ресурсов и уменьшить стоимость потребляемых услуг. Так на базе микроконтроллеров работает система «умный дом», автоматические вентиляторы и светофоры, мини метеостанции, квадрокоптеры и т.д [1].

Основные преимущества платформы Arduino: открытая схема оборудования, открытый код программы, простая и удобная среда программирования, возможность функционирования на различных видах систем, программирование, подключение и питание может осуществляться одним USB-кабелем, приемлемая цена оборудования [2].

В нашем проекте мы использовали плату Arduino Nano, а также плату расширения, датчик BMP-180, акселерометр GY-521 и два сервопривода SG90.

ЧАСТЬ 2. СОЗДАНИЕ БПЛА С ПОЛЕТНЫМ КОНТРОЛЛЕРОМ НА БАЗЕ ARDUINO

2.1. Проектирование устройства

Схема проекта представлена на рисунке 5.

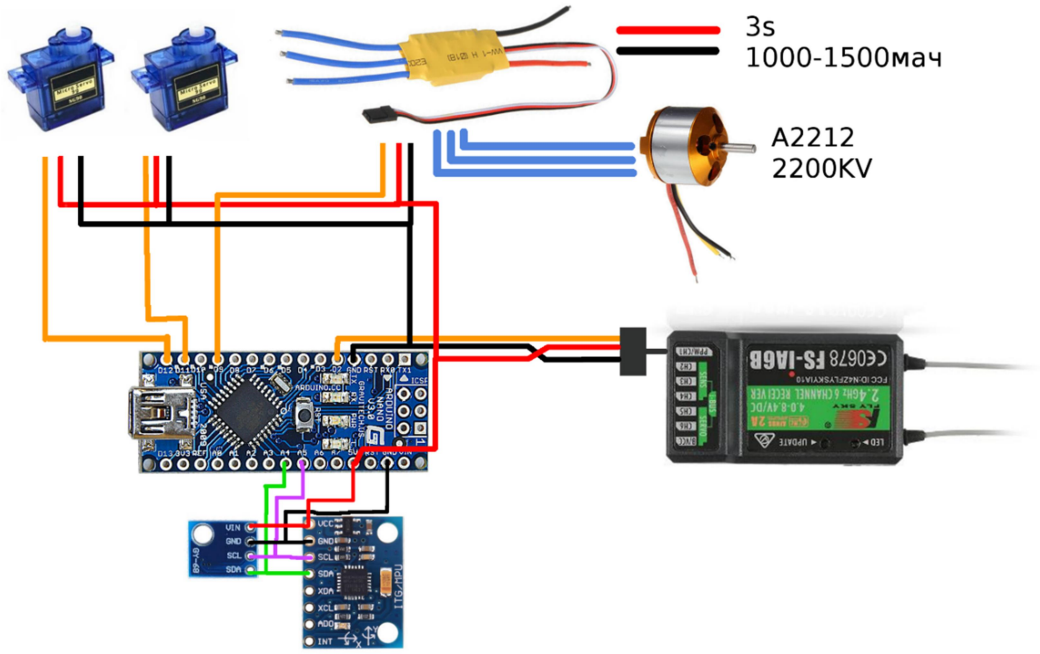


Рисунок 5. Схема проекта с использованием полетного контроллера на базе Arduino

Состав проекта.

1. Плата Arduino Nano.
2. Датчик атмосферного давления BMP-180.
3. Модуль гироскоп + акселерометр GY-521.
4. Радиомодуль Fs-iA6B.
5. Двигатель A2212.
6. Сервоприводы SG-90.
7. Регулятор оборотов ESC.

2.2. Ход работы

За основу было взят открытый код [9], от которого мы оставили только ядро, и переработали основные системы работы с ним. Затем он был залит в плату Arduino Nano (рис. 6). Полный код проекта приведен в Приложении 2.

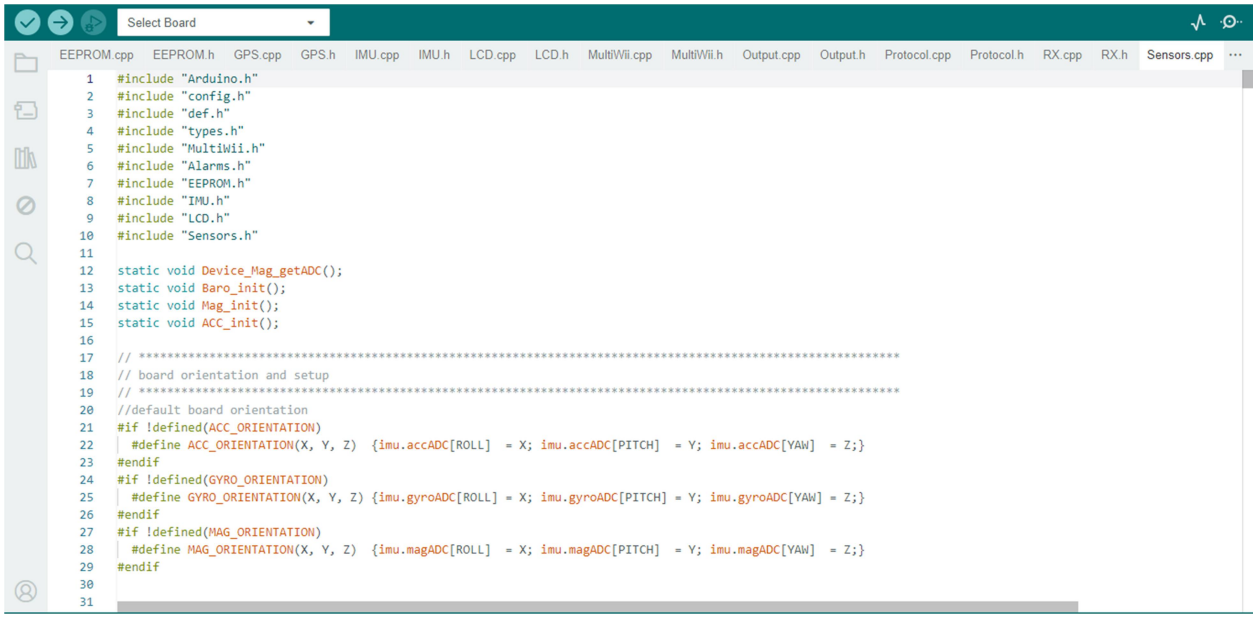


Рисунок 6. Программирование полетного контроллера в среде Arduino IDE

Запрограммированный полетный контроллер был установлен в БПЛА Atom RC Dolphin (рис. 7). Настройка полетного контроллера была произведена через родной конфигуратор.

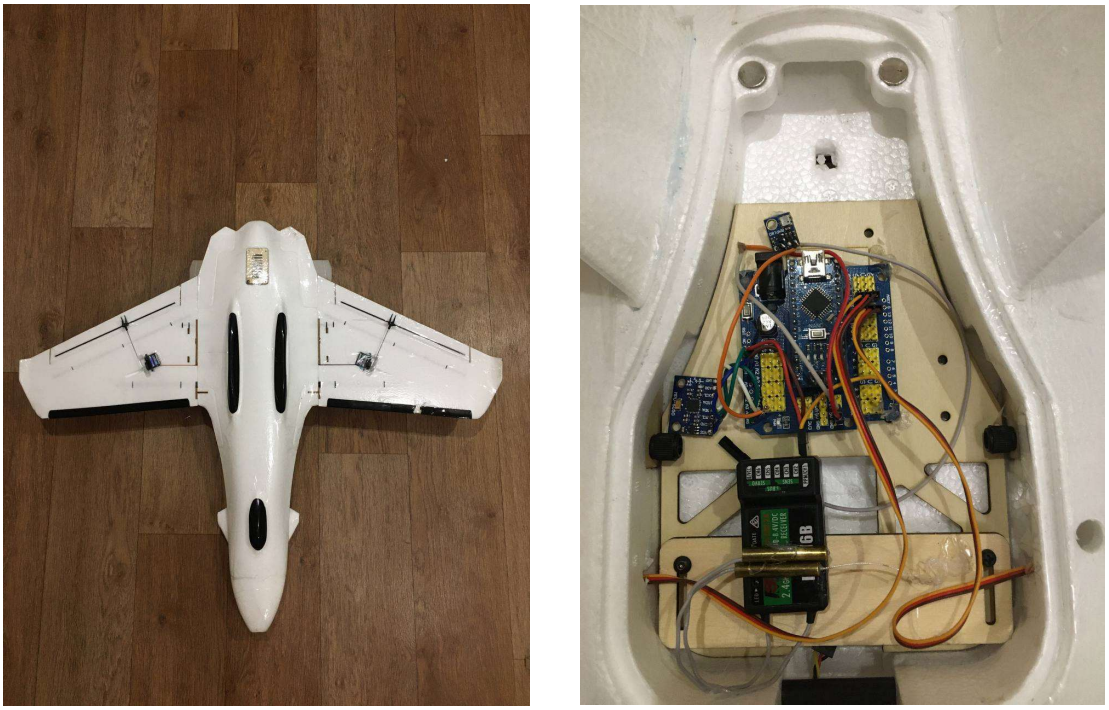


Рисунок 7. Сборка макета БПЛА

2.3. Тестирование устройства

При тестировании устройства были проведены несколько полетов для отладки оборудования (рис. 8). Были найдены и исправлены несколько ошибок в коде. После этого полетный контроллер стал работать стабильно.

Рисунок 8. Тестирование устройства

2.4. Оценка стоимости проекта

Для разработки непосредственно полетного контроллера были задействованы следующие компоненты.

1. Датчик атмосферного давления BMP-180 (примерная стоимость 300 руб.).
2. Модуль гироскоп + акселерометр GY-521 (примерная стоимость 300 руб.).
3. Плата Arduino Nano (примерная стоимость 500 руб.).
4. Плата расширения Shield (примерная стоимость 500 руб.).

Итоговая стоимость полетного контроллера на базе Arduino составляет примерно 1500 руб., что гораздо ниже стоимости аналогичных готовых контроллеров. Для сравнения приведем следующую таблицу 1.

Таблица 1

Сравнение стоимости ПК

Полетный контроллер (ПК)	**SpeedyBee F405 WING MINI**	**Ardupilot APM 2.8**	**SpeedyBee F405 V3 50A BLS 30x30**	**JHEMCU GF30F722-ICM F722 Baro OSD 5V 10V**	**ПК на базе Arduino**
Цена (руб.)	5000	5500	6000	4000	1500

Таким образом, собственный полетный контроллер на базе Arduino является более выгодным решением при построении БПЛА любого типа.

ЗАКЛЮЧЕНИЕ

В ходе выполнения проекта был разработан и успешно протестирован полетный контроллер на базе Arduino, предназначенный для управления радиоуправляемыми моделями самолетов. Благодаря использованию современных технологий и тщательному анализу принципов работы основных компонентов, удалось создать надежное и эффективное решение для стабилизации и безопасности полета.

Результаты исследования показали, что разработанный контроллер способен выполнять сложные маневры и режимы полета, обеспечивая стабильность и комфорт управления моделью. Кроме того, были разработаны подробные инструкции по установке и настройке контроллера, что делает его доступным для широкого круга пользователей.

Проект имеет хорошие перспективы благодаря следующим особенностям:

1. Простота сборки и использования дешевых общедоступных компонентов.
2. Возможность самостоятельного управления самолетом на основе платы Arduino и модуля NRF24L01 для радиосвязи.
3. Легкая интеграция дополнительных функций, таких как стробоскопы, закрылки и механизмы сброса.
4. Возможность обучения основам создания систем управления радиоуправляемыми самолетами на базе Arduino.

Проект представляет собой интересный и перспективный вариант для начинающих и опытных авиамоделистов, желающих создать свою радиоуправляемую модель самолета с уникальными возможностями.

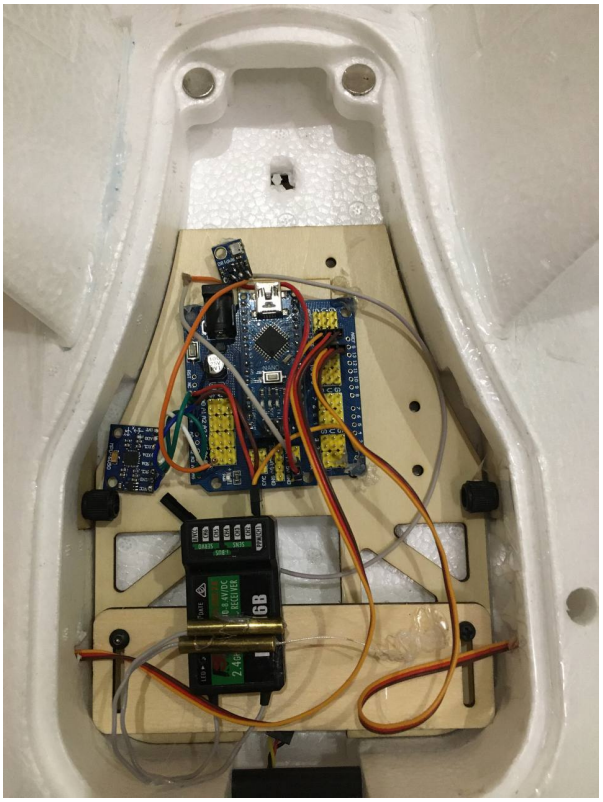
Таким образом, проект «Полетный контроллер на базе Arduino для радиоуправляемой модели самолета» успешно выполнен, и его результаты могут быть использованы для дальнейшего развития и совершенствования систем управления летательными аппаратами.

СПИСОК ЛИТЕРАТУРЫ

1. Белов А. В. ARDUINO От азов программирования до создания практических устройств М.: Изд-во Наука и техника, 2018. 480 с.
2. Аливетри Паоло. Изучаем Arduino. Руководство для начинающих, 2021. 433 с.
3. Общие сведения и положения об аппаратно-вычислительной платформе Arduino: [Электронный ресурс] – Режим доступа: https://www.arduino.cc (дата обращения: 02.11.2024).
4. Бобылев Ф. А., Маркелов М. К. Пример создания полетного контроллера беспилотного летательного аппарата на базе микроконтроллера STM32 // Вестник Пензенского государственного университета. 2023. № 3. С. 72–81.
5. Жиделев А. Р., Скляров А. А. Квадрокоптер как вид беспилотных летательных аппаратов // Вестник современных исследований. 2018. № 53. С. 435–436.
6. Никифоров С. М., Вертегел В. В., Савинов В. В., Симонов И. А. Особенности разработки аппаратной части полетного контроллера БПЛА // Современные проблемы радиоэлектроники и телекоммуникаций. 2019. № 2. С. 53.
7. Коновалов Г. Ф. Радиоавтоматика: учебник. М.: Высш. шк., 2003. 335 с.
8. Маркелов М. К., Ишков А. С., Новичков Д. А., Борисов Н. А. Пример реализации радиоэлектронной системы беспилотного летательного аппарата // Вестник Пензенского государственного университета. 2022. № 4. С. 96–102.
9. Самодельная система стабилизации самолета: [Электронный ресурс] – Режим доступа: https://usamodelkina.ru/13596-samodelnaja-sistema-stabilizacii-samoleta-na-baze-arduino.html?ysclid=m38xeobpow871421789&utm_referrer=https%3A%2F%2Fyandex.ru%2F (дата обращения: 15.11.2024).
10. Гортинский А. А., Мельник Г. И. Разработка модели квадрокоптера // Новые технологии в учебном процессе и производства: материалы XVI межвуз. науч.-техн. конф. (г. Рязань, 17–19 апреля 2018 г.) / под ред. А. А. Платонова, А. А. Бакулиной. Рязань: ИП Жуков В. Ю., 2018. С. 443–446.

ПРИЛОЖЕНИЕ 1

Модель БПЛА с полетным контроллером на базе Arduino




ATOMRC
ATOMRC

ПРИЛОЖЕНИЕ 2

Листинг программного кода

```
#include "Arduino.h"
#include "config.h"
#include "def.h"
#include "Serial.h"
#include "MultiWii.h"

static volatile uint8_t serialHeadRX[UART_NUMBER],serialTailRX[UART_NUMBER];
static uint8_t serialBufferRX[RX_BUFFER_SIZE][UART_NUMBER];
static volatile uint8_t serialHeadTX[UART_NUMBER],serialTailTX[UART_NUMBER];
static uint8_t serialBufferTX[TX_BUFFER_SIZE][UART_NUMBER];

#if defined(TEENSY20)
  unsigned char T_USB_Available(){
    int n = Serial.available();
    if (n > 255) n = 255;
    return n;
  }
#endif

#if defined(PROMINI) || defined(MEGA)
  #if defined(PROMINI)
  ISR(USART_UDRE_vect) {
  #endif
  #if defined(MEGA)
  ISR(USART0_UDRE_vect) {
  #endif
    uint8_t t = serialTailTX[0];
    if (serialHeadTX[0] != t) {
      if (++t >= TX_BUFFER_SIZE) t = 0;
      UDR0 = serialBufferTX[t][0];
      serialTailTX[0] = t;
    }
    if (t == serialHeadTX[0]) UCSR0B &= ~(1<<UDRIE0);
  }
#endif
#if defined(MEGA) || defined(PROMICRO)
  ISR(USART1_UDRE_vect) {
    uint8_t t = serialTailTX[1];
    if (serialHeadTX[1] != t) {
      if (++t >= TX_BUFFER_SIZE) t = 0;
      UDR1 = serialBufferTX[t][1];
      serialTailTX[1] = t;
    }
    if (t == serialHeadTX[1]) UCSR1B &= ~(1<<UDRIE1);
  }
#endif
#if defined(MEGA)
  ISR(USART2_UDRE_vect) {
    uint8_t t = serialTailTX[2];
    if (serialHeadTX[2] != t) {
      if (++t >= TX_BUFFER_SIZE) t = 0;
      UDR2 = serialBufferTX[t][2];
      serialTailTX[2] = t;
    }
    if (t == serialHeadTX[2]) UCSR2B &= ~(1<<UDRIE2);
  }
  ISR(USART3_UDRE_vect) {
```

```
    uint8_t t = serialTailTX[3];
    if (serialHeadTX[3] != t) {
      if (++t >= TX_BUFFER_SIZE) t = 0;
      UDR3 = serialBufferTX[t][3];
      serialTailTX[3] = t;
    }
    if (t == serialHeadTX[3]) UCSR3B &= ~(1<<UDRIE3);
  }
#endif

void UartSendData(uint8_t port) {
  #if defined(PROMINI)
    UCSR0B |= (1<<UDRIE0);
  #endif
  #if defined(PROMICRO)
    switch (port) {
      case 0:
        while(serialHeadTX[0] != serialTailTX[0]) {
           if (++serialTailTX[0] >= TX_BUFFER_SIZE) serialTailTX[0] = 0;
           #if !defined(TEENSY20)
             USB_Send(USB_CDC_TX,serialBufferTX[serialTailTX[0]],1);
           #else
             Serial.write(serialBufferTX[serialTailTX[0]],1);
           #endif
         }
        break;
      case 1: UCSR1B |= (1<<UDRIE1); break;
    }
  #endif
  #if defined(MEGA)
    switch (port) {
      case 0: UCSR0B |= (1<<UDRIE0); break;
      case 1: UCSR1B |= (1<<UDRIE1); break;
      case 2: UCSR2B |= (1<<UDRIE2); break;
      case 3: UCSR3B |= (1<<UDRIE3); break;
    }
  #endif
}

#if defined(GPS_SERIAL)
  bool SerialTXfree(uint8_t port) {
    return (serialHeadTX[port] == serialTailTX[port]);
  }
#endif

void SerialOpen(uint8_t port, uint32_t baud) {
  uint8_t h = ((F_CPU  / 4 / baud -1) / 2) >> 8;
  uint8_t l = ((F_CPU  / 4 / baud -1) / 2);
  switch (port) {
    #if defined(PROMINI)
      case 0: UCSR0A  = (1<<U2X0); UBRR0H = h; UBRR0L = l; UCSR0B |=
(1<<RXEN0)|(1<<TXEN0)|(1<<RXCIE0); break;
    #endif
    #if defined(PROMICRO)
      #if (ARDUINO >= 100) && !defined(TEENSY20)
        case 0: UDIEN &= ~(1<<SOFE); break;
      #endif
      case 1: UCSR1A  = (1<<U2X1); UBRR1H = h; UBRR1L = l; UCSR1B |=
(1<<RXEN1)|(1<<TXEN1)|(1<<RXCIE1); break;
    #endif
    #if defined(MEGA)
```

```
      case 0: UCSR0A  = (1<<U2X0); UBRR0H = h; UBRR0L = l; UCSR0B |=
(1<<RXEN0)|(1<<TXEN0)|(1<<RXCIE0); break;
      case 1: UCSR1A  = (1<<U2X1); UBRR1H = h; UBRR1L = l; UCSR1B |=
(1<<RXEN1)|(1<<TXEN1)|(1<<RXCIE1); break;
      case 2: UCSR2A  = (1<<U2X2); UBRR2H = h; UBRR2L = l; UCSR2B |=
(1<<RXEN2)|(1<<TXEN2)|(1<<RXCIE2); break;
      case 3: UCSR3A  = (1<<U2X3); UBRR3H = h; UBRR3L = l; UCSR3B |=
(1<<RXEN3)|(1<<TXEN3)|(1<<RXCIE3); break;
    #endif
  }
}

void SerialEnd(uint8_t port) {
  switch (port) {
    #if defined(PROMINI)
      case 0: UCSR0B &= ~((1<<RXEN0)|(1<<TXEN0)|(1<<RXCIE0)|(1<<UDRIE0)); break;
    #endif
    #if defined(PROMICRO)
      case 1: UCSR1B &= ~((1<<RXEN1)|(1<<TXEN1)|(1<<RXCIE1)|(1<<UDRIE1)); break;
    #endif
    #if defined(MEGA)
      case 0: UCSR0B &= ~((1<<RXEN0)|(1<<TXEN0)|(1<<RXCIE0)|(1<<UDRIE0)); break;
      case 1: UCSR1B &= ~((1<<RXEN1)|(1<<TXEN1)|(1<<RXCIE1)|(1<<UDRIE1)); break;
      case 2: UCSR2B &= ~((1<<RXEN2)|(1<<TXEN2)|(1<<RXCIE2)|(1<<UDRIE2)); break;
      case 3: UCSR3B &= ~((1<<RXEN3)|(1<<TXEN3)|(1<<RXCIE3)|(1<<UDRIE3)); break;
    #endif
  }
}

void store_uart_in_buf(uint8_t data, uint8_t portnum) {
  #if defined(SERIAL_RX)
    if (portnum == RX_SERIAL_PORT) {
      if (!spekFrameFlags) {
        sei();
        uint32_t spekTimeNow = (timer0_overflow_count << 8) * (64 /
clockCyclesPerMicrosecond());
        uint32_t spekInterval = spekTimeNow -
spekTimeLast;
        spekTimeLast = spekTimeNow;
        if (spekInterval > 2500) {
          serialTailRX[portnum] = 0;
          serialHeadRX[portnum] = 0;
          spekFrameFlags = 0x01;
        }
        cli();
      }
    }
  #endif

  uint8_t h = serialHeadRX[portnum];
  serialBufferRX[h++][portnum] = data;
  if (h >= RX_BUFFER_SIZE) h = 0;
  serialHeadRX[portnum] = h;
}

#if defined(PROMINI)
  ISR(USART_RX_vect)  { store_uart_in_buf(UDR0, 0); }
#endif
#if defined(PROMICRO)
  ISR(USART1_RX_vect)  { store_uart_in_buf(UDR1, 1); }
#endif
```

```
#if defined(MEGA)
  ISR(USART0_RX_vect)  { store_uart_in_buf(UDR0, 0); }
  ISR(USART1_RX_vect)  { store_uart_in_buf(UDR1, 1); }
  ISR(USART2_RX_vect)  { store_uart_in_buf(UDR2, 2); }
  ISR(USART3_RX_vect)  { store_uart_in_buf(UDR3, 3); }
#endif

uint8_t SerialRead(uint8_t port) {
  #if defined(PROMICRO)
    #if defined(TEENSY20)
      if(port == 0) return Serial.read();
    #else
      #if (ARDUINO >= 100)
        if(port == 0) USB_Flush(USB_CDC_TX);
      #endif
      if(port == 0) return USB_Recv(USB_CDC_RX);
    #endif
  #endif
  uint8_t t = serialTailRX[port];
  uint8_t c = serialBufferRX[t][port];
  if (serialHeadRX[port] != t) {
    if (++t >= RX_BUFFER_SIZE) t = 0;
    serialTailRX[port] = t;
  }
  return c;
}

#if defined(SERIAL_RX)
  uint8_t SerialPeek(uint8_t port) {
    uint8_t c = serialBufferRX[serialTailRX[port]][port];
    if ((serialHeadRX[port] != serialTailRX[port])) return c; else return 0;
  }
#endif

uint8_t SerialAvailable(uint8_t port) {
  #if defined(PROMICRO)
    #if !defined(TEENSY20)
      if(port == 0) return USB_Available(USB_CDC_RX);
    #else
      if(port == 0) return T_USB_Available();
    #endif
  #endif
  return ((uint8_t)(serialHeadRX[port] - serialTailRX[port]))%RX_BUFFER_SIZE;
}

uint8_t SerialUsedTXBuff(uint8_t port) {
  return ((uint8_t)(serialHeadTX[port] - serialTailTX[port]))%TX_BUFFER_SIZE;
}

void SerialSerialize(uint8_t port,uint8_t a) {
  uint8_t t = serialHeadTX[port];
  if (++t >= TX_BUFFER_SIZE) t = 0;
  serialBufferTX[t][port] = a;
  serialHeadTX[port] = t;
}

void SerialWrite(uint8_t port,uint8_t c){
  SerialSerialize(port,c);UartSendData(port);
}
```

Рецензия научного руководителя
на проектную работу
«Полетный контроллер на базе Arduino для радиоуправляемой модели самолета»
(автор: Карагулян Андрей Дмитриевич)

VI открытого регионального конкурса исследовательских и проектных работ школьников «Высший пилотаж - Пенза» 2025

Данная работа относится к области разработки прикладного программного обеспечения и конструирования.

В процессе работы над проектом автором были изучены основы электроники, программирования микроконтроллеров и принципы работы авиационных систем. В результате был разработан эффективный и надёжный полётный контроллер, который может быть использован в различных моделях самолётов.

Проект выполнен в соответствии с требованиями и стандартами, а также с учётом современных тенденций в области беспилотных летательных аппаратов. Работа содержит подробное описание всех этапов разработки, включая выбор компонентов, проектирование схемы, написание кода и тестирование готового изделия.

Следует отметить, что автором проекта были успешно решены такие задачи, как обеспечение стабильности полёта, управление двигателями, ориентация в пространстве и обработка сигналов от датчиков. Это свидетельствует о глубоких знаниях и навыках в области электроники и программирования.

В целом, проект «Полетный контроллер на базе Arduino для радиоуправляемой модели самолёта» является актуальным, перспективным и имеет практическую значимость. Он может быть использован как основа для дальнейших исследований и разработок в области беспилотной авиации.

Научный руководитель  Ю.А. Трофимов