|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА - Российский технологический университет»**  **РТУ МИРЭА** |

Институт кибернетики

*(наименование института, филиала)*

Кафедра проблем управления

*(наименование кафедры)*

**ОТЧЁТ ПО ПРАКТИКЕ ПО ППУ И ОПД**

(указать вид практики)

производственная

(указать тип практики)

**Тема практики:** **Разработка программного обеспечения и пользовательских интерфейсов для управления мобильным роботом.**

приказ университета о направлении на практику

от «10» февраля 2020 г. № 759-с

Отчет представлен к

рассмотрению:

Студент группы \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_Белянин С.О.\_\_\_\_\_

*(подпись) (расшифровка подписи)*

«\_\_» \_\_\_\_\_\_ 201\_г.

Отчет утвержден.

Допущен к защите:

Руководитель практики

от кафедры \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(подпись) (расшифровка подписи)*

«\_\_» \_\_\_\_\_\_ 201\_г.

Руководитель практики

от Университета \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.А. Сухоленцева

*(подпись) (расшифровка подписи)*

«\_\_» \_\_\_\_\_\_ 201\_г.

Москва 2020

**ОТЧЕТ**

**по практике ППУ и ОПД**

**студента 3 курса учебной группы КРБО-01-17**

**Института кибернетики**

(фамилия, имя и отчество)

1. Практику проходил с \_\_ \_\_\_\_\_\_ 201\_г. по \_\_ \_\_\_\_\_\_ 201\_г.

в \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(место прохождения практики)

1. Задание на практику выполнил

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(указать: в полном объеме или частично)

Не выполнены следующие задания:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(указать также причины невыполнения)

Подробное содержание выполненной на практике работы и достигнутые результаты: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Предложения по совершенствованию организации и прохождения практики:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_ (Белянин С.О.) «\_\_»\_\_\_\_\_\_\_\_\_\_20\_\_г.

(подпись) (фамилия и инициалы)

Заключение руководителя практики от профильной организации:

Приобрел следующие профессиональные навыки:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Проявил себя как: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Руководитель практики от профильной организации**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(наименование от профильной организации)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_

(должность) (подпись) (фамилия и инициалы)

**Отчет проверил:**

**Руководитель практики от Университета**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (фамилия и инициалы)

|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА - Российский технологический университет»**  **РТУ МИРЭА** |

Институт кибернетики

*(наименование института, филиала)*

Кафедра проблем управления

*(наименование кафедры)*

**ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ НА ПРАКТИКУ ПО ППУ И ОПД**

*(указать вид практики*)

**Студенту 3 курса учебной группы КРБО-01-17**

**Белянину Сергею Олеговичу**

**Место и время практики: аудитория Г-428 СКБ «Алгоритм»**

**Должность на практике: -**

**1. ЦЕЛЕВАЯ УСТАНОВКА:** **Разработка автономного мобильного малогабаритного робота.**

**2. СОДЕРЖАНИЕ ПРАКТИКИ:**

2.1 Изучить: существующие образцы мобильных робототехнических платформ с дистанционным управлением, схемотехнические и программно-алгоритмические решения, используемые в них;

2.2 Практически выполнить: программный комплекс, осуществляющий дистанционное управление в ручном и автономном режимах малогабаритной мобильной робототехнической платформой;

2.3 Ознакомиться: со средствами создания программного обеспечения для систем дистанционного управления.

**3.ДОПОЛНИТЕЛЬНОЕ ЗАДАНИЕ:** -

**4. ОГРАНИЗАЦИОННО-МЕТОДИЧЕСКИЕ УКАЗАНИЯ:** -

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Заведующий кафедрой:

«\_\_»\_\_\_\_\_\_\_\_\_\_20\_\_г. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ М.П. Романов

(подпись) (ФИО)

СОГЛАСОВАНО:

Руководитель практики от кафедры:

«\_\_»\_\_\_\_\_\_\_\_\_\_20\_\_г. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.А. Морозов

(подпись) (ФИО)

Руководитель практики от Университета:

«\_\_»\_\_\_\_\_\_\_\_\_\_20\_\_г.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.А. Сухоленцева

(подпись) (ФИО)

Задание получил:

«\_\_»\_\_\_\_\_\_\_\_\_\_20\_\_г. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ С.О. Белянин

(подпись) (ФИО)

Содержание

[Введение 6](#_Toc41588391)

1. [Мобильные роботы с дистанционным управлением](#_Toc41588392) 7

[2.Описание робота для практической работы 10](#_Toc41588393)

3. [Разработка програмного обеспечения](#_Toc41588394) 13

4. Проверка работоспособности программ и робота....................................14

[Вывод 17](#_Toc41588396)

Библиографический список............................................................................18

Приложение......................................................................................................19

**Введение**

Одной из не менее важных задач при создании мобильного робота является реализация управления и передачи информации. Во многом выбор способа и средств управления опирается на конструкцию робота и особенности рабочей среды. Например, малогабаритным роботам, работающим в закрытом помещении, будет достаточно модуля Wi-Fi или Bluetooth, в то время как тяжеловесные аппараты в некоторых случаях могут управляться с помощью системы с проводным подключением.

Целью данной работы является изучение существующих образцов мобильных роботов с дистанционным управлением, способов и алгоритмов управления, применение полученных знаний для разработки программного обеспечения, которое можно применить в работе с мобильным роботом.

1. **Мобильные роботы с дистанционным управлением**



Рисунок 1. Ровер Curiosity

Curiosity  —  марсоход третьего поколения размером с автомобиль, разработанный для исследования кратера Гейла на Марсе в рамках миссии NASA Марсианская научная лаборатория. Марсоход представляет собой автономную химическую лабораторию в несколько раз больше и тяжелее предыдущих марсоходов «Спирит» и «Оппортьюнити».

Управление марсоходом по большей части автономное, для передачи данных используется спутниковая система, которая состоит из Земных станций DSS (Deep Space Station), спутников-ретрансляторов, и три антенны самого ровера.

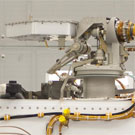


Рисунок 2. Аппаратные средва связи ровер Curiosity

UHF-антенна используется чаще всего. С ее помощью, ровер может передавать данные через спутники MRO и Odyssey на частоте около 400 мегагерц. Использование спутников для передачи сигнала является предпочтительным из-за того, что они находятся в поле зрения DSN-станций гораздо дольше, чем сам ровер, одиноко сидящий на поверхности Марса. К тому же, поскольку они значительно ближе к марсоходу, последнему нужно затрачивать меньше энергии для передачи данных. Скорость передачи может достигать 256кб/с для Odyssey и до 2 мбит/с для MRO. Б*о*льшая часть информации, приходящей от Curiosity, проходит именно через спутник MRO. Сама UHF-антенна находится в задней части ровера, и внешне выглядит как серый цилиндр.

Curiosity также имеет HGA, которую он может использовать для получения команд напрямую с Земли. Эта антенна подвижна (ее можно направить в сторону Земли), то есть для ее использования роверу не приходится менять свое местоположение, достаточно просто повернуть HGA в нужную сторону, а это позволяет сохранять энергию. HGA смонтирована примерно посередине с левого борта ровера, и представляет собой шестигранник диаметром около 30 сантиметров. HGA может передавать данные прямо на Землю со скоростью около 160 бит/сек на 34-метровые антенны, или со скоростью до 800 бит/сек на 70-метровые.

Наконец, третья антенна — это так называемая LGA.  
Она посылает и принимает сигналы в любых направлениях. Работает LGA в X-диапазоне (7-8 ГГц). Тем не менее, мощность этой антенны довольно мала, а скорость передачи оставляет желать лучшего. Из-за этого она в основном используется для приема информации, а не для ее передачи.  
На фото LGA — это белая башенка на переднем плане.  
На заднем плане видна UHF-антенна.  
Стоит отметить, что марсоход генерирует огромное количество научных данных, и не всегда все их удается отправить. Специалисты NASA устанавливают приоритеты важности: информация с наибольшим приоритетом будет передана в первую очередь, а информация с меньшим приоритетом будет ждать следующего коммуникационного окна. Иногда часть наименее важных данных и вовсе приходится удалять.

Curiosity и MRO общаются с помощью UHF-антенны, коммуникационное окно открывается дважды в сол, и продолжается примерно 6-9 минут. MRO выделяет 5Гб в день для данных, полученных с роверов, и хранит их до тех пор, пока не окажется в зоне видимости одной из станций DSN на Земле, после чего передает данные туда. Передача данных к марсоходу осуществляется по такому же принципу. На хранение команд, которые должны быть переданы на марсоход, выделяется 30 Мб/сол.

В том числе, пока ровер находится в зоне видимости, он может получать новые базы данных и алгоритмы автономной работы, что позволяет менять принцип работы для избежания сбора избыточных данных.



# Рисунок 3. Квадрокоптер Xiro Explorer Mini

Один из представителей квадрокоптеров Xiro Explorer Mini обладает стандартной аппаратной частью, что и у многих аналогичных моделей. Из особых свойств можно выделить двойную систему спутниковой навигации (GPS и GLONASS), визуальное позиционирование и отсутствие стандартного пульта управления. Все управление дроном происходит при помощи приложения для смартфона или планшета.

Связь между дроном и устройством управления устанавливается с помощью Wi-Fi сигнала на частоте 2.4ГГц, что позволяет управлять Explorer’ом на дистанции до 100 метров, но на границе зоны сигнала задержка может быть слишком большой.

1. **Описание робота для практической работы**

Для выполнения практической работы была предоставлена колесная мобильная платформа, а также микроконтроллер ESP-WROOM-32(далее ESP32), драйвер моторов и четыре двигателя постоянного тока.

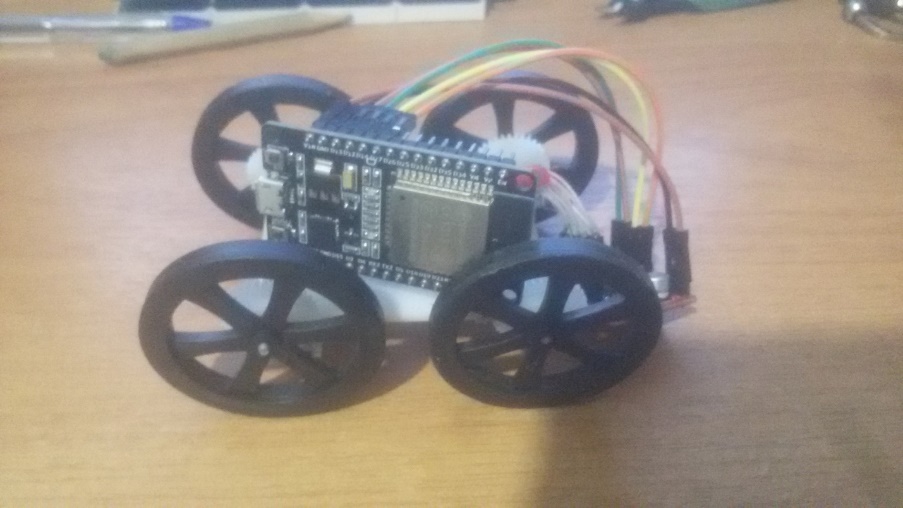


Рисунок 4. Малогабаритный колесный робот

ESP32 была выбрана из соображений компактности, низкого энергопотребления и наличия встроенных Wi-Fi и Bluetooth модулей.

## Характеристики

* Модуль: ESP-WROOM-32
* Частота беспроводной передачи: 2,4 ГГц
* Стандарт Wi-Fi: 802.11 b/g/n
* Стандарт Bluetooth: BLE v4.2 BR/EDR
* Тактовая частота: до 240 МГц
* Flash-память: 448 КБ
* Внешняя Flash-память: 4 МБ
* Оперативная память SRAM: 520 КБ
* Пины общего назначения: 25 ввода-вывода (GPIO) и 4 ввода (GPI)
* Контакты с АЦП: 15
* Разрядность АЦП: 12 бит
* Контакты с ЦАП: 2
* Разрядность ЦАП: 8 бит
* Контакты с ШИМ: 21 (до 16 каналов)
* Разрядность ШИМ: 16 бит
* Контакты ёмкостного сенсора: 8
* Пины с прерываниями: 25
* Аппаратные интерфейсы: 3× SPI, 3× UART, 2× I²C и 2× I²S
* Напряжение логических уровней: 3,3 В
* Максимальный ток на пинах: 12 мА
* Максимальный выходной ток пина 3V3: 1 А
* Входное напряжение через пин Vin: 5–14 В
* Габариты: 51×28 мм

Параметры шасси

* Диаметр колес: 43 мм
* Масса без источника питания: 50 гр
* Материал: пластик
* Габариты: 58×38 мм

1. **Разработка программного обеспечения**

Создание программного обеспечения делится на две части: создание прошивки микроконтроллера и создание мобильного приложения.

Для создания прошивки была использована среда ARDUINO IDE, модифицированная необходимыми библиотеками для работы с ESP32 и беспроводной связью

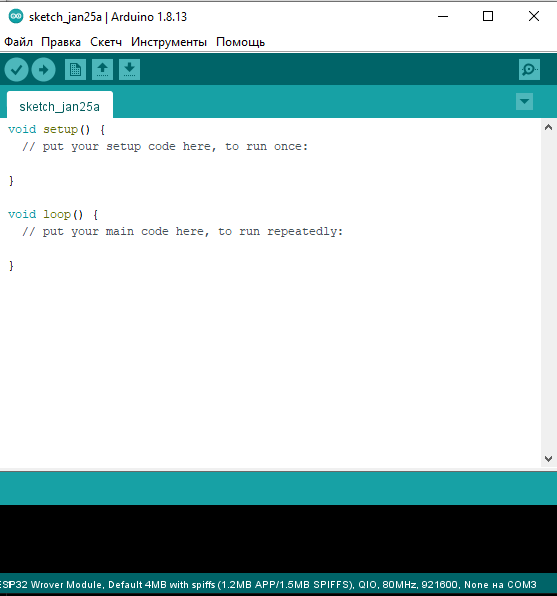


Рисунок 5. Окно разработки Arduino IDE

В прошивку необходимо внести свойства сети Wi-Fi (SSID и пароль), порт по которому будут передаваться данные (в нашем случае 80), парсинг входных данных по ключевым элементам строки и саму логику управления, машину состояний. Листинг программы приведен в приложении A.

Для разработки интерфейса и последующей возможности работать с инструментами дополненой реальности (AR) была выбрана среда разработки Unity 2019.4.16f1.

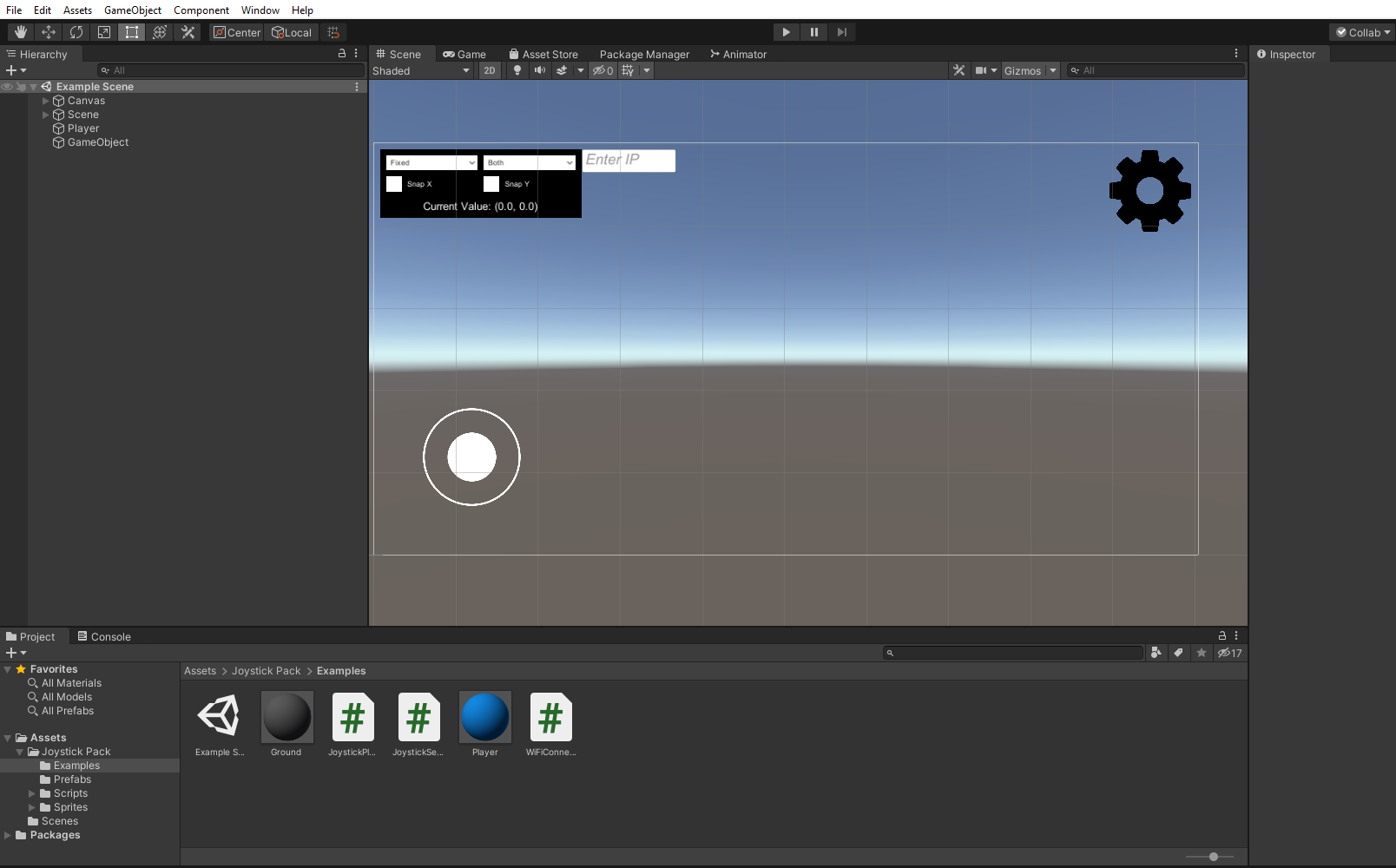
 Помимо возможности работы c AR и другими технологиями можно отметить простоту использования и многочисленные материалы, инструкции по работе с программным обеспечением и реализациями простых проектов.

Рисунок 6. Окно разработки Unity 2019

Был создан стандартный пустой проект, догружены необходимые средства для работы с мобильными платформами (Android SDK). В основном окне программы, с помощью панели компонентов был создан интерфейс, состоящий из окна настроек, джойстика и кнопки «Меню», которая скрывает или отображает окно настроек. В окне настроек можно выбрать один из нескольких режимов джойстика (Статичный, плавающий или динамичный), фиксацию оси управления, переключение передаваемых джойстиком значений между integer и float, для изменеения режима управления двигателями, и ввести IP робота, подключенного к локальной сети. Также создается программа, которая отвечает за передачу значений по адресу (Приложение Б).

1. **Проверка работоспособности программ и робота**

Установка программы для управления роботом производится выполнением JoystickRobotControl.apk файла на Android устройстве. После установки и включения программы на экран выводится интерфейс. В окне настроек пользователь должен ввести IP адрес робота, который будет получен после инициализации последнего.

Рисунок 7. Интерфейс приложения

В Arduino IDE компилируем программу, после на микроконтроллер загружается прошивка. Через монитор порта проверяем подключение к локальной сети и адрес.

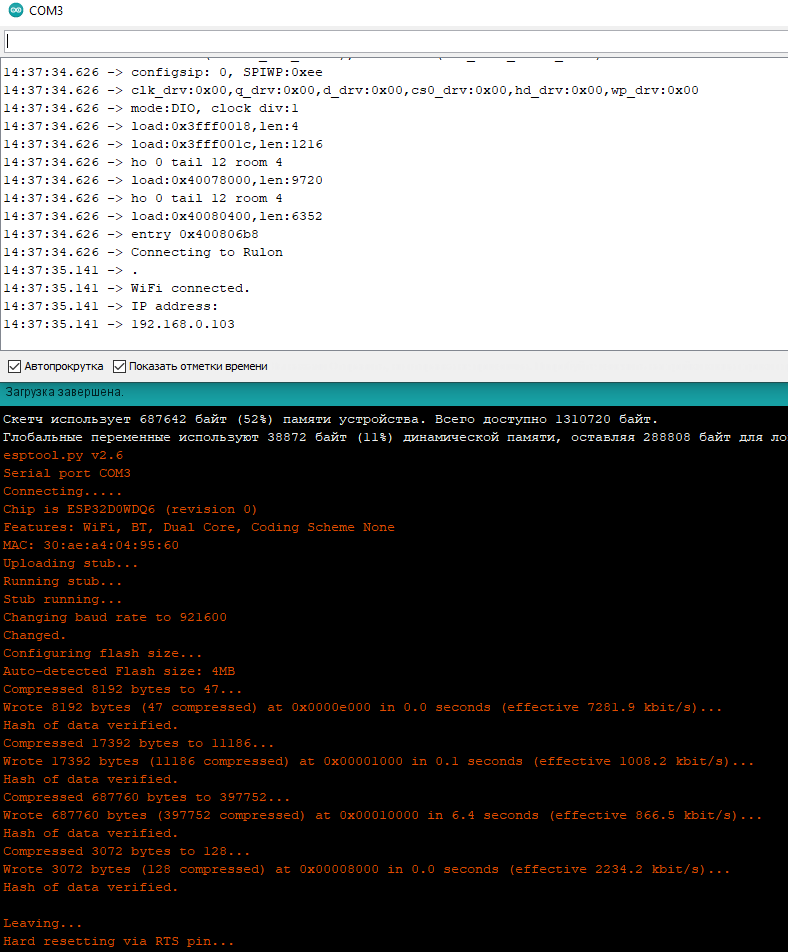


Рисунок 8. Успешная прошивка микроконтроллера и подключение к сети

После прошивки робота можно отсоединить от компьютера и подключить портативный источник питания. После кратковременной инициализации роботом можно управлять с мобильного устройства.

**Вывод**

В результате изучения способов передачи данных, реализации дистанционного управления было разработано программное обеспечение для управления роботом в локальной сети Wi-Fi, а также прошивка для микроконтроллера. Программное обеспечение может быть использовано на большинстве Android устройств благодаря автоматическому масштабированию и нетребовательности к ресурсам. Также были получены знания о средствах создания программного обеспечения для систем дистанционного управления.

Библиографический список

* 1. Дистанционное управление роботами для экстремальных работ // cyberleninka.ru URL: https://cyberleninka.ru/article/n/distantsionnoe-upravlenie-robotami-dlya-ekstremalnyh-rabot / Старовойтов Е.И. - Управление мобильными роботами и робототехническими системами.
  2. База знаний Амперки - http://wiki.amperka.ru/ - Текст: электронный. Информация о микроконтроллере и друких комплектующих.
  3. Старовойтов Е.И. Управление мобильными роботами и робототехническими системами / Е.И. Старовойтов. – Москва : RyjHec 2020 - 264 с. - ISBN 978-5-406-05064-4

**Приложение А**

**Листинг программного кода прошивки микроконтроллера**

#include <WiFi.h>

// вставляем ниже SSID и пароль для своей WiFi-сети:

const char\* ssid = "Rulon";

const char\* password = "US1NGnameSPACE";

// создаем объект сервера и задаем ему порт «80»:

WiFiServer server(80);

// переменная для хранения HTTP-запроса:

String header;

// мотор 1:

int motor1Pin1 = 27;

int motor1Pin2 = 14;

int enable1Pin = 15;

// мотор 2:

int motor2Pin1 = 12;

int motor2Pin2 = 13;

int enable2Pin = 32;

// переменные для свойств широтно-импульсной модуляции (ШИМ):

const int freq = 30000;

const int pwmChannel = 0;

const int resolution = 8;

int dutyCycle = 0;

// переменные для расшифровки HTTP-запроса GET:

String valueString = String(5);

void setup() {

Serial.begin(115200);

// переключаем контакты моторов в режим «OUTPUT»:

pinMode(motor1Pin1, OUTPUT);

pinMode(motor1Pin2, OUTPUT);

pinMode(motor2Pin1, OUTPUT);

pinMode(motor2Pin2, OUTPUT);

// задаем настройки ШИМ-канала:

ledcSetup(pwmChannel, freq, resolution);

// подключаем ШИМ-канал 0 к контактам ENA и ENB,

// т.е. к GPIO-контактам для управления скоростью вращения моторов:

ledcAttachPin(enable1Pin, pwmChannel);

ledcAttachPin(enable2Pin, pwmChannel);

// подаем на контакты ENA и ENB

// ШИМ-сигнал с коэффициентом заполнения «0»:

ledcWrite(pwmChannel, dutyCycle);

// подключаемся к WiFi-сети при помощи заданных выше SSID и пароля:

Serial.print("Connecting to "); // "Подключаемся к "

Serial.println(ssid);

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

// печатаем в мониторе порта

// локальный IP-адрес и запускаем веб-сервер:

Serial.println("");

Serial.println("WiFi connected."); // "Подключились к WiFi-сети."

Serial.println("IP address: "); // "IP-адрес: "

Serial.println(WiFi.localIP());

server.begin();

}

void loop(){

WiFiClient client = server.available();

if (!client) {

return;

}

String req = client.readStringUntil('\r');

Serial.println(req); // Стоит закомментировать для лучшего времени отклика

client.flush();

if (req.indexOf("=") != -1) //Ищем в принятом "=" - это и есть ключ. Извлекаем цифру после "=" и подставляем в ШИМ

{

float valOne = (float)req[req.indexOf("=(") + 3];

float valTwo = (float)req[req.indexOf(",") + 3];

}

if (valTwo >=0.2 && -0.2 < valOne <0.2) {

Serial.println("Forward"); // "Вперед"

digitalWrite(motor1Pin1, LOW);

digitalWrite(motor1Pin2, HIGH);

digitalWrite(motor2Pin1, LOW);

digitalWrite(motor2Pin2, HIGH);

} else if (valOne <=-0.2 && -0.2 < valTwo <0.2) {

Serial.println("Left"); // "Влево"

digitalWrite(motor1Pin1, LOW);

digitalWrite(motor1Pin2, LOW);

digitalWrite(motor2Pin1, LOW);

digitalWrite(motor2Pin2, HIGH);

} else if (-0.2 < valOne <=0.2 && -0.2 < valTwo <0.2) {

Serial.println("Stop"); // "Стоп"

digitalWrite(motor1Pin1, LOW);

digitalWrite(motor1Pin2, LOW);

digitalWrite(motor2Pin1, LOW);

digitalWrite(motor2Pin2, LOW);

} else if (valOne >=0.2 && -0.2 < valTwo <0.2) {

Serial.println("Right"); // "Вправо"

digitalWrite(motor1Pin1, LOW);

digitalWrite(motor1Pin2, HIGH);

digitalWrite(motor2Pin1, LOW);

digitalWrite(motor2Pin2, LOW);

} else if (valTwo <=-0.2 && -0.2 < valOne <0.2) {

Serial.println("Reverse"); // "Назад"

digitalWrite(motor1Pin1, HIGH);

digitalWrite(motor1Pin2, LOW);

digitalWrite(motor2Pin1, HIGH);

digitalWrite(motor2Pin2, LOW);

}

client.flush();

}

**Приложение Б**

**Листинг программного кода передачи данных**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class WiFiConnectionTransfer: MonoBehaviour

{

public string Url = "";

public Text IPstring;

public VariableJoystick variableJoystick;

public Text valueText;

private IEnumerator SendRequest(string URL)

{

var www = new WWW(URL);

while (!www.isDone)

yield return www;

}

// Use this for initialization

void Start()

{

}

// Update is called once per frame

void Update()

{

Url = "http://" + IPstring.text + "/JoystickPosition=" + variableJoystick.Direction;

StartCoroutine(SendRequest(Url));

}

public void Exit()

{

Application.Quit();

}

}

**Приложение В**

**Листинг программного кода интерфейса**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class JoystickSetterExample : MonoBehaviour

{

public VariableJoystick variableJoystick;

public Text valueText;

public Image background;

public Sprite[] axisSprites;

public void ModeChanged(int index)

{

switch(index)

{

case 0:

variableJoystick.SetMode(JoystickType.Fixed);

break;

case 1:

variableJoystick.SetMode(JoystickType.Floating);

break;

case 2:

variableJoystick.SetMode(JoystickType.Dynamic);

break;

default:

break;

}

}

public void AxisChanged(int index)

{

switch (index)

{

case 0:

variableJoystick.AxisOptions = AxisOptions.Both;

background.sprite = axisSprites[index];

break;

case 1:

variableJoystick.AxisOptions = AxisOptions.Horizontal;

background.sprite = axisSprites[index];

break;

case 2:

variableJoystick.AxisOptions = AxisOptions.Vertical;

background.sprite = axisSprites[index];

break;

default:

break;

}

}

public void SnapX(bool value)

{

variableJoystick.SnapX = value;

}

public void SnapY(bool value)

{

variableJoystick.SnapY = value;

}

private void Update()

{

valueText.text = "Current Value: " + variableJoystick.Direction;

}

}