ОЛИМПИАДА ШКОЛЬНИКОВ «ШАГ В БУДУЩЕЕ»

НАУЧНО-ОБРАЗОВАТЕЛЬНОЕ СОРЕВНОВАНИЕ

«ШАГ В БУДУЩЕЕ», СЕКЦИЯ «Робототехнические системы»

Кафедра «СМ7»

36336

регистрационный номер

Робот для выполнения специальных заданий «FORTIS»

Автор: Шилкин Даниил Алексеевич

МБОУ ЦО2 г. Тула

11 класс

Научный руководитель:

Стручев Денис Александрович,

Преподаватель в детском технопарке

«Кванториум» г. Тула

г. Тула

2024

Робот для выполнения специальных заданий «FORTIS»

Аннотация

Цель проекта: Разработка компактного, бюджетного, но при этом проходимого и многофункционального робота с роботизированной рукой на базе микроконтроллера семейства Arduino, предназначенного для выполнения специальных заданий.

Задачи проекта:

- Изучить действия спасателей при обрушении сооружений и опасность этих работ.
- Выделить основные узлы робота и пульта управления.
- Подобрать и рассчитать комплектующие.
- Спроектировать 3Д модель в программе «Компас 3Д», Изготовить необходимые детали.
- Собрать корпус и подключить электронику робота, а также создать и настроить пульт управления.
- Написать и отладить программу для робота и пульта управления.
- Протестировать все узлы, доработать наиболее слабые узлы.

Методы и приёмы:

- Проектирование 3д модели в программе «Компас 3Д».
- Изготовление необходимых деталей с помощью аддитивных технологий и технологий лазерной резки с использованием станков с ЧПУ.
- Написание кода на языке C++ для Arduino в среде программирования Arduino IDE.
- Отладка и тестирования электронных Схем в симуляторах таких, как Tinkercad, Wokwi.

Полученные результаты: В результате разработки получилось создать робота для выполнения специальных заданий. При этом робот имеет высокую проходимость и манипулятор. Также робот оснащен аппаратурой управления и системой fpv камер.

Вывод: Тестирование робота показало, что он обладает высокой проходимостью, достаточным временем работы. В результате данное устройство может использоваться военными и спасателями.

СОДЕРЖАНИЕ

1. **ВВЕДЕНИЕ** 4

1.1. Цель, актуальность и практическое значение разработки 4

1.2. Задачи и методы исследования 6

1.2.1. Определение основных узлов робота, подбор и расчет комплектующих. 7

2. Разработка робота 10

2.1. Основные узлы робота 10

2.2. Расчёт и подбор необходимых комплектующих 10

2.3. Проектирование 14

2.4. Сборка робота 16

2.5. Написание программы для МК, калибровка и отладка всех узлов 23

2.5.1. программа для робота 23

2.5.2. программа для пульта управления роботом 23

2.6. После полной сборки были проведены тесты устройства 24

2.7. Сравнение разработки с уже существующими роботами. 28

2.8. Практическое применение. 29

3. Условные обозначения и сокращения 30

4. Список использованных источников 30

5. Приложения 30

1. ВВЕДЕНИЕ

1.1.Цель, актуальность и практическое значение разработки

Целью проекта является разработка малогабаритного робота с дистанционным управлением для выполнения различных задач в экстремальных или опасных для человека условиях окружающей среды.

В настоящее время МЧС, военные и сотрудники других служб практически всегда при выполнении работ подвергают свою жизнь опасностям. Например: при обрушении различных конструкций сотрудники МЧС вынуждены пробираться под завалы, для поиска людей. При этом зачастую существует угроза обвала. А если проникнуть под завалы возможностей нет, то и вовсе приходится ждать, пока завалы разберет тяжелая техника.

Помимо разбора самих завалов предварительно сотрудники МЧС должны выполнять некоторые действия, которые обезопасят нахождение людей вблизи завала (в примере приводятся действия при разборе завалов каких-либо зданий или искусственных сооружений).

После установления зоны ЧС и её характера спасатели переходят к оценки состояния окружающей среды: наличие очагов пожара, радиоактивного, химического, бактериального заражения.

Далее спасатели устанавливают состояние объектов в зоне ЧС - это различные коммуникации и системы.

Для помощи спасателям в данных этапах и возникла идея собрать робота. На его платформу могут быть установлены дополнительные сканирующие системы, позволяющие оператору находится на некотором расстоянии от робота, и с безопасностью для своей жизни проводить мониторинг и разведку окружающей среды, в том числе поиск людей, поврежденных узлов и коммуникаций. Также осуществлять последующие отключение коммуникаций, которые представляют опасность.

На данный момент в арсенале спецслужб есть некоторые роботы, которые предназначены для тушения пожаров, разминирования или разведки. Но главным минусом всех этих устройств являются их размеры, вес и стоимость (например мобильный РТК Teodor весит в полном снаряжении 375 кг). Большинство машин человек просто не может передвигать самостоятельно, а для этого необходимы специальные грузовики или фургоны.

Не всегда есть возможность подъезда подобных машин в зону работы сотрудников МЧС. Также в силу больших габаритов и высокой стоимости, сложности изготовления больших роботов такие комплексы есть в далеко не в каждом регионе РФ. Именно поэтому я посчитал необходимым разработать относительно легкого, бюджетного, простого в ремонте и управлении робота, но при этом не уступающего по проходимости и функционалу большим роботам. В случае успеха данной разработки любой сотрудник сможет взять робота и самостоятельно переместить его в зону работы. А в случае повреждения каких либо узлов робота его можно было бы быстро заменить на месте и продолжить работу.

Подобный робот может выполнять небольшие, но очень важные задачи, которые требуют оперативного решения. Пример некоторых задач, которые должен выполнять робот:

- Поиск людей и доставку грузов в зону обрушения зданий
- Изучение места ЧС и анализа окружающей среды
- Оценка вероятности обрушения завалов
- Проведение радиоактивного мониторинга
- Разминирование и перемещение подозрительных предметов
- Установка ретрансляторов

Уникальностью робота будут являться его небольшие габариты, вес не более 10кг, а также высокая проходимость и возможность забираться по ступенькам и высоким наклонным поверхностям.

1.2.Задачи и методы исследования

Для создания робота, удовлетворяющего всем требованиям, необходимо было выполнить следующие **задачи**:

- Определить основные узлы робота, в соответствии с этим подобрать необходимые механические и электронные комплектующие.
- Составить структурную схему устройства робота и схему подключения всех его узлов
- Разработать аппаратуру дистанционного управления и разработать протокол передачи данных между роботом и аппаратурой
- Сборка, настройка и калибровка всех узлов робота.
- Написание программы для робота и аппаратуры

Методы исследования:

- Изучение материалов по действию спасателей при обрушении зданий и их последующего разбора.
- Обработка всей информации, постановка основных требований к создаваемому устройству.
- Сбор информации о различных электронных модулях, их характеристики.
- Проектирование модели робота в САПР программе.
- Отладка электронных схем с помощью симуляций

– Программирование микроконтроллера и отладка кода с помощью монитора порта.

1.2.1. Определение основных узлов робота, подбор и расчет комплектующих.

После подробного изучения уже существующих решений данной проблемы были выделены основные узлы, характерные для роботов, предназначенных для применения в подобных ситуациях.

Такими узлами являются:

- Ходовая часть
- Манипулятор
- Электронный блок
- Система камер

Для движения робота необходимо разработать ходовую часть, которую в движение приводят либо бесколлекторные, либо коллекторные моторы. Изучив преимущества и недостатки, а также цены были выбраны коллекторные моторы с редуктором в силу их простого подключения, небольшой цены и удобства использования.

Рисунок 1. Коллекторный мотор

При подборе комплектующих для манипулятора, а именно моторов рассматривались несколько вариантов, а именно: актуаторы, коллекторные моторы с червячными редукторами, сервоприводы и обычные коллекторные моторы. Наиболее компактным и надежным решением

оказались сервоприводы. Они не требуют подключения дополнительных драйверов или установки внешних потенциометров, энкодеров для отслеживания положения манипулятора и удержания заданного угла.

Рисунок 2. Сервопривод

Самым важным и сложным был выбор основного вычислительного и управляющего узла для робота.

Существует множество различных Микроконтроллеров (далее - МК) с различными параметрами и возможностями. Наиболее популярны, просты в использовании и программировании МК линейки Arduino.

Создание робота предполагает подключения большого количества различных устройств и систем, поэтому МК должен иметь большое количество пинов.

После сравнения всех плат семейства Arduino в качестве главного управляющего устройства была выбрана плата Arduino Mega 2560 (как для пульта, так и для робота).

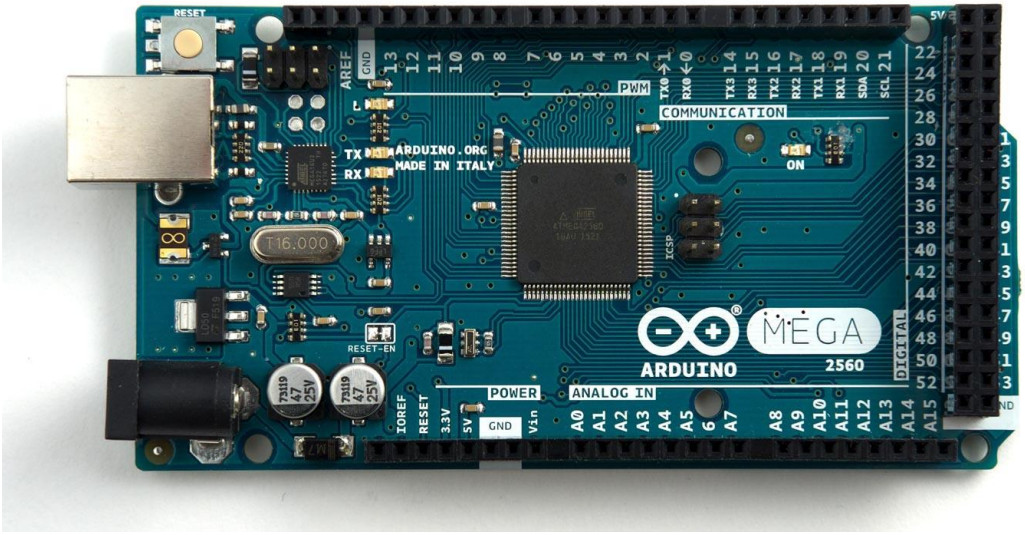


Рисунок 3. Отладочная плата Arduino mega 2560

Помимо основного МК необходимы были и другие компоненты, подбор и расчет которых приводится далее.

Для управления роботом в условиях ограниченной видимости на робот будет установлена система fpv камер с видеопередатчиками, которые имеют достаточно большой радиус действия и простую настройку.

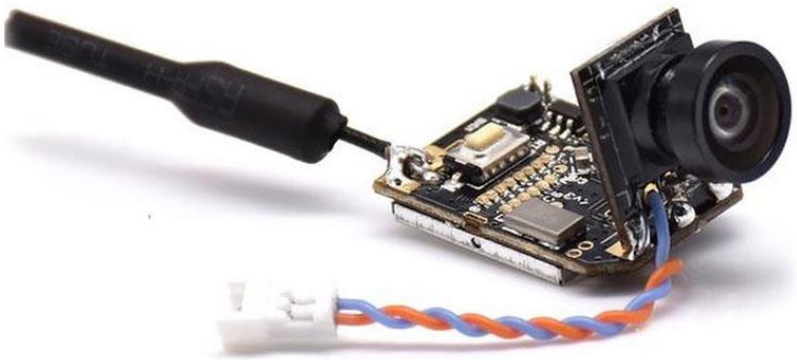


Рисунок 4. Fpv камера с передатчиком

2. Разработка робота

2.1.Основные узлы робота

Робот включает в себя следующие основные узлы:

- Ходовая часть
- Манипулятор
- Камеры и видеопередатчики
- Узел обмена информации с пультом управления
- Основной узел электроники
- Узел распределения и преобразования напряжения

Рисунок 5. Структурная схема робота

2.2.Расчёт и подбор необходимых комплектующих

Самым главным узлом, от которого будут зависеть большинство других показателей робота – это ходовой узел. После анализа всех возможных ходовых частей мой выбор остановился на 4-х колесной базе, но с некоторыми модификациями. Т.к. робот планируется использовать в труднопроходимых местах, то робот должен обладать повышенной проходимостью, и помимо этого уметь преодолевать высокие препятствия, в том числе и ступеньки. Поэтому основную ходовую часть было решено сделать из четырех колес с полным приводом. При этом сделать передний мост

независимым, чтобы даже на неровных поверхностях у робота оставался контакт с поверхностью всеми колесами. Но несмотря на все преимущества данная конструкция не позволила бы роботу забираться на высокие препятствия, более того существовал бы высокий шанс опрокидывания робота, чего допускать нельзя. Поэтому для решения данной проблемы решено было добавить 3-ю пару колес, закреплённых на сервоприводах. Таким образом решалось несколько проблем сразу. При передвижении по поверхности под большим углом оператор робота может разложить колеса и робот станет намного устойчивее, а также повысится его проходимость. Также для преодоления высоких препятствий необходимо будет сложить 3ю пару колес и заехать передней частью робота на возвышенность, после чего третьей парой колес поднять заднюю часть робота.

Для основной ходовой части были выбраны коллекторные моторы постоянного тока с редуктором, выходная скорость которых составляла 120 об/мин. А колеса были выбраны диаметром 12см. Исходя из данных параметров можно было рассчитать приблизительную скорость робота.

Рисунок 6. Колеса для робота

Периметр колеса: P=2πR P=2*3.14*0.00012=7.5*10^-4 км

Оборотов колеса в час: 120*60=7200

Скорость робота U=7200*7.5*10^-4=5.4 км/ч

Следующим этапом необходимо было подобрать драйвера для управления моторами. В ходе исследования зависимости силы тока от нагрузки на мотор было выяснено что один мотор в пике потребляется около 3-4А. После оценки всех параметров разных моделей драйверов было решено остановиться на одноканальном драйвере BTS7960. Максимально на один канал они могут выдавать ток до 43А, чего вполне хватит. Для полноценного управления роботом необходимо было 2 драйвера – по одному драйверу на 2 основных мотора. Тем самым можно было управлять роботом по танковой схеме. Подключив драйвер к пинам МК с ШИМ можно управлять не только направлением вращения моторов, но и скоростью их вращения, благодаря чему появляется возможность сделать несколько режимов работы робота.

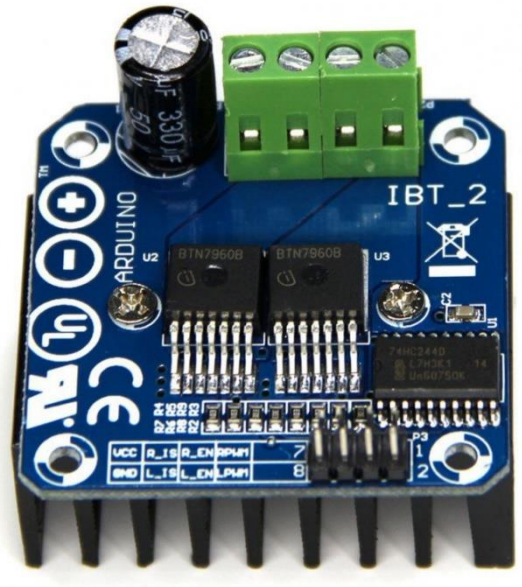


Рисунок 7. Драйвер моторов BTS7960

Далее необходимо было выбрать основной микроконтроллер (далее - МК), который бы обрабатывал всю информацию и управлял всеми узлами. Наиболее подходящим вариантом оказалась плата Arduino MEGA 2560 она имеет достаточное для подключения всех узлов количество пинов, а именно: 54 цифровых пина, 15 пинов с поддержкой ШИМ и 16 аналоговых входов, также важные интерфейсы, такие как SPI, UART и I2C.

Для обмена данными между роботом и пультом управления был выбран Радиомодуль nrf24l01 – он имеет интерфейс подключения SPI и радиус действия до 1000м метров по прямой видимости.

Рисунок 8. Радиомодуль Nrf24l01+

Для передачи и получения видеоизображения решено было использовать fpv системы, состоящие из камер, передатчиков и мониторов.

Для манипулятора были выбраны сервоприводы с крутящим моментом 35 кг*см, также они имели удобное крепление, которое позволяет быстро и надежно соединить несколько секций манипулятора. Для вращения вспомогательных пар колес были выбраны сервоприводы на 40 кг*см это сервоприводы с максимальной мощностью в стандартном корпусе. Использование более крупных и мощных сервоприводов отрицательно сказалось бы на весе и габаритах робота.

Для 2х осевого подвеса камеры были выбраны сервоприводы меньшего формата, т.к. особой нагрузки на этот узел не предполагалось.

Все электронные компоненты также должны получать питание определенного напряжения и достаточного тока для полноценного функционирования всех элементов. Чтобы решить эту проблему необходимо подключить несколько преобразователей напряжения, в данном случае все преобразователи будут понижающими т.к. напряжение источника питания превышает все требуемые напряжения. Для питания логики всех элементов необходимо 5В, для сервоприводов от 6.8 до 7.2 В, а для

остальных компонентов 12В. Чтобы равномерно распределить питание в цепь были добавлены несколько понижающих преобразователей, а также плата распределения питания для удобного подключения всех модулей. Было выбрано 3 модели понижающих преобразователей с наиболее подходящими характеристиками.

Первый преобразователь это XL4015 – понижающий DC-DC преобразователь с максимальным выходным током до 5А. В роботе будут использоваться несколько таких преобразователей. Явным преимуществом является ручная настройка выходного напряжения в диапазоне от 1.25В до 36В, что как раз подходит под компоненты, выбранные для робота.

Рисунок 9. Понижающий DC-DC преобразователь XL4015

Также в роботе использовался преобразователь UBEK Power System 7.2В, 5А(max8А) этот преобразователь питал одну из самых энергозатратных систем робота – манипулятор.

В последнюю очередь, исходя из общего потребления нужно выбрать элемент питания, который сможет обеспечить все узлы робота стабильным питанием. В качестве аккумуляторной батареи был выбран 4S Li-PO аккумулятор с емкостью 5200mah и пиковым током в 100А.

2.3.Проектирование

После подборки некоторых основных компонентов для сборки робота начался процесс проектирования самой конструкции робота.

Особенностью конструкции робота являлись подвижные вспомогательные колеса, независимый передний мост и 2х осевой подвес для камеры.

3Д модель робота создавалась в программе компас 3Д. Такой подход помог грамотно разместить все компоненты внутри робота, рассчитать примерную массу устройства.

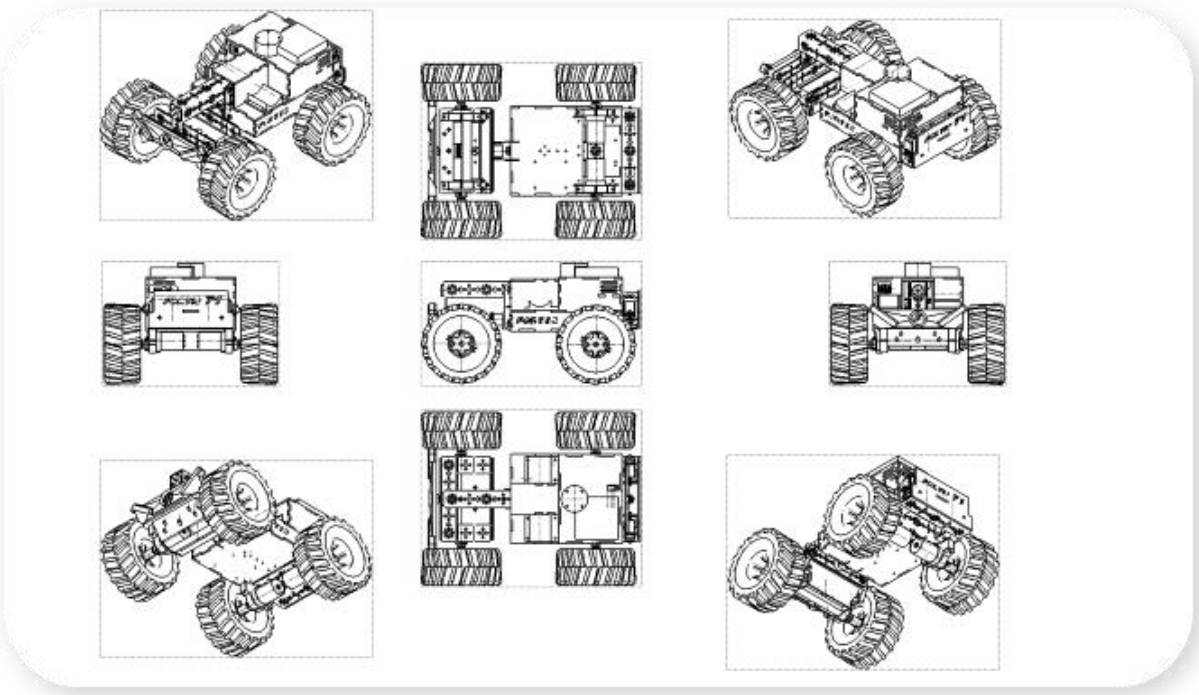


Рисунок 10. Чертеж робота с разных сторон

Основными материалами для создания робота были выбраны несколько материалов.

Для создания «скелета» робота были выбраны алюминиевые профиля. Благодаря их небольшому весу и высокой прочности рама робота будет прочной и легкой.

Вторым материалом была выбрана фанеры толщиной 3мм. Она послужит для обшивки корпуса и повысит прочность робота, а также защитит электронные узлы от внешних воздействий. Также фанеру легко обрабатывать, а вырезать все необходимые элементы можно с помощью лазерного ЧПУ станка, предварительно конвертировав чертежи в DXF формат. Загрузка чертежей происходила в программу Rdworks.

Последним материалом для создания некоторых элементов стал пластик, а именно PLA пластик. Он прост в использовании и с помощью 3Д принтера можно

получить сложные и прочные детали за небольшой промежуток времени. Из пластика создавались такие элементы, как 2х осевой подвес для камеры, бампер, захват манипулятора и некоторые другие части робота. В качестве слайсера для 3д принтера использовалась программа Ultimaker Cura

После проектирования начался этап изготовления всех недостающих деталей. Для этого применялись технологии лазерной резки (для изготовления необходимых деталей по чертежам из фанеры), а также аддитивные технологии для создания наиболее сложных узлов.

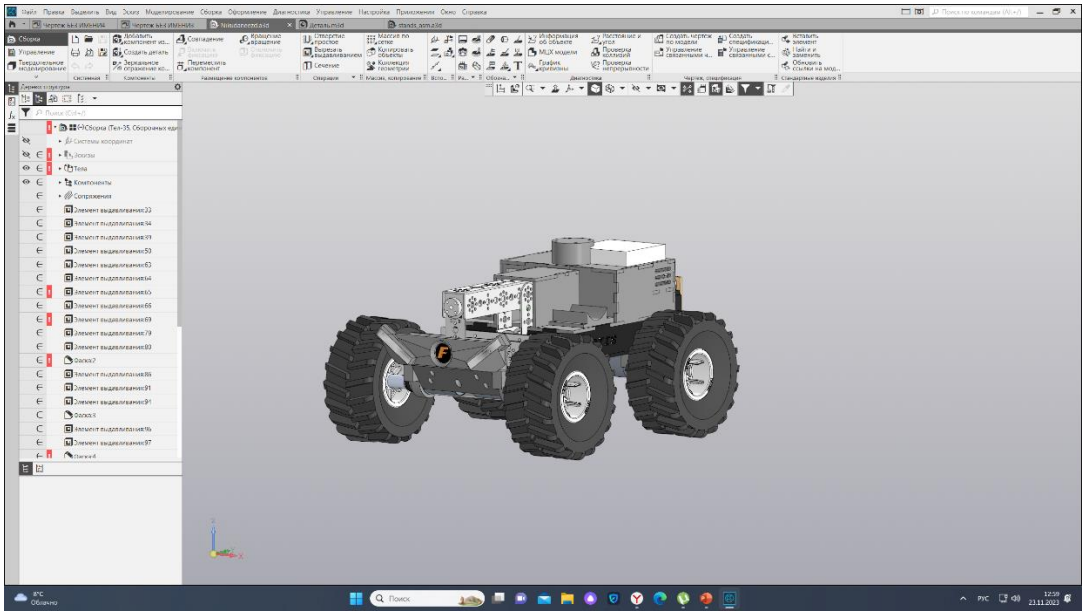


Рисунок 11. Частичная сборка робота в программе Компас 3Д

2.4.Сборка робота

Одним из самых сложных этапов был этап сборки и подключения всех узлов. Необходимо было соединить все изготовленные детали в единое устройство, при этом параллельно соединяя электронные узлы между собой. В процессе сборки преимущественно использовались винтовые крепления, благодаря чему конструкция получилась модульной и замена какого – либо узла не займет много времени. Все соединение также были обработаны фиксатором резьбы для предотвращения произвольного ослабления соединений из-за сильных вибраций.

В первую очередь было собрано основание ходовой части. Для сборки основания послужили алюминиевые профиля, уголки и крепеж. Непростой задачей было собрать передний мост робота, т.к. он должен был свободно перемещаться вокруг оси, позволяя

сохранять контакт с поверхностью практически на любой поверхности и при любых неровностях. Также на этом этапе были поставлены электромоторы и проведены провода от них до места размещения драйверов.

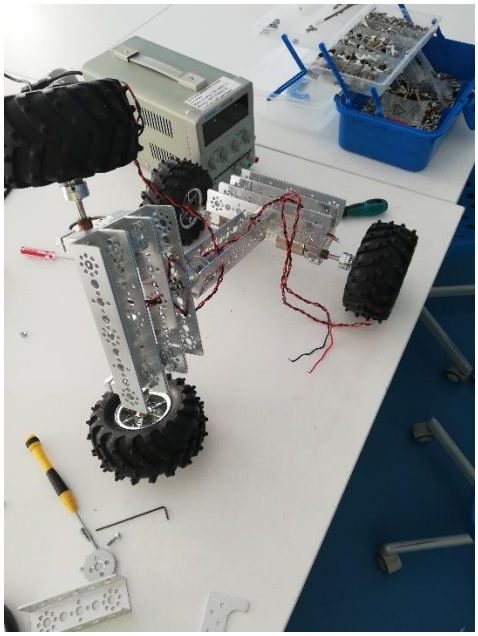


Рисунок 12. Первый этап разработки робота

Следующим этапом стала обшивка корпуса предварительно покрашенными деталями из фанеры. А также установка всех основных узлов электроники и соединение их между собой. Обшивка основания робота началось с нижней части робота, образовав при этом места для размещения первого уровня электроники – это преобразователи напряжения, а также реле с дистанционным управлением на которые ток поступал напрямую с аккумулятора. Была спаяна первая плата распределения питания, которая позволяла припаять провода от входов всех преобразователей.

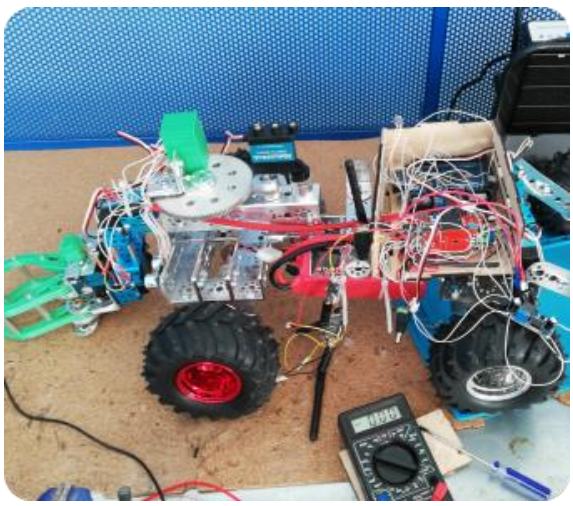


Рисунок 13. Одна из начальных стадий разработки

После сборки этого узла был установлен следующий слой обшивки, который являлся основанием для главного бокса с электроникой. Сборка следующего узла электроники началась с сборки 3-х уровневой платы распределения питания. Плата состояла из 3х макетных плат с клеммами, расположенных друг над другом. Таким образом на первом уровне все клеммы выдавали напряжение напрямую с аккумулятора для подсоединения силовых узлов. На следующем уровне находились клеммы на которые подавалось напряжение с выхода 12 – вольтового преобразователя. А последний уровень отвечал за питание всех логических элементов и имел напряжение 5В. Данная плата позволила удобно и быстро менять или подключать новые устройства и узлы к роботу, а также проводить диагностику питания.

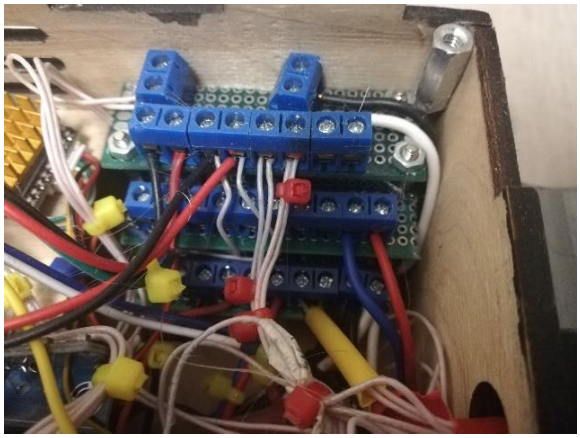


Рисунок 14. 3-х уровневая плата распределения питания

Далее в главный отсек электроники были установлены 2 драйвера моторов, основная плата – Arduino MEGA 2560, модуль радиосвязи -Nrf24l01, а также видеопередатчик. Помимо этого был установлен гироскоп для отслеживания положения робота в пространстве.

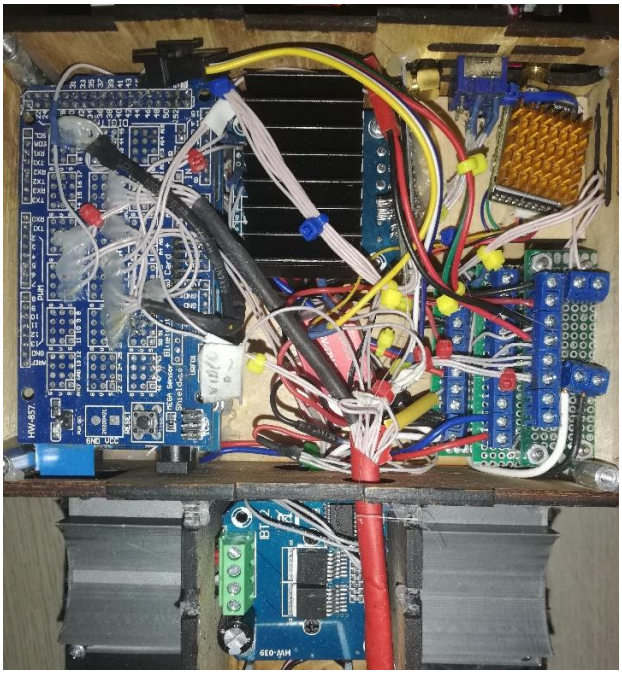


Рисунок 15. Отсек электроники

Заключительным этапом стала разработка манипулятора и его подключение. Для увеличения области работы манипулятора было решено сделать 3 степени свободы. Поворотный механизм манипулятора осуществляется через понижающую передачу, что увеличивает крутящий момент сервопривода. Клешня манипулятора была спроектирована и напечатана на 3д принтере. Крутящий момент с сервопривода передается через зубчатую передачу на 2 части клешни манипулятора – правую и левую. Для питания манипулятора был дополнительно установлен понижающий преобразователь на 7.2В.

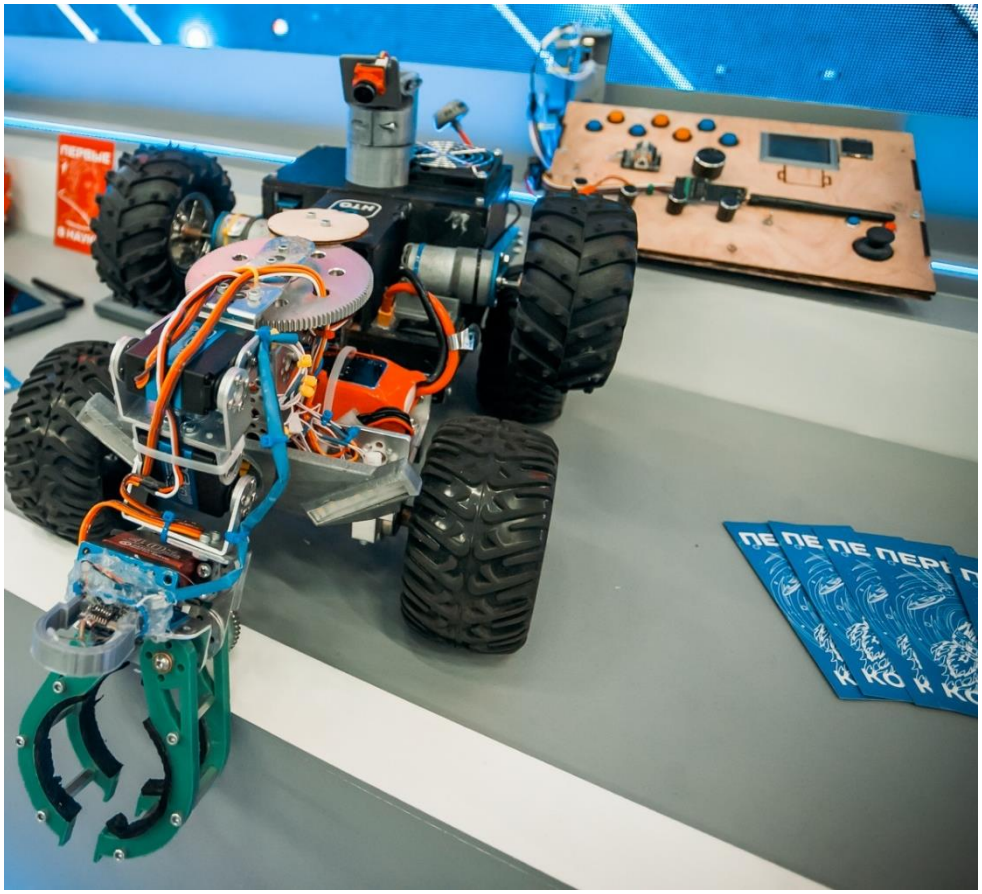


Рисунок 16. готовая модель робота

Помимо самого робота необходимо было собрать и аппаратуру для управления. Чтобы повысить удобство использования именно этого робота я решил собрать собственную аппаратуру. За главный МК также взял Arduino MEGA 2560. Радиопередатчик также выбрал аналогичный, установленному на роботе – nrf24l01. Основными элементами пульта стали потенциометры, джойстики и тумблеры, которые отвечали за управление всеми узлами робота. Для вывода информации телеметрии использовался дисплей. Корпус пульта был изготовлен из фанеры. Питался пульт от двух аккумуляторов 18650.

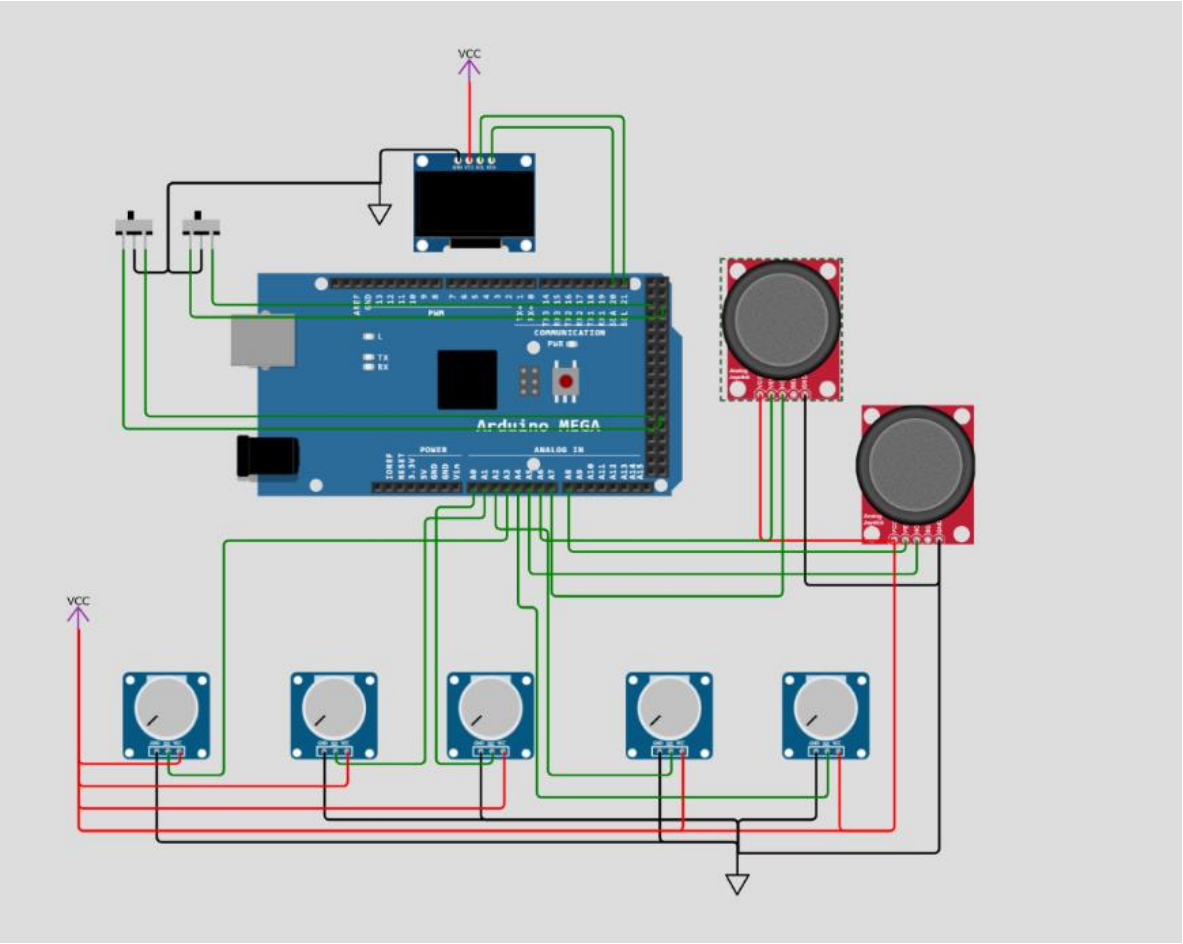


Рисунок 17. Электронная схема подключения пульта.

После сборки робота и пульта была составлена таблица со всеми компонентами и примерной себестоимости робота.

Таблица 1. Расчет себестоимости робота.

Н	Наименование	Стоимость за единицу (руб.)	Кол-во	Общая стоимость (руб.)
1	Arduino MEGA 2560	1000	2	2000
2	Драйвер моторовBTS7960	600	2	1200
3	Радиомодуль Nrf24l01	200	2	400
4	Коллекторные моторы с редуктором	3000	4	12000
5	Колеса с покрышками	1299	6	7794

6	Сервоприводы 35 кг с креплениями	2199	2	4398
7	Сервоприводы 40 кг	1200	4	4800
8	Понижающий преобразователь XL4015	100	2	200
9	Преобразователь понижающий UBEC 7.2B 5A	1590	2	3180
10	Fpv система с мониторами	8999	2	17998
11	Алюминиевые шестерни (набор)	600	1	600
12	Алюминиевые профиля	500 руб/м	2	1000
13	PLA пластик	500	1	500
14	Фанера	893	1	893
15	Платы макетные (набор)	569	1	569
16	Набор крепежа	1499	1	1499
17	Светодиодная лента	300 руб/м	0.5	150
18	Припой	300руб/ 100гр.	5	1500
19	Провода МГТФ	60руб/м	30	1800
20	Гироскоп	150	1	150
21	Шилд для Arduino mega	300	1	300
22	Стяжки для проводов	199	2	398
23	Краска аэрозольная (черная)	399	1	399
24	Джойстик	99	2	198
25	Потенциометры	30	5	150
26	Кнопки с фиксацией	50	7	350
27	Тумблеры трехпозиционные	65	2	70
28	Аккумулятор Li-Po 4s 16.8V	3500	1	3500
29	Аккумуляторы 18650	200	2	400
			ИТОГ:	67046

2.5.Написание программы для МК, калибровка и отладка всех узлов

2.5.1. программа для робота

Последним этапом перед полноценными испытаниями робота было написание программы. Для написания программы использовалась среда разработки Arduino IDE. В основу программы лег пример из библиотеки радиомодуля Nrf24l01 – работа в двустороннем режиме, то есть сам робот как принимал данные с пульта управления, также и отсылал на него данные телеметрии такой подход упрощает управление роботом и помогает отслеживать состояние робота. Данные отправляются в массиве, каждый элемент которого отвечает за работу определенного узла. После получения массива с данными МК обрабатывает их и в соответствии с алгоритмом отправляет сигналы на узлы робота.

2.5.2. программа для пульта управления роботом

В основу программы пульта также лег пример с отправкой и приемом информации. В основном цикле программы МК опрашивал все устройства, подключенные к портам и в зависимости от тех или иных значений заполнял данные массива, после чего отправлял их на робота. В ответ принимал данные телеметрии. Основную роль играл джойстик, который отвечал за направление движения робота. Также для удобства управления роботом имеля тумблер с тремя положениями, который отвечал за переключение скоростей. Т.к. в зависимости от задачи роботу необходимо двигаться либо с большой скоростью, чтобы быстрее преодолеть участок пути или же наоборот – для выполнения точных заданий робот должен был двигаться очень медленно и точно, чтобы оператор мог своевременно отреагировать на изменения окружающей среды и внести корректировки в работу робота. На пульте также имелся маленький экран, на которые выводились некоторые данные телеметрии. Также написав систему тригонометрических уравнений удалось воспроизвести текущее положение манипулятора на экран монитора с помощью 2х прямых, неразрывно соединенных одним узлом, в свою очередь данные для визуализации МК брал с потенциометров, которые отвечали за перемещение манипулятора. Благодаря этому оператор мог быстро сориентироваться в каком положении находится

манипулятор робота и внести какие – либо корректировки в его положение. Помимо основного джойстика на пульте находится и второй джойстик с помощью которого оператор может поворачивать ходовую камеру для изучения местности вокруг робота.

Основные данные, которые передавались с пульта управления на робота:

- Информация о направлении движения
- 4 элемента массива занимали данные и положении каждого из узлов
- Информация о положении вспомогательных колес
- Скорость робота
- 2 элемента массива содержали информацию о положении 2х осевого подвеса с камерой
- Также имеются несколько элементов массива для вспомогательных функций (например включение/выключение оптики робота)

2.6.После полной сборки были проведены тесты устройства

В первую очередь были проверены все функции, заложенные в программу робота. В результате тестирований была составлена тестированная таблица.

Таблица 2. Тестированная таблица

Н теста	Название теста	Итог
1	Движение робота по различным траекториям на 1 скорости	Робот передвигается медленно, как и было заложено в программе.
2	Движение робота по различным траекториям на 2 скорости	Робот имеет достаточно хорошую маневренность, данная скорость является основной
3	Движение робота по различным траекториям на 3 скорости	Ходовая робота выдает максимума, робот очень быстро передвигается,

		моторы слегка нагреваются пре длительной езде. Батарея садится в несколько раз быстрее
4	Движение манипулятора во всех направлениях	Манипулятор двигается в соответствии с заложенным алгоритмом
5	Захват и удержание груза, весом 1кг на вытянутом манипуляторе	Груз был с легкостью захвачен и поднят, из-за сильно инерции при резких изменениях положения манипулятора сервоприводы незначительно греются.
6	Откидывание вспомогательной пары колес	Сервоприводы справляются с весом моторов и коле на конце рычага
7	Поднятие задней части робота с помощью вспомогательных колес.	Робот поднимается, сервоприводы при длительном нахождении в таком положении незначительно греются
8	Тест дистанции	Радиосигнал имеет устойчивое соединение в условиях прямой видимости до 300 м при идеальных условиях. В условиях застройки около 50-80 м.

9	Попытка закрыть вентиль манипулятором	Благодаря удобному управлению манипулятором за минуту удается перевести в закрытое положение рычаг вентиля
10	Подъем робота по лестнице	

Рисунок 18. Тест закрытия вентилей роботом

Рисунок19. Тестовый подъем робота по лестнице

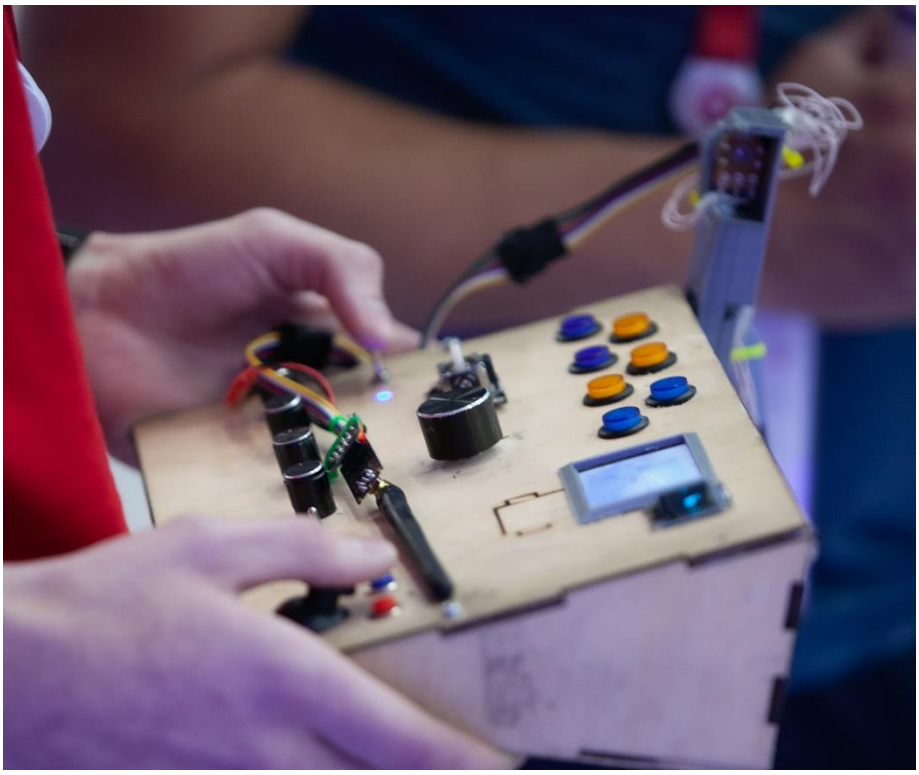


Рисунок 20. Готовый пульт управления

В ходе основных тестов были подтверждены изначально заложенные технические характеристики робота. А также подтверждена работоспособность и надежность всех узлов робота.

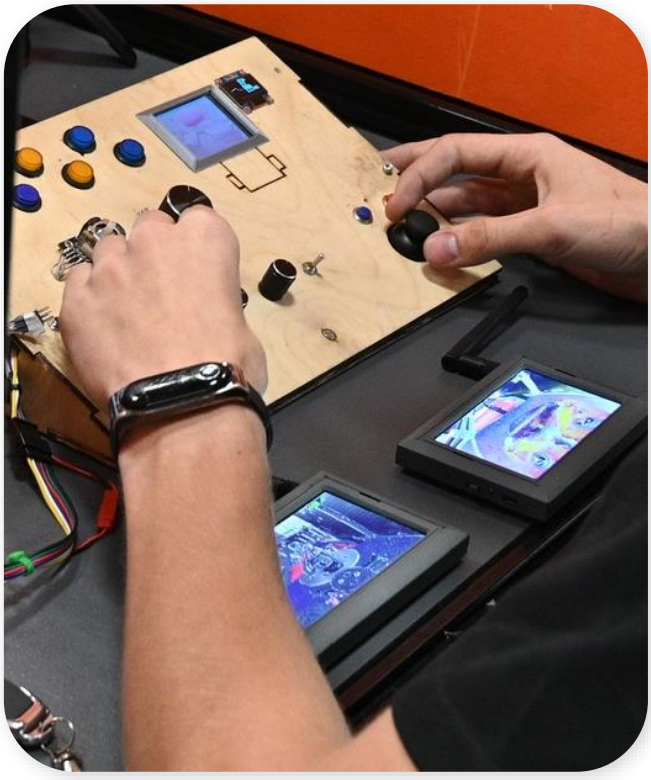
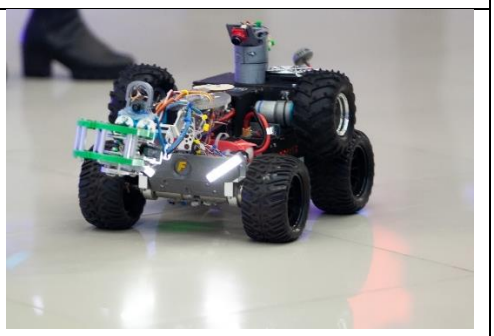


Рисунок 20. Тестирование дальности связи робота

2.7.Сравнение разработки с уже существующими роботами.

Как уже говорилось раннее на данный момент уже существуют рабочие прототипы со схожей областью применения, но отличающимися параметрами.

Название	Мобильный РТК Teodor	Робот для выполнения спец. Заданий Fortis
Фотография		
Вес, кг	375	8
Грузоподъёмность манипулятора, кг	30	3
Грузоподъемность робота, кг	100	10
Макс. Скорость, км/ч	3	5
Кол-во камер	4	2
Возможность отдельного поворота камеры	нет	да
габариты	1,3 *0,68 *1,24	0.45*0.3*0.25
Макс. Угол подъема, град	45	50
Вес аппаратуры управления, кг	Более 9	Менее 1 кг
Время работы	-	До 2 ч

Проведя сравнения с РТК Teodor можно заметить, что разработанный прототип отличается значительно меньшим весом. Как самого робота, так и аппаратуры, также обгоняет и по максимальной скорости, что немаловажно для преодоления дистанций.

РТК Teodor далеко не единственный робот, созданный для помощи спасателям и военным, но по характеристикам наиболее приближен к созданному образцу. Помимо преимуществ разработанный робот имеет и ряд недостатков, которые обусловлены меньшими размерами.

Данные с сравнительной таблице роботы могут использоваться как совместно, выполняя какую либо задачу вместе, перекрывая недостатки друг друга. Так и имеют возможность самостоятельно выполнять некоторые задачи, которые отличаются по масштабу и содержанию.

Более подробно ознакомиться со всеми функциями, а также убедиться в работоспособности робота можно, посмотрев видео (в видео представлен один из прототипов): https://drive.google.com/file/d/1Nj-OgKrdC5ejn9itGhQDuAPFNiHqC0ks/view?usp=sharing

2.8.Практическое применение.

Разработанный робот благодаря наличию ряда преимуществ может использоваться в различных сферах для выполнения разного рода задач. В первую очередь получилось создать робота, в соответствии с изначально поставленными целями, а следовательно робот может использоваться в сферах, для которых изначально и создавался. Это и помощи сотрудникам МЧС в разведки местности, обследование завалов, мониторинг радиоактивного и химического состояния окружающей среды, доставка небольших грузов, перекрытие вентилей и отключение различных коммуникаций в зоне ЧС. Также робот может стать незменимым устройством в военной сфере. Его можно использовать для разведки местности, разведки помещений, разминирования, доставки грузов, отвлечение внимания противника, установки ретрансляторов.

В итоге удалось создать робота в соответствии с поставленными задачами. Поставленную в начале работы цель можно считать достигнутой.

3. Условные обозначения и сокращения

ЧПУ	Числовое программное управление
МК	Микроконтроллер
PLA	Polylactic Acid
САПР	Система автоматизации проектных работ
IDE	Integrated Development Environment
РТК	Робототехнический комплекс

4. Список использованных источников

1. Действия МЧС при поисково-спасательных работах в условиях завалов. https://fireman.club/statyi-polzovateley/avariyno-i-poiskovo-spasatelnyie-rabotyi-v-usloviyah-zavalov/
2. Основные модели и характеристики комплексов РТК https://fireman.club/statyi-polzovateley/robototehnicheskie-kompleksyi-mchs-osnovnyie-modeli-opisanie-i-tth/
3. Сайт мчс РФ https://mchs.gov.ru/deyatelnost/press-centr/vse-novosti/4253837
4. Роботы для МЧС https://dzen.ru/a/ZGdW2BqVYHc2VsCf
5. Информация про робота Teodor https://vk.com/wall-191160544_1005 , http://otvaga2004.ru/na-zemle/na-zemle-11/modern_land_robots_6/
6. Мякишев Г. Я. Физика : Электродинамика : Углублённый уровень : 10—11 классы : учебник / Г. Я. Мякишев, А. З. Синяков. — 8-е изд., стереотип. — М. : Дрофа, 2019. — 476 с.
7. Сайт интернет магазина электроники https://amperka.ru/
8. Описание и характеристики ардуино мегаhttps://all-arduino.ru/product/arduino-mega-2560/
9. Использование и принцип работы радиомодуля https://3d-diy.ru/wiki/arduino-moduli/radio-modul-nrf24l01/

5. Приложения

А. Листинг кода для робота

Fortis.ino

```
//БИБЛИОТЕКИ

#include<Wire.h>
#include <SPI.h>
#include "nRF24L01.h"
#include "RF24.h"
#include <Servo.h>
#include <Average.h>

//ПИНЫ

#define  in1 5
#define  in2 4
#define  in3 3
#define  in4 2
#define in1z 28
#define in2z 29
#define in3z 30
#define in4z 31
#define Rl A1
#define Ll A0
#define laser 26
#define prepatstvie 27
#define led 7

//ОБЪЕКТЫ

Servo servo1;
Servo servo2;
Servo servo3;
Servo servo4;
Servo servo5;
Servo servo6;
Servo servo7;
Servo servo8;
Servo servo9;
RF24 radio(48, 49);

//ПЕРЕМЕННЫЕ


const int MPU_addr=0x68;
int16_t AcX,AcY,AcZ,Tmp,GyX,GyY,GyZ;
int minVal=265;
int maxVal=402;
double x;
double y;
double z;
//
```

```
byte pipeNo;
byte address[][6] = {"1Node", "2Node", "3Node", "4Node", "5Node", "6Node"}; // возможные номера труб
byte potValue[13];   // массив принятых данных
int telemetry[4];       // массив данных телеметрии


byte a = 0;
byte sped;
int RL;
int LL;
int m = 0;
int Rdistance;
unsigned long timer1 = 0;
unsigned long timer2 = 0;
unsigned long timer3 = 0;
bool flag2 = 0;
bool mayk = 0;


void setup() {

 Wire.begin();
 Wire.beginTransmission(MPU_addr);
 Wire.write(0x6B);
 Wire.write(0);
 Wire.endTransmission(true);

 pinMode(in1, OUTPUT);
 pinMode(in2, OUTPUT);
 pinMode(in3, OUTPUT);
 pinMode(in4, OUTPUT);
 pinMode(in1z, OUTPUT);
 pinMode(in2z, OUTPUT);
 pinMode(in3z, OUTPUT);
 pinMode(in4z, OUTPUT);
 pinMode(laser, OUTPUT);
 pinMode(led, OUTPUT);
 pinMode(prepatstvie, INPUT);



 servo1.attach(41);
 servo2.attach(39);
 servo3.attach(38);
 servo4.attach(40);
 servo5.attach(32);
 servo6.attach(33);
 servo7.attach(37);
 servo8.attach(36);
 servo9.attach(42);
```

```
 radio.begin();                      // активировать модуль
 //radio.setAutoAck(1);                // режим подтверждения приёма, 1 вкл 0 выкл
 radio.setRetries(0, 15);            // (время между попыткой достучаться, число попыток)
 radio.enableAckPayload();           // разрешить отсылку данных в ответ на входящий
сигнал
 radio.setPayloadSize(32);           // размер пакета, байт
 radio.openReadingPipe(1, address[0]); // хотим слушать трубу 0
 radio.setChannel(0x51);             // выбираем канал (в котором нет шумов!)
 radio.setPALevel(RF24_PA_MAX);      // уровень мощности передатчика
 radio.setDataRate(RF24_2MBPS);      // скорость обмена должна быть одинакова на приёмнике
и передатчике!
 radio.powerUp();         // начать работу
 radio.startListening();  // начинаем слушать эфир, мы приёмный модуль

 digitalWrite(in1, LOW);
 digitalWrite(in2, LOW);
 digitalWrite(in3, LOW);
 digitalWrite(in4, LOW);
 digitalWrite(laser, LOW);

 servo1.write(80);
 servo2.write(70);
 servo3.write(90);
 servo4.write(90);
 //servo5.write(180);
// servo6.write(90);
 servo7.write(90);
 servo8.write(90);
 servo9.write(180);


 digitalWrite(in1z, LOW);
 digitalWrite(in2z, LOW);
 digitalWrite(in3z, LOW);
 digitalWrite(in4z, LOW);
}

void loop() {

 while (radio.available(&pipeNo)) {     // слушаем эфир
    radio.read(&potValue, sizeof(potValue));  // чиатем входящий сигнал


    //if(potValue[15] == 0){
     //goDistance();

     if(potValue[12] == 2){  // автозахват
       digitalWrite(laser, HIGH);

       if(digitalRead(prepatstvie) == 1 && mayk == 0){
         digitalWrite(in2, LOW);
         analogWrite(in1, 50);
```

```
   digitalWrite(in4, LOW);
   analogWrite(in3, 50);
   servo1.write(80);
  }
  else{
    delay(150);
    digitalWrite(in1, LOW);
    digitalWrite(in2, LOW);
    digitalWrite(in3, LOW);
    digitalWrite(in4, LOW);
    servo1.write(175);
    delay(500);
    //servo2.write(90);
    servo2.write(20);
    mayk = 1;
  }
}
else{
    x = giro();
  if (potValue[5] == 1) {   //линия
     runLine();
  }
  else {
    if(potValue[5] == 3){ // автономный подъём
      x = giro();
      up();
    }
    else{
      if (potValue[5] == 2){ // автономный спуск
        x = giro();
        down();
      }
      else{
         if (potValue[5] == 4){ // фары
        analogWrite(led, 30);
      }
      else{
        analogWrite(led, 0);
      }
       if (potValue[8] == 3) {
         sped = 255;
        }
        else if (potValue[8] == 2) {
          sped = 125;
        }
        else if (potValue[8] == 1) {
          sped = 50;
        }
        servo1.write(potValue[1]);
        servo2.write(potValue[2]);
        servo3.write(potValue[3]);
        servo4.write(potValue[4]);
```

```
servo5.write(potValue[11]-6);
servo6.write(map(potValue[11],180,0,0,180));
servo7.write(potValue[9]);
servo8.write(potValue[10]);

if(potValue[13] == 1){
  digitalWrite(in1z, LOW);
  digitalWrite(in2z, LOW);
  digitalWrite(in3z, LOW);
  digitalWrite(in4z, LOW);
}
if(potValue[12] == 1){
  digitalWrite(laser,HIGH);
  mayk = 0;
}
else{
  digitalWrite(laser,LOW);
   mayk = 0;
}
switch (potValue[0]) {
  case 1:
    digitalWrite(in2, LOW);
    analogWrite(in1, sped);
    digitalWrite(in4, LOW);
    analogWrite(in3, sped);
    if(potValue[6] == 0){
      digitalWrite(in1z,LOW);
      digitalWrite(in2z, HIGH);
      digitalWrite(in3z, LOW);
      digitalWrite(in4z,HIGH);
    }
    else{
      digitalWrite(in1z, LOW);
      digitalWrite(in2z, LOW);
      digitalWrite(in3z, LOW);
      digitalWrite(in4z, LOW);
    }
  break;
  case 10:
    digitalWrite(in2, LOW);
    analogWrite(in1, 50);
    digitalWrite(in4, LOW);
    analogWrite(in3, 50);
    if(potValue[6] == 0){
      digitalWrite(in1z,LOW);
      digitalWrite(in2z, HIGH);
      digitalWrite(in3z, LOW);
      digitalWrite(in4z,HIGH);
    }
    else{
      digitalWrite(in1z, LOW);
      digitalWrite(in2z, LOW);
```

```
    digitalWrite(in3z, LOW);
    digitalWrite(in4z, LOW);

  }
break;
case 0:
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
  digitalWrite(in3, LOW);
  digitalWrite(in4, LOW);
  digitalWrite(in1z, LOW);
  digitalWrite(in2z, LOW);
  digitalWrite(in3z, LOW);
  digitalWrite(in4z, LOW);
break;
case 2:
  digitalWrite(in1, LOW);
  analogWrite(in2, sped);
  digitalWrite(in3, LOW);
  analogWrite(in4, sped);
  if(potValue[6] == 0){
    digitalWrite(in1z,HIGH);
    digitalWrite(in2z, LOW);
    digitalWrite(in3z, HIGH);
    digitalWrite(in4z,LOW);
  }
  else{
    digitalWrite(in1z, LOW);
    digitalWrite(in2z, LOW);
    digitalWrite(in3z, LOW);
    digitalWrite(in4z, LOW);
  }
break;
case 3:
  digitalWrite(in1, LOW);
  analogWrite(in2, sped);
  analogWrite(in3, sped);
  digitalWrite(in4, LOW);
break;
case 4:
  analogWrite(in1, sped);
  digitalWrite(in2, LOW);
  digitalWrite(in3, LOW);
  analogWrite(in4, sped);
break;
case 5:
  analogWrite(in1, sped);
  digitalWrite(in2, LOW);
  analogWrite(in3, sped*50/100);
  digitalWrite(in4, LOW);
break;
case 8:
```

```
                analogWrite(in1, sped*50/100);
                digitalWrite(in2, LOW);
                analogWrite(in3, sped);
                digitalWrite(in4, LOW);
              break;


          }
        }
      }
    }
  }

  telemetry[3] = giro();
   telemetry[0] = analogRead(Rl);
   telemetry[1] = analogRead(Ll);
    telemetry[2]=telemetry[2]+1;
    radio.writeAckPayload(pipeNo, &telemetry, sizeof(telemetry));


  }
}




/*int distance() {
  ave.push(27.728*1.3 * pow(map(analogRead(A2), 0, 1023, 0, 5000) / 1000.0, -1.2045));
  return(ave.mean());
}

void goDistance() {
  if((ave.mean()) > 30){
    digitalWrite(in2, LOW);
    analogWrite(in1, 50);
    digitalWrite(in4, LOW);
    analogWrite(in3, 50);
  }
  else{
   digitalWrite(in1, LOW);
   digitalWrite(in2, LOW);
   digitalWrite(in3, LOW);
   digitalWrite(in4, LOW);
  }
}*/

void runLine() {
  RL = analogRead(Rl);
  LL = analogRead(Ll);
  if (RL < 830 && LL < 830) { // если ллинии нет, едем прямо
   analogWrite(in1, 50);
   digitalWrite(in2, LOW);
   analogWrite(in3, 50);
```

```
   digitalWrite(in4, LOW);
  }
  else if (RL > 830 && LL < 830) { // если линяя на правом датчике, поворачиваем вправо
   analogWrite(in1, 50);
   digitalWrite(in2, LOW);
   digitalWrite(in3, LOW);
   analogWrite(in4, 50);
  }
  else if (LL > 830 && RL < 830) {  // если линяя на левом датчике, поворачиваем налево
   digitalWrite(in1, LOW);
   analogWrite(in2, 50);
   analogWrite(in3, 50);
   digitalWrite(in4, LOW);
  }
  else if (RL > 830 && LL > 830) { // если линяя на обоих датчиках, едем вперёд
   analogWrite(in1, 50);
   digitalWrite(in2, LOW);
   analogWrite(in3, 50);
   digitalWrite(in4, LOW);
  }
}

int giro() {
  Wire.beginTransmission(MPU_addr);
  Wire.write(0x3B);
  Wire.endTransmission(false);
  Wire.requestFrom(MPU_addr,14,true);
  AcX=Wire.read()<<8|Wire.read();
  AcY=Wire.read()<<8|Wire.read();
  AcZ=Wire.read()<<8|Wire.read();
  int xAng = map(AcX,minVal,maxVal,-90,90);
  int yAng = map(AcY,minVal,maxVal,-90,90);
  int zAng = map(AcZ,minVal,maxVal,-90,90);


  y= RAD_TO_DEG * (atan2(-xAng, -zAng)+PI);
  z= RAD_TO_DEG * (atan2(-yAng, -xAng)+PI);
  x = (RAD_TO_DEG * (atan2(-yAng, -zAng)+PI));
  return(x);
}

void up() {
if(x > 190){

  digitalWrite(in2, LOW);
  analogWrite(in1, 120);
  digitalWrite(in4, LOW);
  analogWrite(in3, 120);
  }
  else{
    digitalWrite(in2, LOW);
    analogWrite(in1, LOW);
```

```
    digitalWrite(in4, LOW);
    analogWrite(in3, LOW);
  }
}

void down() {
  if(x < 170){
    digitalWrite(in2, LOW);
    analogWrite(in1, 80);
    digitalWrite(in4, LOW);
    analogWrite(in3, 80);
  }
  else{
    digitalWrite(in2, LOW);
    analogWrite(in1, LOW);
    digitalWrite(in4, LOW);
    analogWrite(in3, LOW);

  }
}
```

Б. листинг кода для пульта управления.

Fortis_Pult.ino

```
//БИБЛИОТЕКИ
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <SPI.h>
#include "nRF24L01.h"
#include "RF24.h"


#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels


#define OLED_RESET     -1
#define SCREEN_ADDRESS 0x3C
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);


//ПИНЫ



#define pot1 A0
#define pot2 A1
#define pot3 A2
#define pot4 A3
```

```
#define pot5 A4
#define pinpoty A5
#define pinpotx A6
#define servox A7
#define servoy A8

#define tumbler3 24
#define tumbler1 25
//#define lGreen 28

//#define lRed 31
//#define rRed 32
#define green 33
#define red 35
#define blue 34
#define sw 36

#define linya 27
#define verh 29
#define niz 30
#define wh3 28
#define led 26



#define tumbler21 44
#define tumbler23 45


// ОБЪЕКТЫ


RF24 radio(48, 49); // "создать" модуль на пинах


// ПЕРЕМЕННЫЕ

int x1 = 0;
int y1 = 32;
int x2 = 84;
int y2 = 32;
int x3 = 84;
int y3 = 32;
int angle1 = 0;
int angle2 = 0;
int angle3 = 0;
int angle5 = 0;

int c;


byte address[][6] = {"1Node", "2Node", "3Node", "4Node", "5Node", "6Node"}; // возможные
номера труб
```

```
byte potValue[13];     // массив пересылаемых данных
int telemetry[4];         // массив принятых от приёмника данных телеметрии
byte rssi;
int trnsmtd_pack = 1, failed_pack;
unsigned long RSSI_timer;
byte oldcount = 0;



void setup() {
  display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS);
   display.clearDisplay();
   display.setTextSize(2); // указываем размер шрифта
   display.setTextColor(SSD1306_WHITE);

  pinMode(verh, INPUT_PULLUP);
  pinMode(linya, INPUT_PULLUP);
  pinMode(wh3, INPUT_PULLUP);
  pinMode(niz, INPUT_PULLUP);
  //   pinMode(button5, INPUT_PULLUP);
  pinMode(tumbler1, INPUT_PULLUP);
  pinMode(tumbler3, INPUT_PULLUP);
  pinMode(tumbler21, INPUT_PULLUP);
  pinMode(tumbler23, INPUT_PULLUP);
  pinMode(sw, INPUT_PULLUP);
  //  pinMode(rRed, OUTPUT);

  pinMode(red, OUTPUT);
  pinMode(green, OUTPUT);
  pinMode(blue, OUTPUT);

  radio.begin();                     // активировать модуль
  //radio.setAutoAck(1);                 // режим подтверждения приёма,
  radio.setRetries(0, 15);           // (время между попыткой достучаться, число попыток)
  radio.enableAckPayload();          // разрешить отсылку данных в ответ на входящий
сигнал
  radio.setPayloadSize(32);          // размер пакета, в байтах
  radio.openWritingPipe(address[0]);
  radio.setChannel(0x51);            // выбираем канал
  radio.setPALevel(RF24_PA_MAX);     // уровень мощности передатчика
  radio.setDataRate(RF24_2MBPS);     // скорость обмена
  radio.powerUp();                   // начать работу
  radio.stopListening();
}


bool flagspedL = false;
bool flagspedH = false;
 byte sped = 3;
byte ab;
unsigned long timer1 = 0;
unsigned long timer2 = 0;
unsigned long timer3 = 0;
```

```
unsigned long timer4 = 0;
unsigned long timer5 = 0;



void loop() {
 int servo1 = analogRead(pot1);
 int servo2 = analogRead(pot2);
 int servo3 = analogRead(pot3);
 int servo4 = analogRead(pot4);
 int servo5 = analogRead(pot5);
 int px = analogRead(servox);
 int py = analogRead(servoy);
 int x = analogRead(pinpotx);
 int y = analogRead(pinpoty);

if(digitalRead(sw) == LOW){
  if(ab == 0){
 oldcount = potValue[9];
 ab = 1;
  }
  potValue[9]= 180;

  }
else if(ab == 1){
   potValue[9] = oldcount;
   ab = 0;
}

 if(px > 650 && millis() - timer1 >= 50 && potValue[9] > 0){
  timer1 = millis();
  potValue[9] = potValue[9]-3;
 }
 if(px < 450 && millis() - timer2 >= 50 && potValue[9] <= 180){
  timer2 = millis();
  potValue[9]= potValue[9]+3;
 }

 if(py > 650 && millis() - timer3 >= 50 && potValue[10] <= 180){
  timer3 = millis();
  potValue[10]= potValue[10]+3;
 }
 if(py < 450 && millis() - timer4 >= 50 && potValue[10] > 0){
  timer4 = millis();
  potValue[10]= potValue[10]-3;
 }

  if(digitalRead(tumbler1) == LOW){
    sped = 1;
  }
   if(digitalRead(tumbler3) == LOW){
    sped = 3;
   }
```

```
   if(digitalRead(tumbler3) == HIGH && digitalRead(tumbler1) == HIGH){
        sped = 2;
   }

    if(digitalRead(tumbler21) == LOW){
   potValue[12] = 0;
  }
   if(digitalRead(tumbler23) == LOW){
    potValue[12] = 2;
   }
   if(digitalRead(tumbler23) == HIGH && digitalRead(tumbler21) == HIGH){
       potValue[12] = 1;
   }


 if(sped == 1){
    digitalWrite(red, LOW);
   digitalWrite(green, HIGH);
    digitalWrite(blue, LOW);
 }
 if(sped == 2){
digitalWrite(red, LOW);
   digitalWrite(green, LOW);
    digitalWrite(blue, HIGH);
 }
 if(sped == 3){
  digitalWrite(red, HIGH);
   digitalWrite(green, LOW);
    digitalWrite(blue, LOW);
 }
 if (y < 50 && x > 500 && x < 600){
  potValue[0] = 1;
 }
  if (y < 400 && y > 49 && x > 500 && x < 600){
  potValue[0] = 10;
 }
 if (y > 650 && x > 500 && x < 600){
  potValue[0] = 2;
 }
  if (x > 650 && y > 500 && y < 600){
  potValue[0] = 3;
 }
 if (x < 450 && y > 500 && y < 600){
  potValue[0] = 4;
 }
 if (x < 50 && y < 50){
  potValue[0] = 5;
 }
 if (x < 50 && y > 927){
  potValue[0] = 6;
 }
 if (x >927 && y > 927){
```

```
 potValue[0] = 7;
}
if (x > 927 && y < 50){
 potValue[0] = 8;
}
if (x > 401 && x < 599){
 if (y > 401 && y < 599) {
   potValue[0] = 0;
 }
}

potValue[1]=map(servo1,0,1023,180,0);
potValue[2]=map(servo2,0,1023,180,0);
potValue[3]=map(servo3,0,1023,180,0);
potValue[4]=map(servo5,0,1023,180,0);
if(digitalRead(linya) == 0){
   potValue[5]=1;
}
else if(digitalRead(niz) == 0){
  potValue[5]=2;
}
 else if(digitalRead(verh) == 0){
  potValue[5]=3;
}
else if(digitalRead(led) == 0){
  potValue[5]=4;
}
else{
  potValue[5]=0;
}

potValue[6]=digitalRead(wh3);/*
potValue[7]=digitalRead(verh);*/
potValue[8]=sped;

potValue[11] =map(servo4, 0,1023,0,180);


 if (radio.write(&potValue, sizeof(potValue))) {

   if (!radio.available()) {   // если получаем пустой ответ

   } else {
     while (radio.available() ) {                    // если в ответе что-то есть
       radio.read(&telemetry, sizeof(telemetry)); // читаем

     }
   }
 }


display.clearDisplay();
```

```
 angle1 = map(servo3,0,1023,180,0);
 angle2 = map(servo2,0,1023,180,0);
 c = map(servo2,0,1023,90,-90);

 display.drawLine   (x1, y1,  x2, y2,0);
 display.drawLine   (x2, y2,  x3, y3,0);

 x2 = 0 + sin(2*PI*angle2/360)*27;
 y2 = 32 - cos(2*PI*angle2/360)*27;
 display.drawLine   (x1, y1,  x2, y2,1);

 display.drawLine   (x2, y2,  x3, y3,0);
 x3 = x2 + sin(2*PI*(angle1+c)/360)*27;
 y3 = y2 - cos(2*PI*(angle1+c)/360)*27;

 display.drawLine   (x2, y2,  x3, y3,1);

 display.setCursor(60,0);
 display.println(telemetry[0]);
 display.setCursor(60,30);
 display.println(telemetry[1]);
 display.setCursor(60,45);
 display.println(telemetry[2]);
 display.setCursor(60,15);
 display.print(telemetry[3]);

  if(millis() - timer5 >= 300 ){
   timer5 = millis();
   display.display();
  }

}
```