МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет прикладной математики и информатики

Кафедра информационных систем управления

ВЕРЕНИЧ ВЛАДИСЛАВ НИКОЛАЕВИЧ

ТЕХНОЛОГИИ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ РОБОТИЗИРОВАННЫХ СИСТЕМ НА ПЛАТФОРМЕ АРДУИНО

Дипломная работа

студента 4 курса 12 группы

Руководитель

Коновалов Олег Леонидович,

доцент

Допущена к защите

«__»_____ 2024 г.

Зав. кафедрой информационных систем управления

профессор, доктор технических наук, доцент

А.М. Недзьведь

Минск

2024

ОГЛАВЛЕНИЕ

Перечень условных обозначений

ПААНК - Портативный автоматический анализатор нуклеиновых кислот
ШИМ - широтно-импульсная модуляция
ШИМ - Pulse Width Modulation
SRAM - Static Random Access Memory
EEPROM - Electrically Erasable Programmable Read-Only Memory
API - Application Programming Interface
IDE - Integrated Development Environment
UML - Unified Modeling Language
UART - Universal Asynchronous Receiver-Transmitter
USART - Universal Synchronous/Asynchronous Receiver-Transmitter
LIN - Local Interconnect Network
I2C - Inter-Integrated Circuit
SPI - Serial Peripheral Interface
CAN - Controller Area Network
USB - Universal Serial Bus
I2S - Inter-IC Sound
DSP - Digital Signal Processor
SAI - Serial Audio Interface
IrDA - Infrared Data Association

Реферат

Дипломная работа, 53 с., 26 рис., 1 таблица, 0 приложений.

Ключевые слова: C/C++, АРДУИНО, МИКРОКОНТРОЛЛЕР, UML, РОБОТИЗИРОВАННАЯ СИСТЕМА.

Объект исследования — микроконтроллер Arduino и использование его возможностей для создания роботизированных систем.

Цели работы — 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.

Методы исследования — а) теоретические: изучение литературы, посвящённой проблеме проектирования роботизированных систем с помощью микроконтроллера Ардуино и других микроконтроллеров; б) практические: обобщение опыта работ с различными семействами микроконтроллеров, проектирование архитектуры и разработка роботизированной системы, проверка работоспособности полученной таким образом системы.

Результатами являются — полноценно функционирующая роботизированная система портативного автоматического анализатора нуклеиновых кислот.

Область применения — медицинская диагностика, исследования и разработки в сфере медицины.

Рэферат

Дыпломная работа, 53 с. , 26 мал., 1 табліца, 0 дадаткаў.

Ключавыя словы: С / С++, АРДУІНА, МІКРАКАНТРОЛЕР, UML, РАБАТЫЗАВАНАЯ СІСТЭМА.

Аб'ект даследавання — мікракантролер Arduino і выкарыстанне яго магчымасцяў для стварэння рабатызаваных сістэм.

Мэты працы — вывучэнне асноў распрацоўкі рабатызаваных сістэм на платформе Arduino, распрацоўка сістэмы кіравання партатыўным аўтаматычным аналізатарам нуклеінавых кіслот на базе гэтай сістэмы, а таксама ацэнка яе магчымасцяў і абмежаванняў.

Метады даследавання — а) тэарэтычныя: вывучэнне літаратуры, прысвечанай праблеме праектавання рабатызаваных сістэм з дапамогай мікракантролера Ардуіно і іншых мікракантролераў; б) практычныя: абагульненне вопыту работ з рознымі сямействамі мікракантролераў, праектаванне архітэктуры і распрацоўка рабатызаваных сістэмы, Праверка працаздольнасці атрыманай такім чынам сістэмы.

Вынікамі з'яўляюцца — паўнавартасна функцыянуючая рабатызаваная сістэма партатыўнага аўтаматычнага аналізатара нуклеінавых кіслот.

Вобласць ужывання — медыцынская дыягностыка, даследаванні і распрацоўкі ў сферы медыцыны.

Abstract

Graduate work, 53 p., 26 illustrations., 1 table, 0 appendixes.

Keywords: C/C++, ARDUINO, MICROCONTROLLER, UML, ROBOTIC SYSTEM.

The object of research is the Arduino microcontroller and the use of its capabilities to create robotic systems.

The purposes are to study the basics of developing robotic systems based on the Arduino platform, to develop a control system for a portable automatic nucleic acid analyzer based on this system, as well as to assess its capabilities and limitations.

Research methods — a) theoretical: study of literature on the problem of designing robotic systems using the Arduino microcontroller and other microcontrollers; b) practical: generalization of experience with various families of microcontrollers, architecture design and development of a robotic system, checking the operability of the system obtained in this way.

The results are a fully functioning robotic system of a portable automatic nucleic acid analyzer.

Scope is medical diagnostics, research and development in the field of medicine.

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 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.

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. История и основные характеристики Arduino

Arduino — это микроконтроллер с открытым исходным кодом, основанный на простом в использовании аппаратном и программном обеспечении. Платы Arduino способны считывать входные данные — палец на сенсоре, свет на датчике или сообщение в Телеграмме — и превращать их в выходные данные — активацию двигателя, включение/выключение светодиода, публикацию чего-либо в Интернете и др. Так Вы можете указать плате, что делать, отправив набор инструкций микроконтроллеру на плате. Для этого вы используете язык программирования Arduino (основанный на языке программирования C++) и программное обеспечение Arduino (IDE).

На протяжении многих лет Arduino был мозгом тысяч проектов, от повседневных предметов до сложных научных инструментов. Вокруг этой платформы с открытым исходным кодом собралось всемирное сообщество разработчиков программного обеспечения — студентов, любителей, художников, программистов и профессионалов, их вклад составил невероятное количество доступных знаний, которые могут оказать большую помощь как новичкам, так и экспертам, ведь своими знаниями они часто делятся на различных онлайн-форумах.

Arduino родился в Институте интерактивного дизайна Ivrea (рисунок 2.1) [10] как простой инструмент для быстрого прототипирования, предназначенный для студентов без опыта работы в электронике и программировании. Однако как только она достигла более широкого сообщества, плата Arduino начала изменяться, чтобы адаптироваться к новым потребностям, дифференцируя свое предложение от простых 8-битных плат до продуктов для приложений IoT, носимых устройств, 3D-печати и встраиваемых сред.

Благодаря простому и доступному пользовательскому интерфейсу Arduino использовалась и продолжает использоваться в тысячах различных проектов и приложений. Программное обеспечение Arduino относительно просто в использовании для начинающих, но достаточно гибкое для опытных пользователей и разработчиков ПО. Он работает на таких операционных системах как Mac, Windows и Linux. Ученые используют его для создания недорогих научных инструментов, для доказательства принципов химии, физики, биологии или для начала работы с робототехникой. Архитекторы /дизайнеры создают

интерактивные прототипы, музыканты и художники используют его для инсталляций и экспериментов с новыми музыкальными инструментами. Создатели, скорее всего, используют его для создания многих проектов, представленных, например, на форуме Maker Faire. Arduino board — ключевой инструмент для изучения инновационных вещей. Любой может начать изобретать, просто следуя пошаговым инструкциям набора или делясь идеями в Интернете с другими членами необъятного сообщества Arduino.

Существует множество других микроконтроллеров и платформ микроконтроллеров, доступных для физических вычислений. Parallax Basic Stamp, BX-24 от Netmedia, Phidgets, Handyboard от MIT и многие другие предлагают аналогичные функции. Все эти инструменты объединяют сложные детали программирования микроконтроллеров в простой в использовании и доступный каждому пакет. Итак, основные преимущества перед другими системами:

Дешевизна — платы Arduino относительно недороги по сравнению с другими семействами микроконтроллеров. Самая дешевая версия платформы Arduino может быть собрана вручную, и даже предварительно собранные платформы Arduino стоят менее 50 долларов США.

Кроссплатформенность. Программное обеспечение для разработки проектов на Arduino, Arduino (IDE), работает в операционных системах Windows, Macintosh OSX и Linux. Большинство микроконтроллерных систем ограничены операционной системой Windows.

Относительно простая и понятная среда программирования. Программное обеспечение Arduino (IDE) просто в использовании для начинающих, но при этом достаточно гибкое, чтобы опытные пользователи могли воспользоваться его преимуществами.

Открытое и расширяемое ПО. Программное обеспечение Arduino публикуется как набор инструментов с открытым исходным кодом, доступный для расширения опытными разработчиками. Язык можно расширить с помощью библиотек C++, и люди, желающие понять технические детали, могут перейти с Arduino на язык программирования AVR C, на котором он основан. Аналогичным образом вы можете добавить код AVR-C непосредственно в свои программы Arduino.

Открытый исходный код и расширяемое оборудование. Планы плат Arduino публикуются под лицензией Creative Commons, поэтому опытные специалисты в области схемотехники могут создавать свою собственную версию платы, расширяя и улучшая ее. Даже относительно неопытные пользователи могут собрать макетную версию платы, чтобы понять, как она работает, и сэкономить деньги.

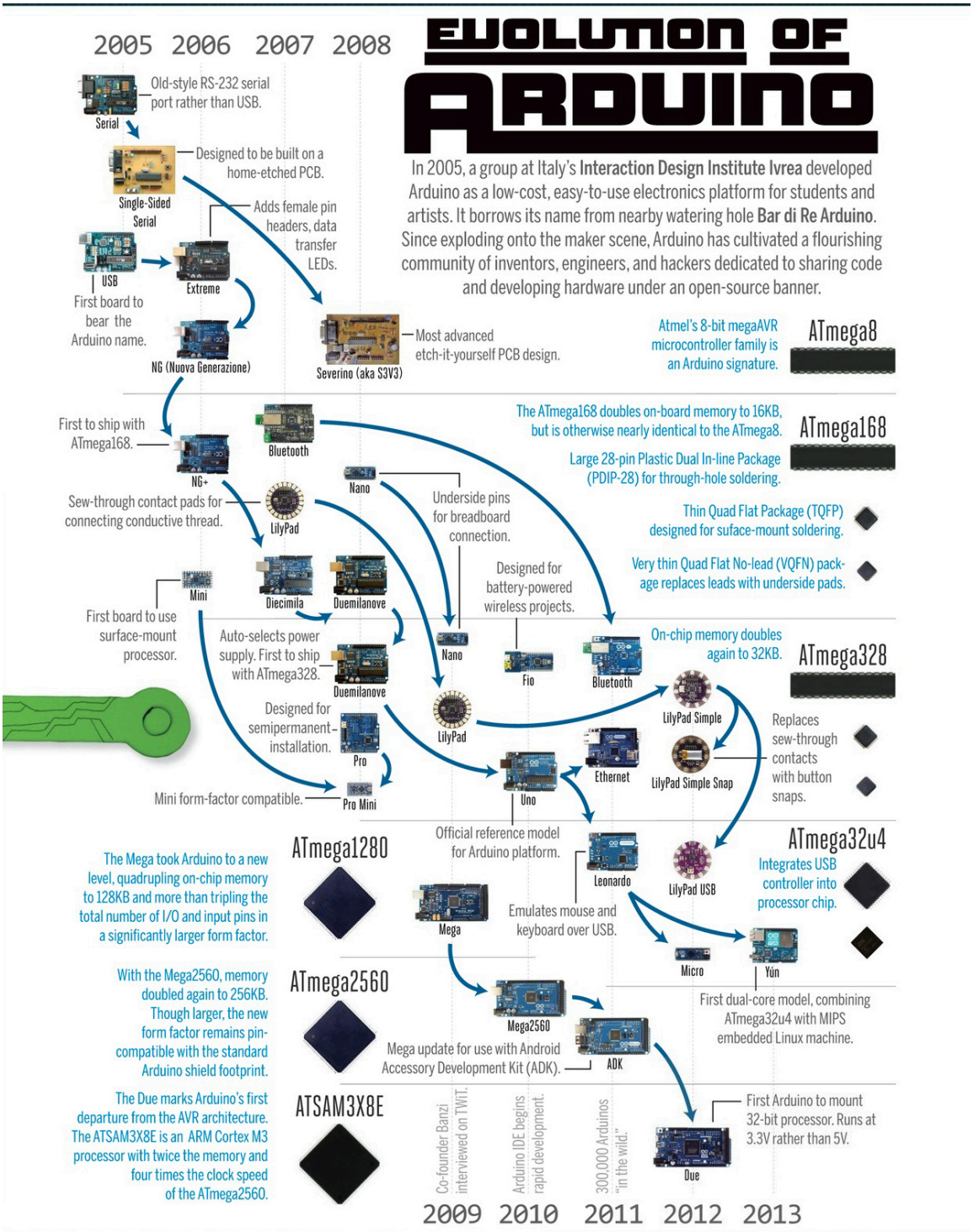

Рисунок 2.1. - Эволюция Ардуино

2.2. Архитектура и основные компоненты платформы

Глядя на плату сверху вниз (рисунок 2.2) [11], вы увидите вот это (выделены части платы, с которыми вы можете взаимодействовать в ходе обычного использования):

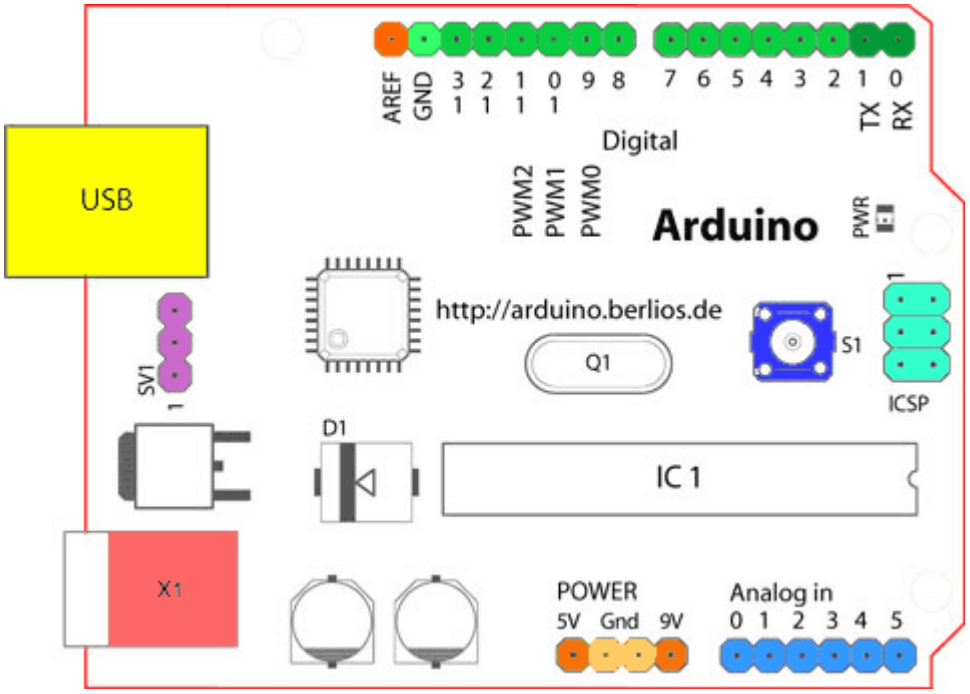

Рисунок 2.2. - Ардуино УНО

Начиная по часовой стрелке от верхнего центра:

- Аналоговый опорный контакт (оранжевый)
- Цифровая земля (светло-зеленый)
- Цифровые контакты 2–13 (зеленые)
- Цифровые контакты 0-1/Serial In/Out — TX/RX (темно-зеленые) — эти контакты нельзя использовать для цифрового ввода-вывода (digitalRead и digitalWrite), если вы также используете последовательную связь (например, Serial.begin).
- Кнопка сброса — S1 (темно-синяя)
- Внутрисхемный последовательный программатор (сине-зеленый)
- Аналоговые входы 0–5 (голубые)
- Контакты питания и заземления (питание: оранжевый, заземление: светло-оранжевый)
- Вход внешнего источника питания (9–12 В постоянного тока) — X1 (розовый)
- Переключает внешнее питание и питание USB (поместите перемычку на два контакта, ближайшие к нужному источнику питания) — SV1 (фиолетовый)

- USB (используется для загрузки скетчей на плату и для последовательной связи между платой и компьютером; может использоваться для питания платы) (желтый)

Характеристики микроконтроллеров Arduino:

ATmega328P (используется на последних платах)

- Контакты цифрового ввода/вывода: 14 (из них 6 обеспечивают выход ШИМ)
- Контакты аналогового входа: 6 (DIP) или 8 (SMD)
- Постоянный ток на контакт ввода-вывода: 40 мА
- Флэш-память: 32 КБ
- SRAM: 2 КБ
- EEPROM: 1 КБ

ATmega168 (используется на большинстве Arduino Diecimila и ранних версиях Duemilanove)

- Контакты цифрового ввода/вывода: 14 (из них 6 обеспечивают выход ШИМ)
- Контакты аналогового входа: 6 (DIP) или 8 (SMD)
- Постоянный ток на контакт ввода-вывода: 40 мА
- Флеш-память 16 КБ:
- SRAM: 1 КБ
- EEPROM: 512 байт.

ATmega8 (использовался на некоторых старых платах)

- Контакты цифрового ввода-вывода: 14 (из них 3 обеспечивают выход ШИМ)
- Аналоговые входные контакты: 6
- Постоянный ток на контакт ввода-вывода: 40 мА
- Флэш-память: 8 КБ
- SRAM: 1 КБ
- EEPROM: 512 байт.

Цифровые контакты

В дополнение к конкретным функциям, которые перечислены ниже, цифровые контакты на плате Arduino можно использовать для ввода и вывода

общего назначения с помощью команд digitalRead(), pinMode() и digitalWrite() . Каждый вывод имеет внутренний подтягивающий резистор, который можно включать/выключать с помощью функции digitalWrite() (со значением HIGH(1) или LOW(0) соответственно), когда вывод настроен как вход. Максимальный ток на вывод составляет 40 мА.

- Последовательный: контакты 0 (RX) и 1 (TX). Используется для приема (RX) и передачи (TX) последовательных данных TTL. На Arduino Diecimila эти контакты подключены к соответствующим контактам последовательного чипа FTDI USB-TTL. На Arduino BT они подключены к соответствующим контактам модуля Bluetooth WT11. В платах Arduino Mini и LilyPad Arduino они предназначены для использования с внешним последовательным модулем TTL (например, адаптером Mini-USB).
- Внешние прерывания: контакты 2 и 3. Эти контакты можно настроить на запуск прерывания при низком значении, нарастающем или падающем фронте или изменении значения. Подробности реализации можно посмотреть в документации к функции AttachInterrupt() .
- PWM: контакты 3, 5, 6, 9, 10 и 11. Обеспечьте 8-битный выход PWM с помощью функции AnalogWrite() . На платах с ATmega8 выход PWM доступен только на контактах 9, 10 и 11.
- BT Reset: контакт 7. (только для Arduino BT) Подключается к линии сброса модуля Bluetooth.
- SPI: контакты 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Эти контакты поддерживают связь по протоколу SPI, которая, хотя и обеспечивается базовым оборудованием, в настоящее время не включена в язык Arduino.
- LED: контакт 13. На Diecimila и LilyPad платах имеется встроенный светодиод, подключенный к цифровому выводу 13. Когда на выводе установлено HIGH(1) значение, светодиод горит, когда на выводе LOW(0) значение, он выключен.

Аналоговые контакты

В дополнение к конкретным функциям, перечисленным выше, аналоговые входные контакты поддерживают 10-битное аналого-цифровое преобразование (ADC) с использованием функции AnalogRead() . Большинство аналоговых входов также можно использовать в качестве цифровых контактов: аналоговый вход 5 — в качестве цифрового контакта 19, аналоговый вход 0 — в качестве цифрового контакта 14. Аналоговые входы 6 и 7 (присутствующие в таких платах, как Mini и BT) нельзя использовать в качестве цифровых контактов.

- I2C: контакты 4 (SDA) и 5 (SCL). Поддержка связи по протоколу I2C (TWI) с использованием библиотеки Wire (документация на веб-сайте Arduino в разделе Wiring).

Силовые контакты

- VIN (иногда также обозначается «9V»). Входное напряжение на плате Arduino при использовании внешнего источника питания (в отличие от 5 В от USB-подключения или другого регулируемого источника питания). Вы можете подать напряжение через этот контакт или, если напряжение подается через разъем питания, получить к нему доступ через этот контакт. Стоит отметить, что разные платы поддерживают разные диапазоны входного напряжения. Поэтому перед началом работы необходимо ознакомиться с документацией к вашей плате. Также стоит отметить, что платы типа LilyPad не имеют контакта VIN и принимает только регулируемый вход.
- 5В. Регулируемый источник питания, используемый для питания микроконтроллера и других компонентов платы. Оно может поступать либо от VIN через встроенный регулятор, либо через USB или другой регулируемый источник питания 5 В.
- 3V3. (Только для платы типа Diecimila) Питание 3,3 В, генерируемое встроенным чипом FTDI.
- GND. Заземляющие штифты.

Другие пины

- AREF. Опорное напряжение для аналоговых входах. Используется с функцией analogReference() .
- Reset. (Только для платы типа Diecimila) Установите на этой линии LOW уровень для сброса микроконтроллера. Обычно данный контакт используется для добавления кнопки сброса к щитам, которые блокируют кнопку на платформе.

2.3. 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 робототехнических проектов.

Преимущества:

- Низкая стоимость: платы Arduino относительно дешевы (особенно если сравнивать с Raspberry Pi), что делает их доступными для любителей робототехники и студентов.
- Простота использования: Arduino проста в использовании, так как доступно множество учебных пособий для начинающих, огромное количество форумов с ответами на самые разные вопросы.
- Широкий выбор компонентов: Arduino имеет широкий спектр компонентов, также известных как “Периферийные устройства”, таких как датчики, двигатели и приводы, которые можно использовать для создания различных робототехнических проектов.

- 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 нуждами. Далее будет приведен пример изменения исходного кода основных библиотек Ардуино.

Недостатки:

- Ограниченная вычислительная мощность: Платы Arduino ограничены с точки зрения вычислительной мощности. Это может ограничить сложность роботов, которые можно построить на основе данного типа микроконтроллеров.
- Ограниченное хранилище: платы Arduino имеют ограниченное хранилище. Данный атрибут 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. Обзор популярных сред разработки для Arduino

Существует несколько Arduino IDE (интегрированных сред разработки), которые можно использовать для разработки своих проектов Arduino. В рамках данной работы я буду придерживаться новой среды Arduino IDE 2.0.x. Однако в этом разделе необходимо также упомянуть о других вариантах IDE.

Ардуино IDE 1.8.x

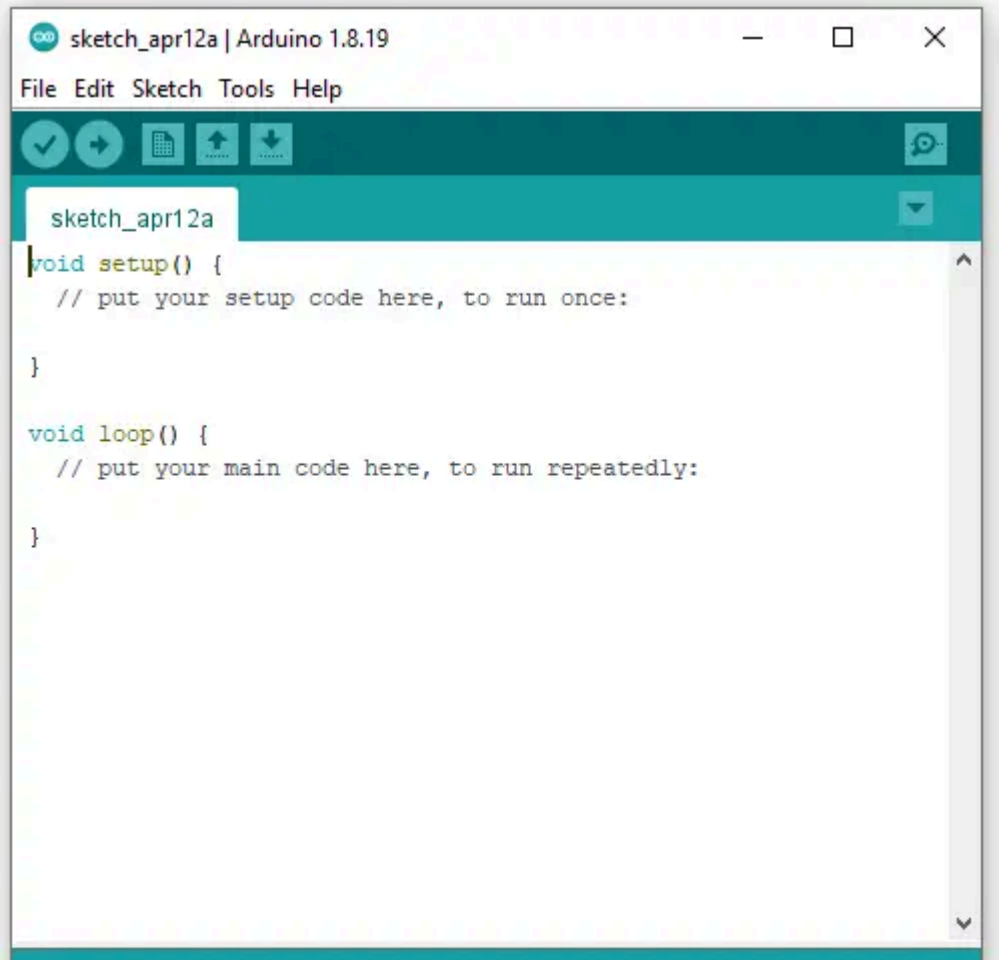

Рисунок 3.1. - Классическая «устаревшая» Arduino IDE

Это классическая среда разработки Arduino, которая уже много лет является стандартной и единственной опцией. В последнее время, после выпуска Arduino IDE v2, классическая Arduino IDE v1.8.x считается «устаревшей» Arduino IDE (рисунок 3.1). Это не значит, что это плохо, что оно исчезнет или будет прекращено. Однако это по-прежнему самая популярная IDE среди всех других вариантов и более дружелюбная к новичкам, даже если она выглядит не так современно, как другие.

Ардуино IDE 2.0.x

Arduino **IDE v2** — это новейшая среда разработки Arduino, выпущенная еще в 2021 году. Она имеет функцию отладчика (очень аппаратно ограниченную), темную тему, автодополнение кода, улучшенную навигацию по нескольким файлам и многие другие функции (рисунок 3.2) [6].

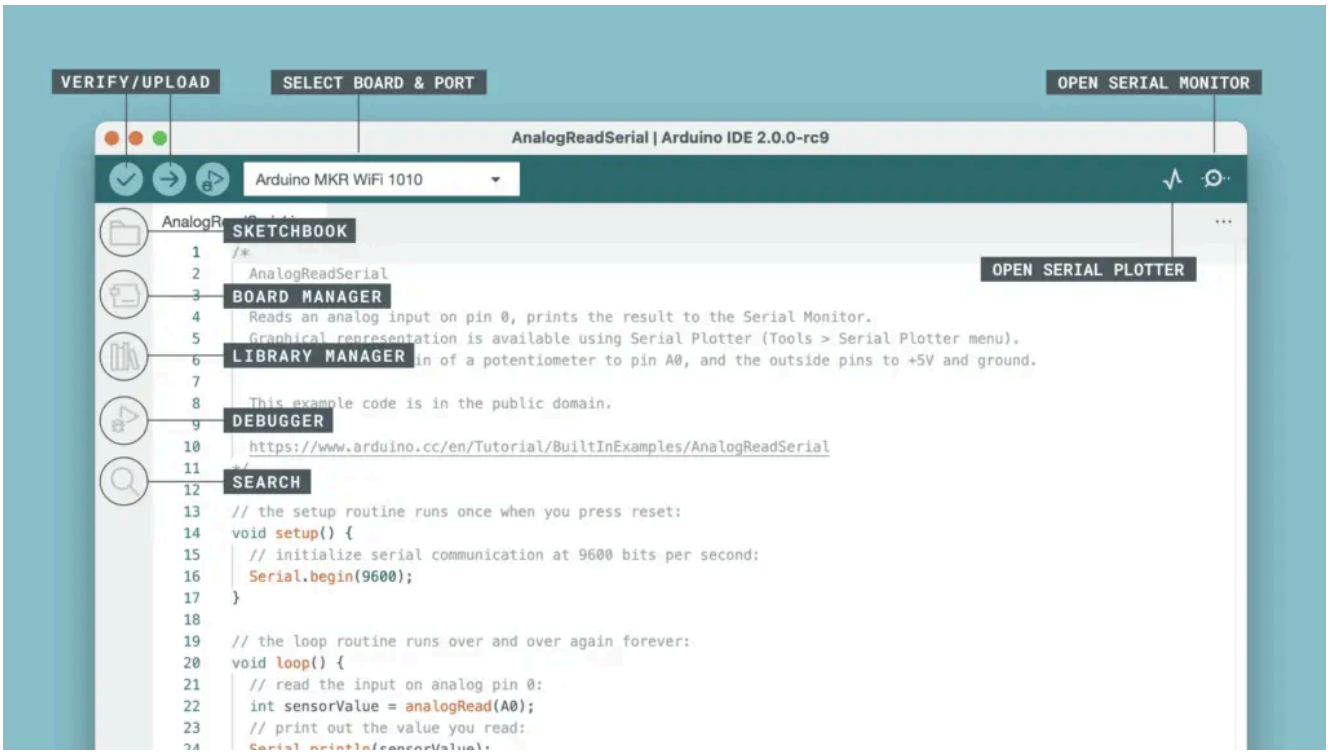

Рисунок 3.2. - Новая IDE Arduino V2.0.x

Усилия компании сейчас направлены на этот новый продукт, его доработку, исправление ошибок и проблем, а также добавление новых функций. Так что, возможно, в будущем мы увидим некоторые серьезные изменения в Arduino IDE v2, которые полностью затмят IDE v1.8.x, которая в настоящее время является самой популярной IDE, несмотря на то, что она считается «устаревшим» редактором.

Веб-редактор Arduino

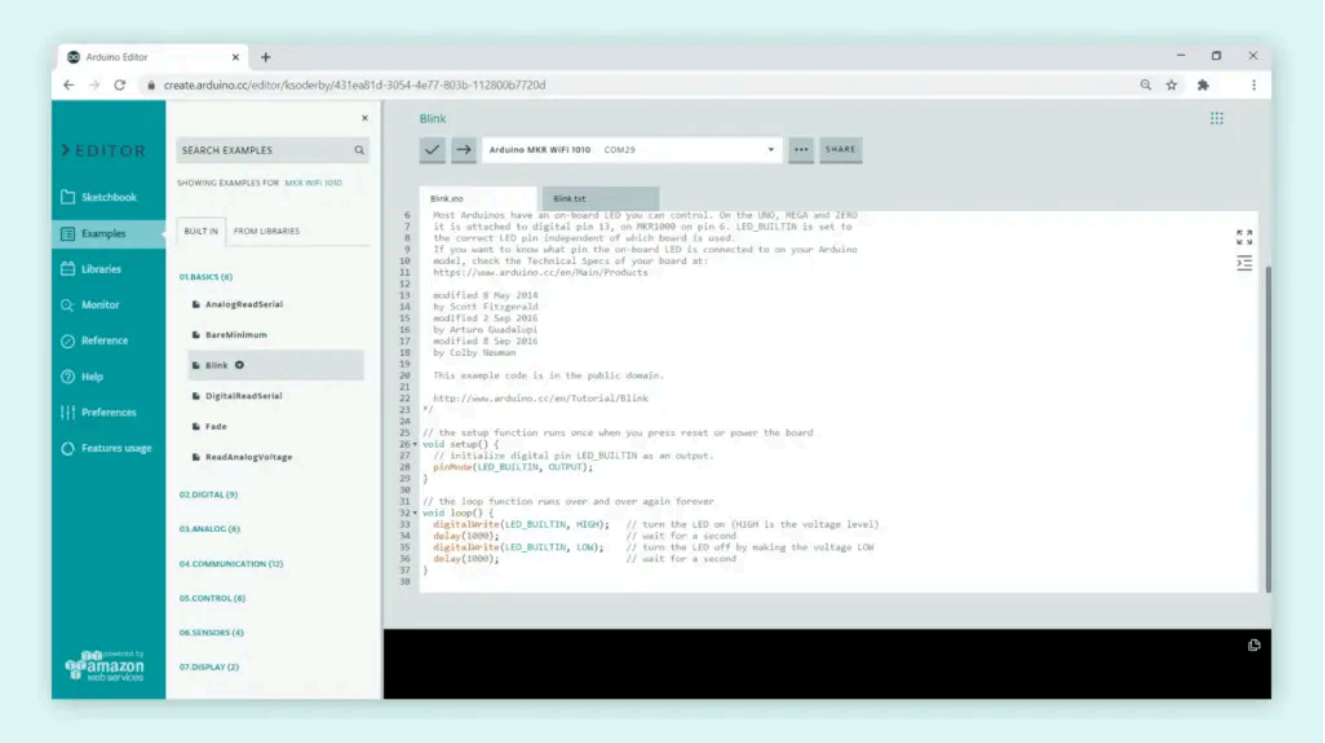

Рисунок 3.3. - Веб-редактор Arduino

Веб -редактор Arduino (рисунок 3.3) — это онлайн-среда разработки Arduino, входящая в состав Arduino Cloud Suite. Он имеет очень схожие функции и возможности с автономными IDE Arduino, за исключением того, что он полностью основан на Интернете и может использоваться онлайн с любого устройства.

3.2. Подключение и настройка Arduino-платы

Выбор платы Arduino в IDE

Для этого необходимо в Arduino IDE (рисунок 3.4), перейти в меню **«Инструменты** » и выбрать подменю **«Плата»** , в котором будут показаны все доступные платы Arduino, которые IDE поддерживает по умолчанию.

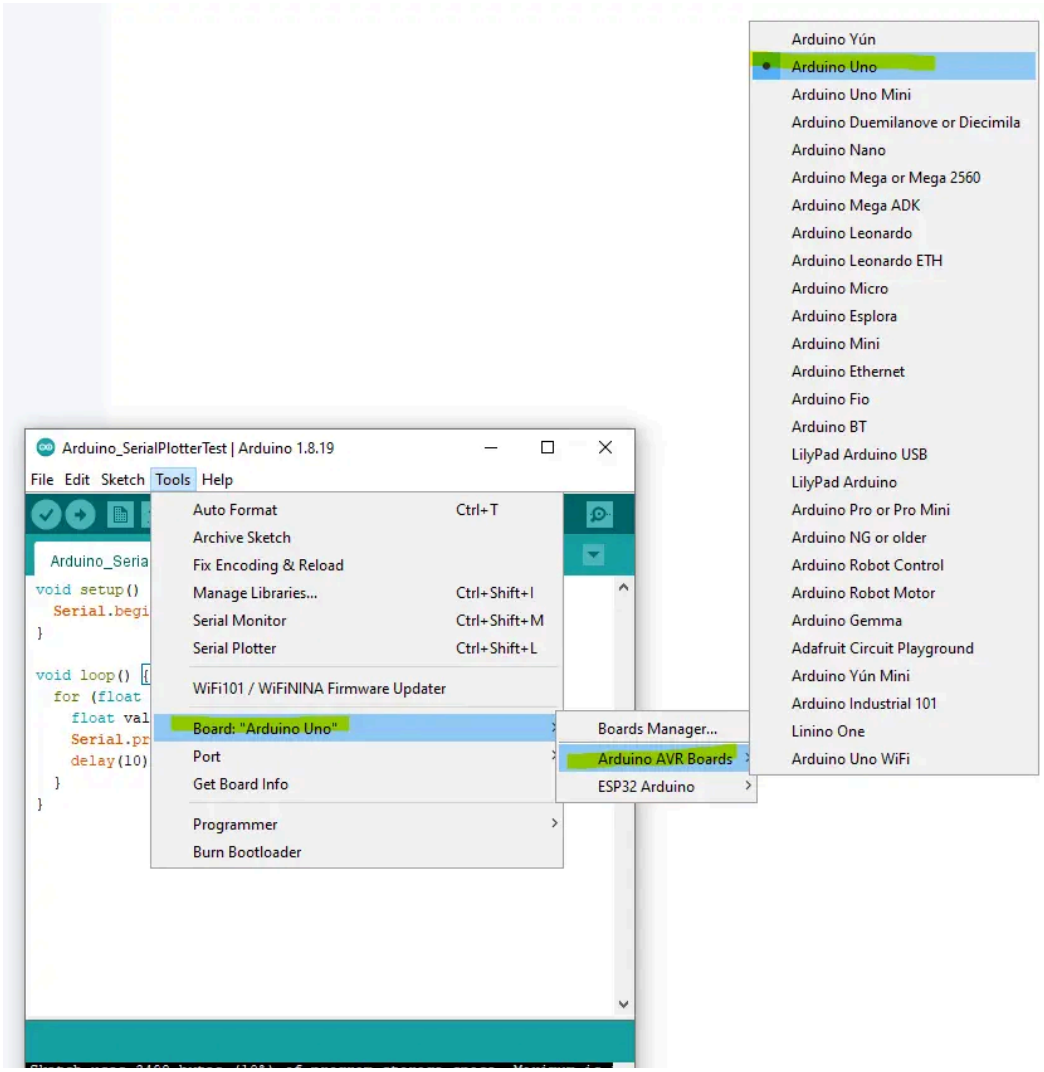

Рисунок 3.4. - Выбор платы Ардуино

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… (рисунок 3.5)

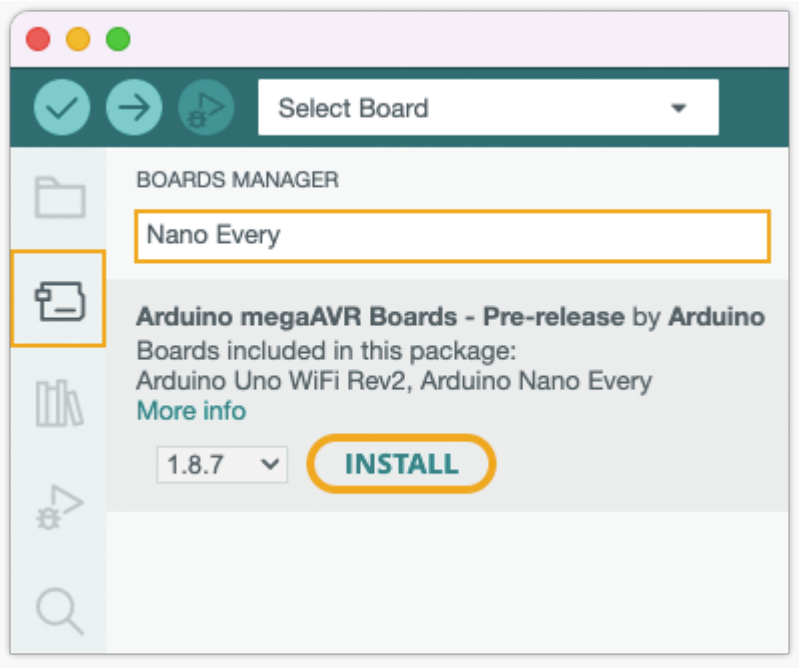

Рисунок 3.5. - Boards Manager in Arduino IDE 2

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 (рисунок 3.6). В противном случае будут возникать постоянные проблемы с загрузкой прошивки при прошивке кода на Arduino.

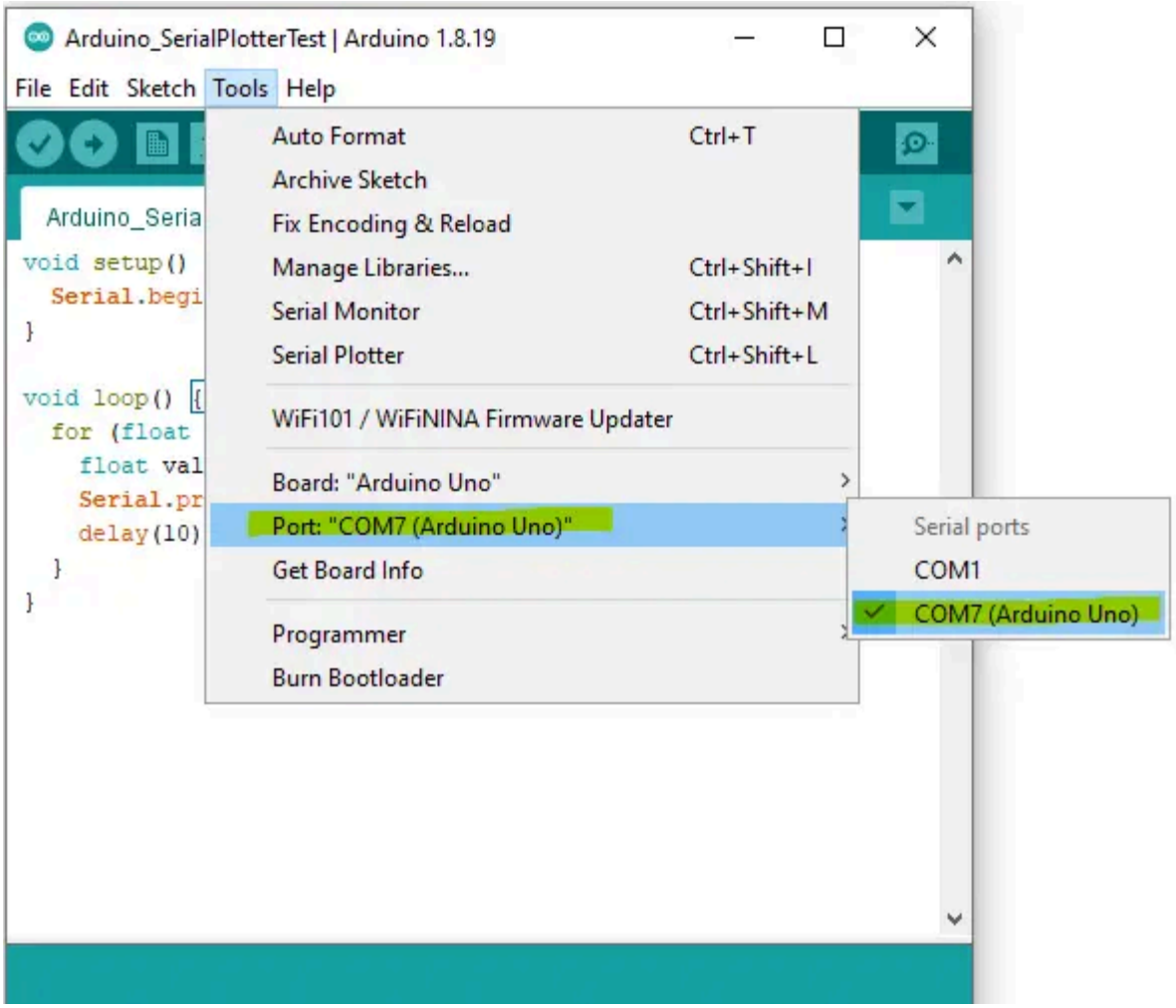

Рисунок 3.6. - Выбор последовательного порта Arduino

3.3. 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 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 исходный 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
}
```

3.4. Возможности расширения функционала с помощью библиотек

Существует очень много библиотек Arduino, как официальных, так и предоставленных сообществом. Эти библиотеки расширяют основные функции API Arduino до гораздо более сложных и продвинутых функций, позволяя производителям и разработчикам легко интегрировать датчики, исполнительные механизмы, протоколы связи и другие аппаратные компоненты в свои проекты.

Менеджер библиотек Arduino (рисунок 3.7) позволяет перемещаться по библиотекам Arduino, искать новые библиотеки и устанавливать необходимые библиотеки в локальный каталог Arduino.

Можно также использовать этот инструмент для установки официальных библиотек или библиотек, предоставленных сообществом, в несколько кликов. И по-прежнему можно вручную устанавливать библиотеки собственной разработки или какие-то специальные библиотеки, не включенные в поиск библиотек.

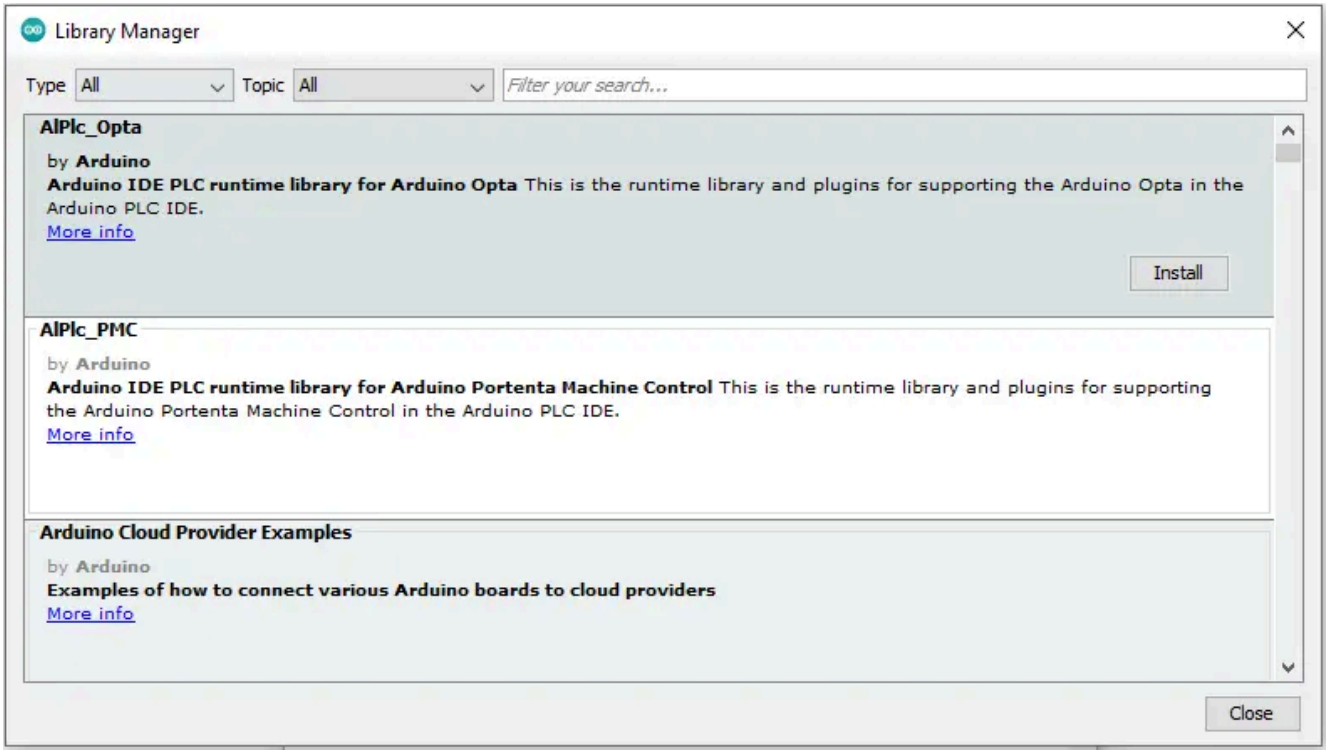

Рисунок 3.7. - Менеджер библиотек

Чтобы использовать библиотеку, скажем, `LiquidCrystal.h` LCD, нужно только включить ее заголовочный файл, как показано ниже.

```
#include <LiquidCrystal.h>
```

И теперь можно использовать функции, встроенные из `LiquidCrystal.h`.

3.5. Анализ преимуществ и ограничений платформы Arduino для разработки роботизированных систем

Приступая к работе с Arduino, необходимо узнать, что такое **датчики** , **исполнительные механизмы** и **драйверы** . Еще не будет лишним 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/робототехнике, 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.). Вот примеры актуаторов:

- 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/актуаторы представляют собой тяжелую нагрузку, которой для работы или выполнения выполняемого действия требуется значительный электрический 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, так как позволяет собирать воедино огромное количество различных периферийных устройств, и именно поэтому 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 достаточно длительное время и полную её схему размещать здесь не рационально. Именно поэтому я приведу упрощенную схему аппаратной части робота (рисунок 4.1), содержащую все датчики, исполнительные механизмы и драйверы. А также распиновку микроконтроллера.

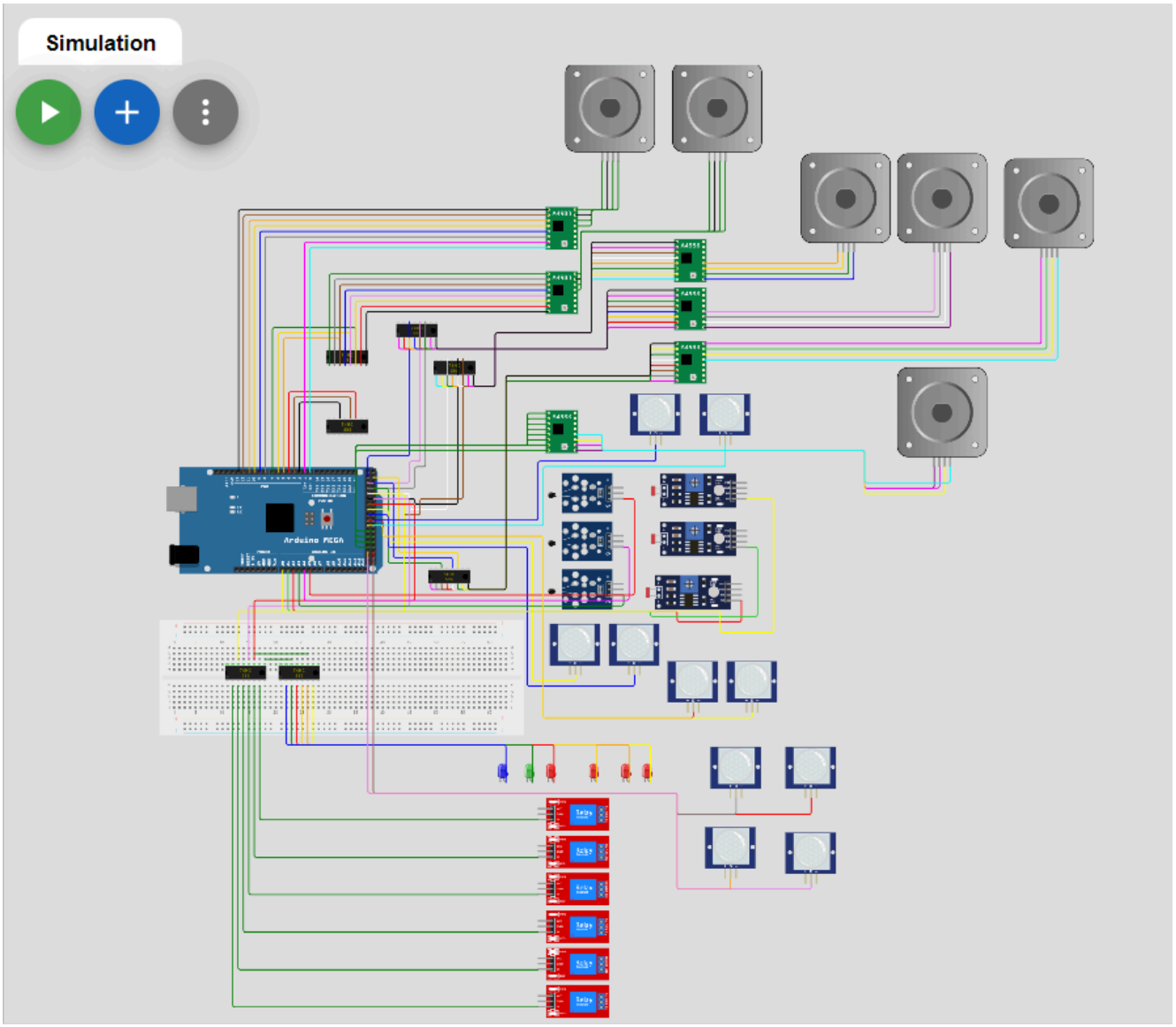

Рисунок 4.1. - Аппаратная часть ПААНК

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 (рисунок 4.2):

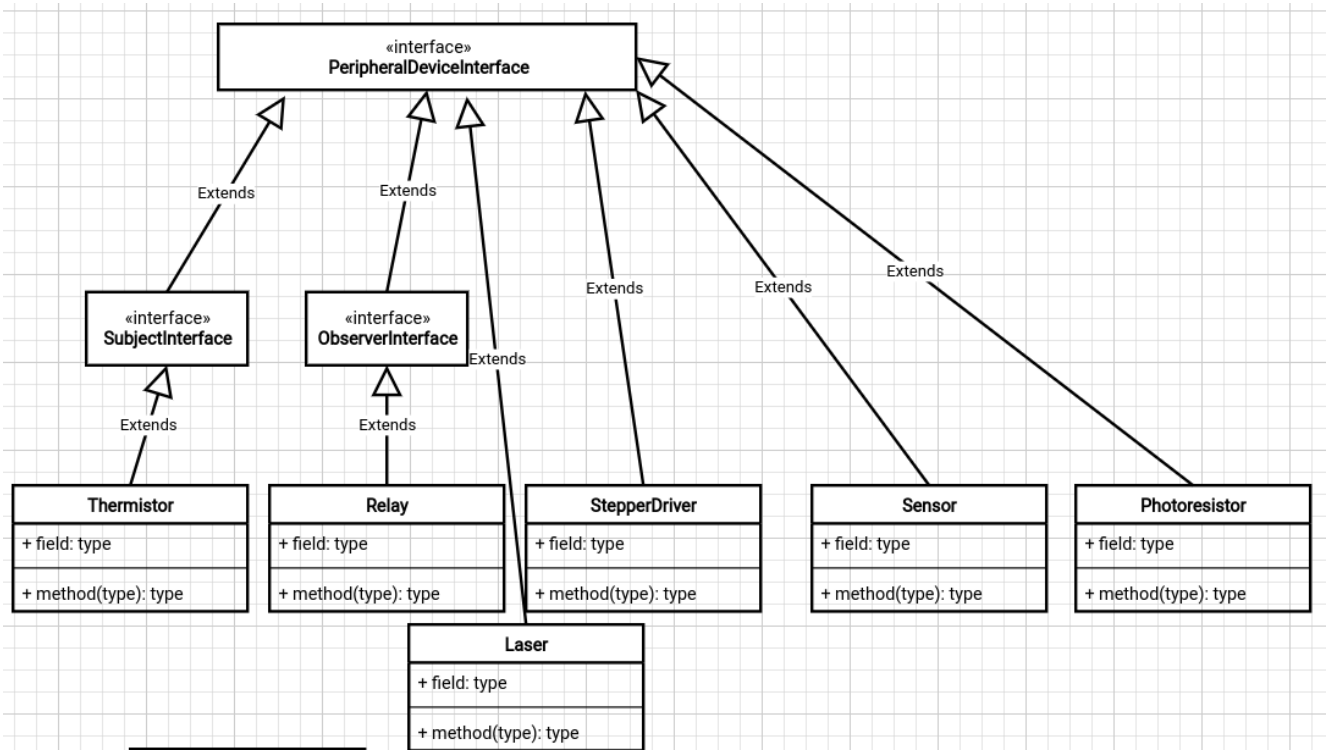

Рисунок 4.2. - Иерархия классов элементарных компонентов системы

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 (рисунок 4.3).
- 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 m_k от некоторого нулевого положения. Этому нулевому положению соответствует сигнал от датчика. Объект класса PositionedStepperZoned (рисунок 4.4).
- 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 (рисунок 4.4).

- 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 (рисунок 4.5).
- 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 (рисунок 4.6).

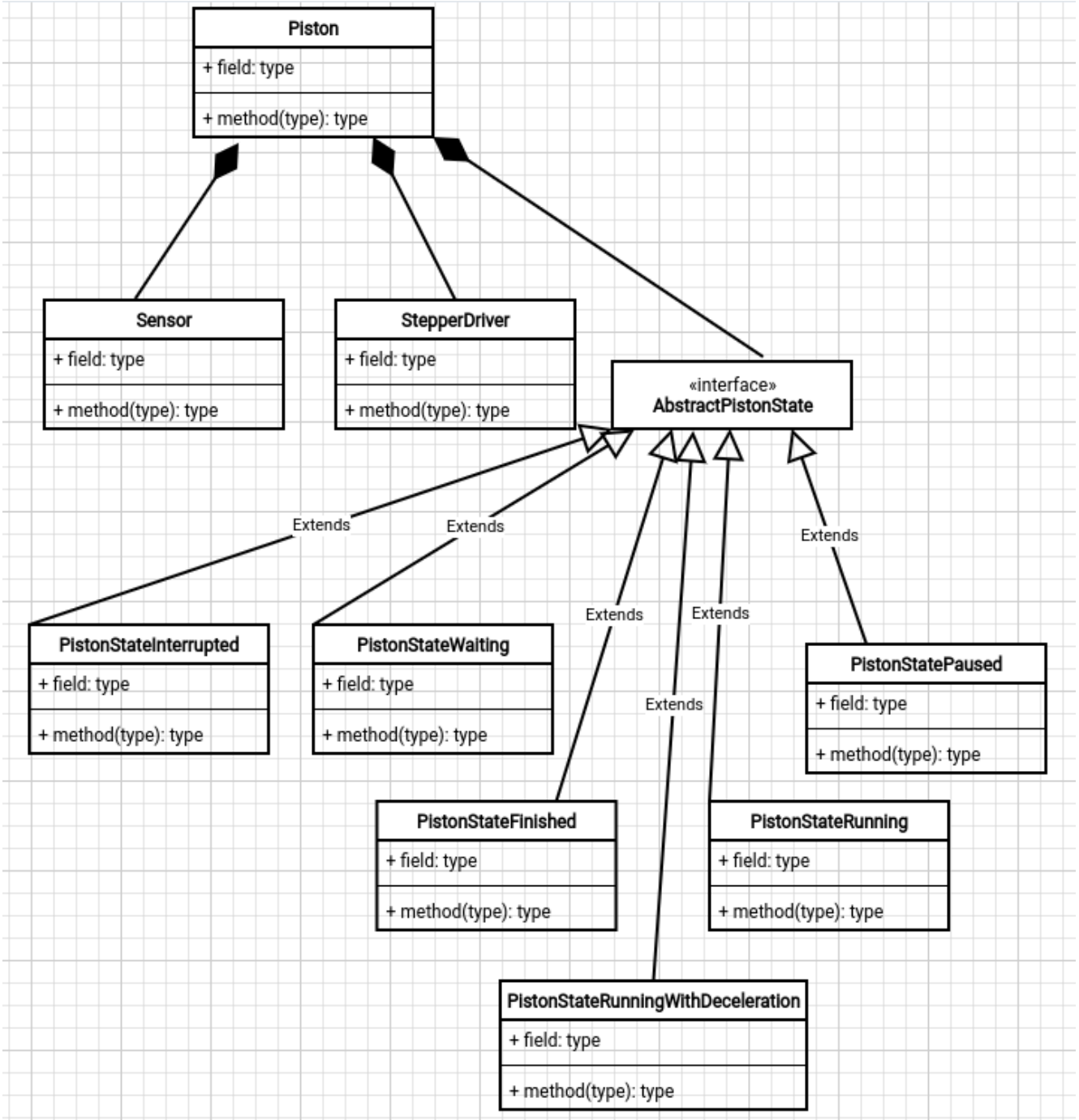

Рисунок 4.3. - UML-диаграмма отношений для класса Piston (классы Sensor и StepperDriver представлены отдельно для наглядности)

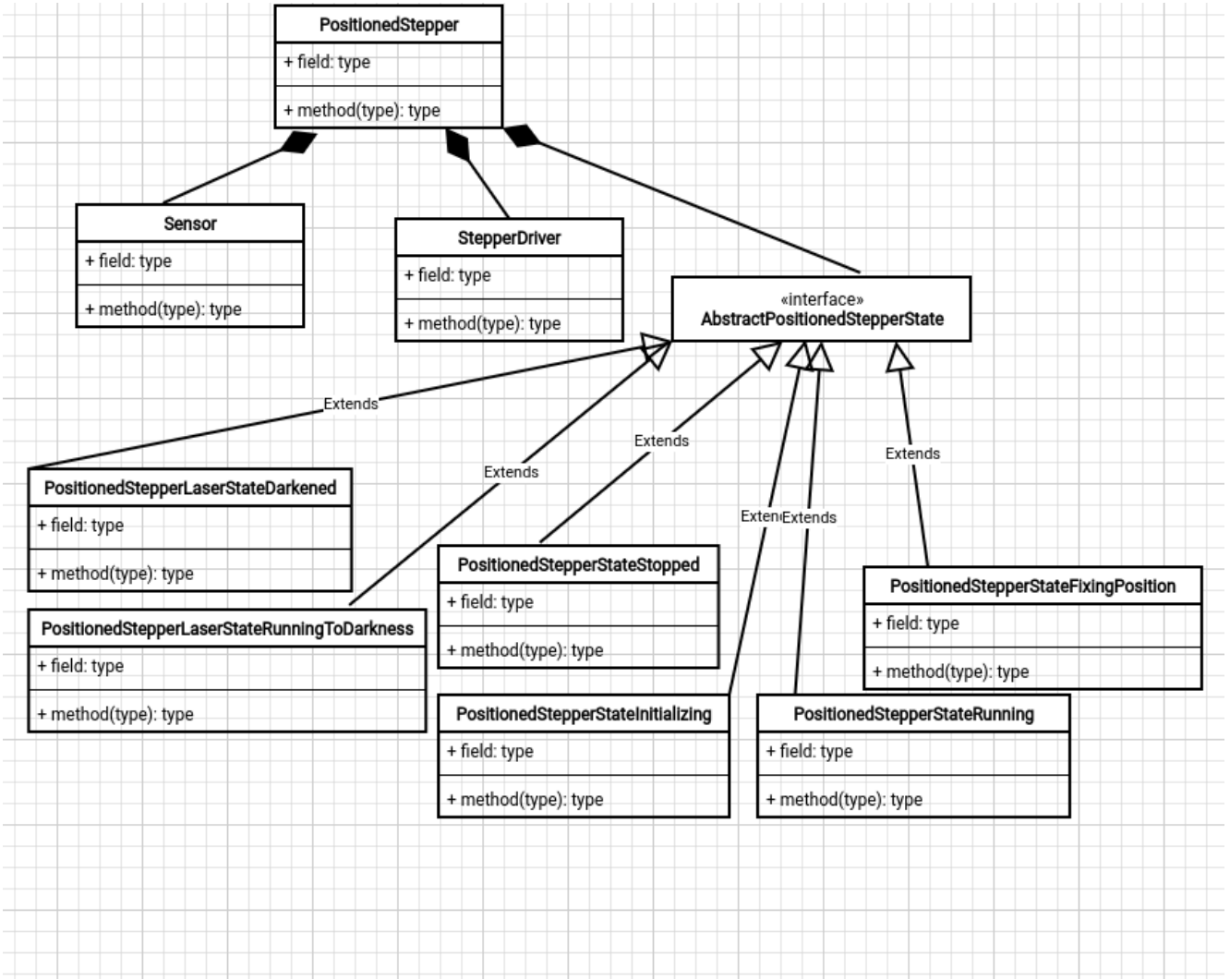

Рисунок 4.4. - UML-диаграмма отношений для класса PositionedStepper (классы Sensor и StepperDriver представлены отдельно для наглядности)

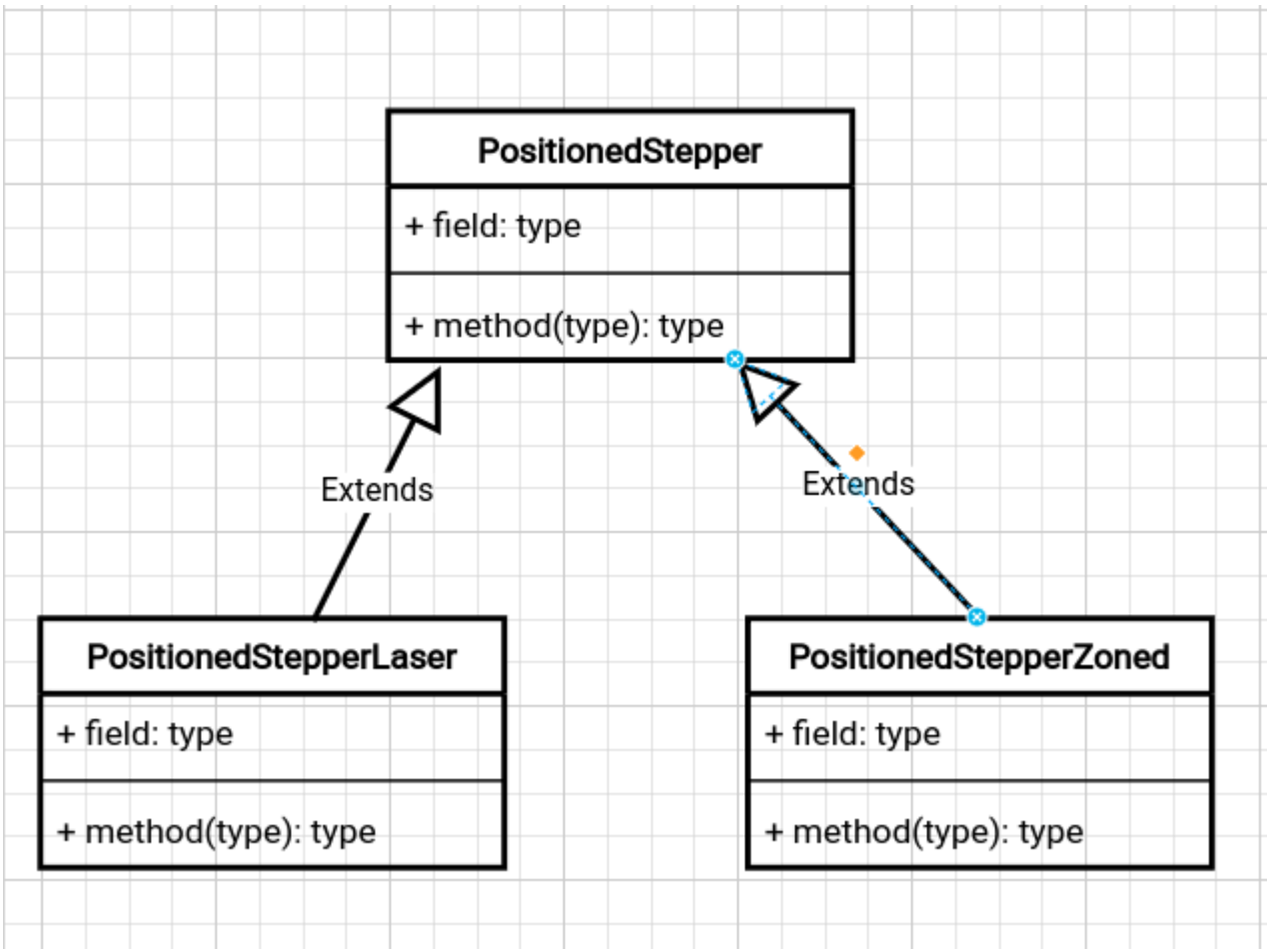

Рисунок 4.5. - Иерархия классов-наследников класса PositionedStepper (класс PositionedStepper представлен отдельно для наглядности)

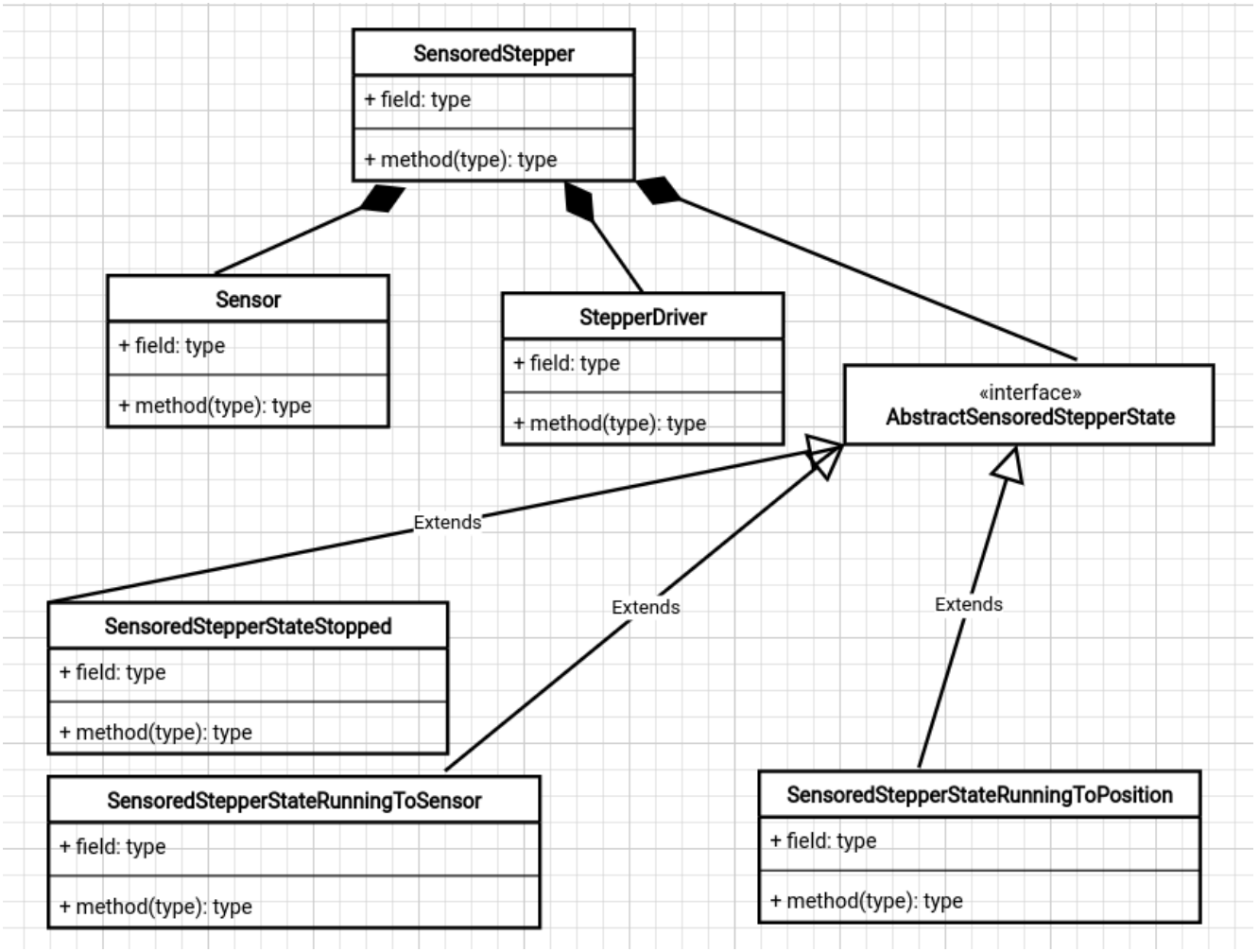

Рисунок 4.6. - UML-диаграмма отношений для класса SensoredStepper (классы Sensor и StepperDriver представлены отдельно для наглядности)

Теперь необходимо определить класс системы непосредственно. Это класс StandModel. Фактически он включает в себя составные компоненты и элементарные компоненты системы (рисунок 4.7).

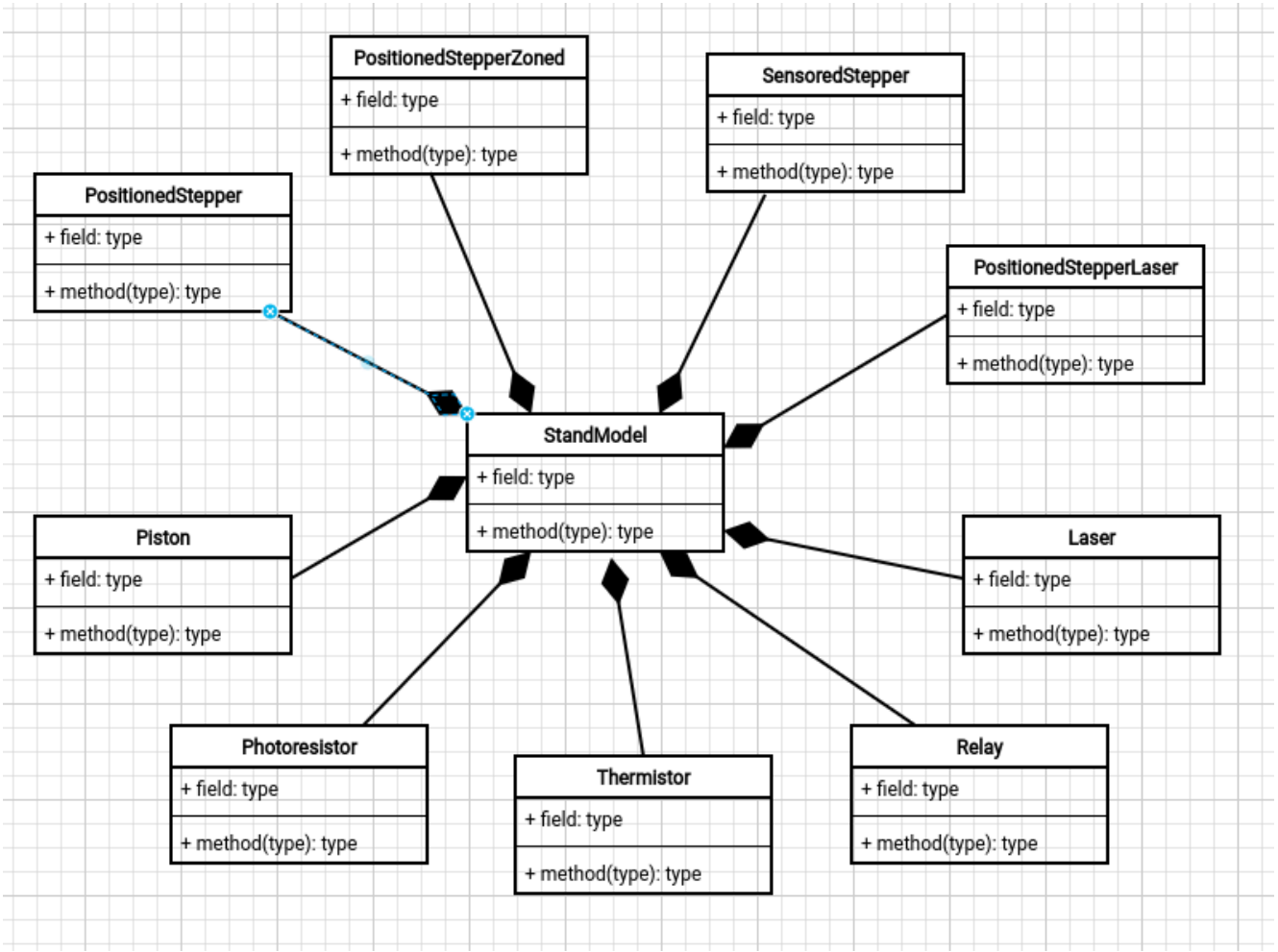

Рисунок 4.7. - UML-диаграмма отношений для класса StandModel (классы PositionedStepperZoned, SensoredStepper, PositionedStepperLaser, Laser, Relay, Thermistor, Photoresistor, Piston представлены отдельно для наглядности)

Также необходимо определить класс MessageHandler (рисунок 4.8). Этот класс зависит от класса StandModel. Он необходим для управления классом StandModel и каждым компонентом в отдельности путем отправки запросов к этому классу.

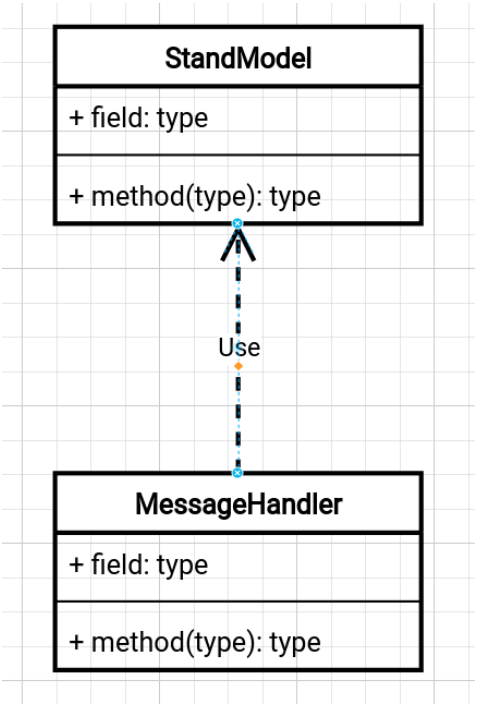

Рисунок 4.8. - UML-диаграмма отношений классов StandModel и MessageHandler

Класс MessageHandler обрабатывает запросы на основании пространства имен CommunicationProtocol (рисунок 4.9) 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!

```
namespace CommunicationProtocol {
    const char c_StartCharacter{'s'};
    const char c_EndCharacter{';'};
    const char c_SpaceCharacter{'_'};

    namespace Controller {
        namespace CommandID {
            // Application commands
            const uint16_t c_PistonStepperStart             = 0;
            const uint16_t c_PistonStepperPause             = 1;
            const uint16_t c_PistonStepperFinish            = 2;
            const uint16_t c_PistonStepperDistance          = 3;
            const uint16_t c_PistonStepperHalfStep          = 4;
            const uint16_t c_PistonStepperQuaterStep        = 5;
            const uint16_t c_PistonStepperEighthStep        = 6;
            const uint16_t c_PistonStepperSpeed             = 7;
```

Рисунок 4.9. 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 (рисунок 4.10).

```
void loop()
{
    messageHandler.receiveMessage();
    standModel.run();
}
```

Рисунок 4.10. - Функция loop()

4.3. Интеграция с дополнительными модулями. Сдвиговый регистр.

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 я 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 (рисунок 4.11):

```
// Define the stepper motor and the pins that is connected to
AccelStepper stepper1(1, 2, 5); // (Type of driver: with 2 pins, STEP, DIR)
```

Рисунок 4.11. - Определение объекта класса AccelStepper

После введения сдвигового регистра и определения для него собственного класса необходимо было использовать новый класс Pin (рисунок 4.12), 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.

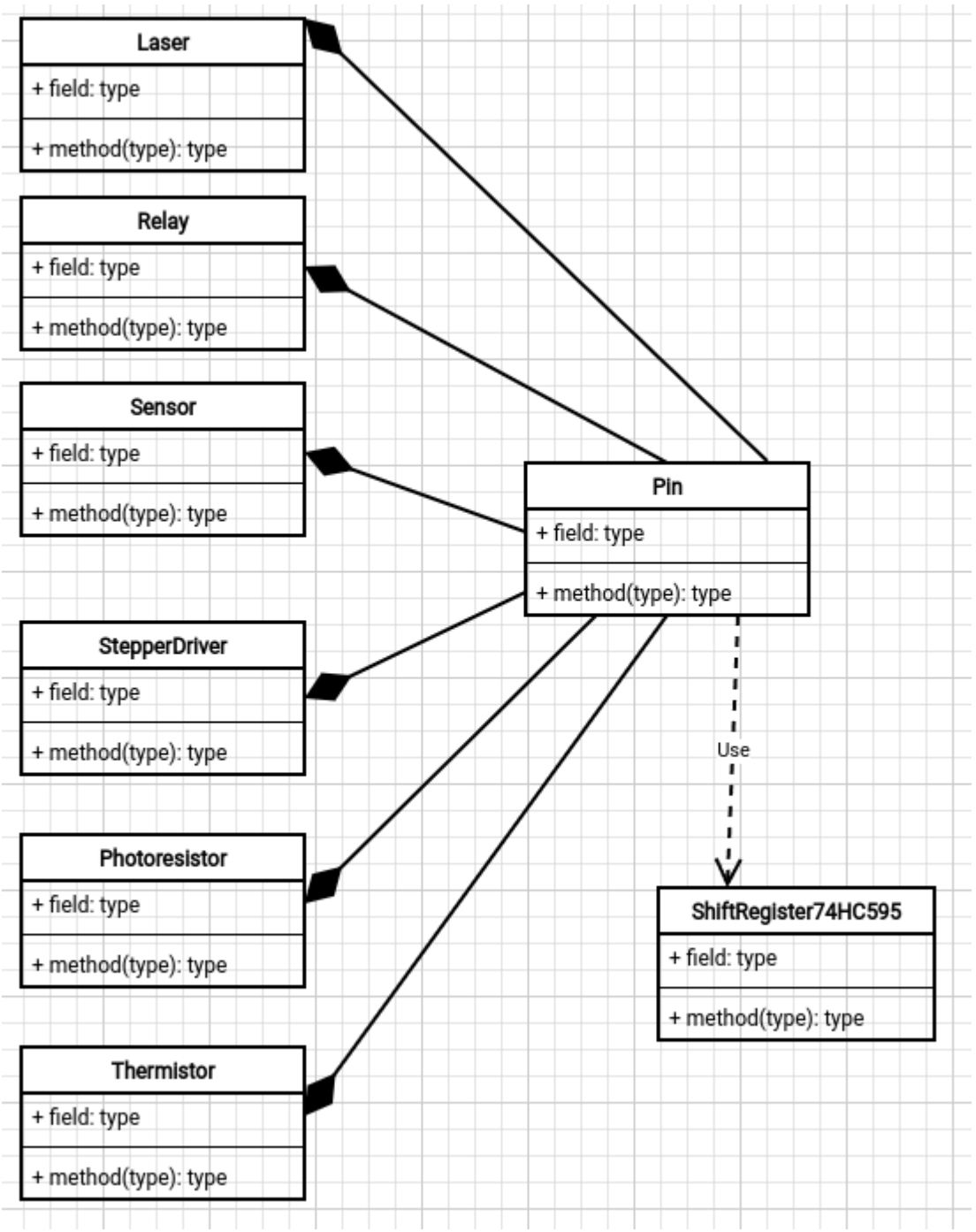

Рисунок 4.12. - UML-диаграмма отношений для класса Pin и ShiftRegister74HC595 (классы Sensor, StepperDriver, Laser, Relay, Thermistor, Photoresistor, Piston представлены отдельно для наглядности)

После этого конструктор класса элементарного компонента и его определение выглядели следующим образом (рисунок 4.13, 4.14):

```
Relay(const Pin& in1, const Pin& in2) noexcept
{
    m_In1 = in1;
    m_In2 = in2;
    pinMode(m_In1, OUTPUT);
    pinMode(m_In2, OUTPUT);
}
```

Рисунок 4.13. - Определение конструктора класса Relay

```
Relay relay1 = Relay(Pin(1, &sr6_7), Pin(2, &sr6_7));
```

Рисунок 4.14. - Определение объекта класса Relay

4.4. Интеграция с дополнительными модулями. Контроллер шагового двигателя.

В некоторый момент времени передо мной была поставлена ещё одна задача: управление контроллером шагового двигателя SI-SMM-E1-1 с помощью Ардуино.

Контроллер шагового двигателя SI-SMM-E1-1 (в дальнейшем SI-SMM-E1-1) предназначен для автоматического управления гибридным шаговым двигателем с биполярными обмотками с использованием программных кривых разгона/торможения. Управление контроллером осуществляется посредством интерфейса RS-485. Контроллер SI-SMM-E1-1 является Slave-устройством на шине с заданным адресом. Обмен данными с Master-устройством выполняется с помощью пакетов.

Что такое RS-485? RS-485 — это стандарт последовательной передачи данных. Этот стандарт был разработан Альянсом электронной промышленности (EIA) и в настоящее время поддерживается Ассоциацией телекоммуникационной индустрии (TIA). Стандарт RS-485 поддерживает скорость передачи данных до 10 Мбит/с. RS-485 — это стандарт связи, который позволяет устройствам отправлять и получать данные на большие расстояния с помощью двухпарного кабеля. Обычно он используется в промышленных помещениях, где устройства расположены на большой территории и должны обмениваться данными друг с другом.

RS-485 часто используется в системах, которым требуется несколько устройств для быстрой передачи данных на большие расстояния, например, в системах управления процессами или системах автоматизации производителей.
Однако, Arduino(из семейства ARM) поддерживает лишь следующие протоколы обмена: UART, USART, LIN, I2C, SPI, CAN, USB, Ethernet, I2S, DSP, SAI, IrDA. Поэтому в данном случае необходим преобразователь TTL(в нашем случае

сигналы контактов TX/RX, которые используют логические уровни TTL) в RS485 MAX485. Модуль преобразователя TTL в RS485 MAX485 — это аппаратный модуль, который обеспечивает связь между устройствами уровня TTL, такими как микроконтроллер Arduino или Raspberry, и сетями связи RS485 . В RS-485 данные передаются по двум проводам: один для передачи и один для приема данных , а сигналы передаются дифференциально.

Модуль предназначен для преобразования сигналов уровня TTL в сигналы, совместимые с RS485, которые можно передавать на большие расстояния (до 1200 метров или 4000 футов) без потери сигнала. он поддерживает скорость передачи данных до 2,5 Мбит/с , но по мере увеличения расстояния максимальная поддерживаемая скорость передачи данных снижается. Чип MAX485 используется в качестве основного компонента модуля, который преобразует сигналы уровня TTL в сигналы, совместимые с RS485.

Среди библиотек, предоставляемых Менеджером библиотек Arduino IDE, имеется RS-485, которая идеально подходит для решения данной задачи. Итак, объект класса RS-485 позволяет обмениваться Arduino пакетами с контроллером. Для удобства управления самим контроллером в рамках проекта был добавлен класс(и создан экземпляр) SI-SMM-E1 (рисунок 4.16), а также классы, описывающие состояние объекта класса SI-SMM-E1 (рисунок 4.15).

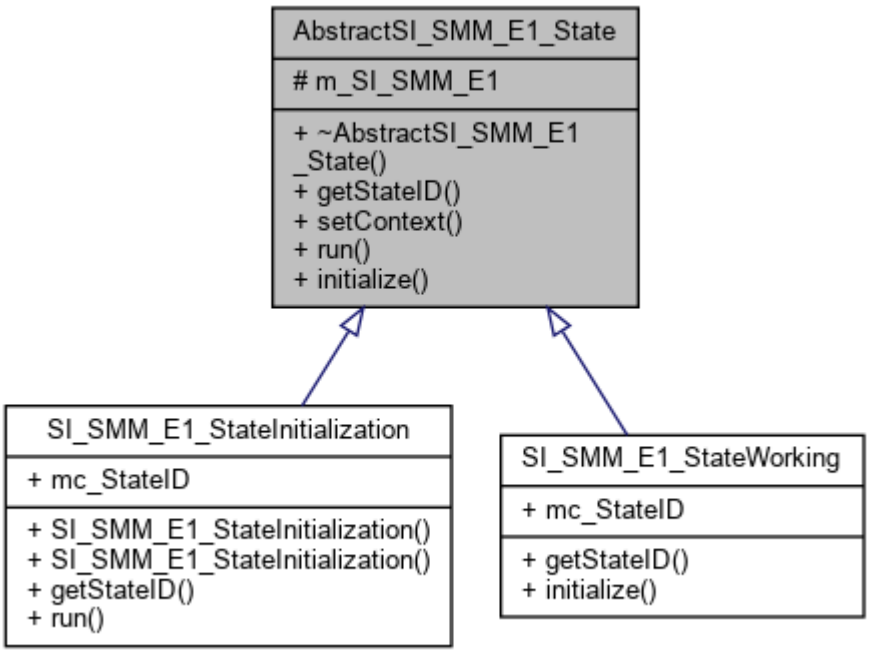

Рисунок 4.15. - UML-диаграмма отношений для класса AbstractSI_SMM_E1_State

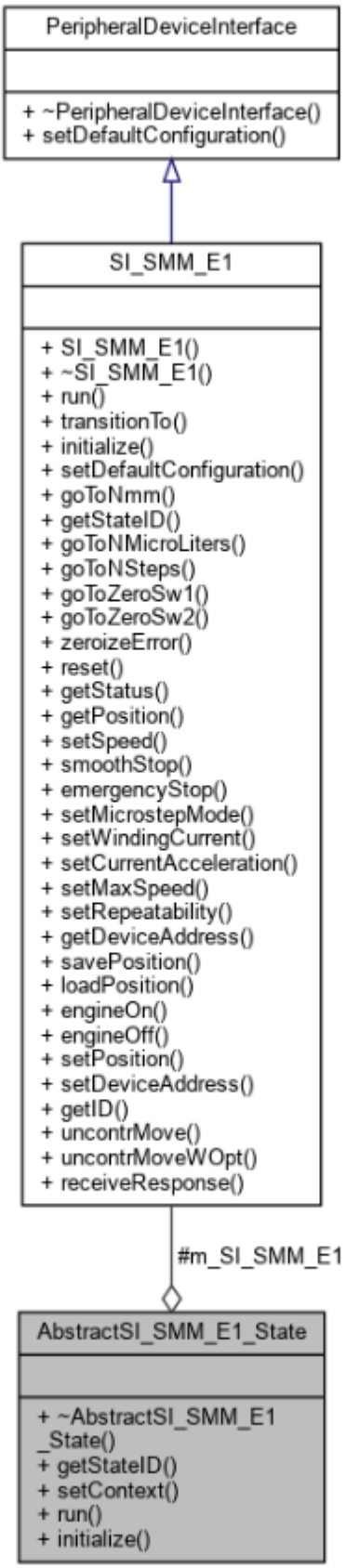

Рисунок 4.16. - UML-диаграмма отношений для класса SI-SMM-E1

Таким образом, интеграция контроллера, управление которым осуществляется посредством интерфейса RS-485, с помощью модуля MAX485 и

Arduino прошла успешно и позволяет Arduino эффективно обмениваться данными с контроллером и управлять им.

4.5. 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 (рисунок 4.15):

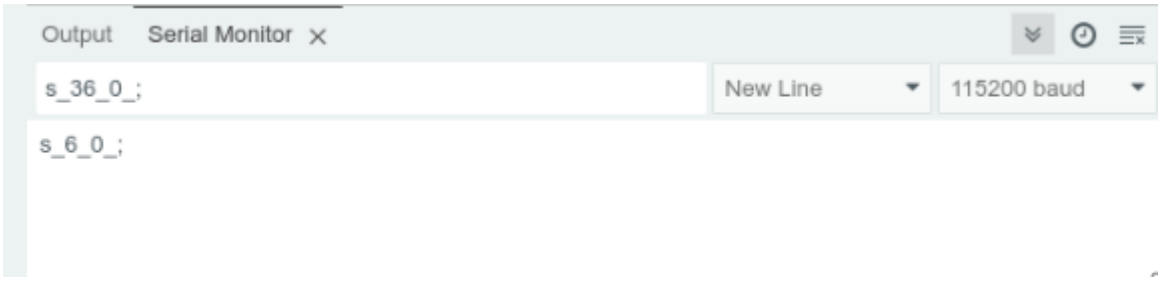

Рисунок 4.15. - Управление Arduino с помощью инструмента SerialMonitor

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. Основные отличия представлены в таблице 1 [12]:

Таблица 5.1 - Сравнение основных семейств микроконтроллеров

	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,	Tiny, Atmega, Xmega, спец. AVR	ARMv4,5,6,7 ...

		PIC24, PIC32		
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

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 включает в себя ARM процессор 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 многим известные 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(IoT). 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, количество предложений сравнимо с количеством предложений Arduino. 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 без использования различного рода адаптеров и т.д.

- 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. Это связано с изобилием библиотек для данного микроконтроллера.

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. Так они способны работать на низком напряжении и потреблять минимальное количество электрической энергии. Это особенно важно для портативных устройств с ограниченным источником питания, таких как мобильные телефоны, планшеты и другие устройства, работающие от аккумуляторов.

PIC-микроконтроллеры также отличаются высокой скоростью работы. Ведь 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. К примеру, 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. Ведь благодаря простоте использования и доступности, Arduino позволяет быстро создавать рабочие прототипы и тестировать идеи. Стоит отметить, что 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 один исполняемый модуль. Платы конкурентов(представленные выше), такие как Raspberry Pi, предлагают многозадачность. Подобно многоядерным процессорам, которые могут запускать несколько исполняемых модулей без снижения скорости системы в целом, Arduino не имеет такой возможности, и мы должны прекратить выполнение одного скетча, 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 разработчиков и любителей, поэтому они могут легко создавать свои прототипы. Данная оптимизация достигается за счет снижения общей мощности самого микроконтроллера. Если тот же микроконтроллер используется при разработке AVR, производительность будет увеличена в несколько раз.

- **Отсутствие связи**
  Платы Arduino ограничены в плане поддержки протоколов Bluetooth и Wi-Fi. Популярные платы Arduino, такие как UNO или Due, не имеют встроенной поддержки связи; мы должны взаимодействовать с внешними аппаратными модулями(адаптерами), чтобы использовать эти функции. Arduino поставляет несколько типов плат с этими технологиями, но общая стоимость такого типа плат заметно выше по сравнению с другими платами, доступными на рынке.

- **Меньший объем памяти**
  Одной из основных особенностей, которой не хватает Arduino, является ограниченный объем памяти. Например, 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 разработки встраиваемых систем 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

1. Официальный сайт Arduino [Электронный ресурс]. - Режим доступа: https://www.arduino.cc/. - Дата доступа: 16.03.2024
2. Онлайн-инструмент моделирования проектов IoT [Электронный ресурс]. - Режим доступа: https://wokwi.com/. - Дата доступа: 10.02.2024
3. Статья про управление шаговым двигателем с помощью драйвера A4988 и Arduino [Электронный ресурс]. - Режим доступа: https://www.makerguides.com/a4988-stepper-motor-driver-arduino-tutorial/. - Дата доступа: 11.03.2024
4. Статья об использовании сдвигового регистра и Arduino [Электронный ресурс]. - Режим доступа: https://www.instructables.com/Arduino-16-LEDs-using-two-74HC595-shift-registers-/ . - Дата доступа: 01.02.2024
5. Статья о принципах работы сдвигового регистра [Электронный ресурс]. - Режим доступа: https://lastminuteengineers.com/74hc595-shift-register-arduino-tutorial/#google_vignette . - Дата доступа: 05.02.2024
6. Статья о настройке среды разработки Arduino IDE [Электронный ресурс]. - Режим доступа: https://deepbluembedded.com/arduino-getting-started-beginners-guide/#google_vignette . - Дата доступа: 10.02.2024
7. Статья о компиляторе AVR-gcc [Электронный ресурс]. - Режим доступа: https://www.instructables.com/AVR-Programming-with-Arduino-AVRdude-and-AVR-gcc/. - Дата доступа: 10.02.2024
8. Сравнительный анализ наиболее известных плат Arduino [Электронный ресурс]. - Режим доступа: https://learn.sparkfun.com/tutorials/what-is-an-arduino/all. - Дата доступа: 10.02.2024
9. Микроконтроллеры семейства AVR [Электронный ресурс]. - Режим доступа: https://en.wikipedia.org/wiki/AVR_microcontrollers. - Дата доступа: 05.03.2024
10. Статья об эволюции платформы Arduino [Электронный ресурс]. - Режим доступа:https://www.unitingdigital.com/articles/2018/10/26/the-history-and-evolution-of-arduino . - Дата доступа: 01.02.2024
11. Статья об аппаратном обеспечении Arduino [Электронный ресурс]. - Режим доступа: http://edu.tsu.ru/eor/resourse/277/html/5.html . - Дата доступа: 06.02.2024
12. Статья о различии между микроконтроллерами AVR, ARM, 8051, PIC [Электронный ресурс]. - Режим доступа: https://learn.sparkfun.com/tutorials/what-is-an-arduino/all. - Дата доступа: 07.02.2024
13. Документация Arduino [Электронный ресурс]. - Режим доступа: https://docs.arduino.cc/ . - Дата доступа: 09.02.2024

14. Статья об использовании компилятора ARM-gcc [Электронный ресурс]. - Режим доступа: https://embeddedinventor.com/a-complete-beginners-guide-to-the-gnu-arm-toolchain-part-1/ . - Дата доступа: 08.03.2024
15. Официальный сайт ООО “Симплетехника” [Электронный ресурс]. - Режим доступа: https://simpletechnica.ru/ru/content/o-nas.html. - Дата доступа: 16.03.2024
16. Официальный сайт ООО “Симплетехника” [Электронный ресурс]. - Режим доступа: https://simpletechnica.ru/ru/content/o-nas.html. - Дата доступа: 16.03.2024
17. Статья о преобразователе MAX485 [Электронный ресурс]. - Режим доступа: https://diyprojectslab.com/max485-modbus-serial-communication-with-arduino/ . - Дата доступа: 16.03.2024