федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

ИНСТИТУТ ЭНЕРГЕТИКИ И ЭЛЕКТРОТЕХНИКИ
(наименование института полностью)
Кафедра «Промышленная электроника»
(кафедра)
11.03.04. «Электроника и наноэлектроника»
(код и наименование направления подготовки, специальности)
«Промышленная электроника»
(направленность (профиль)/специализация)

БАКАЛАВРСКАЯ РАБОТА

на тему: Микроконтроллерное устройство дистанционного управления манипулятором

Студент С.А. Кривов _____
(И.О. Фамилия) (личная подпись)

Руководитель А.В. Прядилов _____
(И.О. Фамилия) (личная подпись)

Допустить к защите
Заведующий кафедрой А.А. Шевцов _____
(ученая степень, звание, И.О. Фамилия) (личная подпись)
«_____»_____20_____г.

Тольятти 2018г.

Аннотация

МИКРОКОНТРОЛЛЕРНОЕ УСТРОЙСТВО ДИСТАНЦИОННОГО УПРАВЛЕНИЯ МАНИПУЛЯТОРОМ

Цель работы: разработка устройства дистанционного управления манипулятором

Задачи работы:

1. Обзор существующих решений и выбор оптимального варианта
2. Выбор и расчет элементов в схему
3. Разработка аппаратной части модуля
4. Разработка программной части модуля
5. Сборка устройства

Работа состоит из четырех глав, в которых решены упомянутые задачи.

Разработка схем и печатных плат осуществлялась с помощью пакетов DipTrace, Компас 3D V16. Разработка кодов программы проводилось в среде разработки Arduino IDE, а так же Studlab.

Областью применения данной работы, создан действующий макет.

Содержание

Введение

Робот (от чеш. robot, robota – занятие, труд, также rob - раб) – машина, механическая, электрическая или программная, которая частично или полностью способна выполнять функции человека. Впервые термин «робот» был введен в научно-фантастической пьесе «Rossumovi univerzální roboti» (R.U.R, «Россумские универсальные роботы») Карелом Чапеком в 1920году, где роботами назывались люди, выращенные из искусственных материалов а так же состоящие из различных механизмов. С развитием электроники и механики появилась робототехника - отдельная наука, занимающаяся разработкой автоматизированных систем и разработкой роботов и кибернетика – наука о закономерностях, алгоритмах и процессах управления самых различных системах (и не только машинных) [4].

В современном мире практически невозможно представить жизнь без роботов, ведь они с каждым днем все больше и больше внедряются в повседневную жизнь, облегчая жизнь человека: в производстве, научно-исследовательской деятельности, в системах здравоохранения, коммерческих, развлекательных системах и даже дома. Некоторые типы роботов из различных сфер использования изображены на рисунке 1.

Наиболее широкое распространение роботы получили в промышленности, способные выполнять работу, исключая при этом вредные и опасные для человека факторы, таких как: перемещение и транспортировка крупногабаритных и тяжелых грузов, выполнение операций в агрессивной среде (высокая температура, сильнозагрязненная среда, радиация и т.д), выполнение сложных технологических операций (сварка, резка, штамповка), а так же монотонную периодичную работу (исключение «человеческого» фактора, тем самым уменьшая брак и повышая качество продукции). На сборочных линиях по всему миру широко применяются универсальные роботы-манипуляторы немецкой фирмы KUKA Roboter, представляющие собой «руку», размещенную на основании, и имеющую в зависимости от модели

несколько осей вращения [7]. На конце «руки», в зависимости от требуемой задачи расположен рабочий орган (сварочный аппарат, лазер или захватно-удерживающее устройство). Поставляются вместе с системами управления, представляющие собой программируемые контроллеры, пульты управления и системы диагностики состояния машины.

Примером сферы использования роботов в научно-исследовательской деятельности можно считать космос. Так называемые космороботы внесли огромный вклад в изучении космоса. Космороботы – роботы, нацеленные работать в космическом пространстве, в большинстве случаев работающие на солнечных батареях. Задачами таких роботов могут быть: сбор образцов различного грунта, пород, материалов, с последующим сканированием их и отправкой собранных данных учёным на Землю. В отличие от «земных», такие роботы обязаны: работать в сложных условиях враждебной среды, иметь как можно меньший вес и энергопотребление, работать в автоматическом режиме и обладать чрезвычайной надежностью.

Самые известные роботы, которые используют в космических исследованиях - это роверы (англ. rover – скиталец), к ним относятся всевозможные луно- и марсоходы (от первых Луноход-1 до современных марсоходов Curiosity). Они функционируют в автоматическом режиме и приспособлены для передвижения по поверхности другой планеты [5].

В системах здравоохранения в наши дни применяются роботы-хирурги, например итальянские манипуляторы ARES способны выполнять операции не повреждая кожные покровы. Пациент проглатывает его по частям, внутри робот собирает сам себя, после чего хирург осуществляет операцию удаленно, с помощью устройств управления. По окончании операции робот выходит через кишечник пациента. Так же актуальны роботы телеприсутствия – это комплексы, оснащенные камерами, дисплеями, динамиками и микрофонами, способные самостоятельно передвигаться. Есть возможность оснащения средствами для проведения диагностики и анализов, это может быть как возможность подключения к аппаратам (УЗИ, МРТ), так и встроенные приборы

(тонометр, анализаторы крови). И роботы-раздатчики и транспортировщики, предназначенные для выдачи лекарств на стационарных пунктах аптеки и доставки этих лекарств пациентам в соответствии с рецептом по заявке больного или доктора соответственно [6].

В домашнем быту, возможно, самыми популярными помощниками являются роботы-пылесосы, выполняющие автоматическую уборку в помещении. Современные роботы способны выполнять как влажную, так и сухую уборку помещения, оснащены датчиками расстояния или видеокамерами (для защиты от столкновения с препятствиями или от падения с лестницы), функциями программирования для задачи выполнения уборки по расписанию (например, по определенным дням в указанный промежуток времени) и функцией запоминания и построения маршрута для выполнения более качественной уборки. Источником питания служат аккумуляторы. Робот контролирует состояние заряда и самостоятельно направляется на «базу» - станцию для зарядки аккумуляторов (среднее время зарядки 2-5 часов), в модели NaviBot-S SR8980 фирмы Samsung база оснащается пылесборником, в который робот самостоятельно сбрасывает мусор после окончания уборки, тем самым повышая удобство в эксплуатации [4].

Рисунок 1 – Робот-манипулятор KUKA, марсоход Curiosity, доставщик лекарств Panasonic и пылесос Samsung

1 Состояние вопроса

1.1 Формирование актуальности, цели и задач проекта

Целью проекта является разработка микроконтроллерного устройства дистанционного управления манипулятором. Как говорилось выше, промышленный робот-манипулятор – это автоматическая машина, стационарная или передвижная, состоящая из исполнительного устройства в виде манипулятора, имеющего несколько степеней подвижности, и перепрограммируемого устройства программного управления для выполнения в производственном процессе двигательных и управляющих функций.

Существует множество систем управления промышленными роботами-манипуляторами. В зависимости от типа манипулятора, требуемой операции и среды использования применяются различные варианты управления. Но практически все системы должны удовлетворять следующим требованиям [4]:

- безопасность,
- эффективность,
- малые габариты,
- низкое энергопотребление,
- простота программирования (задания роботу выполнения требуемой программы) и управления, т.е. интуитивно понятный интерфейс,
- надежность,
- гибкость;

На современном производстве такие требования применяются по ряду следующих причин. В первую очередь система должна обеспечивать безопасность жизнедеятельности на производстве, т.е. исключить непреднамеренные действия, влекущие за собой травмы людей, поломку аппаратуры и другие нарушения технологического процесса. Для этого в системе управления должны быть предусмотрены различные комплексы, препятствующие выполнению непреднамеренных действий манипулятора. В

качестве примеров можно примести функцию автоматической блокировки робота-манипулятора, если в его рабочей зоне оказался человек, или алгоритмы защиты от непреднамеренного вмешательства оператором во время выполнения роботом операции (отключение сварочного аппарата на рабочем органе при попытке изменения его положения).

Эффективность. Согласно стандарту ISO 9000:2015, эффективностью называется отношение между достигнутым результатом и использованными ресурсами, такими как: экономический, временной, человеческий и т.д. Формулы расчета совокупности данных ресурсов и расчета эффективности здесь рассматриваться не будут. В целом можно сказать, что эффективная система управления должна оправдывать наличие манипулятора для выполнения определенной операции. Так же к эффективности можно отнести требования по низкому энергопотреблению и малые габариты, так как электроэнергия и площадь, требуемая для размещения манипулятора, являются ресурсами.

Отдельно следует рассмотреть параметры гибкость и интуитивность интерфейса. Поскольку зачастую именно они вместе с эффективностью влияют на итоговую оценку выбора пользователем нужной ему системы.

Гибкостью называется способность к интеграции в различные автоматизированные системы. Это означает, что одну систему управления можно применять для множества роботов-манипуляторов, выполняющих различные операции. А так же возможность легко адаптировать систему управления роботом к изменяющимся или новым производственным задачам. Реализуется за счет модульной конструкции оборудования и открытой архитектуре программного обеспечения.

В качестве примера можно привести пульт KUKA smartPAD, предназначенный для управления роботом KUKA KR 16-2, способного к быстрому изменению программы по сварке одного узла деталей на выполнение операции сварки иного узла деталей [7].

Пульт KUKA smartPAD и робот KUKA KR 16-2 изображены на рисунках 1.1 и 1.2 соответственно.

Интуитивностью интерфейса называется способность системы быть понимаемой, легко изучаемой, простой в использовании и привлекательной для пользователя в определенных условиях (ISO 25010). Достигается за счет повышения эргономичности – способности устройства управления предоставлять пользователю возможности для понятного, удобного, необременительного пользования.

Рисунок 1.1 – Пульт управления KUKA smartPAD

Рисунок 1.2 - Робот KUKA KR 16-2 в процессе выполнения операции сварки

Для реализации удобного интерфейса используются следующие принципы:

- Принцип обратной связи - пользователь получает информацию о работе системы путем получения понятных для него сигналов (визуальных, звуковых или тактильных, или же всех вместе взятых);

- Принцип «языка пользователя» - система должна использовать понятный для восприятия язык взаимодействия с пользователем;

- Принцип исключения ошибок - продуманный дизайн устройства управления, не позволяющий допустить условия возникновения ошибок. А так же способность своевременно детектировать возможные ошибки и оповещать об этом пользователя;

- Принцип доступности - система должна быть понятной для использования как новичкам, так и опытным пользователям;

- Принцип минимализма – система не должна отображать информацию, которая не нужна пользователю или которая может понадобиться ему в редких случаях;

Исходя из вышеперечисленного, задачами для реализации цели проекта по разработке устройства управления манипулятором проекта являются:

1) Провести анализ исходных данных и современных решений реализации систем управления манипуляторами;

2) Выбрать оптимальную систему, удовлетворяющую исходным требованиям;

3) Разработать схему электрическую принципиальную устройства системы управления, выбрать и рассчитать необходимый набор элементов,

4) Разработать печатную плату;

5) Разработать алгоритм работы устройства и программы управления;

6) Выполнить сборку манипулятора и устройства управления и провести проверку и отладку работы программы.

1.2 Анализ исходных данных и известных решений

Очевидно, что для создания устройства управления манипулятором самым важным основополагающим критерием для начала разработки является сам манипулятор. А именно: тип манипулятора, его конструкция и тип рабочего органа манипулятора.

Исходные данные

В данном проекте рассматривается разработка устройства управления манипулятором SNARM SG90 китайской компании Sinoning, занимающаяся разработкой и продажей наборов для сборки различных типов настольных манипуляторов, предназначенных для обучения студентов колледжей и ВУЗов, а так же для развлекательных целей. Особенностью выпускаемой продукции является наличие линеек моделей для пользователей с различным уровнем знаний в электронике и робототехнике, т.е. от новичка до опытного пользователя. Для новичков выпускаются уже готовые наборы для создания полнофункционального робота. В набор входят манипулятор и плата управления. Плата состоит из микроконтроллера, на который уже установлена программа и устройств управления, причем пользователю доступны на выбор различные виды: кнопочные дистанционные пульты, пульты с джойстиками и т.д. Достаточно собрать все компоненты на плате и манипулятор готов к использованию. Для опытных пользователей предоставляется широкий набор модулей для создания уникального манипулятора и системы управления, начиная от выбора платформы робота (статичной, роликовой или гусеничной) и рабочего органа манипулятора (клешня, крепление для камеры) до выбора компонентов для создания системы управления (микроконтроллеры, дисплеи, датчики и др.). Более подробную информацию можно получить на официальном сайте [10].

Манипулятор SNARM SG90, изображенный на рисунке 1.3, является одной из самых распространённых моделей роботов для новичков благодаря своей доступности и простоты конструкции. Манипулятор представляет собой

сборную модель, состоящую из акриловых деталей и крепежных элементов, имеющий 4 степени свободы. Приводами всех четырех осей являются серводвигатели TowerPro SG90.

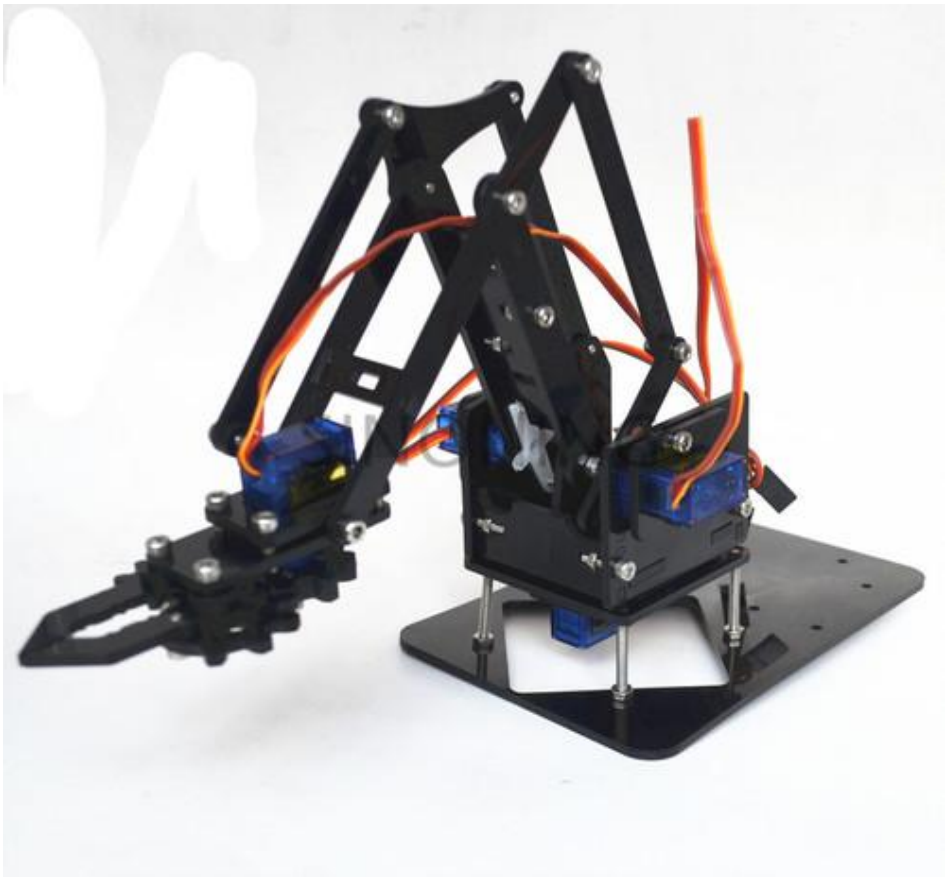


Рисунок 1.3 – Манипулятор SNARM SG90 в сборе

Принцип работы серводвигателя TowerPro SG90

Серводвигатели (или сервоприводы), в отличие от двигателей постоянного тока, имеют обратную связь для определения угла положения вала. Принцип обратной связи реализован с помощью потенциометра, соединенного с приводным валом двигателя. По показаниям потенциометра определяется угол положения сервопривода. Стандартные сервоприводы имеют фиксированный диапазон угла поворота, обычно от 0 до 180 градусов.

С помощью микроконтроллера можно команд можно установить вал сервопривода в положение, в котором он будет находиться до поступления

новых команд. Управление сервоприводом осуществляется подачей микроконтроллером по сигнальному проводу прямоугольного импульса, длина которого определяет угол поворота вала. Для стандартного сервопривода с диапазоном от 0 до 180 градусов подача импульса длительностью 1мс приводит к установке сервопривода в положение 0°, импульс длительностью 1,5мс устанавливает сервопривод в положение 90°, импульс 2мс в положение 180°. После подачи импульса вал сервопривода устанавливается в заданное положение и остается там до поступление новой команды. Но чтобы поддерживать заданный угол в фиксируемом положении, необходимо отправлять импульсы с определенной частотой, для стандартных сервоприводов обычно используется частота 20мс. Временные диаграммы зависимости угла вала сервопривода от длительности импульса приведены на рисунке 1.4.

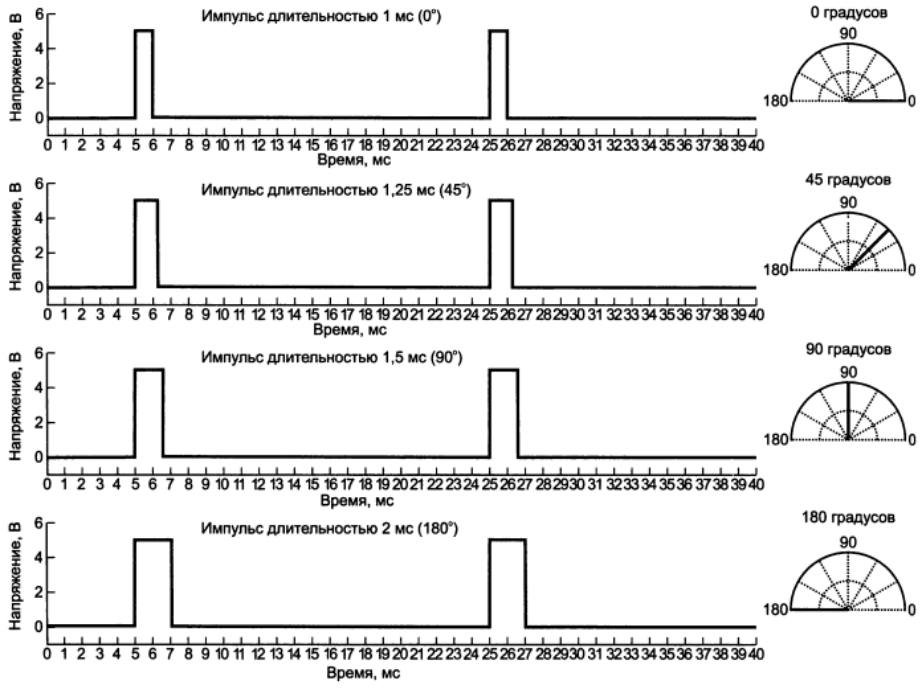
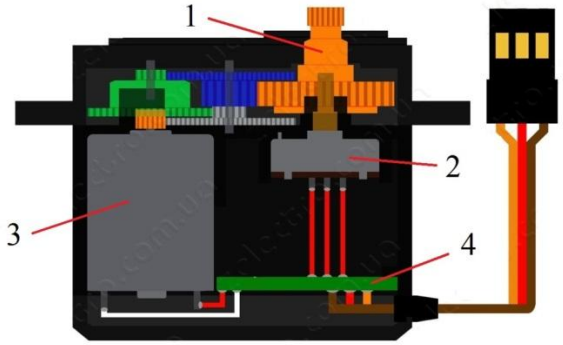


Рисунок 1.4. – Зависимость угла поворота вала от длительности импульса

Параметры и характеристики сервопривода TowerPro SG90

Внешний вид SG90 и его внутреннее устройство изображены на рисунке 1.5 и 1.6 соответственно. Сервопривод состоит из: редуктора , электродвигателя постоянного тока с установленным на валу потенциометром и модуля управления . Модуль обрабатывает входные сигналы (импульсы), поступающие с микроконтроллера. При поступлении входного сигнала модуль вычисляет разницу между значением этого сигнала с текущим значением положения вала благодаря сопротивлению потенциометра. Если разница между показаниями присутствует, модуль подает импульс определенной полярности (в зависимости от направления вращения) длительностью, равной значению этой разницы.

Рисунок 1.5 – Сервопривод SG90, внешний вид



1 – редуктор, 2 – потенциометр, 3 – электродвигатель, 4 – модуль управления

Рисунок 1.6 – Устройство сервопривода TowerPro SG90

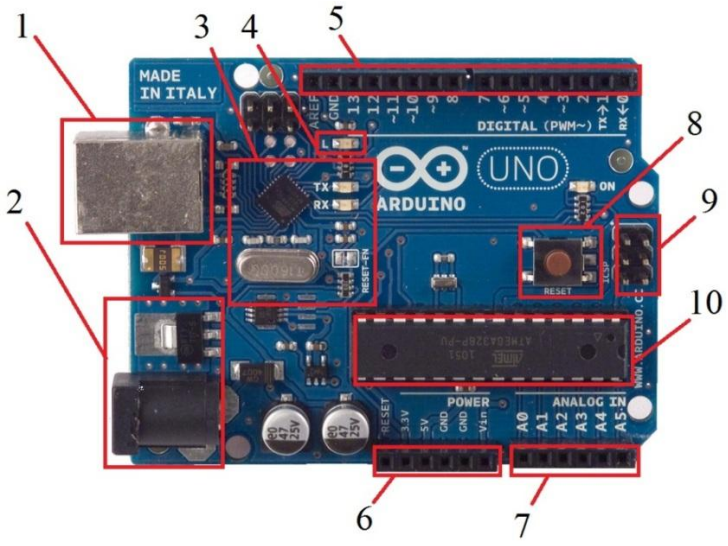
Характеристики TowerPro SG90:

- Материал шестерней редуктора: пластик;
- Рабочее напряжение: 3.5-7.2 В;
- Номинальный потребляемый ток: 50-80 мА;
- Ток в режиме нагрузки 150-250 мА;
- Крутящий момент: 1,8кг/см;
- Рабочая скорость: 0.12сек/60°;
- Допустимый диапазон вращения: 180°;
- Диапазон рабочих температур: -30℃…+60℃;
- Размеры: 23 х 12.2 х 29 мм;
- Вес: 9г.

Зная принцип управления сервоприводом TowerPro SG90, рассмотрим известные актуальные системы управления SNARM SG90.

На сегодняшний день практически все системы управления данным манипулятором выполнены на базе Arduino, что представляет собой данная база?

Arduino – открытая платформа для разработки устройств на микроконтроллерах ATmega с помощью упрощенного языка программирования C++ в Arduino IDE (англ. Integrated Development Environment – интегрированная среда разработки). Все платы Arduino содержат основные компоненты, необходимые для программирования и совместной работы с другими схемами [1]. Общий вид самой распространенной платы Arduino UNO R3 (далее Arduino UNO) приведен на рисунке 1.7. Написание программ значительно упрощает возможность подключения библиотек – разработанный заранее код для решения определенных задач. Достаточно подключить нужную библиотеку и в основном коде вызвать нужную функцию библиотеки соответствующим оператором.



1 – USB-разъем DB90, 2 – разъем для подключения внешнего источника питания 7-12 В, 3 – конвертер последовательного и USB интерфейсов, 4 – отладочный светодиод, 5 – цифровые контакты входа/выхода, 6 – контакты питания и вспомогательные контакты, 7 – входы аналогово-цифрового преобразователя, 8 – кнопка перезагрузки, 9 – контакты интерфейса ICSP для перепрограммирования, 10 – микроконтроллер ATmega

Рисунок 1.7 – Плата Arduino UNO

Основным элементом Arduino, безусловно, является микроконтроллер ATmega фирмы Atmel. На большинстве плат Arduino, включая UNO R3, используется микроконтроллер ATmega328, исключением являются плата Due, использующая микроконтроллер ARM Cortex.

Микроконтроллер выполняет весь скомпилированный код программы, написанный пользователем в IDE. Язык Arduino предоставляет доступ к периферийным устройствам: аналого-цифровым преобразователям (АЦП), цифровым портам ввода-вывода, коммуникационным шинам (I^2C и SPI) и последовательным интерфейсам. На плате все эти порты выведены на штырьковые контакты.

Программирование микроконтроллера через USB-порт возможно без дополнительного программатора, благодаря загрузчику Arduino, записанному в микроконтроллер ATmega на заводе-изготовителе.

Интерфейсом между USB и микроконтроллером служит дополнительный контроллер ATmega16U2 (установлен на большинстве современных плат Arduino, ранее на некоторых версиях платы устанавливался ATmega 8U2).

Загрузчик – фрагмент программного кода, записанный в резервную ячейку памяти микроконтроллера и позволяющий загружать написанную пользователем программу на плату по последовательному порту UART (англ. Universal Synchronous/Asynchronous Receiver/Transmitter - универсальный синхронный /асинхронный приемник/передатчик). Загрузчик запускается сразу после включения Arduino и в течение нескольких секунд ожидает команду программирования от IDE. Если команда поступила, выполняется сброс микроконтроллера и подготовка к перепрограммированию. Далее внешний компьютер начинает отправлять код новой программы. Если команда не поступила, запускается последняя программа, находящаяся в памяти Arduino.

Загрузчик имеет несколько недостатков: во-первых, он занимает место в памяти микроконтроллера (примерно 2 Кбайт), которое можно было бы использовать для написания программ, и во-вторых, как говорилось ранее, после включения Arduino, запуск основной программы будет выполняться с

задержкой, так как загрузчик ожидает запрос на перепрограммирование. Поэтому если наличие загрузчика не требуется, его возможно удалить из памяти микроконтроллера, и, при необходимости перепрограммирования, предусмотрена возможность подключения внешнего программатора к порту интерфейса ICSP. В качестве внешнего программатора можно использовать как специализированное устройство, так и другую плату Arduino, запрограммированную на данную функцию. Примеры таких устройств приведены на рисунке 1.8.

Arduino без загрузчика, подключенная к Arduino плате-программатору (слева). Внешний программатор USBASP ISP V2.0 (справа)

Рисунок 1.8 – Примеры внешних программаторов платы Arduino

Связь платы Arduino с периферийными устройствами ввода-вывода информации (датчики, кнопочные переключатели, дисплеи, иные модули и тд.) осуществляется по комбинированным портам ввода-вывода. Комбинированными они являются потому, что многие из контактов помимо функции ввода-вывода так же имеют дополнительные возможности: различные коммуникационные интерфейсы, последовательные интерфейсы (вывода 0 и 1 Arduino UNO), широтно-импульсные модуляторы (вывода 3,5,6,9,10 и 11

Arduino UNO) и внешние прерывания. Все порты могут служить цифровыми входами и выходами. Порты входов аналоговых сигналов выведены на отдельную группу контактов.

Для автономной работы платы через разъем возможно подключение внешнего источника питания. Рекомендуемое напряжение источника 7-12 В, иначе возможен перегрев и выход из строя стабилизатора напряжения. Дополнительно на плате присутствует группа контактов для подключения внешнего источника питания:

- Контакт 5V используется для подключения источника напряжением 5 В. Если питание осуществляется через разъем, данный контакт можно использовать для подключения других периферийных устройств, работающих от стабилизированного напряжения 5 В;

- Контакт Vin так же используется для питания Arduino от источника с уровнем напряжения 7-12 В;

- Напряжение 3,3 В выведено на контакт 3,3V для подключения внешних устройств;

- Два вывода GND предназначены для подключения платы и других устройств к минусовому выводу источника питания;

- Перезагрузка платы возможна через подачу логического нуля на вход сброса RESET или через кнопку перезагрузки.

Параметры и характеристики Arduino UNO [11]*:*

- Микроконтроллер: ATmega328;

- Тактовая частота: 16 МГц;

- Напряжение логических уровней: 5 В;

- Входное напряжение питания: 7–12 В;

- Количество портов ввода-вывода общего назначения: 20;

- Максимальный ток с одного порта ввода-вывода: 40 мА;

- Максимальный выходной ток порта 3.3V: 50 мА;

- Максимальный выходной ток порта 5V: 800 мА;

- Количество портов с поддержкой широтно-импульсной модуляции: 6;

- Количество портов, подключённых к АЦП: 6;
- Разрядность АЦП: 10 бит;
- Объем Flash-памяти: 32 КБ;
- Объем EEPROM-памяти: 1 КБ;
- Объем оперативной памяти: 2 КБ;
- Габаритные размеры платы: 69×53 мм.

Анализ известных решений

1) Микроконтроллерное устройство с резистивными органами управления

На официальном сайте компании Sinoning представлена система управления SN1900 kit, позволяющая управлять осями манипулятора с помощью потенциометров [10]. Вращая ручку одного из четырех потенциометров, осуществляется вращение оси сервопривода, соответствующего каждой из этих ручек. Фото системы в сборе с манипулятором приведено на рисунке 1.9.

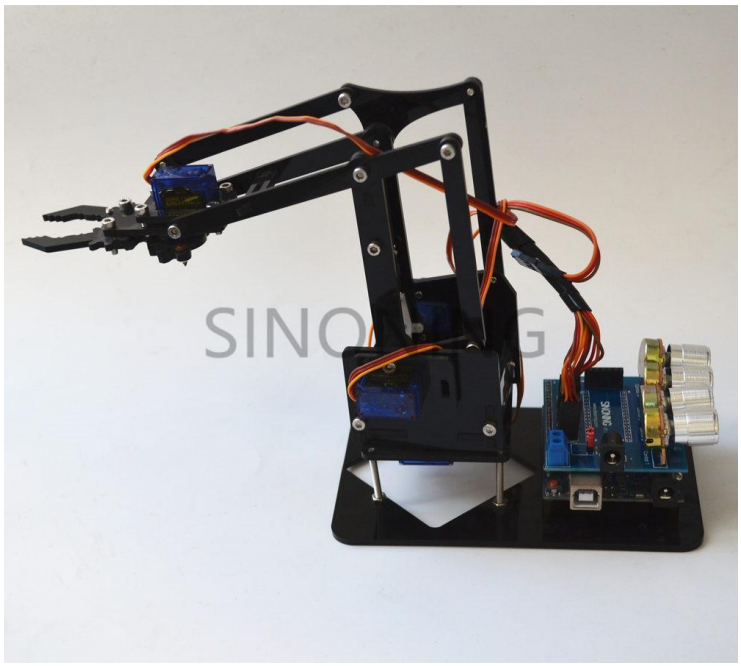


Рисунок 1.9 – Система управления SN1900 kit

В набор данной системы входят:

- Плата Arduino UNO;
- Набор деталей для сборки манипулятора SNARM SG90;
- 4 сервопривода TowerPro SG90;
- Блок питания 5 В, 2А;
- Плата Potentiometer control board.

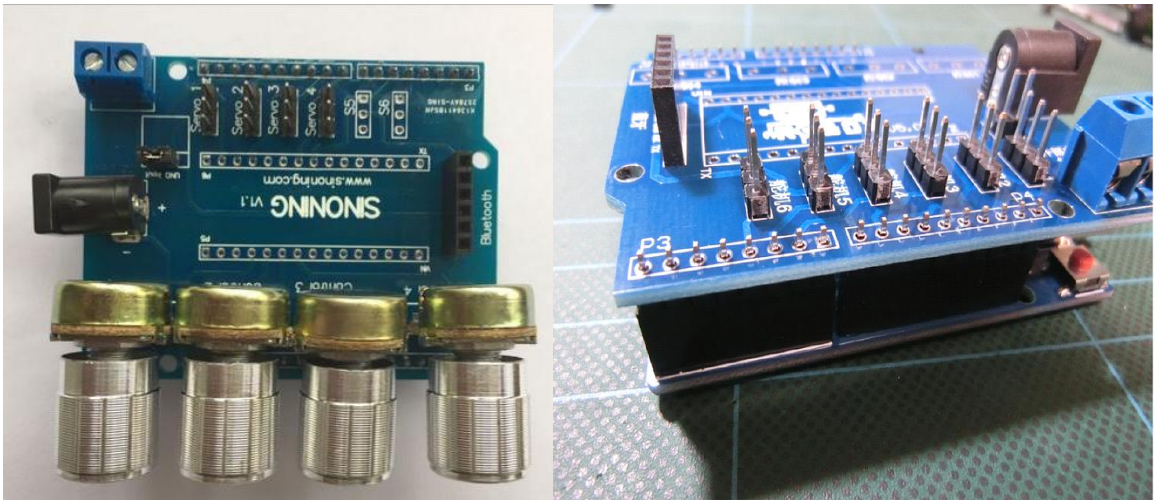
Изображенная на рисунке 1.10 плата Potentiometer control board (рус. плата управления потенциометрами) представляет собой дополнительный модуль расширения, который в сборе с основной платой Arduino UNO образует готовое устройство управления манипулятором SNARM SG90. Плата позволяет подключить сервоприводы и потенциометры к основной плате Arduino UNO без использования макетной платы и дополнительной пайки. Potentiometer control board имеет идентичные с основной платой габаритные размеры и посадочные места под штырьковые контакты для цифровых и аналоговых портов, а так же портов питания.

Выхода потенциометров на плате модуля выведены на штырьковые контакты группы аналоговых входов, сигнальные входы для управления сервоприводами выведены на контакты, поддерживающие широтно-импульсную модуляцию (ШИМ), порты группы питания расположены идентично Arduino UNO. Этими штырьковыми контактами плата-модуль вставляется в порты Arduino UNO.

Так же на плате присутствует порт для подключения Bluetooth модулей типа HC-05 или HC-06 для реализации дистанционного управления (Bluetooth модули не входят в комплект SN1900 и приобретаются отдельно).

Разъем для подключения внешнего источника питания у Potentiometer control board свой, так как суммарный ток потребления всех четырех сервоприводов превышает допустимый выходной ток контакта 5V платы Arduino UNO. На плате-модуле присутствует перемычка UNO input, позволяющая запитать основную плату, но в этом случае наблюдается дребезг

осей сервоприводов, поэтому рекомендуется использовать отдельный источник питания для основной платы.



Potentiometer control board, вид сверху (слева). Плата управления в сборе с Arduino UNO(справа)

Рисунок 1.10 – Плата Potentiometer control board

Таким образом, после соединения двух плат, контакты выходов потенциометров и сигнальные входа для управления сервоприводов подключены к портам основной платы Arduino. После подключения сервоприводов (порядок подключения указан в инструкции) и подачи питания на Potentiometer control board, пользователю необходимо загрузить скачанную с сайта Sinoning.cc программу управления в Arduino UNO. Устройство управления манипулятором готово. Чтобы понять принцип действия данной системы управления, рассмотрим подробнее код программы, находящийся в общем доступе на сайте производителя [9].

```
1 #include "VarSpeedServo.h"
2
```

```
3  VarSpeedServo servo1;
4  VarSpeedServo servo2;
5  VarSpeedServo servo3;
6  VarSpeedServo servo4;
7
8  int potpin1 = 0;
9  int potpin2 = 1;
10 int potpin3 = 2;
11 int potpin4 = 3;
12
13 int val1;
14 int val2;
15 int val3;
16 int val4;
17
18 void setup()
19 {
20   servo1.attach(11);
21   servo2.attach(10);
22   servo3.attach(9);
23   servo4.attach(6);
24 }
25 void loop()
26 {
27     val1 = analogRead(potpin1);
28     val1 = map (val1, 0, 1023, 90, 160);
29     servo1.write(val1);
30     delay(10);
31
32     val2 = analogRead(potpin2);
33     val2 = map (val2, 0, 1023, 0, 179);
34     servo2.write(val2);
35     delay(10);
36
```

```
37    val3 = analogRead(potpin3);
38    val3 = map (val3, 0, 1023, 0, 179);
39    servo3.write(val3);
40    delay(10);
41
42    val4 = analogRead(potpin4);
43    val4 = map (val4, 0, 1023, 40, 179);
44    servo4.write(val4);
45    delay(10);
46  }
```

В начале кода программы (1 строка) выполняется подключение библиотеки VarSpeedServo.h, упрощающая написание кода для управления сервоприводами.

Далее (строки 3-17) создаются объекты сервоприводов, имеющие свое индивидуальное имя, создаются переменные для хранения значений положения ручек потенциометров и углов поворота сервоприводов.

Процедура setup() всегда выполняется один раз при включении микроконтроллера и используется для определения различных функций, конфигурации режимов работы выводов и инициализации переменных. В данном случае (строки 18-24), выполняется присвоение каждого объекта сервопривода к определенному выводу Arduino (номер вывода указан в скобках), а так же запускается процедура Serial, предназначенная для обмена данными между внешним компьютером и Arduino по последовательному порту.

Далее следует процедура loop() которая всегда выполняется циклично на протяжении всей работы программы, в ней и сосредоточен основной код программы. Процедуры setup() и loop() объявляются оператором void.

В процедуре loop (25-46 строки) содержится основной алгоритм управления манипулятором. Его суть состоит в том, что каждому потенциометру соответствует один сервопривод, двигающий определенную ось манипулятора. При вращении ручки потенциометра, нужный сервопривод

сдвигается на требуемый угол. В качестве примера рассмотрим алгоритм управления первой осью сервопривода.

Сначала выполняется присвоение переменной val1 данных, считанных с потенциометра potpin1 (строка 27). В среде Arduino IDE чтение аналоговых данных выполняется командой analogRead (следует обратить внимание, что язык программирования в Arduino IDE является регистр-зависимым, и некорректное написание команды приведет к ошибке компилирования). Эти данные АЦП преобразует в двоичный код с разрядностью 2^{10}, т.е. диапазон таких данных всегда будет от 0 до 1023. Поскольку допустимый диапазон вращения сервопривода TowerPro SG90 составляет 0-180°, то их нужно масштабировать. Это выполняется командой map (строка 28). После масштабирования до допустимого диапазона значение высчитанного угла val1 записывается в сервопривод servo1 командой write (строка 29). Данный цикл выполняется с паузой в 10 миллисекунд. Величина задержки между выполнением цикла задается командой delay (строка 30). Здесь величина задержки влияет на скорость вращения вала сервопривода и значение в 10 миллисекунд недостаточно для установления вала в точное положение. Алгоритмы управления остальными сервоприводами (32-45 строки) аналогичны вышеописанному, отличия заключаются в названии переменных объектов сервоприводов и потенциометров.

Синтаксисы всех ранее упомянутых функций среды Arduino IDE будут подробно описаны в следующей главе.

Рассмотрим достоинства и недостатки системы управления SN1900.

Достоинства:

- Простота сборки;

- Интуитивно-понятный интерфейс управления.

Недостатки:

- Для управления манипулятором необходимо постоянное присутствие пользователя в непосредственной близости;

- При необходимости изменения скорости вращения сервоприводов придется перепрограммировать контроллер;

- Плата-модуль полностью закрывает неиспользуемые контакты основной платы, что делает невозможным аппаратную модификацию и усовершенствование системы кроме подключения Bluetooth модулей, так как перекрываются порты, которые можно было бы использовать для подключения других устройств;

- Следствием из предыдущего недостатка является плохая визуальная сигнализация работы устройства, сигнализирующие светодиоды включенного питания (PWR), отладки (LED 13) и работы последовательного интерфейса (RX и TX) частично закрыты от пользователя, что так же делает затруднительным модификацию и отладку дистанционного управления путем подключения Bluetooth модуля;

- Для стабильной работы системы необходимо использовать два источника питания.

2) Микроконтроллерное устройство с проводным пультом управления

Еще одним продуктом на сайте компании Sinoning в качестве системы управления манипулятором SNARM SG90 выступает набор SNAM2000 [10], позволяющий пользователю управлять рукой манипулятора с помощью проводного пульта с джойстиками в качестве органов управления. Фото системы управления в сборе приведено на рисунке 1.11.

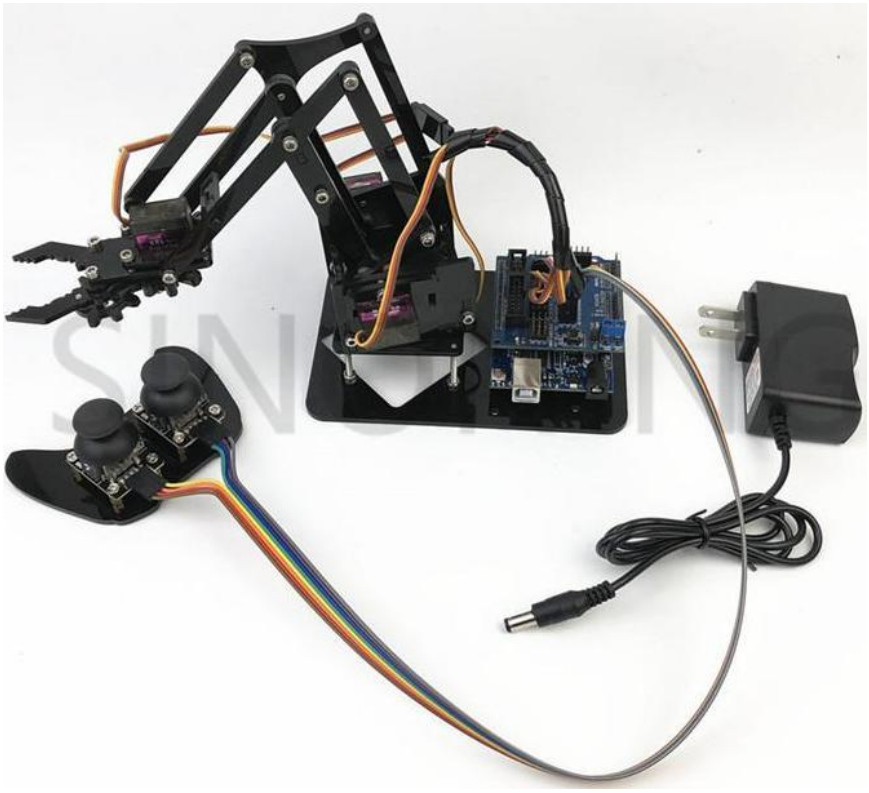


Рисунок 1.11 – Система управления SNAM2000

Комплект содержит:

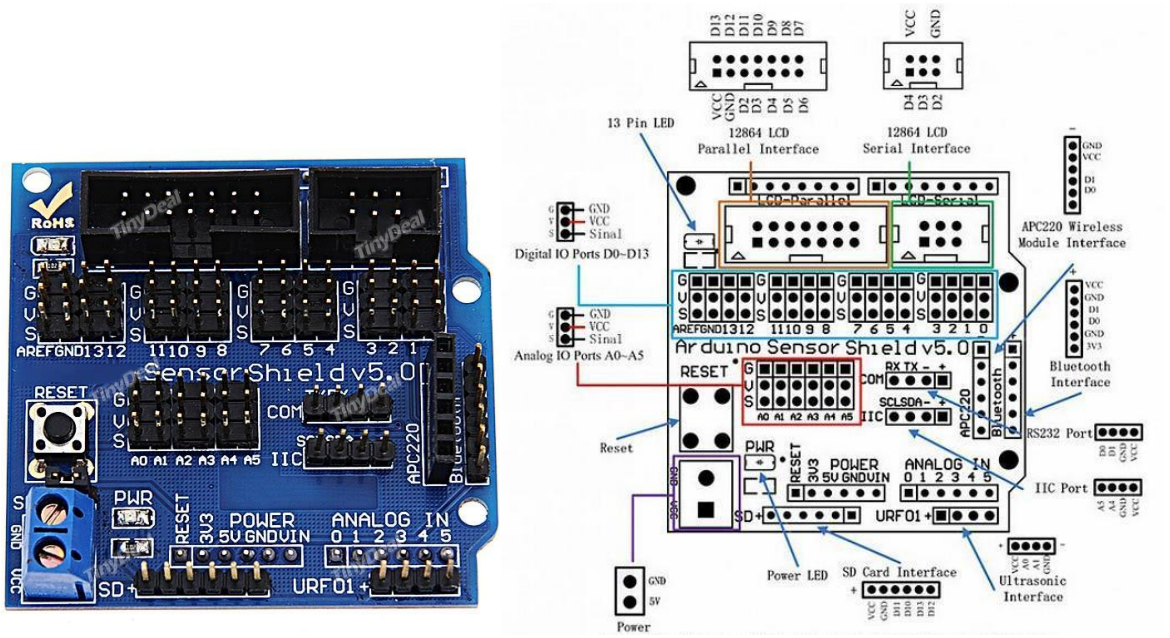
- Плата Arduino UNO;
- Набор деталей для сборки манипулятора SNARM SG90 и пульта управления;
- 4 сервопривода TowerPro SG90;
- Блок питания 5 В, 2А;
- Два джойстика PS2 rocker;
- Провода для подключения пульта к плате;
- Плата Sensor expansion board V5.

Подключение всех исполнительных устройств осуществляется через плату-модуль Sensor expansion board V5 (англ. плата расширения). Исходя из названия следует, что плата, изображенная на рисунке 1.12 а, представляет собой дополнительный модуль расширения для более удобного подключения

всевозможных датчиков информации, устройств ввода/вывода и других устройств к основной плате Arduino UNO без использования макетной платы и пайки. Это достигается благодаря трассировке печатной платы, удачно образующей следующие группы интерфейсов [12]:

- Иинтерфейс для подключения карт памяти SD;
- Интерфейс для подключения ультразвуковых датчиков;
- Разъем для подключения внешнего источника питания (блока питания или батареи);
- Колодка питания Arduino, обозначение выводов: RESET, 3V3, 5V, GND, VIN;
- Иинтерфейс аналоговых входов-выходов;
- Последовательная шина данных для связи интегральных схем;
- Интерфейс RS232 (последовательная шина COM);
- Интерфейс для подключения внешнего Wi-Fi модуля ACP220;
- Иинтерфейс для подключения внешнего устройства Bluetooth;
- Колодка цифровых входов-выходов;
- Параллельный порт для подключения жидкокристаллического дисплея);
-Последовательный порт для подключения жидкокристаллического дисплея).

Функциональная диаграмма интерфейсов приведена на рисунке 1.12 б.



а) внешний вид платы, б) диаграмма интерфейсов

Рисунок 1.12 – Плата Sensor expansion board V5

Аналогично плате Potentiometer control board, модуль Sensor expansion board V5 вставляется своими штырьковыми контактами в порты основной платы Arduino UNO. Для более комфортной работы с основной платой были учтены некоторые конструктивные особенности, например на данном модуле присутствуют сигнализирующие светодиоды отладки и включенного питания, а так же присутствует кнопка сброса Reset, но отсутствуют сигнализирующие светодиоды работы последовательного интерфейса RX и TX.

Питание платы возможно как через Arduino UNO, так и от внешних источников питания, но при этом отсутствует стандартное для Arduino UNO гнездо питания типа DC 2,1x5,5мм, имеется только клеммная колодка, что делает затруднительным подбор другого источника питания. Переключение с источника питания основной платы на внешний источник осуществляется с помощью перемычки SEL.

Изображенный на рисунке 1.13 пульт управления является органом управления манипулятором и выполнен в виде двух джойстиков типа PS2 rocker.

Рисунок 1.13 – Пульт управления системы SNAM2000

Джойстик PS2 rocker является аналого-резистивным, т.е. на каждой оси отклонения джойстика (вертикальной и горизонтальной) установлен потенциометр.

Выходное аналоговое значение каждого потенциометра изменяется при отклонении ручки джойстика. Сигнальные контакты потенциометров выведены на отдельные контакты PS2 rocker и подключается к выходам АЦП Arduino [13]. На джойстике присутствует кнопка, управление которой происходит нажатием на ручку джойстика через механический толкатель. Выход сигнала с кнопки выведен на отдельную ножку и обычно подключается к цифровым портам Arduino. Сигнал с кнопки инвертирован, т.е. в состоянии покоя сигнал имеет логический уровень 1, а при нажатии принимает значение 0.

Механическая конструкция джойстика устроена так, что ручка джойстика всегда возвращается в исходное центральное положение при прекращении на нее воздействия.

В системе управления SNAM2000 пульт с джойстиками подключается к плате Arduino UNO через плату расширения Sensor expansion board V5. Интерфейсом между пультом и платой-модулем является проводная шина, длина провода составляет 500 мм. Кнопки на джойстиках PS2 rocker не задействованы в системе управления. После сборки всех компонентов согласно инструкции и загрузки программы в основную плату система SNAM2000 готова к использованию. При отклонении ручки джойстика в определённой оси, соответствующий ей сервопривод приводится в движение. Для понимания алгоритма работы системы рассмотрим исходный код программы, находящийся в общем доступе на сайте производителя [9].

```
1  #include "VarSpeedServo.h"
2  VarSpeedServo servo1;
3  VarSpeedServo servo2;
4  VarSpeedServo servo3;
5  VarSpeedServo servo4;
6  int potpin1 = 0; int potpin3 = 2;
7  int potpin2 = 1; int potpin4 = 3;
8  int val1; int val2;
9  int val3; int val4;
10 static int s1 = 70; static int s2 = 110;
11 static int s3 = 100; static int s4 = 80;
12 void setup(){
13   servo1.attach(11); servo1.write(70);
14   servo2.attach(10); servo2.write(110);
15   servo3.attach(9); servo3.write(100);
16   servo4.attach(6);  servo4.write(80);
17   //Serial.begin(9600); }
18 void loop(){
```

```
  val1 = analogRead(potpin1);
  if (val1 < 100) {
    s1 = s1 - 1;
    if (s1 <= 10) {
      s1 = 10; }
    //Serial.println(s1);
    servo1.write(s1);
    delay(50); }
  if (val1 > 900) {
    // Serial.println(s1);
    s1 = s1 + 1;
    if (s1 >= 170) {
      s1 = 170; }
    servo1.write(s1);
    delay(50); }
  val2 = analogRead(potpin2);
  if (val2 > 900) {
    s2 = s2 - 1;
    if (s2 <= 10) {
      s2 = 10; }
    //Serial.println(s2);
    servo2.write(s2);
    delay(50); }
  if (val2 < 100) {
    // Serial.println(s1);
    s2 = s2 + 1;
    if (s2 >= 170) {
      s2 = 170; }
    servo2.write(s2);
    delay(50); }
  val3 = analogRead(potpin3);
  if (val3 < 100) {
    s3 = s3 - 1;
    if (s3 <= 10) {
```

```
53      s3 = 10; }
54    //Serial.println(s1);
55    servo3.write(s3);
56    delay(50); }
57   if (val3 > 900) {
58    s3 = s3 + 1;
59    if (s3 >= 170) {
60      s3 = 170; }
61    servo3.write(s3);
62    delay(50); }
63   val4 = analogRead(potpin4);
64   if (val4 < 100) {
65    s4 = s4 - 1;
66    if (s4 <= 80) {
67      s4 = 80; }
68    servo4.write(s4);
69    delay(50); }
70   if (val4 > 900) {
71    s4 = s4 + 1;
72    if (s4 >= 130) {
73      s4 = 130; }
74    servo4.write(s4);
75    delay(50); }}
```

Аналогично описанному ранее коду программы системы управления SG1900, в начале кода SNAM2000 происходит подключение вспомогательной библиотеки ValServoSpeed (1 строка). Далее создаются объекты сервоприводов (строки 2-5), определяются порты АЦП для обработки данных с джойстиков переменные для хранения значений положений джойстиков (строки 6-9). Так же объявляются инициальные значения переменных углов сервоприводов s1-s4 (строки 10-11).

Как уже известно, функция setup выполняется один раз при включении устройства. В ней выполняется определение каждого сервопривода к

цифровому выводу Arduino UNO и подача команд для выставления сервоприводов в положения, указанные в инициальных значениях s1-s4 (строки 13-16), чтобы при включении устройства манипулятор вставал в исходное положение, и только потом принимал дальнейшие команды для управления.

В строке 17 содержится процедура инициализации функции последовательного интерфейса Serial, для обмена данными между внешним компьютером и устройством. Символы // обозначают, что функция обозначена как комментарий. Комментарии не обрабатываются компилятором, следовательно, не выполняются программой. Можно предположить, что функция Serial была использована для отладки кода на стадии проектирования.

Далее в циклической функции loop описан весь алгоритм управления манипулятором (18-75). Поскольку алгоритмы для всех четырех сервоприводов одинаковы, рассмотрим только алгоритм управления одной осью сервопривода в одном направлении (строки 19-26).

```
19  val1 = analogRead(potpin1); // считываение по оси X
20  if (val1 < 100) {// если ручка джойстика в крайнем положении
21    s1 = s1 - 1; // начать декремент s1
22      if (s1 <= 10) {// если сервопривод достиг крайней точки
23        s1 = 10; } // остановить декремент до поступления иной команды
        //Serial.println(s1); // вывод значения угла в монитор последовательного порта
24  для отладки
25  servo1.write(s1); // принять сервоприводу servo1 угол равный s1
26  delay(50); } // задержка после выполнения команд 50мс
```

Вначале в переменную val1 записывается считанное с АЦП значение положения джойстика на контакте potpin1 (строка 19). После этого с помощью функции if выполняется проверка условия: если значение val1 меньше ста (т.е. ручка джойстика находится в крайнем положении), то переменная хранения угла положения сервопривода s1 декрементируется (сервопривод приводится в движение). В теле этой функции содержится еще одно условие: если значение

угла s1 стало меньше порогового значения, то дальнейший декремент невозможен (строки 20-24). С помощью такого алгоритма задаются границы диапазонов вращения осей сервопривода для предотвращения механического повреждения сервопривода в случае получения команды со значением, выходящем за границы заданного диапазона. Далее отправляется команда на изменение угла сервопривода servo1 на значение s1 (строка 25).

Достоинства системы SNAM2000:

- Благодаря плате Sensor expansion board V5 есть возможность модификации системы путем добавления новых устройств;

- Более удобный и интуитивно-понятный интерфейс в сравнении с SN1900;

- Предусмотрена защита от механического повреждения сервоприводов.

Недостатки:

- При необходимости изменения скорости вращения сервоприводов придется перепрограммировать контроллер;

- Дальность дистанционного управления ограничена длиной провода между пультом и устройством управления;

- Модуль Sensor expansion board V5 не имеет разъема для подключения блока питания со штекером DC 2,1x5,5, который поставляется в комплекте. В этом случае работа устройства возможна только при подключении питания через плату Arduino UNO, что может привести к преждевременному выходу платы из строя, так как при работе нескольких сервоприводов ток потребления значительно выше тока отдачи Arduino UNO. Этот недостаток может являться критичным при выборе данной системы.

3)Устройство управления на базе радиоканального модуля связи

Последним предлагаемым продуктом компании Sinoning для управления манипулятором SNARM SG90 является система управления PS2 SNAR10 [10], изображенная на рисунке 1.14.

Рисунок 1.14 – система PS2 SNAR10 в сборе

В комплект системы входят:

- Плата Arduino UNO;
- Набор деталей для сборки манипулятора SNARM SG90;
- 4 сервопривода TowerPro SG90;
- Блок питания 5 В, 2А;
- Плата PS2 shield Arduino;
- Пульт управления PS dual shock 2 и плата приемник PS2 2.4 ГГц;
- Два электродвигателя постоянного тока.

Дистанционное управление манипулятором реализовано по радиоканалу с частотой 2.4 ГГц, передатчиком является беспроводной пульт PS2 dual shock, приемник подключен к плате расширения PS2 shield Arduino, изображенной на рисунке 1.15.

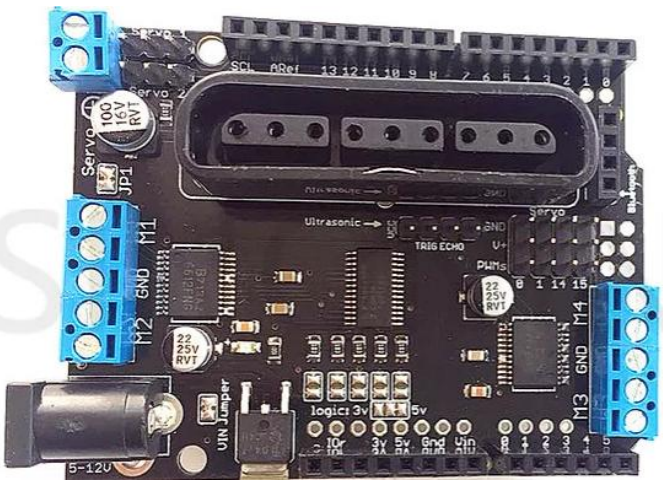


Рисунок 1.15 – Плата PS2 shield Arduino

Данная плата является так же одной из разновидностей модулей расширения функциональных возможностей Arduino UNO. Своими штырьковыми контактами модуль вставляется в основную плату. На PS2 shield Arduino реализованы интерфейсы:

- PS2 receiver interface - интерфейс для подключения приемника PS2 dual shock 2,4G;

- Bluetooth interface - интерфейс для подключения Bluetooth модулей HC-05 или HC-06;

- Интерфейс для подключения серводвигателей;

- Motor shield - колодки для подключения электродвигателей постоянного тока;

- Ultrasonic interface – интерфейс для подключения ультразвуковых датчиков.

Плата дублирует порты Arduino UNO для подключения устройств ввода/вывода информации. Отличительным достоинством в сравнении с Potentiometer control board и Sensor expansion board является наличие гнезда DC 2,1x5,5 для подключения внешнего источника питания и сигнализирующих светодиодов последовательного интерфейса, отладочного светодиода LED13 и светодиода PWR.

В системе управления PS2 SNAR10 модуль PS2 shield Arduino выполняет функцию конвертера сигналов, полученных с приемника PS2 2,4ГГц в язык, понятный для Arduino. В свою очередь, Arduino UNO обрабатывает управляющие сигналы и посылает команды для управления серводвигателями. В комплекте поставляются два электродвигателя постоянного тока, что дает возможность пользователю подключить их к модулю PS2 shield Arduino и установить манипулятор на движущуюся платформу.

Основной базовый код программы уже записан в Arduino UNO. Таким образом, после сборки системы, пользователь может осуществлять контроль манипулятором SNARM SG90 с помощью джойстиков на пульте управления PS2 dual shock. Официальный производитель предоставляет на своем сайте несколько модификаций кодов программы управления, предусматривающих различные конфигурации органов управления: возможность задействовать кнопки на пульте, управление двигателями постоянного тока, конфигурация чувствительности джойстиков и т.д. Рассматривать все существующие коды не имеет смысла, так как они занимают достаточно большой объем.

Достоинства системы управления PS2 SNAR10:

- Возможность аппаратной и программной модификации системы;

- Гибкость системы, возможность калибровки множества параметров управления путем перепрограммирования микроконтроллера;

- Высокая эргономичность пульта управления.

Недостатки:

- Высокая цена. По состоянию на 2018 год цена системы управления составляет 69$.

2. Аппаратная часть

Проведя детальный анализ актуальных известных решений, и подробно рассмотрев их достоинства и недостатки, было решено создать доступную по цене систему дистанционного управления с возможностью настройки параметров работы.

В первую очередь, рассмотрим возможность оптимизации основного устройства – платы управления Arduino. Во всех рассмотренных аналогах используется плата Arduino UNO. По состоянию на 2018 год цена на данное устройство составляет порядка 34$. На рынке продукции Arduino существует ее более дешевый аналог – это плата Arduino nano V3.0 (далее Arduino nano), ее цена составляет около 8$.

Arduino nano, изображенная на рисунке 2.1 — это функциональный аналог Arduino Uno, размещённый на миниатюрной плате. Основные различия заключаются в применении микроконтроллера ATmega328 в корпусе SMD, использовании чипа FTDI FT232RL в качестве интерфейса последовательного порта USART, отсутствии гнезда для подключения внешнего источника питания DC 2,1x5,5 и наличием порта mini-USB для связи с компьютером вместо разъема DB90. Порты ввода/вывода Arduino nano выведены на штырьковые контакты, что позволяет легко устанавливать её на макетную плату. Количество цифровых, аналоговых выводов и портов интерфейсов идентично Arduino UNO [16].

Благодаря своим габаритным размерам и доступной цене, Arduino nano является оптимальным вариантом для использования ее как в качестве микроконтроллера пульта управления-передатчика, так и платы-управления серводвигателями манипулятора-приемника.

Рисунок 2.1 – Arduino nano, внешний вид

Характеристики Arduino nano [15]*:*

- Микроконтроллер: ATmega328;
- Тактовая частота: 16 МГц;
- Напряжение логических уровней: 5 В;
- Входное напряжение питания: 7–12 В;
- Количество портов ввода-вывода общего назначения: 20;
- Максимальный ток с одного порта ввода-вывода: 40 мА;
- Максимальный выходной ток порта 3.3V: 50 мА;
- Максимальный выходной ток порта 5V: 800 мА;
- Количество портов с поддержкой широтно-импульсной модуляции: 6;
- Количество портов, подключённых к АЦП: 6;
- Разрядность АЦП: 10 бит;
- Объем Flash-памяти: 32 КБ;
- Объем EEPROM-памяти: 1 КБ;
- Объем оперативной памяти: 2 КБ;
- Габаритные размеры платы: 18,5×42 мм.

Следующий этап разработки устройства управления манипулятором SNARM SG90 на базе выбранного микроконтроллера Arduino nano – выбор интерфейса между передатчиком и приемником.

Пользователь должен управлять манипулятором дистанционно – это является основным критерием при выборе интерфейса, при этом манипулятор должен находиться в зоне прямой видимости пользователя, т.е. на расстоянии около 5 метров. На сегодняшний день существуют несколько вариантов интерфейсов для Arduino nano, подходящих под эти условия: это передача данных по радиоканалу (как в случае с PS2 SNAR10), по сети Wi-Fi или по Bluetooth, рассмотрим каждый вариант подробнее.

1) Радиоканал на модулях RF 433

Комплект 433Mhz RF Wireless Transmitter Receiver Module состоит из двух модулей: радиопередатчика и приемника, изображенных на рисунке 2.2. Модули настроены на частоту 433,92 МГц и предназначены для создания простого радиоканала в одностороннем режиме, т.е. передача данных происходит без обратной связи. Данный комплект подходит для создания таких устройств, как: беспроводной квартирный звонок, пульт дистанционного управления комнатным освещением, устройство управления электромагнитным замком двери, двигателями штор, жалюзи [17].

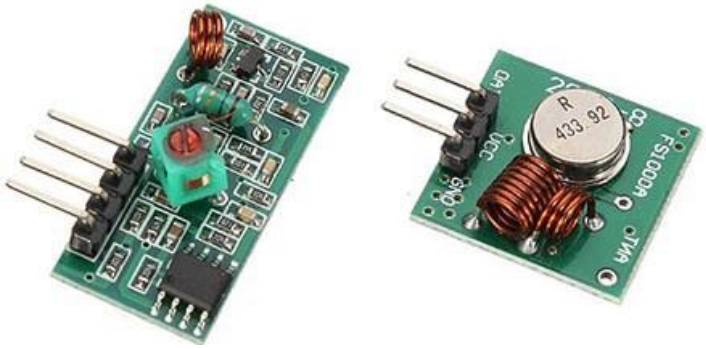


Рисунок 2.2 – Модули радиоприемника (слева) и радиопередатчика (справа) комплекта 433Mhz RF Wireless Transmitter Receiver Module

Описание работы

Устройство передает сигналы с помощью разновидности амплитудной модуляции OOK (On Off Keying). Это альтернатива обычной амплитудной

модуляции ASK (Amplitude Shift Keying) несущей частоты. Отличие между передаваемыми сигналами отображено на рисунке 2.3. Когда на управляющем входе появляется высокий уровень сигнала, т.е. уровень логической 1, передатчик активируется и начинает передавать в эфир сигнал несущей частоты, а низком уровне сигнала передатчик выключается.

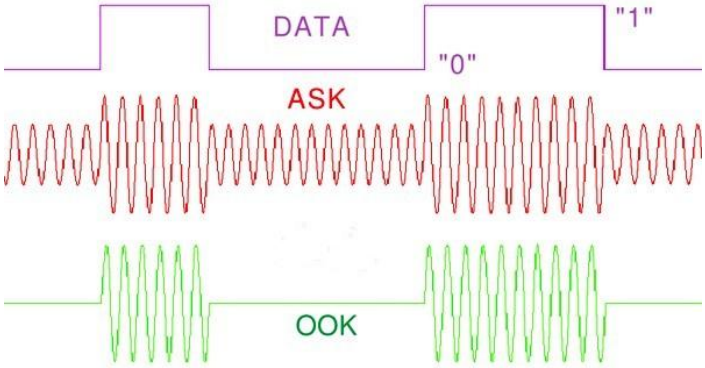


Рисунок 2.3– Зависимости выходных сигналов

Преимуществом применения OOK модуляции можно назвать более низкое энергопотребление в сравнение с ASK модуляцией, но при этом приходится пожертвовать скоростью передачи данных, т.к. затрачивается некоторое время на активацию передатчика при получении логической единицы.

Назначение выводов:

DATA – вход управляющего сигнала;

VCC – вход подключения источника питания;

GND – общий провод.

Характеристики передатчика:

Диапазон напряжений питания: 3,5 – 15 В;

Номинальное напряжение питаний: 5 В

Потребляемый ток: 9 – 40 мА;

Несущая частота: 315 МГц или 433,92 МГц;

Максимальное отклонение частоты: 150 КГц;

Мощность передачи:

- номинальная 10 мВт (при питании 5 В),
- максимальная 25 мВт (на частоте 315 МГц при питании 12 В).

Максимальная скорость передачи данных: 5 Кбит/сек;

Размеры платы: 19 х 19 мм.

Приемник восприимчив к помехам. Для исключения ложного приема требуется цифровое кодирование и декодирование сигналов, содержащее многократные повторы команд. Программная обработка принятых данных на приемной стороне делает цифровой модуль, соединенный с приемником частью приемного устройства. Реализуя с помощью программы фильтрацию ложных сигналов, микроконтроллер использует большой объем передаваемой информации, но этого вполне достаточно для передачи одной команды в 2-3 секунды. При этом значительно расходуется пропускная способность канала, но повышается достоверность распознавания команды.

Характеристики приемника:

Напряжение питания: 4,5 – 5,5 В;

Потребляемый ток: 4 – 5,5 мА;

Частота приема: 315 МГц или 433,92 МГц;

Ширина полосы пропускания: 2 МГц

Чувствительность: 100 dB;

Размеры 30 х 14 х 7 мм.

Назначение выводов:

DATA – выход дешифратора, данный порт продублирован на плате;

VCC – вход подключения источника питания;

GND – общий провод.

Для качественного приема и передачи сигналов желательно оборудовать модули антеннами. Приемник и передатчик 433MHZ рассчитаны на антенны с волновым сопротивлением 50 Ом. Антенны должны устанавливаться как

можно дальше от источников помех и металлических конструкций. При этом источники питания приборов должны иметь хорошую фильтрацию от помех сети 220 В.

Достоинства комплекта 433Mhz RF Wireless Transmitter Receiver Module:

- Низкая стоимость комплекта (примерно 2$);

- Простота реализации радиоканала.

Недостатки:

- Отсутствие обратной связи;

- Низкая скорость передачи данных;

- Низкая помехоустойчивость, при работе манипулятора сервоприводы могут оказывать влияние на работу устройств;

- Частота 433 МГц является достаточно распространенной частотой работы других устройств (радио-брелоки, радиопульты), которые могут вызвать помехи при совместной работе устройств.

2) Wi-Fi интерфейс

Для реализации Wi-Fi интерфейса на платформе Arduino используется Wi-Fi модуль ESP8266, изображенный на рисунке 2.4. Благодаря данному модулю можно легко организовать беспроводное соединение между пультом управления и манипулятором по сети Wi-Fi, так же можно реализовать управление с внешнего компьютера или даже удаленно по сети интернет, но на практике удаленное управление манипулятором является бессмысленным.

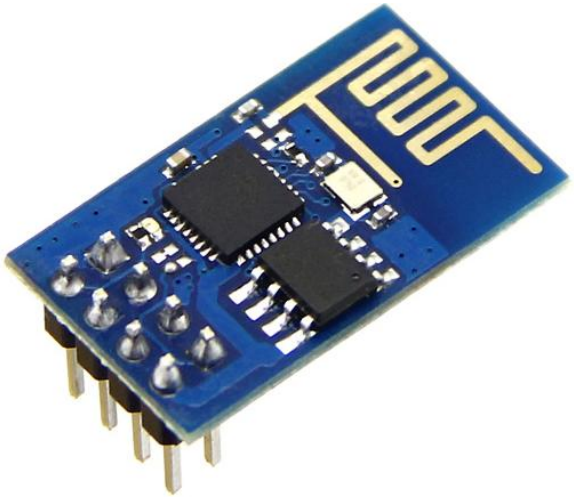


Рисунок 2.4 - Wi-Fi модуль ESP8266

Передача данных между Arduino и ESP8266 осуществляется по интерфейсу последовательного порта UART с помощью AT-команд (от англ. **At**tention - внимание). Посредством этих команд так же выполняется конфигурация модуля, где можно задать различные режимы работы (установить как точку доступа, создать пару с другими устройствами, установить пароль и т.д.) [18].

Параметры ESP8266:

Версия ПО: ESP-01 V090;

Беспроводной интерфейс: Wi-Fi 802.11 b/g/n 2,4 ГГц;

Режимы работы: P2P (клиент), soft-AP (точка доступа);

Максимальная выходная мощность: 89 мВт;

Напряжение питания: 3,3 В;

Максимальный потребляемый ток: 220 мА;

Портов ввода-вывода свободного назначения: 2;

Тактовая частота процессора: 80 МГц;

Объём оперативной памяти: 96 КБ;

Габаритные размеры: 21×13 мм.

Достоинства использования Wi-Fi ESP8266:

- Высокая скорость передачи данных;

- Высокая помехоустойчивость;

- Возможность реализации управления манипулятором с внешнего компьютера;

- Возможность создания защищенного канала связи между приемником-передатчиком.

Недостатки:

- Высокая цена (порядка 8$).

Модуль ESP8266 обладает очень широким набором функций, но для задачи создания системы управления манипулятором эти функции, по сути, являются бессмысленными. И использование данного модуля в качестве беспроводного интерфейса нецелесообразно.

3) Bluetooth интерфейс

Bluetooth — это пожалуй самый распространенный тип связи для коротких дистанций, которым пользуются большинство современных электронных устройств. Телефонные гарнитуры, наушники, клавиатуры и мышки, принтеры и т.д. Главными достоинствами BT можно назвать хорошую устойчивость к широкополосным помехам и простоту реализации. Первое значит, что множество устройств, находящихся в одном месте, могут одновременно передавать данные между собой, не мешая друг другу. Второе же помогло широкому распространению Bluetooth в любительской среде [24].

Самыми доступными на сегодня Bluetooth модулями можно назвать HC-05 и HC-06, изображенные на рисунке 2.5. Эти модули уже упоминались ранее в главе 1,2 как опциональное решение модификации беспроводной связи систем SNARM1900 и SNAM2000.

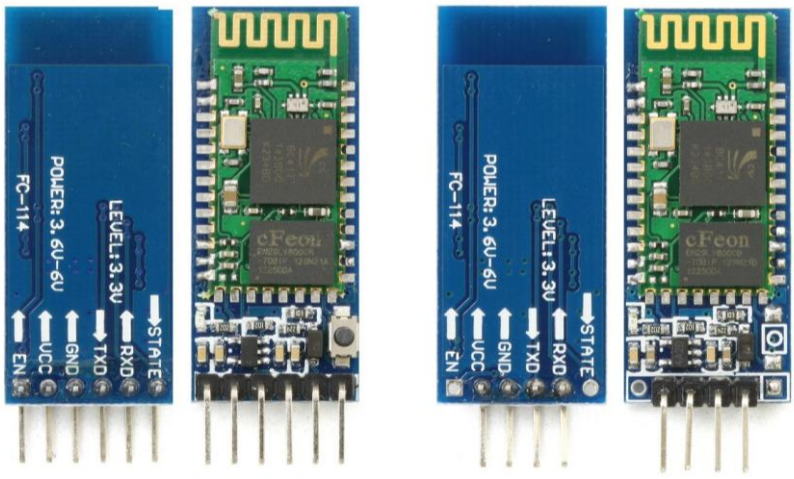


Рисунок 2.5 – Bluetooth модули HC-05 (слева) и HC-06 (справа)

Оба устройства базируются на микроконтроллере CSR BC417, который поддерживает Bluetooth 2.0 с максимальной скоростью 3 Мбит/сек. Главное отличие состоит в том, что HC-06 всегда работает в режиме ведомого устройства (slave). Он не может самостоятельно обнаружить парное устройство и наладить с ним связь, он может лишь подчиниться ведущему устройству (master), в то время как HC-05 может работать как в роли ведущего устройства, так и в режиме ведомого (в зависимости от конфигурации). Как и в случае с модулем ESP8266, интерфейс и конфигурация HC-05 осуществляется посредством AT-команд по последовательному порту UART. Поэтому для создания радиоканала приемник-передатчик лучше использовать именно HC-05.

Назначение выводов Bluetooth модуля:

EN — включение/выключение модуля;

VCC — вывод подключения источника питания 5 В;

GND — общий вывод;

TX, RX — UART интерфейс для общения с контроллером;

STATE — вход управления светодиодом отладки.

При работе с модулем следует учесть тот факт, что для нормальной работы модуля рекомендуется на вход RX подавать сигналы напряжением 3,3 В, а как известно, выходное напряжение портов Arduino составляет 5 В. Поэтому необходима дополнительная схема включения модуля в виде делителя напряжения.

Характеристики HC-05:

- Диапазон частот радиосвязи 2,4–2,48 ГГц;
- Адаптивное переключение канала;
- Мощность передачи 0,25–2,5 мВт;
- Чувствительность –80 dBм;
- Рекомендуемый диапазон 10 м
- Максимальная скорость передачи данных: до 3 Мбит/сек;
- Напряжение питания: 3,3 В;
- Ток потребления: 30-40 мА в режиме поиска устройств, 8 мА в номинальном режиме работы;
- Рабочий диапазон температур –25…75 °C;
- Габаритные размеры 27 x 13 x 2,2 мм;

Достоинства HC-05:

- Доступная цена, стоимость модуля на рынке составляет порядка 3,3$;
- Возможность создания защищенного радиоканала приемник-передатчик;
- Возможность дополнительной конфигурации устройства;
- Высокая скорость передачи данных.

Недостатки:

- Для стабильной работы устройства требуется дополнительная схема включения.

Проанализировав актуальные модули беспроводного интерфейса, и рассмотрев их достоинства и недостатки, был выбран вариант реализации Bluetooth интерфейса посредством модуля HC-05, так как он является наиболее

экономически выгодным, так и подходящим для реализации создания защищенного беспроводного интерфейса приемник-передатчик.

2.1. Разработка схемы электрической принципиальной

Для разработки электрически принципиальной схемы обратимся к структурным схемам устройства и рассмотрим их принципы действия.

Изображенная на рисунке 2.6 а представляет собой структурную схему устройства пульта управления. Поскольку пульт управления дистанционный – его питание осуществляется с аккумулятора (АКБ). Для обеспечения функции зарядки аккумулятора и контроля его состояния заряда в схеме присутствует модуль-плата контроля заряда АКБ.

Большинство аккумуляторов имеют выходное напряжение, равное 3,7 В, и для обеспечения питания платы Arduino nano стабилизированным напряжением 5 В, выход с модуля контроля заряда через выключатель питания подключается к модулю преобразователя напряжения (плата DC-DC преобразователя). DC-DC преобразователь так же обеспечивает выходное напряжение 5 В даже при падении напряжения на выводах АКБ в ходе его разряда.

Основным устройством в пульте управления является плата Arduino nano. Она выполняет обработку аналоговых сигналов с двух джойстиков, подключенных к входам АЦП, преобразовывает их в управляющие сигналы и передает по последовательному порту на Bluetooth модуль HC-05, подключенного к соответствующим контактам Arduino. Модуль HC-05 настроен на режим ведущего устройства и осуществляет передачу управляющих сигналов на плату приемник дистанционно по протоколу Bluetooth 2,0.

Регулировка чувствительности (и соответственно, скорости вращения сервоприводов) джойстиков выполняется посредством вращения ручки, установленной на потенциометре. Выход потенциометра подключен к порту АЦП Arduino nano.

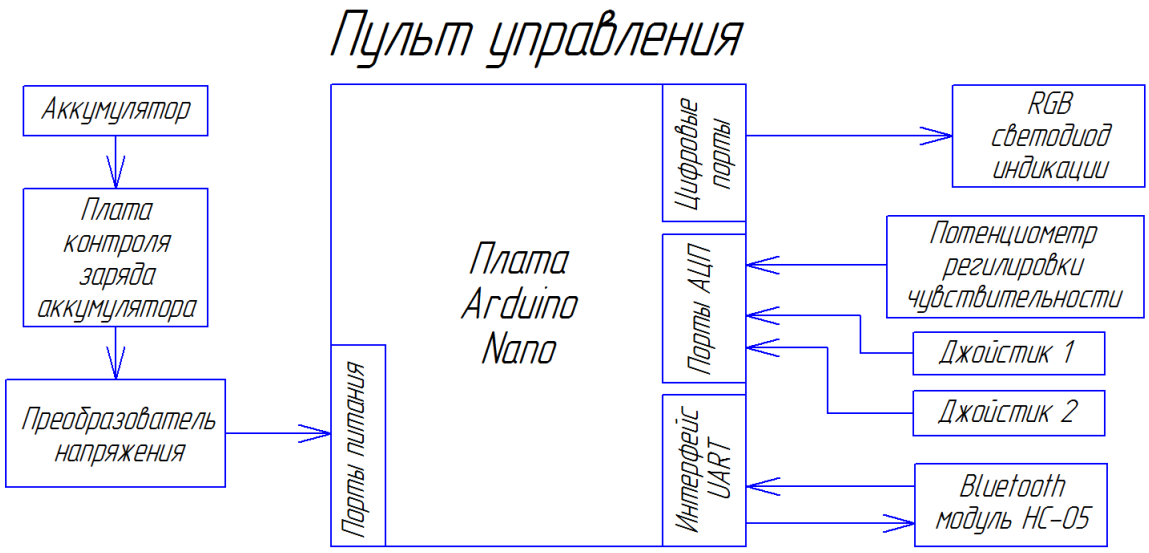


Рисунок 2.6 – Структурная схема пульта управления

Для визуального контроля работы устройства используется RGB светодиод, вывода которого подключены к группе цифровых портов Arduino.

Далее рассмотрим структурную схему платы управления манипулятором, изображенной на рисунке 2.7.

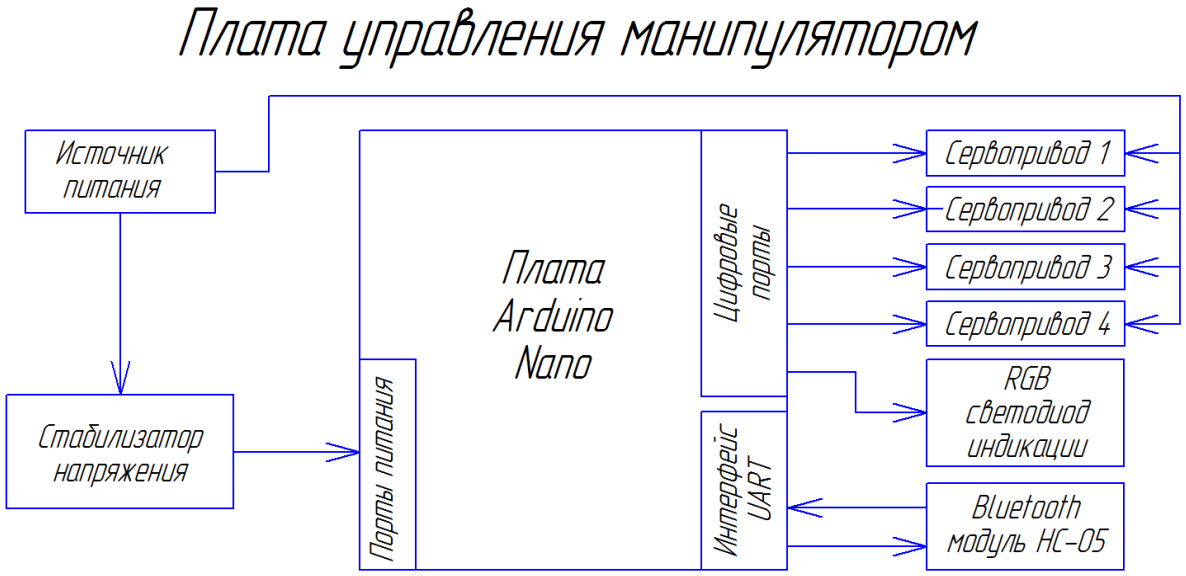


Рисунок 2.7 – Структурная схема платы управления манипулятором

Как и в случае с пультом, платой управления является Arduino nano. Она преобразует данные, поступившие Bluetooth модуля в импульсы управления сервоприводами. Контакты управления сервоприводов подключены к группе цифровых ШИМ портов Arduino.

Для увеличения крутящего момента сервоприводов SG90 необходимо обеспечить их питанием с напряжением 7 В, именно поэтому в схеме применяется источник, обеспечивающий именно такой выходной уровень напряжения. К этому же источнику через стабилизатор напряжения 5 В подключается Arduino nano.

Для визуального контроля работы устройства так же используется RGB светодиод.

Разработка электрических принципиальных схем проводилась с помощью программы Schematic – современного программного инструмента для разработки сложных многоуровневых иерархических принципиальных схем с множеством функций по созданию визуальных и логических связей между выводами компонентов. Программа Schematic входит в состав Dip Trace - это простой в освоении и удобный пакет программ для разработки принципиальных схем и печатных плат.

Исходя из данных структурных схем, были разработаны электрические принципиальные схемы, изображенные на рисунках 2.8 и 2.9.

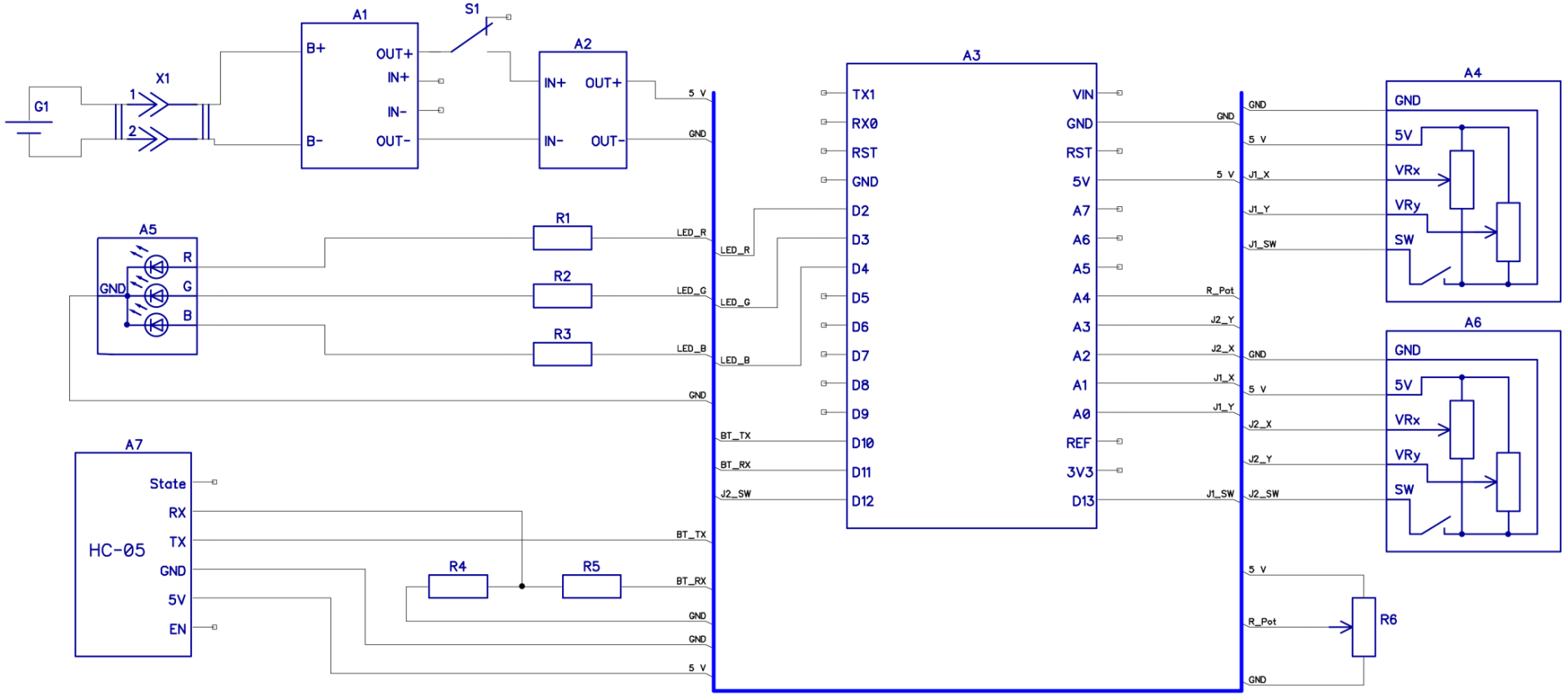


Рисунок 2.8 – Пульт управления, схема электрическая принципиальная

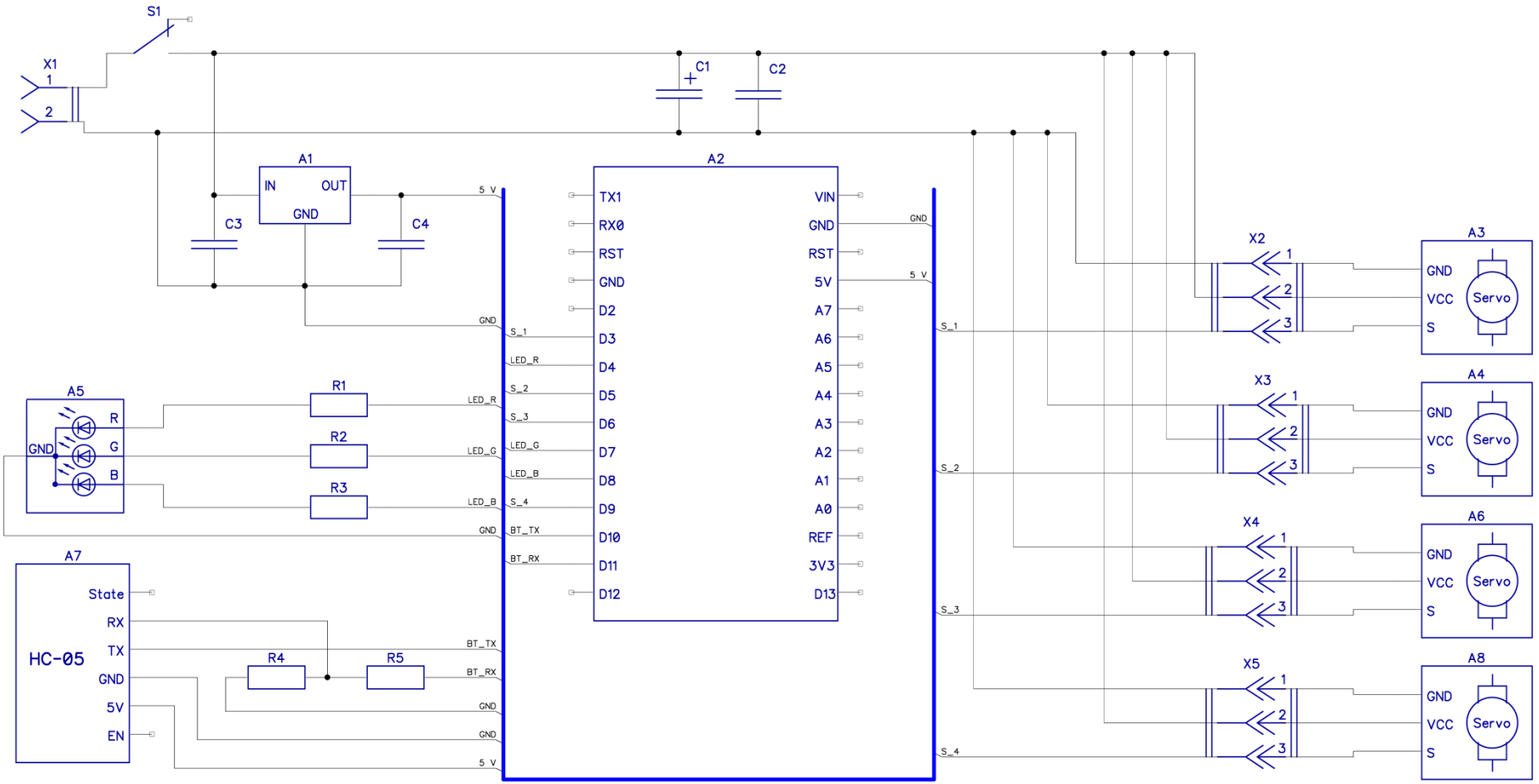


Рисунок 2.9 – Плата управления манипулятором, схема электрическая принципиальная

2.2. Выбор и расчет элементов

В качестве сигнального светодиода был выбран один из самых доступных вариантов на рынке RGB-светодиод BL-L515RGBW-CC, изображенный на рисунке 2.10.

Рисунок 2.10 – RGB светодиод BL-L515RGBW-CC

Светодиод серии L-515 конструктивно выполнен в круглом корпусе диаметром 5мм с линзой сферической формы. Для удобства монтажа в нижней части имеется выступ так называемая "юбка" диаметром 5.9 мм. Особенность этой серии светодиодов заключается в том, что он полноцветный. В корпусе расположено 3 кристалла красный, зеленый и синий, поэтому светодиод имеет 4 вывода, самый длинный является общим катодом [19].

Технические параметры светодиода:

- Цвет свечения: RGB;
- Минимальная сила света: 300 мКд;
- Максимальная сила света: 600 мКд;
- Ток потребления: 20 мА;
- Номинальное напряжение для красного кристалла: 2,1 В;
- Номинальное напряжение для зеленого и синего кристалла: 3,8 В;
- Видимый телесный угол: 30 град;
- Цвет линзы: белый;
- Форма линзы: круглая;

- Размер линзы: 5 мм;
- Максимальное обратное напряжение: 5 В;
- Максимальный импульсный прямой ток: 150 мА;
- Диапазон рабочих температур: минус 40…+85°С.

Поскольку цифровые вывода Arduino nano имеют логический уровень напряжения 5 В, то для исключения выхода RGB светодиода из строя необходимо выбрать токоограничивающий резистор для каждого кристалла светодиода [2]. Расчет сопротивления резистора R_{HL} определяется по следующей формуле:

$$R_{HL} = \frac{V_S - V_{HL}}{I_{HL}} \text{ Ом,} \quad (1)$$

где $V_S -$ напряжение источника питания, В;

$V_{HL} -$ падение напряжения на светодиоде, В;

$I_{HL} -$ номинальный ток потребления светодиода, А.

Вычислив сопротивление резистора, вычисляется его мощность рассеивания P_{HL}:

$$P_{HL} = V_{HL} \cdot I_{HL} \text{ , Вт} \quad (2)$$

Рассчитаем по формуле (1) сопротивления токоограничивающего резистора для красного R_{HLR} кристалла светодиода:

$$R_{HLR} = \frac{5 - 2{,}1}{13 \cdot 10^{-3}} \approx 223 \text{ Ом.} \quad (3)$$

Значение потребляемого тока 13 мА было выбрано исходя из того, что при работе светодиода с потребляемым током 20 мА возможен риск его

быстрого выхода из строя. К тому же, яркость свечения светодиода при данном токе будет практически неотличимой для человеческого глаза [3].

Вычислив сопротивление резистора, найдем его рассеиваемую мощность согласно формуле (2):

$$P_{HLR} = 2{,}1 \cdot 13 \cdot 10^{-3} \approx 0{,}027 \text{ Вт} \approx 27 \text{ мВт}. \quad 4)$$

Согласно расчету (1), выбираем подходящий номинал резистора из стандартного ряда E12 [22], равный 220 Ом. Оптимальным вариантом для уменьшения габаритов будет SMD (англ. Surface Mount Device – устройство поверхностного монтажа)резистор. Корпус SMD резистора выбирается согласно его рассеиваемой мощности, найденной по формуле (2). Таким образом, выбираем SMD резистор в корпусе 0805 с сопротивлением 220 Ом и максимальной мощностью рассеивания 125 мВт. Его технические характеристики представлены ниже:

- Тип корпуса: SMD 0805;
- Номинальное сопротивление: 220 Ом;
- Точность: ±1%;
- Номинальная мощность рассеивания: 125 мВт;
- Максимальное рабочее напряжение: 150 В;
- Рабочая температура: минус 55…+125°С;
- Длина корпуса: 2 мм;
- Ширина корпуса: 1,25 мм.

Аналогично формуле (1) найдем сопротивления токоограничивающих резисторов для зеленого R_{HLG} и синего R_{HLB} кристаллов светодиода. Поскольку уровень падения напряжения на них один и тот же, расчет будет идентичным:

$$R_{HLG,HLB} = \frac{(5 - 3{,}8_)}{13 \cdot 10^{-3}} = 92 \text{ Ом}. \quad 5)$$

Исходя из примера (2), их рассеиваемая мощность P_{HLG} и P_{HLB} будет:

$$P_{HLG,HLB} = 3{,}8 \cdot 13 \cdot 10^{-3} \approx 0{,}049 \text{ Вт} \approx 49 \text{ мВт}. \quad 6)$$

Ближайшим сопротивлением ряда E12 [22] оказывается сопротивление, равное 100 Ом, для унификации схемы выбираем резисторы в корпусе SMD 0805, имеющим следующие параметры:

- Тип корпуса: SMD 0805;
- Номинальное сопротивление: 100 Ом;
- Точность: ±1%;
- Номинальная мощность рассеивания: 125 мВт;
- Максимальное рабочее напряжение: 150 В;
- Рабочая температура: минус 55…+125°С;
- Длина корпуса: 2 мм;
- Ширина корпуса: 1,25 мм.

Сопротивления выбранных резисторов имеют незначительную разницу с полученными значениями (3) и (5). Данная разница никак не скажется на работе светодиода, потому что, как говорилось выше, человеческий глаз не воспримет разницу в яркости свечения светодиода.

В качестве органа управления пульта был выбран джойстик PS2 rocker, используемый в системе управления SNAM2000. Внешний вид джойстика изображен на рисунке 2.11.

Рисунок 2.11– PS2 rocker, внешний вид

Джойстик имеет 5 выводов:

- GND – общий вывод;

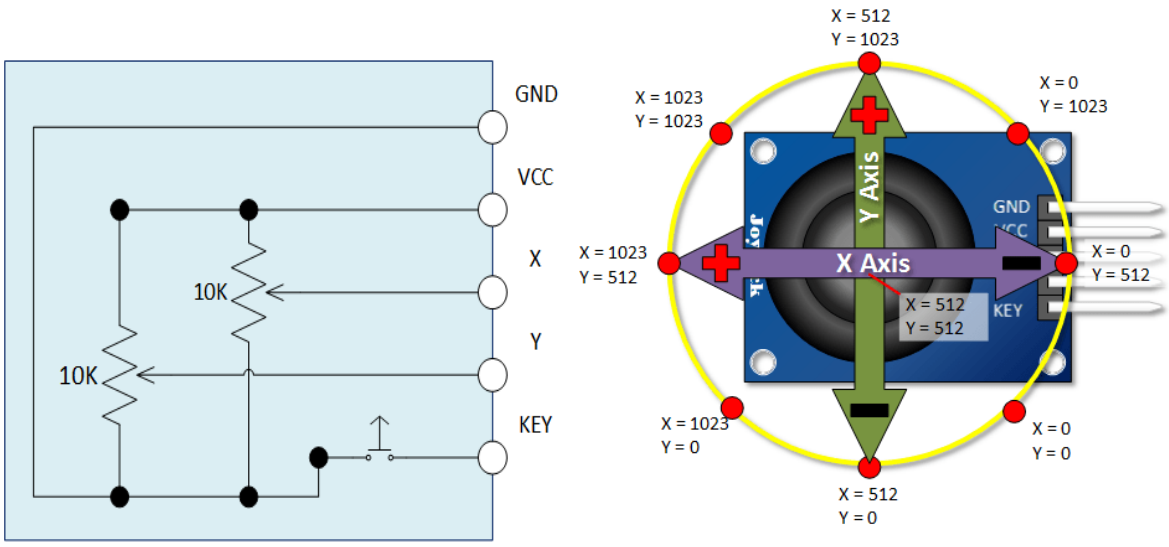
- +5V – вход для подключения питания с напряжением +5 В;

- VRX – аналоговый выход потенциометра горизонтальной оси отклонения;

- VRY – аналоговый выход потенциометра вертикальной оси отклонения;

- SW – цифровой выход сигнала состояния кнопки.

Структурная схема PS2 rocker и диаграмма выходных значений АЦП при положении рукоятки в крайних точках приведены на рисунке 2.12 а и б соответственно.



а) структурная схема джойстика, б) диаграмма выходных значений АЦП

Рисунок 2.12 – Схемы работы PS2 rocker

Параметры PS2 rocker [14]:

- Напряжение питания: 5 В;
- Номинальный ток потребления: 1,1 мА;
- Номинальное сопротивление потенциометров осей: 10 КОм;
- Габаритные размеры: 34x21x30 мм.

Для выбора переменного резистора, с помощью которого будет осуществляться регулировка чувствительности джойстиков, основным фактором является его эргономичность, а именно – ручка резистора не должна мешать пользователю осуществлять контроль манипулятором. Решениями этого критерия могут быть как удобное для пользователя расположение этой ручки на плате, так и использование малогабаритного резистора. Оптимальным решением в этом случае будет применение подстроечного резистора модели RM-065, изображенного на рисунке 2.13.

Рисунок 2.13– Резистор подстроечный RM-065

Резистор RM-065 имеет малые габаритные размеры и утопленную в корпус ручку регулирования. Данное решение обусловлено тем, чтобы после регулировки и выбора оптимальной скорости движения сервоприводов, пользователь не смог случайным образом изменить ее во время управления манипулятором. К достоинствам RM-065 можно отнести: легкий монтаж на печатную плату благодаря специализированному конструктиву корпуса и выводов, высокую надежность подвижных контактов и высокую рассеиваемую мощность [21]. Номинальное сопротивление было выбрано 10 КОм, так как данный номинал является одним из самых распространенных при работе с платами Arduino.

Технические параметры RM-065:

- Тип резистора: подстроечный;
- Тип проводника: металлокерамический;
- Номинальное сопротивление: 10 КОм;
- Функциональная характеристика: А;
- Механический угол поворота ручки: 210 градусов.

На сайте с инструкциями по работе с модулем HC-05 [25] рекомендуется использовать делитель напряжения для стабильной работы интерфейса Bluetooth. Схема включения делителя изображена на рисунке 214.

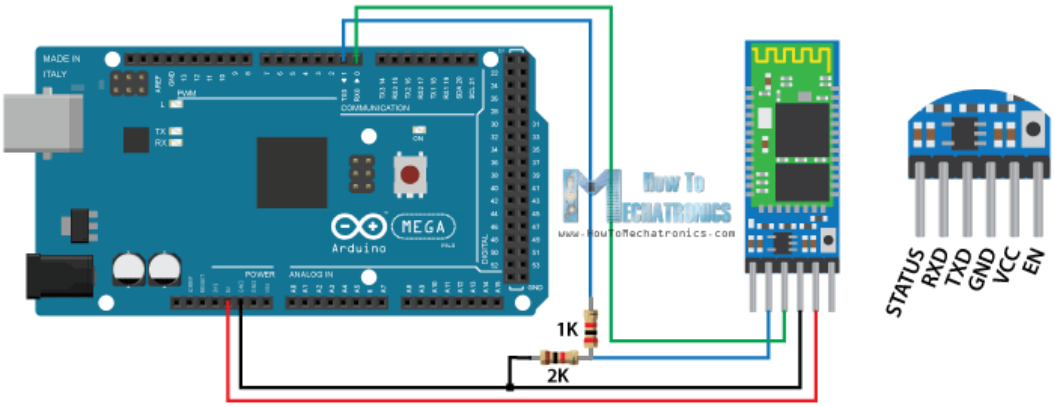


Рисунок 2.14– Рекомендуемая схема включения делителя напряжения

Следует обратить внимание на то, что наличие делителя напряжения требуется только для понижения логического уровня сигналов с 5 В до 3,3 В между выводом TX Arduino и входом RX модуля HC-05. В то время как выход TX модуля подключается к порту RX Arduino без применения делителя, так как выдаваемого напряжения модуля 3,3 В вполне достаточно для восприятия Arduino как уровня логической единицы.

Предлагаемые сопротивления резисторов: 1 КОм для верхнего плеча делителя и 2 КОм для нижнего плеча делителя. Для выбора резисторов для начала рассмотрим типовую схему делителя напряжения [23], изображенную на рисунке 2.15.

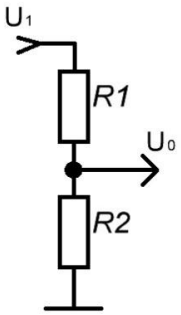


Рисунок 2.15 – Схема включения делителя напряжения

Для расчета рассеиваемой мощности резистора P_{R1} воспользуемся формулой:

$$P_{R1} = \frac{U_1 - U_0\ ^2}{R_1}, \text{Вт} \quad 7)$$

где $U_1 -$ входное напряжение источника питания, В;

$U_0 -$ требуемое выходное напряжение делителя, В;

$R_1 -$ сопротивление резистора верхнего плеча.

Тогда получим:

$$P_{R1} = \frac{(5 - 3{,}3)^2}{1000} \approx 0{,}003 \text{ Вт.} \quad 8)$$

Выбираем из стандартного ряда E24 [22] резистор С1-4, имеющий следующие технические параметры:

- Номинальное сопротивление: 1 КОм;
- Точность: ±5%;
- Номинальная рассеиваемая мощность: 0,125 Вт;
- Максимальное рабочее напряжение: 250 В;
- Диапазон рабочих температур: минус 55…+125°С;
- Длина корпуса: 6,3 мм;
- Ширина корпуса: 2,3 мм.

Рассеиваемая мощность резистора нижнего плеча P_{R2} определяется следующим образом:

$$P_{R2} = \frac{U_0\ ^2}{R_2}, \text{Вт} \quad 9)$$

где U_0 – выходное напряжение с делителя, В.

Рассеиваемая мощность равна:

$$P_{R2} = \frac{3{,}3^2}{2000} \approx 0{,}005 \text{ Вт.} \quad 10)$$

Выбираем из стандартного ряда E24 [22] резистор С1-4, имеющий следующие технические параметры:

- Номинальное сопротивление: 2 КОм;
- Точность: ±5%;
- Номинальная рассеиваемая мощность: 0,125 Вт;
- Максимальное рабочее напряжение: 250 В;
- Диапазон рабочих температур: минус 55…+125°С;
- Длина корпуса: 6,3 мм;
- Ширина корпуса: 2,3 мм.

Поскольку внешний источник питания имеет выходное напряжение, равное 7 В, то для того, чтобы избежать выход из строя платы Arduino nano и модуля HC-05, необходимо применить стабилизатор напряжения 5 В. Поскольку максимальный ток потребления платы управления манипулятором не превышает 1 А, выберем доступный стабилизатор напряжения KIA 7805A, имеющий следующие характеристики:

- Максимальное входное напряжение: 36 В;
- Максимальный выходной ток: 1,5 А;
- Выходное напряжение: 4,8…5,2 В;
- Диапазон рабочих температур: минус 40…+125°С;
- Габаритные размеры: 14 х 9 х 4 мм.

В предоставляемой заводом изготовителем технической документации [26] рекомендуется подключение керамических конденсаторов емкостью 0,33 и 0,1 мкФ, согласно изображенной на рисунке 2.16 схеме:

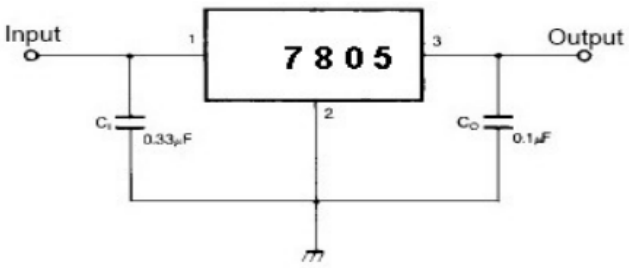


Рисунок 2.16 – Схема включения стабилизатора KIA7805

Согласно рекомендуемым номиналам емкости были выбраны керамические многослойные конденсаторы типа К10-17Б, данный тип конденсаторов имеет следующие технические характеристики:

- Тип: К10-17Б;
- Рабочее напряжение: 50 В;
- Точность: 10%;
- Рабочая температура: минус 55…+125°С;
- Длина корпуса: 4,2 мм;
- Ширина корпуса: 3,2 мм.

В качестве источника питания для пульта управления был выбран литий-ионный аккумулятор (Li-ion) типоразмера 18650 фирмы Bailong. Данное решение обосновано высокой емкостью аккумулятора, что позволяет обеспечить продолжительную работу пульта управления и доступной ценой (порядка 3,5 $). Внешний вид аккумулятора изображен на рисунке 2.17.

Рисунок 2.17 – Аккумулятор Bailong 18650

Параметры и характеристики аккумулятора:

- Номинальное напряжение - 3,7 Вольта;
- Ёмкость аккумулятора – 4200 мА/час;
- Встроенная защита от перезаряда – есть;
- Тип аккумулятора - литий-ионный;
- Типоразмер – 18650.

Для обеспечения функции зарядки аккумулятора используется модуль зарядки литий-ионных аккумуляторов компании NanJing Top Power ASIC Corp, изображенный на рисунке 2.18.

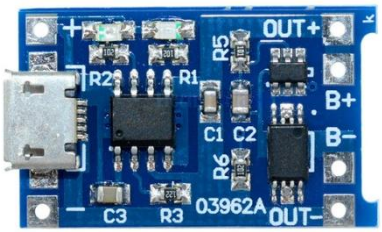


Рисунок 2.18 – Плата заряда литий-ионных аккумуляторов

Данная плата является готовым изделием, обеспечивающим линейный заряд одноэлементных литий-ионных аккумуляторов по принципу постоянное напряжение/постоянный ток [29]. Схема устройства приведена на рисунке 2.19.

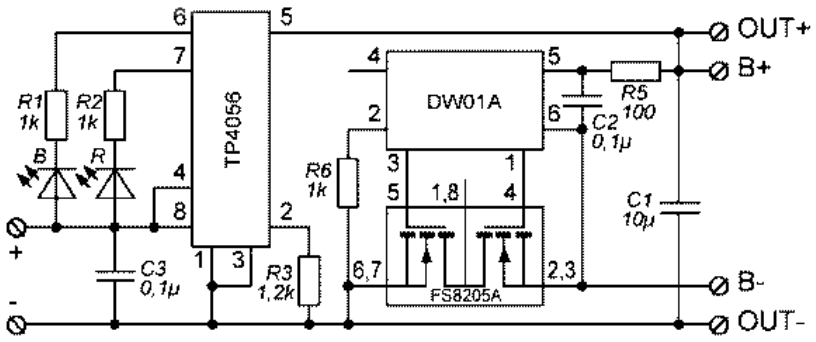


Рисунок 2.19 – Схема платы заряда на TP4056

Основным устройством в модуле является микросхема TP4056, представляющая собой контроллер зарядки литий-ионных аккумуляторов током до 1 А. Величина тока заряда определяется сопротивлением резисторора R3. Статус заряда аккумулятора определяется с помощью светодиодов. Красный светодиод сигнализирует о том, что аккумулятор заряжается, зеленый – об окончании процесса зарядки (напряжение окончания заряда аккумулятора составляет 4,2 В).

На плате присутствуют следующие группы выводов:

- BAT – вход для подключения аккумулятора;

- OUT – выход подключения нагрузки;

- IN – вход для подключения альтернативного источника питания (если не используется Micro-USB разъем).

Параметры и характеристики модуля:

- Поддерживаемый тип аккумуляторов: литий-ионные;

- Тип разъема для подключения зарядного устройства: Micro-USB, в случае использования альтернативного источника питания на плате присутствует отдельная группа контактов;

- Номинальный ток заряда: 1 А;

- Габаритные размеры: 19 х 27 мм;

- Вес: 1,9 г.

Поскольку выходное напряжение литий-ионного аккумулятора составляет около 3,7 В, то для обеспечения стабилизированным напряжением 5 В Arduino nano и Bluetooth модуля HC-05 на пульте управления в схеме необходим преобразователь напряжения. Как и в случае с платой заряда TP4056, было решено использовать готовый модуль повышающего DC-DC преобразователя MT3608. Зачастую на рынке этот модуль предлагается вместе с TP4056. Внешний вид MT3608 изображен на рисунке 2.20.

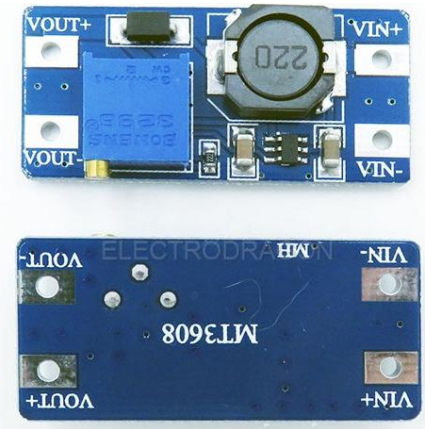


Рисунок 2.20 – Модуль повышающего преобразователя напряжения MT3608

Благодаря своей доступной цене и простотой в использовании, MT3608 получил широкое распространение в среде радиолюбителей. Преобразователь в паре с TP4056 создать готовую систему автономного питания с необходимым выходным напряжением [30]. Группа контактов VIN используется для подключения источника постоянного напряжения, напряжение которого необходимо повысить, и VOUT для снятия преобразованного напряжения. На плате имеется подстроечный резистор, вращая ручку которого регулируется величина выходного напряжения.

Параметры и характеристики MT3608:

- Диапазон входного напряжения: 2…24 В;
- Максимальное выходное напряжение: 28 В;
- Максимальный выходной ток: 2 А;
- КПД: ≤ 93%;
- Габаритные размеры: 36x17x14 мм;
- Вес: 1,4 г.

2.3 Разработка печатной платы

После построения схемы следует этап преобразования в плату. В окне программы Schematic запустив команду «Преобразовать в плату», DipTrace автоматически запустит программу PCB Layout – компонент пакета DipTrace, позволяющий проектировать печатные платы. Скриншот работы с программой приведен на рисунке 2.21

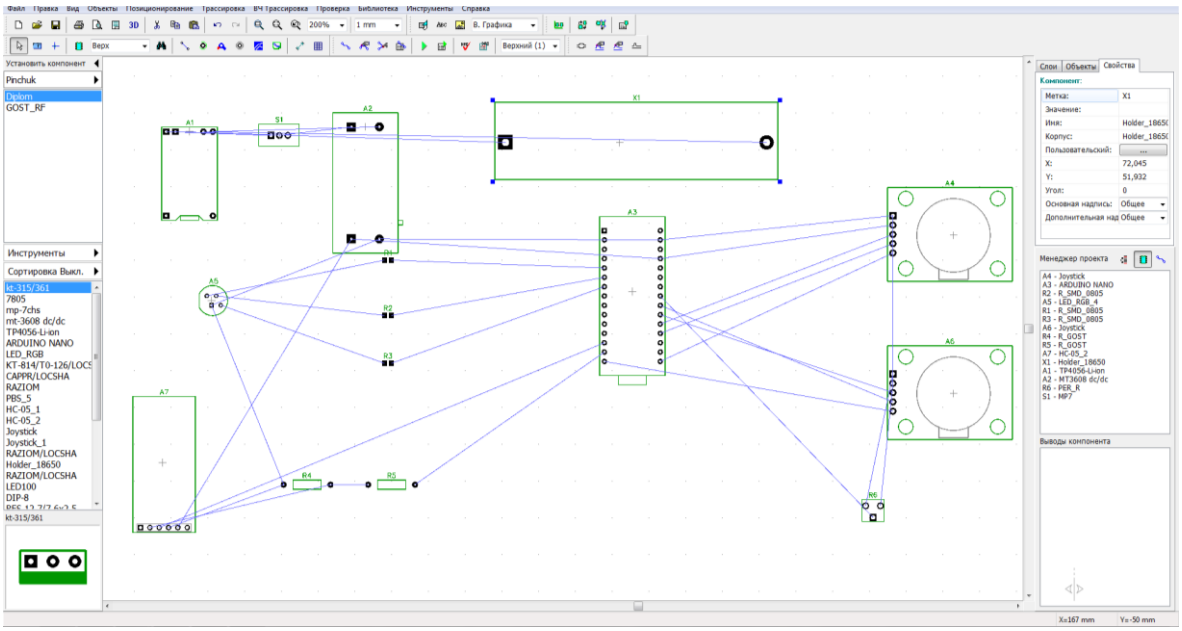


Рисунок 2.21 – Dip Trace PCB Layout, главное окно программы

В качестве материала печатной платы был выбран стеклотекстолит односторонний с толщиной медного слоя 35 мкм и габаритными размерами 150х100мм. Для разработки печатной платы пульта управления учитывался эргономический аспект. Габариты платы подобраны так, чтобы пользователю было удобно держать устройство.

После обозначения границ печатной платы необходимо скомпоновать все компоненты, в программе предусмотрена функция автопозиционирования, которая позволяет расставить все элементы так, чтобы обеспечить оптимальную трассировку платы. Для печатной платы пульта управления

данный вариант не подходит. В процессе ручной компоновки все элементы были расставлены максимально удобно для пользователя: джойстики расположены по краям, светодиод индикации находится сверху для удобного визуального контроля, отсек для аккумулятора установлен снизу для обеспечения оптимального весового баланса. Компоновка элементов изображена на рисунке 2.22.

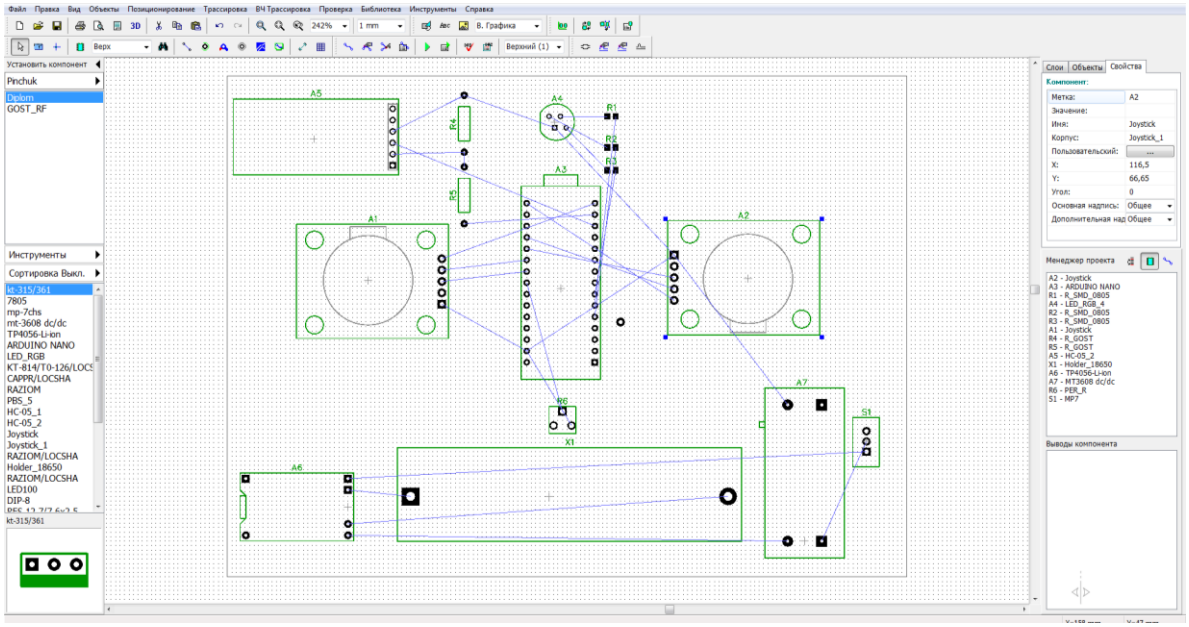


Рисунок 2.22 – Компоновка элементов печатной платы пульта управления

Ширина питающих дорожек была определена с помощью онлайн калькулятора [27]. Скриншот работы с программой с входными данными приведен на рисунке 2.23.

Входные данные:

Ток	0.5	Ампер
Толщина слоя фольги	35	мкм

Дополнительные входные данные:

Повышение температуры	1	Градусы C
Температура окружающей среды	25	Градусы C
Длина дорожки	115	мм

Результаты для внутренних слоёв:

Рекомендуемая ширина дорожки	1.22	мм
Сопротивление	0.0462	Ом
Падение напряжения	0.0231	Вольт
Потери мощности	0.0115	Вт

Результаты для внешних слоёв(на воздухе):

Рекомендуемая ширина дорожки	0.467	мм
Сопротивление	0.120	Ом
Падение напряжения	0.0600	Вольт
Потери мощности	0.0300	Вт

Рисунок 2.23 – Расчет ширины дорожек печатной платы пульта управления

Для трассировки был выбран режим автотрассировщика Grid Router. Полученная печатная плата изображена на рисунке 2.24.

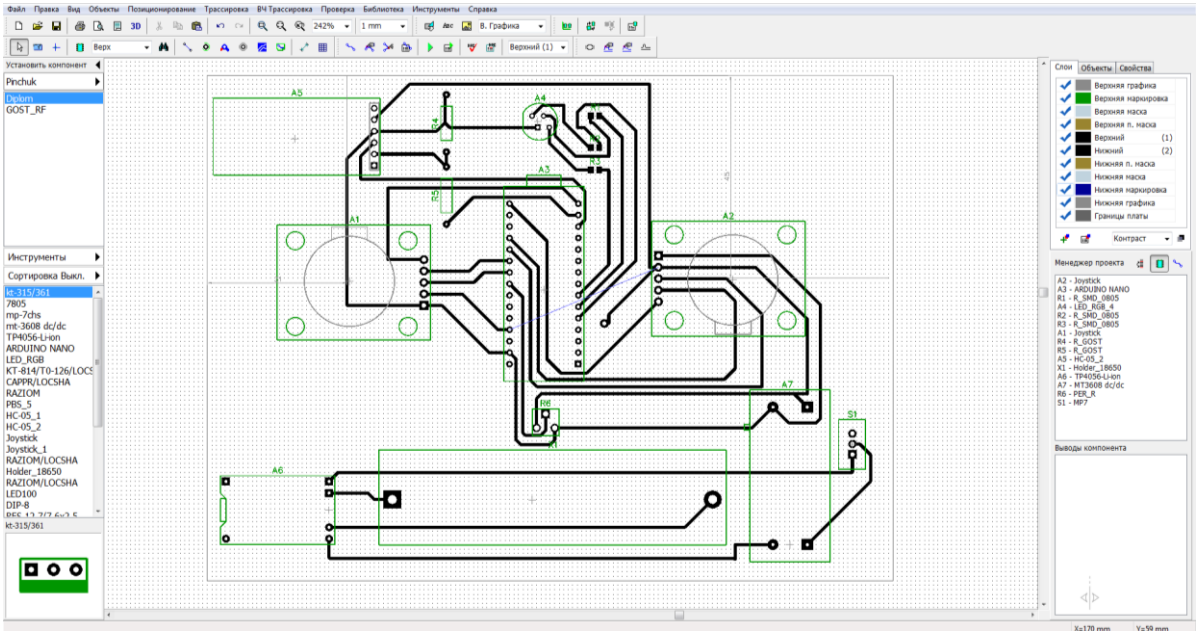


Рисунок 2.24 – Печатная плата пульта управления

Аналогичным образом была разработана печатная плата устройства управления манипулятором, изображенная на рисунке 2.25. Параметры платы приведены на рисунке 2.26.

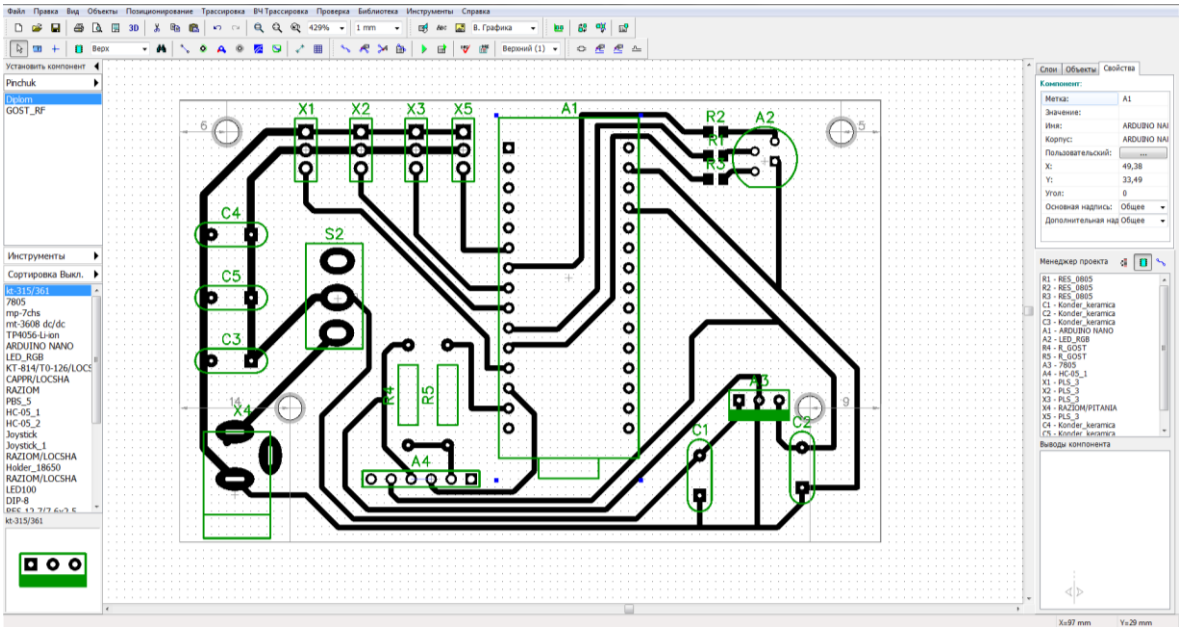


Рисунок 2.25 – Печатная плата устройства управления манипулятором

Входные данные:

Ток	1	Ампер
Толщина слоя фольги	35	мкм

Дополнительные входные данные:

Повышение температуры	1	Градусы C
Температура окружающей среды	25	Градусы C
Длина дорожки	115	мм

Результаты для внутренних слоёв:

Рекомендуемая ширина дорожки	3.16	мм
Сопротивление	0.0177	Ом
Падение напряжения	0.0177	Вольт
Потери мощности	0.0177	Вт

Результаты для внешних слоёв(на воздухе):

Рекомендуемая ширина дорожки	1.22	мм
Сопротивление	0.0462	Ом
Падение напряжения	0.0462	Вольт
Потери мощности	0.0462	Вт

Рисунок 2.26 – Параметры печатной платы устройства управления манипулятором

3. Программная часть

3.1 Разработка алгоритма работы

Для написания программы для начала необходимо разработать алгоритм работы, опираясь на который можно четко составить последовательность действий выполняемой работы. Разработка алгоритма проводилась в бесплатной среде Studlab, расположенной в бесплатном доступе на сайте Studlab.com [28]. Скриншот работы с программой приведен на рисунке 3.1.

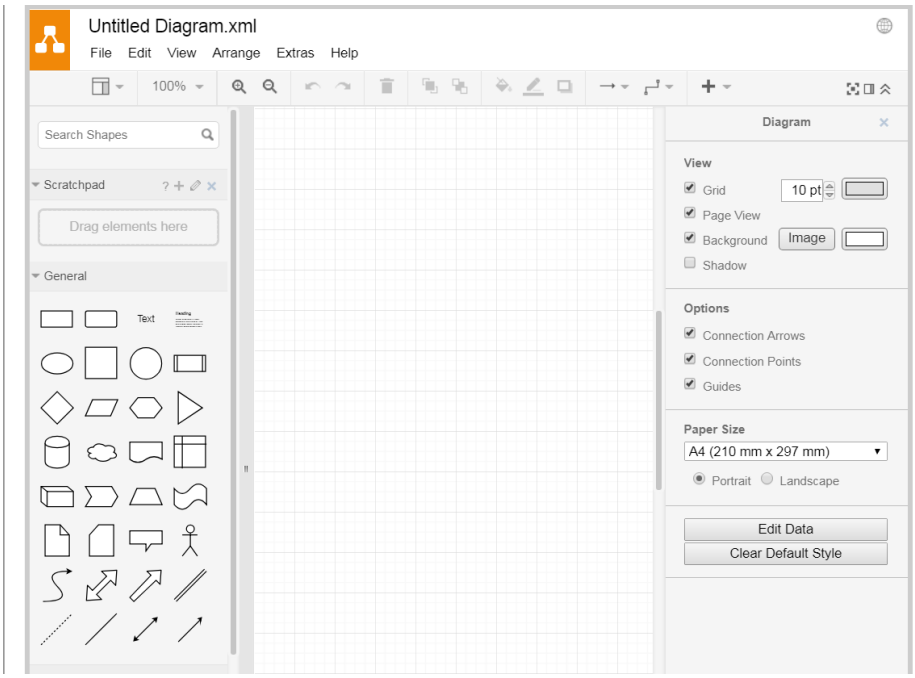


Рисунок 3.1 – Работа с программой Studlab

В ходе работы были разработаны алгоритмы работы для пульта управления и для платы управления манипулятором. Алгоритмы изображены на рисунках 3.2 и 3.3.

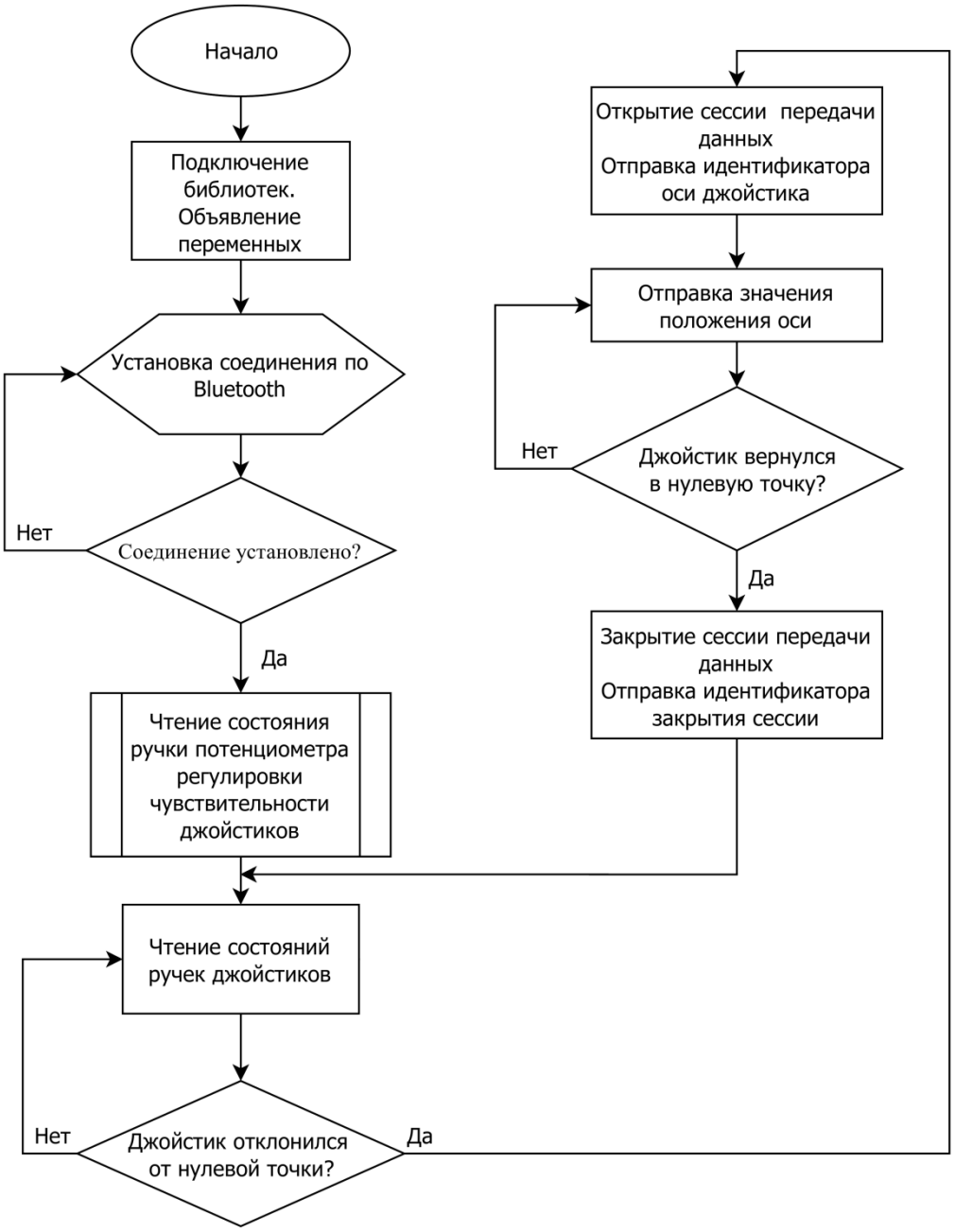


Рисунок 3.2 – Алгоритм работы программы пульта управления манипулятором

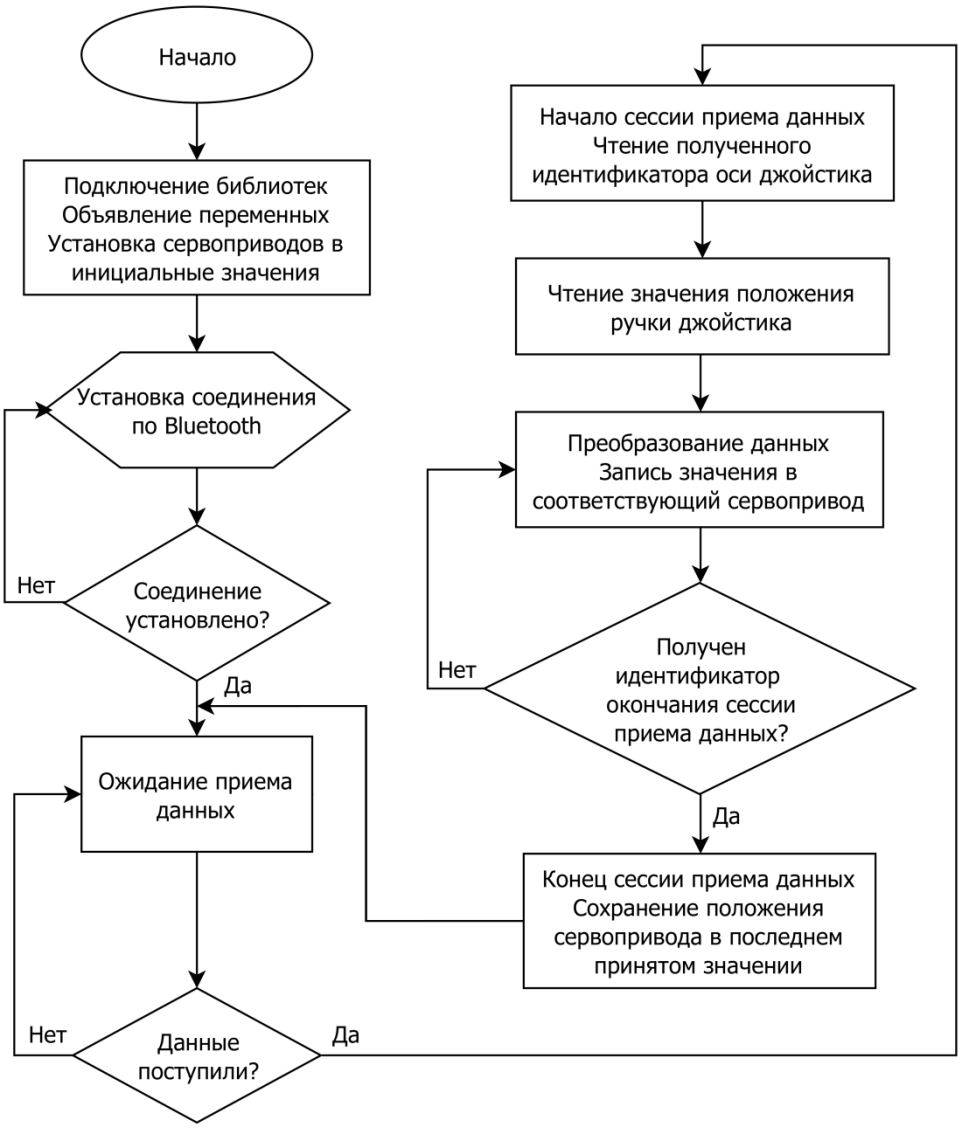


Рисунок 3.3 – Алгоритм работы платы управления манипулятором

3.2 Разработка программы управления манипулятором

Разработка программ управления для пульта и для платы управления проводилась в среде Arduino IDE. Главное окно программы и интерфейс изображен на рисунке 3.4. Код разработанной программы пульта управления с комментариями и пояснениями указан в приложении А. Код программы платы управления манипулятором приведен в приложении Б

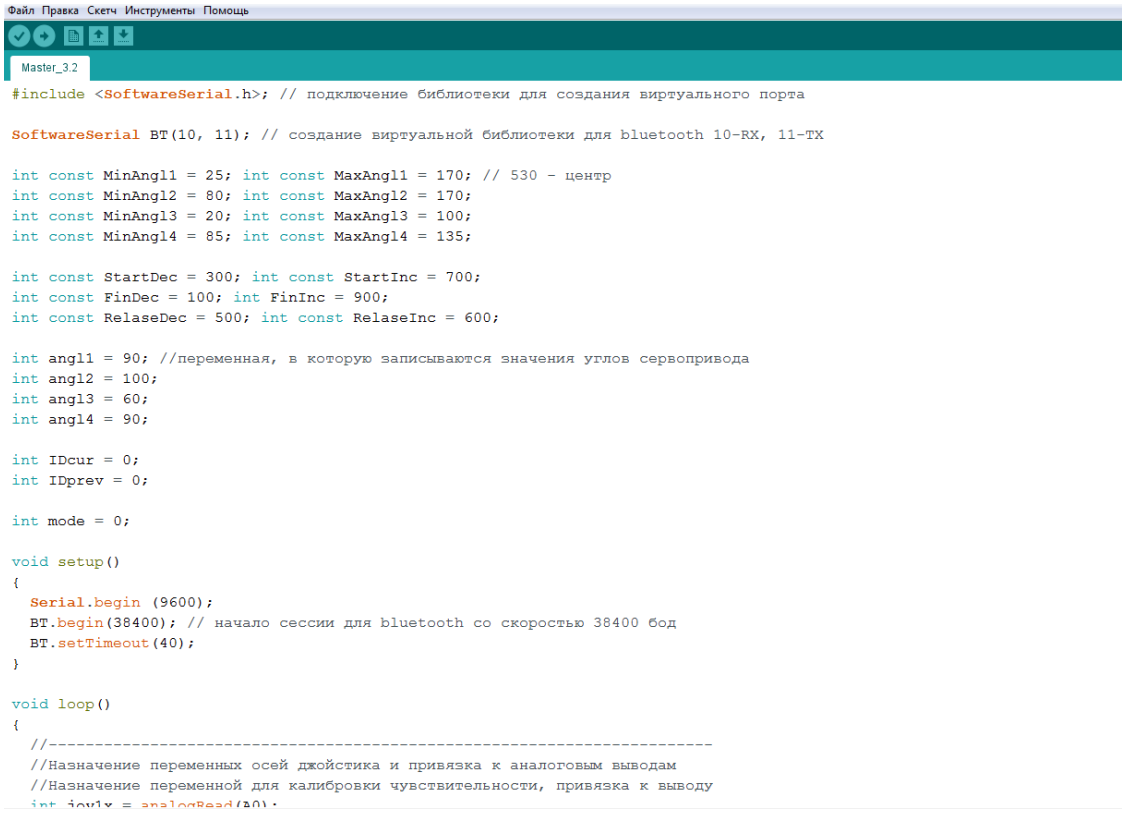


Рисунок 3.4 – Работа в программе Arduino IDE

4. Изготовление устройства

4.1 Сборка манипулятора

Сборка манипулятора SNARM SG90 осуществлялась согласно инструкции, взятой с официального сайта производителя [9]. Некоторые этапы сборки изображены на рисунках 4.1 – 4.6.

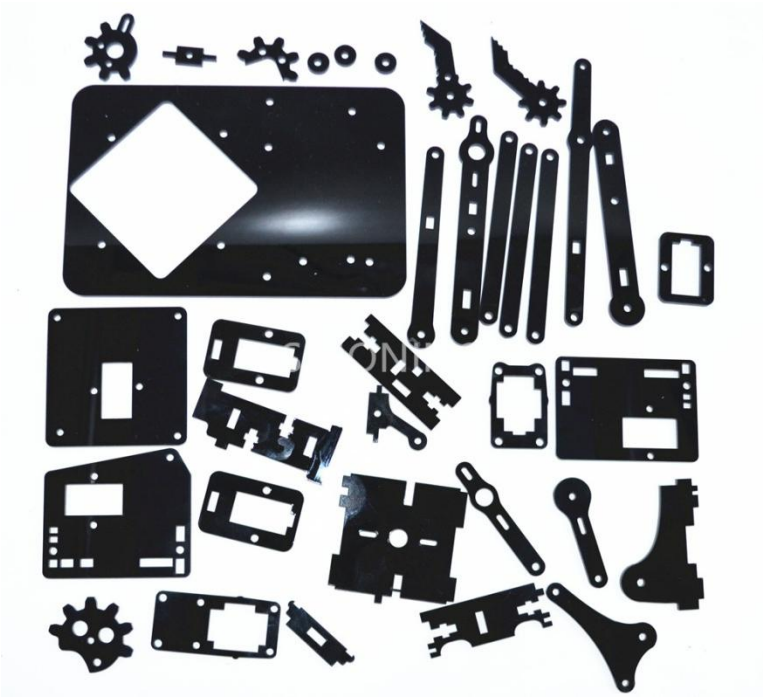


Рисунок 4.1 – Комплект деталей для сборки манипулятора

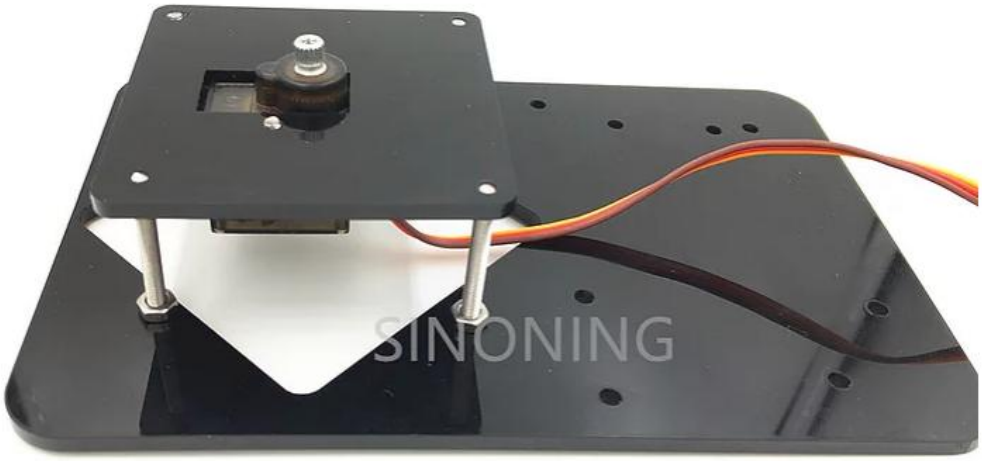


Рисунок 4.2 – Сборка платформы манипулятора

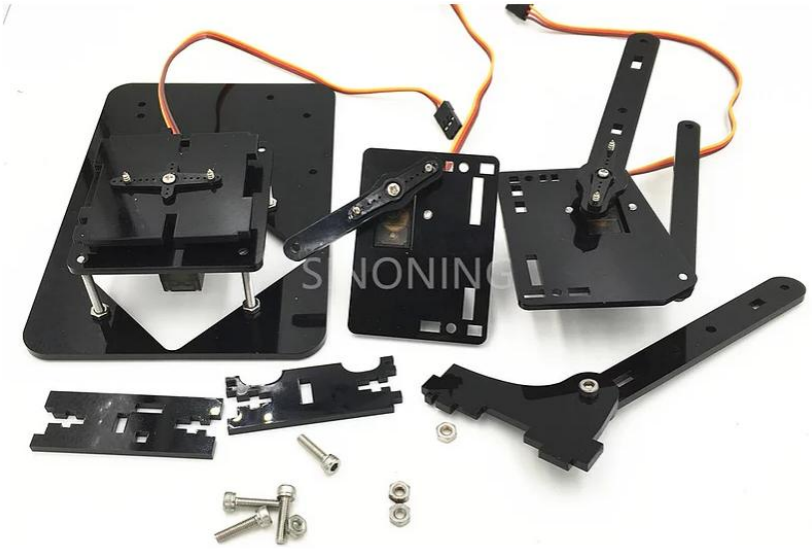


Рисунок 4.3 – Сборка каркаса руки манипулятора

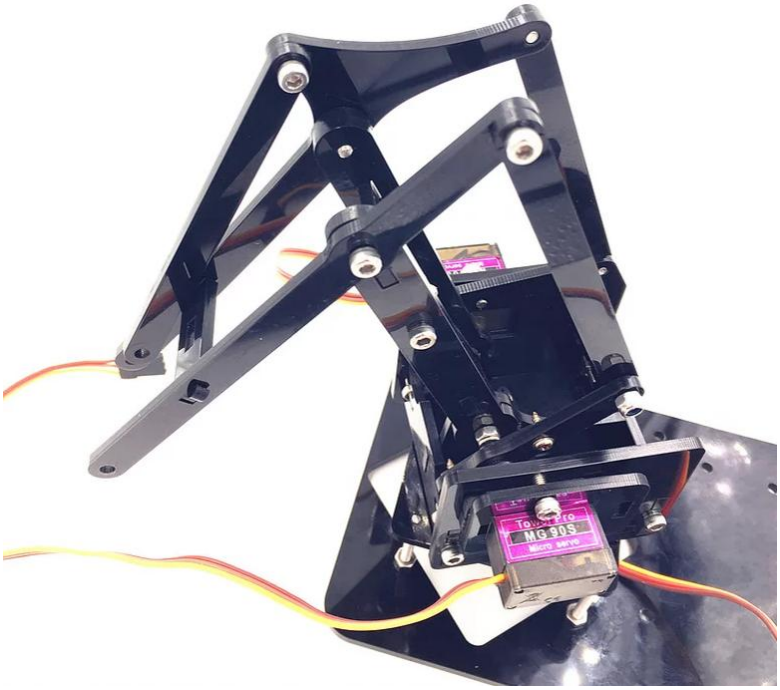


Рисунок 4.4 – Установка руки манипулятора на платформу

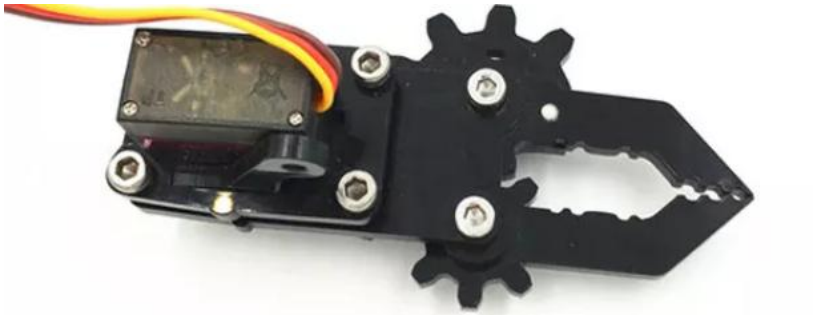


Рисунок 4.5 – Сборка рабочего органа управления

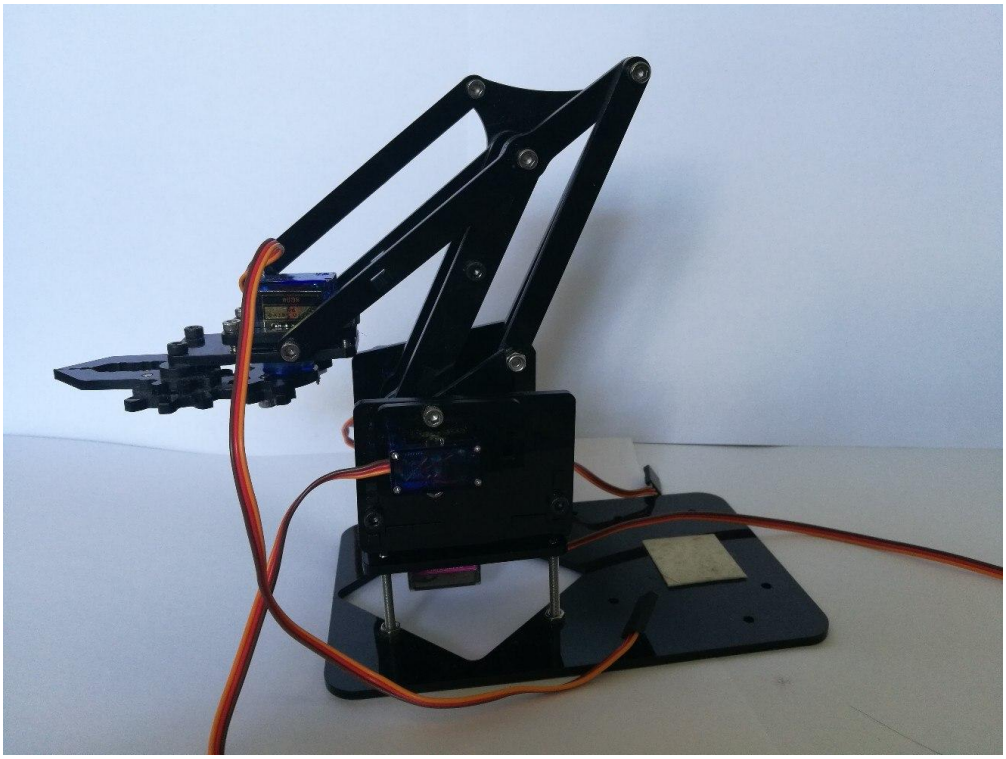


Рисунок 4.6 – Манипулятор SNARM SG90 в сборе

4.2 Изготовление схемотехнической части устройства

Для создания защищенного радиоканала необходимо выполнить конфигурацию Bluetooth модулей HC-05. В состоянии поставки модуль сконфигурирован в роли ведомого устройства. Для функции управления роботом это недопустимо, так как модуль будет подключаться к любому устройству, пытающимся установить с ним связь, последствия при этом могут быть непредсказуемыми.

Конфигурация осуществляется посредством отправке AT команд с персонального компьютера по шине UART. AT-команды (англ. Attention — внимание) — набор команд, разработанных в 1977 году компанией Hayes для собственной разработки модема «Smartmodem 300 baud». Набор команд состоит из серий коротких текстовых строк, которые объединяют вместе, чтобы сформировать полные команды операций. Команды должны быть записаны в специфической форме. Каждая команда всегда начинается буквами AT, далее идет требуемая команда и завершение команды - это символы перевода строки «\r\n». Команды воспринимаются устройством только тогда, когда оно находится в «командном режиме».

Для подключения HC-05 к шине UART используется конвертер USB-UART TTL, изображенный на рисунке 4.7.

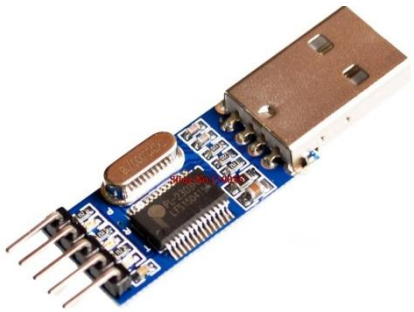


Рисунок 4.7 – Конвертер USB-UART TTL

Краткие характеристики конвертера:

- Скорость обмена данными по UART: 300Бит/сек - 1Мбит/сек;
- поддержка USB 2.0 12Мбит/сек;
- встроенный стабилизатор питания 3.3В 100мА.

Схема подключения конвертера USB-UART TTL к HC-05 приведена на рисунке 4.8.

Рисунок 4.8 – Схема подключения конвертера USB-UART TTL к Bluetooth модулю HC-05

Для перевода Bluetooth модуля в режим АТ-команд необходимо на контакт KEY подключить питание 3,3 В. Переход в командный режим сигнализируется миганием светодиода на плате HC-05 с частотой 2 Гц.

Далее настройка производится с помощью программы-терминала Termite 3.0, расположенная в свободном доступе в сети интернет. Скриншот работы с программой приведен на рисунке 4.9.

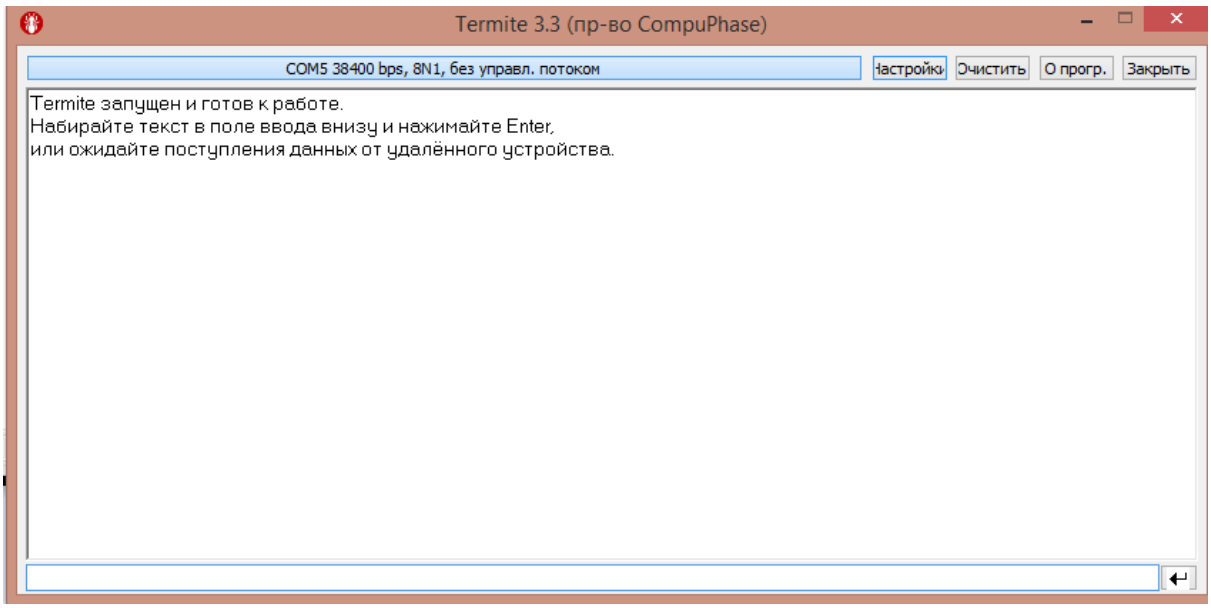


Рисунок 4.9 – Главный экран программы Termite 3.0

Далее необходимо выбрать порт, соответствующий подключенному конвертеру USB-UART TTL и выставить скорость обмена данными по последовательному порту. Скриншот настройки программы приведен на рисунке 4.10.

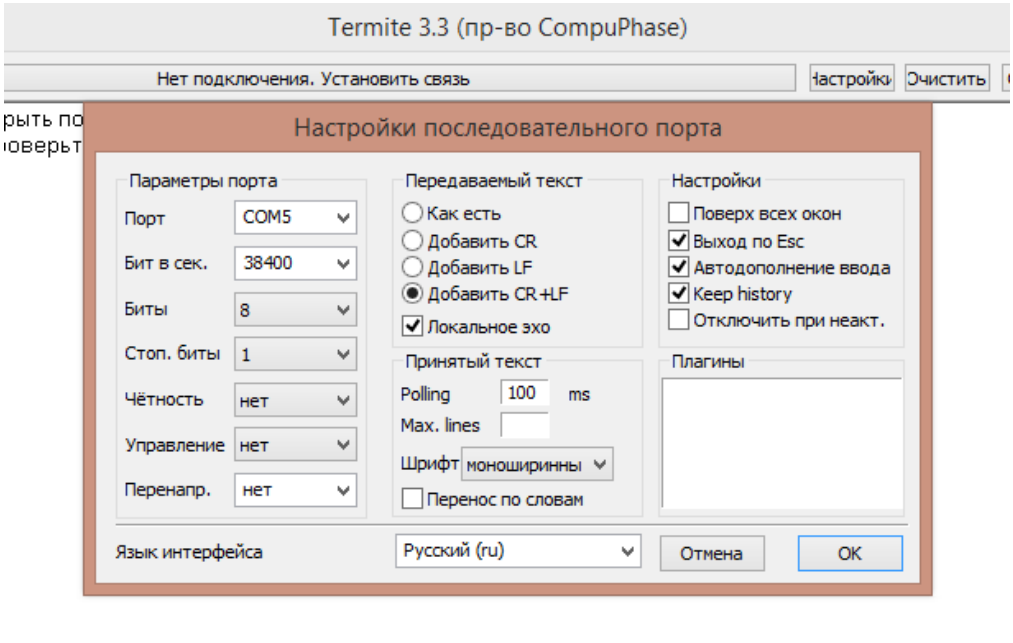


Рисунок 4.10 – Настройка программы Termite 3.0

Конфигурация модуля выполняется последовательно. Каждая команда начинается с **AT+** и отправляется с новой строки. Список команд взят с источника [8].

Листинг конфигурации модуля приемника:

```
AT  // Проверка установки соединения
OK
AT+ORGL // Сброс предыдущих настроек
OK
AT+RMAAD // Сброс ранее спаренных устройств
OK
AT+NAME=ROBOT // Присвоение имени
OK
AT+PSWD=641942 // Установка пароля для подключения
OK
AT+ROLE=0 // Перевод модуля в режим ведомого устройства (приемника)
OK
AT+ADDR? // Запросить адрес модуля
+ADDR:98d3:33:812531
OK
AT+RESET // Перезагрузить модуль
OK
```

Листинг конфигурации модуля передатчика:

```
AT // Проверка установки соединения
OK
AT+ORGL // Сброс предыдущих настроек
OK
AT+RMAAD // Сброс ранее спаренных устройств
OK
AT+ROLE=1 // Перевод модуля в режим ведущего устройства (передатчика)
OK
```

AT+NAME=REMOTE_CONTROL // Присвоение имени

OK

AT+INQ // Запрос на поиск ближайших устройств

+INQ:98D3:33:812531,0,7FFF //

OK

AT+RNAME? 98D3,33,812531 // Узнать имя устройства по адресу

+RNAME:ROBOT

AT+BIND=98d3,33,812531 // Адрес парного устройства (приемника), с которым нужно уставливать связь при включении

OK

AT+PAIR=98D3,33,812531,5 // Создать пару с указанным адресом (приемником)

OK

AT+PSWD=641942 // Запись в память пароля для подключения

OK

AT+CMODE=0 // Запрос на запрет соединяться с другими устройствами кроме этого

OK

AT+RESET // Перезагрузить модуль

OK

После проверки работоспособности устройств на макетных платах был проведен монтаж элементов на изготовленные печатные платы. Фотографии макетов устройств и последовательность сборки изображены на рисунках 4.11-4.16.

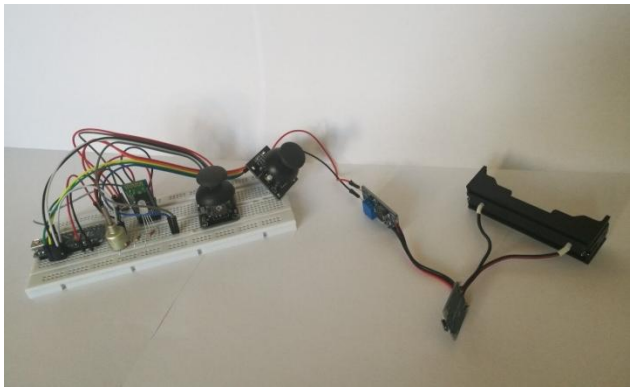


Рисунок 4.11 – Макет пульта управления

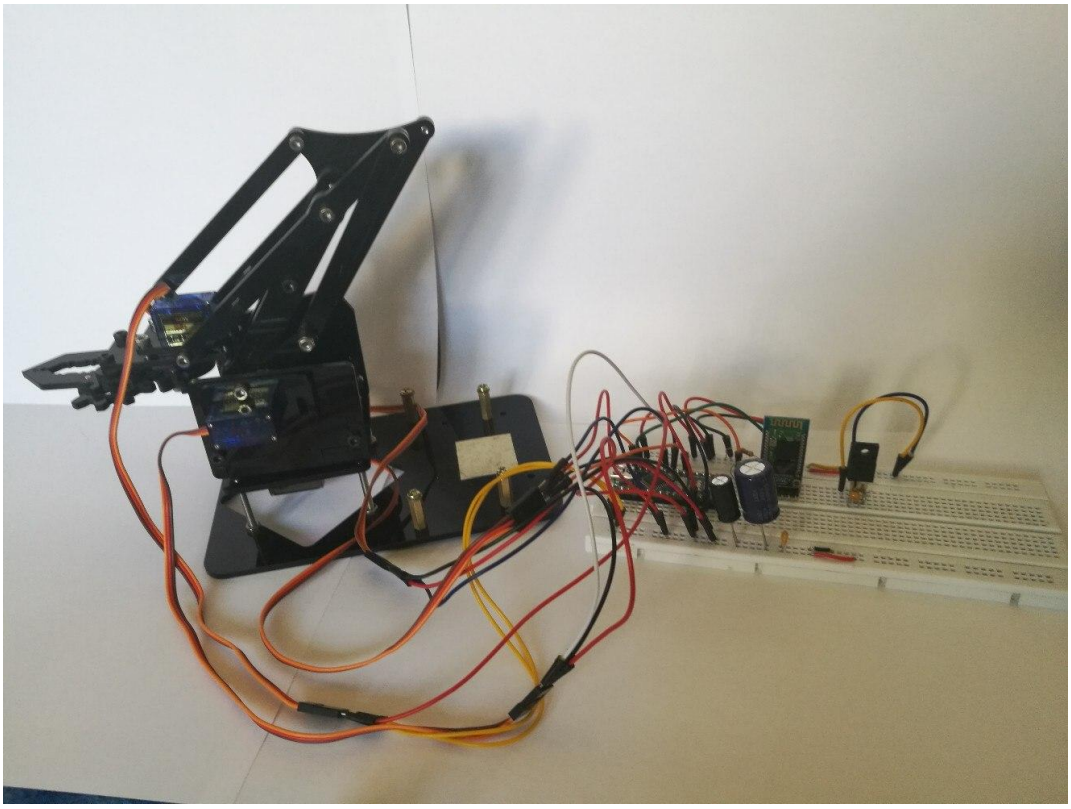


Рисунок 4.12 – Макет платы управления манипулятором

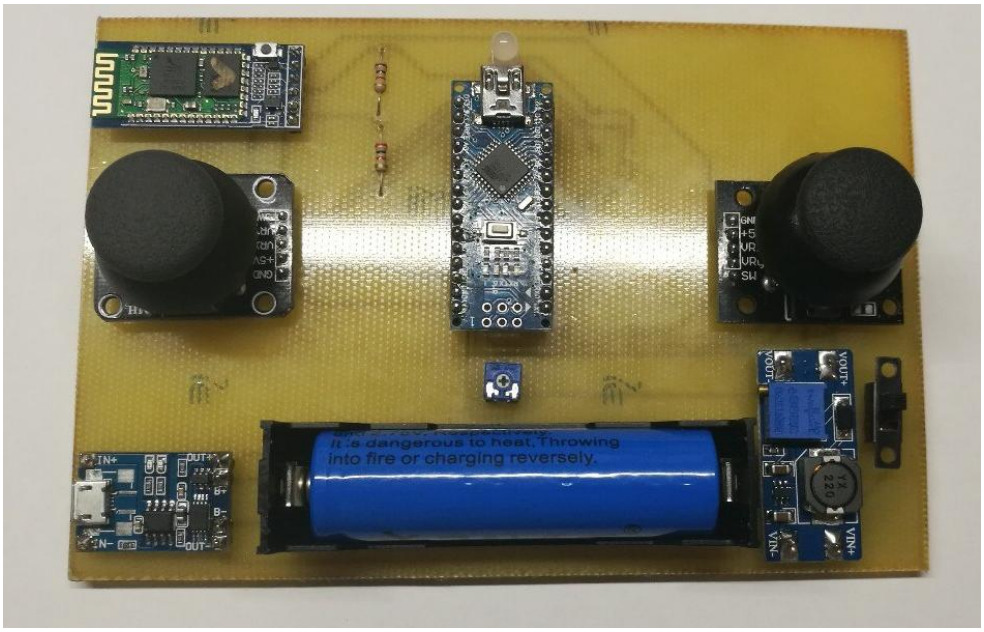


Рисунок 4.13 – Пульт управления манипулятором

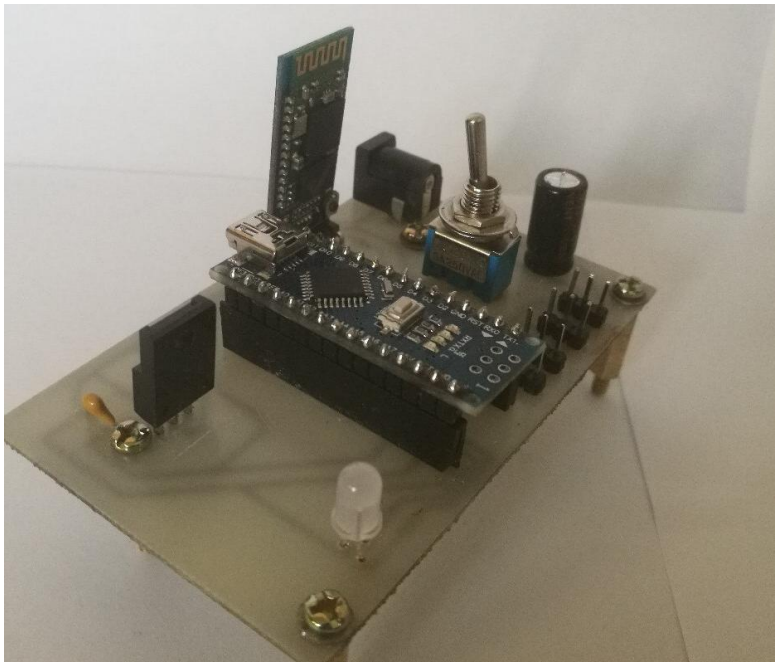


Рисунок 4.14 – Плата управления манипулятором

Для крепления платы управления на корпус манипулятора были использованы металлические стойки, изображенные на рисунке 4.15.

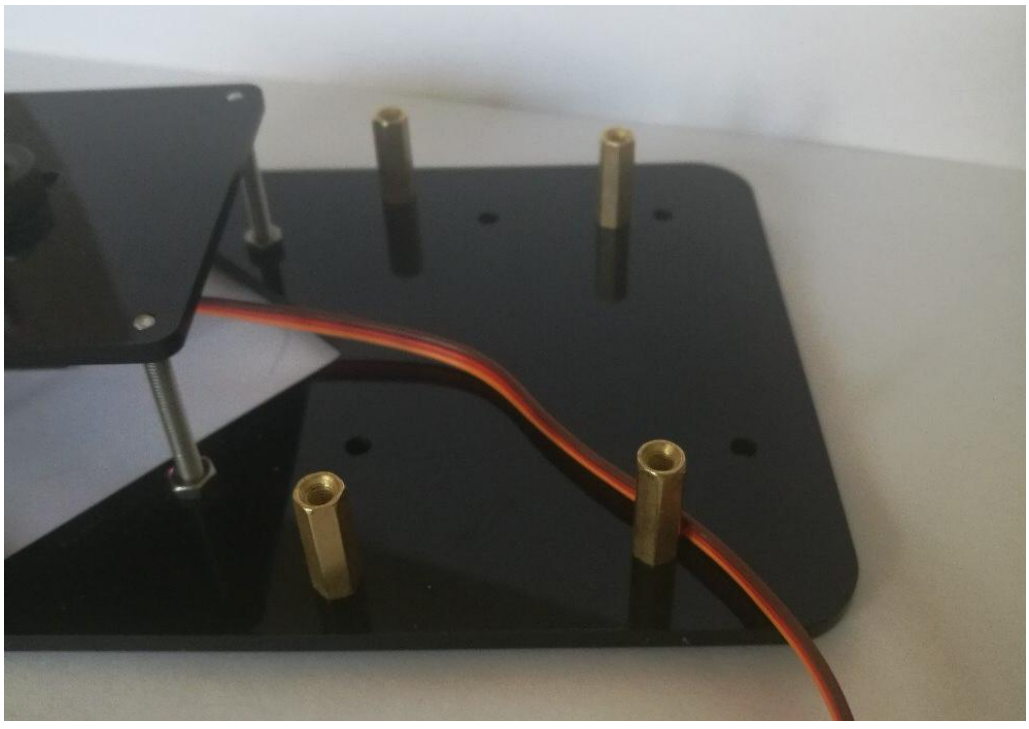


Рисунок 4.15 – Стойки крепления платы управления

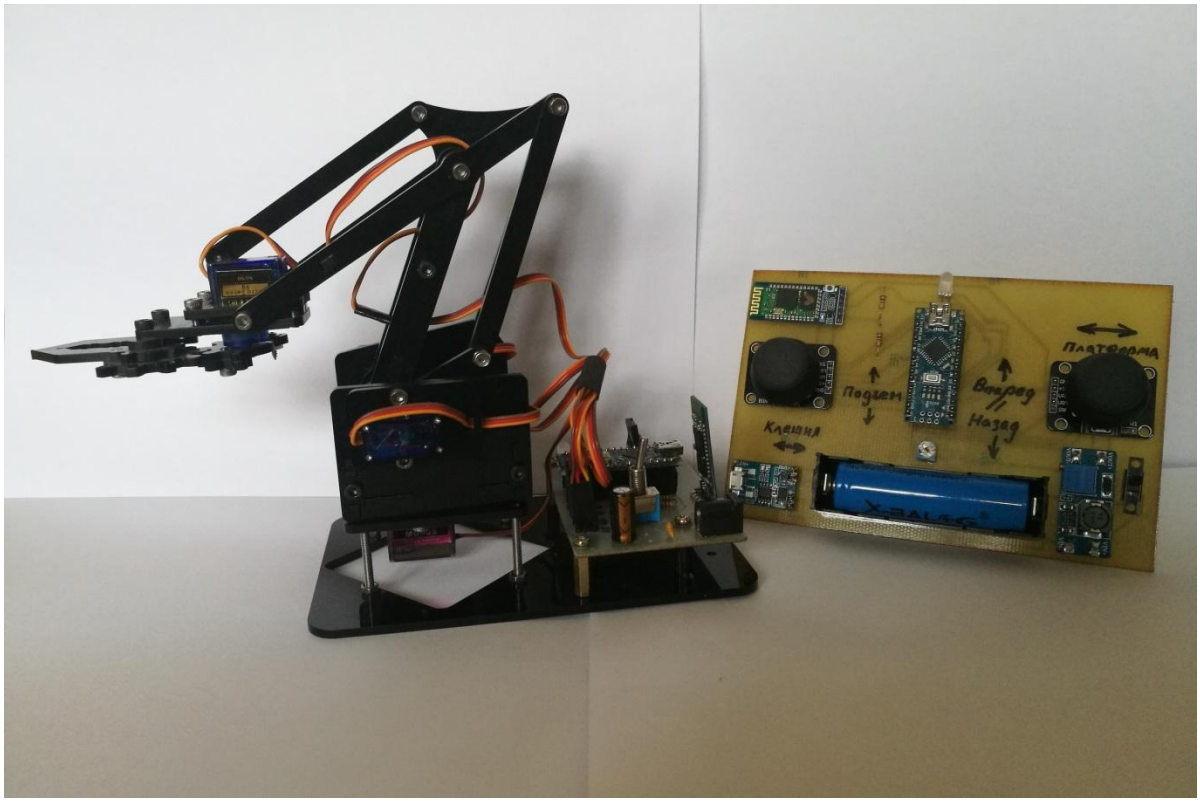


Рисунок 4.16 – Готовое устройство дистанционного управления манипулятором

5. Отладка и экспериментальные исследования

После запуска устройства управления манипулятором были проведены некоторые отладочные работы, а именно: калибровка диапазонов предельных значений углов поворота сервоприводов, записанных в переменные типа *MinAngl* и *MaxAngl*. А так же калибровка инициальных значений углов сервоприводов для установки манипулятора в стартовое положение при запуске устройства, записанных в переменные типа *StartAngl*.

В ходе проверки работоспособности манипулятора было установлено, что грузоподъёмность манипулятора слишком низкая. Для увеличения грузоподъемности было принято решение заменить сервоприводы SG90, установленные на платформе и главной оси руки манипулятора на более оптимальную модель MG90s, изображенную на рисунке 5.1.

Рисунок 5.1 - Сервопривод MG90s

Основное отличие данной модели от SG90 заключается в том, что в редукторе MG90 используются металлические шестерни вместо пластиковых, что дает во-первых, увеличение срока службы сервопривода, и во-вторых, повышение грузоподъёмности манипулятора.

Параметры сервопривода MG90s:

- Напряжение питания: 4.8…7,2 В
- Скорость без нагрузки: 0.1 сек /60 градусов;
- Крутящий момент: 2.8 кг*см;
- Рабочая температура: минус 30...+60 °С;
- Вес: 13.4г;
- Габаритные размеры: 23х12.2х29 мм.

Заключение

В ходе проделанной работы было разработано устройство дистанционного управления манипулятором. Устройство состоит из передатчика - пульта управления и приемника - платы управления сервоприводами. Передача данных между приемником и передатчиком осуществляется по защищенному Bluetooth каналу.

Был проведен анализ современных аналогов, в результате которого было выбрано оптимальное решение. Разработка схемы электрической принципиальной и печатной платы проводилось с помощью пакета программ Dip Trace. Разработка алгоритмов программ выполнялось в программе Studlab. Написание кодов программ осуществлялось в среде разработки Arduino IDE.

Данная работа соответствует заданию, цели и задачи проекта достигнуты.

Список используемой литературы

Для написания данного материала использовались следующие источники:

1 Блум, Д. Изучаем Arduino: инструменты и методы технического волшебства [Текст] / Пер. с англ. — СПб.: БХВ-Петербург, 2015. — 336 с: ил.; 2000 экз. - ISBN 978-5-9775-3585-4.

2 Ревич Ю. Занимательная электроника, 4е издание [текст] / БХВ-Петербург, 2017. – 640 с: ил.; 1500 экз. - ISBN 978-5-9775-3764-3.

3 Хоровиц, П., Хилл У. Искусство схемотехники [Текст] / Пер. с англ. – БИОНОМ, 2016. – 700 с: ил.; 3000 экз. – ISBN 978-5-9518-0351-1.

4 Общая информация о роботах. // bio-nica.narod.ru. (Дата обращения: 1.05.2018).

5 Применение роботов в науке. // roboting.ru (Дата обращения: 1.05.2018).

6 Применение роботов в медицине. // habrahabr.ru. (Дата обращения: 1.05.2018).

7 Применение роботов в промышленности geektimes.ru. (Дата обращения: 1.05.2018).

8 Калибровка Bluetooth модуля HC-05. // http://wiki.iarduino.ru/page/at-komandy-bluetooth-hc-05. (Дата обращения: 28.04.2018).

9 SINONING manipulators online store. // www.sinoning.cc/product-page/snarm. (Дата обращения: 28.04.2018).

10 SINONING tutorial and instructions site. // www.sinoning.com (Дата обращения: 15.04.2018).

11 Информация о Arduino UNO R3. // http://wiki.amperka.ru/%D0%BF%D1%80%D0%BE%D0%B4%D1%83%D0%BA%D1%82%D1%8B/. (Дата обращения: 17.05.2018).

12 Информация о плате расширения Sensor Board. // http://robot-kit.ru/product_info.php/info/p597_Plata-rasshireniya-Arduino-Sensor-Shield-V5-0--SensorShieldV5-.html. (Дата обращения: 05.05.2018).

13 Устройство работы джойстика. // https://ru.wikipedia.org/wiki/Джойстик. (Дата обращения: 05.05.2018.).

14 Arduino PS2 joystick tutorial. // http://henrysbench.capnfatz.com/henrys-bench/arduino-sensors-and-input/arduino-ps2-joystick-tutorial-keyes-ky-023-deek-robot/. (Дата обращения: 05.05.2018).

15 Параметры платы Arduino nano. // https://www.chipdip.ru/product/nano-v3.0-ft232-with-usb-cable. (Дата обращения: 17.05.2018).

16 Основные отличия Arduino UNO от Nano. // http://edurobots.ru/dictionary/arduino-nano/ (Дата обращения: 17.05.2018).

17 Информация о радиомодулях. // http://arduino-kit.ru/catalog/id/priemnik-%2B-peredatchik-433mhz-_komplekt-moduley_ (Дата обращения: 20.05.2018).

18 Параметры Wi-Fi модуля ESP. // http://amperka.ru/product/esp8266-wifi-module (Дата обращения: 20.05.2018) .

19 Параметры RGB-светодиода. //https://www.chipdip.ru/product/bl-l515rgbw-cc. (Дата обращения: 22.05.2018).

20 Параметры SMD резистора 0805. // https://www.chipdip.ru/product0/867824335(Дата обращения: 22.05.2018).

21 Параметры подстроечного резистора. // https://www.chipdip.ru/product/ca6v-10-kom. (Дата обращения: 22.05.2018).

22 Таблица стандартных рядов сопротивлений. // http://katod-anod.ru/articles/2 (Дата обращения: 23.05.2018).

23 Онлайн-калькулятор делителя напряжения. // https://h4e.ru/komplektuyushchie/117-raschet-delitelya-napryazheniya-na-rezistorakh-kondensatorakh-i-induktivnostyakh (Дата обращения: 23.05.2018).

24 Bluetooth HC-05 datasheet. // http://www.electronicaestudio.com/docs/istd016A.pdf (Дата обращения: 28.04.2018).

25 Arduino and Bluetooth HC-05 tutorial. // https://howtomechatronics.com/tutorials/arduino/arduino-and-hc-05-bluetooth-module-tutorial/ (Дата обращения: 28.04.2018).

26 KIA7805a voltage stabilizer datasheet. // http://pdf1.alldatasheet.com/datasheet-pdf/view/223217/ESTEK/7805A.html - (Дата обращения: 22.05.2018).

27 Онлайн-калькулятор дорожек печатной платы http://radioaktiv.ru/raschet-shiriny-dorozhki-pechatnoy-platy.html (Дата обращения: 25.05.2018).

28 Онлайн-сервис разработки блок-схем алгоритмов. // https://www.impulsi.ru/catalog/istochniki_pitaniya_i_batarei/akkumulyatory/131789/ - аккум (Дата обращения: 18.05.2018).

29 Параметры платы заряда аккумуляторов. // http://we.easyelectronics.ru/part/zaryadnoe-ustroystvo-dlya-li-ion--na-tr4056.html (Дата обращения: 23.05.2018).

30 Техническая документация на преобразователь напряжения http://roboparts.ru/products/mt3608 (Дата обращения: 24.05.2018).

```
1  #include <SoftwareSerial.h>; // подключение библиотеки для создания виртуального порта
2
3  SoftwareSerial BT(10, 11); // создание виртуальной библиотеки для bluetooth. В скобках
   указываются порты виртуального порта (10-RX, 11-TX)
4
5  //Обьявление области углов поворота сервоприводов в градусах,
6  //где MinAngl - минимальное предельное значение, MaxAngl - максимальное предельное
   значение
7  int const MinAngl1 = 25; int const MaxAngl1 = 170; // для сервопривода платформы вращения
8  int const MinAngl2 = 80; int const MaxAngl2 = 170; // для сервопривода вертикальной оси
   отклонения руки манипулятора
9  int const MinAngl3 = 20; int const MaxAngl3 = 100; // для сервопривода горизонтальной оси
   отклонения руки манипулятора
10 int const MinAngl4 = 85; int const MaxAngl4 = 135; // для сервопривода рабочего органа
   манипулятора (клешни захвата)
11 //--------------------------------
12
13 // Объявление переменных с инициальными значениями углов сервоприводов
14 int const StartAngl1 = 90; // для сервопривода платформы вращения
15 int const StartAngl2 = 50; // для сервопривода вертикальной оси отклонения руки
   манипулятора
16 int const StartAngl3 = 80; // для сервопривода горизонтальной оси отклонения руки
   манипулятора
17 int const StartAngl4 = 120; // для сервопривода рабочего органа манипулятора (клешни
   захвата)
18 //--------------------------------
19
20 // Определение областей ручки джойстика
21 int const StartDec = 300; int const StartInc = 700; // Определение значений начала отклонения
   по оси
22 int const FinDec = 100; int FinInc = 900; // Определение максимальных значений отклонения по
   оси
23 int const RelaseDec = 500; int const RelaseInc = 600; // Определение области нулевой точки
```

```
24 //--------------------------------
25
26 // Обьявление переменных хранения значений углов поворота сервоприводов
27 int angl1;  // для сервопривода платформы вращения
28 int angl2; // для сервопривода вертикальной оси отклонения руки манипулятора
29 int angl3;  // для сервопривода горизонтальной оси отклонения руки манипулятора
30 int angl4;  // для сервопривода рабочего органа манипулятора (клешни захвата)
31 //--------------------------------
32
33 // объявление переменных для определения идентификаторов сессий
34 int IDcur = 0;
35 int IDprev = 0;
36
37 //-----------------------------------------
38
39 int mode = 0; //Обьявление переменной идентификатора векторов отклонения ручки
   джойстика по осям
40
41 void setup() // цикл процедур при запуске контроллера
42 {
43   //Serial.begin (9600); // открытия последовательного порта обмена данными, используется
   для отладки
44   BT.begin(38400); // начало сессии для bluetooth со скоростью 38400 бод
45   BT.setTimeout(40);
46 }
47
48 void loop() // основной цикл программы
49 {
50
51 // Объявление переменных осей джойстика и привязка к аналоговым выводам
52 // Назначение переменной для калибровки чувствительности sens, привязка к выводу
53   int joy1x = analogRead(A0); // подключение оси X первого джойстика к порту 0 АЦП
54   int joy1y = analogRead(A1); // подключение оси Y первого джойстика к порту 1 АЦП
55   int joy2x = analogRead(A2); // подключение оси X второго джойстика к порту 2 АЦП
56   int joy2y = analogRead(A3); // подключение оси Y второго джойстика к порту 3 АЦП
```

```
  int sens = analogRead (A4); // подключение выхода потенциометра к порту 4 АЦП
  sens = map(sens, 0, 1023, 1, 20); // масштабирование диапазона АЦП с 0-1023 до 1-20. При вращении ручки потенциометра переменная sens будет изменяться в диапазоне от 1 до 20
//---------------------------------------------------------------------

//Условия для единичной отправки идентификатора осей
  if ((joy1x <= StartDec || joy1x >= StartInc) && IDprev == 0)   // если ось X первого джойстика отклонилась от нулевой точки
  {
    IDcur = 1; // присвоить значение 1
  }
  else if ((joy1y <= StartDec || joy1y >= StartInc) && IDprev == 0) // для оси Y первого джойстика
  {
    IDcur = 2;
  }
  else if ((joy2x <= StartDec || joy2x >= StartInc) && IDprev == 0) // для оси X второго джойстика
  {
    IDcur = 3;
  }
  else if ((joy2y <= StartDec || joy2y >= StartInc) && IDprev == 0) // для оси Y второго джойстика
  {
    IDcur = 4;
  }
//-------------------------------------------------

// Условия для открытия сесии передачи данных по Bluetooth
  if (IDcur == 1 && IDprev == 0) //если джойстик отклонился и сессия передачи данных не активна
  {
    BT.write(1); // отправить идентификатор со значением "1" по Bluetooth
    Serial.print(1); // вывод значения для отладки
    IDcur = 0; // выставление флага в 0, чтобы при удержании ручки на джойстике постоянно не отсылались идентификаторы
    IDprev = 1; // присвоить значение 1
  }
```

```
  else if (IDcur == 2 && IDprev == 0)
  {
    BT.write(2);
    Serial.print(2);
    IDcur = 0;
    IDprev = 2;
  }
  else if (IDcur == 3 && IDprev == 0)
  {
    BT.write(3);
    Serial.print(3);
    IDcur = 0;
    IDprev = 3;
  }
  else if (IDcur == 4 && IDprev == 0)
  {
    BT.write(4);
    Serial.print(4);
    IDcur = 0;
    IDprev = 4;
  }
//-----------------------------------

//Условия для остановки передачи данных
  if ((joy1x >= RelaseDec && joy1x <= RelaseInc) && IDprev == 1) // если ручка джойстика вернулась в нулевое положение и сессия была открыта
  {
    IDprev = 0; //
    BT.write(9); // отправить идентификатор окончания сессии
    //Serial.println(9); // для отладки
  }
  else if ((joy1y >= RelaseDec && joy1y <= RelaseInc) && IDprev == 2)
  {
    IDprev = 0;
    BT.write(9);
```

```
122     //Serial.println(9);
123    }
124    else if ((joy2x >= RelaseDec && joy2x <= RelaseInc) && IDprev == 3)
125    {
126     IDprev = 0;
127     BT.write(9);
128     //Serial.println(9);
129    }
130    else if ((joy2y >= RelaseDec && joy2y <= RelaseInc) && IDprev == 4)
131    {
132     IDprev = 0;
133     BT.write(9);
134     //Serial.println(9);
135    }
136  //---------------------------------------------------------------------
137
138  // Условия для изменения переменных значений углов сервопривода
139    if (joy1x >= StartInc && IDprev == 1) // если ручка джойстика сдвинута и был отправлен
     идентикатор оси по Bluetooth
140    {
141     mode = 1; // присвоить значение 1
142    }
143    else if (joy1x <= StartDec && IDprev == 1)
144    {
145     mode = 2;
146    }
147    else if (joy1y >= StartInc && IDprev == 2)
148    {
149     mode = 3;
150    }
151    else if (joy1y <= StartDec && IDprev == 2)
152    {
153     mode = 4;
154    }
155    else if (joy2x >= StartInc && IDprev == 3)
```

```
156   {
157     mode = 5;
158   }
159   else if (joy2x <= StartDec && IDprev == 3)
160   {
161     mode = 6;
162   }
163   else if (joy2y >= StartInc && IDprev == 4)
164   {
165     mode = 7;
166   }
167   else if (joy2y <= StartDec && IDprev == 4)
168   {
169     mode = 8;
170   }
171   else
172   {
173     mode = 9; // значение для остановки отправки данных
174   }
175
176   switch (mode) // определение алгоритмов в соответствии с значением переменной mode
177   {
178     case 1:
179       if (joy1x >= FinInc && angl1 == MaxAngl1) // если угол сервопривода достиг максимального
      значения
180       {
181         mode = 9; // остановить отправку данных
182       }
183       else // иначе
184       {
185         sens = map(joy1x, StartInc, 1023, 1, sens); // регулировка скорости вращения сервопривода
      в зависимости от настройки чувствительности
186         angl1 = angl1 + sens; // увеличить угол вращения сервопривода 1 на значение sens
187         angl1 = constrain (angl1, MinAngl1, MaxAngl1); // функция ограничения диапазона угла
      поворота в пределах максимального и минимального значения
```

```
188       BT.println('\n'); // команда для последовательной отправки данных
189       BT.print(angl1); // отправить значение угла поворота сервопривода
190      }
191      //delay(40);
192      //Serial.print("angl1="); // для отладки
193      //Serial.println(angl1);
194      break;
195     case 2:
196      if (joy1x <= FinDec && angl1 == MinAngl1)
197      {
198       mode = 9;
199      }
200      else
201      {
202       sens = map(joy1x, StartDec, 0, 1, sens);
203       angl1 = angl1 - sens;
204       angl1 = constrain (angl1, MinAngl1, MaxAngl1);
205       BT.println('\n');
206       BT.print(angl1);
207
208      }
209      //delay(40);
210      //Serial.print("angl1=");
211      //Serial.println(angl1);
212      break;
213     case 3:
214      if (joy1y >= FinInc && angl2 == MaxAngl2)
215      {
216       mode = 9;
217      }
218      else
219      {
220       sens = map(joy1y, StartInc, 1023, 1, sens);
221       angl2 = angl2 - sens;
222       angl2 = constrain (angl2, MinAngl2, MaxAngl2);
```

```
223         BT.println('\n');
224         BT.print(angl2);
225       }
226       //delay(40);
227       //Serial.print("angl2=");
228       //Serial.println(angl2);
229       break;
230     case 4:
231       if (joy1y <= FinDec && angl2 == MinAngl2)
232       {
233         mode = 9;
234       }
235       else
236       {
237         sens = map(joy1y, StartDec, 0, 1, sens);
238         angl2 = angl2 + sens;
239         angl2 = constrain (angl2, MinAngl2, MaxAngl2);
240         BT.println('\n');
241         BT.print(angl2);
242       }
243       //delay(40);
244       //Serial.print("angl2=");
245       //Serial.println(angl2);
246       break;
247     case 5:
248       if (joy2x >= FinInc && angl3 == MaxAngl3)
249       {
250         mode = 9;
251       }
252       else
253       {
254         sens = map(joy2x, StartInc, 1023, 1, sens);
255         angl3 = angl3 + sens;
256         angl3 = constrain (angl3, MinAngl3, MaxAngl3);
257         BT.println('\n');
```

```
258         BT.print(angl3);
259       }
260       //delay(40);
261       //Serial.print("angl3=");
262       //Serial.println(angl3);
263       break;
264     case 6:
265       if (joy2x <= FinDec && angl3 == MinAngl3)
266       {
267         mode = 9;
268       }
269       else
270       {
271         sens = map(joy2x, StartDec, 0, 1, sens);
272         angl3 = angl3 - sens;
273         angl3 = constrain (angl3, MinAngl3, MaxAngl3);
274         BT.println('\n');
275         BT.print(angl3);
276       }
277       //delay(40);
278       //Serial.print("angl3=");
279       //Serial.println(angl3);
280       break;
281     case 7:
282       if (joy2y >= FinInc && angl4 == MaxAngl4)
283       {
284         mode = 9;
285       }
286       else
287       {
288         sens = map(joy2y, StartInc, 1023, 1, sens);
289         angl4 = angl4 + sens;
290         angl4 = constrain(angl4, MinAngl4, MaxAngl4);
291         BT.println('\n');
292         BT.print(angl4);
```

```
293       }
294       //delay(40);
295       //Serial.print("angl4=");
296       //Serial.println(angl4);
297       break;
298     case 8:
299       if (joy2y <= FinDec && angl4 == MinAngl4)
300       {
301         mode = 9;
302       }
303       else
304       {
305         sens = map(joy2y, StartDec, 0, 1, sens);
306         angl4 = angl4 - sens;
307         angl4 = constrain(angl4, MinAngl4, MaxAngl4);
308         BT.println('\n');
309         BT.print(angl4);
310       }
311       //delay(40);
312       //Serial.print("angl4=");
313       //Serial.println(angl4);
314       break;
315     case 9:
316       return;
317   }
318   delay (40); // частота выполнения основного цикла программы
319 }
```

```
1  #include <Servo.h>; // Подключене библиотеки управления сервоприводами
2
3  #include <SoftwareSerial.h>; // подключение библиотеки для создания виртуального порта
4
5  SoftwareSerial BT(10, 11); // создание виртуальной библиотеки для bluetooth 10-RX, 11-TX
6
7  //Обьявление области углов поворота сервоприводов в градусах,
8  //где MinAngl - минимальное предельное значение, MaxAngl - максимальное предельное
   значение
9  int const MinAngl1 = 25; int const MaxAngl1 = 170; // для сервопривода платформы вращения
10 int const MinAngl2 = 80; int const MaxAngl2 = 170; // для сервопривода овертикальной оси
   отклонения руки манипулятора
11 int const MinAngl3 = 20; int const MaxAngl3 = 100; // для сервопривода горизонтальной оси
   отклонения руки манипулятора
12 int const MinAngl4 = 85; int const MaxAngl4 = 135; // для сервопривода рабочего органа
   манипулятора (клешни захвата)
13 //--------------------------------
14
15 // Объявление переменных с инициальными значениями углов сервоприводов
16 int const StartAngl1 = 90; // для сервопривода платформы вращения
17 int const StartAngl2 = 50; // для сервопривода овертикальной оси отклонения руки
   манипулятора
18 int const StartAngl3 = 80; // для сервопривода горизонтальной оси отклонения руки
   манипулятора
19 int const StartAngl4 = 120; // для сервопривода рабочего органа манипулятора (клешни
   захвата)
20 //--------------------------------
21
22 // Обьявление переменных хранения значений углов поворота сервоприводов
23 int angl1;  // для сервопривода платформы вращения
24 int angl2; // для сервопривода овертикальной оси отклонения руки манипулятора
25 int angl3;  // для сервопривода горизонтальной оси отклонения руки манипулятора
26 int angl4;  // для сервопривода рабочего органа манипулятора (клешни захвата)
```

```
//---------------------------------

// Обьявление переменных хранения значений реальных углов поворота сервоприводов
int Rangl1;  // для сервопривода платформы вращения
int Rangl2; // для сервопривода овертикальной оси отклонения руки манипулятора
int Rangl3;  // для сервопривода горизонтальной оси отклонения руки манипулятора
int Rangl4;  // для сервопривода рабочего органа манипулятора (клешни захвата)
//---------------------------------

// Создание объектов сервоприводов для каждой оси
Servo axis1; // для сервопривода платформы вращения
Servo axis2; // для сервопривода овертикальной оси отклонения руки манипулятора
Servo axis3; // для сервопривода горизонтальной оси отклонения руки манипулятора
Servo axis4; // для сервопривода рабочего органа манипулятора (клешни захвата)
//---------------------------------

int ID; // иднтификатор, поступающий с передатчика

int IDses = 0; // идентификатор сессии

void setup()
{
  axis1.attach(3); // Подключение  сервопривода платформы к выводу D3 Arduino
  axis2.attach(5); // Подключение сервопривода вертикальной оси к выводу D5 Arduino
  axis3.attach(6); // Подключение сервопривода горизонтальной оси к выводу D6 Arduino
  axis4.attach(9); // Подключение сервопривода клешни к выводу D9 Arduino

  // Установка сервоприводов в стартовое положение при включении устройства
  axis1.write( StartAngl1);
  delay(15);
  axis2.write(StartAngl2);
  delay(15);
  axis3.write(StartAngl3);
  delay(15);
  axis4.write(StartAngl4);
```

```
62    delay(15);
63  //--------------------------------
64
65    BT.begin(38400); // начало сессии для bluetooth со скоростью 38400 бод
66    BT.setTimeout(40);
67  //  Serial.begin(9600); // для отладки
68  }
69
70  void loop()
71  {
72    if (BT.available() > 0) // если приходят какие либо данные
73    {
74      ID = BT.read(); // они записываются в переменную ID
75      if (ID == 1) // если пришло 1
76      {
77        IDses = 1; // идентификатор сессии принимает значение 1
78      }
79      else if (ID == 2)
80      {
81        IDses = 2;
82      }
83      else if (ID == 3)
84      {
85        IDses = 3;
86      }
87      else if (ID == 4)
88      {
89        IDses = 4;
90      }
91      else if (ID == 9) // если пришло 9
92      {
93        IDses = 0; // идентификатор становится 0
94      }
95
96      if (IDses == 1 && BT.available()> 0) // если IDses в значении 1 и данные продолжают
```

```
    поступать
 97   {
 98    angl1 = BT.parseInt(); // данные записываются в переменную угла поворота
 99    Rangl1 = axis1.read(); // выполняется считывание текущего угла положения сервопривода
100     if (Rangl1 - angl1 > 50) // если разница между ними составляет 50 (большая разница)
101     {
102       delay(40); // ожидание следующих валидных данных
103     }
104     else
105     {
106       axis1.write(angl1 = constrain(angl1, MinAngl1, MaxAngl1)); // запись в сервопривод угла
107       //delay(25);
108     }
109    //Serial.print("Angl1=");
110    //Serial.println(angl1);
111   }
112   else if (IDses == 2 && BT.available()> 0)
113   {
114     angl2 = BT.parseInt();
115       Rangl2 = axis2.read();
116         if (Rangl2 - angl2 > 50)
117         {
118           delay(40);
119         }
120         else
121         {
122           axis2.write(angl2 = constrain(angl2, MinAngl2, MaxAngl2));
123           //delay(25);
124         }
125     Serial.print("Angl2=");
126     Serial.println(angl2);
127   }
128   else if (IDses == 3 && BT.available()> 0)
129   {
130     angl3 = BT.parseInt();
```

```
131       Rangl3 = axis3.read();
132         if (Rangl3 - angl3 > 50)
133         {
134           delay(40);
135         }
136         else
137         {
138           axis3.write(angl3 = constrain(angl3, MinAngl3, MaxAngl3));
139           //delay(25);
140         }
141       Serial.print("Angl3=");
142       Serial.println(angl3);
143     }
144     else if (IDses == 4 && BT.available()> 0)
145     {
146       angl4 = BT.parseInt();
147       Rangl4 = axis4.read();
148         if (Rangl4 - angl4 > 50)
149         {
150           delay(40);
151         }
152         else
153         {
154           axis4.write(angl4 = constrain(angl4, MinAngl4, MaxAngl4));
155           //delay(25);
156         }
157       Serial.print("Angl4=");
158       Serial.println(angl4);
159     }
160   }
161
162    delay(40);
163 }
```