МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

Кафедра информационных систем управления

ВЕРЕНИЧ ВЛАДИСЛАВ НИКОЛАЕВИЧ

**ТЕХНОЛОГИИ РАЗРАБОТКИ ПО РОБОТИЗИРОВАННЫХ СИСТЕМ НА ПЛАТФОРМЕ АРДУИНО**

Курсовой проект

студента 4 курса 12 группы

| “Допустить к защите“  Зав. кафедройИСУ  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  “\_\_\_” \_\_\_\_\_\_\_\_\_\_\_ 2023 г.  Минск 2023 | **Руководитель**  Коновалов Олег Леонидович,  к.т.н. |
| --- | --- |
|  |  |

**ОГЛАВЛЕНИЕ**

[**ВВЕДЕНИЕ 3**](#_f3izha18izkj)

[**Основные определения 4**](#_7jcr0canqxv2)

[**Глава 1. ПОСТАНОВКА ЗАДАЧ 5**](#_iz773t82lulj)

[1.1. Цель и задачи исследования 5](#_ykcr2srrtm0z)

[**Глава 2. ОБЗОР ПЛАТФОРМЫ ARDUINO 6**](#_yypx4dq3syp)

[2.1. Преимущества и ограничения использования Arduino для разработки роботизированных систем 6](#_drc6t3kal4c)

[**Глава 3. ОПИСАНИЕ ЯЗЫКА ПРОГРАММИРОВАНИЯ ARDUINO 7**](#_dmqlalylz5ac)

[3.1. Подключение и настройка Arduino-платы 7](#_awe4npuof1e2)

[3.2. Программирование с использованием Arduino IDE. Основные операции взаимодействия с платформой (ввод/вывод, работа с датчиками и актуаторами и т.д.) 10](#_or8y69hzfidw)

[3.3. Анализ преимуществ и ограничений платформы Arduino для разработки роботизированных систем 12](#_owhvx0twqnzd)

[**Глава 4. РАЗРАБОТКА ПО ПААНК НА ПЛАТФОРМЕ АРДУИНО 14**](#_qysxslydirsk)

[4.1. Проектирование и разработка аппаратной части ПААНК на Arduino 14](#_j7ddaaql76op)

[4.2. Программирование и разработка проекта Arduino для ПААНК 15](#_ea496bjt1fqt)

[4.3. Интеграция с дополнительными модулями 24](#_b6t860b52ckt)

[4.4. Управление ПААНК с помощью инструмента Serial Monitor 26](#_u94fzue613yg)

[**Глава 5. СРАВНЕНИЕ С ДРУГИМИ МИКРОКОНТРОЛЛЕРАМИ И ПЛАТФОРМАМИ 27**](#_4iuend3yk7ig)

[5.1. Основные семейства микроконтроллеров 27](#_cmnylraq0c1o)

[5.2. Преимущества и недостатки Arduino в контексте конкретных задач 30](#_xld5hshz2em9)

[5.3. Заключение 31](#_vn74jbjha2qa)

[**Заключение 32**](#_lnehdamo410w)

[Список использованных источников 33](#_7d0bjt8v24gj)

# 

# ВВЕДЕНИЕ

## 

Что такое Ардуино? Arduino — это электронная платформа с открытым исходным кодом, которая имеет простую в использовании физическую программируемую плату и программное обеспечение. Сегодня это одна из самых популярных систем для создания роботов и систем «умного дома» в отрасли.

Хотя некоторые из этих [проектов Arduino](https://www.lifewire.com/arduino-project-ideas-2495311) могут показаться легкомысленными, технология учитывает несколько тенденций, которые сделают ее потенциально важной силой в отрасли. Интернет вещей (IoT) — популярное словосочетание, используемое в техническом сообществе для описания повседневных предметов, подключенных к Интернету и способных обмениваться информацией. Интеллектуальные счетчики энергии являются часто используемым примером, который может регулировать использование приборов для экономии денег на энергии.

Общественное восприятие смещается в сторону интеграции технологий в ткань повседневной жизни. Небольшой форм-фактор Arduino позволяет применять его ко всем видам повседневных объектов. Фактически, форм-фактор Arduino LilyPad позволяет использовать носимые устройства Arduino.

Проекты с открытым исходным кодом, такие как Arduino, снижают входной барьер для разработчиков, желающих экспериментировать с интерактивными объектами. Эти новаторы смогут быстро создавать прототипы и экспериментировать с интерактивными устройствами, используя платформу Arduino, прежде чем создавать готовое к производству предложение.

Именно поэтому я и решил рассмотреть данную электронную платформу.

В данной работе рассмотрена реализация ПО ПААНК(Портативный автоматический анализатор нуклеиновых кислот). Данное инновационное решение будет являться новым направлением в области портативных приборов для лабораторной диагностики – Мини-лабораторией, сопоставимой по функционалу с обычной диагностической лабораторией с точки зрения возможностей проведения генетических тестов и иммуноферментных исследований, работающей на базе единого Анализатора с использованием различных тест-систем в виде одноразовых картриджей с заранее внесенными при производстве реактивами.

# 

# 

# Основные определения

**ПААНК** - Портативный автоматический анализатор нуклеиновых кислот

# Глава 1. ПОСТАНОВКА ЗАДАЧ

## 1.1. Цель и задачи исследования

Цели:

1. Изучить основы разработки роботизированных систем на платформе Arduino.

2. Исследовать возможности и ограничения использования Arduino для разработки робототехнических проектов.

3. Разработать систему управления портативным автоматическим анализатором нуклеиновых кислот на базе Arduino.

Задачи:

1. Провести обзор литературы и источников, касающихся платформы Arduino и разработки роботизированных систем.

2. Изучить архитектуру и основные компоненты платформы Arduino.

3. Определить типовые задачи и приложения робототехники, в которых можно применить Arduino.

4. Изучить язык программирования Arduino и основные операции взаимодействия с платформой.

5. Подготовить среду разработки, установить и настроить необходимые программные инструменты для работы с Arduino.

6. Изучить возможности подключения и управления датчиками и актуаторами на платформе Arduino.

7. Разработать аппаратную часть робота, выбрав необходимые компоненты и создавая схемы подключения.

8. Программировать функциональность робота на Arduino, включая управление движением, сенсорными данными и коммуникацию.

9. Протестировать и отладить разработанный робот, проверить его работоспособность и соответствие поставленным требованиям.

10. Проанализировать преимущества и ограничения платформы Arduino в контексте разработки роботизированных систем.

11. Описать результаты исследования и разработки в курсовой работе, включая описание проекта, использованные методы и полученные результаты.

12. Сделать выводы о применимости Arduino для разработки робототехнических проектов и предложить рекомендации по дальнейшему улучшению и развитию платформы.

# Глава 2. ОБЗОР ПЛАТФОРМЫ ARDUINO

## 2.1. Преимущества и ограничения использования Arduino для разработки роботизированных систем

Итак, Arduino можно использовать для различных проектов робототехники: от простых роботов, следующих по линиям, до сложных автономных роботов. Широкий спектр компонентов и инструментов программирования Arduino делает его хорошо подходящим для проектов робототехники. Arduino можно использовать для управления роботизированными руками, создания автономных роботов и роботов, следующих по линиям. Arduino также используется для создания мобильных роботов, таких как самобалансирующиеся роботы и роботы, избегающие препятствий.

Однако, использование Arduino для проектов робототехники имеет как преимущества, так и недостатки. Вот некоторые преимущества и недостатки использования Arduino для робототехники.

Преимущества:

* Низкая стоимость: платы Arduino относительно недороги, что делает их доступными для любителей и студентов.
* Простота использования: Arduino проста в использовании, доступно множество онлайн-документации и учебных пособий.
* Широкий выбор компонентов: Arduino имеет широкий спектр компонентов, таких как датчики, двигатели и приводы, которые можно использовать для создания различных роботов.
* Открытый исходный код: Arduino — это платформа с открытым исходным кодом, что означает, что пользователи имеют доступ к исходному коду и могут изменять его в соответствии со своими потребностями.

Недостатки:

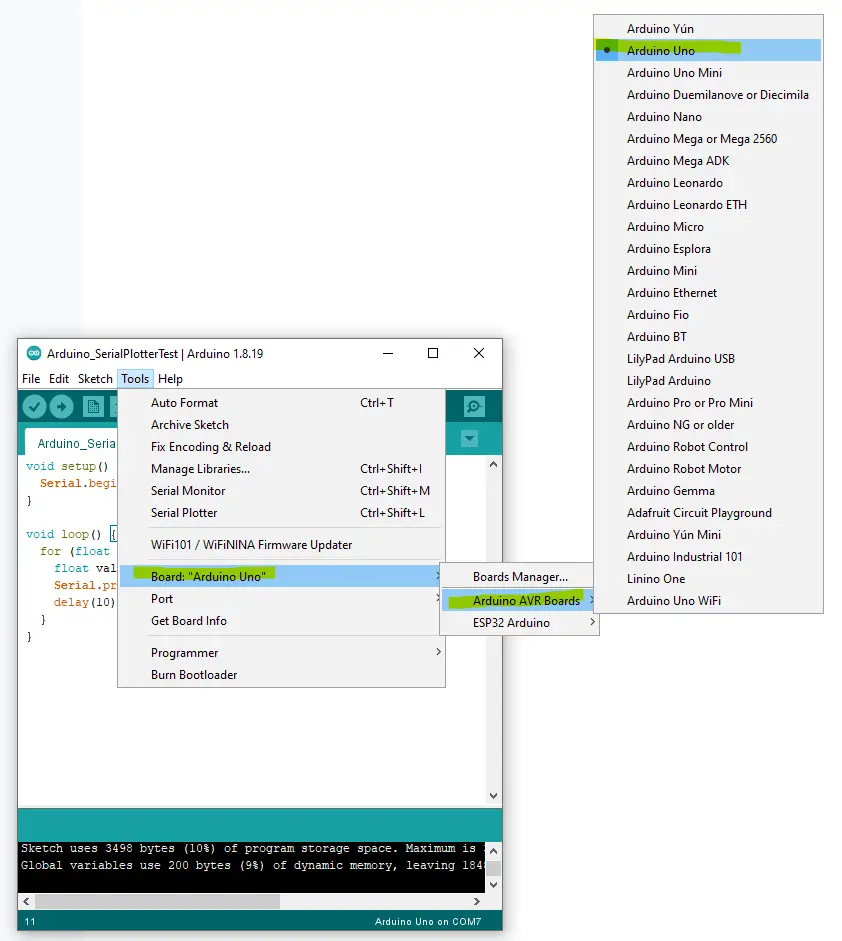
* Ограниченная вычислительная мощность: Платы Arduino ограничены с точки зрения вычислительной мощности, что может ограничить сложность роботов, которые можно построить.
* Ограниченное хранилище: платы Arduino имеют ограниченное хранилище, что может ограничить объем данных, которые могут храниться на плате.
* Отсутствие поддержки: платы Arduino не имеют технической поддержки, поэтому пользователям приходится обращаться за помощью к онлайн-форумам и учебным пособиям.

# Глава 3. ОПИСАНИЕ ЯЗЫКА ПРОГРАММИРОВАНИЯ ARDUINO

## 3.1. Подключение и настройка Arduino-платы

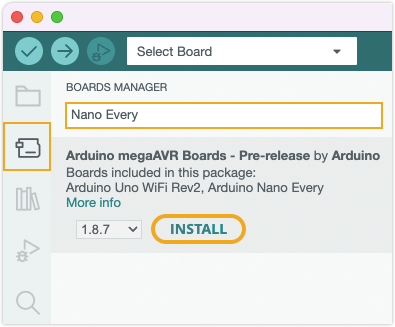
Выбор платы Arduino в IDE

Для этого необходимо в Arduino IDE, перейтив меню **«Инструменты** » и выбрать подменю **«Плата»** , в котором будут показаны все доступные платы Arduino, которые IDE поддерживает по умолчанию.



*Рис. 3.1. Выбор платы Ардуино*

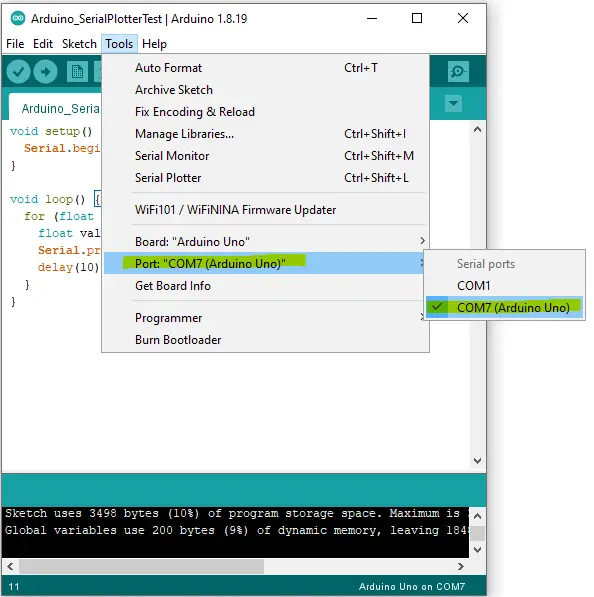
Если же в списке не будет нужной платы, всегда можно воспользоваться Менеджером плат и найти необходимую. Он находится в Tools меню под Board: "<Selected Board>",Boards Manager…



*Рис. 3.2. Boards Manager in Arduino IDE 2.*

Выбор последовательного порта Arduino

Далее всегда нужно убедиться, что выбран правильный порт(COMx для ОС Windows, ttyACMx или ttyUSBx для ОС Linux) для используемой платы Arduino. В противном случае будут возникать постоянные проблемы с загрузкой прошивки при прошивке кода на Arduino.



*Рис. 3.3. Выбор последовательного порта Arduino*

## 3.2. Программирование с использованием Arduino IDE. Основные операции взаимодействия с платформой (ввод/вывод, работа с датчиками и актуаторами и т.д.)

**API Ардуино**

Язык программирования Arduino (Arduino API) — это вариант языка программирования C++ с множеством встроенных функций и библиотек, которые можно использовать прямо в среде Arduino IDE.

Документация [по API Arduino](https://www.arduino.cc/reference/en/) действительно помогает начать работу с Arduino и узнать больше об основных функциях и строительных блоках, которые можно использовать для создания собственных проектов.

**Скетчи Ардуино**

В терминологии Arduino искодный код называется « **скетч** ». Это основной файл проекта с расширением . ino , и он должен находиться в папке с таким же именем. В проекте может быть несколько файлов в одной папке, например, файлы заголовков или что-то еще.

**Структура программы Arduino**

Любая программа Arduino должна иметь как минимум следующие две фундаментальные функции:

* void setup ( ) Эта функция выполняется только один раз во время загрузки Arduino. Независимо от того, было ли оно только что включено или сброшено, контроллер Arduino сразу же начнет выполнять логику внутри этой функции, а затем перейдет к функции цикла(). Обычно она используется для инициализации библиотек или переменных.
* void loop ( ) Это основная логика вашего приложения, которая будет выполняться в цикле постоянно.

Обе вышеуказанные функции должны присутствовать в каждом скетче Arduino, даже если он будет просто пустой. Это минимальные требования для скетча Arduino.

// the setup function that runs only once at start-up

void setup() {

}

// the loop function runs over and over again forever

void loop() {

}

## 

Вот пример из альбома скетчей Arduino. Это пример мигания светодиода, который переключает светодиод, подключенный к встроенному выходному контакту светодиода (контакт ввода-вывода 13).

// the setup function runs once when you press reset or power the board

void setup() {

// initialize digital pin LED\_BUILTIN as an output.

pinMode(LED\_BUILTIN, OUTPUT);

}

// the loop function runs over and over again forever

void loop() {

digitalWrite(LED\_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)

delay(1000); // wait for a second

digitalWrite(LED\_BUILTIN, LOW); // turn the LED off by making the voltage LOW

delay(1000); // wait for a second

}

## 3.3. Анализ преимуществ и ограничений платформы Arduino для разработки роботизированных систем

Приступая к работе с Arduino, необходимо узнать, что такое **датчики** , **исполнительные механизмы** и **драйверы** . И знать различные типы и варианты использования каждого из них.

Датчик — это электронное устройство, которое преобразует физический сигнал или величину в электрический сигнал, который может быть аналоговым или цифровым. Большинство сигналов являются аналоговыми по своей природе, и датчики предназначены для преобразования этих величин в электрические аналоговые сигналы. Существует очень много типов датчиков, которые можно использовать в проектах по электронике, например:

* Датчики света
* Датчики температуры, влажности и дождя
* Датчики обнаружения движения
* Датчики расстояния и обнаружения объектов
* IMU (инерционные единицы измерения)
* Медицинские датчики
* Датчик химических газов
* Датчики силы, изгиба и веса

Актуатор — это устройство, которое используется для приведения в действие или изменения физического состояния. Его можно рассматривать как преобразователь электрической энергии в другие формы энергии (свет, тепло, движение, звук и т. д.). Примеры приводов включают, помимо прочего, следующее:

* Световые блоки (например, светодиоды)
* Колонки
* Двигатели (постоянного тока, сервоприводы, шаговые двигатели и т. д.)
* Соленоиды/клапаны
* Реле (Контакторы)
* Электромагниты
* Нагреватели/охладители

Существует очень много исполнительных механизмов, которые мы можем использовать для выполнения действий в окружающей среде с помощью микроконтроллера Arduino. Обычно приводы представляют собой тяжелую нагрузку, которой для работы или выполнения выполняемого действия требуется значительный ток.

Ток нагрузки привода всегда превышает возможности цифровых выводов ввода-вывода микроконтроллера Arduino. Вы просто не можете управлять двигателем постоянного тока 1 А напрямую с помощью цифрового контакта ввода-вывода, максимальный безопасный предел которого обычно составляет около 20 или 25 мА. Поэтому нам нужны так называемые «Драйверы» для управления тяжелыми нагрузками (например, приводами) с помощью Arduino.

Драйвер может быть как простым, например, одним транзистором (BJT или MOSFET), так и более сложной специализированной платой для управления определенным типом нагрузки, например:

* **H-мосты** : для привода двигателей постоянного тока
* **ESC** : для вождения бесщеточных двигателей
* **Платы шагового драйвера** : для управления шаговым двигателем
* **Драйверы RGB-лент** : для управления и управления светодиодными лентами RGB
* **Инверторные мосты** : для систем PMSM или преобразователей.
* **Усилители мощности звука** : для громкоговорителей

В робототехнике Arduino в основном используется для управления двигателями, датчиками и исполнительными механизмами. Двигатели используются для управления движением робота и могут управляться с помощью сигналов ШИМ (широтно-импульсной модуляции) Arduino. Датчики используются для обнаружения изменений в окружающей среде и предоставления входных данных для Arduino, который затем может использовать эту информацию для управления поведением робота.

Таким образом, использование микроконтроллера Arduino подходит для управления роботами, и именно поэтому широко данный микроконтроллер широко применяется в области робототехники.

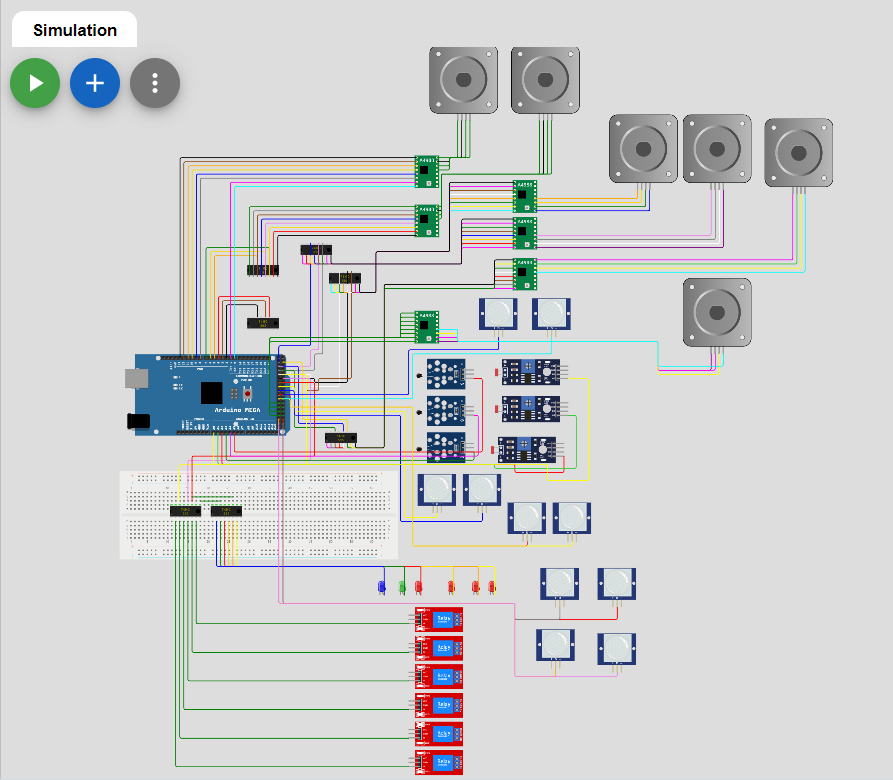
## 

# 

# Глава 4. РАЗРАБОТКА ПО ПААНК НА ПЛАТФОРМЕ АРДУИНО

## 4.1. Проектирование и разработка аппаратной части ПААНК на Arduino

Аппаратная часть робота разрабатывалась нашей командой достаточно длительное время и полную её схему размещать здесь не рационально. Именно поэтому я приведу упрощенную схему аппаратной части робота, содержащую все датчики, исполнительные механизмы и драйверы. А также распиновку микроконтроллера.



Изучить подробно составленную мной упрощенную схему можно по ссылке: <https://wokwi.com/projects/383366768675609601>

## 4.2. Программирование и разработка проекта Arduino для ПААНК

**Замечания**

Для реализации данного проекта будет использована Arduino Due — первая плата Arduino, основанная на 32-битном микроконтроллере с ядром ARM.

**Требования к ПО**

Необходимо реализовать систему для управления приведенной выше схемой с помощью микроконтроллера Arduino. Система должна позволять управлять всеми актуаторами выше параллельно. Учесть, что Arduino Due не поддерживает ни многопоточность, ни многопроцессорность на аппаратном уровне.

**Описание работы**

Необходимо отметить, что представленная выше схема максимально упрощена для наглядности! На самом деле практически все представленные выше актуаторы входят в состав более сложных устройств и их компонентов.

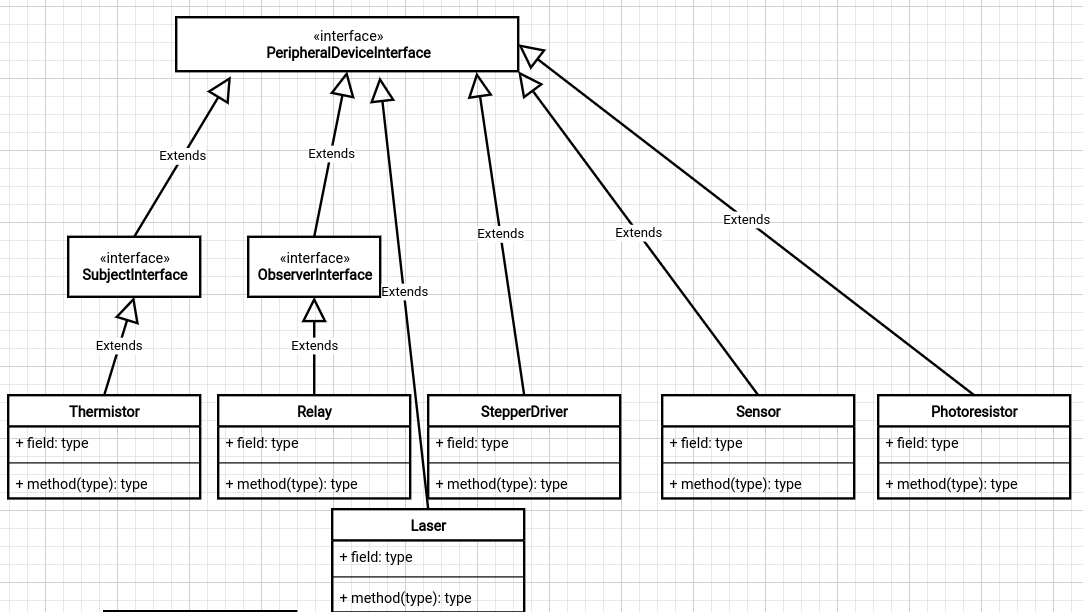
Для полного понимания необходимо представить весь список элементарных компонентов системы:

* Регистр сдвига(об этом актуаторе подробнее в п. 4.3)
* Датчик
* Шаговый двигатель(вместе с драйвером)
* Лазер
* Реле
* Термистор

Итак, мной были определены следующие классы-сущности соответственно:

* ShiftRegister74HC595(подробнее в п. 4.3)
* Sensor
* StepperDriver
* Laser
* Relay
* Thermistor

В результате я получил следующую иерархию:



*Рис. 4.1. Иерархия классов элементарных компонентов системы*

Теперь представим список составных элементов системы(состоят из нескольких элементарных компонентов):

* Поршень(Состоит из одного шагового двигателя и двух датчиков)

Поршень представляет собой систему, способную перемещать шток на определенные расстояния в миллиметрах. Это осуществляется за счет движения двигателя. Объект класса Piston.

* Позиционируемый двигатель выбора позиции (Состоит из одного шагового двигателя и одного датчика)

Позиционируемый двигатель выбора позиции предназначен для выбора реакторного отсека. Реактор представляет собой n отсеков. Причем перемещение в позицию k означает поворот вала двигателя на угол от некоторого нулевого положения. Этому нулевому положению соответствует сигнал от датчика. Объект класса PositionedStepperZoned.

* Позиционируемый двигатель для выбора цвета лазера(Состоит из одного шагового двигателя и одного датчика)

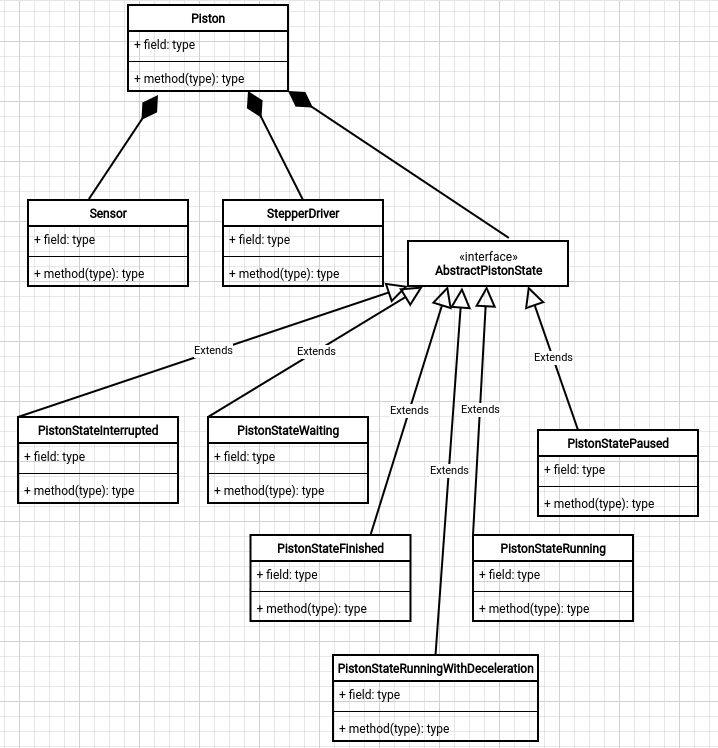
Позиционируемый двигатель для выбора цвета лазера работает как и позиционируемый двигатель выбора позиции, но служит для переключения цвета лазера путем поворота на некоторый угол, а не для поворота в определенный отсек реактора, имеет иные алгоритмы движения. Объект класса PositionedStepperLaser.

* Позиционируемый двигатель для выбора канала работы(Состоит из одного шагового двигателя и одного датчика)

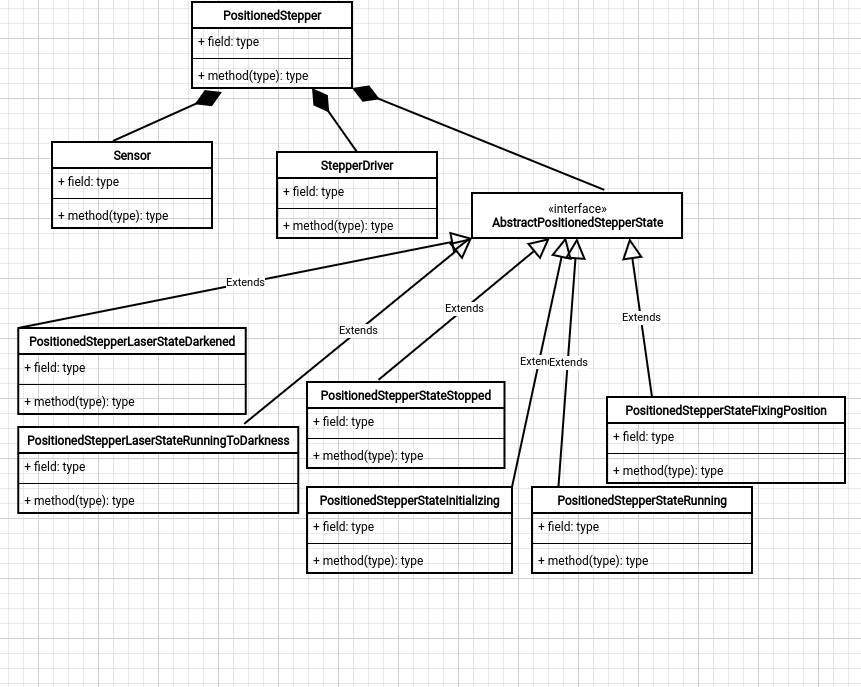
Позиционируемый двигатель для выбора цвета лазера работает как и позиционируемый двигатель выбора позиции, но перемещается по различным каналам, а не отсекам реактора, имеет иные алгоритмы движения. Объект класса PositionedStepper.

* Двигатель для перемещения магнита(Состоит из одного шагового двигателя и двух датчиков)

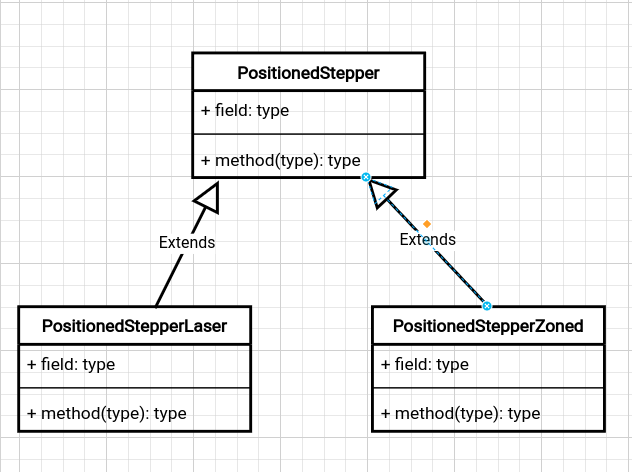
Двигатель для перемещения магнита способен находится в остановленном состоянии только в двух определенных точках: верхнее и нижнее положение. Этим точкам соответствуют датчики верхнего и нижнего положения соответственно. Объект класса SensoredStepper.



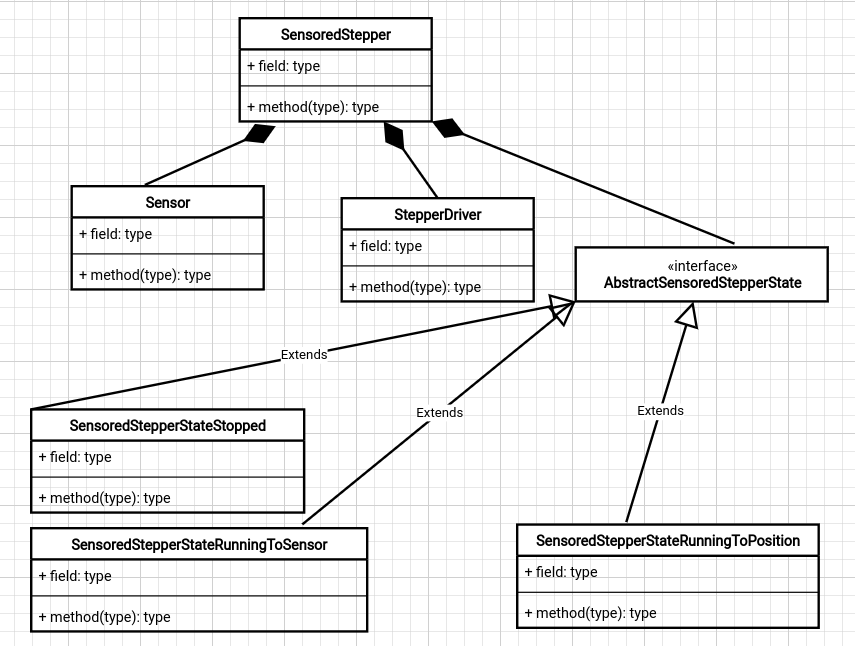
*Рис. 4.2. UML-диаграмма отношений для класса Piston (классы Sensor и StepperDriver представлены отдельно для наглядности)*



*Рис. 4.3. UML-диаграмма отношений для класса PositionedStepper (классы Sensor и StepperDriver представлены отдельно для наглядности)*

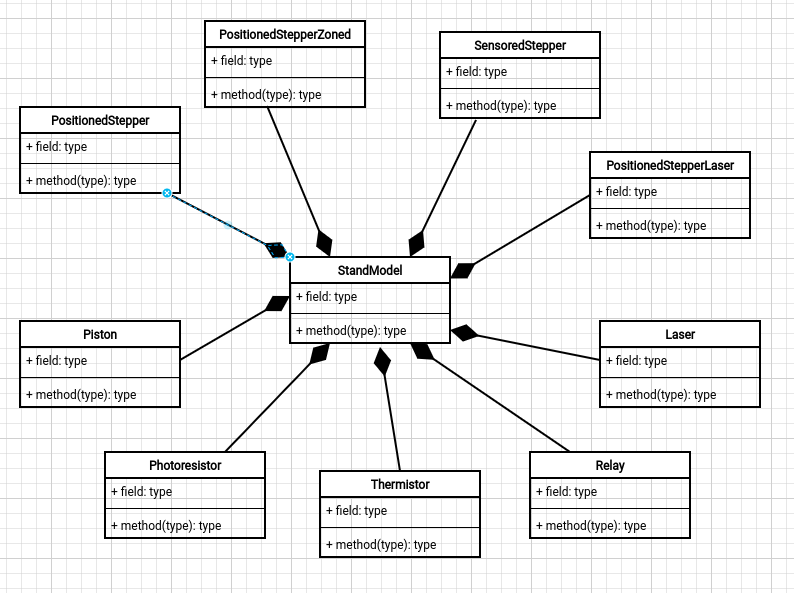


*Рис. 4.4. Иерархия классов-наследников класса PositionedStepper (класс PositionedStepper представлен отдельно для наглядности)*



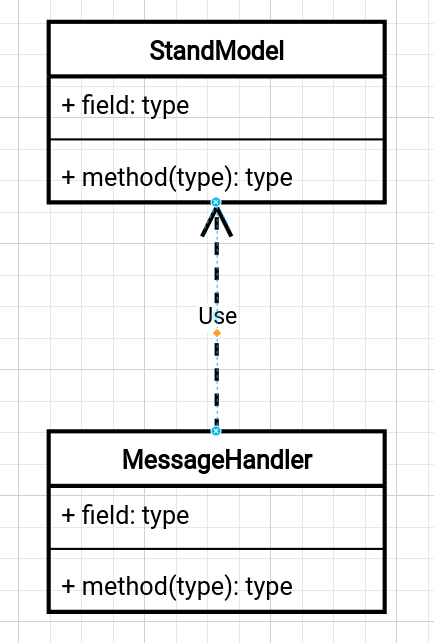
*Рис. 4.5. UML-диаграмма отношений для класса SensoredStepper (классы Sensor и StepperDriver представлены отдельно для наглядности)*

Теперь необходимо определить класс системы непосредственно. Это класс StandModel. Фактически он включает в себя составные компоненты и элементарные компоненты системы.



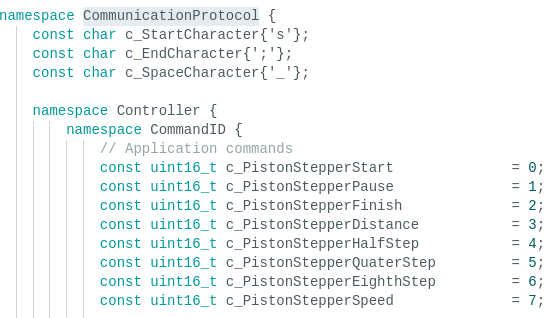
*Рис. 4.6. UML-диаграмма отношений для класса StandModel (классы PositionedStepperZoned, SensoredStepper, PositionedStepperLaser, Laser, Relay, Thermistor, Photoresistor, Piston представлены отдельно для наглядности)*

Также необходимо определить класс MessageHandler. Этот класс зависит от класса StandModel. Он необходим для управления классом StandModel и каждым компонентом в отдельности путем отправки запросов к этому классу.



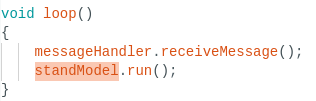
*Рис. 4.7. UML-диаграмма отношений классов StandModel и MessageHandler.*

Класс MessageHandler обрабатывает запросы на основании пространства имен CommunicationProtocol в котором определены ID команд и символьные константы для формирования команд. Таким образом, он позволяет осуществлять управление системой с помощью встроенного в Arduino IDE последовательного монитора сразу после загрузки скетча в плату!



*Рис. 4.8. Фрагмент кода определения пространства имен CommunicationProtocol.*

Стоит отметить еще одну очень важную деталь, значительно влияющую на архитектуру подобной системы. Как уже было сказано ранее, обязательными функциями скетча Arduino являются функции setup() и loop(). Причем ф-ция setup() служит для настройки параметров, а ф-ция loop() представляет собой бесконечный цикл. Как уже было сказано в требованиях, все компоненты ситемы должны работать параллельно, то есть, система должна иметь возможность отвечать на любой запрос в любой момент времени без задержки. Именно поэтому в ф-ции MessageHandler::receiveMessage()(проверяет наличие запроса к системе, если запрос есть, обрабатывает его) и StandModel::run()(поддерживает работу системы) выполняются каждую итерацию цикла.



*Рис. 4.9. Функция loop().*

## 4.3. Интеграция с дополнительными модулями

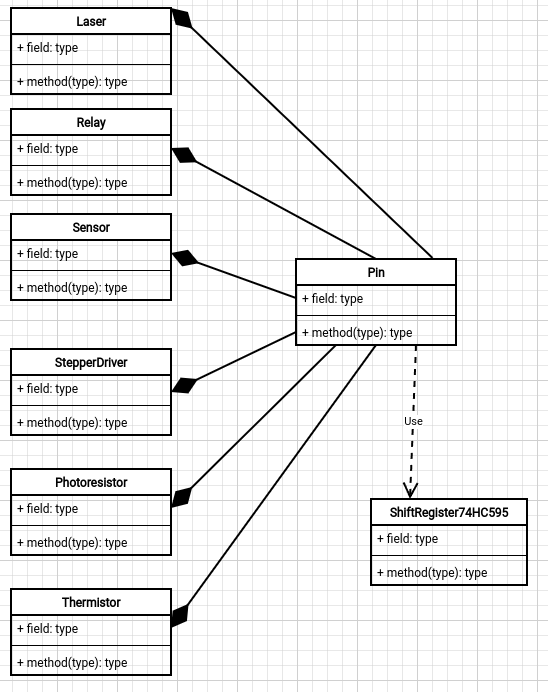
В какой-то момент времени можно столкнуться с проблемой отсутствия достаточного количества контактов на ардуино для удовлетворения потребностей проекта или прототипа. Решение этой проблемы? Сдвиговый регистр, а точнее Arduino сдвиговый регистр 74hc595. Да, с такой проблемой столкнулась и моя команда в определенный момент времени и необходимо было адаптировать все существующие классы под новый вид контакта: контакт сдвигового регистра.

Фактически каждый элементарный компонент системы до использования сдвигового регистра был присоединен к контактам ардуино и делалось это просто указанием номера контакта, например, следующим образом:



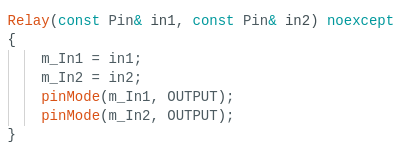
*Рис. 4.10. Определение объекта класса AccelStepper.*

После введения сдвигового регистра и определения для него собственного класса необходимо было использовать новый класс Pin, который вторым необязательным параметром конструктора принимает указатель на объект сдвигового регистра, а первым параметром принимает номер контакта. Если второй параметр игнорируется, то считается, что контакт принадлежит Ардуино, а не регистру.

**

*Рис. 4.11. UML-диаграмма отношений для класса Pin и ShiftRegister74HC595 (классы Sensor, StepperDriver, Laser, Relay, Thermistor, Photoresistor, Piston представлены отдельно для наглядности)*

После этого конструктор класса элементарного компонента и его определение выглядели следующим образом:



*Рис. 4.12. Определение конструктора класса Relay.*



*Рис. 4.13. Определение объекта класса Relay.*

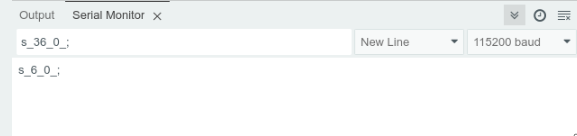
## 4.4. Управление ПААНК с помощью инструмента Serial Monitor

## 

На самом деле все запросы, отправляемые к Arduino, отправляются и должны отправляться от десктопного приложения StandBioMS, которое позволяет управлять аппаратной частью ПААНК с помощью элементарных контролов(кнопок, слайдеров, полей для ввода текста и т.д.). Однако, как было сказано в п. 4.2. управление может осуществляться и с помощью инструмента Serial Monitor Arduino IDE.

Итак, согласно определенному пространству имен CommunicationProtocol, запрос имеет следующую структуру: |c\_StartCharacter |c\_SpaceCharacter |CommandID |c\_SpaceCharacter |CommandValue |c\_SpaceCharacter |c\_EndCharacter |. Префикс “c\_” говорит о том, что значение символа константа. Следовательно, команду можно определить уникально парой (CommandID, CommandValue). Ответ на запрос имеет аналогичную структуру.

Вот пример получения ID состояния поршня:



*Рис. 4.14. Управление Arduino с помощью инструмента SerialMonitor.*

# Глава 5. СРАВНЕНИЕ С ДРУГИМИ МИКРОКОНТРОЛЛЕРАМИ И ПЛАТФОРМАМИ

## 5.1. Основные семейства микроконтроллеров

В настоящий момент времени на рынке представлено большое количество микроконтроллеров. Именно поэтому необходимо рассмотреть разницу между основными семействами микроконтроллеров: AVR, ARM, 8051 и PIC.

|  | ESP | PIC | AVR | ARM |
| --- | --- | --- | --- | --- |
| Разрядность | 832бит | 8/16/32 бит | 8/32 бит | 32 бит, иногда 64 бит |
| Интерфейсы | UART, GPIO, ADC,SPI,I2C | PIC, UART, USART, LIN, CAN, Ethernet, SPI, I2S | UART, USART, SPI, I2C, иногда CAN, USB, Ethernet | UART, USART, LIN, I2C, SPI, CAN, USB, Ethernet, I2S, DSP, SAI, IrDA |
| Скорость | 1 такт на инструкцию | 4 такта на инструкцию | 1 такт на инструкцию | 1 такт на инструкцию |
| Память | ROM, SRAM, FLASH | SRAM, FLASH | Flash, SRAM, EEPROM | Flash, SDRAM, EEPROM |
| Шинная архитектура | RISC | Частично RISC | RISC | RISC |
| Архитектура памяти | Гарвардская | Гарвардская | Модифицированная | Модифицированная гарвардская |
| Энергопотребление | Низкое | Низкое | Низкое | Низкое |
| Семейства | ESP8266EX, ESP8285 | PIC16,PIC17, PIC18, PIC24, PIC32 | Tiny, Atmega, Xmega, спец. AVR | ARMv4,5,6,7 ... |
| Производители | Espressif Systems | Microchip | Atmel (Microchip) | Apple, Nvidia, Qualcomm, Samsung Electronics, TI ... |
| Стоимость | Низкая | Средняя | Средняя | Низкая |
| Популярные микроконтроллеры | ESP8266 NodeMCU, ESP-12E, ESP-01 | PIC18fXX8, PIC16f88X, PIC32MXX | Atmega8, 16, 32; вариации для Arduino | LPC2148, ARM Cortex-M0, ARM Cortex-M3, ARM Cortex-M7 |

*Табл. 1. Сравнение семейств микроконтроллеров.*

Как известно, микроконтроллеры Arduino принадлежат к семейству AVR(хотя есть и представители семейства ARM, например, Arduino Due). Рассмотрим представителей других семейств более подробно.

**Raspberry Pi**

Raspberry Pi – это не только микроконтроллер, но и полноценный одноплатный компьютер. Он оснащен ARM процессором и представляет огромные возможности для разработки проектов в области интернета вещей (IoT) и автоматизации домашней электроники. Raspberry Pi имеет широкую поддержку сообщества и обширный набор разъемов и интерфейсов. Кроме того, Raspberry Pi поддерживает множество операционных систем, включая Raspbian, Ubuntu, Windows 10 IoT и многие другие.

**STM32**

STM32 – серия микроконтроллеров, разработанных компанией STMicroelectronics. Это одна из самых популярных платформ для разработки сложных и масштабных проектов. STM32 отличаются высокой производительностью, широким набором периферийных устройств и обширной поддержкой программного обеспечения.

Основные особенности микроконтроллеров STM32:

* Мощные вычислительные возможности – STM32 оснащены мощными ядрами ARM Cortex-M, которые обеспечивают высокую производительность и быстродействие. Это позволяет эффективно решать сложные задачи и обрабатывать большие объемы данных.
* Богатые периферийные возможности – на платформе STM32 представлен широкий набор периферийных устройств, таких как UART, SPI, I2C, USB, DMA, ADC и другие. Это позволяет подключать различные внешние устройства, расширяя функциональность микроконтроллера.
* Обширная экосистема – STM32 поддерживается различными инструментальными средствами, такими как среды разработки STM32CubeIDE и Keil MDK, а также разнообразным программным обеспечением и библиотеками. Это значительно упрощает процесс разработки и позволяет сосредоточиться на решении конкретных задач.
* Разнообразие моделей – серия STM32 включает в себя множество моделей, от различных производителей и с разными характеристиками. Выбор подходящего микроконтроллера обеспечивает оптимальное соотношение функциональности, производительности и стоимости для конкретного проекта.

**ESP8266**

ESP8266 — популярный микроконтроллер, который широко используется для создания проектов в области интернета вещей. Он предоставляет широкие возможности для подключения к беспроводным сетям и обмена данными с удаленными серверами.

Преимущества ESP8266:

* Низкая стоимость: ESP8266 является одним из самых доступных микроконтроллеров на рынке. Это делает его идеальным выбором для проектов с ограниченным бюджетом.
* Беспроводные возможности: микроконтроллер поддерживает протоколы Wi-Fi и TCP/IP, что позволяет ему подключаться к Интернету и взаимодействовать с удаленными серверами.
* Простота использования: программирование ESP8266 осуществляется с использованием популярной Arduino IDE. Это позволяет даже новичкам быстро начать разрабатывать проекты с использованием микроконтроллера.
* Совместимость с другими устройствами: ESP8266 может легко взаимодействовать с другими микроконтроллерами, модулями и датчиками, расширяя возможности вашего проекта.

**PIC**

PIC (Peripheral Interface Controller) — это семейство микроконтроллеров, разработанных компанией Microchip Technology. Они отличаются надежностью и энергоэффективностью, что делает их популярным выбором для различных проектов, включая промышленные, бытовые и электронику потребительских товаров.

Особенностью микроконтроллеров PIC является их энергоэффективность. Они способны работать на низком напряжении и потреблять минимальное количество энергии. Это особенно важно для портативных устройств с ограниченным источником питания, таких как мобильные телефоны, планшеты и другие батарейные устройства.

PIC-микроконтроллеры также отличаются высокой скоростью работы. Они могут выполнять сложные задачи на высокой частоте, что важно для реализации реального времени и других приложений, требующих быстрой обработки данных.

Кроме того, PIC-микроконтроллеры обладают множеством периферийных функций. Они имеют встроенные модули для работы с аналоговыми и цифровыми сигналами, такие как АЦП (аналогово-цифровой преобразователь) и ШИМ (широтно-импульсная модуляция), а также интерфейсы для связи с внешними устройствами, такие как I2C, SPI и UART.

## 

## 5.2. Преимущества и недостатки Arduino в контексте конкретных задач

Очевидно, что Arduino подходит для проектов, которые требуют простого и быстрого прототипирования. Благодаря простоте использования и доступности, Arduino позволяет быстро создавать рабочие прототипы и тестировать идеи. Это особенно полезно для студентов, хобби-разработчиков и тех, кто только начинает свой путь в электронике и программировании. Кроме того, Arduino идеально подходит для проектов, которые требуют взаимодействия с физическими устройствами и сенсорами. Благодаря широкому выбору совместимых модулей и компонентов, Arduino позволяет легко подключать и контролировать различные устройства, такие как светодиоды, датчики температуры, акселерометры и многое другое.

Однако платы Ардуино не всегда являются лучшим выбором, так как у них есть свои недостатки в рамках конкретных проектов:

* **Отсутствие многозадачности**

Платы Arduino могут одновременно запускать только одну программу. Другие платы конкурентов, такие как Raspberry Pi, предлагают многозадачность. Подобно многоядерным процессорам, которые могут запускать несколько программ без снижения скорости всей системы, Arduino не имеет такой возможности, и мы должны закрыть один скетч, чтобы выполнить другой.

* **Не оптимизирован для производительности**

Микроконтроллеры, используемые в большинстве плат Arduino, не готовы выдавать свою полную производительность. Среда разработки Arduino оптимизирована для начинающих, поэтому они могут легко создавать эскизы. Вся эта оптимизация достигается за счет снижения общей мощности микроконтроллера. Если тот же микроконтроллер используется при разработке AVR, производительность будет увеличена в разы.

* **Отсутствие связи**

Платы Arduino ограничены в плане поддержки Bluetooth и Wi-Fi. Популярные платы Arduino, такие как UNO, не имеют встроенной поддержки связи; мы должны взаимодействовать с внешними аппаратными модулями, чтобы включить эти функции. Arduino поставляет несколько плат с этими технологиями, но общая стоимость увеличилась по сравнению с другими платами, доступными на рынке.

* **Меньший объем памяти**

Одной из основных особенностей, которой не хватает Arduino, является ограниченный объем памяти. Arduino UNO имеет 2 КБ SRAM и 32 КБ флэш-памяти, которые могут хранить только скетчи с сотнями строк. Из-за этого Arduino имеет ограниченное применение в мире робототехники и не может использоваться в проектах промышленного масштаба.

## 5.3. Заключение

Arduino является идеальной платформой для быстрого прототипирования и создания роботизированных систем. Она предназначена для начинающих разработчиков и студентов, которые хотят познакомиться с программированием и электроникой. Однако, стоит отметить, что Arduino имеет ограниченную вычислительную мощность и не рекомендуется для использования в суровых промышленных условиях. Если вам требуется разработать сложный проект, который требует быстрой и высокой вычислительной мощности, лучшим выбором может быть микропроцессорная плата, такая как Raspberry Pi, которая обладает более широким набором функций и возможностей.

# Заключение

## 

Целью данного исследования было изучение основ разработки роботизированных систем на платформе Arduino, разработка системы управления портативным автоматическим анализатором нуклеиновых кислот на базе этой системы, а также оценка ее возможностей и ограничений. Для достижения этой цели был выполнен ряд задач, включающих обзор литературы и источников, изучение архитектуры и компонентов Arduino, анализ типовых задач робототехники, изучение языка программирования Arduino и программных инструментов, подготовку среды разработки, изучение возможностей подключения и управления датчиками и актуаторами, разработку аппаратной части и программирование функциональности робота, тестирование и отладку, а также анализ преимуществ и ограничений платформы Arduino.

В результате исследования был разработан и реализован конкретный проект робота на базе Arduino, а также получены знания и навыки в области разработки роботизированных систем. Основные результаты исследования были описаны в курсовой работе, включая описание проекта, использованные методы и полученные результаты.

Выводы исследования показали, что Arduino является эффективной и доступной платформой для разработки робототехнических проектов, особенно для начинающих разработчиков и студентов. Однако, следует учитывать ограниченную вычислительную мощность Arduino и его несоответствие промышленным требованиям. Рекомендуется дальнейшее улучшение и развитие платформы Arduino, чтобы расширить ее функциональность и возможности в разработке роботизированных систем.

# Список использованных источников

<https://www.arduino.cc/>

<https://docs.arduino.cc/>

<https://wokwi.com/>

<https://www.makerguides.com/a4988-stepper-motor-driver-arduino-tutorial/>

<https://www.instructables.com/Arduino-16-LEDs-using-two-74HC595-shift-registers-/>

<https://lastminuteengineers.com/74hc595-shift-register-arduino-tutorial/#google_vignette>

<https://deepbluembedded.com/arduino-getting-started-beginners-guide/#google_vignette>

<https://www.instructables.com/AVR-Programming-with-Arduino-AVRdude-and-AVR-gcc/>

<https://learn.sparkfun.com/tutorials/what-is-an-arduino/all>

<https://en.wikipedia.org/wiki/AVR_microcontrollers>