МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ
Кафедра многопроцессорных систем и сетей

ЩЕРБАЧ
Анастасия Константиновна

РАЗРАБОТКА МУЗЫКАЛЬНОГО ИНТЕРФЕЙСА НА ОСНОВЕ ПЛАТФОРМЫ ARDUINO

Дипломная работа

Научный руководитель:
кандидат физ.-мат. наук,
доцент С.В. Марков

Допущена к защите
«___»_____2025 г.
Зав. кафедрой многопроцессорных систем и сетей
кандидат физ.-мат. наук, доцент И.Е.Андрушкевич

Минск, 2025

Реферат

Дипломная работа, 42 с., 18 рис., 13 источников, 0 приложений.

Ключевые слова: ARDUINO, МИКРОКОНТРОЛЛЕР, ЛАЗЕР, DFPLAYERMINI, PYTHONQT.

Объект исследования — микроконтроллер Arduino и использование его возможностей для создания музыкального симулятора.

Цели работы — изучение основ разработки на платформе Arduino, разработка аппаратно-программного комплекса, симулирующего игру на музыкальном инструменте, создание приложения для управления этим устройством.

Методы исследования — а) теоретические: изучение литературы, посвящённой разработке с помощью микроконтроллера Ардуино и языка Arduino, литературе по схемотехнике; б) практические: разработка и сборка прототипа описанного аппаратно-программного комплекса, разработка приложения для коммуникации с устройством, проверка работоспособности.

Результатами являются — полноценно функционирующий музыкальный интерфейс на основе Ардуино.

Область применения — использование в качестве обучающей игрушки, в охранных системах, в системе «умный дом», в качестве музыкального инструмента.

Рэферат

Дыпломная работа, 42 с., 18 мал., 13 крыніц, 0 дадаткаў.

Ключавыя словы: ARDUINO, МІКРАКАНТРОЛЕР, ЛАЗЕР, DFPLAYERMINI, PYTHONQT.

Аб'ект даследавання — мікракантролер Arduino і выкарыстанне яго магчымасцяў для стварэння музычнага сімулятара.

Мэты працы — вывучэнне асноў распрацоўкі на платформе Arduino, распрацоўка прылады, якая сімулюе гульню на музычнай прыладзе, стварэнне дадатка для кіравання гэтай прыладай.

Метады даследавання — а) тэарэтычныя: вывучэнне літаратуры, прысвечанай распрацоўцы з дапамогай мікракантролера Ардуіна і мовы Arduino, літаратуры па схематэхніцы; б) практычныя: распрацоўка і зборка прататыпа апісанай прылады, распрацоўка дадатка для камунікацыі з прыладай, праверка працаздольнасці.

Вынікамі з'яўляюцца — паўнавартасна функцыявальны музычны інтэрфейс на аснове Ардуіна.

Вобласць ужывання — выкарыстанне ў якасці адукацыйнай цацкі, у ахоўных сістэмах, у сістэме «разумны дом», у якасці музычнага інструмента.

Abstract

Graduate work, 42 p., 18 illustrations., 13 sources, 0 appendixes.

Keywords: ARDUINO, MICROCONTROLLER, LASER, DFPLAYERMINI, PYTHONQT.

The object of research is the Arduino microcontroller and the use of its capabilities to create musical simulator.

The purposes are to study the fundamentals of development on the Arduino platform, develop a device that simulates playing a musical instrument, and create an application to control this device.

Research methods — a) theoretical: studying literature related to development using the Arduino microcontroller and the Arduino programming language, as well as literature on circuit design; b) practical: designing and assembling a prototype of the described device, developing an application for communication with the device, and verifying its operability.

The results are a fully functional musical interface based on Arduino.

Field of application — as an educational toy, in security systems, within a smart home system, and as a musical instrument.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Все чаще при выборе между традиционными музыкальными инструментами и их электронными симуляторами люди отдают предпочтение симуляторам. Создавать музыку с помощью приложений действительно проще: можно легко переключаться между различными инструментами, они не занимают места в пространстве, с ними проще экспериментировать. Но при всех этих плюсах приложения-симуляторы не дают того же ощущения, как игра на реальном музыкальном инструменте. На стыке этих подходов существует моя работа, совмещая в себе технологичность и традиции.

Использование лазеров в качестве струн создает визуально впечатляющее представление, которое может стать центром внимания на различных мероприятиях. Эстетика лазерного света в сочетании с музыкальным исполнением создает особенную атмосферу, что особенно актуально для концертов, выставок и фестивалей. Использование mp3-плеера с sd-картой позволяет загружать любые коллекции звуков, что открывает множество возможностей для использования аппаратно-программного комплекса в обучении, а лазерный свет может привлекать внимание ребенка. К примеру, можно загружать звуки животных, алфавит или ноты. Тот же принцип с лазерами и фотоэлементами можно использовать в охранных системах и системах умного дома.

Arduino — это электронная платформа с открытым исходным кодом, основанная на простом в использовании аппаратном и программном обеспечении. С помощью различных датчиков платы Arduino способны считывать входные данные, как в случае моего проекта, свет или его отсутствие на фотоэлементе, и превращать их в вывод, выполняя различные команды. Плата является своеобразным вспомогательным инструментом для взаимодействия со стоящим на ней микроконтроллером. Для отправки инструкций на микроконтроллер используется язык программирования Arduino (основанный на Wiring) и программное обеспечение Arduino (IDE), основанное на Processing. Arduino широко применяется на проектах среднего и невысокого уровней сложности в различных областях, таких как автоматизация, IoT (интернет вещей), робототехника, искусство и многое другое. Это делает Arduino универсальным инструментом для реализации самых разных идей.

В качестве платформы для реализации мной был выбран Arduino Uno– один из самых доступных и актуальных микроконтроллеров семейства Arduino. Arduino Uno построен на ATmega328. Платформа имеет 14 цифровых вход/выходов (6 из

которых могут использоваться как выходы ШИМ), 6 аналоговых входов, кварцевый генератор 16 МГц, разъем USB, силовой разъем, разъем ICSP и кнопку перезагрузки. Для работы необходимо подключить платформу к компьютеру посредством кабеля USB, либо подать питание при помощи адаптера AC/DC или батареи.

Обозначим цели и задачи дипломной работы:

Целью данной работы является создание аппаратно-программного комплекса, которое объединяет технологические инновации и традиционные подходы к музыкальному исполнению. Разрабатываемая система использует лазерные струны, что не только сохраняет ощущение игры на музыкальном инструменте, но и добавляет визуальное впечатление, превращая процесс исполнения в эффектное зрелище.

Задачи:

1. Исследовать особенности платформы Arduino и выявить её преимущества для реализации поставленной задачи, исследовать возможности модуля MP3-плеера DFPlayer Mini.
2. Определить аспекты существующих реализаций, которые работа может улучшить.
3. Проанализировать и выбрать программные средства для разработки.
4. Разработать описанное устройство и создать приложение для коммуникации с ним.
5. Провести тестирование работоспособности аппаратно-программного комплекса.

Актуальность выбранной темы:

Актуальность проекта обусловлена его способностью адресовать современные запросы пользователей, стремящихся к качественному, интерактивному и эстетичному музыкальному опыту. Совмещение традиционных аспектов исполнения (ощущение от игры на реальном инструменте) с цифровыми технологиями (управление через приложение, гибкость библиотеки звуков) выгодно выделяет проект. Помимо музыкальной сферы, проект может применяться в таких сферах, как образование дошкольников и детей младшего школьного возраста, возможно также использовать схожий принцип в охранных

системах и системах «умный дом». Такая мультифункциональность и адаптивность аппаратно-программного комплекса делают его конкурентоспособным среди существующих решений и подчеркивает его актуальность.

Объектом исследования будет являться язык программирования Arduino и его библиотеки.

Сам музыкальный интерфейс представляет из себя электронное устройство, состоящее из блока лазерных светодиодов, направленных на фоторезисторы, блока фоторезисторов, и MP3-модуля. При прерывании луча напряжение многократно уменьшается и с входа, к которому подключен этот фоторезистор, считывается низкое значение напряжения. В зависимости от того, на каком из входов зафиксировано низкое напряжение, с помощью MP3 модуля DFPlayer Mini проигрывается соответствующая нота. Для переключения между различными библиотеками звуков было написано приложение.

Приложение было разработано с использованием библиотек языка Python PyQt5, PySerial.

ГЛАВА 1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

1.1 Обоснование выбора средств для разработки

Для программирования платы была выбрана специализированная среда разработки для Arduino, Arduino IDE. Она оптимизирована для программирования микроконтроллеров Arduino, обеспечивает быструю компиляцию, прошивку и отладку (если она предусмотрена самой платой), а также автоматическую настройку конфигурации, что сокращает время перехода от идеи к прототипу. Встроенная поддержка множества стандартных библиотек и модулей позволяет реализовывать сложные системы без необходимости решать вопросы низкоуровневого взаимодействия с аппаратным обеспечением. Благодаря оптимизации под особенности микроконтроллеров Arduino, данная среда демонстрирует высокую стабильность и производительность по сравнению с универсальными IDE.

Для разработки приложения был выбран модуль PyQt5, обладающий следующими преимуществами:

- Кроссплатформенность позволяет охватить широкий круг пользователей без дополнительных затрат на адаптацию под разные ОС.
- Фреймворк предлагает широкий спектр готовых виджетов, инструментов для построения сложных пользовательских интерфейсов и продвинутых элементов управления, что существенно ускоряет процесс разработки.
- Механизм сигналов и слотов позволяет легко организовывать взаимодействие между компонентами интерфейса, обеспечивая отзывчивость и динамичность приложений.
- Возможности стилизации, кастомизации и интеграции с другими модулями Python.

В качестве среды разработки рассматривались PyCharm, Sublime Text, Thonny и Visual Studio Code. Я выбрала последний вариант, так как VS Code значительно легче, чем полноценные IDE, что обеспечивает быструю загрузку и низкое потребление ресурсов. Благодаря огромному количеству расширений можно адаптировать среду под любые задачи, включая поддержку PyQt5, автоматическую компиляцию .ui-файлов и интеграцию с инструментами отладки. Интегрированный терминал позволяет запускать скрипты, управлять зависимостями и взаимодействовать с системой без необходимости

переключаться между окнами, а встроенные инструменты контроля версий упрощают работу с репозиториями.

Для разработки чертежа корпуса я решила использовать AutoCAD 2024, так как это профессиональный графический редактор, используемый многими инженерами. AutoCAD обеспечивает высокую точность черчения, удобную работу с 2D и 3D моделями и поддержку современных стандартов проектирования. Его инструменты аннотирования, автоматизации и работы со слоями ускоряют создание технических чертежей.

Теперь, после выбора средств, с которыми я буду работать, можно перейти к подробному теоретическому обзору.

1.2 Теоретические сведения о платформе Arduino

1.2.1 Преимущества и недостатки разработки на Arduino

Главным преимуществом платформы Arduino является простота и низкий порог вхождения. Программы пишутся на интуитивно понятном C-подобном языке Arduino, при этом всегда можно заменить конкретные команды или библиотеки аналогичными на C++. Стоит также отметить огромную кодовую базу и активное сообщество.

Arduino идеально подходит для проектов простого и среднего уровня, но сложные системы реализуются хуже, в первую очередь из-за отсутствия многозадачности. Малая частота процессора и объем памяти, выделенный для программы, также не улучшают ситуацию. Из плюсов стоит выделить наличие энергонезависимой памяти EEPROM с огромным (заявлено 100000, на практике даже больше) запасом циклов перезаписи.

Питание, программирование и связь с устройством осуществляется с помощью одного кабеля USB. Коммуникация по последовательному интерфейсу и SPI отлично реализована.

Arduino IDE является кроссплатформенным и может работать на Windows, Mac, Linux. Arduino IDE также поддерживает широкий диапазон плат Arduino и совместимых устройств, таких как ESP32, STM32, Teensy и другие. Возможно также использовать Arduino IDE для программирования других микроконтроллеров, таких как AVR, PIC или ARM, используя сторонние библиотеки. С другой стороны, Arduino IDE является далеко не самой удобной средой разработки, не поддерживая такие инструменты, как рефакторинг кода, отладка, управление версиями или юнит-тестирование. Для более сложных

проектов лучше использовать Atmel Studio (Microchip Studio), чтобы напрямую общаться с микроконтроллером Atmega.

1.2.2 Описание аппаратной части платы Arduino Uno

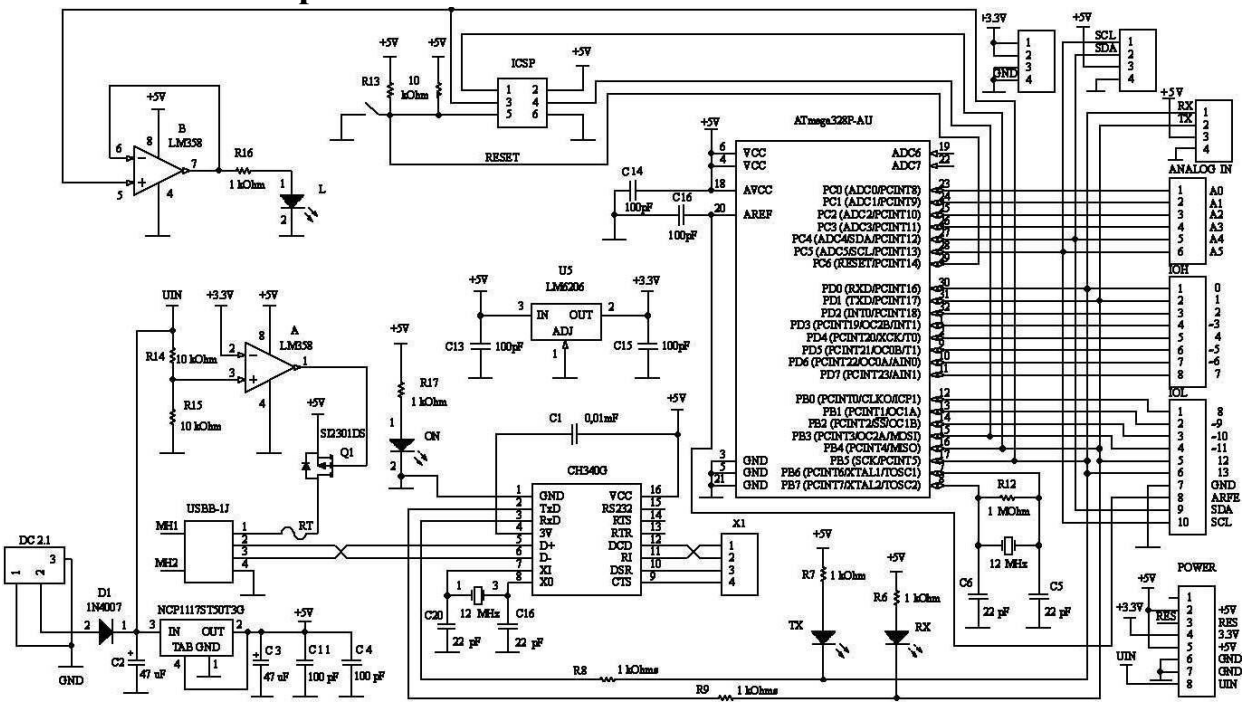

Рис.1 Принципиальная схема Arduino Uno

1.2.2.1 Память

Плата основана на микроконтроллере ATMega328 семейства AVR. Микроконтроллер ATmega328 располагает 32 кБ флэш памяти, из которых 0,5 кБ используется для хранения загрузчика, а также 2 кБ ОЗУ (SRAM) и 1 Кб EEPROM, которая читается и записывается с помощью библиотеки EEPROM.

1.2.2.2 Питание

Arduino Uno может получать питание через USB-порт или от внешнего источника питания. Источник питания выбирается автоматически.

Внешнее питание может подаваться через блок питания или аккумуляторной батареей. Преобразователь напряжения подключается посредством разъема 2.1 мм с центральным положительным полюсом. На плате имеется стабилизатор напряжения, поэтому к качеству питающего напряжения устройство нетребовательно.

Рекомендуемый диапазон напряжения питания от 7 В до 12 В. При напряжении питания ниже 7 В, вывод 5V может выдавать менее 5 В, при этом платформа может работать нестабильно. При использовании напряжения выше 12 В регулятор напряжения может перегреться и повредить плату.

Силовые контакты:

- Vin. Вход используется для подачи питания от внешнего источника (в отсутствие 5 В от разъема USB или другого регулируемого источника питания).
- 5V. Регулируемый источник напряжения, используемый для питания микроконтроллера и компонентов на плате. Питание может подаваться от вывода VIN через регулятор напряжения, или от разъема USB, или другого регулируемого источника напряжения 5 В.
- 3V3. Напряжение на выводе 3,3 В генерируется встроенным регулятором на плате. Максимальное потребление тока 50 мА.
- GND. Выводы заземления.

1.2.2.3 Порты ввода-вывода

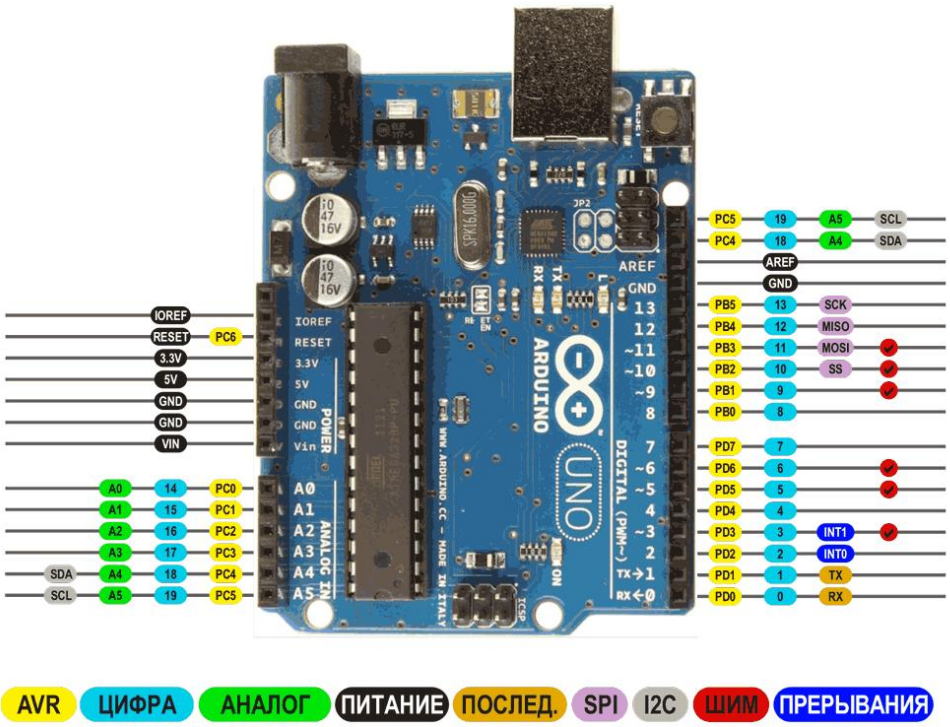

Рис.2 Внешний вид и расшифровка функций выводов Arduino Uno

На плате Arduino Uno располагается 14 цифровых портов ввода-вывода, 6 из которых поддерживают широтно-импульсную модуляцию (помечены на плате

знаком «~»). Каждый из 14 цифровых выводов Uno может настроен как вход или выход. Выводы работают при напряжении 5 В. Каждый вывод имеет нагрузочный резистор (по умолчанию отключен) 20-50 кОм и может пропускать до 40 мА.

Выводы с особыми функциями:

- Последовательная шина: 0 (RX) и 1 (TX). Выводы используются для получения (RX) и передачи (TX) данных TTL. Данные выводы подключены к соответствующим выводам микросхемы последовательной шины ATmega8U2 USB-to-TTL.
- Внешнее прерывание: 2 и 3. Данные выводы могут быть сконфигурированы на вызов прерывания либо на младшем значении, либо на переднем или заднем фронте, или при изменении значения.
- ШИМ: 3, 5, 6, 9, 10, и 11. Любой из выводов обеспечивает ШИМ с разрешением 8 бит при помощи функции analogWrite().
- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Посредством данных выводов осуществляется связь SPI, для чего используется библиотека SPI.
- LED: 13. Встроенный светодиод, подключенный к цифровому выводу 13. Если значение на выводе имеет высокий потенциал, то светодиод горит.

Установлены также 6 аналоговых входов (обозначенных как A0-A5, при их использовании в качестве цифровых обозначаются как 14-19), каждый разрешением 10 бит. Стандартно выводы имеют диапазон измерения до 5 В относительно земли, тем не менее имеется возможность изменить верхний предел посредством вывода AREF и функции analogReference().

Выводы с дополнительными функциями:

- I2C: 4 (SDA) и 5 (SCL). Посредством выводов осуществляется связь I2C (TWI), для создания которой используется библиотека Wire.

Дополнительная пара выводов платформы:

- AREF. Опорное напряжение для аналоговых входов. Используется с функцией analogReference().
- Reset. Низкий уровень сигнала на выводе перезагружает микроконтроллер. Обычно применяется для подключения кнопки перезагрузки на плате расширения, закрывающей доступ к кнопке на самой плате Arduino.

1.2.2.4 Связь

ATmega328 поддерживают последовательный интерфейс UART TTL (5 В), осуществляемый выводами 0 (RX) и 1 (TX). Установленная на плате микросхема ATmega8U2 направляет данный интерфейс через USB, программы на стороне компьютера "общаются" с платой через виртуальный COM-порт. Прошивка ATmega8U2 использует стандартные драйвера USB COM. Мониторинг последовательной шины (Serial Monitor) позволяет посылать и получать текстовые данные при подключении к платформе. Светодиоды RX и TX на платформе будут мигать при передаче данных через микросхему FTDI- или USB-подключение (но не при использовании последовательной передачи через выводы 0 и 1).

ATmega328 поддерживает интерфейсы I2C (TWI) и SPI. В Arduino включена библиотека Wire для удобства использования шины I2C.

Для связи с компьютером на плате Arduino Uno имеется разъем USB-BF.

1.3 Теоретические основы разработки проекта на Arduino

1.3.1 Обзор среды разработки Arduino IDE

Для написания программ для Arduino и их загрузки в микроконтроллер используется среда разработки Arduino IDE.

Интегрированная среда разработки Arduino - или Arduino IDE - содержит текстовый редактор для написания кода, область сообщений, текстовую консоль, панель инструментов с кнопками для общих функций и ряд меню. Она подключается к оборудованию Arduino для загрузки программ и связи с ними.

Программы, написанные с помощью программного обеспечения Arduino (IDE), называются скетчами(sketch). Они пишутся в текстовом редакторе и сохраняются с расширением файла .ino. В редакторе есть функции для вырезания/вставки и для поиска/замены текста. Область сообщений дает обратную связь при сохранении и экспортировании, а также отображает ошибки. Консоль отображает текст, выводимый программным обеспечением Arduino (IDE), включая полные сообщения об ошибках и другую информацию. Кнопки на панели инструментов позволяют проверять и загружать программы, создавать, открывать и сохранять скетчи, а также открывать последовательный монитор.

Справа находится вкладка с скетчами, менеджером плат, менеджером библиотек, дебагом (если плата поддерживает такую функцию) и поиском.

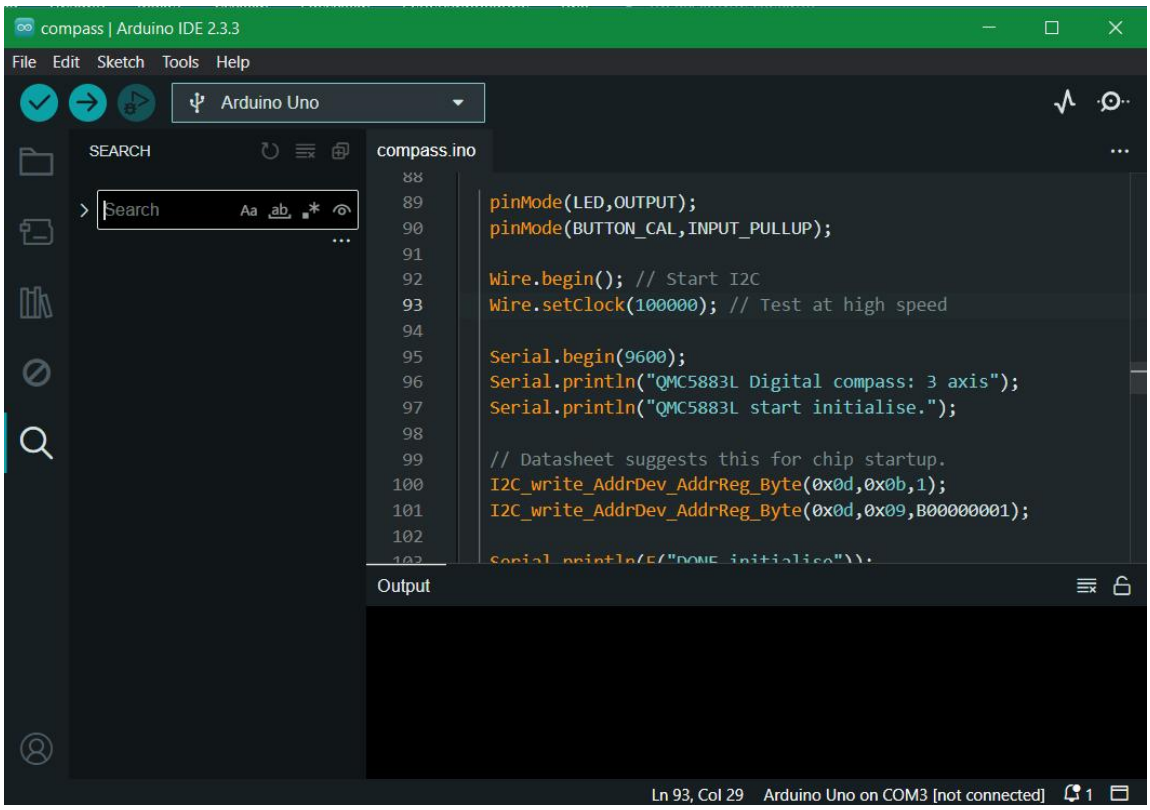

Рис. 3 Скетч в Arduino IDE

Для того, чтобы загрузить программу на плату, необходимо её подключить. Для этого в меню “Tools” нужно выбрать подменю “Board”. В нем отображаются все доступные платы, поддерживаемые средой разработки. В подменю “Port” будет указан последовательный порт, необходимо проверить. Чтобы он был правильным. В ОС Mac последовательный порт может обозначаться как dev/tty.usbserial-1B1 (для платы USB) или /dev/tty.USA19QW1b1P1.1 (для платы последовательной шины, подключенной через адаптер Keyspan USB-to-Serial). В ОС Windows порты могут обозначаться как COM1 или COM2 (для платы последовательной шины) или COM4, COM5, COM7 и выше (для платы USB). Определение порта USB производится в поле Последовательной шины USB Диспетчера устройств Windows. В ОС Linux порты могут обозначаться как /dev/ttyUSB0, /dev/ttyUSB1.

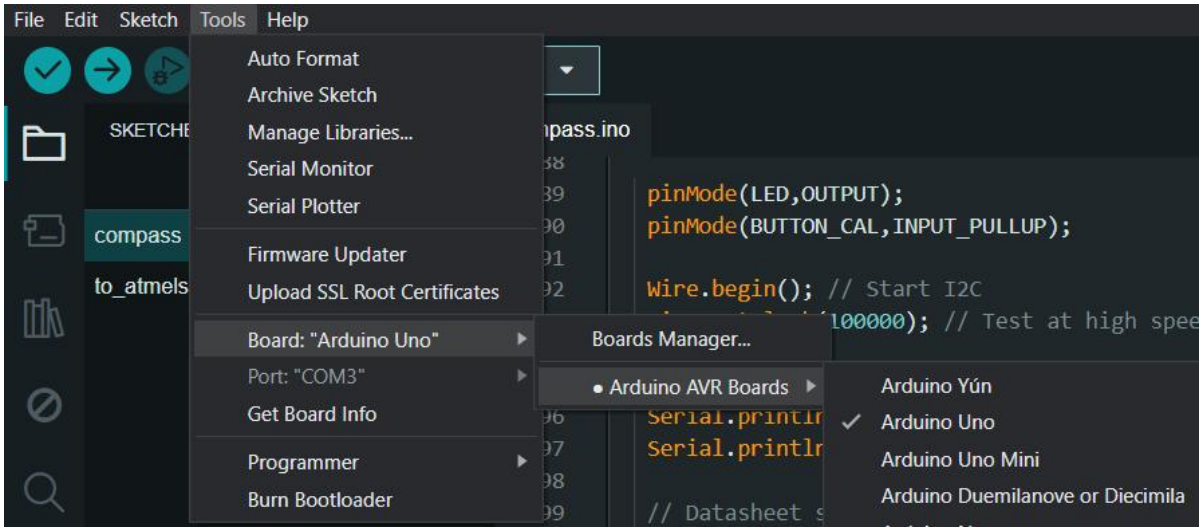

Рис. 4 Меню выбора платы и порта

1.3.2 Программирование на языке Arduino

Язык Arduino имеет C-подобный синтаксис, и любой его метод можно заменить на аналогичный на C++ или C. В основе любой программы на Arduino лежат две фундаментальные функции:

1) void setup()

Эта функция выполняется один раз сразу при загрузке контроллера. Обычно туда помещают начальные настройки, необходимые для корректной работы программы, такие как режим работы пинов платы, настройки последовательного порта, интерфейсов связи, различных подключенных датчиков и других параметров, которые должны быть настроены сразу по началу работы программы. Наиболее часто в setup() выполняются следующие задачи:

- pinMode с указанием номера и типа пина. Это инструкция определяет режим работы пинов .
- Serial.begin с указанием скорости обмена данными по последовательному порту.
- Инструкции подключения и инициализации различных объектов библиотек.
- Инициализацию глобальных переменных, если мы по каким-то причинам не можем сделать это при определении переменных в глобальной области видимости.
- Настройка периферии, проверка подключения, диагностика и калибровка датчиков.

- Настройка таймеров, активация прерываний.

2) void loop()

Функция loop() представляет собой бесконечный цикл, в котором прописан основной функционал программы. После завершения всех команд внутри цикла микроконтроллер автоматически возвращается в начало функции, что обеспечивает постоянное выполнение программы до выключения платы. Помимо базовой логики работы, в loop() часто выполняются следующие задачи:

- Обрабатываются входных сигналов от датчиков, выполняется управление моторами, дисплеями и другой периферией, а также происходят вычисления по алгоритмам управления.
- Обработка событий и прерываний: loop() может включать вызовы функций для опроса флагов, установленных в обработчиках прерываний, и корректное реагирование на внешние события.
- Использование функций времени, таких delay(), блокирующее всю работу микроконтроллера на заданное время, и millis(), отсчитывающее время с начала работы микроконтроллера, которое также можно использовать для приостановки выполнения конкретной задачи, не останавливая остальные процессы.

Любая программа для Arduino должна содержать как минимум эти две функции.

Помимо программных особенностей, в программировании микроконтроллеров важно учитывать и аппаратные, такие как малое количество оперативной памяти и флеш-памяти. Рекомендуется следить за размерами переменных, не используя без надобности 64-х и 32-х битные типы данных, использовать арифметические операции, рассчитывая, сколько памяти они используют. Также стоит учитывать энергоэффективность, если устройство будет питаться не от компьютера. Для этого можно использовать специальные режимы работы.

1.3.3 Использование документации DFPlayer Mini для разработке программы

Любая работа с какими-либо датчиками или микросхемами должна начинаться с изучения документации. В ней подробно описаны технические характеристики микросхемы, такие как вольтаж, диапазон измерений, погрешности и т.п., распиновка и описание режимов работы для каждого пина, список команд и кодов ответа, организация памяти, если такая есть.

Проанализировав задачу, видно, что единственные функции, которые должен выполнять плеер – проигрывание заданного звука из заданной папки и регулировка громкости динамика, а взаимодействие будет осуществляться исключительно через интерфейс. Для выполнения этих целей разработка библиотеки будет избыточной, достаточно реализовать функцию для отправки команд.

Обратившись к документации модуля, узнаём, в каком формате на DFPlayer отправляются команды. В даташите описан протокол обмена, согласно которому пакет команды состоит из 10 байт и имеет следующую структуру:

1. Стартовый байт, позволяющей плееру распознать начало новой команды.
2. Версия пакета. Этот параметр гарантирует, что плеер правильно интерпретирует последующую информацию.
3. Длина оставшейся части пакета. Так как длина пакета 10 байт, всегда имеет значение 6, указывающее, что далее следуют 6 байт данных.
4. Код команды, который определяет, какое действие должен выполнить плеер. Все коды команд перечислены и описаны в документации.
5. Флаг обратной связи. В случае, если обратная связь не требуется, выставляется в 0x00.
6. Параметр команды, состоящий из двух байтов, где байт 5 – старший, а байт 6 – младший.
7. Контрольная сумма, также состоящая из двух байтов. Вычисляется как отрицательная сумма байтов от версии до младшего байта параметра, что помогает плееру проверить корректность полученной команды.
8. Конечный байт, сигнализирующий об окончании пакета. Плеер заканчивает принимать информацию и начинает обработку.

Format: $S VER Len CMD Feedback para1 para2 checksum $O		
$S	Start byte 0x7E	Each command feedback begin with $, that is 0x7E
VER	Version	Version Information
Len	the number of bytes after “Len”	Checksums are not counted
CMD	Commands	Indicate the specific operations, such as play / pause, etc.
Feedback	Command feedback	If need for feedback, 1: feedback, 0: no feedback
para1	Parameter 1	Query high data byte
para2	Parameter 2	Query low data byte
checksum	Checksum	Accumulation and verification [not include start bit $]
$O	End bit	End bit 0xEF

Рис. 5 Формат последовательной коммуникации

Изучив список команд, обнаружим интересующие нас коды команд:

1. Код 0x06 используется для установки уровня громкости. Это позволяет динамически регулировать громкость динамика в диапазоне от минимального до максимального значения (0–30).
2. Код 0x0F отвечает за воспроизведение трека. Здесь параметр команды объединяет номер папки и номер трека, что даёт возможность легко структурировать библиотеку аудиофайлов на SD-карте. Подробнее об объединении параметров для команды 0x0F написано в главе 2 разделе 3.

Таким образом, на данный момент для коммуникации с плеером будут существовать функция, формирующая и отправляющая пакет команды, и функции, использующие её для реализации конкретных задач. В дальнейшей разработке, если понадобится, новые функции можно вводить по тому же принципу, или разработать полноценную библиотеку для взаимодействия с плеером.

CMD	Function Description	Parameters(16 bit)
0x01	Next	
0x02	Previous	
0x03	Specify tracking(NUM)	0-2999
0x04	Increase volume	
0x05	Decrease volume	
0x06	Specify volume	0-30
0x07	Specify EQ(0/1/2/3/4/5)	Normal/Pop/Rock/Jazz/Classic/Base
0x08	Specify playback mode (0/1/2/3)	Repeat/folder repeat/single repeat/ random
0x09	Specify playback source(0/1/2/3/4)	U/TF/AUX/SLEEP/FLASH
0x0A	Enter into standby – low power loss	
0x0B	Normal working	
0x0C	Reset module	
0x0D	Playback	
0x0E	Pause	
0x0F	Specify folder to playback	1~10(need to set by user)

Рис. 6 Список кодов команд плеера

1.4 Обзор существующих решений

Хотя существует множество проектов на Arduino, симулирующих тот или иной музыкальный инструмент, моя работа имеет неоспоримое преимущество, так как совмещает в себе сразу несколько инструментов. Для сравнения рассмотрим другие реализации музыкальных инструментов на Arduino: пианино и лазерную арфу.

Пианино реализовано с помощью кнопок, что существенно удешевляет и упрощает сборку, но при этом теряет в эстетическом аспекте. Однако использование лазерных светодиодов и фоторезисторов требует высокой точности установки, так как луч должен идеально попадать на маленький фотоэлемент. Для этого необходимо размещать все элементы на каркас, тогда как для пианино нужна всего лишь макетная плата. Таким образом, пианино выигрывает в компактности, хотя и проигрывает в визуальной части.

Из всех музыкальных проектов лазерная арфа имеет больше всего схожестей с моей работой. Её реализаций много, и в каждой из них есть технологическое решение, которое, как мне кажется, мой проект может улучшить. Рассмотрим их:

1. Вариант лазерной арфы пользователя YouTube Science Shack. Он использовал один лазерный луч, расщеплённый на несколько с помощью зеркала. Зеркало управляется высокоскоростным гальванометром, который заставляет зеркало вибрировать с частотой 20000 позиций в секунду. Лазер постоянно включается и выключается, каждый цикл включения-выключения совпадает с циклом смены позиции лазера. Благодаря высокой скорости переключения лучей, глаз воспринимает их как существующие одновременно, но по факту, в момент времени существует только один луч. Прерывание луча обнаруживается благодаря оптическим датчикам, которые улавливают отражённый от специальных насадок на пальцы свет. Так как контроллер всегда знает, какой из лучей действительно существует, при прерывании он сгенерирует соответствующий MIDI-сигнал. Возможно менять генерируемые звуки путём изменения кода. Из плюсов такого технологического решения можно выделить возможность изменять количество струн из кода, а также менять звуки. Минусами такого подхода являются сложность расчётов, необходимых для синхронизации зеркала и светодиода, усложнение конструкции, чрезмерная сложность программы, привязка к компьютеру, отсутствие закрытого корпуса и дорогие комплектующие.

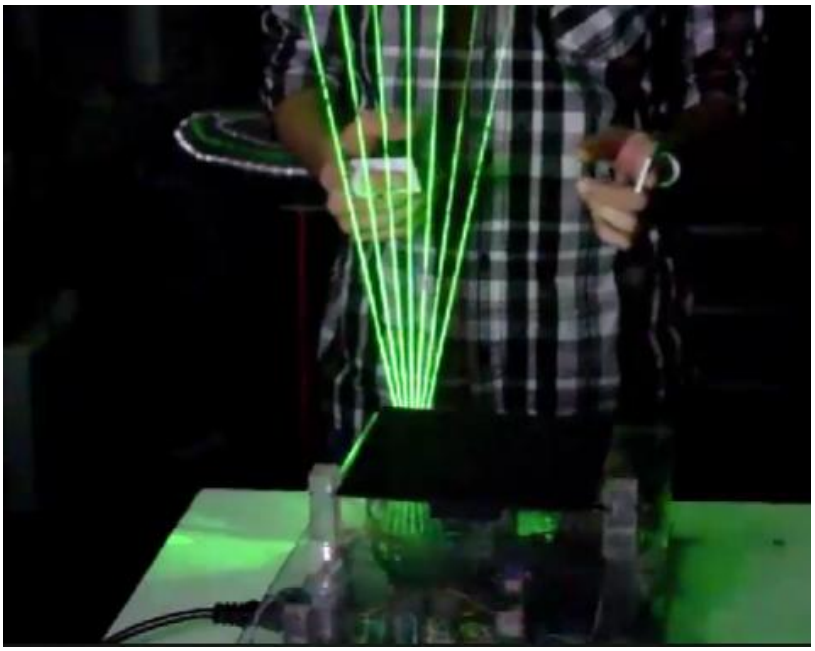

Рис. 7 Лазерная арфа Science Shack

2. Вариант лазерной арфы пользователя портала www.instructables.com yaroshka представляет собой более классическую лазерную арфу, имеющую 9 лазерных струн, 2 из которых отвечают за понижение и повышение тона. В этом проекте разрыв луча обнаруживается с помощью фоторезисторов. Звук получается посредством генерации MIDI-сигнала, что привязывает устройство к MIDI-девайсу и компьютеру. Из плюсов можно выделить крепкий закрытый корпус, простую схему работы, возможность смены октавы не меняя код. Из минусов: привязка к компьютеру, возможная путаница с множеством лучей, которые легко случайно задеть. В моем проекте менять октавы, а также менять инструмент будет специальное приложение.

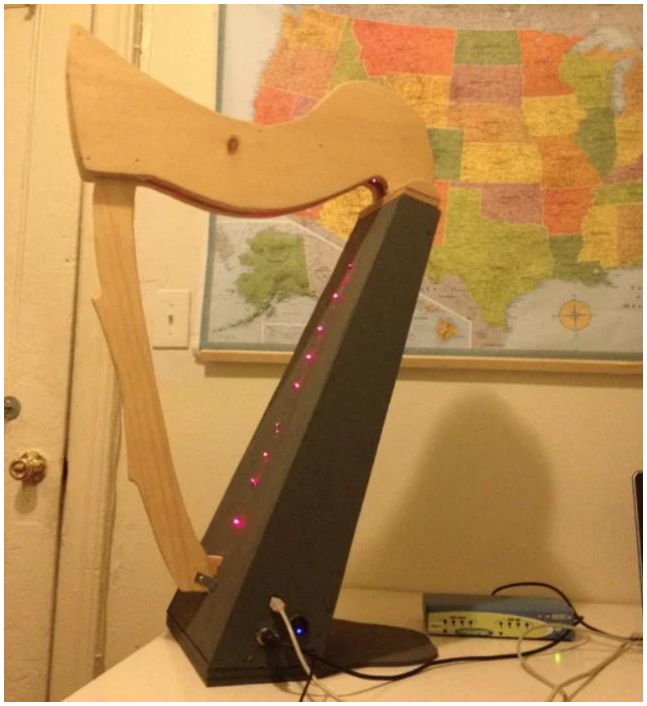

Рис.8 Лазерная арфа yaroshka

3. Типичным лазерным арфам, массово продающимся на маркетплейсах, свойственно использование динамика и функции tone() или ее аналога для воспроизведения, отсутствие возможности отрегулировать громкость или изменить набор звуков, легкий открытый корпус, не защищающий компоненты схемы от возможных повреждений. Из плюсов можно выделить малый габарит корпуса, из минусов крайне ограниченный функционал хрупкость устройства.

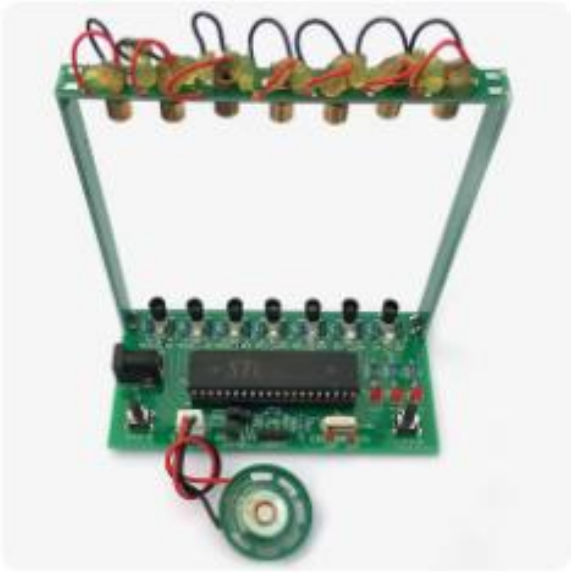

Рис. 9 Пример лазерной арфы с маркетплейсов

После тщательного рассмотрения других устройств, можно определить преимущества моего проекта:

1. MP3-модуль позволяет управлять уровнем громкости, считывать треки с sd-карты, непосредственно подключённой к нему, что позволяет устройству функционировать без подключения к компьютеру. MP3-модуль также может работать с до 10000 треками, записанными на sd-карте, на которую пользователь может загрузить любые звуки, кастомизируя таким образом аппаратно-программный комплекс для своих нужд. С использованием простого динамика эти функции или недоступны, или сложнее в реализации. Кроме того, MP3-модуль имеет довольно доступную цену, и не увеличивает общую стоимость аппаратно-программного комплекса так, как увеличивает использование MIDI-интерфейса, особенно при массовом производстве.
2. Неоспоримым преимуществом является наличие приложения, через которое осуществляется управление библиотеками звуков. На данный момент функционал приложения включает в себя добавление новых библиотек, поиск по названию, переключение между наборами звуков.
3. Закрытый корпус защищает устройство во время использования, облегчает транспортировку, имеет эстетичный вид. Он легко открывается, чтобы произвести замену батареек или провести нужный ремонт. Сбоку корпуса выведены копка Reset, USB-B порт, DC Power Jack платы Arduino, а также включатель.

4. Возможность использования аппаратно-программного комплекса без подключения к компьютеру.

Были адаптированы следующие свойства конкурентных решений:

1. Возможность смены библиотеки звуков.
2. Небольшой размер корпуса.
3. Использования нескольких лазерных светодиодов.
4. Использование фоторезистора в качестве датчика прерывания луча.

В результате проведенного анализа можно сделать вывод, что моя работа, объединяющая несколько музыкальных инструментов, предлагает ряд преимуществ по сравнению с аналогами, такими как лазерная арфа и пианино на Arduino.

ГЛАВА 2. РЕАЛИЗАЦИЯ ПРОЕКТА

2.1 Подготовка принципиальной схемы

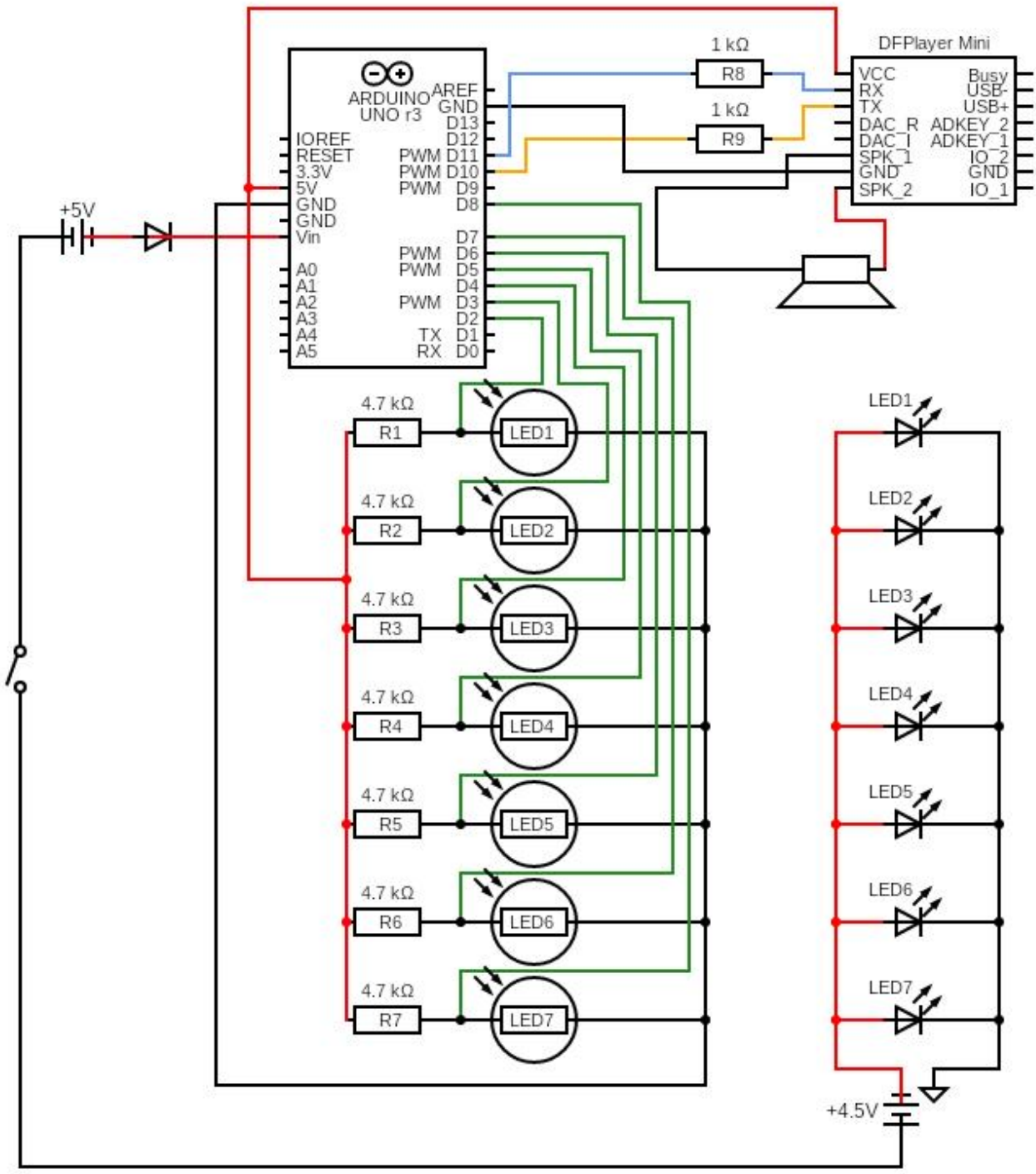

Рис. 10 Принципиальная схема

2.2 Подготовка электрических компонентов микросхемы

Для успешной работы аппаратно-программного комплекса необходимо подобрать правильные электрические компоненты. В прошлой главе было указано, что для работы выбрана плата Arduino Uno. Выберем лазерные светодиоды. Так как использование лазерных светодиодов мощностью более 5 мВт без особых норм безопасности представляет опасность для зрения даже при кратковременном воздействии на глаза, но при слишком малой мощности на фотоэлементы не будет падать достаточное количество света, в проекте используются лазерные модули мощностью 5мВт с допустимым питанием 3-5 В.

Рис. 11 Лазерный модуль

DFPlayer Mini - это компактный модуль аудио-плеера, который позволяет воспроизводить звуковые файлы в формате MP3, WMA и WAV. Легко интегрируется в различные электронные устройства и системы, такие как Arduino, Raspberry Pi и другие микроконтроллеры. Он имеет небольшой размер, низкое потребление энергии и простой интерфейс управления, что делает его идеальным выбором для различных проектов DIY и встраиваемых систем.

Рис. 12 MP3 модуль DFPlayer Mini

Модуль может распознавать карты объемом до 32G и файловой системой FAT16, FAT32. Файлы и папки на sd карте для управления по UART должны хранится строго в определенном формате. Имя файла трека обязательно должно начинаться с 4-х цифр. Например: 0001.mp3 или 0010track.mp3. Имена папок состоят из 2 цифр. 00 - 99. В общем виде структура файлов на sd карте должна иметь следующий вид: \0001.mp3. или \02\0001.mp3. Плата поддерживает 3 режима управления: через интерфейс UART (Serial Mode), управление кнопками (AD Key Mode), упрощенный режим(I/O Mode).

DFPlayer Mini обладает следующими техническими характеристиками:

- Напряжение питания: от 3,2 до 5 В;
- Поддерживаемые форматы аудиофайлов: MP3, WMA и WAV;
- Максимальная емкость карты памяти: 32 ГБ (Micro SD/TF);
- Максимальное количество папок: до 100;
- Количество треков в папке: до 255;
- Выходная мощность: 3 Вт (при 4 Ом нагрузке);
- Разрядность ЦАП: 24 бита;
- Сигнал-шум: до 85 дБ;
- Частотный диапазон: 20 Гц - 20 кГц;
- Количество уровней громкости:30;
- Режимов эквалайзера: 6 (Normal/Pop/Rock/Jazz/Classic/Base);
- Интерфейс управления: Serial Mode, AD Key Mode, I/O Mode;
- Размер модуля: 24 мм x 18 мм;

Фоторезисторы можно использовать любые, для них не существует конкретного значения сопротивления, так как оно изменяется в зависимости от

уровня освещенности. Маркировка (у меня 250 Ом) указывает на сопротивление при очень ярком свете, в темноте же оно возрастает многократно. Фоторезисторы утоплены в корпус, чтобы вне зависимости от уровня освещения пребывать в темноте. Аппаратно-программный комплекс тестировался при различных уровнях освещения и исправно работало на каждом. Для большего эффекта использовать его желательно с дымо- или парогенератором, иначе лазерных лучей не будет видно. Сопротивления фоторезистора при отсутствии света от светодиодов вкупе с сопротивлением резисторов на 4,7кОм хватает, чтобы при прерывании луча значение напряжения на соответствующих пинах было низким.

Питание на пины через фоторезисторы идет от пина 5V. Аппаратно-программный комплекс может работать как питаясь от компьютера, так и автономно. В корпус установлены 2 аккумулятора: первый питает лазерные модули, второй плату, когда на неё не поступает питание от компьютера. Оба источника напряжения включаются одним переключателем, поэтому во избежание сгорания схемы второй источник напряжения подключен через к плате диод. И лазерные модули, и плата принимают напряжение 5 вольт.

Итоговый список компонентов:

- Управляющая плата Arduino Uno;
- MP3 модуль DFPlayer Mini;
- SD-карта;
- 7 лазерных светодиодов мощностью 5мВт;
- 7 фоторезисторов с сопротивлением 250 Ом;
- 7 резисторов с сопротивлением 4,7 кОм;
- 1 резистор с сопротивлением 220 Ом;
- Диод 0.6 В;
- 2 источника напряжения 5 В.

В дальнейшем для улучшения пользовательского взаимодействия с аппаратно-программным комплексом планируется реализовать изменение уровня громкости и переключение между предустановленными (т.е. не пользовательскими) библиотеками с помощью потенциометров, подключенных к аналоговым пинам Arduino. При переключении рычажка потенциометра будет изменяться сопротивление, следовательно, на пин будет поступать разное напряжение. Напряжение от максимального до минимального сдвига рычажка потенциометра будет разбито на диапазоны, и каждому назначен определенный уровень громкости (0-30) и номер встроенной библиотеки звуков (1-4). Также возможно

использовать больше (например, 3) плееров с динамиками, что позволит формировать созвучия и аккорды, так как на данный момент единовременно может звучать только 1 звук.

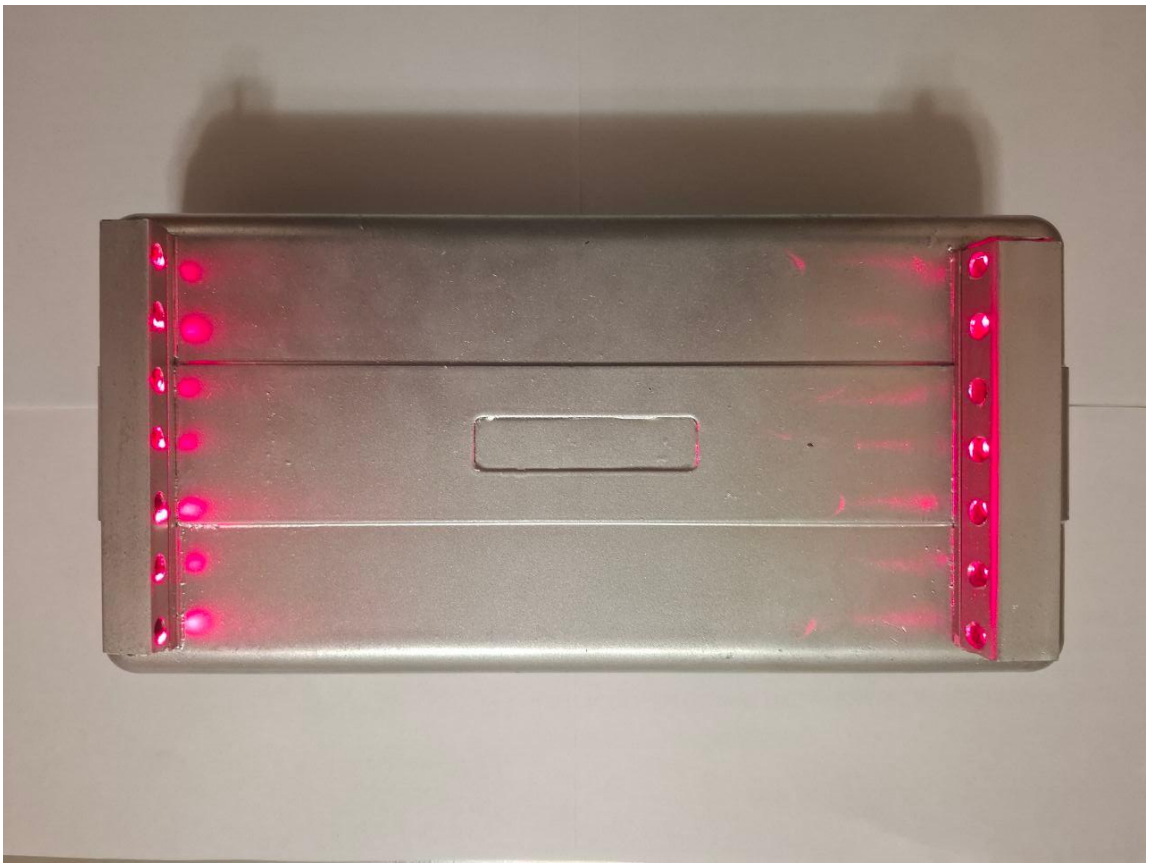

Рис. 13 Внешний вид аппаратно-программного комплекса

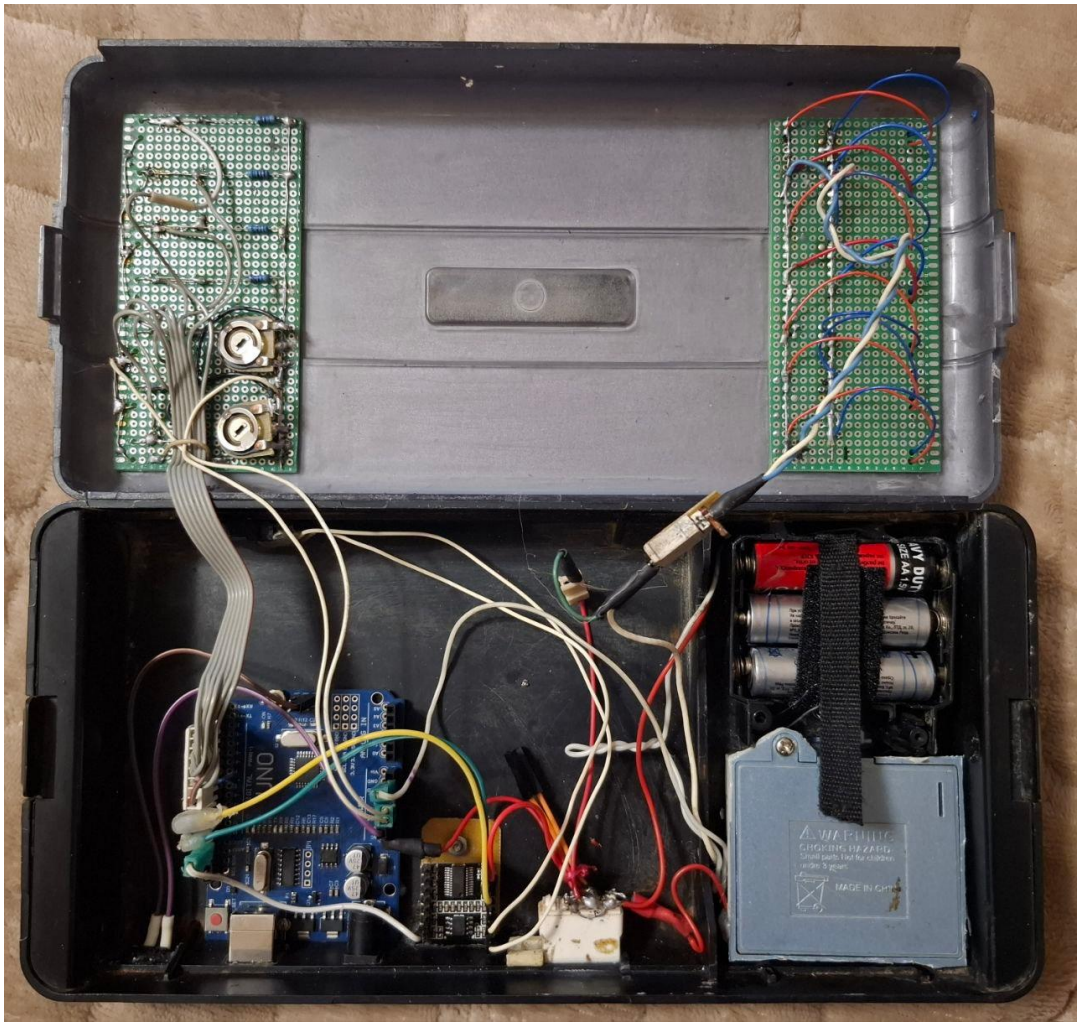

Рис. 14 Внутреннее устройство

2.3 Программная реализация

Файл music.ino:

```
#include <SoftwareSerial.h>
#include <EEPROM.h>

#define reseivePin 10
#define transmitePin 11

const int photoResistors[7] = {2, 3, 4, 5, 6, 7, 8};
uint8_t folderNumber;
SoftwareSerial dfSerial(reseivePin, transmitePin);
```

```
      void sendDFPlayerCommand(uint8_t command, uint16_t parameter) {
        uint8_t packet[10];
        packet[0] = 0x7E;
        packet[1] = 0xFF;
        packet[2] = 0x06;
        packet[3] = command;
        packet[4] = 0x00;
        packet[5] = (uint8_t)(parameter >> 8);
        packet[6] = (uint8_t)(parameter & 0xFF);
        uint16_t checksum = 0 - (0xFF + 0x06 + command + 0x00 + packet[5] +
packet[6]);
        packet[7] = (uint8_t)(checksum >> 8);
        packet[8] = (uint8_t)(checksum & 0xFF);
        packet[9] = 0xEF;
          for (int i = 0; i < 10; i++) {
          dfSerial.write(packet[i]);
        }
      }
      void playTrackInFolder(uint8_t folder, uint8_t track) {
            uint16_t parameter = ((uint16_t)folder << 8) | track;
            sendDFPlayerCommand(0x0F, parameter);
      }

      void setup() {
          Serial.begin(9600);
          dfSerial.begin(9600);

          folderNumber = EEPROM.read(0);
          if (folderNumber < 1 || folderNumber > 99) {
              folderNumber = 1;
          }
```

```
    sendDFPlayerCommand(0x06, 30);

    for (int i = 0; i < 7; i++) {
        pinMode(photoResistors[i], INPUT);
    }
}

void loop() {
    if (Serial.available() > 0) {
        int inputNumber = Serial.parseInt();
        if (inputNumber >= 1 && inputNumber <= 99) {
            folderNumber = inputNumber;
            EEPROM.write(0, folderNumber);
        }
    }

    for (int i = 0; i < 7; i++) {
        if (!digitalRead(photoResistors[i])) {
            playTrackInFolder(folderNumber, (uint8_t)i);
        }
    }
}
```

Разберём подробно код программы:

В начале программы подключаются библиотеки SoftwareSerial.h и EEPROM.h. SoftwareSerial.h используется для связи с DFPlayer Mini по виртуальному последовательному порту. EEPROM.h позволяет работать с энергонезависимой памятью микроконтроллера, в которой будет храниться последний прочитанный номер папки с аудио на случай отключения от питания или сброса.

Для создания виртуального последовательного порта в строках с определением пинов через директивы #define задаются номера пинов для приёма и

передачи (10 и 11 соответственно). После этого создаётся объект SoftwareSerial dfSerial, с помощью которого осуществляется связь с DFPlayer Mini.

Объединение пинов фоторезисторов. Для удобства пины, к которым подключены фоторезисторы (цифровые пины 2–8), объединены в массив photoResistors[]. Это позволяет быстро настроить все пины на входной режим и после работать с ними в цикле, кроме того, делает код более читаемым.

Инициализация переменных для хранения номера папки. Переменная folderNumber содержит номер папки, где находится набор звуков для воспроизведения.

Функция sendDFPlayerCommand() принимает два параметра: номер команды в соответствии с документацией на DFPlayer Mini и её параметр. В функции происходит формирование и отправка 10-байтового пакета данных для модуля плеера. Структура пакета следующая:

Байт 0: стартовый байт (0x7E).

Байт 1: версия (0xFF).

Байт 2: длина оставшейся части пакета (0x06).

Байт 3: код команды (передается через параметр command функции).

Байт 4: отсутствие обратной связи (0x00).

Байты 5–6: параметр команды, разбитый на старший (номер папки) и младший байты (номер трека), который передается через параметр parameter.

Байты 7–8: контрольная сумма, вычисляемая как отрицательная сумма значений от байта версии до младшего байта параметра, разделённая на старший и младший байты.

Байт 9: конечный байт (0xEF). После формирования массива packet все его байты последовательно отправляются в модуль DFPlayer Mini с помощью метода dfSerial.write().

Функция playTrackInFolder принимает два параметра: номер папки (folder) и номер трека (track). В ней происходит объединение этих двух значений в одно 16-битное число, где старший байт соответствует номеру папки, а младший – номеру трека. Полученное значение передается в функцию sendDFPlayerCommand с

командой 0x0F, что инициирует воспроизведение выбранного аудиотрека из указанной папки на DFPlayer Mini.

В функции setup() находятся команды, которые выполняются один раз при включении аппаратно-программного комплекса. Настройка скорости обмена данными для аппаратного и виртуального последовательного порта (9600 бит в секунду). Далее выполняется чтение значения folderNumber из EEPROM (ячейка 0). Если считанное значение не соответствует диапазону от 1 до 99 (то есть EEPROM либо пуста, либо содержит некорректное значение), переменной присваивается значение 1 по умолчанию.

В функции loop() находятся команды, которые выполняются постоянно, пока на плату подаётся питание. В бесконечном цикле происходит обработка данных, полученных через Serial. Программа проверяет, поступили ли данные через последовательный порт. Если данные получены, они считываются как число. Если введённое число находится в диапазоне от 1 до 99, оно становится новым значением переменной folderNumber. Если введённое число не проходит проверку, прежнее значение сохраняется. После проверки входных данных новое значение folderNumber записывается в EEPROM. Это позволяет при следующих запусках аппаратно-программного комплекса использовать последний корректно выбранный номер папки с библиотекой звуков. Также в бесконечном цикле программа проходит по массиву пинов фоторезисторов. Если на каком-то из фоторезисторов происходит падение напряжения, сигнал на соответствующем пине сменяется с высокого(1) на низкий(0), то вызывается функция playTrackInFolder, которая воспроизводит аудиофайл, соответствующий номеру среагировавшего пина, из папки с номером folderNumber.

ГЛАВА 3. РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ ВЗАИМОДЕЙСТВИЯ С ПЛАТОЙ

3.1 Обзор средств для разработки

Основной целью являлась разработка программного приложения, обеспечивающего двустороннюю связь между ПК и платой Arduino через последовательный интерфейс (USB/serial). Для достижения поставленной цели были решены следующие задачи:

- Реализация функции установления и разрыва соединения с платой по последовательному порту.
- Обеспечение отправки пользовательских комманд на микроконтроллер.
- Дополнительная обработка данных перед отправкой на Arduino.
- Разработка интуитивно понятного графического интерфейса пользователя для управления текущим состоянием платы.
- Использование кроссплатформенных библиотек и фреймворков для обеспечения совместимости и простоты установки.

Программный модуль разработан с использованием языка программирования Python 3.13. Ключевыми внешними зависимостями модуля являются:

1. PySerial: Библиотека Python, предназначенная для работы с последовательными интерфейсами (UART, RS-232 и др.). Она предоставляет унифицированный кроссплатформенный интерфейс, работающий как в Windows, так и в Linux/macOS, включая поддержку USB-to-Serial адаптеров. Из особенностей можно выделить поддержку синхронного и асинхронного обмена, расширенную настройку протокола соединения, возможность работы с виртуальными портами и возможность использовать pyserial-asynco для интеграции механизма асинхронных событийных очередей.
2. PyQt5: Один из более популярных фреймворков для создания графических интерфейсов. Посредством Python-API, библиотека предоставляет полный доступ к библиотеке Qt, разработанный на C++. PyQt5 позволяет создавать как простые окна с кнопками и текстовыми полями, так и сложные многооконные приложения с поддержкой вкладок, графики, OpenGL, работы с сетью, потоками и анимацией. PyQt5 поддерживает модель сигналов и слотов, которая является основной концепцией взаимодействия между компонентами, обеспечивая асинхронную реакцию на события. Также библиотека включает мощную модель событий и обработки событий

клавиатуры, мыши, таймеров и системных сообщений. Поддерживаются различные виджеты: кнопки, списки, таблицы, текстовые поля, комбинированные списки, группы переключателей, вкладки, иерархические представления данных и прочее. Для построения адаптивных интерфейсов предусмотрены механизмы компоновки.

3. PyQt5-sip: Вспомогательный модуль, необходимый для корректной работы библиотеки PyQt5. Он основан на системе генерации привязок SIP, которая позволяет связывать библиотеки, написанные на C или C++, с Python-кодом.

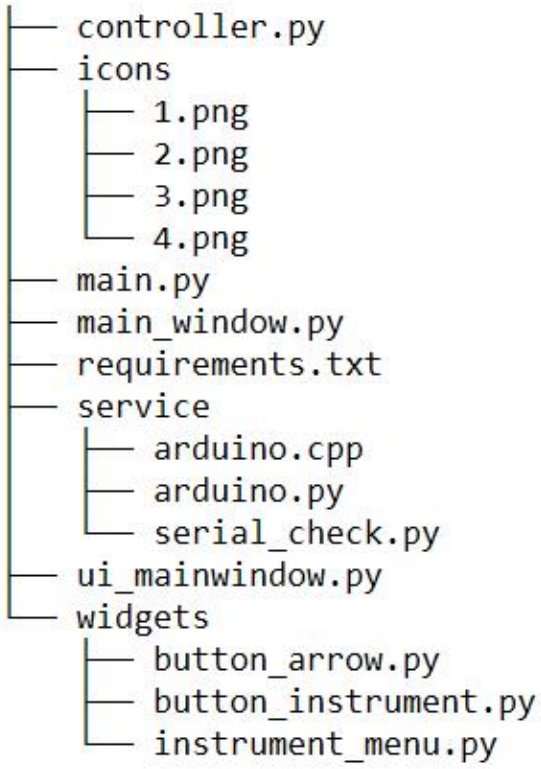

Рис.15 Файловое дерево приложения

3.2 Архитектура приложения

Архитектура разработанного программного модуля основана на принципах разделения ответственностей, что обеспечивает его модульность и расширяемость. Запуск приложения осуществляется через главный скрипт main.py, который инициализирует и отображает основное окно приложения, описанное в модуле main_window.py. Визуальное представление главного окна и его базовых компонентов определяется в файле ui_mainwindow.py. Для расширения стандартного набора элементов управления и создания более специализированного пользовательского опыта, в директории widgets содержатся

виджеты. Они интегрируются в ui_mainwindow.py для формирования конечного интерфейса.

Сервисный слой, расположенный в директории service, инкапсулирует всю логику взаимодействия с внешними устройствами и утилитарные функции. Ключевым компонентом здесь является service/arduino.py, который отвечает за установление, поддержание и управление сеансом связи с микроконтроллером Arduino через последовательный порт. Этот модуль использует библиотеку PySerial для отправки команд и приема данных от микроконтроллера. Вспомогательный модуль service/serial_check.py предоставляет функциональность для обнаружения доступных последовательных портов и проверки их состояния, что упрощает процесс подключения к Arduino. Важно отметить наличие service/arduino.cpp в этой же директории; он представляет собой исходный код прошивки для самого микроконтроллера Arduino, определяя протокол и логику работы на стороне аппаратно-программного комплекса, с которым взаимодействует Python-приложение. Хотя arduino.cpp не является частью исполняемого Python-кода, он является неотъемлемой частью общей архитектуры системы "ПК-микроконтроллер".

Графические ресурсы, используемые для оформления интерфейса, хранятся в директории icons. Зависимости проекта, необходимые для его корректной работы, перечислены в файле requirements.txt, обеспечивая воспроизводимость окружения. Архитектура модуля четко разделяет представление, управление и сервисы.

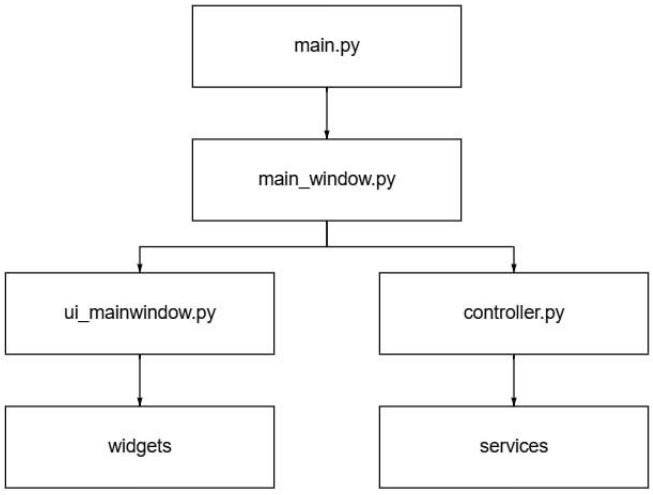

Рис. 16 Архитектура приложения

3.3 Функционал приложения

На данный момент приложение имеет следующие возможности:

1. Смена библиотеки звуков: Приложение позволяет пользователю выбрать нужную библиотеку звуков из представленного списка. При нажатии кнопки, иллюстрирующей выбранную библиотеку на Arduino отправляется сообщение, содержащее порядковый номер библиотеки, в результате чего в переменную, содержащую номер папки с треками, записывается новое значение.
2. Загрузка собственных библиотек: Пользователь может добавлять новые библиотеки звуков, импорти руя их через графический интерфейс. При этом для каждой библиотеки можно:
    - Назначить уникальное название, что делает идентификацию и поиск более удобными.
    - Загрузить обложку (cover image) для визуального представления библиотеки в списке, что повышает интуитивность интерфейса.
3. Поиск библиотеки по названию: Так как плеер позволяет иметь до 100 библиотек звуков, возможно собрать большую коллекцию, в которой будет сложно ориентироваться. Для решения этой проблемы реализован функционал поиска по названию.

В дальнейшем планируется добавить функции изменения уровня громкости и настройки эквалайзера.

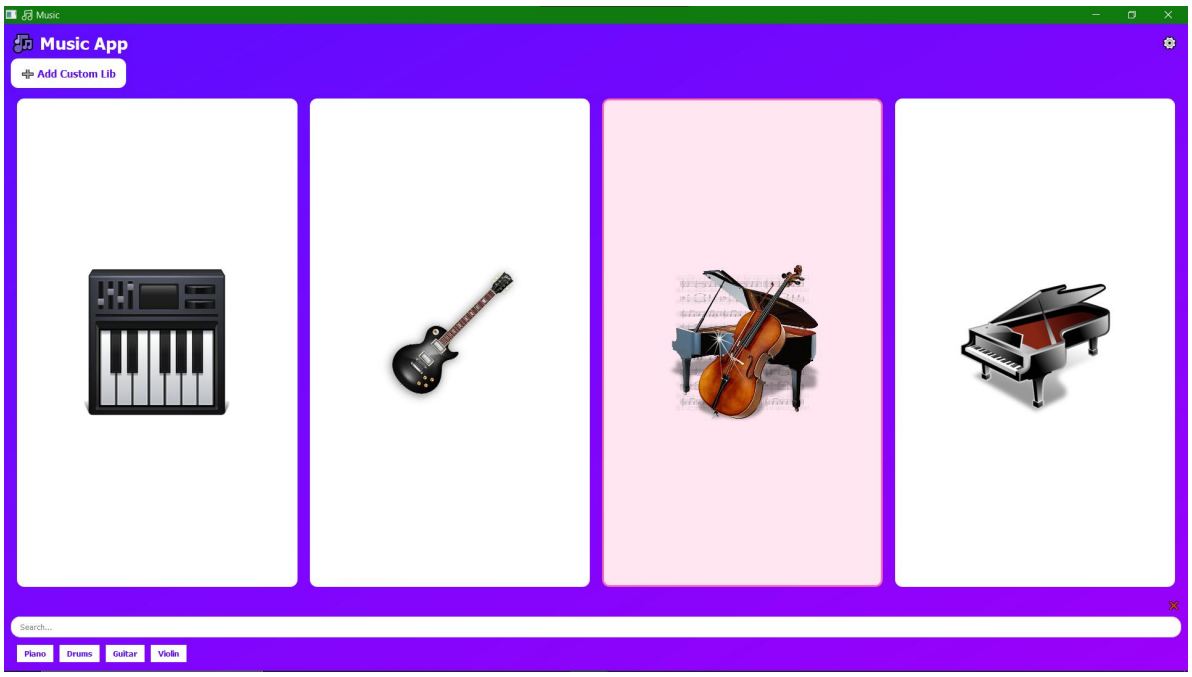

Рис. 17 Интерфейс приложения

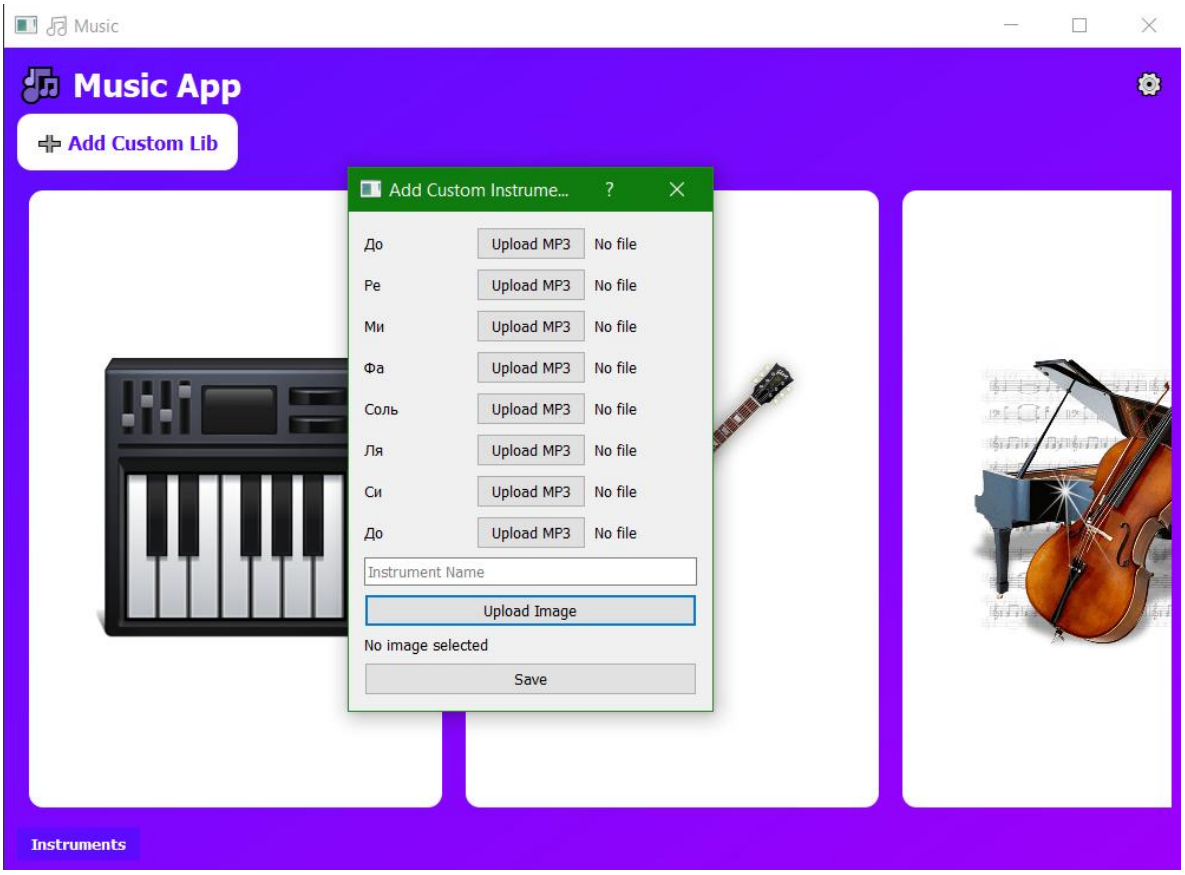

Рис. 18 Загрузка файлов для формирования пользовательской библиотеки звуков

ЗАКЛЮЧЕНИЕ

В результате выполнения дипломной работы были выполнены следующие задачи:

- Разработан аппаратно-программный комплекс для реализации музыкального интерфейса на платформе Arduino.
- Исследованы особенности платформы Arduino и выявлены её преимущества для реализации поставленной задачи, исследованы возможности модуля MP3-плеера DFPlayer Mini.
- Определены аспекты существующих реализаций, которые работа улучшает.
- Проанализированы и выбраны программные средства для разработки.
- Изучен и применен на практике язык программирования Arduino.
- Построены дальнейшие планы по модернизации существующего аппаратно-программного комплекса.

При разработке проекта были использованы следующие средства: сервис Circuit Diagram - A Circuit Diagram Maker для создания принципиальной схемы, графический редактор AutoCAD 2024 для создания чертежа корпуса устройства, среда разработки Arduino IDE для реализации программной части и прошивки, среда разработки Visual Studio Code, язык программирования Python 3.13, его библиотеки PyQt5, PySerial, микроконтроллерная плата Arduino Uno.

Были определены следующие перспективы развития аппаратно-программного комплекса:

1. Интеграция Bluetooth-модуля, перенос управления на мобильное устройство, что позволит пользователям удалённо контролировать аппаратно-программный комплекс, повысит его мобильность и удобство использования.
2. Замена Arduino на современные микроконтроллеры, такие, как STM32, может обеспечить более сложную обработку аудиосигналов, поддержку цифровых эффектов и возможность работы с большим количеством входных/выходных сигналов. Также возможно переписать код с языка Arduino на C/C++ для оптимизации.
3. Устройство со схожим принципом работы можно использовать в охранных системах или как датчик в системе «умный дом», включая определённые сигналы при прерывании лазерного луча.

4. Также аппаратно-программный комплекс можно использовать для обучения маленьких детей, загружая библиотеки с алфавитом, цифрами, нотами, звуками животных и т.п. Эстетика лазерного света может привлечь внимание ребенка, а возможность загружать пользовательские звуки открывает множество возможностей для обучения.
5. При увеличении размеров корпуса и с использованием дымо- или парогенераторов аппаратно-программный комплекс может использоваться на концертах, выставках, вечеринках и других мероприятиях в качестве музыкального инструмента или интерактивного объекта, с которым может взаимодействовать аудитория.

Данная работа служит основой для дальнейших исследований и разработок в области музыкальных технологий.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Монк С. Программируем Arduino. Профессиональная работа со скетчами / пер. с англ. А. Макарова. — СПб.: Питер, 2017. — 256 с.
2. Среда разработки Arduino | Аппаратная платформа Arduino [Электронный ресурс]. — Режим доступа: https://arduino.ru/Arduino_environment
3. Хоровиц П., Хилл У. Искусство схемотехники: в 3 т. Т. 1. 4-е изд., перераб. и доп. / пер. с англ. Б. Н. Бронина, А. И. Коротова, М. Н. Микшиса, О. Л. Соболевой. — М.: Мир, 1993. — 418 с.
4. Техническая документация DFPlayer Mini [Электронный ресурс]. — Режим доступа: Microsoft Word - MP3－TF－16P V1.0_en.doc
5. Arduino Software (IDE) | Arduino [Электронный ресурс]. — Режим доступа: https://www.arduino.cc/en/Guide/Environment
6. Arduino Laser Harp : 10 Steps (with Pictures) - Instructables by yaroshka [Электронный ресурс]. — Режим доступа: https://www.instructables.com/Arduino-Laser-Harp-1/
7. Arduino Uno [Электронный ресурс]. — Режим доступа: http://m-elek.h1n.ru/about/project_files/arduino-new/vvedenie/arduino-uno.html
8. Building LASER Harp with an Arduino by ScienceShack [Электронный ресурс]. — Режим доступа: https://www.youtube.com/watch?v=pddGR5Eyc0Q&ab_channel=ScienceShack
9. PySerial’s documentation [Электронный ресурс]. — Режим доступа: https://pythonhosted.org/pyserial/
10. PySide6.QtCore - Qt for Python [Электронный ресурс]. — Режим доступа: https://doc.qt.io/qtforpython-6/PySide6/QtCore/index.html#module-PySide6.QtCore
11. UNO R3 | Arduino Documentation [Электронный ресурс]. — Режим доступа: https://docs.arduino.cc/hardware/uno-rev3/
12. Void loop () и void setup () в Arduino скетче. Примеры и описание [Электронный ресурс]. — Режим доступа: https://arduinomaster.ru/program/arduino-void-loop-i-void-setup/?ysclid=m4ul8ei4fs517762664
13. What is Arduino? | Arduino [Электронный ресурс]. — Режим доступа: https://www.arduino.cc/en/Guide/Introduction