Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

Факультет математики и информатики
Кафедра информационных и управляющих систем
Направление подготовки /специальность 09.04.04 – Программная инженерия
Направленность (профиль) образовательной программы Управление разработкой программного обеспечения

ДОПУСТИТЬ К ЗАЩИТЕ
Зав. кафедрой
_____А.В. Бушманов
«_____»_____2022 г.

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

на тему: Разработка системы управления манипуляционным устройством-«Рука»

Исполнитель
студент группы 057-ом _____ М. С. Потемкин
(подпись, дата)

Руководитель
доцент, канд. техн. наук _____ А. В. Бушманов
(подпись, дата)

Руководитель научного
содержания программы
магистратуры
профессор, док. техн. наук _____ И. Е. Еремин
(подпись, дата)

Нормоконтроль
инженер кафедры _____В. Н. Адаменко
(подпись, дата)

Рецензент
доцент, канд. техн. наук _____А. Н. Рыбалев
(подпись, дата)

Благовещенск 2022

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

Факультет математики и информатики
Кафедра информационных и управляющих систем

Утверждаю
Зав. кафедрой
_____А.В. Бушманов
«_____»_____2022 г.

ЗАДАНИЕ

К выпускной квалификационной работе студента Потемкина Михаила Сергеевича

1. Тема выпускной квалификационной работы: Разработка системы управления манипуляционным устройством-«Рука»
(Утверждено приказом от 07.02.2022 №228-уч)

2. Срок сдачи студентом законченной работы (проекта) 23.06.2022 г.

3. Исходные данные к выпускной квалификационной работе: предметная область, отчеты по практической подготовке, результаты выступления на научной конференции

4. Содержание выпускной квалификационной работы (перечень подлежащих разработке вопросов): анализ предметной области проводимого исследования, разработка и реализация системы управления манипуляционным устройством

5. Дата выдачи задания: 31.01.2022 г.

6. Руководитель выпускной квалификационной работы: А. В. Бушманов, доцент, канд. техн. наук
(фамилия, имя, отчество, должность, уч. степень, уч. звание)

Задание принял к исполнению (31.01.2022)_____

РЕФЕРАТ

Магистерская диссертация содержит 88 с., 59 рисунков, 50 источников

МАНИПУЛЯТОР, МИКРОКОНТРОЛЛЕР, СМАРТФОН, BLUETOOTH, РЕЖИМ УПРАВЛЕНИЯ, АЛГОРИТМ УПРАВЛЕНИЯ, ПОЛЬЗОВАТЕЛЬСКИЙ ИНТЕРФЕЙС

Целью работы является создание приложения содержащее интерфейс пользователя на ОС Android, разработка алгоритмов управления манипуляционным устройством. Разработанная система управления манипулятором может использоваться на различных производствах для перемещения предметов в рабочей зоне манипулятора.

В данной работе описывается технология разработки системы управления манипуляционным устройством, копирующем движения человеческой руки. Подробно документирован алгоритм обмена данными между микроконтроллером и смартфоном(ОС Android) посредством Bluetooth связи. Создано приложения пользователя для дистанционного взаимодействия пользователя с манипулятором.

Задачи выпускной работы:

– анализ подходов и определение требований к управлению манипуляционным устройством;

– разработка и исследование системы управления манипуляционным устройством;

– создание аппаратно-программного продукта;

– исследование разработанной системы, на реальном образце.

СОДЕРЖАНИЕ

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

СУ – система управления;

МР – манипуляционный робот;

ОС – операционная система;

ПК – персональный компьютер;

IDE – среда разработки для создания различного программного обеспечения;

Android SDK – комплект для разработки программного обеспечения;

ПО – программное обеспечение;

мс – миллисекунды;

AVR – семейство восьмибитных микроконтроллеров, ранее выпускавшихся фирмой Atmel;

MAC-адрес – уникальный идентификатор активного оборудования в компьютерных сетях Ethernet;

Bluetooth – стандарт беспроводной технологии ближнего действия;

UART – универсальный асинхронный Приемо-Передатчик (universal asynchronous receiver-transmitter);

APK – формат архивных исполняемых файлов-приложений для Android и ряда других операционных систем, основанных на Android;

CPU – центральный процессор (ЦП);

А – ампер;

МГц – мегагерц;

КБ – килобайт;

см – сантиметр;

ПЗУ EEPROM – электрически стираемое перепрограммируемое постоянное запоминающее устройство;

ВВЕДЕНИЕ

Сегодня для выполнения в опасных для человека условиях технологических задач, в современной промышленности используются роботы манипуляторы, повторяющие поведение человека.

На данный момент в робототехнике актуально использование следующих типов роботов: роботы у которых задан жесткий алгоритм выполняемых действий, управление такими устройствами осуществляется человеком – оператором; роботы обладающие искусственным интеллектом (интегральные) и работающие целенаправленно, без человеческого контроля, т. е. имеющие разум. Большинство используемых устройств в настоящее время – роботы манипуляторы [1].

Для использования манипуляционных устройств в производственном процессе предприятия требуются специализированные системы управления. Данные системы управления необходимы для осуществления взаимодействия манипуляционного устройства – робота с человеком – оператором, для выполнения широкого спектра технологических задач.

В связи с нынешней напряженной геополитической обстановкой, возникают затруднения в использовании импортных СУ МР. Из – за этого возникает необходимость отказаться от зарубежных поставщиков и технической поддержки имеющихся СУ и внедрять в технологический процесс собственные отечественные разработки не уступающие в функциональных возможностях зарубежным разработкам.

Сегодня в промышленности большую часть роботов составляют механические манипуляторы. Механические манипуляторы являются пространственным механизмом в виде кинематических звеньев, образующих кинематические пары с угловым или поступательным относительным движением и системой установленных на сочленении приводов приводящих в движение звенья устройства, обычно раздельных для каждой степени подвижности. Как правило, манипуляторы заканчиваются рабочим органом [3].

Объектом исследования является система управления автоморфным манипуляционным устройством.

Предметом исследования является методика разработки алгоритмов управления звеньями и обмена данными между микроконтроллерным устройством Arduino и смартфоном ОС Android.

Целью работы является создание приложения содержащее интерфейс пользователя на ОС Android, разработка алгоритмов управления манипуляционным устройством. Разработанная система управления манипулятором может использоваться на различных производствах для перемещения предметов в рабочей зоне манипулятора.

Для достижения поставленной цели необходимо решить следующие задачи:

– анализ подходов и определение требований к управлению манипуляционным устройством;

– разработка и исследование системы управления манипуляционным устройством;

– создание аппаратно-программного продукта;

– исследование разработанной системы, на реальном образце.

1 ПРЕДМЕТНАЯ ОБЛАСТЬ ПРОВОДИМОГО ИССЛЕДОВАНИЯ

1.1 Предметная область

Робототехника – наука, развивающаяся в направлении разработки автоматизированных технических систем [4]. Основное предназначение разрабатываемых робототехнических систем заключается в выполнении поставленных задач в труднодоступных или опасных для выполнения человеком технологических операций.

Робот – автоматическое устройство, являющееся принципиально новым видом машин [5]. К функциям, которые должен выполнять робот относятся следующие:

– функция манипулирования и перемещения. Роботы должны выполнять полезную работу и обеспечивать перемещение предметов в своей рабочей области;

– функция диалога. Для взаимодействия человека оператора с роботом должна поддерживаться связь в режиме диалога в виде управляющих сигналов, получаемых от оператора и ответа на полученные сигналы.

Роботы манипуляторы представляют собой манипуляционные устройства закрепленные на платформе и выполняющие задачи по перемещению предметов в пределах рабочего пространства.

В настоящее время современные роботы представляют манипуляторы закрепленные на платформе и предназначенные для выполнения однообразной работы перемещения предметов в пространстве [6]. Роботы также представляют собой устройства управляемые дистанционно, для выполнения задач на больших глубинах, в космосе и т.д.

Робототехника являет собой такую научную область, в рамках которой изучаются вопросы выстраивания систем технического порядка, которые по функциональности полностью идентичны работе тела человека.

Робототехника сосредоточена на вопросах выстраивания систем технического порядка, что по функциям полностью идентично базовым важнейшим системам организма людей. Те задачи, которые обозначены как конечная цель робототехники и есть процесс, естественным образом определяющий направленность развития.

Робототехника включает в себя область механики, связанную с разработкой механических конечностей. Раздел робототехники занимающийся производством человекоподобных конечностей (рук или ног) для определенной производственной операции и приспособлением технологии производства изделий для возможностей конечностей роботов и именуется теперь промышленной робототехникой. Если смотреть в динамике, то именно эта отрасль развивается максимально стремительно. Именно тут удалось добиться максимальных высот и результативности, за счет того, что начался процесс активного внедрения роботов промышленного характера, теперь получается автоматизировать множество операций промышленного характера [7].

Есть несколько классификаций продукции отдела робототехники, мы рассмотрим стандартную классификацию роботов, а именно:

— те, которые имеют программное управление, их деятельность осуществляется в рамках заданного программой алгоритма, но можно менять порядок исполнения заданных программ;

— те, которые имеют управление адаптивного типа, у которых достаточно средств для очувствления. Именно по этой причине их деятельность возможна в рамках условий, которые не подчинены регламентам и могут меняться, они оснащен особыми и очень чувствительными сенсорными датчиками, но, адаптационные программы заранее продуманы и процесс их разработки окончен;

— роботы, у которых есть интеллектуальное управление, речь о пресловутом искусственном интеллекте, способном обрабатывать данные, за

счет чего формируются и значительно расширяются возможности создания некой поведенческой модели интеллектуального типа, которая будет максимально имитировать человеческое поведение в аналогичных условиях [8].

Главное отличие интеллектуальных систем в том, что конечная цель операции заранее неизвестна, но задан набор объектов и их характеристики, алгоритм работы с объектами, который позволяет перепланировать действия.

Манипуляционным устройством принято называть тип роботов с функциями, аналогичными функциями человеческой руки. Манипулятор может быть, как самостоятельным устройством, так и находиться в составе более сложного роботизированного комплекса. Звенья манипулятора имеют соединения, допускающие вращательное или поступательное движение [9].

1.2 Описание манипуляционного устройства

Манипуляционное устройство представляет собой пространственный механизм обладающий шестью степенями свободы, копирующий функциональные возможности человеческой руки [10]. Данное устройство содержит шарнирно соединенные звенья, вращаясь вокруг оси систем координат. Звенья устройства приводятся в движение с помощью сервоприводов, установленных на сочленении звеньев. Для позиционирования манипулятора необходимо постоянно контролировать положение и скорость движения звеньев, для этого в системе используется микроконтроллерное устройство.

Степени подвижности манипулятора:

– 1 – ая степень подвижности – вращение манипулятора. Вращение манипулятора осуществляется в пределах рабочей области 0-180°;

– 2 – ая степень подвижности – плечо. Управление сгибом плеча происходит в пределах 25-170°;

– 3 – я степень подвижности – локоть. Сгибание локтевой части манипулятора происходит в пределах 30-150°;

– 4 – ая степень подвижности – сгибание кисти. Сгибание кисти манипулятора происходит в пределах 0-180°;

– 5 – ая степень подвижности – поворот кисти. Поворот кисти манипулятора происходит в пределах 0-180°;

– 6 – ая степень подвижности – кисть. С помощью кисти манипулятор производит захватывающие движения, сгибание пальцев происходит в пределах 25-100°.

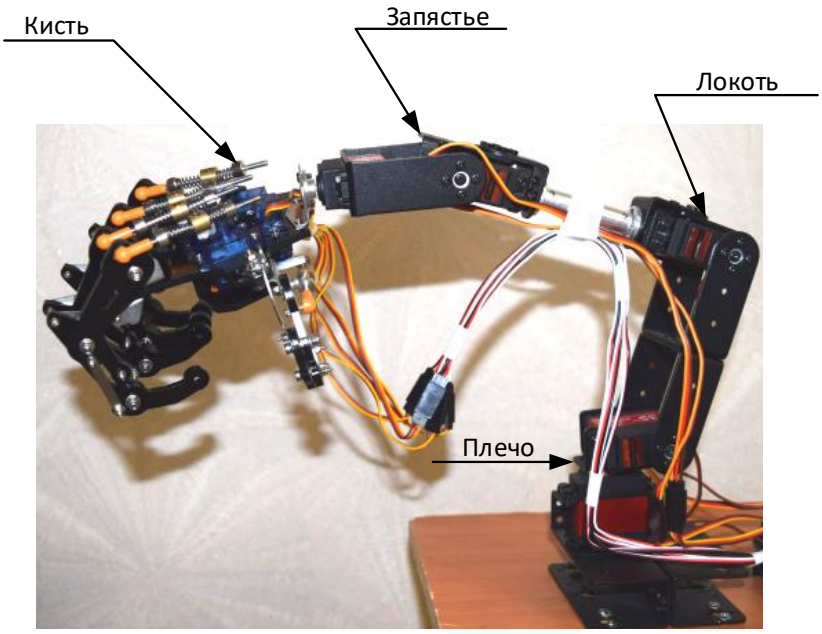

Рисунок 1 – Внешний вид манипуляционного устройства

Для управления манипулятором используется микроконтроллер, который обеспечивает перемещение его звеньев по заданному алгоритму и поддержку связи для обмена данными с смартфоном.

1.3 Программные среды для управления роботами

На данный момент имеется широкий выбор программных средств для управления роботами манипуляторами. Среды программирования делятся на две группы: визуальные и текстовые.

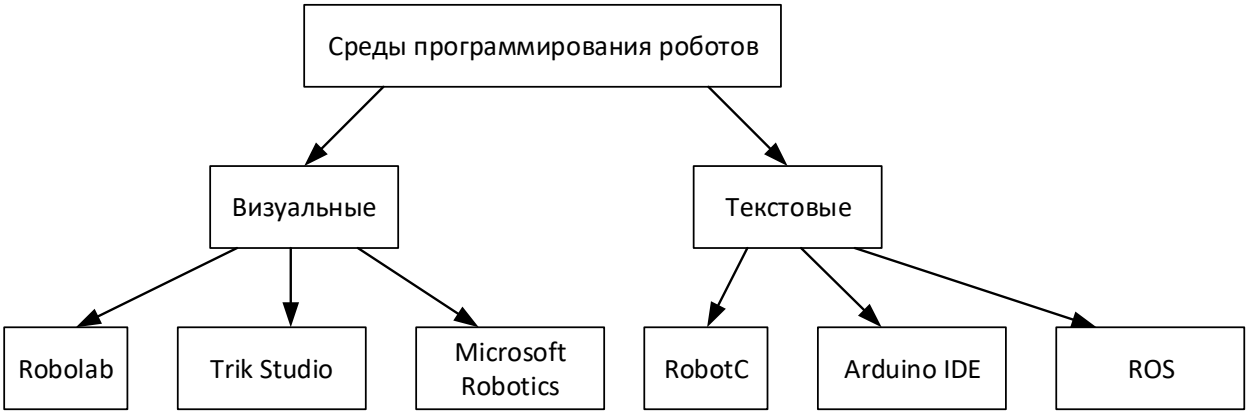

Рисунок 2 – Примеры сред программирования

Robolab является многофкнциональной средой графического программирования. Основой языка программирования является Си подобный язык, но в отличие от программирования на языке Си, Robolab базируется на использовании графических элементов при создании программ вместо многострочного кода. В Robolab предоставляется возможность проектировать схемы и выполнять по ним расчеты, после чего тестировать их. Robolab позволяет программировать несколько типов микроконтроллеров - Control Lab, RCX, NXT, также проводить независимые расчеты на компьютере [11].

Trik Studio является платным программным обеспечением, предоставляющее возможность разрабатывать управляющие программы для учебных роботов фирмы Trik. Интерфейс программы включает в себя симулятор, позволяющий выполнять тестирование созданной программы без использования реального робототехнического объекта [12].

Microsoft Robotics Developer Studio (MRDS) – это программное обеспечение, предназначением которого является создание управляющих программ для роботов, управление и их симуляции.

Visual Programming Language (VPL) – язык программирования визуального типа, который используется для того, чтобы описывать алгоритмы моделей поведения роботов, применяется начинающими программистами, если говорить о более профессиональном уровне, то используется C# [13].

RobotC является еще одним языком программирования, он базируется на C и у него есть среда для того, чтобы писать и отлаживать программы. Сегодня только он используется для того, чтобы программировать роботов, который предоставляет развитый режим отладки во время выполнения программ. RobotC является кросс-платформенным решением, которое позволит студентам и ученикам изучить C-подобный язык, используемый в большинстве образовательных и профессиональных приложений. Является платным программным обеспечением[14].

Arduino IDE – среда программирования роботов на базе Arduino. Данный софт позволяет компьютеру взаимодействовать с Arduino как для передачи данных, так и для прошивки кода в контроллер [15].

Robot Operating System (ROS) – это фреймворк для разработки программного обеспечения управления робототехническими устройствами. Данная среда включает в себя набор программных библиотек и инструментов, которые помогают создавать приложения для роботов [16].

Таблица 1 – Анализ сред программирования роботов

Название среды	Достоинства	Недостатки
1	2	3
Robolab	программа получается наглядной; легко прослеживается логика и алгоритм; Robolab позволяет организовывать циклы; подзадачи, переменные, массивы и т.д..	сложен в изучении; маленький функционал в точном управлении; отсутствует бесплатная версия.
Trik Studio	включает симулятор, что позволяет тестировать программы без реального робототехнического набора.	громоздкость программ; ориентирована под конкретного робота.
Microsoft Robotics	подходит для высокоуровневого программирования; включает симулятор, что позволяет тестировать программы без реального робототехнического набора; поддерживает управление различными роботами.	сложность определения переменных; громоздкость программ; невозможность загрузить программу в память робота.

Продолжение таблицы 1

RobotC	основано на языке программирования Си и обладает простой средой разработки.	распространение программы – платная.
Arduino IDE	точность управления микроконтролером; программирование основано на языке C++; включает большое количество библиотек для управления различными модулями.	готовые проекты занимают большое количество памяти микроконтроллера.
Robot Operating System	обширная документация; дружелюбное и многочисленное сообщество пользователей; наличие множества уже написанных пакетов.	сложная архитектура предоставляет проблему в использовании новичкам; стсутствие поддержки ОС Windows.

Вывод: исходя из вышеприведенного, для управления манипуляционным устройством выбирается платформа Arduino. Данная платформа имеет ряд преимуществ перед другими выше перечисленными: низкая стоимость, кросс платформенность, простота среды программирования, ПО с возможностью расширения и открытым исходным текстом.

1.4 Обзор программ для создания интерфейса пользователя

Создание человеко-машинного интерфейса на базе смартфона (ОС Android) может осуществляться в трех средах разработки. Рассмотрим популярные среды разработки приложений на ОС Android.

Eclipse – Бесплатная IDE, разработанная компанией Eclipse Foundation. Эта программа является базой, которая регулирует процессы приложений [17].

Преимущества Eclipse:

– имеет дополнительные функции (для серверной работы и анализа базы данных);

– имеет возможность подключения к модулям;

– имеет возможность работать в групповом режиме.

Недостатки Eclipse:

– не поддерживается компанией Google.

Intellij idea – подобно Eclipse, данная среда позволяет создавать приложения и программы на нескольких программных языках [18].

Преимущества Intellij idea:

– предусмотрен автозаполнитель методов;

– есть рефакторинг;

– интерфейс более понятный и лаконичный;

– поддерживает программирование на языке Java.

Недостатки Intellij idea:

– предоставляется только платная версия.

Android Studio – интегрированная среда разработки созданная компанией Google для проектирования и тестирования приложений под ОС Android. Программная среда предоставляет возможность создавать приложения на ПК ОС Windows, Mac и Linux. Android Studio содержит Android SDK, как и все версии ОС Андроид, все применяемые эмуляторы, чтобы запускать приложения и множество компонентов, чтобы тестировать и проводить отладку.

У Android Studio есть масса преимуществ, среди которых стоит выделить следующие:

— можно одновременно работать с несколькими языками программирования, в том числе, можно задействовать самые популярные, такие как C/C++, Java;

— наличие кодовых редакторов, которые максимально удобны в использовании и эксплуатации;

— можно разрабатывать приложения для гаджетов и ПК, приставок ТВ и тд;

— можно запускать тесты с целью выявления корректности работы для игрушек, приложений и разного рода утилит, можно делать замеры производительности, что в эмуляторе производится постоянно;

— проводить рефакторинг кода, который уже написан;

— обширная библиотека, где масса готовых шаблонов и элементов для того, чтобы успешно заниматься разработкой ПО;

— разрабатывать приложения для системы Андроид, а именно, ее последней версии;

— предварительно проверять готовое приложение и тестировать его;

— широкая панель инструментария для тестов игр и приложений;

— есть подробная инструкция, как стоит использовать Android Studio, которая размещается на официальном сайте.

Несмотря на уйму преимуществ, есть у нее и некоторые недочеты, а именно:

— в процессе тестов приложения требуется высокие показатели производительности ПК;

— невозможно создать серверный проект на Java для ПК, Android устройств [19].

Резюмируя все то, о чем сказано выше, нужно выбрать Android Studio, что обуславливается тем фактором, что у нее огромный инструментарий, есть возможности настройки, которые очень гибкие, можно сразу запускать тестирование, она поддерживает несколько языков и у нее есть встроенный эмулятор.

2 АЛГОРИТМ ПРОГРАММНОГО РЕШЕНИЯ ЗАДАЧИ

Для приведения в движение манипуляционного устройства на сочленении звеньев устанавливаются сервоприводы. Сервопривод – электрическое исполнительное устройство.

Управление сервопривода, с использованием импульсов деятельности переменного типа. Базовыми параметрами импульсов такого типа считаются: минимальные показатели длительности на фоне максимальной длительности и частотности повторений. Угол поворота сервопривода можно определить как длительность импульсов, что именуется как модуляция широтно-импульсного типа. Сам сервопривод может находится в ожидании импульса с шагом 20 мс, а длительность определяет, как далеко надо повернуть мотор. К примеру, импульс длительностью 1,5мс задает поворот мотора на 90 С. При получении сервоприводом команды на вектор перемещения, орган управления будет перемещен в заданное положение на удержание. Ели же среда внешней силы оказывает действие на сервопривод, в рамках удержания заданного положения, отчетливо видно ярое сопротивление тому, чтобы он переместился из этого положения [20]. Чтобы управлять сервоприводами применяется микроконтроллерная платформа Arduino.

Arduino – микроконтроллерная платформа основанная на базе микроконтроллеров семейства AVR. Которая имеет возможность управления различными устройствами, передавать и принимать данные [21].

Arduino используется для подачи управляющих сигналов сервоприводам, установленным на звеньях манипулятора и приводящие его звенья в движение. Для поддержания связи Микроконтроллер-Смартфон, посредством Bluetooth, реализуется текстовый протокол обмена данными, микроконтроллер получая команды от смартфона, выполняет обработку полученных команд и в зависимости от полученной команды управляет приводами установленными на сочленении звеньев манипулятора.

В приложении смартфона пользователю предоставляется выбор режимов управления манипулятором. В программе микроконтроллера содержатся алгоритмы работы управления звеньями манипулятора, который обрабатывает полученные сигналы позиционирования робота в значения угла поворота сервопривода и далее формирует соответствующую команду. Смартфон поддерживает связь с микроконтроллером через Bluetooth.

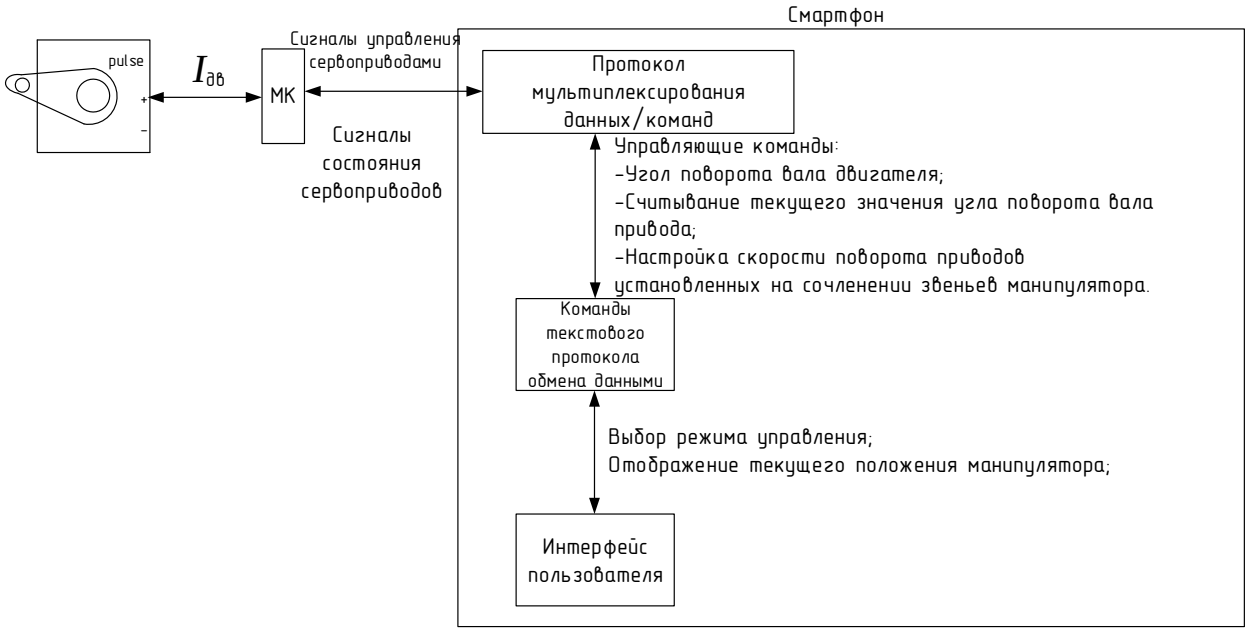

Рисунок 3 – Структурная схема системы управления роботом манипулятором

Протокол обмена данными представляет собой набор правил [22], позволяющие осуществлять взаимодействие между различными программами, в данном проекте между смартфоном и микроконтроллером. В реализованной системе управления используется протокол обмена данными, команды которого состоят из коротких текстовых строк, для обмена информацией между приложением верхнего уровня (Смартфон) и микроконтроллером. Набор команд начинается с символов «AT», в переводе с английского слова (attention) внимание. Стандартным ответом на команду является последовательность символов «OK».

После получения спец. символов сообщающих, о начале передачи команды (AT), следует последовательность символов, чисел, параметров. Заканчивается команда символами перевода строки «\r\n», где символы «\r» и «\n» возврат в начало и переход на новую строку соответственно.

В нашем случае мы будем использовать эти символы в десятеричной форме, где 13 соответствует перемещению курсора в начало строки, а 10 соответствует переходу на новую строку [23].

Таблица 2- Набор используемых команд

Команда	Описание
1	2
– AT	Проверка связи. Возвращает “OK”. Команда представляет последовательную запись символов («AT», 13, 10)
– ATA1 – ATA2 – ATA3 – ATA4 – ATA5	Данная команда передает угол поворота вала двигателя для управления звеньями 1-5 и пальцами руки методу MoveTo() класса ServoClass. Команда представляет последовательную запись символов («ATA1=», angle, 13, 10), где angle заданный угол поворота вала привода.
– ATS1 – ATS2 – ATS3 – ATS4 – ATS5 – ATS6	Данная команда передает значение скорости поворота вала двигателя для управления звеньями 1-5 и пальцами. Полученное значение в микроконтроллере пересчитывается в числовое значение задержки поворота вала. Команда представляет последовательную запись символов («ATS1=», speed, 13, 10), где speed значение скорости вращения.

Продолжение таблицы 2

1	2
– ATML – ATMR – ATMU – ATMD – ATMB1 – ATMB2 – ATMF1 – ATMF2 – ATMS1 – ATMS2	Команда предназначена для ручного управления манипулятором. Команда представляет последовательную запись символов («ATML», 13, 10). Функциональные предназначения команд: – ATML/ATMR – поворот манипулятора влево/вправо (от англ. left/right); – ATMU/ATMD – опускание/подъем кисти манипулятора (от англ. up/down); ATMB1/ATMB2 – поворот кисти по часовой и против часовой стрелки соответственно (от англ. brush); – ATMF1/ATMF2 – сжатие/разжатие кисти соответственно (от англ. fingers); – ATMS1/ATMS2 – управление положением плеча (от англ. shoulder).
– ATR	Команда предназначена для получения текущего положения манипулятора. Команда представляет последовательную запись символов («ATR», 13, 10). Ответом на данную команду является строка с значениями положений вала приводов установленных на сочленении манипулятора.

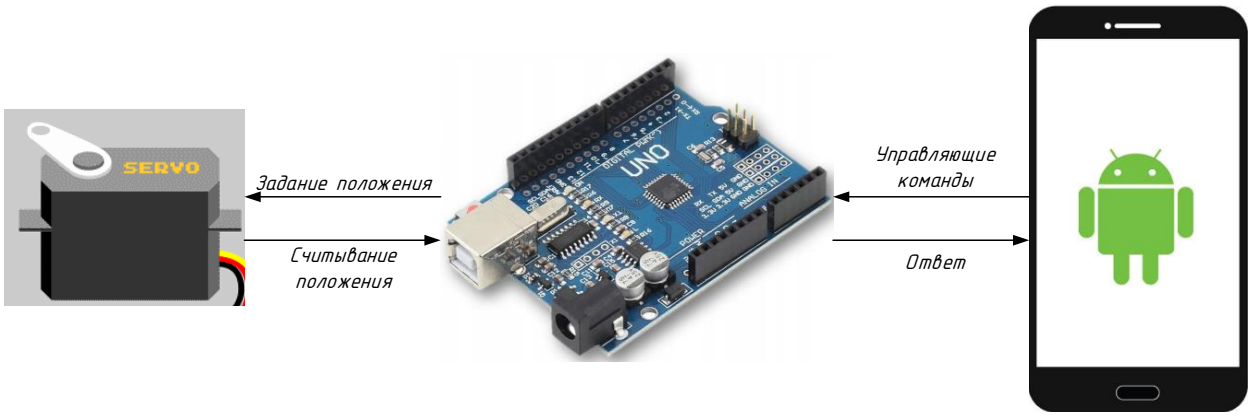

Рисунок 4 – Взаимодействие программных модулей системы управления

В разработанной управленческой системе обязательно присутствие 3 модулей программного типа, как указано на рисунке выше:

— управленческий модуль, который необходим для того, чтобы управлять приводом, чтобы позиционировать звенья манипулятора;

— модуль для обработки команды протокола в текстовом виде, чтобы наладить протокол обмена информацией;

— модуль интерфейса пользователя, чтобы осуществлять управление устройством манипуляционного типа, реализует это человек-оператор.

Сам по себе модуль контроля сервопривода сохраняется в памяти микроконтроллера и приставляет собой класс, включающий в себя перечень функциональных составляющих для управления сервоприводом, которые установлены на сочленении звеньев манипулятора.

Модуль, который обрабатывает команды текстовых протоколов обмена информацией тоже сохранен в памяти микроконтроллера, чтобы обеспечить разбор команд, которые были получены.

Этот модуль необходим для поддержания связи между смартфоном и микроконтроллером.

Модуль пользовательского интерфейса модуль размещается на смартфоне оператора, в состав ПО входят несколько режимов управления и меню настроек скорости перемещения звеньев манипулятора, также обеспечивает настройку сети Bluetooth. Главной задачей пользовательского интерфейса является взаимодействие человека с манипулятором.

Для того чтобы реализовать управление звеньями манипулятора с использованием текстового протокола, необходимо представить задачу в виде отдельных простых блоков выполняющих полное законченное действие.

Рассмотрим алгоритм работы программы для микроконтроллера.

2.1 Алгоритм управления звеньями манипулятора по текстовому протоколу

При запуске микроконтроллера программа переходит к выполнению первых команд.

Программа начинается с объявления используемых библиотеки ServoClass и переменных. Библиотека ServoClass предназначена для управления сервоприводами: их настройка, регулирование скорости и проверка текущего положения. Также вызывается функция setup для определения режимов работы портов ввода-вывода микроконтроллера.

Затем идет алгоритм для базового цикла программы, этот самый основной цикл необходим для проверки последовательности символов, которые поступили на микроконтроллер с порта UART, в той ситуации, если же было обнаружено совпадение символов с той командой, которая была задана, реализуется процесс выполнения алгоритма действий по управлению сервоприводом, согласно той команде, которая была получена. Теперь подробно рассмотрим работу базового цикла программы.

Она начинается с проверки доступных байт для чтения из последовательного интерфейса связи. В случае нахождения доступных байт в буфере последовательного порта, происходит их запись в массив Text, одновременно ведется проверка истечения тайм-аута между символами. Время тайм-аута необходимо для записи пришедших данных в массив, в котором хранится полученная последовательность символов (см. рис. 5).

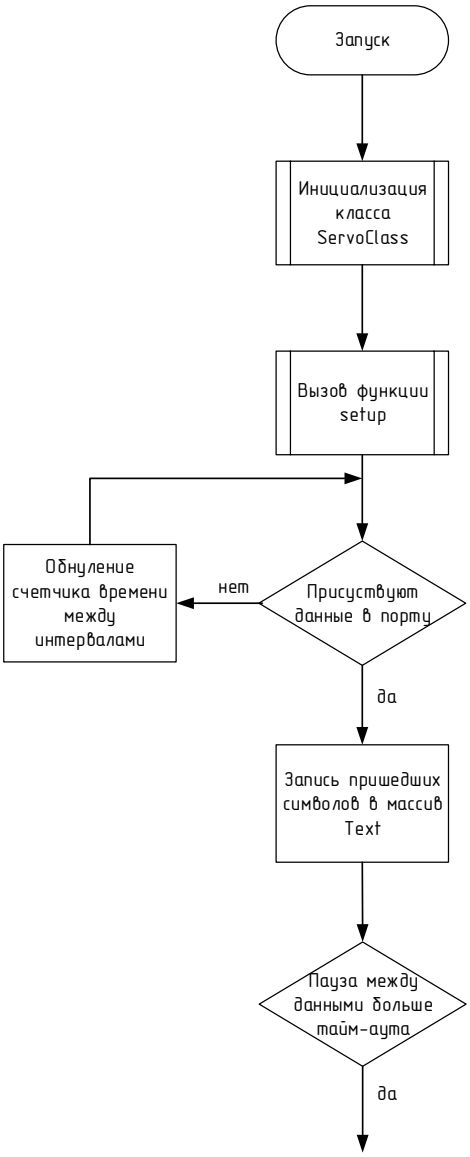

Рисунок 5 – Начало работы программы микроконтроллера

На рисунке 6 приведен алгоритм по обработке команды типа «AT», что соответствует обработке команды по проверки взаимосвязи смартфона и микроконтроллера, затем, когда время закончится, программа начнет обрабатывать компоненты Text, в котором есть полученная команда, будет анализирование соответствия полученной команды с последовательностью символов команды, которые могут быть заданы. Если же совпадения не обнаружено, то обработка подпрограммы компонентов Text, выходит, а при совпадении последовательности символов с заданной командой происходит

вывод сообщения «OK», об успешной обработки команды в последовательный интерфейс связи и очистка массива Text.

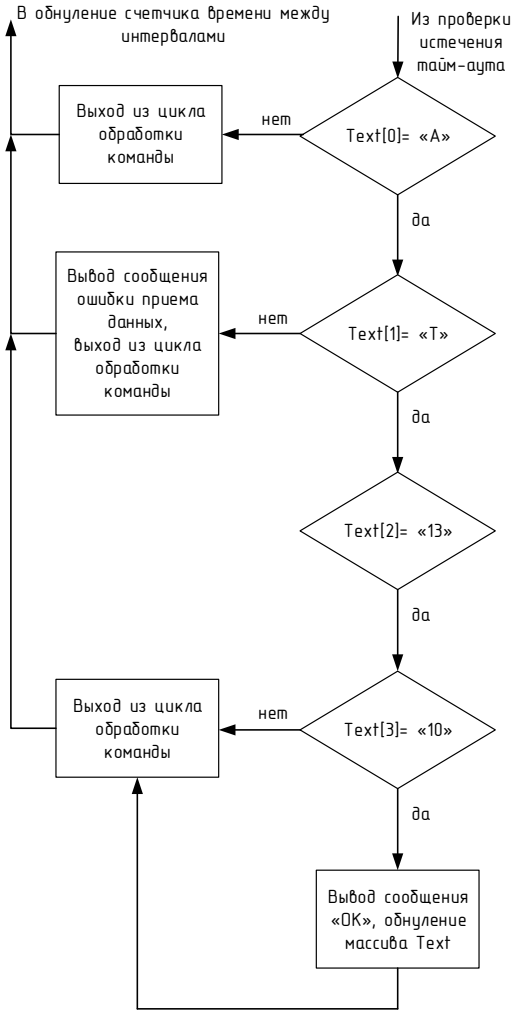

Рисунок 6 – Обработка команды проверки связи «AT»

При несовпадении третьего символа массива с кодом символа возврата каретки – «13», происходит проверка элемента массива Text на совпадение с символом «A».

После успешной обработки команды следует обработка заданного угла поворота вала сервопривода. Число представляет последовательность цифр, которая перезаписывается в временный массив до момента обнаружения символа конца передачи команды.

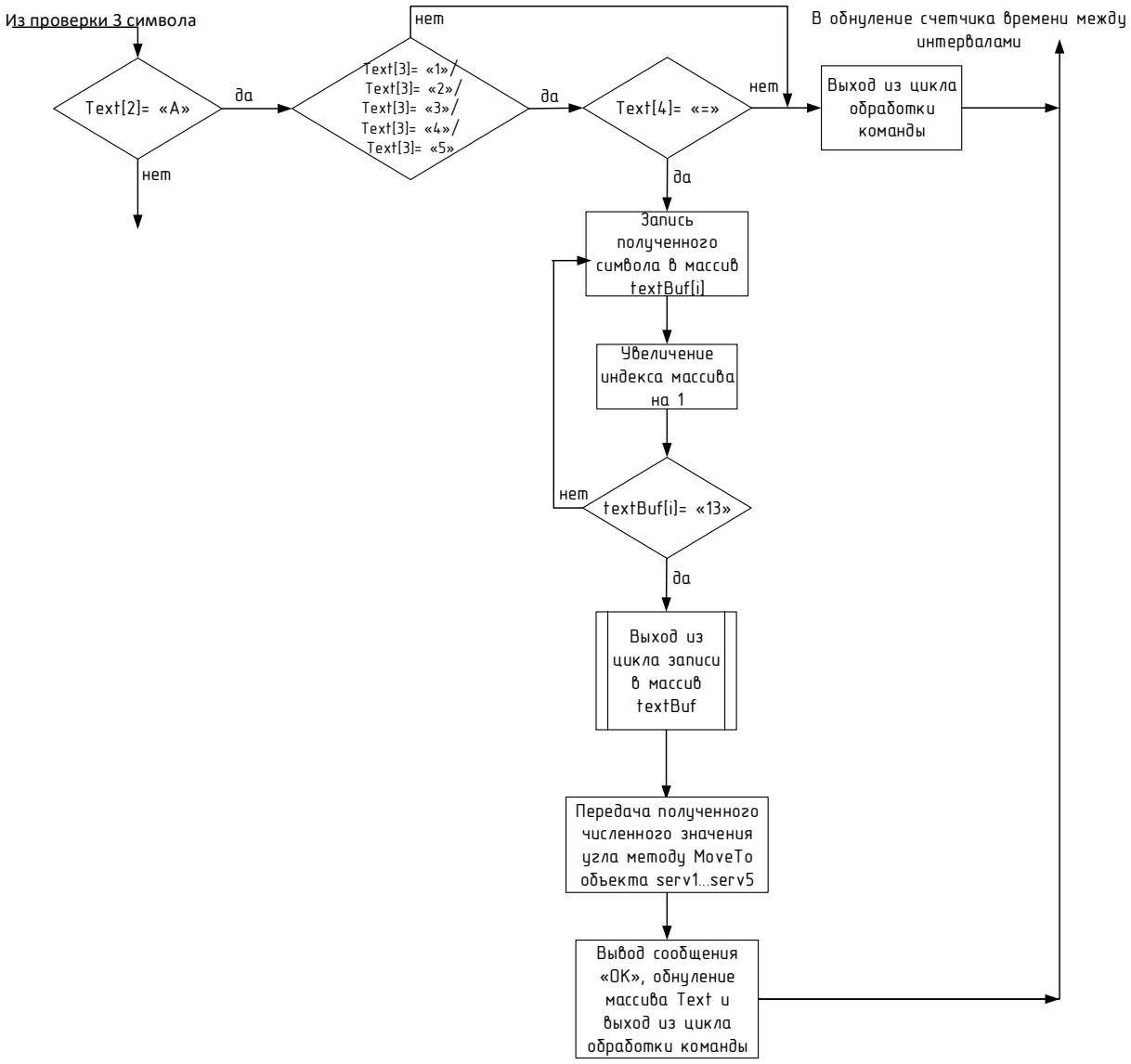

Рисунок 7 – Обработка команды передачи значения угла поворота сервоприводу «ATA1» … «ATA5»

Далее в зависимости от команды определяется для какого сервопривода пришла эта команда. После происходит передача численного значения угла поворота методу MoveTo объектов класса ServoClass: serv1, serv2, serv3, serv4, serv5. Следом происходит вывод сообщения «OK» об успешной обработки команды в последовательный порт и обнуление содержимого массива Text (см. рис. 7).

Если же третий символ массива Text не совпадает с символом «A», то следует проверка на совпадение с символом «S». В случае несовпадения символа

ни с одним из вышеперечисленных, продолжается проверка на совпадение с следующими символами. При удачной идентификации символа «S», аналогично команде передачи значения угла поворота вала привода, осуществляется обработка заданного значения скорости вращения вала сервопривода. Число представляет последовательность цифр, которая перезаписывается в временный массив до момента обнаружения символа передачи команды.

Далее согласно пришедшей команде определяется для какого сервопривода пришла эта команда. После происходит сохранение значения скорости поворота в переменные: speed1, speed2, speed3, speed4, speed5, speed6, и происходит вывод сообщения «OK» об успешной обработки команды в последовательный порт и обнуление содержимого массива Text (см. рис. 8).

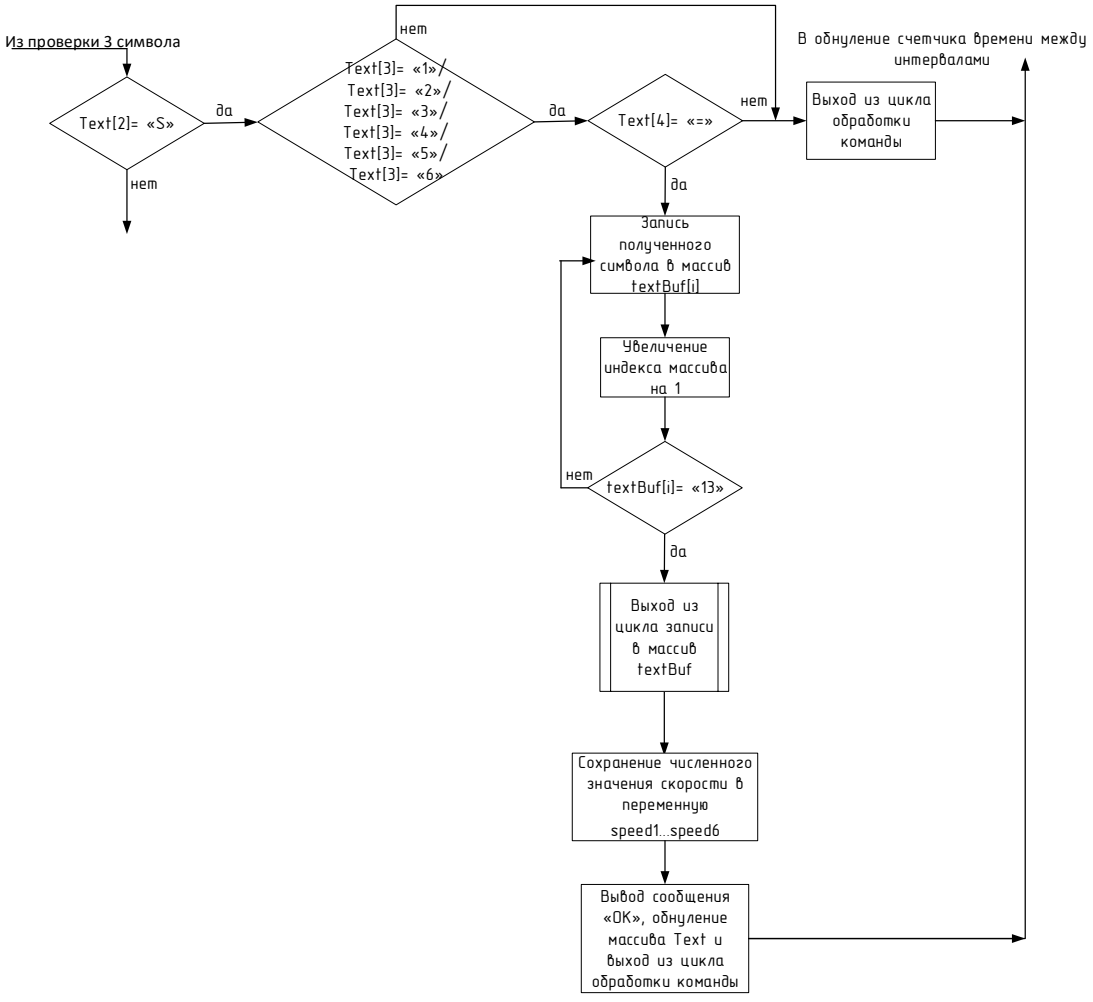

Рисунок 8 – Обработка команды передачи значения скорости поворота сервопривода «ATS1» … «ATS6»

При совпадении третьего символа массива Text с символом, код которого совпадает с кодом символа «M», происходит проверка четвертого символа массива. В зависимости от четвертого символа команды происходит соответствующее управление манипулятором: если четвертый символ массива «L», то происходит поворот манипулятора налево, «R» - поворот направо, «U» - подъем манипулятора, «D» - опускание манипулятора. При выполнении команд поворота кисти, сгибания пальцев и поворота плеча происходит проверка пятого символа массива. Если четвертый символ массива «B» происходит поворот кисти в зависимости от пятого символа массива вращение происходит по часовой стрелке – «2» и против – «1», «F» - происходит управление пальцами манипулятора, сгибание если пятый символ массива – «1», разгибание пальцев – «2», если четвертый символ массива – «S», происходит сгибание плеча и разгибание. Процесс движения происходит до тех пор пока сервоприводы не достигнут заданных критических значений угла поворота вала или в последовательный порт микроконтроллера не поступит новая команда (см. рис. 9,10).

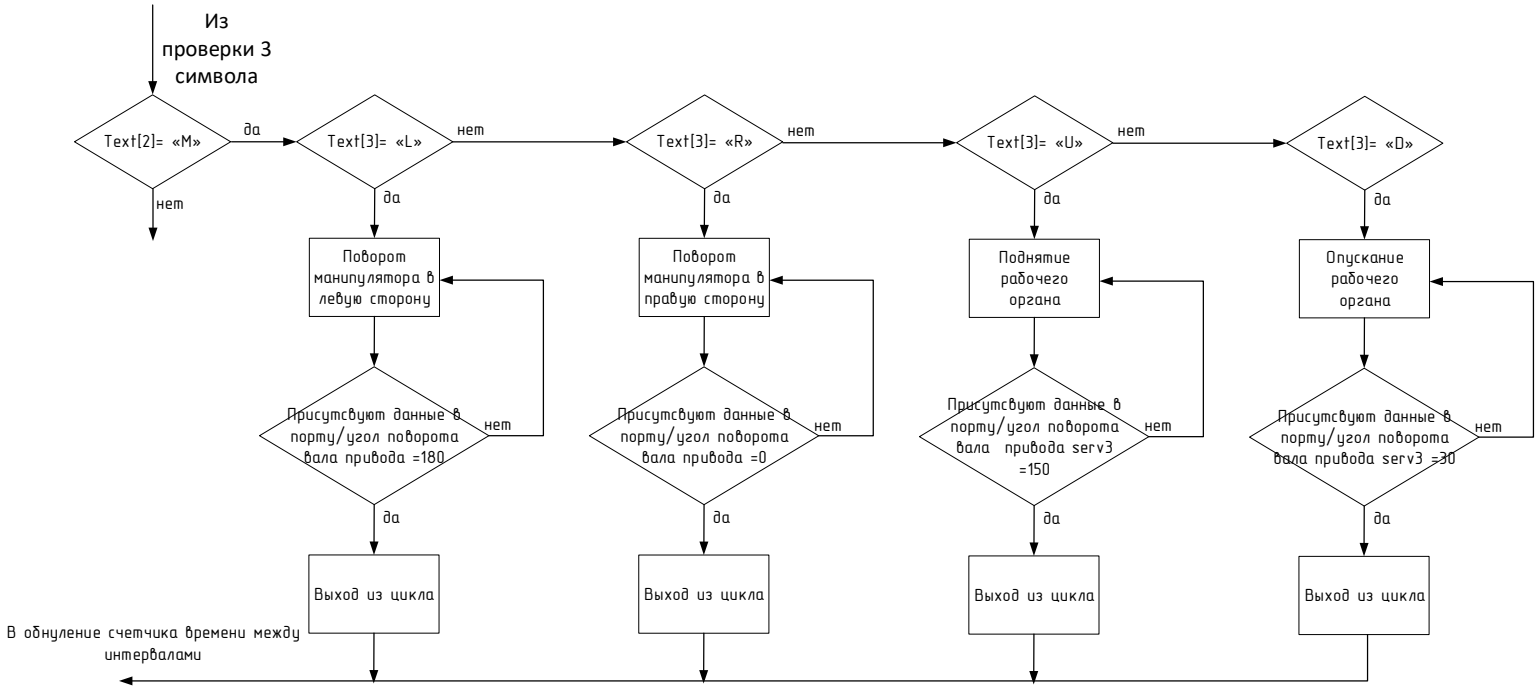

Рисунок 9 – Обработка команд ручного управления

Рисунок 10 – Обработка команд ручного управления

При обнаружении третьего символа массива Text на совпадение с символом «R», то выполняется отправка значений угла поворота вала приводов

в последовательный порт микроконтроллера и обнуление содержимого массива Text (см. рис. 11).

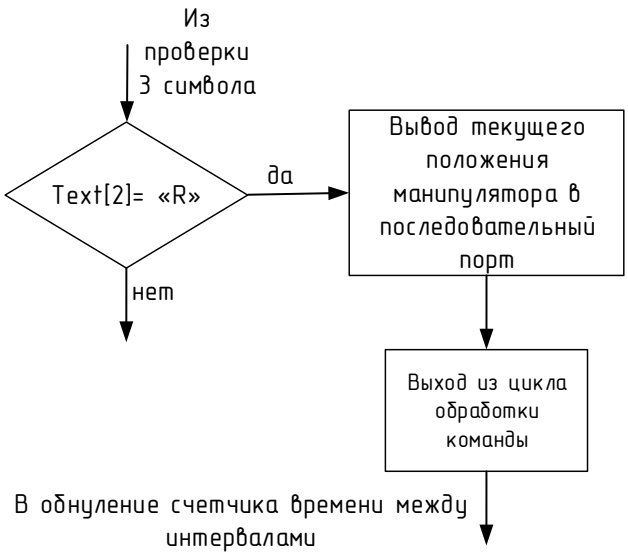

Рисунок 11 – Обработка команды получения текущего положения манипулятора

2.2 Алгоритм работы пользовательского ПО на смартфоне (ОС Android)

Для управления манипуляционным устройством с смартфона (ОС Android) разрабатывается ПО обеспечение которое включает в себя следующие функции:

– включение адаптера Bluetooth;

– поиск доступных устройств с включенным Bluetooth – адаптерами;

– подключаться к выбранному устройству;

– обмениваться командами и ответами на них используя текстовый протокол обмена данными;

– регулировка скорости перемещения звеньев манипуляционного устройства;

– вывод информации пользователю о текущем занимаемом положении манипулятора;

– возможность управления манипуляционным устройством в ручном, в позиционном режиме.

Рассмотрим алгоритм работы ПО управления манипуляционном устройством.

При запуске приложения выполняется проверка используемого телефона на наличие Bluetooth модуля, это необходимо для избежания дальнейших ошибок при работе в ПО. Для того чтобы подключиться к манипулятору посредством Bluetooth связи пользователю необходимо включить Bluetooth модуль. При включении Bluetooth система Android запросит разрешение на использование Bluetooth в приложении.

Для подключения к доступным устройствам запускается сканирование или выполняется запрос списка ранее сопряженных устройств.

При сканировании осуществляется поиск доступных Bluetooth модулей вокруг смартфона пользователя, процесс сканирования занимает 10-15 сек.. Если в поле досягаемости окажется устройство с разрешенным Bluetooth, оно отправит в ответ на запрос некоторую информацию о себе: имя, класс, свой уникальный MAC адрес. На основе этой информации можно организовать соединение и передачу данных [24].

При подключении к ранее сопряженным устройствам на экран смартфона выводятся уже известные системе Bluetooth устройства, т.е. те с которыми уже было выполнено соединение ранее.

Далее для работы с манипуляционным устройством пользователю необходимо выполнить настройки скорости перемещений звеньев и выбрать один из режимов управления, переход по элементам меню осуществляется только при условии, что манипулятор сопряжен с смартфоном оператора.

В режиме настройки оператору предоставляется окно с настроечными полями, после заполнения всех полей нажимается кнопка «Установить», после нажатия выполняется проверка заполненных полей на наличие данных. В случае если одно из указанных полей не заполнено на экран выводится сообщение «Введите все необходимые данные для настройки» и продолжается ввод данных.

После успешного заполнения всех полей настройки происходит считывание введенных данных и формирование управляющих команд микроконтроллеру с последующей отправкой по Bluetooth. В результате успешной обработки полученных данных микроконтроллером на экран смартфона выведется сообщение, полученное от микроконтроллера, об успешно выполненной операции. Если пользователем нажимается кнопка «Сброс», то очищаются все ранее заполненные поля. При нажатии на «Меню», выполняется выход в начальное окно, где предоставляется возможность выбрать другой режим управления.

Режим позиционного управления имеет схожий функционал и режим работы как в настройках. На экран пользователя выводятся поля с помощью которых происходит управление занимаемым положением манипулятора. При нажатии на кнопку «Установить», выполняется проверка на корректность введенных данных пользователем, в случае неуспешной проверки на экран пользователя выводится сообщение «Проверьте введенные данные для позиционирования», диапазон задаваемого положения указан в полях ввода в виде подсказок. После успешной проверки введенных данных происходит формирование управляющих команд микроконтроллеру с последующей отправкой по Bluetooth. В результате успешной обработки полученных данных микроконтроллером на экран смартфона выведется сообщение, полученное от микроконтроллера, об успешно выпаленной операции. При нажатии на кнопку «Проверка» в микроконтроллер отправляется запрос в виде соответствующей команды, в ответ на данную команду смартфон получает значения текущего положения манипулятора, после чего полученные данные обрабатываются и выводятся в поля значений положений. Кнопка «Меню» предназначена для выхода в начальное окно, где предоставляется возможность выбрать другой режим управления.

При работе в ручном режиме пользователю предоставляется окно с управляющими кнопками для управления изменения положения манипулятора. При удержании кнопок в микроконтроллер отправляется соответствующая

команда при выполнении которой происходит перемещение звеньев манипуляционного устройства. При нажатии на кнопку «Меню» происходит переход на начальный экран (см. рис. 12).

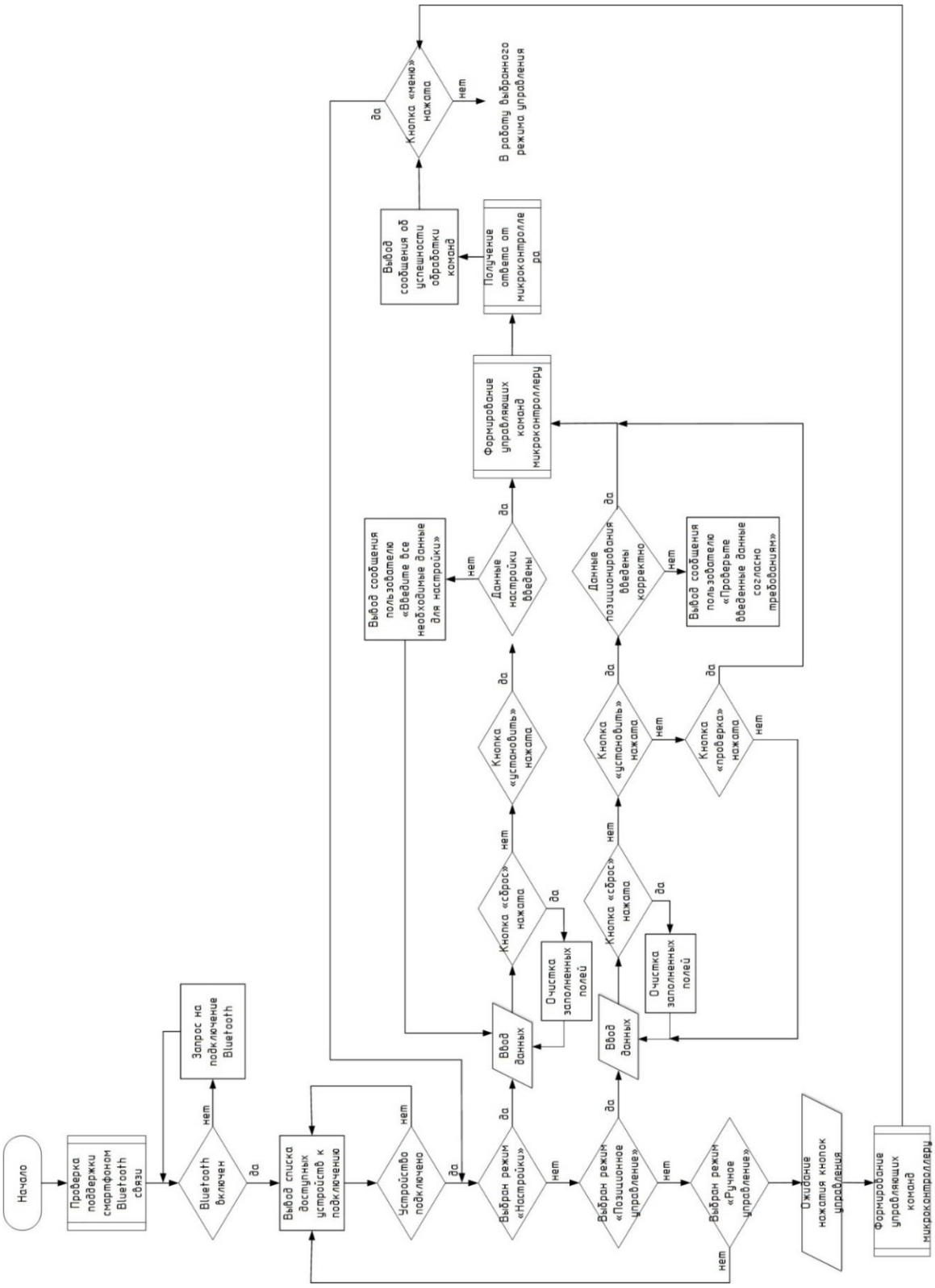

Рисунок 12 – Алгоритм работы пользовательского ПО на смартфоне

2.3 Обзор практических возможностей профильного программного обеспечения

2.3.1 Обзор возможностей ПО для разработки мобильного приложения

Android Studio — интегрированная среда разработки для Android-программирования, созданная на базе популярной среды IntelliJ IDEA.

Установка Android Studio включает в себя:

– Android SDK — последняя версия Android SDK;

– инструменты и платформенные средства Android SDK — средства отладки и тестирования приложений;

– образ системы для эмулятора Android — позволяет создавать и тестировать приложения на различных виртуальных устройствах [25].

Android studio содержит различный инструментарий, для умпешной разработки приложения. В связи с тем, что на базе Android studio присутствует модуль SDK исчезает необходимость отладки приложения на реальном устройстве, чтобы посмотреть результаты проделанной работы. Встроенный эмулятор Андроид позволяет посмотреть, как работает приложение в нужных для вас условиях [26].

Инструмент Анализатор APK показывает значение размера и количество файлов, которые содержит разрабатываемый проект. Так же присутствует возможность сравнения двух пакетов приложений – старая и новая (см. рис. 13)[27].

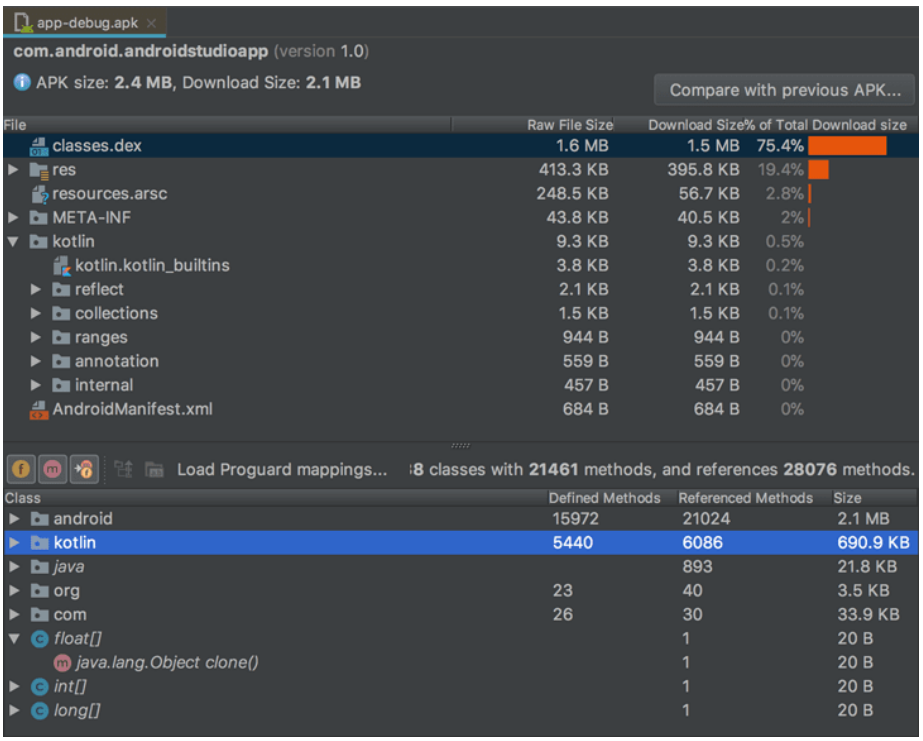

Рисунок 13 – Анализатор APK в Android Studio

Для создания и настройки пользовательского интерфейса, используется инструмент – Редактор макетов. Данный инструмент обеспечивает возможность создавать и редактировать элементы пользовательского интерфейса вручную с помощью перетаскивания необходимых элементов (см. рис. 14).

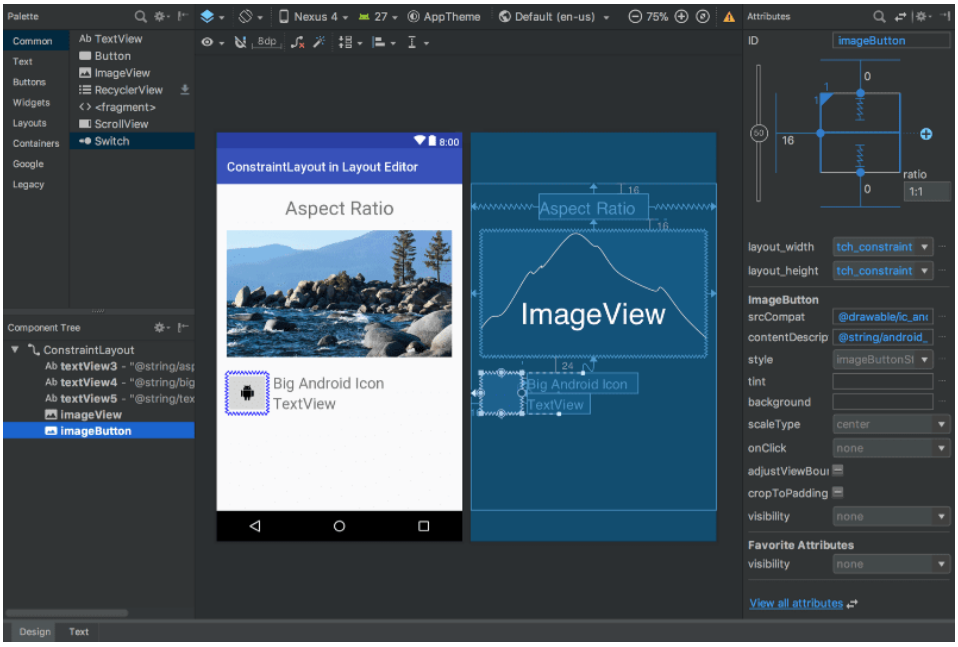

Рисунок 14 – Редактор макетов в Android Studio

Инструмент Профилирование, предоставляет данные потребления ресурсов в реальном времени. Доступен график просмотра использования CPU мобильного устройства, расход заряда батареи, использование памяти и сети. Благодаря данному набору инструментов облегчается процесс оптимизации разрабатываемого ПО и уменьшения потребления используемых ресурсов. (см. рис. 15).

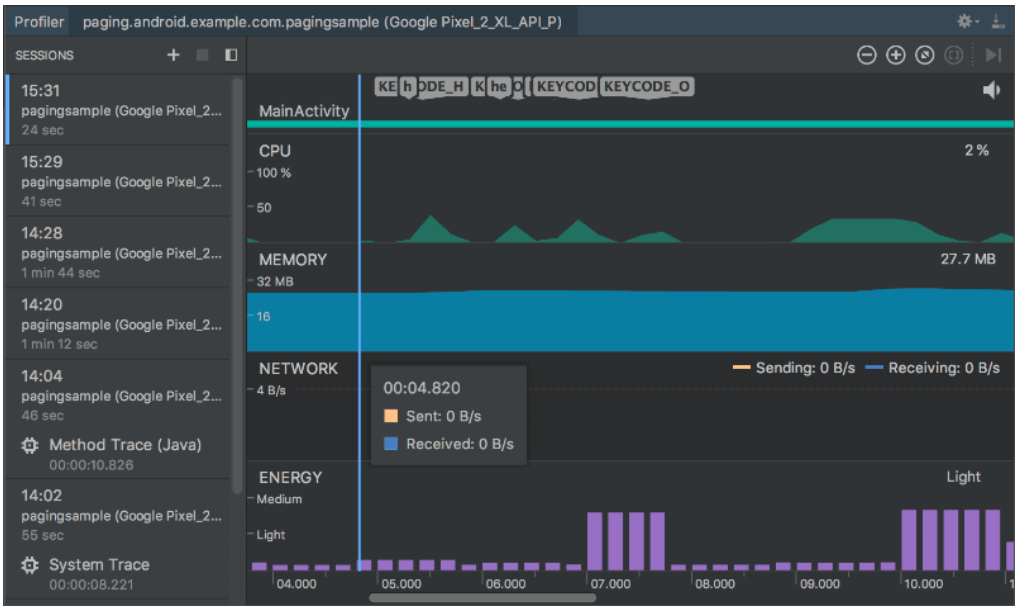

Рисунок 15 – Инструмент Профилирование в Android Studio

Гибкая система сборки приложения, позволяет получать на выходе разные конфигурации программы, для определенных задач или целевой аудитории:

– отладочная или окончательная версия;

– для платного и/или бесплатного распространения;

– с определенным набором возможностей под разные регионы;

– оптимизированная сборка под определенную версию Android или разрешение экрана.

Среди других возможностей так же стоит отметить:

– перепроектирование кода, не меняя общее поведение программы;

– встроенную утилиту для подписи приложений;

– наличие шаблонов основных макетов и компонентов;

– инструменты для нескольких языков программирования [28].

2.3.2 Обзор возможностей ПО разработки управляющей программы на Arduino

Arduino IDE – интегрированная среда разработки для Windows, MacOS и Linux, разработанная на базе языков программирования такие, как Си и C ++, обеспечивает создание и загрузку программы на Arduino и прочие совместные платы других производителей. Данный софт позволяет компьютеру взаимодействовать с Arduino как для информацией, так и для прошивки кода в микроконтроллер [29].

Интерфейс среды разработки Arduino содержит следующие основные элементы, которые перечислены на рисунке 16 – Редактор кода Arduino IDE:

1. Пункты меню. Через пункты меню Вы можете получить доступ ко всем функциям Arduino IDE.
2. Проверить скетч (компилировать). Проверка кода программы на ошибки.
3. Загрузить скетч в Arduino. Компиляция кода и его загрузка в плату.
4. Вкладка с названием текущего скетча. Имя Вашего скетча.
5. Создать новый скетч. Новое окно с чистой рабочей областью.
6. Открыть скетч. Список программ из папки со скетчами.
7. Сохранить скетч. Сохранение скетча в указанную Вами папку.
8. Номера строк. Количество и номера строк, которые занимает код.
9. Монитор последовательного порта. Окно для обмена сообщениями с платой через COM-порт.
10. Управление вкладками. Переключение между вкладками с разными скетчами.
11. Область вкладок. Область вкладок для быстрого переключения между скетчами.
12. Рабочая область скетча. Текстовый редактор для написания кода программы.

13. Область уведомлений. Область, информирующая пользователя обо всех событиях при записи или проверке кода. В том числе, здесь будут появляться и сообщения об ошибках.

14. Название платы. Модель выбранной платы.

15. Номер порта. COM-порт, к которому подключена плата [30].

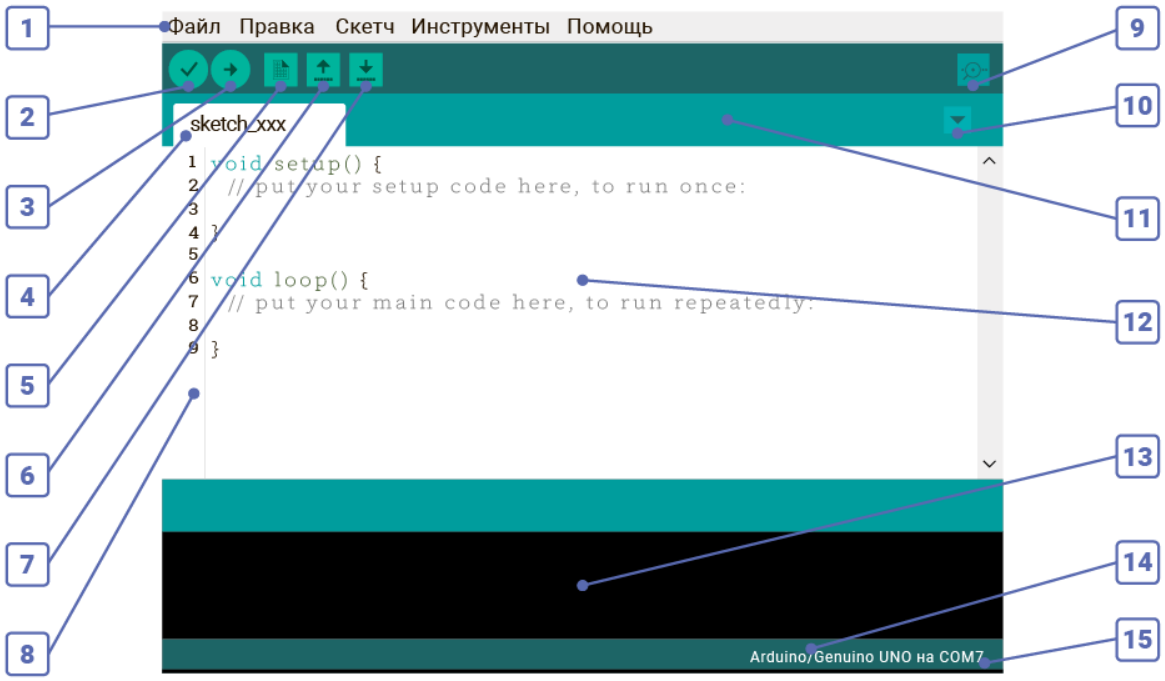

Рисунок 16 – Редактор кода Arduino IDE

Преимущества Arduino IDE

– доступность;

– удобный для использования и понимания интерфейс;

– программа совместима со всеми версиями операционных систем Windows;

– наличие необходимых для работы инструментов;

– несколько вариантов языков программирования;

– возможность углубить знания языка C++;

– встроенный набор примеров программ;

– функции сохранения, экспорта, проверки, поиска, замены скетчей.

2.4 Обоснование выбора программно-технического обеспечения и его подробная характеристика

Программно – техническое обеспечение решения задачи представляет собой совокупность технического и программного обеспечения.

2.4.1 Обоснование выбора электронных элементов системы

Для управления звеньями манипулятора используются устройство с электрическим мотором – сервомоторы (сервоприводы). Сервопривод управляется с помощью импульсов переменной длительности. Преимущества сервоприводов перед их аналогами – шаговыми двигателями в том, что они имеют обратную связь, осуществляют точное позиционирование, обладают меньшим уровнем шума. По сравнению с шаговыми двигателями, сервоприводы управляются по одному каналу связи. В данном проекте для управления звеньями манипулятора используются сервоприводы TD-8135MG, для управления движениями пальцев - SG90 (см. рис. 17).

Технические характеристики используемых сервоприводов TD-8135MG:

- материал редуктора: Сталь;
- размер - 40 x 20 x 40,5 мм;
- скорость - 0,26 сек/60 гр. (4,8В), 0,22 сек/60 гр. (6В);
- угол поворота 180 градусов;
- усилие - 32,7 кг/см (4.8В), 35,2 кг/см (6В);
- рабочее напряжение - 4.8В - 6.6В;
- рабочая температура - 0 С - 55 С;
- разъем - JR (Fits JR and Futaba);
- вес - 60 гр. [31].

Технические характеристики используемых сервоприводов SG90:

- скорость (4.8В): 0.14 сек/60 градусов;
- усилие (4.8В): 1.98 кг*см;
- угол поворота: 180 градусов;
- материал редуктора: пластик;

– рабочая температура: -30 – +60 °С;
– рабочее напряжение: 3.5 – 8.4В;
– габариты: 22.6 х 21.8 х 11.4 мм;
– масса: 13 г. [32].

Рисунок 17 – Внешний вид используемых сервоприводов

Микроконтроллерная платформа Arduino не поддерживают беспроводную связь, а в данном проекте это необходимо. Т.к. в проекте будет использоваться удаленное управление манипулятором, необходимо использовать Bluetooth-модуль подключаемый к Arduino.

Для того, чтобы организовать процесс связи двустороннего типа с использованием технологии Bluetooth, которая свяжет микроконтроллерную платформу и смартфон, наиболее подходит модуль HC-05 (см.рис.18), он может работать на поиск устройства и устанавливать связь, а еще можно пользоваться принципами реализации ведомого устройства, то есть slave.

Технические характеристики модуля HC-05:

– чип Bluetooth: HC-05(BC417143);
– диапазон частот радиосвязи: 2,4–2,48 ГГц;
– мощность передачи: 0,25–2,5 мВт;
– чувствительность: –80 dBм;
– напряжение питания: 3,3–5 В;

– потребляемый ток: 50 мА;
– радиус действия: до 10 метров;
– интерфейс: последовательный порт;
– режимы: master, slave;
– температура хранения: –40…85 °С;
– рабочий диапазон температур: –25…75 °С;
– габариты: 27 х 13 х 2,2 мм [33].

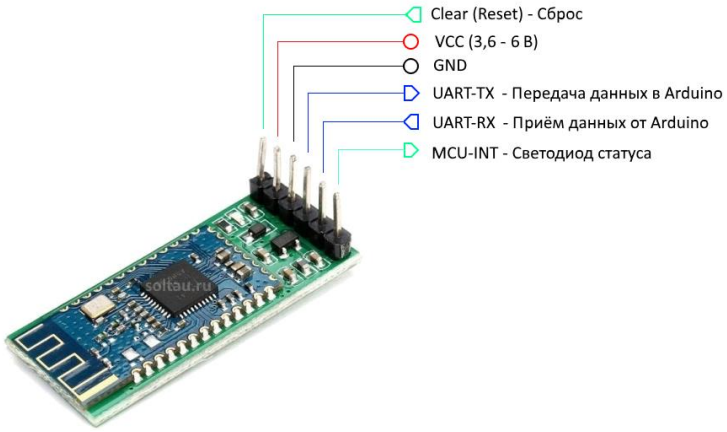

Рисунок 18 – Bluetooth модуль HC-05

Для управления звеньями манипуляционного устройства по выше приведенному алгоритму используется микроконтроллерная платформа Arduino UNO R3. В настоящее время существует огромное количество программируемых микроконтроллеров которые подойдут для выполнения поставленных задач, но у Arduino имеется ряд преимуществ перед аналогичными микроконтроллерами:

– небольшие размеры платы;
– программирование на языке C++ с дополненными функциями ввода/вывода на контактах платы;
– большое количество различных модулей;
– подключение к внешней периферии без пайки;
– имеет встроенный программатор;
– стоимость контроллера.

Arduino — печатная плата на которой размещен микроконтроллер и набор элементов необходимых для поддержания стабильной работы. На плате также имеются контакты, к которым можно подключать всевозможные компоненты: лампочки, датчики, моторы, чайники, роутеры, магнитные дверные замки и вообще всё, что работает от электричества.

В процессор Arduino (см. рис. 19) можно загрузить программу, которая будет управлять всеми этими устройствами по заданному алгоритму [34].

Плата Arduino Uno версии R3 обладает следующими техническими характеристиками:

– максимальная величина выходного тока пина с напряжением 5V: 0,8A;

– тактовая частота кварцевого процессора: 16 МГц;

– количество аналоговых и цифровых портов ввода-вывода: 20;

– число портов, поддерживающих широтно-импульсную модуляцию:6;

– разрядность аналого-цифрового преобразователя: 10 бит;

– максимально допустимая величина тока с пина ввода-вывода: 0,04 А;

– ОЗУ микроконтроллера: 2 КБ;

– флэш-память: 32 КБ (0.5 КБ которых отведены под bootloader);

– размерные параметры устройства: 6,9×5,3 см;

– допустимое входное напряжение на разъеме питания: 7-12 В;

– электрически стираемое перепрограммируемое ПЗУ EEPROM: 1 кб;

– количество встроенных светодиодов: 1 (расположен на 13 порту)[35].

Рисунок 19 – Arduino UNO R3

2.4.2 Обоснование выбора программных средств разработки

Arduino IDE – представляет собой такую среду разработки, которая является полностью интегрированной, она отлично подходит для взаимодействия со следующими операционными системами: Windows, MacOS и Linux, разработана для использования на языках Си и C++, используется для того, чтобы формировать и загружать программы на платы Arduino, или на совместимые с ней устройства, можно использовать иные платы, другого производства [36].

Это максимальна понятная и очень удобная среда, чтобы осуществлять процессы разработки, ориентированные на формирование своего ПО, которое будет осуществлять управление множеством разных устройств, которые были собраны начинающими электронщиками и их опытными коллегами.

Соединение ПК с микроконтроллером реализовано через интерфейс USB. Код на языке Си и C++ пишется в редакторе, в котором есть подсветка команд и спеллчекер [37].

Android Studio - это довольно новая IDE (интегрированная среда разработки), которая бесплатно предоставляется Google для разработчиков Android. Android Studio основана на IntelliJ IDEA, среде разработки, которая также предлагает отличную среду разработки Android [38].

3 КОМПЬЮТЕРНАЯ РЕАЛИЗАЦИЯ ВЫБРАННОГО АЛГОРИТМА РЕШЕНИЯ ИССЛЕДУЕМОЙ ЗАДАЧИ

3.1 Программная реализация алгоритма управления звеньями манипулятора

Для реализации управляющей программы на микроконтроллерной платформе Arduino UNO R3 были выбраны язык программирования C++ и среда программирования Arduino IDE.

Работа программы микроконтроллера начинается с объявления библиотек и переменных (см. рис. 20).

```
#include "ServoClass.h"

char Text[10];      // тестовый буфер
char  angleBuf[2]; // буфер для угла поворота
char  textBuf[4]; // тестовый буфер
int s;  // индекс массива Text[10]
int i; // индекс массива angleBuf[2]
#define TIME_OUT 50 // время тайм-аута между командами (* 0,25 мс)
int  timeOutCount; // счетчик времени между приемом данных
int angle = 0;  //угол поворота сервопривода
unsigned long timing; // Переменная для хранения точки отсчета
int speed1, speed2, speed3, speed4, speed5, speed6;//Переменная для хранения значения скорости
int polojeniye;// Положение сервопривода
int j, f, z; // счетчики угла поворота сервопривода при подъеме/опускании
unsigned int fig_1, fig_2, fig_3, fig_4, fig_5;

ServoClass serv1;// Вращение манипулятора
ServoClass serv2;// Поворот плеча
ServoClass serv3;// Сгибание локтя
ServoClass serv4;// Сгибание кисти
ServoClass serv5;// Поворот
ServoClass serv6;// Большой палец
ServoClass serv7;// Указательный палец
ServoClass serv8;// Средний палец
ServoClass serv9;// Безымянный палец
ServoClass serv10;// Мезинец
```

Рисунок 20 – Запуск программы

В начале работы программы объявляется библиотека ServoClass.h. Библиотека ServoClass.h представляет набор функций для управления сервоприводами.

Библиотека ServoClass.h является собственной библиотекой. Метод AttachServo служит для указания номера выхода микроконтроллера к которому подключен сервопривод и начальное значение положения вала, CheckPosition

возвращает текущее положение вала сервопривода, Stop останавливает вращение вала сервопривода, MoveTo предназначен для передачи значения угла и скорости вращения вала, Update предназначен для осуществление вращения вала сервопривода (см. рис. 21).

```
#include<Servo.h>
class ServoClass
{
    Servo serv; // Собственно сам сервопривод, управляемый стандартной
    // библиотекой servo
    int servPosition; // Текущая позиция сервопривода, градусов
    int posInit;  // Исходная позиция сервопривода, градусов
    // Переменные, которые отвечают за перемещение
    boolean move; // Должен ли двигатся ли сервопривод в настоящий момент
    int movePosition; // Позиция, которую должен принять сервопривод, градусы
    float moveSpeed;  // Скорость, с которой сервопривод должен двигаться, об/мин
    int moveIncrement;  // изменение положения за один интервал времени, градусов
    int moveInterval; // Интервал между движениями для обеспечения  скорости, миллисекунды
    unsigned long lastTimeCheck;  // Время последнего движения сервопривода, в мс от начала работы программы

  public:
    ServoClass ()
    {
      posInit = 0;
      servPosition = 0;
      lastTimeCheck = millis();
    }
    void AttachServo (int pin, int posInit = 0)
    {
      serv.attach(pin); // Подключаем сервопривод к указанному пину
      servPosition = posInit; // Позиция сервопривода должна сответствовать исходному положению
      serv.write(servPosition); // Даем команду изменить положение
    }
    int CheckPosition()
    {
      int polojeniye = serv.read();
      return polojeniye;
    }
    void Stop()
    {
      int polojeniye = serv.read();
      serv.write(polojeniye);
      Serial.println(Serial.read());
    }
    void MoveTo (int movepos)  // Если скорость не указана - просто даем команду на перемещение сервопривода
    {
      serv.write(movepos); // Даем команду изменить положение
    }
    void MoveTo (int movepos, float movespeed)  // Если скорость  указана - организуем плавное перемещение
    {
      movePosition = movepos;
      moveSpeed = movespeed; // Устанавливаем скорость, с которой должен перемещаться сервопривод
      int Angle = abs(movepos - servPosition); // На сколько должен переместиться сервопривод
      // вне зависимости от направления
      if (movepos >= servPosition) // Определяем, в какую сторону от текущего положения нам нужно двигаться
        moveIncrement = 1; // Если вперед - будем добавлять один градус за один интервал времени
      else
        moveIncrement = -1; //  Если назад - то отнимать
      float RotationSpeed = moveSpeed * 360 / 60; // Перводим скорость из оборотов в минуту в градусы за секунду
      moveInterval = (int)1000 / RotationSpeed; // Вычисляем интервал между перемещениями на один градус
      move = true; // Даем комнду начать плавное перемещение
    }
   void Update ()
   {
     if (((millis() - lastTimeCheck) >= moveInterval) && (move == true)) // проверяем, сколько прошло времени с последнего
       //действия и сравниваем с нужным нам интервалом.
       //Если времени прошло больше - совершаем следующее действие
     {
       servPosition += moveIncrement;    // Изменяем угол на вычисленное ранее значение
       serv.write(servPosition);   // Посылаем сервоприводу команду с новым углом поворота
       lastTimeCheck = millis(); // Обновляем счетчик времени, для того чтобы следующий отсчет времени начался с этого момента
       if (servPosition == movePosition) // Если мы достигли нужной позиции-
         move = false;   // остановить движение
     }
   }
};
```

Рисунок 21 – Библиотека ServoClass.h

Далее происходит объявление массивов, переменных и присвоение им начальных значений, также происходит объявление объектов класса ServoClass: serv1 – serv10 (см. рис. 20).

После объявления библиотек и глобальных переменных запускается функция void setup(), в которой происходит запуск работы последовательного интерфейса связи (UART) и задается ему скорость обмена данными 9600 бит/сек. После чего назначаем выходы к которым подключены сервоприводы и устанавливаем начальное положение вала (см. рис. 22).

```
void setup()
{
  Serial.begin(9600);//Запуск последовательного порта
  serv1.AttachServo(3, 90);     // Первый сервопривод подключен к 3 пину, начальное значение 90°
  serv2.AttachServo(5, 75);     // Второй сервопривод подключен к 5 пину, начальное значение 75°
  serv3.AttachServo(6, 90);     // Третий сервопривод подключен к 6 пину, начальное значение 90°
  serv4.AttachServo(9, 90);     // Четвертый сервопривод подключен к 9 пину, начальное значение 90°
  serv5.AttachServo(10, 90);     // Пятый сервопривод подключен к 10 пину, начальное значение 90°
  serv6.AttachServo(14, 130);    // Первый сервопривод подключен к 14 пину, начальное значение 130°
  serv7.AttachServo(15, 90);     // Второй сервопривод подключен к 15 пину, начальное значение 90°
  serv8.AttachServo(16, 90);     // Третий сервопривод подключен к 16 пину, начальное значение 90°
  serv9.AttachServo(17, 90);     // Четвертый сервопривод подключен к 17 пину, начальное значение 90°
  serv10.AttachServo(18, 90);     // Пятый сервопривод подключен к 18 пину, начальное значение 90°
}
```

Рисунок 22 – Настройка работы микроконтроллера

После определения режима работы микроконтроллера следует основной цикл программы.

```
106 void loop()
107 {
108   if (Serial.available() == 0)
109   {
110     s = 0;
111   }
112   while (Serial.available())
113   {
114     char sym = Serial.read();
115     Text[s] = sym;
116     s++;
117     delay(16);
118     unsigned long timing = millis();
119   }
```

Рисунок 23 – Обработка пришедших данных в порт UART

В начале основного цикла программы проверяется наличие байтов в последовательном порте МК, если порт пуст обнуляется индекс массива Text. Если поступили символы и пока они поступают в порт происходит запись символов в массив Text и обнуляется счетчик тайм-аута timing, между поступлениями символов удерживается пауза в 16 мс. Задержка необходима т.к. UART работает медленнее, чем CPU микроконтроллера (см.рис. 23).

Переходим к обработке поступившей команды от смартфона, первой обрабатывается команда проверки связи между МК и Смартфоном.

```
if ( millis() - timing > TIME_OUT )
{
  while (true)
  {
    if ( Text[0] != 'A') break; //Обработка команды проверки связи "AT\r\n"
    if ( Text[1] != 'T')
    {
      Serial.print("Bad\r\n");
      clearText();
      break;
    }
    if ( Text[2] == 13 )
    {
      if ( Text[3] != 10 ) break; // ошибка
      Serial.print("OK\r\n");
      clearText();
      break;
    }
```

Рисунок 24 – Обработка команды «AT»

Если время тайм-аута истекло и данные не поступают в порт, начинается проверка элементов массива Text на последовательность символов одной из набора команд (см. табл.2). Проверка начинается с нулевого элемента, если он не соответствует символу «A», значит пришла некорректная команда и выходит из подпрограммы обработки массива, иначе если совпадает проверяется первый элемент массива на соответствие символу «T», при несоответствии символа выводится в порт сообщение об ошибке «Bad» и выходим из подпрограммы обработки массива. Далее ведется проверка на наличие кода символа возврата

каретки «13», при несоответствии выходим из подпрограммы, по аналогии проверяется код символа перехода на новую строку «10». При успешной обработки команды в последовательный порт выводится сообщение «OK» и обнуляется массив Text, содержащий полученную команду, с помощью вызова функции clearText() (см. рис. 24).

Функция clearText() занимается очисткой массива в котором хранится полученная текстовая команда от смартфона, это необходимо для того, что корректно интерпретировать следующую полученную команду от смартфона (см. рис. 25).

```
void clearText()//Очистка текстового буфера
{
  for (int q = 0; q < s; q++)
  {
    Text[q] = 0;
  }
}
```

Рисунок 25 – Функция очистки текстового буфера

Если второй элемент массива Text не является символом возврата каретки – «13», то происходит сравнение с кодом символа «A». Согласно заданной системе команд после третьего символа команды, т.е. второго элемента массива, следует номера управляемых сервоприводов - «1» / «2» / «3» / «4» / «5». После этого начинается проверка четвертого элемента массива на наличие символа «=», в случае неудачной идентификации символа происходит выход из обработки пришедшей команды (см. рис. 26).

```
else if (Text[2] == 'A')
{
  if ((Text[3] == '1')||(Text[3] == '2')||(Text[3] == '3')||(Text[3] == '4')||(Text[3] == '5'))
  {
    if (Text[4] != '=') break;
```

Рисунок 26 – Обработка команды «ATA» - установка угла поворота вала сервопривода

После символа «=» следует последовательность символов соответствующая числу угла поворота вала сервопривода. Из массива Text, хранящем пришедшую команду переписывается последовательность символов в массив textBuf. В параллельном процессе переписывания угла поворота в массив производится проверка на наличие символа с кодом возврата каретки «13», при его обнаружении прекращается заполнение массива textBuf и проверяется наличие символа перехода на новую строку «10» при его обнаружении команда считается полной, иначе происходит выход из подпрограммы обработки пришедшей команды (см. рис. 27).

```
i = 0;
while (i < 6)
{
  textBuf[i] = Text[i + 5];
  if ( textBuf[i] == 13 ) break; // конец команды
  i++;
}
if ( (textBuf[i] != 13) || (i > 5) ) break; // ошибка
textBuf[i] = 0;
if ( Text[i + 6] != 10 ) break;
```

Рисунок 27 – Обработка значения угла поворота вала

После успешной обработки команды передачи значения угла поворота для сервопривода, проверяется содержание четвертого элемента массива, в соответствии с которым происходит управление с одним из 1-5 сервоприводами, путем передачи целочисленного значения угла поворота методу MoveTo() одному из объектов класса ServoClass. После выполнение команды в последовательный интерфейс связи выводится сообщение «OK» и обнуляется массив Text, содержащий полученную команду и происходит выход из обработки команды (см. рис. 28).

```
switch (Text[3])
{
  case '1':
    if (serv1.CheckPosition() != atoi(textBuf)) {
      serv1.MoveTo(atoi(textBuf), speed1);
    }
    break;
  case '2':
    if (serv2.CheckPosition() != atoi(textBuf)) {
      serv2.MoveTo(atoi(textBuf), speed2);
    }
    break;
  case '3':
    if (serv3.CheckPosition() != atoi(textBuf)) {
      serv3.MoveTo(atoi(textBuf), speed3);
    }
    break;
  case '4':
    if (serv4.CheckPosition() != atoi(textBuf)) {
      serv4.MoveTo(atoi(textBuf), speed4);
    }
    break;
  case '5':
    if (serv5.CheckPosition() != atoi(textBuf)) {
      serv5.MoveTo(atoi(textBuf), speed5);
    }
  }
  Serial.print("OK\r\n");
  clearText();
  break;
}
break;
```

Рисунок 28 – Передача угла поворота методу MoveTo()

Аналогично обработке команды «ATA» - задание угла поворота вала сервопривода, происходит обработка команды «ATS», если второй элемент массива Text не символом – «A», то происходит сравнение с кодом символа «S». Согласно заданной системе команд после третьего символа команды, т.е. второго элемента массива, следует номера управляемых сервоприводов - «1» / «2» / «3» / «4» / «5» / «6». После этого начинается проверка четвертого элемента массива на наличие символа «=», в случае неудачной идентификации символа происходит выход из обработки пришедшей команды (см. рис. 29).

```
else if (Text[2] == 'S')//Обработка команды установки скорости поворота вала привода "ATS1=00\r\n"
{
  if ((Text[3] == '1') || (Text[3] == '2') || (Text[3] == '3') || (Text[3] == '4') || (Text[3] == '5') || (Text[3] == '6'))
  {
    if (Text[4] != '=') break;
    i = 0;
    while (i < 6)
    {
      textBuf[i] = Text[i + 5];
      if ( textBuf[i] == 13 ) break; // конец команды
      i++;
    }
    if ( (textBuf[i] != 13) || (i > 5) ) break; // ошибка
    textBuf[i] = 0;
    if ( Text[i + 6] != 10 ) break;
```

Рисунок 29 – Обработка команды ATS - установка скорости поворота вала сервопривода

Далее после символа «=» следует последовательность символов соответствующая числу скорости вращения вала сервопривода. Из массива Text, хранящем пришедшую команду переписывается последовательность символов в массив textBuf. В параллельном процессе переписывания угла поворота в массив производится проверка на наличие символа с кодом возврата каретки «13», при его обнаружении прекращается заполнение массива textBuf и проверяется наличие символа перехода на новую строку «10» при его обнаружении команда считается полной, иначе происходит выход из подпрограммы обработки пришедшей команды.

После успешной обработки команды передачи скорости вращения вала сервопривода, проверяется содержание четвертого элемента массива, в соответствии с которым происходит сохранение целочисленного значения скорости в переменные, хранящие значения скоростей сервоприводов. После выполнение команды в последовательный интерфейс связи выводится сообщение «OK» и обнуляется массив Text, содержащий полученную команду и происходит выход из обработки команды (см. рис. 30).

```
    switch (Text[3])
    {
      case '1':
        speed1 = atoi(textBuf);
        break;
      case '2':
        speed2 = atoi(textBuf);
        break;
      case '3':
        speed3 = atoi(textBuf);
        break;
      case '4':
        speed4 = atoi(textBuf);
        break;
      case '5':
        speed5 = atoi(textBuf);
        break;
      case '6':
        speed6 = atoi(textBuf);
        break;
    }
    Serial.print("OK\r\n");
    clearText();
    break;
  }
  break;
}
```

Рисунок 30 – Сохранение численного значение скорости

При обнаружении соответствия второго элемента массива Text коду символа «R», то распознана команда получения текущего положения звеньев манипулятора. При получении данной команды микроконтроллер считывает текущие положения валов сервоприводов и отправляет ответ с информацией о положении манипулятора (см. рис. 31).

```
else if (Text[2] == 'R')//Получение текущего положения приводов "ATR\r\n"
{
  if ((Text[3] != 13) || (Text[4] != 10)) break;
  polojeniye = serv1.CheckPosition();
  Serial.print("drive1=");
  Serial.print(polojeniye);
  Serial.print(";");
  polojeniye = serv2.CheckPosition();
  Serial.print("drive2=");
  Serial.print(polojeniye);
  Serial.print(";");
  polojeniye = serv3.CheckPosition();
  Serial.print("drive3=");
  Serial.print(polojeniye);
  Serial.print(";");
  polojeniye = serv4.CheckPosition();
  Serial.print("drive4=");
  Serial.print(polojeniye);
  Serial.print(";");
  polojeniye = serv5.CheckPosition();
  Serial.print("drive5=");
  Serial.print(polojeniye);
  Serial.print(";\r\n");
  clearText();
  break;
```

Рисунок 31 – Вывод текущего положения манипулятора

При соответствии второго элемента массива Text коду символа «M», следовательно, распознана команда ручного управления. Третий символ означает соответствующее перемещение манипулятора: «L» - поворот манипулятора в левую сторону, «R» - поворот манипулятора в правую сторону, «U» - поднятие рабочего органа манипулятора, «D» - опускание рабочего органа манипулятора, «B» - вращение кисти, «F» - управление пальцами манипулятора, «S» - поворот плеча манипулятора.

Рассмотрим программный код выполняющий алгоритм движения звеньев в ручном режиме.

После успешной проверки второго элемента массива на соответствие коду символа «M», следует проверка третьего элемента массива на соответствие символам «L» или «R». При соответствии третьего элемента массива Text, т.е. четвертого символа команды коду символа «L» или «R», происходит считывание текущего положения вала сервопривода – serv1, который отвечает за поворот

манипуляционного устройства, и сохраняется в переменную j. Далее при «L» происходит инкрементирование переменной, т.е. увеличение угла поворота вала привода. Если же третий элемент Text соответствуют коду символа «R», происходит декрементирование переменной j, т.е. уменьшение угла поворота вала привода, и передача ее численного значения методу MoveTo, объекта serv1 класса ServoClass (см. рис.32).

При соответствие третьего элемента массива символам «U» или «D», следует следующий алгоритм действий. Происходит запись текущих положений сервоприводов сгибания локтя – serv3 и сгибания кисти – serv4 в переменные j и f. Далее при поднятии манипулятора (третий элемент массива – «U») происходит инкрементирование переменных и проверка на принятие новой команды или чтобы значение угла поворота вала сервопривода установленного на сгибании локтя не достигло значения, при опускании манипулятора (третий элемент массива – «D») происходит декрементирование переменных и проверка на принятие новой команды или чтобы значение угла поворота вала сервопривода установленного на сгибании локтя не достигло значения 30 и передача численных значений методу MoveTo, объектов serv3 и serv4 класса ServoClass (см. рис. 32).

```
else if (Text[2] == 'M')
{
  if ((Text[3] == 'L') || (Text[3] == 'R'))
  {
    j = serv1.CheckPosition();
    if(Text[3]=='L')
    {
      j++;
      if (Serial.available() > 0) break;
    }
    else if(Text[3]=='R')
    {
      j--;
      if (Serial.available() > 0) break;
    }
      serv1.MoveTo(j, speed1);
    break;
  }
  if ((Text[3] == 'U') || (Text[3] == 'D'))
  {
    j = serv3.CheckPosition();
    f = serv4.CheckPosition();
    if(Text[3]=='U')
    {
      if ((Serial.available() > 0)||(j>=150)) break;
      j++;
      f++;
    }
    else if(Text[3]=='D')
    {
      if ((Serial.available() > 0)||(j<=30)) break;
      j--;
      f--;
    }
    serv3.MoveTo(j, speed3);
    serv4.MoveTo(f, speed4);
    break;
  }
```

Рисунок 32 – Обработка команд ручного управления «ATMU», «ATMD», «ATML», «ATMR»

Если третий элемент массива Text – «B», то происходит дальнейшая проверка 4 элемента массива на соответствие символам «1» или «2». Далее происходит считывание положение сервопривода установленного на кисти манипулятора и запись численного значения в переменную z, при символе «1» происходит декрементирование переменной, при «2» инкрементирование и передача ее численного значения методу MoveTo, объекта serv5 класса ServoClass (см. рис. 33).

```
if (Text[3] == 'B') //Brush - кисть
{
  z = serv5.CheckPosition();
  if (Text[4] == '1') //Вращение против часовой стрелки
  {
    z--;
    if (Serial.available() > 0) break;
  }
  if (Text[4] == '2') //Вращение по часовой стрелке
  {
    z++;
    if (Serial.available() > 0) break;
  }
  serv5.MoveTo(z, speed5);
  break;
}
```

Рисунок 33 – Обработка команд поворота кисти «ATMB1», «ATMB2»

При третьем элементе массива Text – «F» считываются положения сервоприводов приводящих в движение пальцы манипулятора в переменные: fig_1, fig_2, fig_3, fig_4, fig_5. Далее продолжается проверка четвертого элемента на соответствие символам «1» или «2». При «1» происходит декрементирование переменных fig_1, fig_2, fig_3, fig_4, fig_5, при «2» происходит инкрементирование этих переменных. После этого происходит передача численных значений переменных fig_1, fig_2, fig_3, fig_4, fig_5 методу MoveTo, объектов serv6, serv7, serv8, serv9, serv10 класса ServoClass (см. рис. 34).

```
if (Text[3] == 'F') // Fingers - пальцы
{
  fig_1 = constrain(fig_1, 25, 130);
  fig_1 = serv6.CheckPosition();
  fig_2 = serv7.CheckPosition();
  fig_3 = serv8.CheckPosition();
  fig_4 = serv9.CheckPosition();
  fig_5 = serv10.CheckPosition();
  if (Text[4] == '1') //Сжатие
  {
    if ((Serial.available() > 0) || (fig_1 <= 25) || (fig_2 <= 20) || (fig_3 <= 20) || (fig_4 <= 20) || (fig_5 <= 20)) break;
    fig_1--;
    fig_2--;
    fig_3--;
    fig_4--;
    fig_5--;
  }
  if (Text[4] == '2') //Расжатие
  {
    if ((Serial.available() > 0) || ((fig_1 >= 130) && (fig_2 >= 100) && (fig_3 >= 100) && (fig_4 >= 100) && (fig_5 >= 100))) break;
    fig_1++;
    fig_2++;
    fig_3++;
    fig_4++;
    fig_5++;
  }
    serv6.MoveTo(fig_1, speed6);
    serv7.MoveTo(fig_2, speed6);
    serv8.MoveTo(fig_3, speed6);
    serv9.MoveTo(fig_4, speed6);
    serv10.MoveTo(fig_5, speed6);
  break;
}
```

Рисунок 34 – Обработка команд управления пальцами кисти «ATMF1» и «ATMF2»

Если же третий элемент массива Text – «S», то считываются численные значения угла поворота валов сервоприводов установленных на плече – serv2 и локте – serv3 манипулятора в переменные j и f соответственно. После этого следует обработка четвертого элемента массива Text, если он содержит «1», то численные значения переменных j и f декрементируются, если «2» инкрементируются. В конце цикла обработки команды численные значения переменных j и f передаются методам MoveTo, объектов serv2 и serv3 класса ServoClass (см. рис. 35).

```
    if (Text[3] == 'S') //Shoulder - плечо
    {
      j = serv2.CheckPosition();
      f = serv3.CheckPosition();
      if (Text[4] == '1') //Вытягивание руки
      {
        if ((Serial.available() > 0) || (j <= 25)) break;
        j--;
        f--;
      }
      if (Text[4] == '2')
      {
        if ((Serial.available() > 0) || (f >= 170)) break;
        j++;
        f++;
      }
        serv2.MoveTo(j, speed2);
        serv3.MoveTo(f, speed3);
      break;
    }
    for (int q = 0; q < s; q++)
    {
      Text[q] = 0;
    }
    break;
  }
  break;
}
```

Рисунок 35 – Обработка команд управления плечом «ATMS1» и «ATMS2»

3.2 Программная реализация алгоритма работы пользовательского по на смартфоне (ОС Android)

3.2.1 Описание системы

Первоначально пользователю предоставляется экран, на котором он может выполнить следующие действия, а именно включение/выключение сети Bluetooth на устройстве. Управление состоянием Bluetooth выполняется с помощью кнопок «Включить», «Выключить». Для поиска активных Bluetooth устройств нужно нажать кнопку «Поиск», после поиска все найденные устройства выводятся в виде списка на экран пользователя. С помощью кнопки «Сопряженные устройства» выводятся ранее сопряженные устройства (см. рис. 36).

Рисунок 36 – Начальное окно пользовательского интерфейса

После выбора устройства и подключения к нему пользователю предоставляется возможность перейти в режим «Настройки» с помощью списка элементов меню (см. рис. 37).

Рисунок 37 – Выбор из списка элемента меню

Для начала работы системы управления манипуляционным устройством необходимо выполнить настройку скорости перемещений звеньев, это создано для безопасности оператора, т.к. перемещение звеньев на большой скорости может нанести вред здоровью оператора.

Меню настроек позволяет задать скорость вращения валов установленных на сочленении звеньев сервоприводов, для каждого звена скорость задается отдельно. С помощью кнопки «Установить» передаются введенные значения на микроконтроллер. При нажатии на кнопку «Сброс», выполняется очистка введённых значений пользователем, с помощью кнопки «Меню» происходит переход на начальный экран (см. рис. 38).

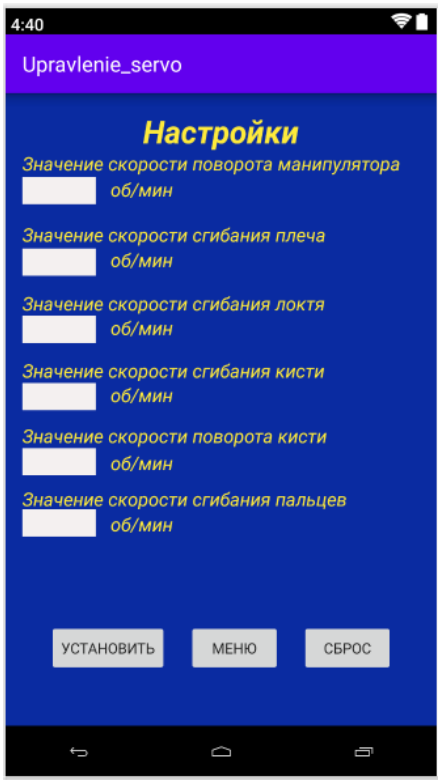

Рисунок 38 – Режим настроек

Для того чтобы пользователь не смог ввести некорректные данные, при настройки элементов editText указывается атрибут android:inputType=number [39].

При работе в ручном режиме пользователю предоставляется окно в с управляющими кнопками для управления изменения положения манипулятора. При удержании кнопок выполняется соответствующее движение устройства при нажатии на кнопку «Меню» происходит переход на начальный экран (см. рис. 39).

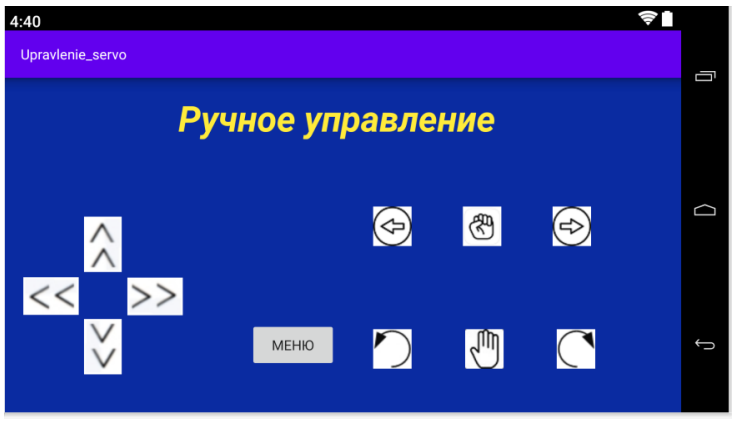

Рисунок 39 – Режим ручного управления

Для управления манипуляционным устройством в позиционном управлении пользователю предоставляется окно, в котором происходит задание занимаемого положения манипулятора в элементах editText аналогично меню настроек указывается атрибут android:inputType= number. При нажатии кнопки «Установить» передаются введенные значения на микроконтроллер. При нажатии на кнопку «Сброс», выполняется очистка введённых значений пользователем. Кнопа «Проверка» предназначена для вывода текущего положения занимаемого манипулятором. Также для перехода в меню используется кнопка «Меню» (см. рис. 40).

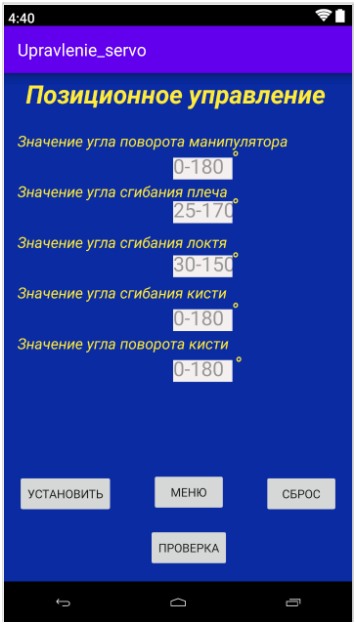

Рисунок 40 – Режим позиционного управления

3.2.2 Описание программной реализации продукта

Для работы с Bluetooth модулем смартфона создается отдельный класс, в котором происходит инициализация Bluetooth адаптера (Bluetooth-приёмник), получение сокета для подключения устройств, также реализованы функции для подключения к устройству с заданным MAC адресом и отключения устройства, на рисунке 41 показан класс BluetoothManager.

```
public class BluetoothManager {
    private BluetoothAdapter adapter;
    private BluetoothSocket socket;

    private static BluetoothManager instance;

    private BluetoothManager() {
        adapter = BluetoothAdapter.getDefaultAdapter();
        instance = this;
    }
    public BluetoothAdapter getAdapter() { return this.adapter; }
    public static BluetoothManager getInstance() {
        if(BluetoothManager.instance == null) {
            BluetoothManager.instance = new BluetoothManager();
        }
        return BluetoothManager.instance;
    }
    public BluetoothSocket getActiveSocket() { return this.socket; }

    public void connect(String address) {
        close();
        BluetoothDevice device = getAdapter().getRemoteDevice(address);
        try {
            socket = device.createRfcommSocketToServiceRecord(MY_UUID);
            socket.connect();
        } catch (IOException e) {
            close();
        }
    }
    public void close() {
        if (socket != null) {
            try {
                socket.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
            socket = null;
        }
```

Рисунок 41 – Класс BluetoothManager

При выборе устройства из списка (см. рис. 36) происходит передача MAC адреса устройства методу connect() класса BluetoothManager (см. рис .42) [40]. Выполнение кода выполняется по событию setOnItemClickListener() списка [41].

```
address = info.substring(info.length()-17);
BluetoothManager.getInstance().connect(address);
```

Рисунок 42 – Подключение к выбранному устройству

Установка соединения с выбранным объектом Bluetooth в окнах режимов управления осуществляется следующим способом (см. рис. 43) [42].

```
mConnectedThread = new ConnectedThread(BluetoothManager.getInstance().getActiveSocket());
mConnectedThread.start();
```

Рисунок 43 – Работа с текущим устройством Bluetooth

Переход между элементами меню осуществляется с помощью метода intent() (см. рис. 44) [43].

```
public boolean onOptionsItemSelected(MenuItem item) {
    int id = item.getItemId();
    if(id==R.id.settings)
    {
        Intent i = new Intent( packageContext: MainActivity.this, Settings.class);
        startActivity(i);
    }
    else if(id==R.id.manual)
    {
        Intent i = new Intent( packageContext: MainActivity.this, Manual_Control.class);
        startActivity(i);
    }
    else if(id==R.id.position)
    {
        Intent i = new Intent( packageContext: MainActivity.this, Pozicionnoye_upravleniye.class);
        startActivity(i);
    }
    return super.onOptionsItemSelected(item);
}
```

Рисунок 44 – Переключение между элементами меню

Получение данных от Arduino происходит следующим образом, после того как прочитан очередной блок данных из входящего потока он посылается в главную деятельность посредствам вызова метода Handler. В котором происходит получение строки, определение конца строки, извлечение текста и вывод на экран пользователя с помощью Toast.makeText() (см. рис. 45) [44].

```
h = (Handler) handleMessage(msg) → {
        switch (msg.what) {
            case RECIEVE_MESSAGE:                                    // если приняли сообщение в Handler
                byte[] readBuf = (byte[]) msg.obj;
                String strIncom = new String(readBuf,  offset: 0, msg.arg1);
                sb.append(strIncom);                                 // формируем строку
                int endOfLineIndex = sb.indexOf("\r\n");             // определяем символы конца строки
                if (endOfLineIndex > 0) {                            // если встречаем конец строки,
                    String sbprint = sb.substring(0, endOfLineIndex);// то извлекаем строку
                    sb.delete(0, sb.length());                       // и очищаем sb
                    Toast.makeText(getBaseContext(),  text: "Ответ от Arduino: " + sbprint, Toast.LENGTH_SHORT).show();
                    set.setEnabled(true);
                    reset.setEnabled(true);
                    menu.setEnabled(true);
                }
                break;
        }
    };
```

Рисунок 45 – Обработка входящих сообщений

Отправка управляющих команд происходит с помощью метода write() (см. рис. 46).

```
mConnectedThread.write("ATR\r\n");
public void write(String message) {
    byte[] msgBuffer = message.getBytes();
    try {
        mmOutStream.write(msgBuffer);
    } catch (IOException e) {
        Log.d(TAG,  msg: "...Ошибка отправки данных: " + e.getMessage() + "...");
    }
}
```

Рисунок 46 – Оправка команды микроконтроллеру

Таким образом, выше показана концепция программы для управления манипуляционным устройством. В данном ПО взаимодействие с подключенным устройством Bluetooth реализуется через отдельно созданный java класс который поддерживает связь на протяжении работы Bluetooth.

3.3 Практическое тестирование программного продукта

Тестирование — это проверка созданного программного продукта на соответствие заданным требованиям, и на отсутствие дефектов.

Тестирование подразумевает запуск программы (или ее компонентов) при помощи ручных (или автоматизированных) программных средств тестирования; тестироваться могут и отдельные компоненты (или отдельные свойства) программы.

Тестирование производится для поиска ошибок, случайных «пропусков» по невнимательности, либо направлено на соблюдение прописанных требований к софту [45].

3.3.1 Тестирование программного кода управляющей программы Arduino

Работа управляющей программы на микроконтроллере Arduino в обработке полученных команд от смартфона, поэтому тестирование программного продукта выполняется в встроенной терминальной утилите Arduino IDE.

Монитор порта Arduino – это утилита, встроенная в среду программирования Arduino IDE и служит она для связи компьютера с контроллером. С помощью монитора последовательного порта производится отладка прошивки микроконтроллера, получение информации о работе программы и отправка команд к микроконтроллеру по USB. (см. рис. 47) [46].

Рисунок 47 – Монитор порта Arduino

Управляющая программа на микроконтроллере выполняет алгоритм управления согласно полученным командам, полученным через COM порт. В качестве тестирования работоспособности программы на микроконтроллере, зададим в микроконтроллер несколько команд текстового протокола и получим на них соответствующие ответы.

В качестве первого теста программы введем команду проверки связи «AT», согласно описанию Команда представляет последовательную запись символов («AT», 13, 10), в терминальной программе символы возврата каретки (13) и переход на новую строку (10) подставляются автоматически. Ответом на данную команду является последовательность символов («OK», 13, 10) (см. рис. 35).

Рисунок 35 –Тестирование команды «AT»

Как видим из рис. 35 при отправке команды проверки связи мы получаем ответ «OK», который характеризует успешную связь и обмен данными устройства с микроконтроллером.

Протестируем корректность выполнения команд установки угла поворота валов сервоприводов установленных на сочленении манипулятора.

Установим угол поворота манипулятора равным 45°, т.е. установим вал первого сервопривода в 45°, далее установим сгибание плеча равным 50°, т.е. вал второго сервопривода, сгибание в локте 80°, т.е. вал третьего сервопривода, поворот кисти 135°, т.е. вал четвертого сервопривода, поворот кисти установим 60°, т.е. вал пятого сервопривода. Для утсновки угла поворота используется команды «ATA1» … «ATA5».

Для получения текущего положения манипулятора введем команду «ATR», результат выполнения команды на рисунке ниже (см. рис. 36). Тем самым видим успешно выполненные команды позиционирования манипулятора «ATA1» … «ATA5».

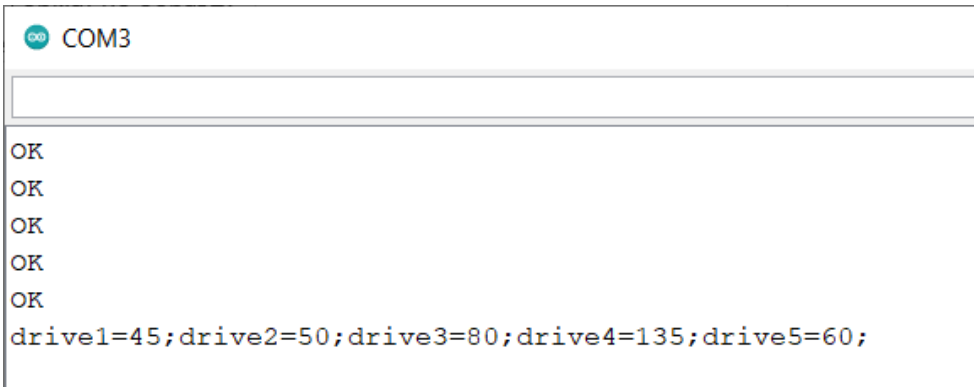

Рисунок 48 – Тестирование команд «ATA1» … «ATA5», «ATR»

Размер скомпилированной программы Arduino на рисунке ниже (см. рис. 49).

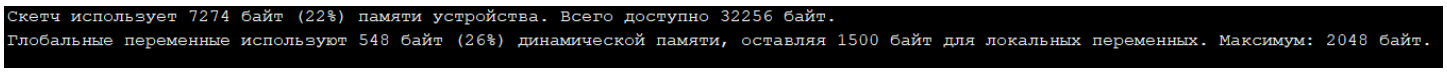

Рисунок 49 – Размер программы Arduino

3.3.2 Тестирование программного кода пользовательского по на смартфоне (ОС Android)

Тестирование программного продукта происходит на реальном устройстве с ОС Android. Первоначально необходимо установить приложение на устройство, название приложения «Upravlenie_servo» и имеет следующую иконку (см. рис. 50).

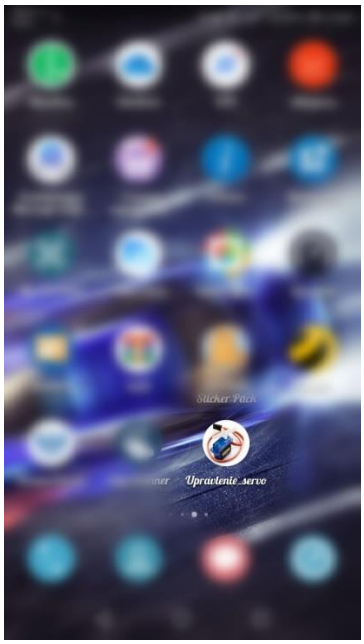

Рисунок 50 – Внешний вид установленного приложения

Запустив приложение пользователь попадает на главный экран, где ему необходимо запустить Bluetooth, система Android запросит разрешение на использование Bluetooth в нашем приложении (см. рис. 51).

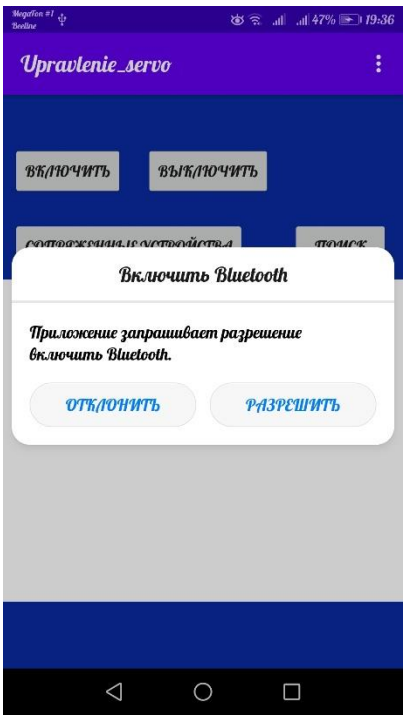

Рисунок 51 – Запрос на разрешение использования Bluetooth

После запуска Bluetooth, для сопряжения с устройством, необходимо выполнить поиск доступных устройств (см. рис. 52) или вывод на экран ранее сопряженных устройств (см. рис. 53).

Рисунок 52 – Вывод доступных Bluetooth устройств

Рисунок 53 – Вывод ранее сопряженных устройств

В начале работы системы пользователю нужно перейти в меню «Настройки» нажав кнопку «три точки», откроется меню (см. рис. 54).

Рисунок 54 – Переход между элементами меню

После введенных данных в меню «Настройки» нужно выполнить отправку значений в микроконтроллер, нажав на кнопку «Установить» предварительно выполняется проверка на корректность заполненных полей (см. рис. 55). В ответ от микроконтроллера на экран пользователя выводится сообщение об успешном получении команды в виде текста «OK» (см. рис. 56).

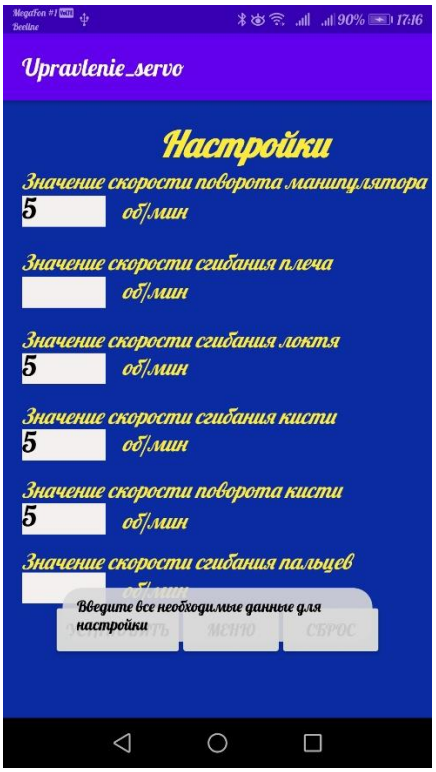

Рисунок 55 – Проверка введенных данных на корректность

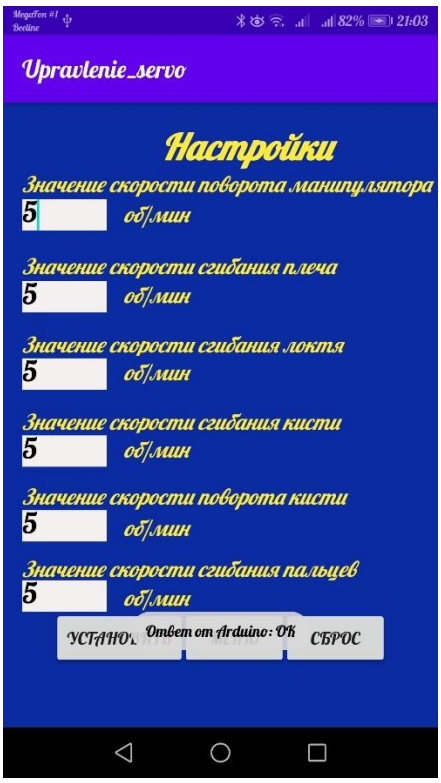

Рисунок 56 – Работа меню настроек

В меню ручного управления на смартфоне пользователя располагается набор управляющих кнопок. В данном режиме управления нет отображения обратной связи от микроконтроллера, возможно лишь визуально наблюдать за перемещением манипулятора (см. рис. 57).

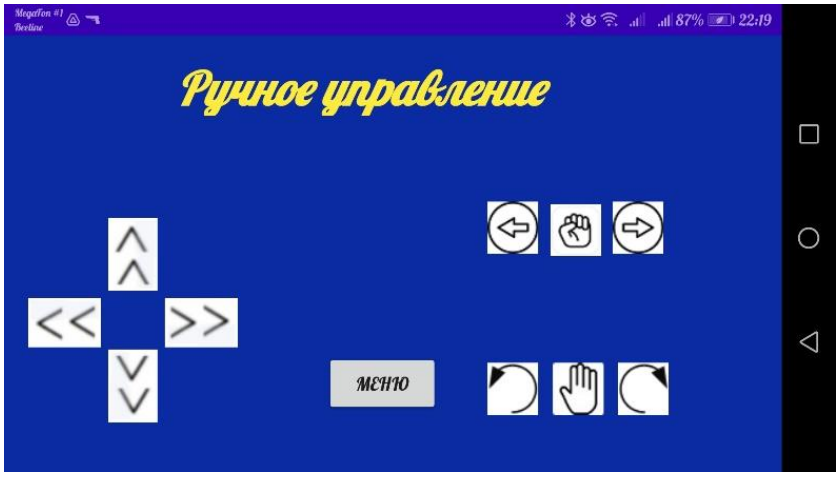

Рисунок 57 – Ручное управление

В меню позиционного управления пользователь может проверить текущее положение манипулятора, нажатием кнопки «Проверка» и задать занимаемое положение, нажав кнопку «Задать». Получение и задача положения происходит через поля ввода текста (см. рис. 58). По аналогии с режимом «Настройки» перед отправкой данных в микроконтроллер проверяется корректность введенных данных (см. рис. 59).

Рисунок 58 – Позиционное управление

Рисунок 59 – Проверка введенных данных на корректность

Цель проведения испытаний состояла в том, чтобы рассмотреть все возможные варианты работы программы, протестировать ее в нормальных условиях, выявить недостатки и устранить их, если таковые имели место.

В результате испытаний на контрольных примерах было доказано, что данная программа работает согласно заданному алгоритму. Тестирование показало, что разработанный программный продукт работоспособен.

ЗАКЛЮЧЕНИЕ

В ходе выполнения научно – исследовательской работы была разработана система управления автоморфным манипуляционным устройством. Авторская разработка заключается в предложении системы управления с использованием Bluetooth связи для взаимодействия смартфона Android и микроконтроллерной платформой Arduino.

Для обмена данными между Arduino и смартфоном пользователя был разработан протокол обмена данными, включающий в себя следующие функции управления манипуляционным устройством:

– поворот вала сервоприводов установленных на сочленении звеньев на заданный угол;

– управление скоростью поворота валов сервоприводов;

– управление манипуляционным устройством в ручном режиме;

– получение текущего положения манипулятора.

Для управления манипуляционным устройством разработано пользовательское приложение, которое предусматривает управление работы Bluetooth на смартфоне оператора, настройки скорости перемещений звеньев манипулятора и несколько режимов управления.

Охарактеризована предметная область исследования, в которой указаны сущность робототехники как прикладной науки, представлено описание используемого манипуляционного устройства и ЭВМ. Проведен анализ существующих методов управления манипуляционными устройствами, перечислены преимущества и недостатки каждого из них. Выполнен обзор возможностей профильного ПО разработки мобильного приложения – Android Studio и разработки управляющей программы на Arduino – Arduino IDE.

Научная новизна работы определяется разработкой и реализацией новых подходов к решению проблемы управления манипуляционным устройством и заключается в следующем: Предложение структуры и принципа построения

системы управления манипуляционным устройством, основанной на применение микроконтроллера и мобильного устройства - смартфона. Позволяющие эффективно решать задачи траектории и управления электроприводами.

Практическую значимость будут иметь следующие результаты работы:

– аппаратная реализация, алгоритмы и программное обеспечение системы управления манипуляционным устройством с использованием микроконтроллера и смартфона;

– структура программных средств системы управления манипуляционным устройством, и способы распределения вычислительных ресурсов между МК и смартфоном;

– имеющиеся текстовый протокол с помощью команд которого поддерживается связь на расстоянии между МК и смартфона посредством Bluetooth.

БИБЛИОГРАФИЧЕСКИЕ ССЫЛКИ

1 myrobot.ru: Что такое робот? [Электронный ресурс]: myrobot.ru. Режим доступа: https://myrobot.ru/wiki/index.php?n=Articles.Robot (Дата обращения - 07.02.2022);

2 Варков А.А. Разработка и исследование системы управления манипуляционным промышленным роботом на базе контроллера движения: дис., на соискание ученой степени канд. техн. наук: 05.09.03. – Ивановский государственный энергетический университет, Иваново, 2015 - 146 с.;

3 Чернухин Ю.В. «Введение в роботехнику: Учебное пособие» - Таганрог: ТРТИ, 1990 - с.46;

4 Попов Е.П., Письменный Г.В. Основы робототехники: Введение в специальность М.: Высшая школа, 1990. 224 с.;

5 Юревич Е.И. Основы робототехники / Е.И. Юревич. –2-е изд., перераб. и доп. –СПб.: БХВ-Петербург, 2005. –416 с.;

6 Промышленные роботы. Конструирование и применение: Учеб. пособие. – 2-е изд., перераб. и доп.–К.: Высшая шк., 1991 – 311 с.: ил.;

7 Никитина Т.В. Образовательная робототехника как направление инженерно-технического творчества школьников [Текст]: учебное пособие / Т.В. Никитина. – Челябинск: Изд-во Челяб. гос. пед. ун-та, 2014. – 169 с.;

8 Введение в мехатронику: Уч. пособие / Грабченко А.И., Клепиков В.Б., Доброскок В.Л., Крыжный Г.К., Анищенко Н.В., Кутовой Ю.Н., Пшеничников Д.А., Гаращенко Я.Н. – Х.: НТУ «ХПИ», 2014 – 274 с.;

9 www.4ne.ru: Манипуляторы. [Электронный ресурс]: 4ne.ru. Режим доступа: http://www.4ne.ru/stati/robotetxnika/manipulyatory.html (Дата обращения - 07.02.2022);

10 top3dshop.ru: Роботы-манипуляторы, что они могут и где применяются. [Электронный ресурс]: Top 3D Shop. Режим доступа: https://top3dshop.ru/blog/manipulator-robots-features-and-applications.html (Дата обращения - 07.02.2022);

11 smartep.ru: Обзор программы RoboLab для программирования NXT роботов. [Электронный ресурс]: smartep.ru. Режим доступа: http://smartep.ru/lego/other/Robolab%20for%20NXT.pdf (Дата обращения – 04.07.2021);

12 TRICK STUDIO: TRICK STUDIO. [Электронный ресурс]: trikset.com. Режим доступа: https://trikset.com/products/trik-studio (Дата обращения – 09.02.2022);

13 hmong.press: Студия разработчиков робототехники Microsoft. [Электронный ресурс]: www.hmong.press Режим доступа: https://www.hmong.press/wiki/Microsoft_Robotics_Developer_Studio (Дата обращения – 07.02.2022);

14 Сайт «Робототехника»: Среды управления роботами. [Электронный ресурс]: www.sites.google.com Режим доступа: https://www.sites.google.com/site/sajtprepodovatela/sredy-upravlenia-robotami (Дата обращения – 07.02.2022);

15 Arduino.ru: Среда разработки Arduino. [Электронный ресурс]: arduino.ru Режим доступа: http://arduino.ru/Arduino_environment (Дата обращения – 07.02.2022);

16 docs.voltbro.ru: Что такое ROS. [Электронный ресурс]: docs.voltbro.ru Режим доступа: http://docs.voltbro.ru/starting-ros/ros-about.html (Дата обращения – 07.02.2022);

17 dudom: C ide for android. [Электронный ресурс]: dudom.ru Режим доступа: https://dudom.ru/kompjutery/c-ide-for-android/ (Дата обращения – 07.02.2022);

18 skolkocom: Python на Android — 10 лучших приложений для программирования. [Электронный ресурс]: skolkocom.ru Режим доступа: https://skolkocom.ru/info/python-na-android-10-luchshih-prilozhenij-dlya-programmirovaniya (Дата обращения – 07.02.2022);

19 AndroidAPP: Топ – 3 среды разработки для Android. [Электронный ресурс]: app-android.ru Режим доступа: https://app-android.ru/blog/environment-develop-android (Дата обращения – 07.02.2022);

20 Проекты с использованием контроллера Arduino. – 2-е изд., перераб. и доп. – СПб.: БХВ-Петербург, 2015.- 464 с.: ил. – (Электроника);

21 Arduino.ru: Что такое Ардуино? [Электронный ресурс]: Аппаратная платформа Arduino | Arduino.ru Режим доступа: http://arduino.ru/About (Дата обращения-07.03.2022);

22 LAZY SMART: Разбираемся с понятиями «Интерфейс» и «Протокол» [Электронный ресурс]: lazysmart.ru Режим доступа: http://lazysmart.ru/osnovy-avtomatiki/razbiraemsya-s-ponyatiyami-interfejs-i/ (Дата обращения-13.03.2022);

23 autoit-script.ru: Коды ASCII символов [Электронный ресурс]: autoit-script.ru Режим доступа: https://autoit-script.ru/docs/appendix/ascii.htm (Дата обращения-13.03.2022);

24 MobiLab.ru: Использование Bluetooth в Android [Электронный ресурс]: Мобильные технологии | MobiLab.ru Режим доступа: http://www.mobilab.ru/androiddev/bluetoothinandroid.html (Дата обращения-13.03.2022);

25 Android. Программирование для профессионалов. 2-е изд. — СПб.: Питер, 2016. — 640 с.: ил. — (Серия «Для профессионалов»);

26 rAndroid: Android Studio для Windows: что это за программа [Электронный ресурс]: randroid.ru Режим доступа: https://randroid.ru/dev/android-studio-dlya-windows-chto-eto-za-programma (Дата обращения-13.03.2022);

27 Много толка: Андроид студио системные требования [Электронный ресурс]: mnogotolka.ru Режим доступа: https://mnogotolka.ru/info/android-studio-sistemnye-trebovanija/ (Дата обращения-13.03.2022);

28 Androfon.ru: Что такое Android studio [Электронный ресурс]: Androfon.ru Режим доступа: https://androfon.ru/prodvinut/chto-takoe-android-studio-gde-skachat-i-kak-ustanovit (Дата обращения-13.03.2022);

29 Arduino.ru: Среда разработки Arduino. [Электронный ресурс]: arduino.ru Режим доступа: http://arduino.ru/Arduino_environment (Дата обращения – 13.03.2022);

30 robotraffik.ru: Знакомство со средой Arduino IDE [Электронный ресурс]: Робототраффик | robotraffik.ru Режим доступа: https://robotraffik.ru/page/roborace-arduino-ide/ (Дата обращения – 13.03.2022);

31 rc-today.ru: Сервомашинка цифровая с маркировкой Tiankong RC Professional Servo - TD-8135MG [Электронный ресурс]: RC-TOFAY.RU Режим доступа: https://rc-today.ru/product/servomashinka-tsifrovaya-s-markirovkoi-tiankong-rc-professional-servo-td-8135mg/ (Дата обращения – 13.03.2022);

32 www.chipdip.ru: SG90 analog servo, Сервомотор аналоговый 1.8кг.см 180° [Электронный ресурс]: CHIPDIP.RU Режим доступа: https://www.chipdip.ru/product/sg90-analog-servo (Дата обращения – 13.03.2022);

33 3d-diy.ru: Обзор модуля Bluetooth HC – 05 [Электронный ресурс]: 3DiY | 3d-diy.ru Режим доступа: https://3d-diy.ru/wiki/arduino-moduli/bluetooth-modul-hc-05/#Techno (Дата обращения – 13.03.2022);

34 Сайт «Робототехника»: Среды управления роботами. [Электронный ресурс]: www.sites.google.com Режим доступа: https://www.sites.google.com/site/sajtprepodovatela/sredy-upravlenia-robotami (Дата обращения – 07.02.2022);

35 cleverdiy.ru: Описание Arduino UNO [Электронный ресурс]: CleverDIY | cleverdiy.ru. Режим доступа: https://cleverdiy.ru/opisanie-arduino-uno#i-2 (Дата обращения – 13.03.2022);

36 Петин В. А. Проекты с использованием контроллера Arduino. — СПб.: БХВ-Петербург, 2014. — 400 с.;

37 arduino-ide.com: Arduino IDE [Электронный ресурс]: arduino-ide.com Режим доступа: https://arduino-ide.com/ (Дата обращения – 13.03.2022);

38 ENVATO TUTS+: Начало работы с Android Studio [Электронный ресурс]: ENVATO TUTS+ | code.tutsplus.com Режим доступа:

https://code.tutsplus.com/ru/tutorials/getting-started-with-android-studio--mobile-22958 (Дата обращения – 13.03.2022);

39 betacode: Руководство Android EditText [Электронный ресурс]: betacode.net Режим доступа: https://betacode.net/12657/android-edittext (Дата обращения – 23.12.2021);

40 developers: Подключение устройств Bluetooth [Электронный ресурс]: developer.android.com Режим доступа: https://developer.android.com/guide/topics/connectivity/bluetooth/connect-bluetooth-devices (Дата обращения – 23.12.2021);

41 AndroidBugFix: java.lang.ClassCastException: android.widget.LinearLayout cannot be cast to android.widget.TextView [Электронный ресурс]: www.androidbugfix.com Режим доступа: https://www.androidbugfix.com/2022/03/javalangclasscastexception.html (Дата обращения – 23.12.2021);

42 Руководства по API: Bluetooth [Электронный ресурс]: android-doc.github.io Режим доступа: https://android-doc.github.io/guide/topics/connectivity/bluetooth.html (Дата обращения – 23.12.2021);

43 METANIT.COM: Создание меню [Электронный ресурс]: metanit.com. Режим доступа: https://metanit.com/java/android/7.1.php (Дата обращения – 23.12.2021);

44 web-shpargalka: Android studio работа с bluetooth [Электронный ресурс]: ВЭБ – ШПАРГАЛКА.РУ. Режим доступа: https://top3dshop.ru/blog/manipulator-robots-features-and-applications.html (Дата обращения – 23.12.2021);

45 testengineer.ru: Тестирование. Что это такое, описание, виды тестирования [Электронный ресурс]: testengineer.ru. Режим доступа: https://testengineer.ru/testirovanie-chto-ehto-takoe/ (Дата обращения – 23.12.2021);

46 роботехника18.рф: Монитор порта ардуино команды [Электронный ресурс]: роботехника18.рф. Режим доступа: https://роботехника18.рф/монитор-порта-ардуино/ (Дата обращения – 25.04.2022).

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1 А. Баканов, А. Обознов. Эргономика пользовательского интерфейса. От проектирования к моделированию человеко-компьютерного взаимодействия. – М.: 2011. – 176 с.;

2 Белов А. В. Программирование ARDUINO. Создаем практические устройства + виртуальный диск. – СПБ.: Наука и техника, 2018 – 272 с., ил.;

3 Белов А.В. Самоучитель разработчика устройств на микроконтроллерах AVR. – СПб.: Наука и Техника, 2008 – 544 с.: ил. +CD;

4 Блум Джереми Изучаем Arduino: инструменты и методы технического волшебства: Пер. с англ. - СПб.: БХВ-Петербург, 2015. - 336 с.: ил.;

5 Борисенко, Л.А. Теория механизмов, машин и манипуляторов: учеб. пособие/Л.А. Борисенко – Минск: Новое знание; М.: ИНФРА-М, 2011.-285 с.: ил. – (Высшее образование);

6 Варков А.А. Разработка и исследование системы управления манипуляционным промышленным роботом на базе контроллера движения: дис., на соискание ученой степени канд. техн. наук: 05.09.03. – Ивановский государственный энергетический университет, Иваново, 2015 - 146 с.;

7 Введение в мехатронику: Уч. пособие / Грабченко А.И., Клепиков В.Б., Доброскок В.Л., Крыжный Г.К., Анищенко Н.В., Кутовой Ю.Н., Пшеничников Д.А., Гаращенко Я.Н. – Х.: НТУ «ХПИ», 2014 – 274 с.;

8 Гриффитс Дэвид, Гриффитс Дон Head First. Программирование для Android. 2-е изд. – СПб.: Питер, 2018. –912 с.: ил. – (Серия «Head First O’Reilly»);

9 Дарвин, Ян Ф. Android. Сборник рецептов: задачи и решения для разработчиков приложений, 2-е изд.: Пер. с англ. – СПб.: ООО «Альфа-книга», 2018. – 768 с.: ил. – Парал. тит. англ.;

10 Дейтел П., Дейтел Х., Уолд А. Android для разработчиков. 3-е изд. – СПб.: Питер, 2016. – 512 с.: ил. – (Серия «Библиотека программиста»);

11 Джон Хофман Освоение Arduino: Проектный подход к электронике, схемам и программированию: Пер. с англ. - СПб.: БХВ-Петербург, 2018. - 302 с.: ил.;

12 Егоров О. Д. Конструирование механизмов роботов. Учебник/О.Д. Егоров – М.: Абрис, 2012-444 с.: ил.;

13 Клифтон Я. Проектирование пользовательского интерфейса в Android/пер. с англ. А. Н. Киселева. 2-е изд. – М.: ДМК Пресс, 2017 – 452 с.: ил.;

14 Козырев Ю.Г. 'Промышленные роботы. Справочник. 2-е издание, переработанное и дополненное' - Москва: Машиностроение, 1988 - с.392;

15 Конструируем роботов на Arduino. Первые шаги / Дж. Бейктал пер. с англ. О. А. Трефиловой. — М. : Лаборатория знаний, 2016. – 320 с. : ил.;

16 Корендясев А. И. Теоретические основы робототехники. В 2 кн./А.И. Корендясев, Б.Л. Саламандра, Л.И. Тывес; отв. ред. С.М. Каплунов; Ин-т машиноведения им. А.А. Благонравова РАН. – М.: Наука, 2006;

17 Куликов, С. С. Тестирование программного обеспечения. Базовый курс / С. С. Куликов. – 3-е изд. – Минск: Четыре четверти, 2020. — 312 с.;

18 Мовк, Саймон. Практическая электроника: иллюстрированное руководство для радиолюбителей. : Пер. с англ. – СПб. : ООО "Диалектика", 2020. - 352 с. : ил. - Парал. тит. англ.;

19 Никитина Т.В. Образовательная робототехника как направление инженерно-технического творчества школьников [Текст]: учебное пособие / Т.В. Никитина. – Челябинск: Изд-во Челяб. гос. пед. ун-та, 2014. – 169 с.;

20 Петин В. А. Проекты с использованием контроллера Arduino. — СПб.: БХВ-Петербург, 2014. — 400 с.;

21 Попов Е.П., Письменный Г.В. Основы робототехники: Введение в специальность М.: Высшая школа, 1990. 224 с.;

22 Проекты с использованием контроллера Arduino. – 2-е изд., перераб. и доп. – СПб.: БХВ-Петербург, 2015.- 464 с.: ил. – (Электроника);

23 Промышленные роботы. Конструирование и применение: Учеб. пособие. – 2-е изд., перераб. и доп.–К.: Высшая шк., 1991 – 311 с.: ил.;

24 Прохоренок Н. А. Основы Java. – СПб.: БХВ-Петербург, 2017. – 704 с.: ил.;

25 Руководства по API: Bluetooth [Электронный ресурс]: android-doc.github.io Режим доступа: https://android-doc.github.io/guide/topics/connectivity/bluetooth.html (Дата обращения – 23.12.2021);

26 С. Монк Программируем Arduino. Профессиональная работа со скетчами . — СПб.: Питер, 2017.;

27 С.Ф. Бурдаков А., В.А. Дьяченко. Проектирование манипуляторов промышленных роботов и роботизированных комплексов. М.: Высшая школа, 1986. – 264 с;

28 Соммер У. Программирование Микроконтроллерных плат Arduino/Freeduino/ - Петербург, 2012. - 256 с. ил - (Электроника);

29 Тимофеев А.В. Адаптивные робототехнические комплексы / Машиноведение.1988. – 392 с.;

30 Угрюмов Е. П. Цифровая схемотехника. СПб, БХВ-Петербург, 2004;

31 Филиппов С. А. Ф53 Уроки робототехники. Конструкция. Движение. Управление [Электронный ресурс] / С. А. Филиппов ; сост. А. Я. Щелкунова. — 2-е изд., испр. и доп. (эл.);

32 Фрайман Зэев Создание приложений для смартфонов и планшетов под ОС Android: Практический курс. Для школьников… и не только – М.: Ленанд. 2019 – 504 с.;

33 Фу К., Гонсалес Р., Ли К. Робототехника: Пер. с англ. – М.: Мир, 1989 – 624 с., ил.;

34 Хорстманн, Кей С. Java/ Библиотека профессионала, том 1. Основы. 11-е изд.: Пер. с англ. – СПб.: ООО «Диалектика», 2019 – 864 с.: ил. – Парал. тит. англ.;

35 Хуанг, Б. Arduino для изобретателей. Обучение электронике на 10 занимательных проектах: Пер. с англ./Б. Хуанг, Д. Ранберг. – СПб.: БХВ – Петербург, 2019. – 288 с.: ил.;

36 Чан Джейми Java: Быстрый старт. – СПб.: Питер, 2021. – 272 с.: ил. – (Серия «Библиотека программистов»);

37 Чернухин Ю.В. «Введение в роботехнику: Учебное пособие» - Таганрог: ТРТИ, 1990 - с.46;

38 Шахинпур М. Курс робототехники. М.: Мир– 1990. – 527 с.;

39 Юревич Е.И. Основы робототехники / Е.И. Юревич. –2-е изд., перераб. и доп. –СПб.: БХВ-Петербург, 2005. –416 с.;

40 4ne.ru [Электронный ресурс]: 4ne.ru Режим доступа: http://www.4ne.ru/stati/robotetxnika/manipulyatory-zaxvatnye-ustrojstva.html (Дата обращения – 29.05.2022);

41 Android. Программирование для профессионалов. 2-е изд. — СПб.: Питер, 2016. — 640 с.: ил. — (Серия «Для профессионалов»);

42 autoit-script.ru: Коды ASCII символов [Электронный ресурс]: autoit-script.ru Режим доступа: https://autoit-script.ru/docs/appendix/ascii.htm (Дата обращения-13.03.2022);

43 betacode: Руководство Android EditText [Электронный ресурс]: betacode.net Режим доступа: https://betacode.net/12657/android-edittext (Дата обращения – 23.12.2021);

44 developers: Подключение устройств Bluetooth [Электронный ресурс]: developer.android.com Режим доступа: https://developer.android.com/guide/topics/connectivity/bluetooth/connect-bluetooth-devices (Дата обращения – 23.12.2021);

45 docs.voltbro.ru: Что такое ROS. [Электронный ресурс]: docs.voltbro.ru Режим доступа: http://docs.voltbro.ru/starting-ros/ros-about.html (Дата обращения – 07.02.2022);

46 Marco Schwartz Проекты домашней автоматики на Arduino: Пер. с англ. - СПб.: БХВ-Петербург, 2016. - 113 с.: ил.;

47 McWhorter,W. I. The Effectiveness of Using Lego MindStorms Robotics Activities to Influence Self-Regular Learning in a University Introductory Computer Programming Course Dissertation of Doctor of Philosophy, Denton, Texas. 2008, 144

р. [Электронный ресурс] – Режим доступа: http://digital.library.unt.edu/ark:/67531/metadc6077/ (Дата обращения – 29.05.2022);

48 smartep.ru: Обзор программы RoboLab для программирования NXT роботов. [Электронный ресурс]: smartep.ru. Режим доступа: http://smartep.ru/lego/other/Robolab%20for%20NXT.pdf (Дата обращения – 04.07.2021);

49 StardAndroid [Электронный ресурс]: StardAndroid.ru Режим доступа: https://startandroid.ru/ru/uroki/vse-uroki-spiskom.html (Дата обращения – 29.05.2022);

50 TRICK STUDIO: TRICK STUDIO. [Электронный ресурс]: trikset.com. Режим доступа: https://trikset.com/products/trik-studio (Дата обращения – 09.02.2022).