

Содержание

Техническая поддержка	1.1
Введение	1.2
Глоссарий	1.3
Безопасность при сборке	1.4
Правила эксплуатации мультикоптеров	1.5
Сборка	1.6
Настройка	1.7
Калибровка датчиков	1.7.1
Настройка пульта	1.7.2
Полетные режимы	1.7.3
Настройка питания	1.7.4
Настройка Failsafe	1.7.5
Ручной полет	1.8
Упражнения	1.8.1
Работа с Raspberry Pi	1.9
Образ для RPi	1.9.1
Подключение по Wi-Fi	1.9.2
Подключение к полетному контроллеру	1.9.3
QGroundControl по Wi-Fi	1.9.4
SSH-доступ	1.9.5
Командная строка	1.9.6
Автоматическая проверка	1.9.7
Просмотр видеострима с камер	1.9.8
Программирование	1.10
Модули и датчики	1.10.1
Датчик температуры и влажности	1.10.1.1
Инфракрасный модуль (датчик движения)	1.10.1.2
Модуль часов реального времени	1.10.1.3
Датчик дождя	1.10.1.4
Датчик звука	1.10.1.5
Ультразвуковой дальномер	1.10.1.6
Датчик огня	1.10.1.7
Датчик лазерный	1.10.1.8
Светочувствительный датчик	1.10.1.9
Датчик влажности почвы	1.10.1.10

Датчик обхода препятствий	1.10.1.11
Датчик вибраций	1.10.1.12
Датчик газа	1.10.1.13
Датчик наклона	1.10.1.14
Модуль трассировки пути	1.10.1.15
Лазерный дальномер	1.10.1.16
Настройка камеры	1.10.2
Визуальные маркеры (ArUco)	1.10.3
Распознавание маркеров	1.10.3.1
Навигация по карте маркеров	1.10.3.2
Автономный полет (OFFBOARD)	1.10.4
Системы координат	1.10.5
Работа с GPIO	1.10.6
Примеры кода	1.10.7
Компьютерное зрение	1.10.8
Автозапуск ПО	1.10.9
Использование JavaScript	1.10.10
Блочное программирование	1.10.11
ROS	1.10.12
Дополнительные материалы	1.11
Светодиодная лента	1.11.1
Радио-телеметрия	1.11.2
Гид по автономному полету	1.11.3
Имя хоста	1.11.4
Настройка PID	1.11.5
Навигация по настенным маркерам	1.11.6
Подключение к VPN ZeroTier	1.11.7
Подключение к VPN Hamachi	1.11.8
Управление дроном при помощи 4G связи	1.11.9
Jetson Nano	1.11.10
Настройка сети RPi	1.11.11
Интерфейс UART	1.11.12
Использование мультиметра	1.11.13
Неисправности радиоаппаратуры	1.11.14
Прошивка ESC контроллеров	1.11.15
Взаимодействие с Arduino	1.11.16
Подключение GPS	1.11.17
Магнитный захват	1.11.18

Управление в режиме тренера	1.11.19
Техника лужения	1.11.20
Типы силовых разъемов	1.11.21
Светодиодная лента	1.11.22

Учебник

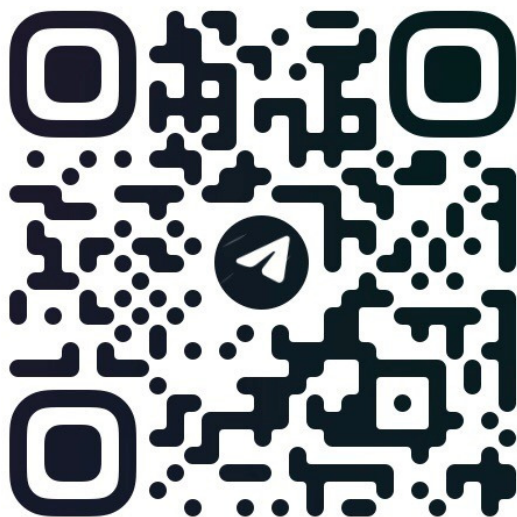
Учебно-методическое пособие	2.1
Контрольные материалы	2.2

Ссылки

Скачать эту документацию в PDF	3.1
Скачать образ для Raspberry	3.2

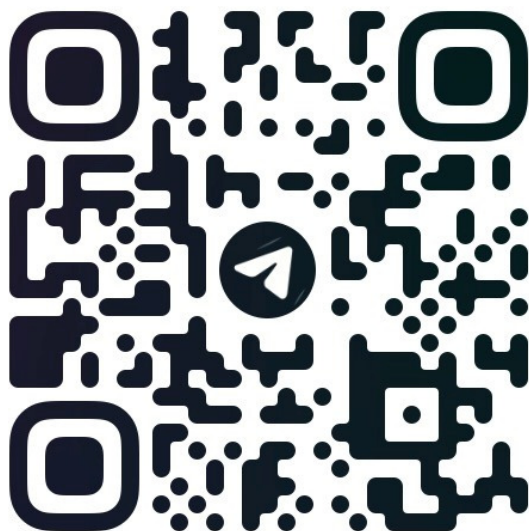
Техническая поддержка

- [Группа технической поддержки пользователей](#)



@TEZONA_TECH

- [Чат-бот технической поддержки](#)



@TEZONA_BOT

Конструкторы, развивающие и ресурсные наборы

Конструкторы

Конструктор БПЛА «Пиксель-Вжик»



- [Инструкции и документация](#)

Конструктор БПЛА «Пиксель-Вжик» создан специально для обучения детей младшего и среднего школьного возраста. Он идеально подходит для введения в мир беспилотных летательных аппаратов и отработки практических навыков проектирования, программирования и моделирования дронов, а также знакомства с сопутствующими физическими процессами. Сделан из нетоксичного и безопасного ABS-пластика, простой в сборке и совместим с деталями конструктора "LEGO", что позволяет смоделировать и собрать квадрокоптер собственной конструкции.

Конструктор БПЛА «Пиксель-Вжик - рой дронов»



- [Инструкции и документация](#)

Конструктор БПЛА «Пиксель-Вжик - рой дронов» состоит из 10 дронов и предназначен для детей от 6 до 14 лет. С помощью конструктора вы сможете одновременно управлять этими дронами путем программирования и

создавать творческие решения для них. Конструктор дает возможность индивидуального программирования, программирования светового шоу и поддержку графического программирования. Кроме самого дрона и его комплектующих, в комплект входят: роутер, карта меток, транспортировочный кейс.

Конструктор БПЛА мультироторного типа «Оса»



- [Инструкции и документация](#)

Конструктор БПЛА мультироторного типа «Оса» предназначен для изучения конструкции беспилотных летательных аппаратов мультироторного типа, их проектирования, сборки, обучения основам визуального пилотирования, пилотирования в полуавтономном и полностью автономном режимах и основам программирования с использованием различного набора датчиков.

В комплектацию конструктора входят все необходимые комплектующие для сборки БПЛА, а также система радиуправления и передачи видеосигнала, что позволяет собрать полностью готовый беспилотный летательный аппарат.

В стандартной комплектации конструктор имеет систему круговой защиты, что позволяет использовать его в закрытых помещениях.

Конструктор БПЛА самолетного типа «Орленок»



- [Инструкции и документация](#)

Конструктор БПЛА самолетного типа «Орленок» предназначен для изучения конструкции, ее проектирования, сборки, обучения основам визуального пилотирования, пилотирования в полуавтономном и полностью автономном режимах и основам программирования с использованием различного набора датчиков.

В комплектацию конструктора входят все необходимые комплектующие для сборки, а также система радиуправления и передачи видеосигнала, что позволяет собрать полностью готовый беспилотный летательный аппарат.

Конструктор спортивного БПЛА мультироторного типа для FPV-пилотирования «Олимпиец»



- [Инструкции и документация](#)

Конструктор спортивного БПЛА мультироторного типа для FPV-пилотирования «Олимпиец» предназначен для развития ранней профориентации в области конструирования, сборки и FPV пилотирования

беспилотных летательных аппаратов мультироторного типа в общеобразовательных организациях для детей от 12 лет.

Использование дрона «Олимпиец» в образовательном процессе позволит развить навыки FPV пилотирования дронов у учащихся и обеспечит возможность подготовки к участию в Дрон-рейсинге - виде соревновательной деятельности, представляющий собой гоночные соревнования квадрокоптеров (дронов), управляемых спортсменами-участниками соревнования, проводимые на специально оборудованных трассах.

Дрон «Олимпиец» позволяет развивать навыки конструирования и моделирования БПЛА. Тип конструктора: сборно-разборная модель, модульная многоцветная сборка.

Развивающие и ресурсные наборы

Развивающий набор для FPV-пилотирования «Чижик»



- [Инструкции и документация](#)

Развивающий набор для FPV-пилотирования «Чижик» предназначен для обучения FPV-пилотированию. Набор развивает навыки FPV-пилотирования. 3-х скоростной режим позволяет начинать обучение на низкой скорости без использования FPV-шлема (визуальное управление БПЛА) и впоследствии, приобретая опыт, увеличивать до максимальной с использованием FPV-шлема.

Данный БПЛА имеет небольшие размеры, поэтому его разрешается использовать внутри помещений. Набор предназначен для детей от 8 лет.

Образовательный ресурсный набор для программирования конструктора БПЛА «Пиксель-Вжик»



- [Инструкции и документация](#)

Образовательный ресурсный набор для программирования конструктора БПЛА «Пиксель-Вжик» представляет собой комплект различных ЛЕГО-совместимых модулей для получения опыта программирования, развития творческих способностей и логического мышления обучающихся, для подготовки к участию в конкурсах и соревнованиях. Набор является необходимым дополнением к конструктору беспилотного летательного аппарата «Пиксель-Вжик».

Глоссарий

БПЛА

Беспилотный летательный аппарат. Примеры: квадрокоптер, гексакоптер, самолет, летающее крыло, конвертоплан (VTOL), вертолет.

Квадрокоптер

Беспилотный летательный аппарат с 4-мя винтами и электронной системой стабилизации.

Мультикоптер

Беспилотный летательный аппарат с электронной системой стабилизации и числом пропеллеров, равным 3 (трикоптер), 4 (квадрокоптер), 6 (гексакоптер), 8 (октокоптер) или более.

Полетный контроллер / автопилот

1. Специализированная плата, спроектированная для управления дроном. Примеры: Pixhawk, ArduPilot, Naze32, CC3D.
2. Программное обеспечение для платы управления дроном. Примеры: PX4, APM, CleanFlight, BetaFlight.

Прошивка

Программное обеспечение, управляющее работой какого-либо устройства, например, полетного контроллера или регулятора мотора (ESC).

Мотор

Электродвигатель - устройство которое вращает пропеллеры дрона. Обычно используются бесколлекторные электродвигатели.

ESC / регулятор оборотов двигателя / "регуль"

Electronic Speed Controller. Специализированная плата, которая управляет скоростью вращения бесколлекторного электродвигателя.

АКБ / аккумулятор / батарея

Перезаряжаемый источник тока для БПЛА. В дронах обычно применяются Li-ро (литий-полимерные) аккумуляторы.

Ячейка / "банка" АКБ

Составная часть АКБ, непосредственный источник тока. Обычно АКБ для БПЛА состоят из нескольких (2–6) ячеек, соединенных последовательно. Максимальное напряжение одной Li-ро ячейки – 4.2 В; общее напряжение АКБ равно суммарному напряжению ячеек. Количество ячеек обозначается буквой S, например: 2S, 3S, 4S.

Пульт / аппаратура радиоуправления / "аппа"

Пульт для управления дроном, работает по радиоканалу. Для работы пульта к полетном контроллеру необходимо подключить ресивер (приемник радиосигнала).

Телеметрия

1. Передача данных о состоянии дрона или другого аппарата на расстояние.
2. Совокупность данных о состоянии аппарата, так таковая (высота, ориентация, глобальные координаты и т. д.).
3. Система для передачи данных о состоянии аппарата или команд для него по воздуху. Примеры: радиомодемы (RFD900, 3DR Radio Modem), Wi-Fi модули (ESP-07). Raspberry Pi также может быть использован в качестве модуля для телеметрии: [использование QGroundControl через Wi-Fi](#).

Арминг

Armed – состояние коптера готовности к полету. При поднятии стика газа либо при посылке внешней команды с целевой точкой – дрон полетит. Обычно он начинает вращать пропеллерами при переходе в состояние "armed" даже если стик газа находится внизу.

Противоположным состоянием является Disarmed.

Raspberry Pi

[Популярный одноплатный микрокомпьютер.](#)

Образ SD-карты

Полная копия содержимого SD-карты, представленная в виде файла. Такой файл можно загрузить на SD-карту, воспользовавшись специальной утилитой, например [Etcher](#). SD-карта, вставленная в Raspberry Pi является единственным его долговременным хранилищем и полностью определяет, что он будет делать.

Образ SD-карты можно скачать [по этой ссылке](#).

APM / ArduPilot

Полетный контроллер с открытым исходным кодом изначально создан для платы Arduino. Впоследствии был портирован на Pixhawk, Pixracer и другие платы.

MAVLink

Протокол для взаимодействия дронов, наземных станций и других аппаратов по радиоканалам. Обычно именно этот протокол используется для телеметрии.

ROS

Популярный фреймворк для написания сложных роботехнических приложений.

MAVROS

Библиотека - связующее звено между аппаратом, работающем по протоколу MAVLink, и ROS.

UART

Последовательный асинхронный интерфейс передачи данных, применяемый во многих устройствах. Например, GPS антенны, Wi-Fi роутеры или Pixhawk.

IMU

Inertial measurement unit - комбинация датчиков (гироскоп, акселерометр, магнитометр), которая помогает БПЛА рассчитывать ориентацию и положение в пространстве.

Estimation

Процесс определения ПО полетного контроллера состояния дрона: положения в пространстве, скоростей, углов наклона и т.д. Для этого используется смешивание данных с установленных датчиков и различные алгоритмы фильтрации, например [фильтр Калмана](#).

В прошивке АРМ эту функцию выполняет подсистема [ЕКФ2](#).

Инструкция по безопасности при сборке

Пайка

Работы, связанные с пайкой и лужением, должны проводиться в специально оборудованных и предварительно подготовленных помещениях. Обязательно должна присутствовать система вентиляции.



Перед началом работы необходимо:

1. Привести в порядок рабочее место, ничего не должно мешать процессу. Рабочее место должно быть хорошо освещено.
2. Проверить целостность проводки и штепсельной вилки.
3. Паяльник, находящийся в рабочем состоянии, установить в зоне действия местной вытяжной вентиляции, в специальную подставку.
4. Перед началом работы надеть защитный халат, очки и, при необходимости, перчатки.

Во время пайки:

1. Паяльник следует держать только за ручку, так как жало имеет высокую температуру.



2. Жало паяльника нагревается до очень высокой температуры, поэтому, в случае его прикосновения к электрическому проводу в ходе пайки изоляция будет повреждена в считанные мгновения, с последующим коротким замыканием.
3. Для перемещения изделий применять специальные инструменты (пинцеты, клещи или другие инструменты), обеспечивающие безопасность при пайке.
4. Во избежание ожогов расплавленным припоем при распайке не выдергивать резко с большим усилием паяемые провода.

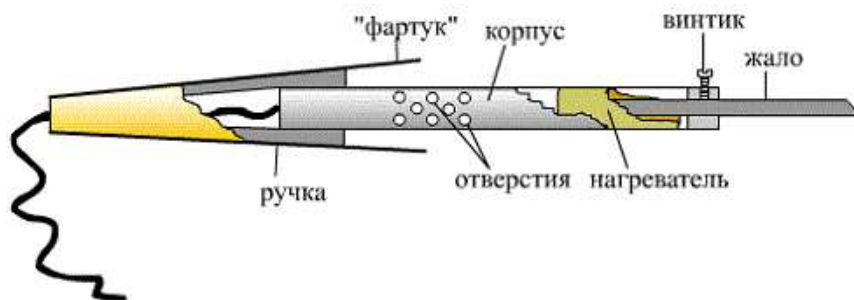
- При пайке мелких и подвижных изделий пользоваться специальным держателем.



- Паяльник переносить за корпус, а не за провод или рабочую часть. При перерывах в работе паяльник отключать от электросети.

При обнаружении неисправной работы паяльника или возникновении возгорания отключить его от питающей электросети.

Канифоль и припой при плавлении выделяют значительное количество вредных веществ. Настоятельно рекомендуется проветривать помещение после каждой пайки. Через каждые 30 минут нужно делать небольшие перерывы со сквозным проветриванием помещения и не забывать при этом отключать паяльник.



Правила эксплуатации

Правила техники безопасности при эксплуатации мультикоптеров

Следует помнить, что беспилотные летательные аппараты (БПЛА), к которым относятся мультикоптеры, являются источниками повышенной опасности и способны причинить серьёзные травмы, а также ущерб имуществу, что, в свою очередь, может стать причиной привлечения оператора БПЛА, организатора занятий и других ответственных лиц к административной и уголовной ответственности в соответствии с действующим законодательством.

Внимательно прочтите и запомните приведённые ниже правила — их неукоснительное соблюдение является необходимым условием успешной и безаварийной эксплуатации мультикоптера, бортового и наземного оборудования, а также вашей собственной безопасности и безопасности окружающих.

Приступая к эксплуатации БПЛА и его компонентов вы подтверждаете, что внимательно прочли все перечисленные ниже пункты, понимаете их смысл и готовы их ответственно соблюдать.

- Перед началом эксплуатации мультикоптера ознакомьтесь с действующими законодательными нормами в части порядка использования воздушного пространства, регистрации и учёта БПЛА и неукоснительно соблюдайте их;
- Перед первым включением убедитесь в правильности сборки мультикоптера, отсутствии коротких замыканий и целостности проводки. Помните, что несоблюдение этого пункта может привести к пожару;
- Перед каждым полётом проверяйте мультикоптер и все его детали и элементы конструкции на целостность, они не должны иметь механических повреждений;
- Перед выполнением первого полёта убедитесь, что вы внимательно изучили настоящее методическое пособие, поняли все приведённые в нём указания и рекомендации и готовы их ответственно соблюдать;
- Сборка и эксплуатация мультикоптера лицами, не достигшими 18-летнего возраста, должна осуществляться под присмотром компетентных и ответственных взрослых;
- Запрещено обслуживание, включение и управление БПЛА в состоянии алкогольного и наркотического опьянения;
- Запрещены полёты БПЛА вблизи линий электропередач, радиолокационных станций и других источников сильного электромагнитного излучения;
- Запрещено присутствие свободно передвигающихся домашних животных в местах полётов БПЛА;

- Полёты мультикоптеров массой до 250г выполняются в просторных помещениях либо на открытых площадках в безветренную погоду, полёты мультикоптеров массой свыше 250 гр выполняются на открытых площадках при ветре не более 8 м/с;
- При выполнении полётов в помещении необходимо наличие защитной сетки с размером ячейки менее размера мультикоптера между местом выполнения полёта и зрителями и другими лицами. Рекомендуется использование куба с длиной стороны 3 метра (набор «Аэрокуб»);
- Всегда сначала включайте передатчик системы радиуправления, затем подключайте силовой аккумулятор мультикоптера;
- Во время включения мультикоптера убедитесь, что расстояние между ним и передатчиком системы радиуправления составляет более одного метра;
- Перед проверкой работоспособности мультикоптера отойдите от него на расстояние не менее двух метров. В процессе выполнения полёта не допускайте нахождения БПЛА на расстоянии более двух метров от вас и других людей до момента выключения двигателей;
- Остерегайтесь вращающихся деталей БПЛА, таких как воздушные винты. Помните, что они могут стать причиной серьёзных травм, в том числе опасных для жизни;
- При полётах БПЛА визуально (без FPV-оборудования) разрешено нахождение зрителей и других лиц только за спиной оператора, в этом случае мультикоптер находится в воздухе строго перед оператором;
- При полётах на дальние расстояния, в том числе в автоматическом режиме, запрещается нахождение зрителей и других лиц на траектории полёта БПЛА;
- Прежде чем подходить к мультикоптеру как в случае его падения, так и после штатной посадки и брать его в руки, убедитесь, что электродвигатели выключены и их случайный запуск невозможен;
- Выключайте передатчик системы радиуправления только после выключения мультикоптера;
- При эксплуатации БПЛА в первую очередь руководствуйтесь здравым смыслом и принимайте решения, адекватные ситуации.

Правила техники безопасности при хранении, транспортировке и эксплуатации аккумуляторов

В составе комплекта мультикоптера и его оборудования могут использоваться LiPo, LiIon, LiFe и LiHV аккумуляторы. Все перечисленные выше типы относятся к литиевым аккумуляторам и при неправильном обращении могут стать причиной возникновения пожара, а также серьёзных травм, в том числе опасных для жизни.

Приступая к эксплуатации аккумуляторов из состава данного набора вы подтверждаете, что внимательно прочли все перечисленные ниже пункты, понимаете их смысл и готовы их ответственно соблюдать.

- Изучите действующие законодательные нормы в части хранения и транспортировки литиевых аккумуляторов и неукоснительно соблюдайте их;
- Храните аккумуляторы исключительно в ёмкостях из плотных негорючих материалов;
- В процессе хранения и транспортировки не допускайте короткого замыкания аккумуляторов. Категорически запрещён контакт аккумуляторной батареи и её силового и балансировочного разъемов с металлическими предметами.
- Утилизируйте использованные и вышедшие из строя литиевые аккумуляторы в соответствии с действующим федеральным и местным законодательством;
- Перед началом эксплуатации аккумулятора проведите его осмотр. Аккумулятор, его кабели и разъемы не должны иметь механических повреждений. Также запрещена эксплуатация аккумуляторов, имеющих вздутие или подозрительный запах. Проводите осмотр перед каждым применением аккумулятора;
- Убедитесь, что ваше зарядное устройство поддерживает выбранный тип аккумулятора и используется соответствующая программа. Помните, что использование несовместимого зарядного устройства, либо неправильного алгоритма заряда, ведёт к взрыву аккумулятора и последующему пожару;
- Не заряжайте аккумулятор без присмотра. В случае вздутия или нагрева аккумулятора в процессе заряда незамедлительно прервите процесс, поместите аккумулятор в огнеупорную ёмкость и утилизируйте в соответствии с действующим законодательством;
- Внимательно изучите инструкции производителей ко всем используемым вами аккумуляторам и зарядным устройствам и неукоснительно соблюдайте их.

Безопасность при подготовке к вылету

- Убедиться, что Li-ion аккумуляторы заряжены.
- Убедиться, что батарейки в аппаратуре управления заряжены.
- Устанавливать пропеллеры только перед вылетом.

Проверить надёжность следующих узлов:

- Затянутость гаек пропеллеров.
- Крепление и целостность защит пропеллеров.
- Надёжность крепления проводов, отсутствие болтающихся проводов.

Безопасность перед вылетом

- Располагать зрителей за спиной пилота или за линией, проходящей через оба плеча пилота за спиной пилота.
- Не допускать выхода зрителей в полусферу перед лицом пилота.
- Знать и помнить время полёта, на которое рассчитан данный коптер и его аккумулятор.

- ДО подключения Li-ion аккумулятора включить аппаратуру управления (пульт), перевести левый стик (газ) в нулевое положение.
- Подключать Li-ion аккумулятор только перед взлётом, отключать сразу после взлёта.
- Стоять на расстоянии не менее 3 м от коптера.
- Взлетать с земли с ровной площадки, на расстоянии не менее 3 метров от препятствий.

Безопасность в полете

- Выполнять все указания преподавателя или лётного инструктора.
- Заранее обозначить зону пилотажа. Летать только в обозначенной зоне и не допускать вылета за её пределы. Не залетать за собственную спину.
- При обучении полётам летать на уровне ниже собственного роста.
- Летать рядом с собой на расстоянии, на котором вам видна ориентация коптера в пространстве. Не улетать далеко от себя. В случае сомнений в ориентации коптера немедленно выполнить посадку на месте. Не пытаться взлететь. Подойти ближе к коптеру и выполнить взлёт.
- При управлении все движения стиками выполнять аккуратно и плавно. Не допускать резких движений. При необходимости изменить направление полёта двигать стиками следует энергично, но не резко.
- Летать следует осторожно и выполнять только те элементы, в которых нет сомнений. Запрещается выполнять фигуры пилотажа, в успехе которых возникают сомнения и фигуры, связанные с риском.
- Соблюдать скоростной режим. Скорость полёта коптера держать в пределах скорости идущего человека.
- Вернуть коптер к месту посадки к рассчитанному времени, не допускать полной разрядки аккумулятора в полёте.
- Посадку выполнять только на ровную открытую площадку вдали от препятствий.

Аварийная посадка

В случае удара об землю или жесткой посадки выполнить следующие действия:

1. Прекратить полёт. Посадить коптер на землю.
2. Disarm.
3. Отключить Li-ion аккумулятор на коптере.
4. Выключить пульт.
5. Осмотреть коптер и при необходимости отремонтировать.

Запланированная посадка




После запланированной посадки выполнить следующие действия:





1. Disarm
2. Отключить Li-ion аккумулятор на коптере.
3. Выключить пульт.

Датчик температуры и влажности

Инструкции по сборке и настройке

В этом разделе находятся статьи с инструкциями по сборке и настройке БПЛА.

<p>Конструктор «Пиксель-Вжик»</p>		<p>Программное обеспечение</p> <ul style="list-style-type: none"> • Программное обеспечение • Драйверы для Win 8/10 • Драйверы для Win 7 • Драйверы для Mac <p>Документация</p> <ul style="list-style-type: none"> • Технический паспорт «Пиксель-Вжик» • Инструкция по эксплуатации «Пиксель-Вжик»
<p>Конструктор «Пиксель-Вжик - рой дронов»</p>		<p>Программное обеспечение</p> <ul style="list-style-type: none"> • Программное обеспечение • Драйверы для Win 8/10 • Драйверы для Win 7 • Драйверы для Mac <p>Документация</p> <ul style="list-style-type: none"> • Технический паспорт «Пиксель-Вжик - рой дронов» • Инструкция по эксплуатации «Пиксель-Вжик - рой дронов»
<p>Конструктор мультироторного типа «Оса»</p>		<p>Документация</p> <ul style="list-style-type: none"> • Технический паспорт • STL файлы • Инструкция по сборке PDF • Видеоинструкция сборки • Видеоинструкция калибровки

<p>Конструктор самолетного типа «Орленок»</p>		<p>Документация</p> <ul style="list-style-type: none"> • Технический паспорт • STL файлы • Инструкция по сборке PDF • Видеоинструкция сборки часть 1 • Видеоинструкция сборки часть 2 • Видеоинструкция прошивки полетного контроллера
<p>Конструктор спортивного мультироторного типа для FPV-пилотирования «Олимпиец»</p>		<p>Документация</p> <ul style="list-style-type: none"> • Описание • Технический паспорт • Инструкция по сборке и настройке PDF
<p>Развивающий набор для FPV-пилотирования «Чижик»</p>		<p>Документация</p> <ul style="list-style-type: none"> • Инструкция по эксплуатации • Технический паспорт
<p>Образовательный ресурсный набор для программирования конструктора БПЛА «Пиксель-Вжик»</p>		<p>Документация</p> <ul style="list-style-type: none"> • Инструкция по эксплуатации • Технический паспорт

Первоначальная настройка

Установка QGroundControl

QGroundControl – программное обеспечение, необходимое для прошивки, настройки и калибровки полетного контроллера.

Скачайте и установите установочный файл для Windows/Linux/macOS с [официального сайта QGroundControl](#). В случае необходимости согласитесь с установкой дополнительных драйверов при установке.

См. также [основную документацию по QGroundControl](#).

MicroSD-карта

Подготовьте MicroSD-карту для полетного контроллера.

Загрузка прошивки в полетный контроллер

[Прошивка ArduCopter 4.3.2.](#)

1. Отключите полетный контроллер от компьютера (если он подключен).
2. Запустите программу QGroundControl.
3. Перейдите в панель *Vehicle Setup* (кликнув на логотип QGroundControl в левом верхнем углу) и выберите меню *Firmware*.
4. Подключите полетный контроллер к компьютеру по USB.
5. Для загрузки прошивки:
 - o Выберите *Advanced settings*.
 - o В выпадающем меню выберите *Custom firmware file...*
 - o Нажмите *OK* и выберите скаченный файл прошивки.

Для загрузки последней версии **стандартной прошивки** сразу нажмите *OK*.

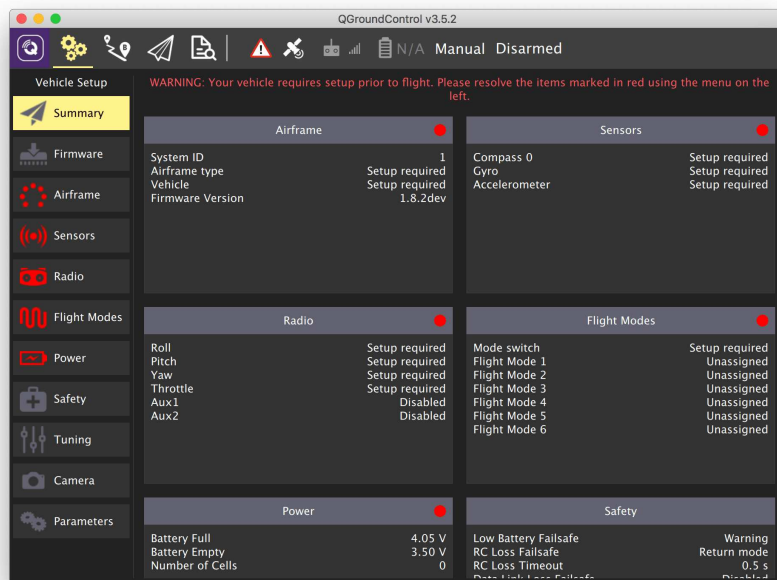
Дождитесь, пока QGroundControl загрузит прошивку и выполнит перезагрузку полетного контроллера.

Не отключайте полетный контроллер от компьютера в процессе загрузки прошивки.

Настройка полетного контроллера

Все дальнейшие настройки и калибровки полетного контроллера могут быть выполнены без проводов с применением [доступа к полетному контроллеру по Wi-Fi через Raspberry Pi](#).

Обзор главного окна настроек QGroundControl:



1. Параметры, нуждающиеся в настройке: *Airframe*, *Radio*, *Sensors*, *Flight Modes*.
2. Текущая прошивка контроллера.
3. Текущий полетный режим.
4. Сообщения об ошибках.

Выбор рамы

1. Зайдите во вкладку *Vehicle Setup*.
2. Выберите меню *Airframe*.
3. Выберите тип рамы *Quad H*.
4. Переместитесь в начало списка и нажмите кнопку *Apply and Restart*, подтвердите нажатием *Apply*.
5. Дождитесь применения настроек и перезагрузки полетного контроллера.

Настройка PID-регуляторов

Усредненные коэффициенты PID для конструктора мультироторного типа

- `MC_PITCHRATE_P` = 0.140
- `MC_PITCHRATE_I` = 0.090
- `MC_PITCHRATE_D` = 0.0008
- `MC_ROLLRATE_P` = 0.140
- `MC_ROLLRATE_I` = 0.100

- `MC_ROLLRATE_D` = 0.0006

Усредненные коэффициенты PID для конструктора самолетного типа

- `MC_PITCHRATE_P` = 0.179
- `MC_PITCHRATE_I` = 0.150
- `MC_PITCHRATE_D` = 0.002
- `MC_ROLLRATE_P` = 0.195
- `MC_ROLLRATE_I` = 0.170
- `MC_ROLLRATE_D` = 0.002

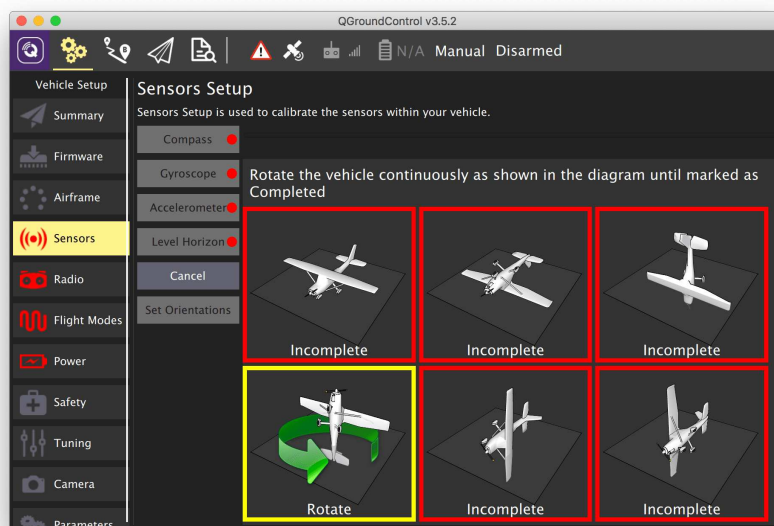
Необходимо учитывать, что для идеального полета параметры PID-регуляторов подбираются вручную для каждого конкретного собранного дрона. Вы можете узнать больше об этом в статье ["Настройка PID-регуляторов"](#).

Далее: [Калибровка датчиков.](#)

Калибровка датчиков

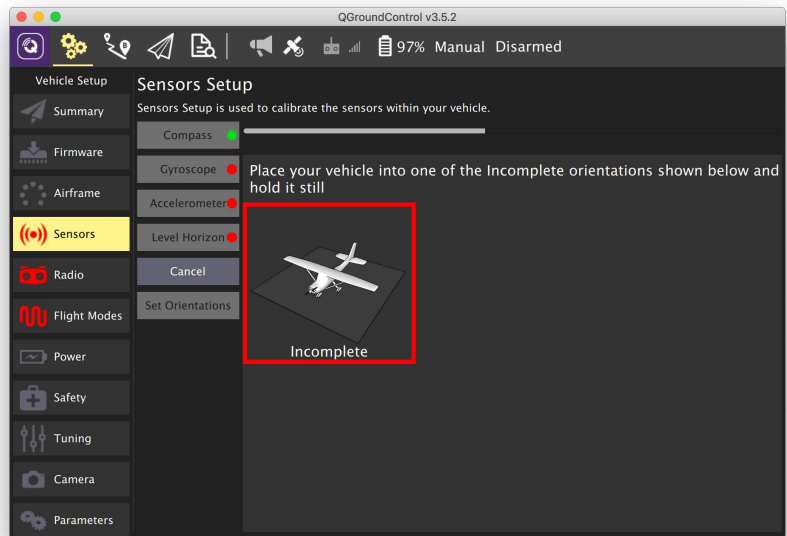
Чтобы откалибровать датчики зайдите во вкладку *Vehicle Setup* и выберите меню *Sensors*.

Компас



1. Выберите меню *Compass*.
2. Выберите ориентацию полетного контроллера – *ROTATION_NONE* при условии, что полетный контроллер ориентирован передом к носу дрона.
3. Нажмите *OK*.
4. Последовательно устанавливайте дрон в каждую из указанных ориентаций до появления желтой рамки.
5. Вращайте дрон по направлению стрелки до появления зеленой рамки.

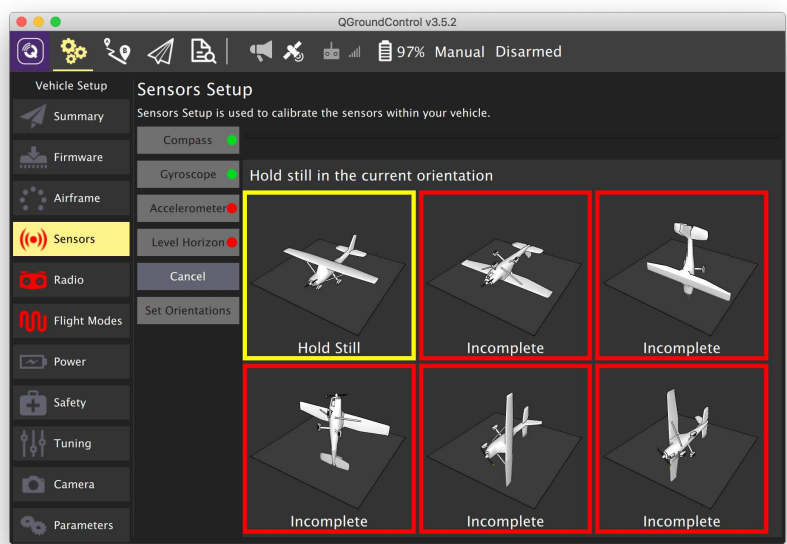
Гироскоп



1. Выберите меню *Gyroscope*
2. Установите дрон на ровную поверхность.
3. Нажмите *OK*.
4. Дождитесь окончания калибровки.

Во время калибровки гироскопа дрон не должен менять своего положения, шататься и т. д.

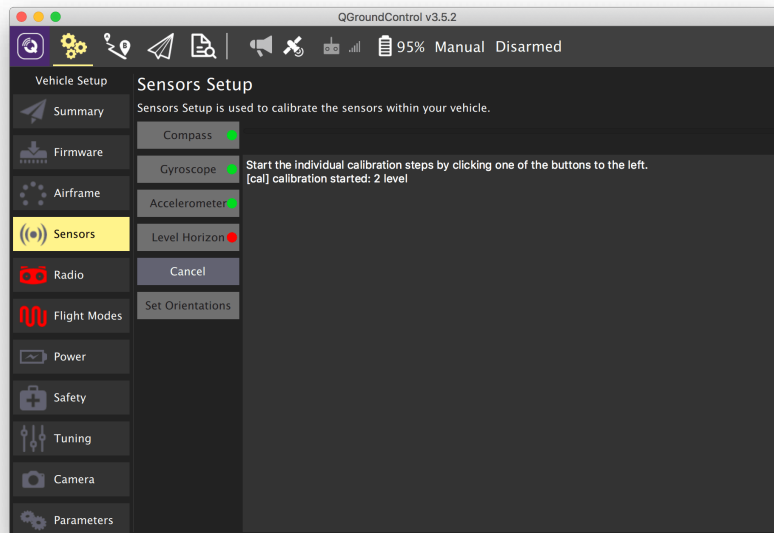
Акселерометр



1. Выберите меню *Accelerometer*.
2. Выберите ориентацию полетного контроллера – *ROTATION_NONE* при условии, что полетный контроллер ориентирован передом к носу дрона.

3. Последовательно устанавливайте дрон в каждую из указанных ориентаций до появления желтой рамки.
4. Держите дрон неподвижно до появления зеленой рамки.

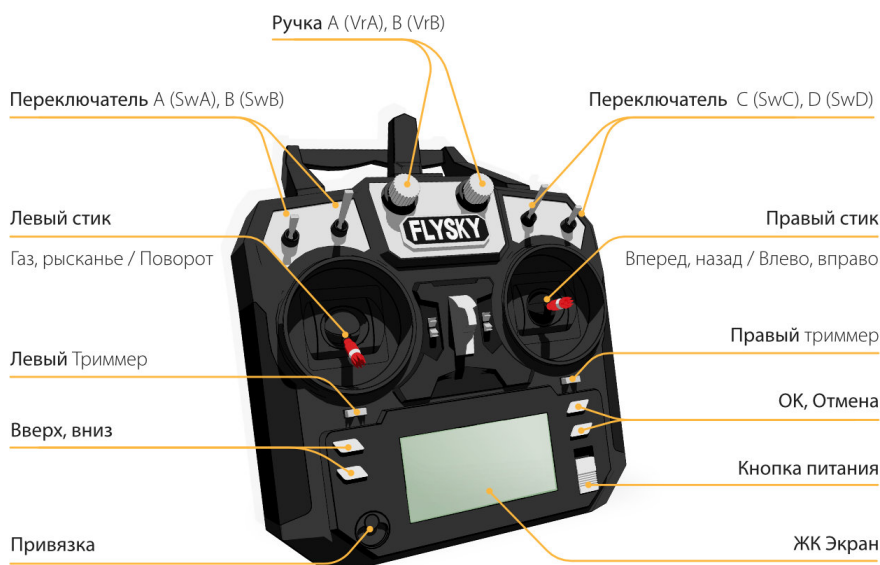
Уровень горизонта



1. Выберите меню *Level Horizon*.
2. Выберите ориентацию полетного контроллера – *ROTATION_NONE* при условии, что полетный контроллер ориентирован передом к носу дрона.
3. Установите дрон на ровную поверхность.
4. Нажмите *OK*.
5. Дождитесь окончания калибровки.

Далее: [Настройка пульта](#).

Настройка пульта



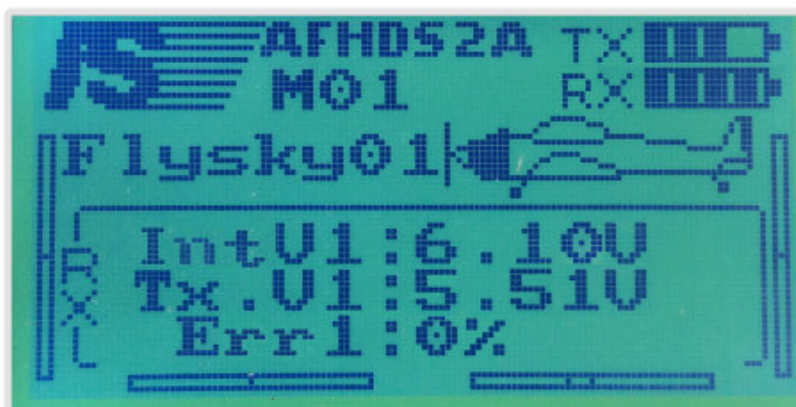
Перед подключением и калибровкой пульта убедитесь, что:

- К коптеру не подключено внешнее питание АКБ.
- Пропеллеры не установлены на моторах.

Подключение пульта

1. В программе QGroundControl перейдите в панель *Vehicle Setup* и выберите меню *Radio*.
2. Включите пульт, переводя переключатель *POWER* в верхнее положение.
3. Убедитесь, что связь с приемником установлена.

На ЖК Экране пульта высвечивается индикация:



Светодиод на приемнике должен гореть непрерывно красным. При наличии проблем с подключением читайте статью "[Неисправности радиоаппаратуры](#)".

Калибровка пульта

1. Нажмите кнопку *Calibrate*.
2. Установите триммеры *Throttle, Yaw, Pitch, Roll* в 0.
 - o Триммеры позволяют задавать смещение коптеру.
 - o Чтобы установить один из триммеров в 0, необходимо на пульте переместить указатель в центр до длительного звукового сигнала (писка).
3. Нажмите *OK*.
4. Переведите левый стик (газа) в минимум и нажмите *Next*.
5. Повторяйте движения стиками вслед за анимацией и читайте подсказки.
6. При появлении надписи "*Move all transmitter switches and/or dials back and forth to their extreme positions*" переключите SwA, SwD, VrA, VrB в их конечные положения.
7. Нажмите *Next*.
8. При появлении надписи "*All settings have been captured. Click Next to write the new parameters to your board*" нажмите *Next*.

Дополнительная информация:

<https://docs.qgroundcontrol.com/en/SetupView/Radio.html>.

Далее: [Настройка полетных режимов](#).

Работа с приёмником Flysky FS-X6B

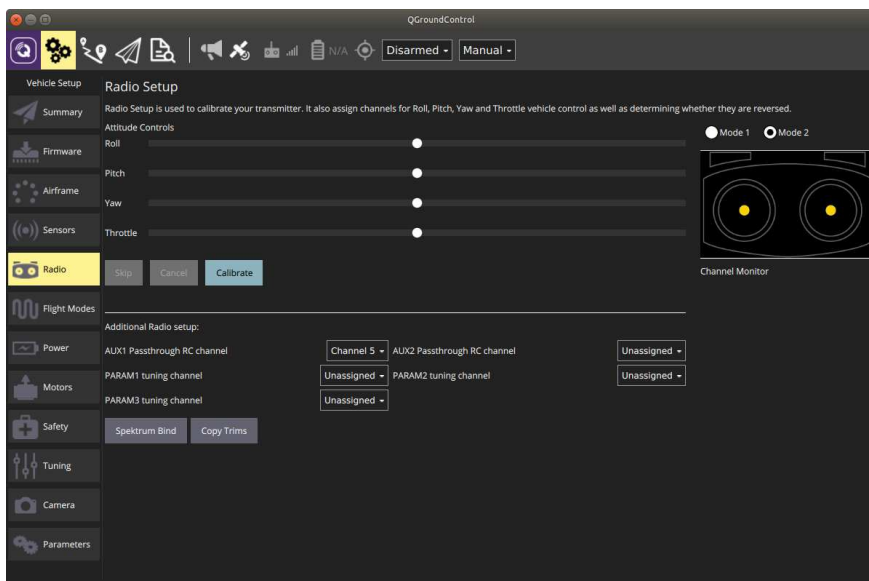
Приёмник Flysky FS-X6B совместим с пультами Flysky FS-i6 и FS-i6x. Связь с полётным контроллером может происходить как с использованием аналогового протокола PPM, так и при помощи цифровых протоколов S.Bus/i-Bus.

Для подключения к полётному контроллеру рекомендуется использовать протокол S.Bus.

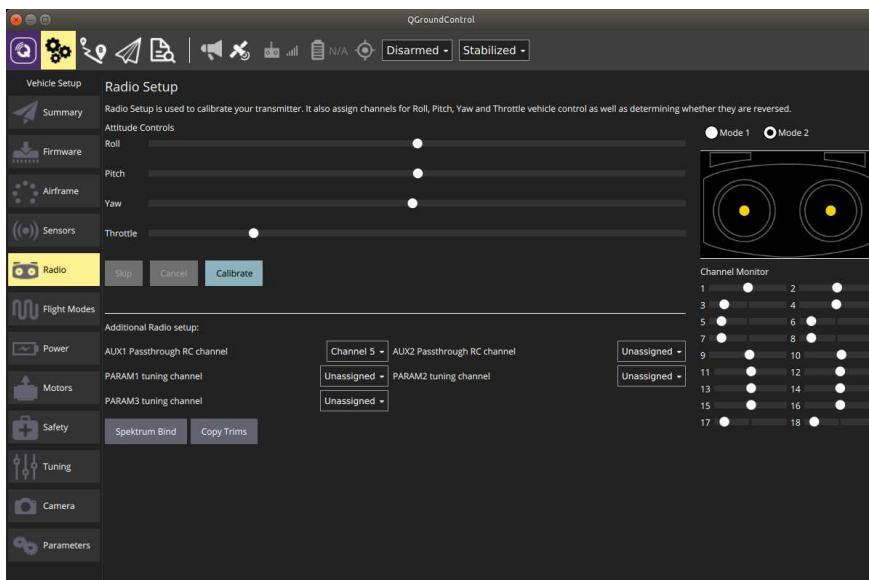
Подключите приёмник к полётному контроллеру к порту RC IN:

Выбор режима приёмника

Откройте QGroundControl и подключите полётный контроллер к компьютеру. Откройте вкладку Radio:



Если справа (под изображением пульта) не показано ни одного канала, нажмите кнопку **BIND** на приёмнике на 2 секунды. Должно появиться 18 каналов:



Полетные режимы

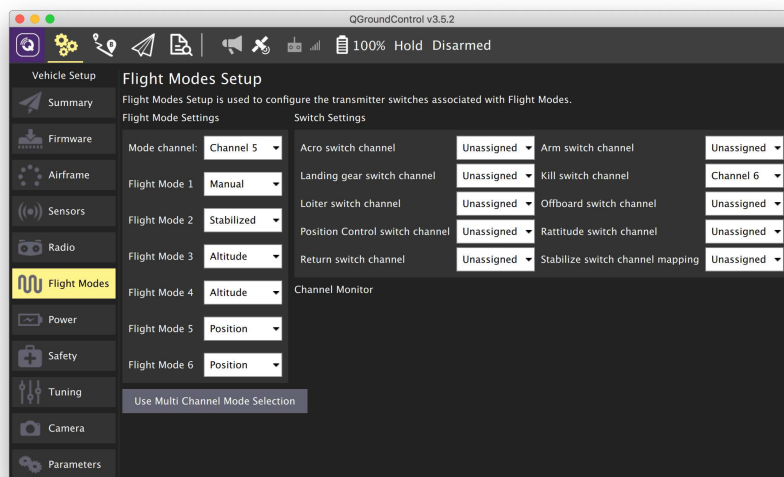
Режим полетного контроллера определяет, как именно дрон должен себя вести: каким образом интерпретировать входящие команды и сигналы с пульта. Режим переключается одним из переключателей на пульте радиуправления.

Чтобы настроить полетные режимы:

1. В программе QGroundControl перейдите в панель *Vehicle Setup*.
2. Выберите меню *Flight Modes*.
3. Установите переключатель режимов (*Mode Channel*) на переключатель SwC (*Channel 6*).
4. Опционально, установите экстренное отключение пропеллеров (*Emergency Kill Switch Channel*) на переключатель SwA (*Channel 5*).
5. Выберите необходимые полетные режимы.

Рекомендуемые полетные режимы:

- o Flight Mode 1: *Stabilized*.
 - o Flight Mode 4: *Altitude*.
 - o Flight Mode 6: *Position*.
6. Проверьте корректность переключения режимов, переключая переключатель на пульте.
 7. Назначьте аварийное отключение моторов (*Kill switch*) на переключатель SwA (Channel 5).



Подробное описание полетных режимов

Ручное управление

При ручном управлении пилот управляет дроном напрямую. GPS, данные с компьютерного зрения и барометр не используются. Для полетов в этих режимах необходимы хорошие навыки пилотирования мультикоптеров.

- **STABILIZED/MANUAL** — режим стабилизации горизонтального положения. Управление газом, углами наклона коптера по тангажу и крену, угловой скоростью по рысканью.
- **ACRO** — управление газом и угловой скоростью коптера по тангажу, крену и рысканью. Используется дрон-рейсерами и в шоу 3D-пилотирования для выполнения трюков.
- **RATTITUDE** — в центре правый стик аналогичен STABILIZED, по краям переходит в режим ACRO.

С использованием дополнительных датчиков

- **ALTCTL** (*Altitude*) — управление скоростью изменения высоты полета, углами по тангажу и крену и угловой скоростью по рысканью. Используется барометр (или иной датчик высоты).
- **POSCTL** (*Position*) — управление скоростями набора высоты, скоростью движения вперед/назад и вправо/влево, угловой скоростью по рысканью. Наиболее простой для полетов режим. Используется барометр, GPS, компьютерное зрение, другие датчики.

Автоматический полет

В этих режимах дрон игнорирует сигналы с пульта и летает по какой-либо автоматической программе.

- **OFFBOARD** — управление полетом с внешнего компьютера (например, [Raspberry Pi](#)). Этот режим используется для программирования автономных полетов.
- **AUTO.MISSION** — дрон выполняет заранее загруженную в него миссию. Миссия загружается при помощи QGroundControl. Этот режим чаще всего применяется для автоматических полетов по точкам с использованием GPS, например, для фотограмметрии.
- **AUTO.RTL** — коптер автоматически возвращается в точку взлета.
- **AUTO.LAND** — коптер выполняет посадку.

Далее: [Настройка питания](#).

Настройка питания

Чтобы откалибровать параметры, связанные с электропитанием, зайдите во вкладку *Vehicle Setup* и выберите меню *Power*.

Калибровка делителя напряжения

Калибровка делителя напряжения должна выполняться с подключенным АКБ.

1. В программе QGroundControl перейдите в панель *Vehicle Setup* и выберите меню *Power*.
2. Установите параметр *Number of cells* в соответствии с количеством банок в АКБ.
3. Откалибруйте делитель напряжения:
 - Подключите индикатор напряжения к балансировочному разъему АКБ.
 - Нажмите кнопку *Calculate* напротив надписи *Voltage Divider*.
 - Введите в открывшемся поле суммарное значение напряжения с индикатора напряжения.
 - Нажмите *Close*, чтобы сохранить рассчитанное значение.

Калибровка регуляторов (ESC)

Никогда не выполняйте калибровку регуляторов с установленными пропеллерами. В некоторых случаях моторы могут начать вращаться на максимальной скорости.

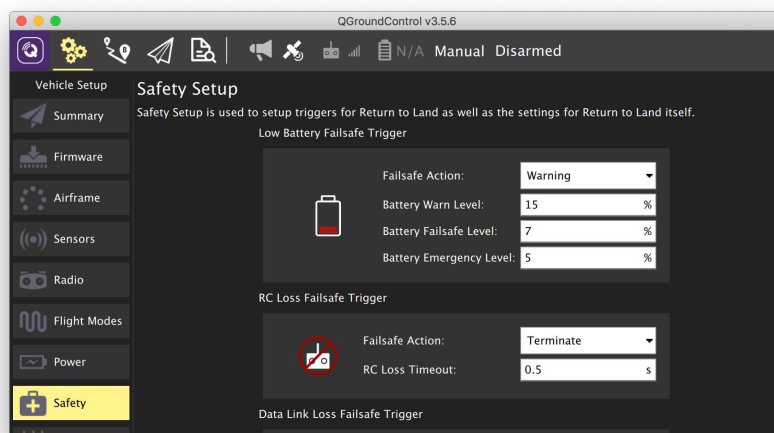
1. Убедитесь, что АКБ не подключена и пропеллеры сняты
2. Нажмите *Calibrate*.
3. После появлении надписи *Connect the battery now* подсоедините АКБ.
4. Дождитесь появления надписи *Calibration complete*.

Далее: [настройка Failsafe](#).

Настройка Failsafe

Во вкладке *Safety* настраиваются реакции дрона на различные нештатные ситуации. Рекомендуется включить как минимум реакцию на потерю связи с пультом управления:

1. В программе QGroundControl перейдите в панель *Vehicle Setup* и выберите меню *Safety*.
2. В блоке *RC Loss Failsafe Trigger* выберите один из рекомендуемых вариантов реакции на потерю связи с пультом:
 - o *Land mode* – переход в режим посадки;
 - o *Terminate* – аварийное отключение моторов.
3. В поле *RC Loss Timeout* выберите значение таймаута, по истечении которого связь с пультом считается потерянной. Рекомендуемое значение – 2 s.



Полет

Этот раздел объясняет основы управление дроном с использование пульта радиопередачи в различных режимах (для автономных полетов смотрите раздел "[Программирование](#)").

Основные возможности радиоаппаратуры

Прежде чем запускать ваш коптер, необходимо разобраться, как работает пульт радиопередачи ("аппаратура" в общепринятой терминологии авиамоделизма).

Управление дроном происходит с помощью двух стиков на аппаратуре. По умолчанию левый стик отвечает за газ и рысканье, а правый за крен и тангаж. Данные термины используются для всех летательных судов, от самолетов до дронов.

- Газ (*throttle*) – отвечает за скорость вращения двигателей.
- Рысканье (*yaw*) – отвечает за повороты вокруг вертикальной оси (Z), по часовой (при наклоне вправо) и против часовой (при наклоне влево) стрелки.
- Тангаж (*pitch*) – отвечает за наклон или движение вперед/назад.
- Крен (*roll*) – отвечает за наклон или движение влево/вправо.

Полетные режимы

Ручное полет с использованием полетного контроллера может происходить с использованием разных полетных режимов, которые определяют назначения стиков радиопульта и другие характеристики полета. Полный список полетных режимов приведен в статье "[Полетные режимы](#)".

Основные ручные режимы разобраны далее.

STABILIZED - режим стабилизации горизонтального положения. В данном режиме коптер будет удерживать горизонт, если им не управлять.

Назначение стиков:

- Газ – усредненная скорость вращения моторов.
- Рысканье – угловая скорость вокруг вертикальной оси.
- Тангаж – угол наклона вокруг поперечной оси (вперед/назад).
- Крен – угол наклон вокруг продольной оси (влево/вправо).

POCTL – режим удержания позиции (требуется включенная система позиционирования). Назначение стиков:

- Газ - вертикальная скорость полета.
- Рысканье - угловая скорость вокруг вертикальной оси.

- Тангаж - линейная скорость полета дрона (вперед/назад).
- Крен - линейная скорость полета дрона (влево/вправо).

ACRO – режим управление средней скоростью вращения моторов и угловыми скоростями дрона. Этот режим является наиболее сложным для пилотирования и чаще всего применяется дрон-рейсерами и в шоу 3D-пилотирования для выполнения трюков. Назначение стиков:

- Газ – усредненная скорость вращения моторов.
- Рысканье – угловая скорость вокруг вертикальной оси.
- Тангаж – угловая скорость вокруг поперечной оси (вперед/назад).
- Крен – угловая скорость вокруг продольной оси (влево/вправо).

В других полетных контроллерах аналогичные полетные режимы могут называться по-другому.

Подготовка к полету

Состояния готовности к полету

Прежде чем начинать полет, необходимо перевести коптер в состояние *Armed*.

- Состояние *Armed* – моторы вращаются в соответствии с положением стика газа, коптер готов к полету.
- Состояние *Disarmed* – моторы не вращаются, коптер не реагирует на стик газа.

По умолчанию коптер находится в состоянии *Disarmed* и переходит в него в случае если вы долго не взлетаете.

Для перевода коптера в состояние *Armed* есть несколько способов:

- С помощью стика – переведите левый стик вниз вправо и подождите



пару секунд.

- С помощью тумблера – состояния Armed/Disarmed можно настроить на один из тумблеров. Подробнее о настройке в смотрите в статье про [полетные режимы](#).
- С помощью QGC – вы можете заармить ваш дрон программно. Для этого нажмите на надпись *Disarmed* в шапке и выберите другое состояние.
- С помощью [программы](#) - коптер может перейти в состояние *Armed*, если в навигационной команде, такой как `navigate` , `set_position` и т.д., указан параметр `auto_arm=True` .

Kill switch

При активации тумблера *Kill Switch* на моторы перестают посылаются сигналы управления, и моторы перестают вращаться. Эта функция используется в крайних случаях, к примеру, если вы потеряли управление над коптером.

Будьте внимательны, *Kill Switch* не переводит коптер в состояние *Disarmed*!

Перед отключением *Kill Switch* убедитесь, что стик газа находится в нижнем положении и коптер находится в состоянии *Disarmed*. В случае, если стик газа не находится в нижнем положении, при отключении *Kill Switch* на моторы будет подан сигнал соответствующий положению стика в данный момент, что приведет к резкому рывку коптера.

Далее: [Упражнения для управления коптером](#).

Упражнения для управления дроном

Далее описаны рекомендуемые упражнения для тех, кто учится летать на дроне в первый раз. Повторяйте каждое упражнение необходимое количество раз, пока не будете чувствовать себя уверенно в нем.

В случае, если рядом есть человек умеющий управлять дроном, используйте [режим тренера](#).

Настоятельно рекомендуется первые полеты проводить за защитной сеткой. В случае отсутствия таковой, полетная зона должна быть не менее 6х6 м.

Включение, выключение моторов, изменение режимов

Для удобства подключитесь к коптеру с помощью [QGC через Wi-Fi](#) и включите звук. Это позволит наблюдать за изменением полетных режимов. Если не имеется возможности подключиться через Wi-Fi, для проверки полетных режимов подключитесь по USB.

Убедитесь, что настроили полетные режимы на один из тумблеров. Для этого переключите тумблер, в разные позиции и убедитесь, что режимы изменяются.

Рекомендуется настроить *Kill Switch*. Для его проверки совершите следующие действия:

- Включите *Kill Switch*, проверьте, что в QGC появилось соответствующее уведомление.
- Переведите коптер в состояние *Armed*, а затем включите *Kill Switch*. Убедитесь, что моторы выключились. Затем переключите тумблер *Kill Switch* в изначальное положение. Если коптер автоматически не перешел в состояние *Disarmed* из-за бездействия, моторы снова начнут вращаться.

Переводите коптер в состояние *Armed* только на полетной зоне.

Убедитесь, что режимы переключаются удобными для вас тумблерами. В противном случае измените их в соответствии со [статьей по настройке полетных режимов](#). Повторите приведенные действия несколько раз, для того, чтобы запомнить какие тумблеры за что отвечают.

Работа с газом

Первым делом необходимо почувствовать отзывчивость коптера на движение стика газа и научиться им управлять. Каждый коптер имеет немного различные запасы мощности и соответственно отрывается от земли при разных положениях стика.

В данном упражнении необходимо использовать только стик газа. Во время выполнения упражнения рекомендуется не использовать остальные стики.

Основные задания упражнения:

1. Дрейф коптера по земле, не отрываясь от земли.

Предполетные проверки

Перед взлетом выполняйте следующие действия:

1. Проверьте целостность коптера и возможность вращения пропеллеров.
2. Убедитесь, что коптер находится задней частью к вам.
3. Включите коптер путем подключения АКБ.
4. Отойдите на безопасное расстояние. Рекомендуется соблюдать расстояние до коптера минимум 4–5 м.
5. Убедитесь, что коптер находится в режиме *Stabilized*.

Не пытайтесь сразу оторвать коптер от земли, найдите минимально возможное положение стика для того, чтобы коптер начал дрейфовать по земле. В противном случае это может привести к поломкам или травмам.

В случае потери контроля над коптером необходимо сразу включать *Kill Switch*.

Упражнение №1. Медленно поднимайте стик газа вверх, пока коптер не начнет двигаться. В этот момент он начнет медленно дрейфовать по земле. Оставьте стик газа в таком положении и подождите пару секунд, затем переведите стик в изначальное положение, чтобы посадить коптер. После посадки коптера выключите моторы переводя в состояние *Disarmed*. Повторите упражнение 5-10 раз, чтобы лучше чувствовать отзывчивость коптера на стик газа.

Упражнение №2. Медленно поднимайте стик газа вверх, пока коптер не начнет немного отрываться от земли. Оставьте стик газа в таком положении и подождите пару секунд, затем посадите коптер аналогично упражнению №1. Повторите упражнение 10-15 раз.

Упражнение №3. Поднимайте стик газа, пока коптер не начнет дрейфовать по земле, подождите секунду и продолжайте увеличивать газ до момента, когда коптер начнет отрываться от земли, подождите секунду и посадите коптер. Для закрепления повторяйте упражнения 10-15 раз, при необходимости увеличивая количество повторений.

Работа с креном и тангажом

После освоения управления газом коптера, необходимо научиться управлять его горизонтальным положением. За это отвечает правый стик на радиоаппаратуре.

Управление данными осями интуитивно понятно:

- Стик наклонен вперед (вверх) – коптер движется вперед.
- Стик наклонен назад (вниз) – коптер движется назад.
- Стик наклонен вправо – коптер движется вправо.
- Стик наклонен влево – коптер движется влево.

Чем сильнее стик будет наклонен в сторону, тем сильнее коптер будет наклоняться в сторону быстрее двигаться.

Основные задания упражнения:

1. Полет по оси X, вперед/назад.
2. Полет по оси Y, влево/вправо.
3. Стабилизация коптера на одном месте.
4. Полет по квадрату по часовой стреле и против.

Старайтесь всегда находиться позади коптера, таким образом, чтобы его задняя часть была направлена к вам, иначе вы можете потерять управление над ним, перепутав стороны.

Как и в случае с управлением газом, перед полетом выполняйте [следующие действия](#).

Если коптер сильно вращается вокруг своей оси, посадите его и повторно откалибруйте магнитометр и гироскоп.

Упражнение №1. Аналогично упражнениям по управлению газом поднимайте стик газа, пока коптер не начнет дрейфовать по земле или немного подпрыгивать, затем опустите стик газа, оставив его в таком положении, и поднимайте стик тангажа, сначала вверх, на протяжении секунды, затем вниз. При этом коптер будет постепенно перемещаться сначала от вас, а затем к вам. Повторите упражнение 5-10 раз, пока не почувствуете отзывчивость коптера на движение стика.

Упражнение №2. Поднимайте стик газа, пока коптер не начнет дрейфовать, затем оставьте его и перемещайте стик крена сначала вправо, на протяжении секунды, затем влево. При этом коптер будет постепенно перемещаться сначала вправо, а затем влево. Повторите упражнение 5-10 раз, пока не почувствуете отзывчивость коптера на движение стика.

Упражнение №3. Поднимайте стик газа, пока коптер не начнет дрейфовать, затем оставьте его. Совместите первое и второе упражнение и постарайтесь стабилизировать коптер в одной точке, компенсируя его дрейф с помощью стика. Удерживайте коптер 20-30 секунд.

Упражнение №4. Поднимайте стик газа, пока коптер не начнет дрейфовать, затем оставьте его. Почувствовав отзывчивость коптера на изменения стиков выполните фигуру "квадрат" со стороной 1 м, сначала по часовой стрелке, а

затем против. Выполняйте фигуры по 2-3 раза.

Воздушная подушка и управление в ней

Понятие *воздушной подушки* очень важно во всей летательной технике. Сама по себе воздушная подушка является зоной повышенного давления, возникающая за счет воздуха пропускаемого через пропеллеры. Данная область характеризуется турбулентностями и воздушными потоками влияющими на полет коптера.

Пилоты стараются избегать полетов в воздушной подушке, но на ее границе имеется стабильная область, в которой коптер может зависнуть при минимальном значении газа. В таком случае создается ощущение, что коптер "сел" на воздушную подушку.

Главная особенность и преимущество такого полета заключается в том, что коптер не будет изменять высоту при одном значении газа.

Основные задания:

1. Стабилизация коптера на одном месте.
2. Полет по квадрату.
3. Полет по кругу.

Аналогично с предыдущими упражнениями перед взлетом выполните [следующие действия](#).

Упражнение №1. Поднимайте стик газа, пока коптер не пролетит воздушную подушку и не окажется над ней (высота от пола ~25-30 см). Коптер не должен подниматься вверх или проваливаться вниз, высота полета должна стабилизироваться. Как и в предыдущем упражнении корректируйте позицию коптера по осям X, Y с помощью стика крена и тангажа. В результате коптер должен зависнуть в одной точке с небольшими покачиваниями по сторонам. Удерживайте коптер 30-40 секунд.

Упражнение №2. Поднимите коптер на воздушную подушку и стабилизируйте его в одной точке. Далее пролетите по квадрату со стороной 1 м сначала по часовой стрелке, потом против часовой стрелки. Повторите траекторию в каждую сторону 2-3 раза.

Упражнение №3. Поднимите коптер на воздушную подушку и стабилизируйте его в одной точке. Попробуйте описать коптером круг с диаметром 1 м, по часовой и против часовой стрелки. Повторите траекторию в каждую сторону 2-3 раза.

Работа с рысканьем

При визуальном управлении дроном, рысканье не играет на столько важной роли, как с самолетной технике, поскольку коптер может передвигаться в любую сторону вне зависимости от того, куда он направлен.

Термин *рысканье (yaw)* обозначает поворот коптера вокруг вертикальной оси.

Основные задания:

1. Оборот коптера вокруг себя, ориентируя заднюю часть коптера к себе.
2. Оборот вокруг коптера, ориентируя заднюю часть к себе.

Для выполнения представленных упражнений рекомендуется найти большое свободное пространство.

Аналогично с предыдущими упражнениями перед взлетом выполните [предполетные проверки](#).

Упражнение №1. Поднимите коптер на воздушную подушку и стабилизируйте его в одной точке. Описывайте коптером круг вокруг себя, на расстоянии 2-3 м, при этом поворачивая его таким образом, чтобы задняя часть коптера всегда была направлена на вас. Выполняйте упражнение по часовой стрелке и против. Повторите упражнение 4-5 раз.

Упражнение №2. Поднимите коптер на воздушную подушку и стабилизируйте его в одной точке. Обойдите коптер вокруг, при этом поворачивая его таким образом, чтобы задняя часть была направлена на вас. Обойдите коптер по часовой стрелке и против. Повторите упражнение 4-5 раз.

Дополнительные упражнения значительно сложнее обычных и не обязательны к выполнению. Приступайте к ним, только если вы уже уверенно управляете коптером.

Дополнительное упражнение №1. Поднимите коптер на воздушную подушку и стабилизируйте его в одной точке. Разверните коптер передней частью к себе и пробуйте управлять им задом наперед.

Дополнительное упражнение №2. Поднимите коптер на воздушную подушку и стабилизируйте его в одной точке. Выполняйте полет таким образом, чтобы передняя часть коптера всегда смотрела в сторону его движения.

Свободный полет

Если вы можете выполнить каждое из описанных выше упражнений, скорее всего, вы уже умеете свободно взлетать и управлять коптером. Далее будут представлены некоторые упражнения для закрепления полученных навыков.

Упражнения:

- Полет по вертикальному квадрату.
- Полет по граням куба.
- Полет по вертикальному кругу.
- Полет по восьмерке.
- Подъем коптера по спирали.

Датчик температуры и влажности

Закрепляйте полученные навыки необходимое для вас количество раз.

Raspberry Pi

Raspberry Pi – это свободно помещающийся на ладони одноплатный компьютер, созданный на базе мобильного микропроцессора ARM. Он обладает низким энергопотреблением и может работать даже от солнечных батарей.



Технические характеристики:

- Broadcom BCM2711 SoC;
- 64-битный четырехъядерный ARMv8 Cortex-A72 процессор с тактовой частотой 1.5 ГГц;
- Графический сопроцессор VideoCore VI;
- Память на 1/2/4/8ГБ LPDDR4 SDRAM;
- Gigabit Ethernet;
- USB3.0;
- 2 x micro-HDMI;
- 2.4 ГГц и 5 ГГц IEEE 802.11.b/g/n/ac Wi-Fi + Bluetooth 5.0 Low Energy (BLE).

Более подробную информацию читайте на странице [Raspberry Pi 4 Model B](#).

Raspberry Pi подключается к полетному контроллеру и используется как вспомогательный компьютер. Он позволяет [подключаться к дрону по Wi-Fi](#), программировать автономные полеты, работать с периферией и многое другое.

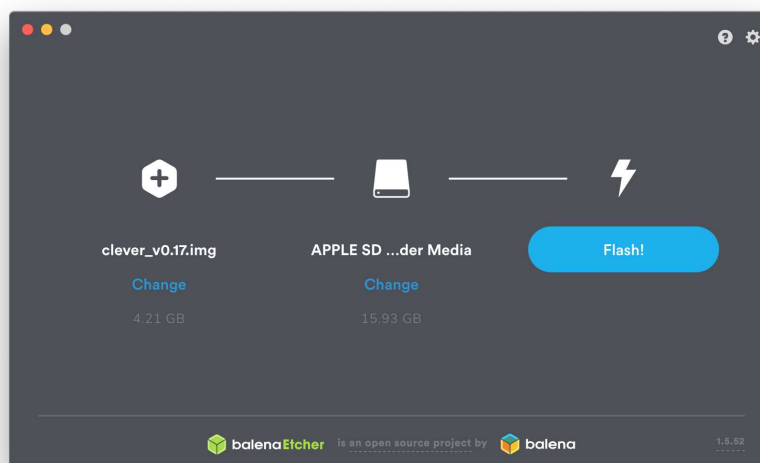
Далее: [образ для Raspberry Pi](#).

Образ для Raspberry Pi

Образ RPi включает в себя все необходимое ПО для удобной работы с дроном и [программирования автономных полетов](#). Платформа основана на операционной системе [Raspbian](#) и популярном робототехническом фреймворке [ROS](#).

Использование

1. Скачайте образ — [ссылка](#).
2. Скачайте и установите [программу для записи образов Etcher](#) (доступна для Windows/Linux/macOS).
3. Установите MicroSD-карту в компьютер (используйте адаптер при необходимости).
4. Запишите скачанный образ на карту, используя Etcher.
5. Установите карту в Raspberry Pi.



После записи образа на SD-карту, вы можете подключаться к дрону [по Wi-Fi](#), использовать [беспроводное соединение в QGroundControl](#), получать [доступ по SSH](#) и использовать остальные функции.

Далее: [Подключение по Wi-Fi](#).

Подключение по Wi-Fi

На [образе для RPi](#) преднастроена раздача Wi-Fi с SSID `drone-xxxx`, где `xxxx` – 4 случайных цифры, назначаемых при первом включении Raspberry Pi.

Подключитесь к Wi-Fi, используя пароль `dronewifi`.

Для изменения настроек Wi-Fi или получения более детальной информации о устройстве сети на Raspberry Pi прочитайте статью "[Настройка Wi-Fi](#)".

Веб-интерфейс

После подключения к по адресу <http://192.168.11.1> будет доступен веб-интерфейс. В нем доступны основные веб-инструменты: просмотр топиков с изображениями, веб-терминал (Butterfly) а также полная копия данной документации.

Далее: [Подключение Raspberry Pi к полетному контроллеру](#).

Подключение Raspberry Pi к полетному контроллеру

Для программирования [автономных полетов](#) и других функций необходимо соединение Raspberry Pi и полетного контроллера.

Подключение по USB

Основным способом подключения является подключение по интерфейсу USB.

1. Соедините Raspberry Pi и полетный контроллер micro-USB to USB кабелем.
2. [Подключитесь в Raspberry Pi по SSH](#).
3. Убедитесь в работоспособности подключения, [выполнив на Raspberry Pi](#):

```
rostopic echo /mavros/state
```

Поле `connected` должно содержать значение `True` .

Далее: [Подключение QGroundControl по Wi-Fi](#).

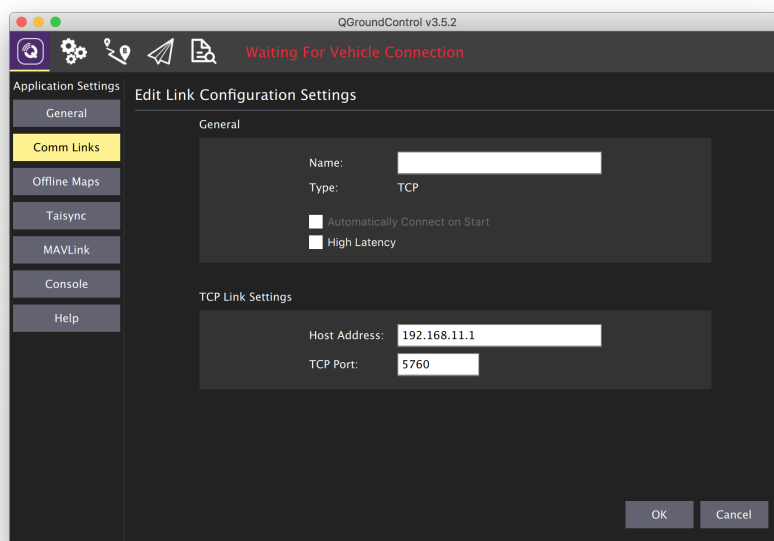
Подключение QGroundControl по Wi-Fi

Возможны контроль, управление, калибровка и настройка полетного контроллера дрона с помощью программы QGroundControl по Wi-Fi. Для этого необходимо [подключиться к Wi-Fi](#) сети `drone-xxxx`.

Подключение

По умолчанию используйте подключение QGroundControl по протоколу TCP.

1. На первой вкладке QGroundControl выберите меню *Comm Links*.
2. Нажмите кнопку *Add*, чтобы добавить новое подключение.
3. Введите параметры подключения:
 - o Name: *Drone*.
 - o Type: *TCP*.
 - o Host Address: *192.168.11.1*.
 - o TCP Port: *5760*.
4. Нажмите *OK* для сохранения параметров.
5. Выберите созданное подключение и нажмите *Connect*.



UDP

Также возможна настройка подключения по протоколу UDP. Для выбора различных вариантов подключения по UDP необходимо отредактировать параметр `gcs_bridge` в `launch`-файле

```
/home/pi/catkin_ws/src/drone/drone/launch/drone.launch
```

После изменения `launch`-файла необходимо перезагрузить сервис `drone`:

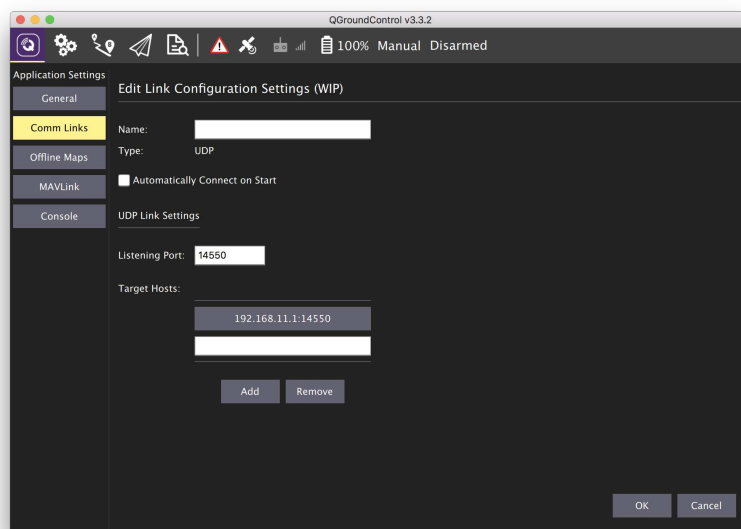
```
sudo systemctl restart drone
```

UDP-бридж с автоматическим подключением

1. Измените параметр `gcs_bridge` на `udp-b` .
2. При открытии программы QGroundControl соединение должно установиться автоматически.

UDP-бридж без автоматического подключения

1. Измените параметр `gcs_bridge` на `udp` .
2. В QGroundControl создайте подключение со следующими настройками:



3. Выберите созданное подключение и нажмите *Connect*.

UDP broadcast-бридж

Особенностью UDP broadcast-бриджа является возможность просмотра телеметрии дрона одновременно с нескольких устройств. Также он хорошо подходит для организации сети из устройств при помощи роутера.

1. Измените параметр `gcs_bridge` на `udp-pb` .
2. При открытии программы QGroundControl соединение должно установиться автоматически.

Далее: [Доступ по SSH](#).

Доступ по SSH к Raspberry Pi

На [образе для RPi](#) преднастроен доступ по SSH для редактирования файлов, загрузки данных и запуска программ.

Для доступа по SSH необходимо [подключиться к Raspberry Pi по Wi-Fi](#) (также возможно подключение через Ethernet-кабель).

В GNU/Linux или macOS необходимо запустить Терминал и выполнить команду:

```
ssh pi@192.168.11.1
```

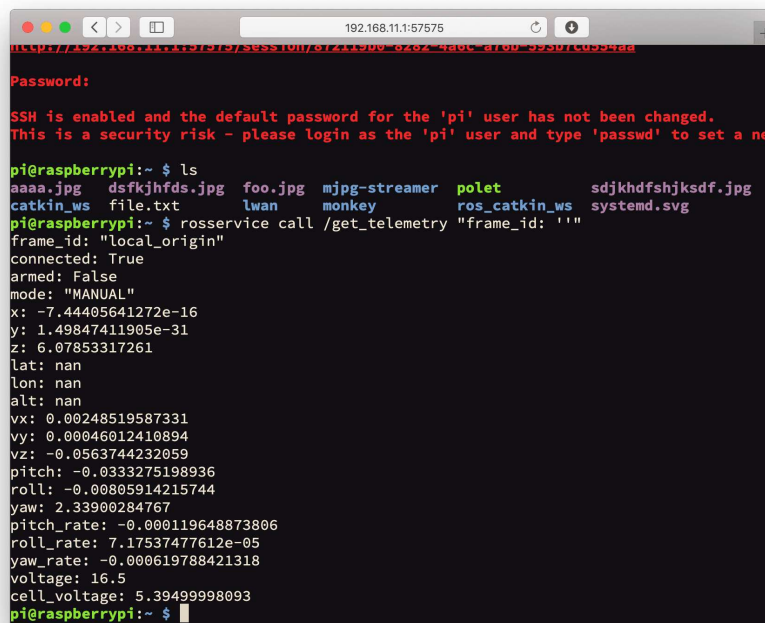
Пароль: `raspberrypi`.

Для доступа по SSH из Windows можно использовать [PuTTY](#) или веб-доступ (см. далее).

Подробнее: <https://www.raspberrypi.org/documentation/remote-access/ssh/README.md>.

Веб-доступ

Доступ к шеллу также доступен через веб-браузер (с использованием [Butterfly](#)). Для доступа откройте страницу <http://192.168.11.1> и выберите на ней ссылку *Open web terminal*:



```
192.168.11.1:57575
http://192.168.11.1:57575/3955101/87441900-9282-4a0c-af00-39307c0594ad
Password:
SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set a ne

pi@raspberrypi:~$ ls
aaaa.jpg  dsfkhjfds.jpg  foo.jpg  mjpg-streamer  polet  sdjkhdfshjksdf.jpg
catkin_ws  file.txt       lwan    monkey         ros_catkin_ws  systemd.svg
pi@raspberrypi:~$ rosservice call /get_telemetry "frame_id: ''"
frame_id: "local_origin"
connected: True
armed: False
mode: "MANUAL"
x: -7.44405641272e-16
y: 1.49847411905e-31
z: 6.07853317261
lat: nan
lon: nan
alt: nan
vx: 0.00248519587331
vy: 0.00046012410894
vz: -0.0563744232059
pitch: -0.0333275198936
roll: -0.00805914215744
yaw: 2.33900284767
pitch_rate: -0.000119648873806
roll_rate: 7.17537477612e-05
yaw_rate: -0.000619788421318
voltage: 16.5
cell_voltage: 5.39499998093
pi@raspberrypi:~$
```

Далее: [Командная строка](#).

Командная строка

В Linux-системах, к семейству которых принадлежит используемая на Raspberry Pi ОС Raspbian, основным способом взаимодействия пользователя с системой является командная строка. Для работы с командной строкой [откройте SSH-соединение](#) с Raspberry Pi.

Базовые команды

Двойное нажатие клавиши `Tab` `↵` позволяет автоматически дополнить вводимую команду или аргумент.

Показать содержимое текущей директории:

```
ls
```

Перейти в директорию:

```
cd catkin_ws/src/drone/drone/launch/
```

Перейти на директорию выше:

```
cd ..
```

Вывести путь к текущей директории:

```
pwd
```

Вывести содержимое файла `file.py` :

```
cat file.py
```

Запустить Python-скрипт `file.py` :

```
python3 file.py
```

Перезагрузить Raspberry Pi:

```
sudo reboot
```

Для завершения работающей программы нажмите комбинацию клавиш

```
ctrl + c .
```

Читайте больше о командах Linux в документации Raspberry Pi:

<https://www.raspberrypi.org/documentation/linux/usage/commands.md>.

Редактирование файлов

Используйте редактор **nano** для того, чтобы создавать или редактировать файлы на Raspberry Pi. Среди текстовых редакторов, доступных в терминале, он является наиболее простым и интуитивным.

1. Для редактирования файла введите команду:

```
nano путь/к/файлу
```

2. Отредактируйте файл.

3. Для выхода с сохранением нажмите `ctrl + x`, `y`, `Enter`.

4. При изменении `.launch`-файлов необходимо перезапустить пакет `drone`:

```
sudo systemctl restart drone
```

Для редактирования файлов также можно использовать и другие редакторы, например, **vim**.

Автоматическая проверка

[Образ](#) включает в себя средство автоматической проверки корректности настроек и работы всех подсистем дрона — **selfcheck.py**.

Для запуска наберите в [консоли Raspberry Pi](#):

```
roslaunch drone selfcheck.py
```

Описание некоторых проверок:

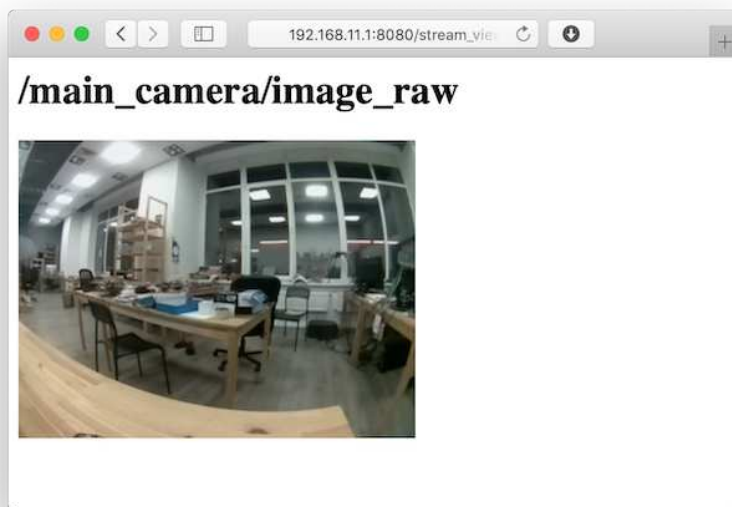
- FCU – проверка корректности соединения с полетным контроллером;
- IMU – проверка корректности данных с IMU;
- Local position – наличие локальной позиции дрона;
- Velocity estimation – оценка скоростей дрона (**запрещено выполнять автономный взлет при ошибках в этой проверке!**);
- Global position (GPS) – наличие глобальной позиции (требуется GPS);
- Camera – корректная работа камеры Raspberry.
- ArUco – проверка работы [распознавания ArUco-маркеров](#).
- VPE – проверка правильности работы VPE.
- Rangefinder – проверка работы [дальномера](#).
- RPi health – проверка состояния [бортового компьютера](#).
- CPU usage – проверка загруженности процессора бортового компьютера.

Просмотр изображений с камер

Для просмотра изображений с камер (или других ROS-топиков) можно смотреть их через браузер, используя `web_video_server`.

Просмотр через браузер

Для просмотра видеострима нужно [подключиться к Wi-Fi drone-xxxx](#), перейти на страницу <http://192.168.11.1:8080/> и выбрать топик.



Если передача картинки работает слишком медленно, можно ускорить ее, указав тип передаваемых данных `mjpeg` и меняя GET-параметр `quality` (от 1 до 100), который отвечает за сжатие видеострима, например:

[http://192.168.11.1:8080/stream_viewer?
topic=/main_camera/image_raw&type=mjpeg&quality=1](http://192.168.11.1:8080/stream_viewer?topic=/main_camera/image_raw&type=mjpeg&quality=1)

По URL выше будет доступен стрим с основной камеры в минимальном возможном качестве.

Также доступны параметры `width`, `height` и другие. Подробнее о `web_video_server` : http://wiki.ros.org/web_video_server.

Программирование

Платформа позволяет использовать [Raspberry Pi](#) для того, чтобы запрограммировать автономный полет дрона. Чаще всего программа для автономного полета пишется на языке Python. Программа может [получать телеметрию](#) (заряд батареи, ориентацию, положение, скорости) и отправлять команды, например: [полететь в точку](#), [установить ориентацию](#), [установить угловую скорость](#).

Платформа основывается на [фреймворке ROS](#), который обеспечивает связь между пользовательской программой и сервисами, которые запущены в фоне в виде systemd-демона `drone`.

Для автономного полета используется [режим OFFBOARD](#). API переводит дрон в этом режим автоматически. В случае необходимости прерывания автономного полета, необходимо перевести дрон в любой другой режим, используя стик переключения режимов на пульте.

Модули и датчики

Во вкладке [модули и датчики](#) вы можете посмотреть их подключение и программирование.

Система позиционирования

Для того, чтобы дрон мог зависать на месте или летать между точками, необходимо использование системы позиционирования. Такая система вычисляет и сообщает дрону, где он находится. Предполагается использование систем позиционирования: [лазерный дальномер](#), [визуальные маркеры](#) (используется камера и маркеры, наклеенные на пол или потолок), GPS и других.

ArUco-маркеры

Технология визуальных маркеров позволяет рассчитать позицию дрона относительно распознанных маркеров и передать эту информацию в полетный контроллер.

Читайте [статью про ArUco-маркеры](#) для получения подробностей.

GPS (уличный полет)

Использование GPS позволяет также использовать для навигации глобальные координаты – широту и долготу (функция `navigate_global`).

Основная статья: [подключение GPS](#).

Автономный полет

Для изучения языка программирования Python можно обратиться к [самоучителю](#).

После настройки системы позиционирования становится возможным написание скриптов для автономных полетов. Для выполнения скриптов [подключитесь в Raspberry Pi по SSH](#).

Перед первым полетом рекомендуется проверить конфигурацию при помощи [утилиты selfcheck.py](#):

```
roslaunch drone selfcheck.py
```

Для того, чтобы запустить Python-скрипт, используйте команду `python3` :

```
python3 flight.py
```

Пример программы для полета (взлет, пролет вперед, посадка):

```
import rospy
from drone import srv
from std_srvs.srv import Trigger

rospy.init_node('flight')

get_telemetry = rospy.ServiceProxy('get_telemetry', srv.GetTelemetry)
navigate = rospy.ServiceProxy('navigate', srv.Navigate)
navigate_global = rospy.ServiceProxy('navigate_global', srv.NavigateGlobal)
set_position = rospy.ServiceProxy('set_position', srv.SetPosition)
set_velocity = rospy.ServiceProxy('set_velocity', srv.SetVelocity)
set_attitude = rospy.ServiceProxy('set_attitude', srv.SetAttitude)
set_rates = rospy.ServiceProxy('set_rates', srv.SetRates)
land = rospy.ServiceProxy('land', Trigger)

# Взлет на высоту 1 м
navigate(x=0, y=0, z=1, frame_id='body', auto_arm=True)

# Ожидание 3 секунды
rospy.sleep(3)

# Пролет вперед 1 метр
navigate(x=1, y=0, z=0, frame_id='body')

# Ожидание 3 секунды
rospy.sleep(3)

# Посадка
land()
```

Функция `navigate` не ожидает, пока дрон долетит до целевой точки; скрипт продолжит выполнение сразу. Для блокирующей версии смотрите пример функции [navigate_wait](#) .

Обратите внимание, что параметр `auto_arm` установлен на `true` только у первого вызова функции `navigate`. Этот параметр армит дрон и переводит его в режим автономного полета (OFFBOARD).

Параметр `frame_id` задает систему координат, относительно которой задаются целевая точка для полета дрона:

- `body` связана с корпусом дрона;
- `navigate_target` связана с предыдущей целевой точкой полета;
- `map` связана с локальной системой координат дрона;
- `aruco_map` связана с картой ArUco-маркеров;
- `aruco_N` связана ArUco-маркером с ID=N.

Подробности описаны в статье "[Системы координат](#)".

Полное описание API приведено в статье "[Автономный полет](#)".

Также доступна поддержка [блочного программирования](#) автономных полетов.

Дополнительное оборудование

Платформа также имеет API для работы с периферией. Читайте соответствующие статьи для подробностей:

- [работа со светодиодной лентой](#);
- [лазерный дальномер](#);
- [GPIO](#);
- [ультразвуковой дальномер](#);
- [камера](#).

Модули и датчики

Датчик – это устройство, воспринимающее внешние воздействия и реагирующее на них изменением электрических сигналов.

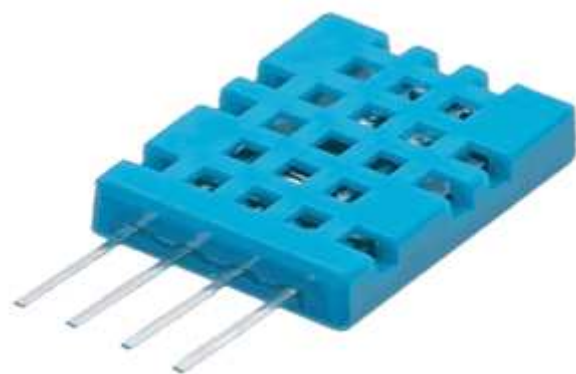
Модуль — функционально завершённый узел радиоэлектронной аппаратуры, оформленный конструктивно как самостоятельный продукт.

Если вы будете использовать/устанавливать сторонние библиотеки для программирования модулей/датчиков, необходимо [настроить сеть RPi в режим клиента](#). В этом случае появится доступ к интернету на RPi. **Рекомендация** Рекомендуем создавать разные директории под разные модули, в которых вы будете создавать скрипты и устанавливать библиотеки: создаете директорию (`mkdir` название-директории, если нужно создать директорию в определенном месте, переходите туда с помощью команды `cd` путь-к-директории, затем создаете свою в этом месте). **Подсказка** Если не указано иное, все скрипты используют язык **Python** с соответствующим расширением файла `.py`. Для выполнения любого скрипта вам необходимо: **1)** создать файл с названием и расширением (`nano` имя-файла.расширение), **2)** записать в файл ваш код, **3)** запустить файл через командную строку (`sudo` имя-файла).

№	The module name	Наименование датчика
1	DH11 temperature and humidity sensor module	Датчик температуры и влажности
2	HC-SR501 infrared human body induction module	Инфракрасный индукционный модуль для человеческого тела (датчик движения)
3	DS1302 real time clock module	Модуль часов реального времени
4	The rain sensor module	Датчик дождя
5	Sound sensor module	Датчик звука
6	HC-SR04 ultrasonic sensor	Ультразвуковой дальномер
7	The flame sensor module	Датчик огня
8	KY-008 laser head sensor module	Датчик лазерный
9	Photosensitive resistance sensor module	Светочувствительный датчик сопротивления
10	The YL-69 soil moisture sensor	Датчик влажности почвы
11	Obstacle avoidance sensor	Датчик обхода препятствий
12	Vibration sensor module	Датчик вибраций
13	MQ-2 gas sensor module	Датчик газа
14	The tilt sensor module	Датчик наклона
15	A path tracing module	Модуль трассировки пути
16	VL53L1X laser ranging sensor module	Лазерный дальномер

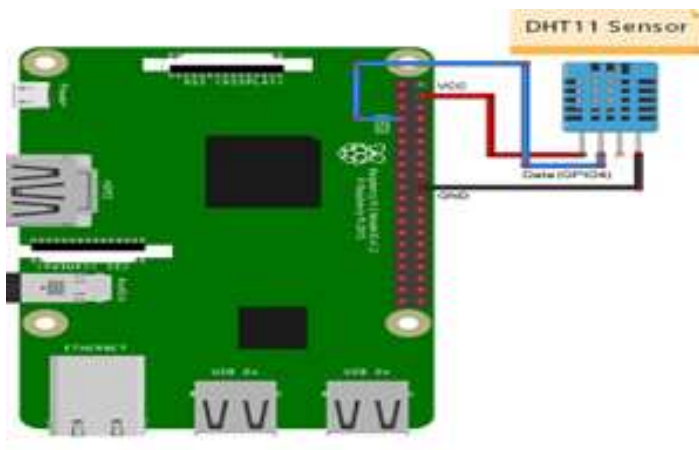
Датчик температуры и влажности (DHT11 temperature and humidity sensor module)

Датчик температуры и влажности DHT11 - это датчик, который обеспечивает цифровые показания температуры и влажности. Его легко настроить, и для передачи данных требуется всего один провод. Эти датчики популярны для использования на удаленных метеостанциях, в системах мониторинга почвы и домашней автоматизации.



- Датчик DHT11 измеряет и выдает значения влажности и температуры последовательно по одному проводу.
- Он может измерять относительную влажность в процентах (от 20 до 90% относительной влажности) и температуру в градусах Цельсия в диапазоне от 0 до 50 ° C.
- Он имеет 4 контакта, один из которых используется для передачи данных в последовательной форме.
- Импульсы разного уровня громкости и кратности декодируются как логический 1 или логический 0, или как начальный импульс, или как конец кадра.

Подключение



Программирование

Вывод на SSH-терминал

Будем использовать библиотеку Python Adafruit DHT11. Вы можете загрузить библиотеку с помощью Git, поэтому, если Git еще не установлен на вашем Pi, введите это в командной строке:

```
sudo apt-get install git-core
```

Если при установке Git возникает ошибка, запустите **sudo apt-get update** и повторите попытку.

Чтобы установить библиотеку Adafruit DHT11:

Введите это в командной строке, чтобы загрузить библиотеку:

```
git clone https://github.com/adafruit/Adafruit_Python_DHT.git
```

Измените каталоги с помощью:

```
cd Adafruit_Python_DHT
```

Теперь введите это:

```
sudo apt-get install build-essential python-dev
```

Затем установите библиотеку с помощью:

```
sudo python setup.py install
```

Код

```
#!/usr/bin/python
import sys
import Adafruit_DHT

while True:

    humidity, temperature = Adafruit_DHT.read_retry(11, 4)

    print 'Temp: {0:0.1f} C Humidity: {1:0.1f} %'.format(temperature, humidity)
```

Вывод на дисплей

Чтобы вывести показания DHT11 на дисплей, нам нужно установить еще одну библиотеку Python под названием RPLCD для управления дисплеем. Чтобы установить библиотеку RPLCD, нам сначала нужно установить индекс пакетов Python или PIP. Проверьте, установлен ли PIP на вашем RPi:

```
sudo apt-get install python-pip
```

После установки PIP установите библиотеку RPLCD:

```
sudo pip install RPLCD
```

После установки библиотеки вы можете использовать следующий код для вывода показаний DHT11 на ЖК-дисплей.

Код

```
#!/usr/bin/python
import sys
import Adafruit_DHT

from RPLCD import CharLCD

lcd = CharLCD(cols=16, rows=2, pin_rs=37, pin_e=35, pins_data=[33, 31, 29, 23])

while True:
    humidity, temperature = Adafruit_DHT.read_retry(11, 4)

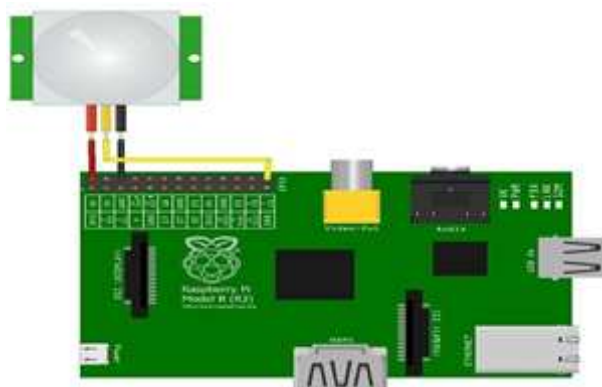
    lcd.cursor_pos = (0, 0)
    lcd.write_string("Temp: %d C" % temperature)
    lcd.cursor_pos = (1, 0)
    lcd.write_string("Humidity: %d %" % humidity)
```


Инфракрасный индукционный модуль для человеческого тела (датчик движения) (HC-SR501 infrared human body induction module)

HC-SR501 — недорогой датчик PIR, который полностью автономный, способный работать сам по себе или в сопряжении с микроконтроллером. Датчик имеет регулировку чувствительности, которая позволяет определять движение от 3 до 7 метров, а его выход можно настроить так, чтобы он оставался высоким в течение времени от 3 секунд до 5 минут. Так же, датчик имеет встроенный стабилизатор напряжения, поэтому он может питаться от постоянного напряжения от 4,5 до 20 вольт и потребляет небольшое количество тока.



Подключение



Код

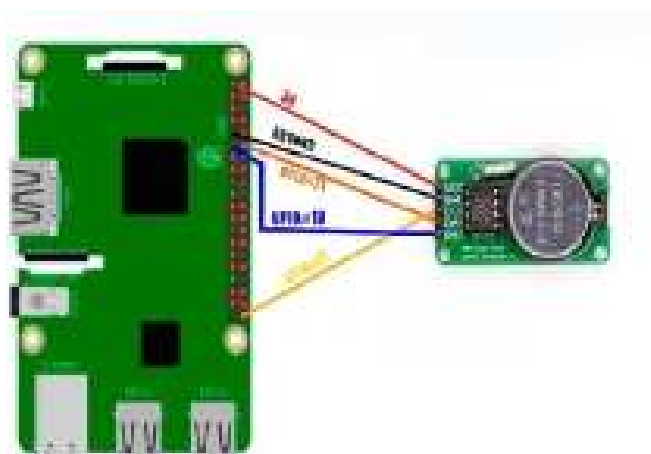
```
import RPi.GPIO as GPIO          #Import GPIO library
import time                      #Import time library
GPIO.setmode(GPIO.BOARD)        #Set GPIO pin numbering
pir = 26                         #Associate pin 26 to pir
GPIO.setup(pir, GPIO.IN)        #Set pin as GPIO in
print "Waiting for sensor to settle"
time.sleep(2)                   #Waiting 2 seconds for the sensor to initiate
print "Detecting motion"
while True:
    if GPIO.input(pir):          #Check whether pir is HIGH
        print "Motion Detected!"
        time.sleep(2)           #D1- Delay to avoid multiple detection
        time.sleep(0.1)        #While loop delay should be less than detection delay
```

Модуль часов реального времени (DS1302 real time clock module)

Модуль часов выполнен на основе чипа ds-1302. Часы позволяют считать секунды, минуты, часы, день недели, день месяца, месяц, год с учетом високосных лет до 2100 года. Есть возможность вести 12 или 24 часовой учет времени.



Подключение



Код

Создайте файл `rtc-pi.c` с кодом:

```

#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>
#include <time.h>
#include <string.h>
#include <errno.h>
#include <sys/mman.h>
#include <sys/time.h>

#define GPIO_ADD    0x20200000L // physical address of I/O peripherals on the /
#define GPIO_SEL1   1           // Offset of SEL register for GP17 & GP18 into
#define GPIO_SEL2   2           // Offset of SEL register for GP21 into GPIO ba
#define GPIO_SET    7           // Offset of PIN HIGH register into GPIO bank
#define GPIO_CLR    10          // Offset of PIN LOW register into GPIO bank
#define GPIO_INP    13          // Offset of PIN INPUT value register into GPIO
#define PAGE_SIZE   4096
#define BLOCK_SIZE  PAGE_SIZE

/* RTC Chip register definitions */
#define SEC_WRITE    0x80
#define MIN_WRITE    0x82
#define HOUR_WRITE   0x84
#define DATE_WRITE   0x86
#define MONTH_WRITE  0x88
#define YEAR_WRITE   0x8C
#define SEC_READ     0x81
#define MIN_READ     0x83
#define HOUR_READ    0x85
#define DATE_READ    0x87
#define MONTH_READ   0x89
#define YEAR_READ    0x8D

int mem_fd      = 0;
char *gpio_mmap = NULL;
char *gpio_ram  = NULL;
volatile unsigned int *gpio = NULL;

/* These 'defines' map the peripheral pin functions to our circuits DS1302 pins
/* See DS1302 datasheet REV: 110805, and Broadcom BCM2835-ARM-Peripherals.pdf (
#define IO_INPUT      *(gpio+GPIO_SEL1) &= 0xF8FFFFFFL
#define IO_OUTPUT     *(gpio+GPIO_SEL1) &= 0xF8FFFFFFL; *(gpio+GPIO_SEL1) |= 0x0
/* RPi v1:
#define SCLK_OUTPUT   *(gpio+GPIO_SEL2) &= 0xFFFFFC7L; *(gpio+GPIO_SEL2) |= 0x0
RPi v2: */
#define SCLK_OUTPUT   *(gpio+GPIO_SEL2) &= 0xFF1FFFFFFL; *(gpio+GPIO_SEL2) |= 0x0
#define CE_OUTPUT     *(gpio+GPIO_SEL1) &= 0xFF1FFFFFFL; *(gpio+GPIO_SEL1) |= 0x0
#define IO_HIGH       *(gpio+GPIO_SET) = 0x00040000L
#define IO_LOW        *(gpio+GPIO_CLR) = 0x00040000L
/* RPi v1:
#define SCLK_HIGH     *(gpio+GPIO_SET) = 0x00200000L
#define SCLK_LOW      *(gpio+GPIO_CLR) = 0x00200000L
RPi v2: */
#define SCLK_HIGH     *(gpio+GPIO_SET) = 0x08000000L
#define SCLK_LOW      *(gpio+GPIO_CLR) = 0x08000000L
#define CE_HIGH       *(gpio+GPIO_SET) = 0x00020000L
#define CE_LOW        *(gpio+GPIO_CLR) = 0x00020000L
#define IO_LEVEL      *(gpio+GPIO_INP) & 0x00040000L

void setup_io()
{

```

```

/* open /dev/mem to get access to physical ram */
if ((mem_fd = open("/dev/mem", O_RDWR|O_SYNC) ) < 0) {
    printf("can't open /dev/mem. Did you run the program with administrator ?\n");
    exit (-1);
}

/* Allocate a block of virtual RAM in our application's address space */
if ((gpio_ram = malloc(BLOCK_SIZE + (PAGE_SIZE-1))) == NULL) {
    printf("allocation error \n");
    exit (-1);
}

/* Make sure the pointer is on 4K boundary */
if ((unsigned long)gpio_ram % PAGE_SIZE)
    gpio_ram += PAGE_SIZE - ((unsigned long)gpio_ram % PAGE_SIZE);

/* Now map the physical addresses of the peripheral control registers
into our address space */
gpio_mmap = (unsigned char *)mmap(
    (caddr_t)gpio_ram,
    BLOCK_SIZE,
    PROT_READ|PROT_WRITE,
    MAP_SHARED|MAP_FIXED,
    mem_fd,
    GPIO_ADD
);

if ((long)gpio_mmap < 0) {
    printf("unable to map the memory. Did you run the program with administrator ?\n");
    exit (-1);
}

/* Always use a volatile pointer to hardware registers */
gpio = (volatile unsigned *)gpio_mmap;

/* Now we have access to the hardware registers we can start twiddling I/O pins */

/* Switch GPIO 0, 1 and 2 to output mode */
SCLK_OUTPUT;
IO_OUTPUT;
CE_OUTPUT;

/* Set the SCLK, IO and CE pins to default (low) */
SCLK_LOW;
IO_LOW;
CE_LOW;

/* Short delay to allow the I/O lines to settle. */
usleep(2);
}

unsigned char read_rtc( unsigned char add )
{
    unsigned char val;
    int lp;

    val = add;

    /* Check LSB is set */
    if ( !(add & 1 ) ) {
        printf("Incorrect read address specified - LSB must be set.\n");
        exit (-1);
    }

    /* Check address range is valid */

```

```

if ( ( add < 0x81) || (add > 0x91) ) {
    printf("Incorrect read address specified - It must be in the range 0x81.
    exit (-1);
}

CE_HIGH;

usleep(2);

for (lp=0; lp<8; lp++) {
    if (val & 1)
        IO_HIGH;
    else
        IO_LOW;
    val >>= 1;
    usleep(2);
    SCLK_HIGH;
    usleep(2);
    SCLK_LOW;
    usleep(2);
}

IO_INPUT;

for (lp=0; lp<8; lp++) {
    usleep(2);
    val >>= 1;
    if (IO_LEVEL)
        val |= 0x80;
    else
        val &= 0x7F;
    SCLK_HIGH;
    usleep(2);
    SCLK_LOW;
    usleep(2);
}

/* Set the I/O pin back to it's default, output low. */
IO_LOW;
IO_OUTPUT;

/* Set the CE pin back to it's default, low */
CE_LOW;

/* Short delay to allow the I/O lines to settle. */
usleep(2);

return val;
}

void write_rtc( unsigned char add, unsigned char val_to_write )
{
    unsigned char val;
    int lp;

    /* Check LSB is clear */
    if ( add & 1 ) {
        printf("Incorrect write address specified - LSB must be cleared.\n");
        exit (-1);
    }

    /* Check address range is valid */
    if ( ( add < 0x80) || (add > 0x90) ) {
        printf("Incorrect write address specified - It must be in the range 0x80
        exit (-1);
    }
}

```

```

}

CE_HIGH;

usleep(2);

val = add;

for (lp=0; lp<8; lp++) {
    if (val & 1)
        IO_HIGH;
    else
        IO_LOW;
    val >>= 1;
    usleep(2);
    SCLK_HIGH;
    usleep(2);
    SCLK_LOW;
    usleep(2);
}

val = val_to_write;

for (lp=0; lp<8; lp++) {
    if (val & 1)
        IO_HIGH;
    else
        IO_LOW;
    val >>= 1;
    usleep(2);
    SCLK_HIGH;
    usleep(2);
    SCLK_LOW;
    usleep(2);
}

/* Set the I/O pin back to it's default, output low. */
IO_LOW;

/* Set the CE pin back to it's default, low */
CE_LOW;

/* Short delay to allow the I/O lines to settle. */
usleep(2);
}

int main(int argc, char **argv)
{
    int lp;
    unsigned char val;
    int year,month,day,hour,minute,second;
    time_t epoch_time;
    struct tm time_requested;
    struct timeval time_setformat;

    /* Check that the program was called correctly */
    if ( argc > 2 ) {
        printf("Too many arguments specified.\nRun as:\nrtc-pi\nor\nrtc-pi CCYYMMDD\n");
        exit (-1);
    }

    /* Set up gpi pointer for direct register access */
    setup_io();

    if ( argc == 2 ) {

```

```

/* If the number of arguments are two, that means the user enter a date &
/* Read that value and write it to the RTC chip */

sscanf(argv[1], "%4d%2d%2d%2d%2d", &year, &month, &day, &hour, &minute, &second);

/* Validate that the input date and time is basically sensible */
if ( (year < 2000) || (year > 2099) || (month < 1) || (month > 12) ||
      (day < 1) || (day > 31) || (hour < 0) || (hour > 23) || (minute < 0) ||
      (minute > 59) || (second < 0) || (second > 59) ) {
    printf("Incorrect date and time specified.\nRun as:\nrtc-pi\nor\nrtc-pi\n");
    exit (-1);
}

/* Got valid input - now write it to the RTC */
/* The RTC expects the values to be written in packed BCD format */
write_rtc(SEC_WRITE, ( (second/10) << 4) | (second % 10) );
write_rtc(MIN_WRITE, ( (minute/10) << 4) | (minute % 10) );
write_rtc(HOUR_WRITE, ( (hour/10) << 4) | (hour % 10) );
write_rtc(DATE_WRITE, ( (day/10) << 4) | (day % 10) );
write_rtc(MONTH_WRITE, ( (month/10) << 4) | (month % 10) );
write_rtc(YEAR_WRITE, ( ((year-2000)/10) << 4) | (year % 10) );

/* Finally convert to it to EPOCH time, ie the number of seconds since Jan 1 1970
time_requested.tm_sec = second;
time_requested.tm_min = minute;
time_requested.tm_hour = hour;
time_requested.tm_mday = day;
time_requested.tm_mon = month-1;
time_requested.tm_year = year-1900;
time_requested.tm_wday = 0; /* not used */
time_requested.tm_yday = 0; /* not used */
time_requested.tm_isdst = -1; /* determine daylight saving time from the

epoch_time = mktime(&time_requested);

/* Now set the clock to this time */
time_setformat.tv_sec = epoch_time;
time_setformat.tv_usec = 0;

printf("Set UNIX timestamp to RTC: %lld\n", (long long) epoch_time );

lp = settimeofday(&time_setformat, NULL);

/* Check that the change was successful */
if ( lp < 0 ) {
    printf("Unable to change the system time. Did you run the program as root?");
    printf("The operation returned the error message \"%s\"\n", strerror(errno));
    exit (-1);
}
} else {
/* The program was called without a date specified; therefore read the date from
/* the RTC chip and set the system time to this */
second = read_rtc(SEC_READ);
minute = read_rtc(MIN_READ);
hour = read_rtc(HOUR_READ);
day = read_rtc(DATE_READ);
month = read_rtc(MONTH_READ);
year = read_rtc(YEAR_READ);

/* Finally convert to it to EPOCH time, ie the number of seconds since Jan 1 1970
/* Bearing in mind that the format of the time values in the RTC is packed BCD

time_requested.tm_sec = (((second & 0x70) >> 4) * 10) + (second & 0x0F);
time_requested.tm_min = (((minute & 0x70) >> 4) * 10) + (minute & 0x0F);
time_requested.tm_hour = (((hour & 0x30) >> 4) * 10) + (hour & 0x0F);

```



```
time_requested.tm_mday = (((day & 0x30) >> 4) * 10) + (day & 0x0F);
time_requested.tm_mon = (((month & 0x10) >> 4) * 10) + (month & 0x0F) - 1;
time_requested.tm_year = (((year & 0xF0) >> 4) * 10) + (year & 0x0F) + 2000;
time_requested.tm_wday = 0; /* not used */
time_requested.tm_yday = 0; /* not used */
time_requested.tm_isdst = -1; /* determine daylight saving time from the

epoch_time = mktime(&time_requested);

/* Now set the clock to this time */
time_setformat.tv_sec = epoch_time;
time_setformat.tv_usec = 0;

printf("Read UNIX timestamp from RTC: %lld\n", (long long) epoch_time );

lp = settimeofday(&time_setformat, NULL);

/* Check that the change was successful */
if ( lp < 0 ) {
    printf("Unable to change the system time. Did you run the program as root?");
    printf("The operation returned the error message \"%s\"\n", strerror(errno));
    exit (-1);
}

return 0;
}
```

Запустите код

```
cc rtc-pi.c -o rtc-pi
```

Установите время:

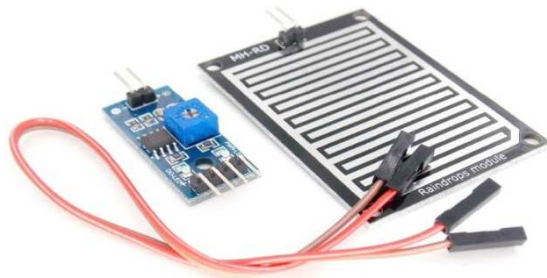
```
sudo rtc-pi YYYYMMDDhhmmss
```

Читайте время с модуля:

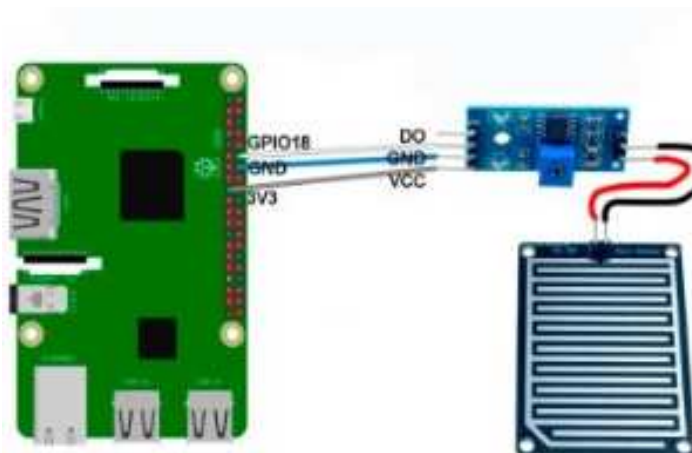
```
sudo rtc-pi
```

Датчик дождя (The rain sensor module)

Датчик дождя позволяет определять наличие или отсутствие дождя. Он может быть полезным для отслеживания погодных условий и определения вероятности осадков.



Подключение



Код

```
# raindrop sensor D0 connected to GPIO18
# HIGH = no rain, LOW = rain detected
# Buzzer on GPIO13
from time import sleep
from gpiozero import Buzzer, InputDevice

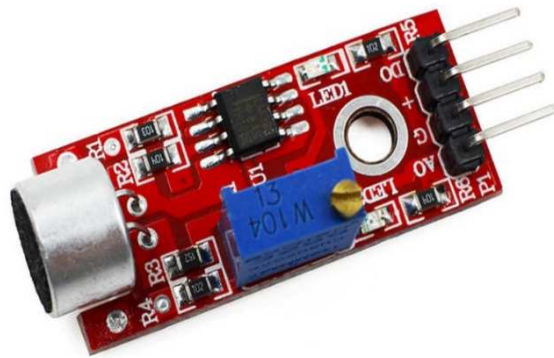
buzz = Buzzer(13)
no_rain = InputDevice(18)

def buzz_now(iterations):
    for x in range(iterations):
        buzz.on()
        sleep(0.1)
        buzz.off()
        sleep(0.1)

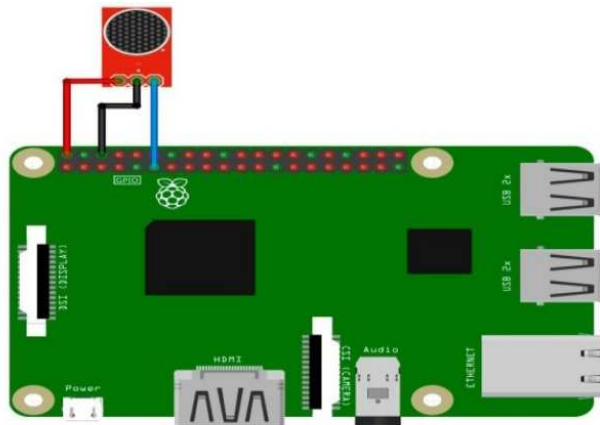
while True:
    if not no_rain.is_active:
        print("It's raining!")
        buzz_now(5)
        # insert your other code or functions here
        # e.g. SMS, email, take a photo etc.
        sleep(1)
```

Датчик звука (Sound sensor module)

Звуковой датчик функционирует как микрофон, который используется для приема звуковых волн. Датчик имеет встроенный конденсаторный электретный микрофон, чувствительный к звуку. Звуковые волны заставляют электретную пленку в микрофоне вибрировать, что приводит к изменению емкости и небольшому напряжению, соответствующему изменению. Затем это напряжение преобразуется в напряжение 0-5 В, которое принимается приемником данных после аналого-цифрового преобразования и передается на микроконтроллер.



Подключение



Код

```
#!/usr/bin/python
import RPi.GPIO as GPIO
import time

#GPIO SETUP
channel = 17
GPIO.setmode(GPIO.BCM)
GPIO.setup(channel, GPIO.IN)

def callback(channel):
    if GPIO.input(channel):
        print "Sound Detected!"
    else:
        print "Sound Detected!"

GPIO.add_event_detect(channel, GPIO.BOTH, bouncetime=300) # let us know when 1
GPIO.add_event_callback(channel, callback) # assign function to GPIO PIN, Run

# infinite loop
while True:
    time.sleep(1)
```

Работа с ультразвуковым дальномером

Ультразвуковой дальномер («сонар») — это датчик расстояния, принцип действия которого основан на измерении времени распространения звуковой волны (с частотой около 40 кГц) до препятствия и обратно. Сонар может измерять расстояние до 1,5–3 м с точностью до нескольких сантиметров.

Дальномер HC-SR04

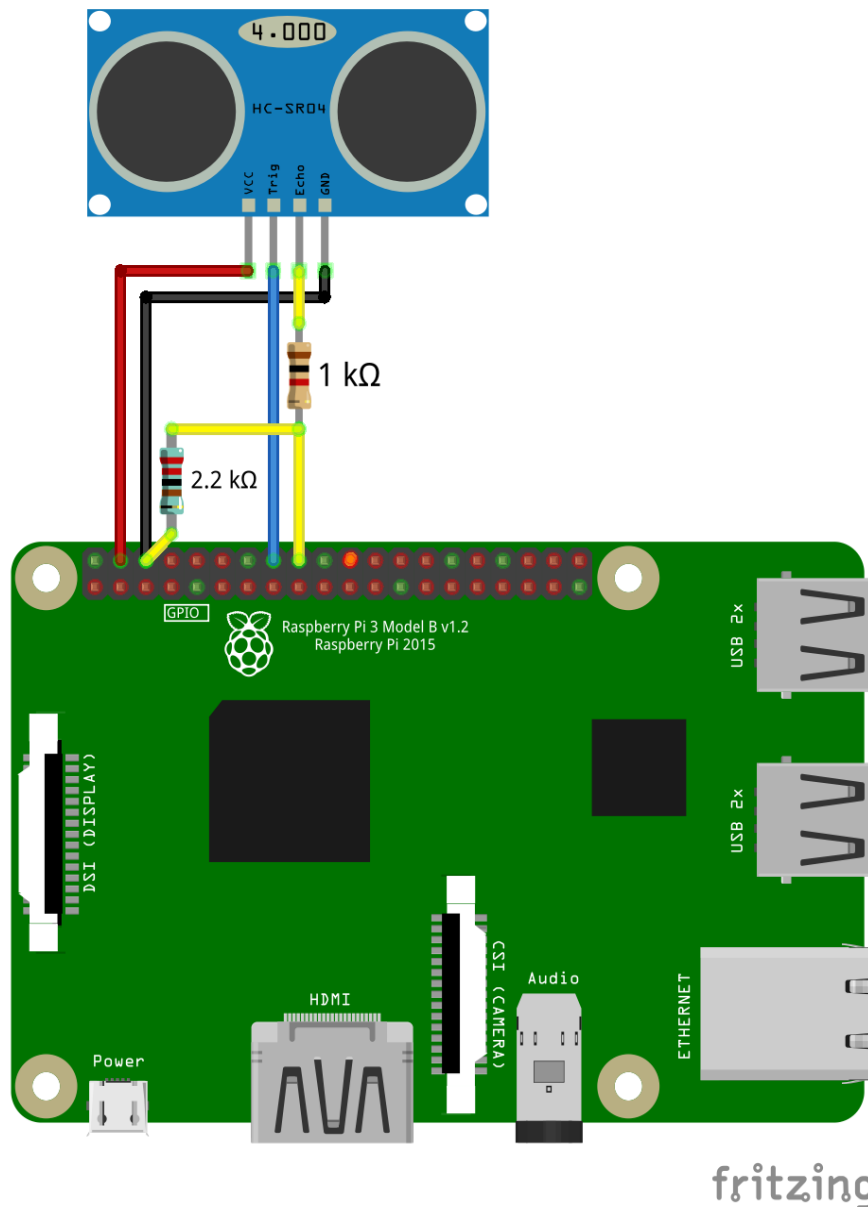


Установка

Дальномер закрепляется к корпусу с помощью двухстороннего скотча. Для получения приемлемых результатов необходимо использование виброразвязки. В качестве виброразвязки можно использовать кусок поролона.

Подключение

Подключите HC-SR04 к Raspberry Pi согласно схеме подключения. Используйте резисторы на 1,0 и 2,2 кОм и любые свободные GPIO-пины, например 23 и 24:



Вместо резистора на 2,2 кОм можно использовать два резистора на 1 кОм, соединенные последовательно.

На Raspberry Pi есть несколько взаимозаменяемых пинов **GND** и **VCC 5V**. Используйте [распиновку](#), чтобы найти их.

Чтение данных

Чтобы считать данные с датчика HC-SR04, используется библиотека для работы с GPIO – `pigpio`. Эта библиотека предустановлена на [образе](#). Для работы с `pigpio` необходимо запустить соответствующий демон:

```
sudo systemctl start pigpiod.service
```

Вы также можете включить автоматический запуск `pigpiod` при старте системы:

```
sudo systemctl enable pigpiod.service
```

Таким образом становится возможным взаимодействие с демоном `pigpiod` из языка Python:

```
import pigpio
pi = pigpio.pi()
```

См. подробное описание Python API в [документации pigpio](#).

Пример кода для чтения данных с HC-SR04:

```
import time
import threading
import pigpio

TRIG = 23 # пин, к которому подключен контакт Trig датчика
ECHO = 24 # пин, к которому подключен контакт Echo датчика

pi = pigpio.pi()
done = threading.Event()

def rise(gpio, level, tick):
    global high
    high = tick

def fall(gpio, level, tick):
    global low
    low = tick - high
    done.set()

def read_distance():
    global low
    done.clear()
    pi.gpio_trigger(TRIG, 50, 1)
    if done.wait(timeout=5):
        return low / 58.0 / 100.0

pi.set_mode(TRIG, pigpio.OUTPUT)
pi.set_mode(ECHO, pigpio.INPUT)
pi.callback(ECHO, pigpio.RISING_EDGE, rise)
pi.callback(ECHO, pigpio.FALLING_EDGE, fall)

while True:
    # Читаем дистанцию:
    print(read_distance())
```

Фильтрация данных

Для фильтрации (сглаживания) данных и удаления выбросов может быть использован [фильтр Калмана](#) или более простой [медианный фильтр](#).

Пример реализации медианной фильтрации:


```
import collections
import numpy

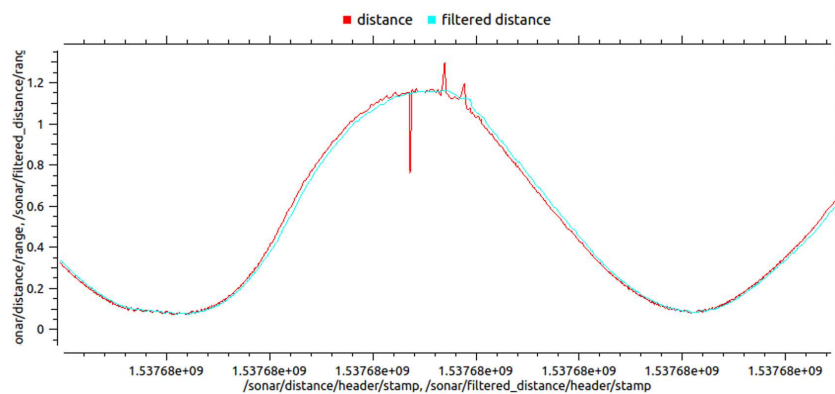
# ...

history = collections.deque(maxlen=10) # 10 - количество сэмплов для усреднения

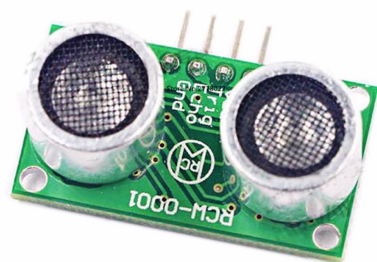
def read_distance_filtered():
    history.append(read_distance())
    return numpy.median(history)

while True:
    print(read_distance_filtered())
```

Пример графиков исходных и отфильтрованных данных:



Дальномер RCW-0001



Ультразвуковой дальномер RCW-0001 совместим с дальномером HC-SR04. Используйте инструкцию выше для подключения и работы с ним.

Полет

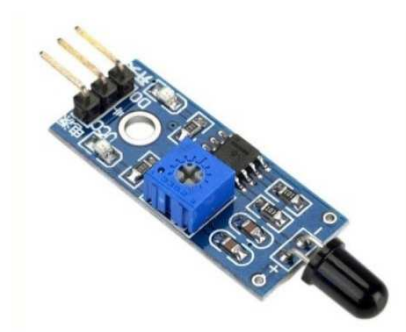
Пример полетной программы с использованием [simple_offboard](#), которая заставляет коптер лететь вперед, пока подключенный ультразвуковой дальномер не обнаружит препятствие:

```
set_velocity(vx=0.5, frame_id='body', auto_arm=True) # полет вперед со скоростью 0.5 м/с

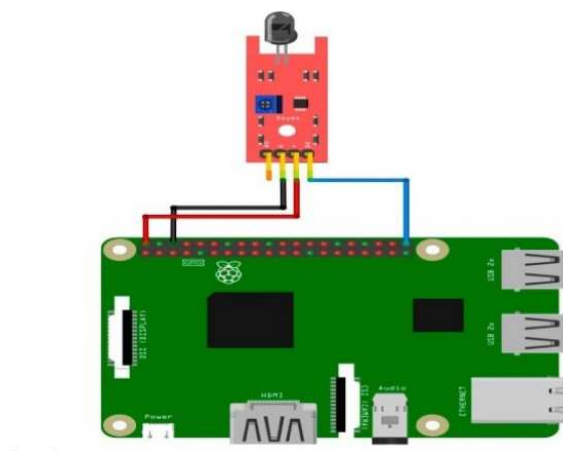
while True:
    if read_distance_filtered() < 1:
        # если препятствие ближе, чем в 1 м, зависаем в точке
        set_position(x=0, y=0, z=0, frame_id='body')
        rospy.sleep(0.1)
```

Датчик огня/пламени (The flame sensor module)

Модуль датчика огня (Flame sensor module). Модуль реагирует на открытое пламя. Воспринимающим элементом датчика служит фотодиод получающий инфракрасное излучение.



Подключение



Код

Датчик температуры и влажности

```
#!/usr/bin/python
import RPi.GPIO as GPIO
import time

#GPIO SETUP
channel = 21
GPIO.setmode(GPIO.BCM)
GPIO.setup(channel, GPIO.IN)

def callback(channel):
    print("flame detected")

GPIO.add_event_detect(channel, GPIO.BOTH, bouncetime=300) # let us know when 1
GPIO.add_event_callback(channel, callback) # assign function to GPIO PIN, Run

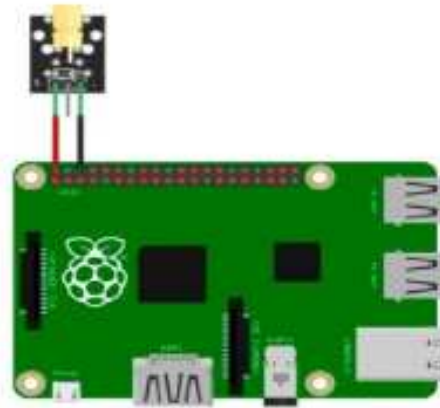
# infinite loop
while True:
    time.sleep(1)
```

Датчик лазерный (KY-008 laser head sensor module)

Лазерный датчик используется в качестве лазерного целеуказателя, сигнализации, передатчика азбуки морзе и т.д. Представляет собой плату простой конструкции с установленным на ней небольшим лазерным диодом в коллиматоре с фиксированным фокусным расстоянием и минимальным набором компонентов для управления током диода. Модуль позволяет реализовывать управление лазером с помощью микроконтроллера.



Подключение

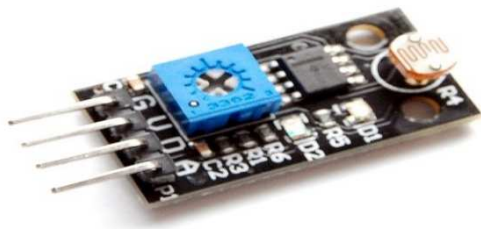


Код

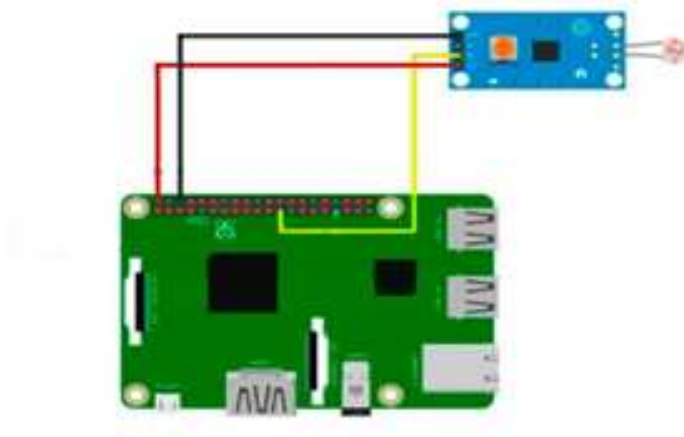
Этот модуль не требует кода для работы.

Светочувствительный датчик сопротивления (Photosensitive resistance sensor module)

Светочувствительный датчик чувствителен к окружающему свету и очень подходит для определения яркости окружающего освещения. Представляет из себя фоторезистор, или светозависимый резистор (LDR), или фотозаэлемент, сопротивление которого уменьшается при увеличении интенсивности падающего света; другими словами, он проявляет фотопроводимость.



Подключение

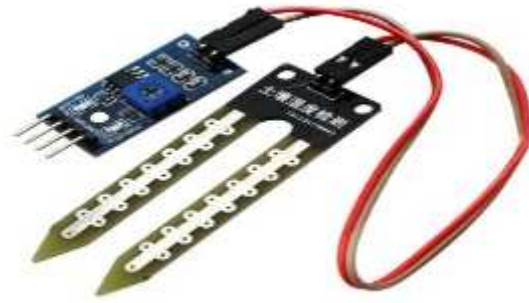


Код

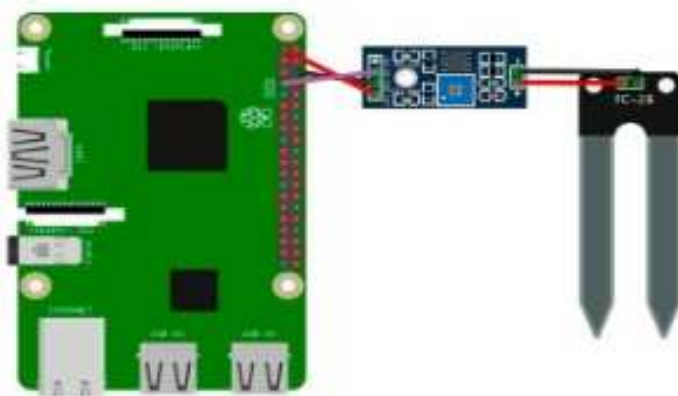
```
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(4, GPIO.IN)
for i in range(0,5):
    print GPIO.input(4)
```

Датчик влажности почвы (The YL-69 soil moisture sensor)

Датчики влажности почвы измеряют количество воды в почве для поддержания постоянных и идеальных почвенных условий для растений.



Подключение



Код

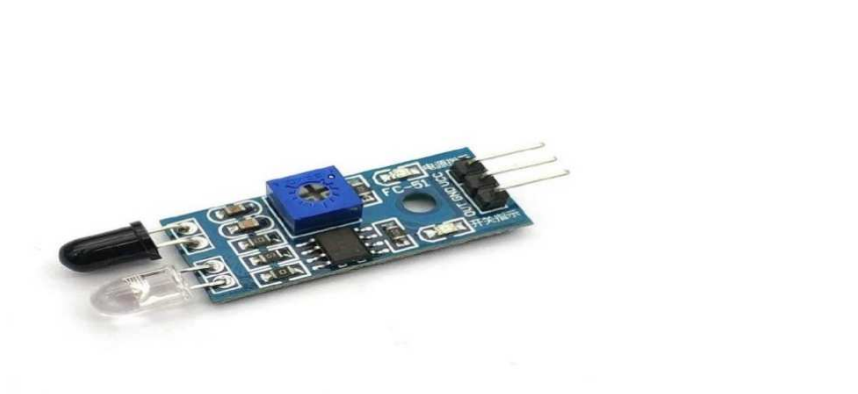
```
from gpiozero import DigitalInputDevice
import time

d0_input = DigitalInputDevice(4)

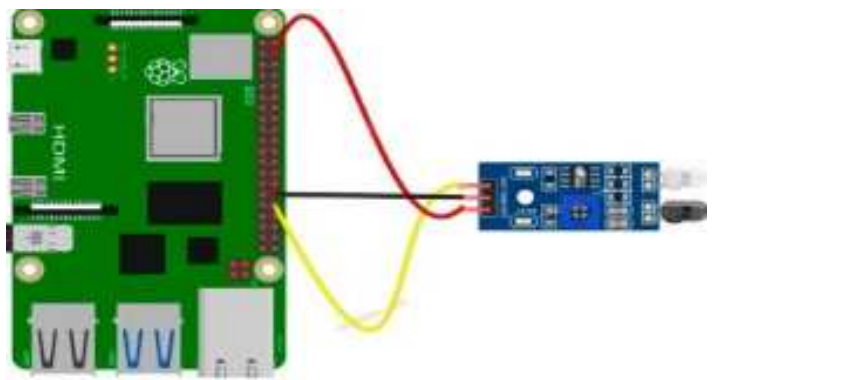
while True:
    if (not d0_input.value):
        print('Moisture threshold reached!!!')
    else:
        print('You need to water your plant!')
        time.sleep(2)
```

Датчик обхода препятствий (Obstacle avoidance sensor)

Датчик обхода препятствий (инфракрасный датчик) представляет способен определять присутствие препятствий перед ним с помощью инфракрасных сигналов. Дальность обнаружения составляет от 2 см до 30 см, и ее можно регулировать с помощью встроенного потенциометра.



Подключение



Код

Убедитесь, что библиотека **RPi.GPIO** установлена:

```
sudo apt-get update  
sudo apt-get install python3-rpi.gpio
```



```
import RPi.GPIO as GPIO
import time

# Set the GPIO mode to BCM (Broadcom SOC channel numbering)
GPIO.setmode(GPIO.BCM)

# Set the pin number connected to the ir obstacle avoidance sensor
SENSOR_PIN = 12

# Set the GPIO pin as an input
GPIO.setup(SENSOR_PIN, GPIO.IN)

# Variable to track the ir obstacle avoidance sensor state
prev_obstacle_state = GPIO.HIGH # Assuming no obstacle initially

try:
    while True:
        obstacle_state = GPIO.input(SENSOR_PIN)

        if obstacle_state != prev_obstacle_state:
            if obstacle_state == GPIO.LOW:
                # obstacle is detected
                print("An obstacle is detected")
            else:
                # An obstacle is removed
                print("An obstacle is removed")

        prev_obstacle_state = obstacle_state
        time.sleep(0.1) # A small delay to debounce the input

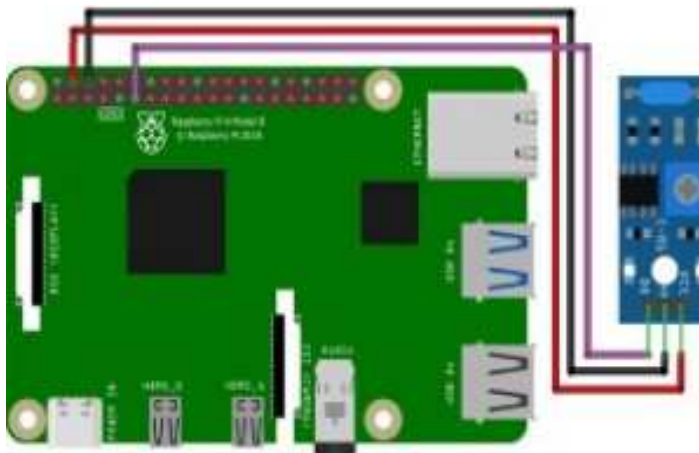
except KeyboardInterrupt:
    # Clean up the GPIO settings on program exit
    GPIO.cleanup()
```

Датчик вибрации (Vibration sensor module)

Датчик вибрации использовать для обнаружения любого вида вибрации и движения. Датчик может обнаруживать любые вибрации и движение в определенном диапазоне.



Подключение



Код

```
#!/usr/bin/python
import RPi.GPIO as GPIO
import time

#GPIO SETUP
channel = 17
GPIO.setmode(GPIO.BCM)
GPIO.setup(channel, GPIO.IN)

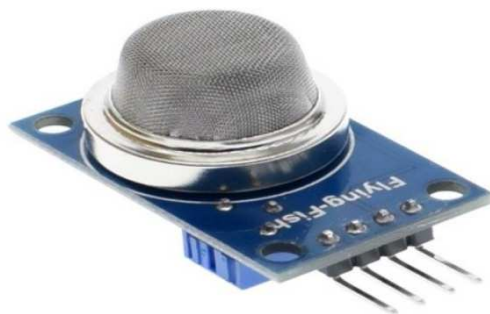
def callback(channel):
    if GPIO.input(channel):
        print "Movement Detected!"
    else:
        print "Movement Detected!"

GPIO.add_event_detect(channel, GPIO.BOTH, bouncetime=300) # let us know when 1
GPIO.add_event_callback(channel, callback) # assign function to GPIO PIN, Run

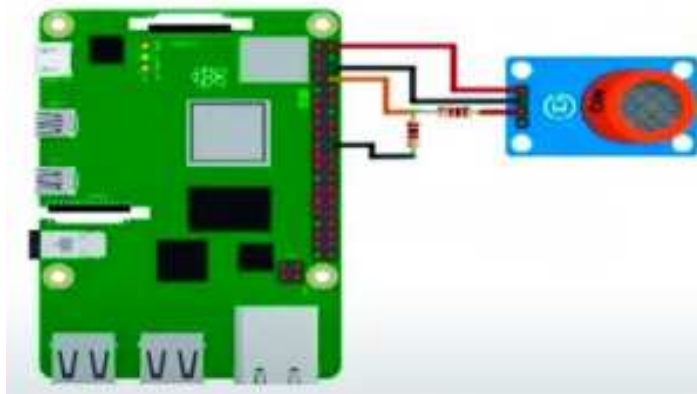
# infinite loop
while True:
    time.sleep(1)
```

Датчик газа (MQ-2 gas sensor module)

Датчик газа MQ-2 - это недорогой электронный датчик, используемый для определения концентрации газов в окружающей среде. Газовый датчик MQ-2 используется в системах безопасности для обнаружения потенциально опасных газов или легковоспламеняющегося дыма.



Подключение



Код

```
from mq2 import MQ2
import utime

pin=26

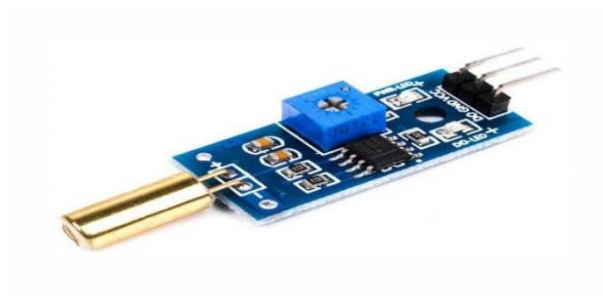
sensor = MQ2(pinData = pin, baseVoltage = 3.3)

print("Calibrating")
sensor.calibrate()
print("Calibration completed")
print("Base resistance:{0}".format(sensor._ro))

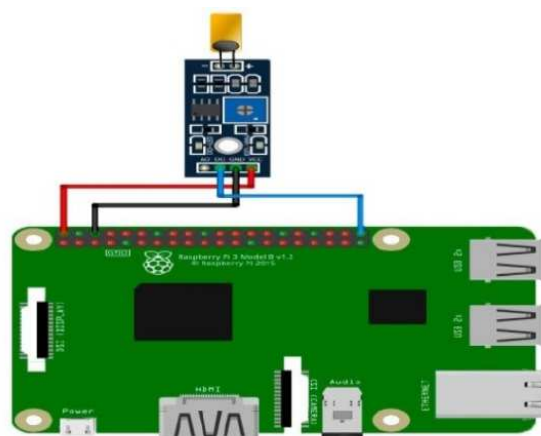
while True:
    print("Smoke: {:.1f}".format(sensor.readSmoke())+" - ", end="")
    print("LPG: {:.1f}".format(sensor.readLPG())+" - ", end="")
    print("Methane: {:.1f}".format(sensor.readMethane())+" - ", end="")
    print("Hydrogen: {:.1f}".format(sensor.readHydrogen()))
    utime.sleep(0.5)
```

Датчик наклона (The tilt sensor module)

Определяет, наклонен ли объект.



Подключение



Код

Датчик температуры и влажности

```
#!/usr/bin/env python
import RPi.GPIO as GPIO

channel = 21
GPIO.setmode(GPIO.BCM)
GPIO.setup(channel, GPIO.IN, pull_up_down=GPIO.PUD_UP)

def alert(ev=None):
    print("Tilt Detected")

def loop():
    GPIO.add_event_detect(channel, GPIO.FALLING, callback=alert, bouncetime=100)
    while True:
        pass

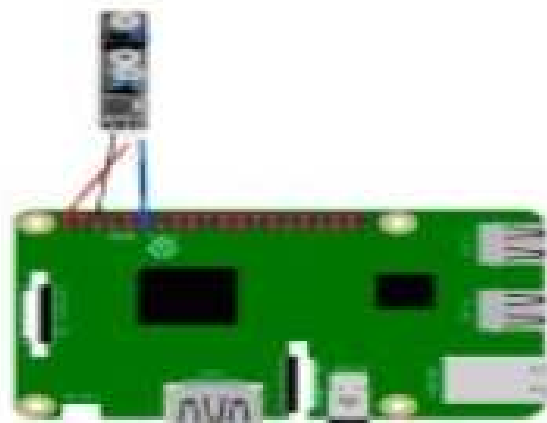
if __name__ == '__main__':
    try:
        loop()
    except KeyboardInterrupt:
        GPIO.cleanup()
```

Модуль трассировки пути

Модуль трассировки пути – переключатель отражения ИК-света, полезный для обхода препятствий или следования по прямой. Для обхода препятствий вы можете разместить этот модуль перед отправляющими / принимающими диодами, что приведет к вытягиванию выходного штыря на расстояние 1 см.



Подключение



Код


```
#!/usr/bin/env python
import RPi.GPIO as GPIO

IRTrackingPin = 11
OutLedPin = 12

def setup():
    GPIO.setmode(GPIO.BOARD) # Set the GPIO pins as numbering
    GPIO.setup(OutLedPin, GPIO.OUT) # Set the OutLedPin's mode is output
    GPIO.setup(IRTrackingPin, GPIO.IN, pull_up_down=GPIO.PUD_UP)
    GPIO.output(OutLedPin, GPIO.HIGH) # Set the OutLedPin high(+3.3V) to off led

def loop():
    while True:
        if GPIO.input(IRTrackingPin) == GPIO.LOW:

            print '14CORE | IR Tracking Test Code'
            print '-----'
            print 'The sensor detects white color line'

            GPIO.output(OutLedPin, GPIO.LOW) # Set the OutLedPin turn HIGH/ON
            else:

            print '14CORE | IR Tracking Test Code'
            print '-----'
            print 'The sensor detects black color line'
            GPIO.output(OutLedPin, GPIO.HIGH) # Set the OutLedPin turn LOW/OFF

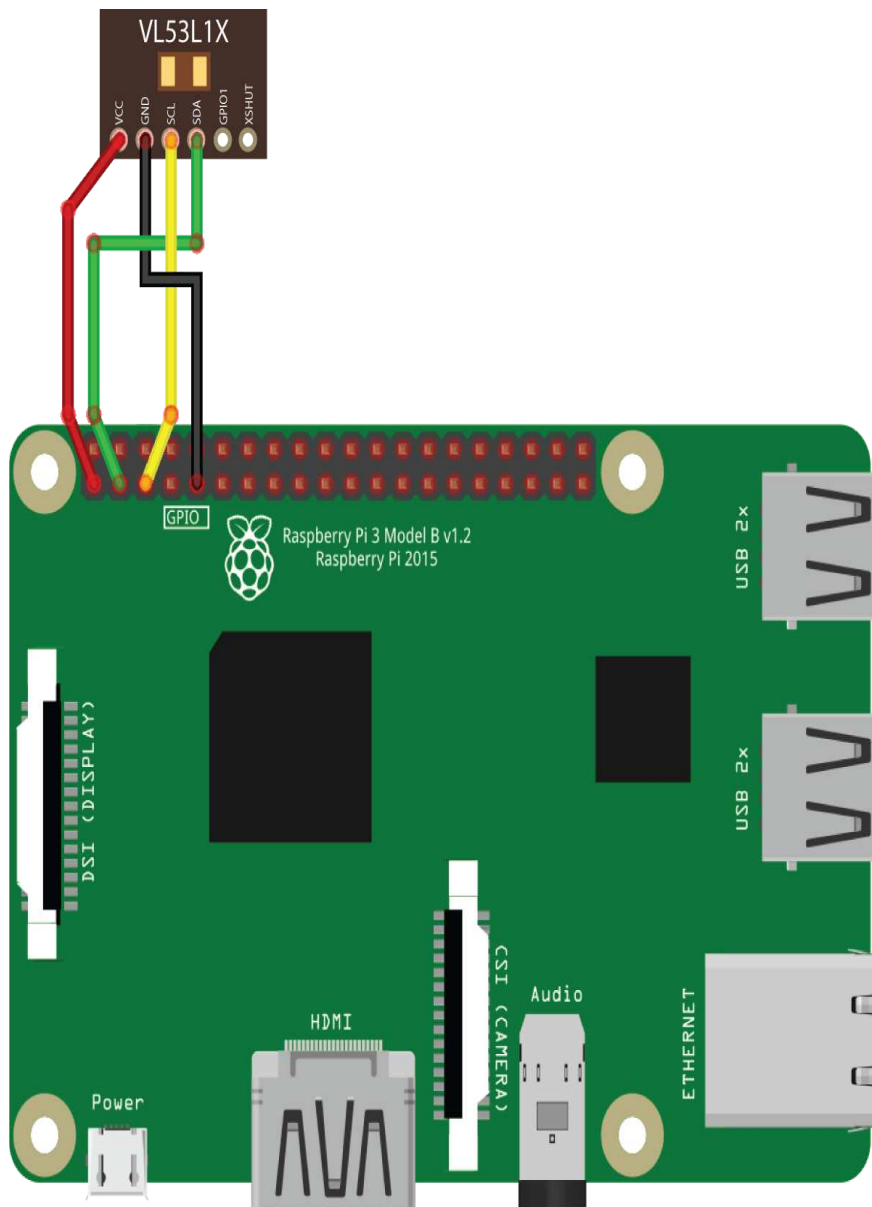
def destroy():
    GPIO.output(OutLedPin, GPIO.HIGH) # Set the OutLedPin turn HIGH
    GPIO.cleanup() # Release resource

if __name__ == '__main__': # The Program will start from here
    setup()
    try:
        loop()
    except KeyboardInterrupt: # When control c is pressed child program destroy() v
        destroy()
```

Лазерный дальномер (VL53L1X laser ranging sensor module)

Подключение

Подключите дальномер по интерфейсу I²C к пинам 3V, GND, SCL и SDA:



Если обозначенный пин GND занят, можно использовать другой свободный, используя [распиновку](#).

По интерфейсу I²C возможно подключать несколько периферийных устройств одновременно. Используйте для этого параллельное подключение.

Включение

Подключитесь по SSH и отредактируйте файл

`~/catkin_ws/src/drone/drone/launch/drone.launch` так, чтобы драйвер VL53L1X был включен:

```
<arg name="rangefinder_vl53l1x" default="true"/>
```

Для просмотра данных из топика используйте команду:

```
rostopic echo /rangefinder/range
```

Настройки

При использовании EKF2 (`SYS_MC_EST_GROUP = ekf2`):

- `EKF2_HGT_MODE = 2` (Range sensor) – при полете над горизонтальным полом;
- `EKF2_RNG_AID = 1` (Range aid enabled) – в остальных случаях.

При использовании LPE (`SYS_MC_EST_GROUP = local_position_estimator, attitude_estimator_q`):

- В параметре `LPE_FUSION` включен флажок "pub agl as lpos down" – при полете над горизонтальным полом.

Получение данных из Python

Для получения данных из топика создайте подписчика:

```
import rospy
from sensor_msgs.msg import Range

rospy.init_node('flight')

def range_callback(msg):
    # Обработка новых данных с датчика
    print('Rangefinder distance:', msg.range)

rospy.Subscriber('rangefinder/range', Range, range_callback)

rospy.spin() # дальнейший код программы
```

Также существует возможность однократного получения данных с датчика:

```
from sensor_msgs.msg import Range

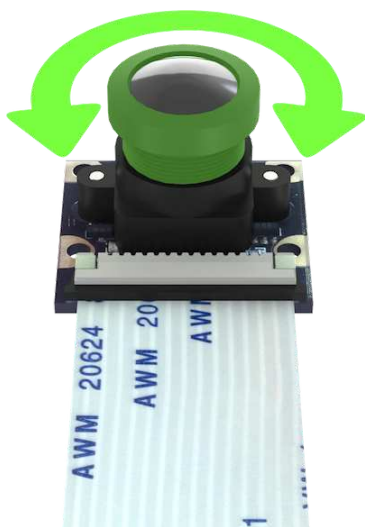
dist = rospy.wait_for_message('rangefinder/range', Range).range
```

Настройка камеры

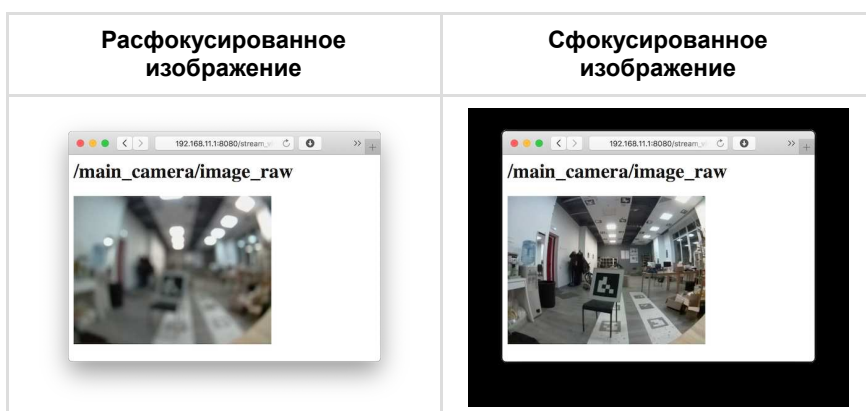
Для корректной работы всех функций, связанных с компьютерным зрением (в том числе [полета по ArUco-маркерам](#)) необходимо сфокусировать основную камеру, а также выставить ее расположение и ориентацию. Улучшить качество работы также может опциональная калибровка камеры.

Настройка фокуса камеры

Для успешного осуществления полетов с использованием камеры, необходимо настроить фокус камеры.



1. Откройте трансляцию изображения с камеры используя [web_video_server](#).
2. С помощью вращения объектива камеры добейтесь максимальной резкости деталей (предпочтительно на расстоянии предполагаемой высоты полета – 2–3 м).



Настройка расположения камеры

Расположение и ориентация камеры [задается в файле](#)

```
~/catkin_ws/src/drone/drone/launch/main_camera.launch :
```

```
<arg name="direction_z" default="down"/> <!-- direction the camera points: down  
<arg name="direction_y" default="backward"/> <!-- direction the camera cable points: backward
```

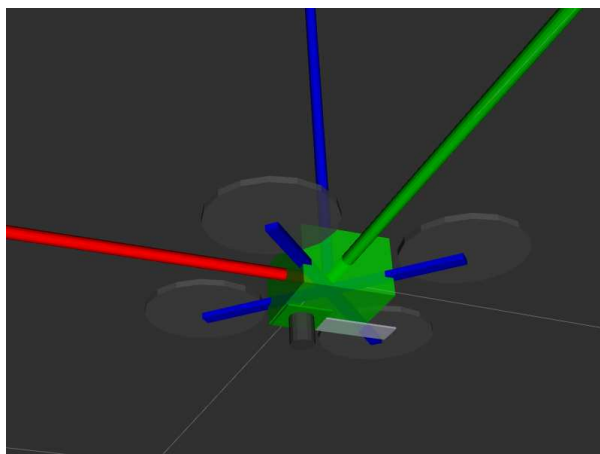
Для того, чтобы задать ориентацию, необходимо установить:

- направление обзора камеры `direction_z` : вниз (`down`) или вверх (`up`);
- направление, в которое указывает шлейф камеры `direction_y` : назад (`backward`) или вперед (`forward`).

Примеры

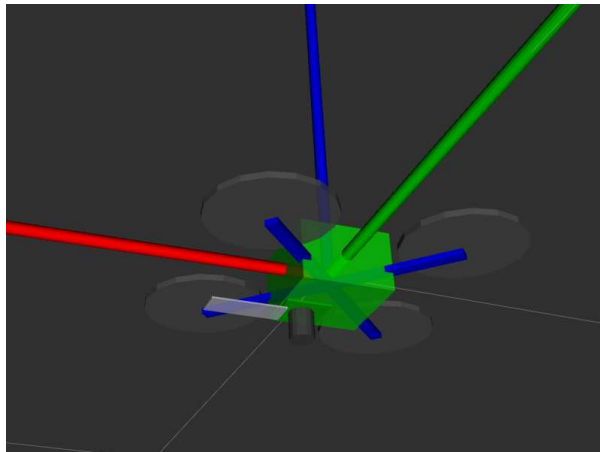
Камера направлена вниз, шлейф назад

```
<arg name="direction_z" default="down"/>  
<arg name="direction_y" default="backward"/>
```



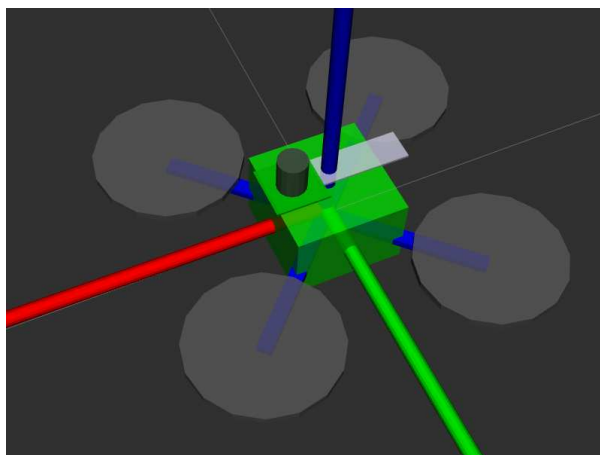
Камера направлена вниз, шлейф вперед

```
<arg name="direction_z" default="down"/>  
<arg name="direction_y" default="forward"/>
```



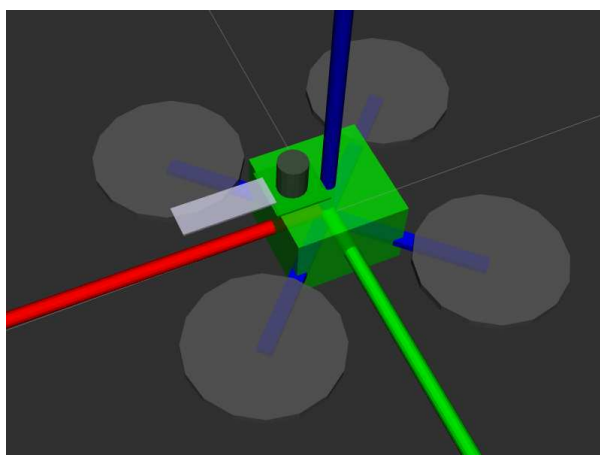
Камера направлена вверх, шлейф назад

```
<arg name="direction_z" default="up"/>  
<arg name="direction_y" default="backward"/>
```



Камера направлена вверх, шлейф вперёд

```
<arg name="direction_z" default="up"/>  
<arg name="direction_y" default="forward"/>
```



Утилита `selfcheck.py` выдает словесное описание установленной в данный момент ориентации основной камеры.

Произвольное расположение камеры

Также возможны произвольное расположение и ориентация камеры. Для этого раскомментируйте запуск ноды, подписанной как `Template for custom camera orientation` :

```
<!-- Template for custom camera orientation -->
<!-- Camera position and orientation are represented by base_link -> main_camera
<!-- static_transform_publisher arguments: x y z yaw pitch roll frame_id child_
<node pkg="tf2_ros" type="static_transform_publisher" name="main_camera_frame"
```

Эта строка задает статическую трансформацию между фреймом `base_link` (соответствует корпусу полетного контроллера) и камерой (`main_camera_optical`) в формате:

```
сдвиг_x сдвиг_y сдвиг_z угол_рысканье угол_тангаж угол_крен
```

Фрейм камеры задается таким образом, что:

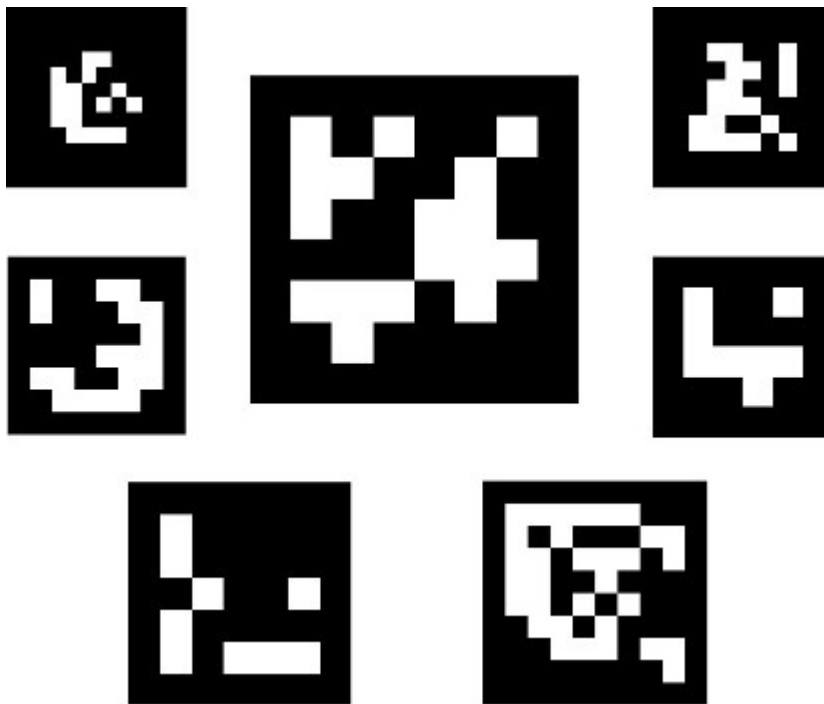
- **x** указывает направо на изображении;
- **y** указывает вниз на изображении;
- **z** указывает от плоскости матрицы камеры.

Калибровка

Для улучшения качества работы алгоритмов также рекомендуется произвести калибровку камеры, процесс которой описан в [отдельной статье](#).

ArUco-маркеры

ArUco-маркеры — это популярная технология для позиционирования робототехнических систем с использованием компьютерного зрения.



При печати визуальных маркеров необходимо использовать максимально матовую бумагу. Глянцевая бумага будет бликовать на свету, сильно ухудшая качество распознавания.

Для быстрого генерирования маркеров для печати можно использовать онлайн-инструмент: <http://chev.me/arucogen/>.

На образе для RPi предустановлен пакет `aruco_pose`, предназначенный для работы с ArUco-маркерами.

Режимы работы

Образ имеет несколько предустановленных режимов работы с ArUco-маркерами:

- [распознавание и навигация по отдельным маркерам](#);
- [распознавание и навигация по картам маркеров](#).

Распознавание ArUco-маркеров

Для распознавания маркеров модуль камеры должен быть корректно подключен и [сконфигурирован](#).

Модуль `aruco_detect` распознает ArUco-маркеры и публикует их позиции в ROS-топики и в `TF`.

Эта функция полезна для применения совместно с какой-либо системой позиционирования для дрона, такой как [GPS](#), визуальная одометрия, ультразвуковое ([Marvelmind](#)) или UWB-позиционирование ([Pozyx](#)).

Также возможно применение совместно с [навигацией по карте маркеров](#).

Настройка

Аргумент `aruco` в файле `~/catkin_ws/src/drone/drone/launch/drone.launch` должен быть в значении `true` :

```
<arg name="aruco" default="true"/>
```

Для включения распознавания маркеров аргумент `aruco_detect` в файле `~/catkin_ws/src/drone/drone/launch/aruco.launch` должен быть в значении `true` :

```
<arg name="aruco_detect" default="true"/>
```

Для правильной работы в этом же файле также должны быть выставлены аргументы:

```
<arg name="placement" default="floor"/> <!-- расположение маркеров, см. далее .
<arg name="length" default="0.33"/> <!-- размер маркеров в метрах (не вклю
```

Значение аргумента `placement` следует выставлять следующим образом:

- если все маркеры наклеены на полу (земле), выставить значение `floor` ;
- если все маркеры наклеены на потолке, выставить значение `ceiling` ;
- в противном случае удалить строку с параметром.

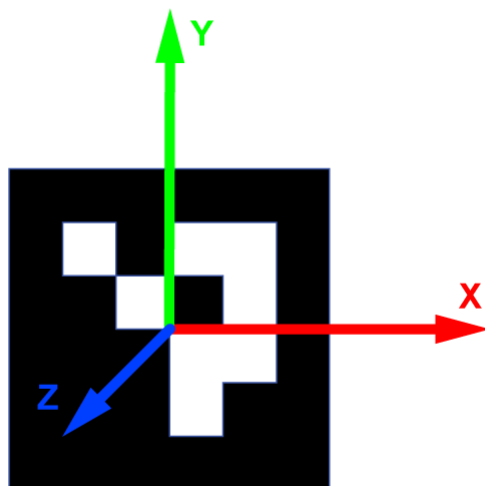
Если некоторые маркеры имеют размер, отличный значения `length` , их размер может быть переопределен с помощью параметра `length_override` ноды `aruco_detect` :

```
<param name="length_override/3" value="0.1"/> <!-- маркер с id 3 имеет разме
<param name="length_override/17" value="0.25"/> <!-- маркер с id 17 имеет разм
```

Система координат

С маркером связана следующая система координат:

- ось **x** указывает на правую сторону маркера;
- ось **y** указывает вверх маркера;
- ось **z** указывает от плоскости маркера.

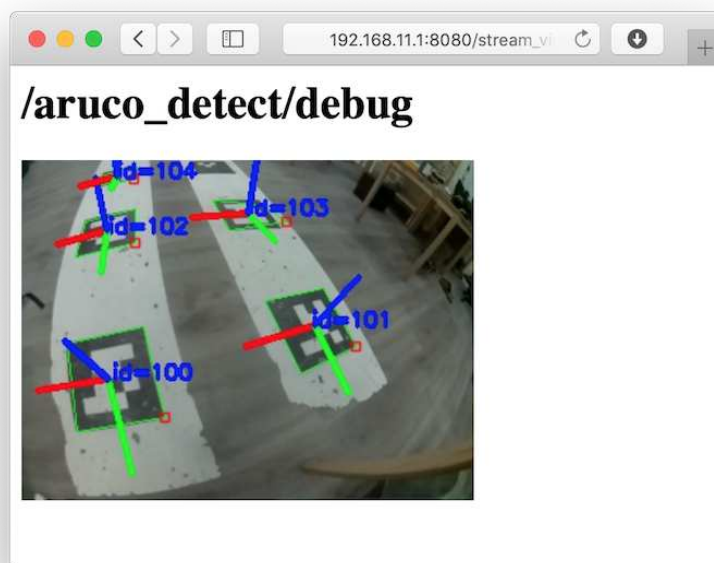


Работа с распознанными маркерами

Наглядно распознанные маркеры можно видеть в топике `aruco_detect/debug`.

Посмотреть его можно с помощью `web_video_server` по ссылке

http://192.168.11.1:8080/snapshot?topic=/aruco_pose/debug:



Распознанные маркеры и их позиции публикуются в топик

`aruco_detect/markers` . Чтение топика из Bash:

```
rostopic echo /aruco_detect/markers
```

Навигация по маркерам

С использованием модуля `simple_offboard` можно осуществлять навигацию по маркерам используя соответствующие TF-фреймы.

Полет в точку над маркером 5 на высоту 1 метр:

```
navigate(frame_id='aruco_5', x=0, y=0, z=1)
```

Полет в точку на метр левее маркера 7 на высоте 2 метра:

```
navigate(frame_id='aruco_7', x=-1, y=0, z=2)
```

Если необходимый маркер не появится в поле зрения в течение полусекунды, дрон продолжит выполнять предыдущую команду.

Подобные значения `frame_id` можно использовать и в других сервисах, например `get_telemetry` . Получение расположения дрона относительно маркера 3:

```
telem = get_telemetry(frame_id='aruco_3')
```

Если необходимый маркер не появится в поле зрения в течение полусекунды, в полях `telem.x` , `telem.y` , `telem.z` , `telem.yaw` будет значение `NaN` .

Работа с результатом распознавания из Python

Чтение топика `aruco_detect/markers` из Python:

```
import rospy
from aruco_pose.msg import MarkerArray
rospy.init_node('my_node')

# ...

def markers_callback(msg):
    print('Detected markers:'):
    for marker in msg.markers:
        print('Marker: %s' % marker)

# Подписываемся. При получении сообщения в топик aruco_detect/markers будет вы
rospy.Subscriber('aruco_detect/markers', MarkerArray, markers_callback)

# ...

rospy.spin()
```

Сообщения будут содержать ID маркера, его угловые точки на изображении и его позицию (относительно камеры).

См. далее: [навигация по картам маркеров](#).

Навигация по картам ArUco-маркеров

Для распознавания маркеров модуль камеры должен быть корректно подключен и [сконфигурирован](#).

Модуль `aruco_map` распознает карты ArUco-маркеров, как единое целое. Также возможна навигация по картам ArUco-маркеров с использованием механизма Vision Position Estimate (VPE).

Конфигурирование

Аргумент `aruco` в файле `~/catkin_ws/src/drone/drone/launch/drone.launch` должен быть в значении `true` :

```
<arg name="aruco" default="true"/>
```

Для включения распознавания карт маркеров аргументы `aruco_map` и `aruco_detect` в файле `~/catkin_ws/src/drone/drone/launch/aruco.launch` должны быть в значении `true` :

```
<arg name="aruco_detect" default="true"/>
<arg name="aruco_map" default="true"/>
```

Для включения передачи координат в полетный контроллер по механизму VPE, аргумента `aruco_vpe` должен быть в значении `true` :

```
<arg name="aruco_vpe" default="true"/>
```

Настройка карты маркеров

Карта загружается из текстового файла, каждая строка которого имеет следующий формат:

```
id_маркера размер_маркера x y z угол_z угол_y угол_x
```

Где `угол_N` – это угол поворота маркера вокруг оси N в радианах.

Файлы карт располагаются в каталоге `~/catkin_ws/src/drone/aruco_pose/map` . Название файла с картой задается в аргументе `map` :

```
<arg name="map" default="map.txt"/>
```

Файл карты может быть сгенерирован с помощью инструмента `genmap.py` :

```
roslaunch aruco_pose genmap.py length x y dist_x dist_y first -o test_map.txt
```

Где `length` – размер маркера, `x` – количество маркеров по оси `x`, `y` – количество маркеров по оси `y`, `dist_x` – расстояние между центрами маркеров по оси `x`, `dist_y` – расстояние между центрами маркеров по оси `y`, `first` – ID первого (левого нижнего) маркера, `test_map.txt` – название файла с картой. Дополнительный ключ `--bottom-left` позволяет нумеровать маркеры с левого нижнего угла.

Пример:

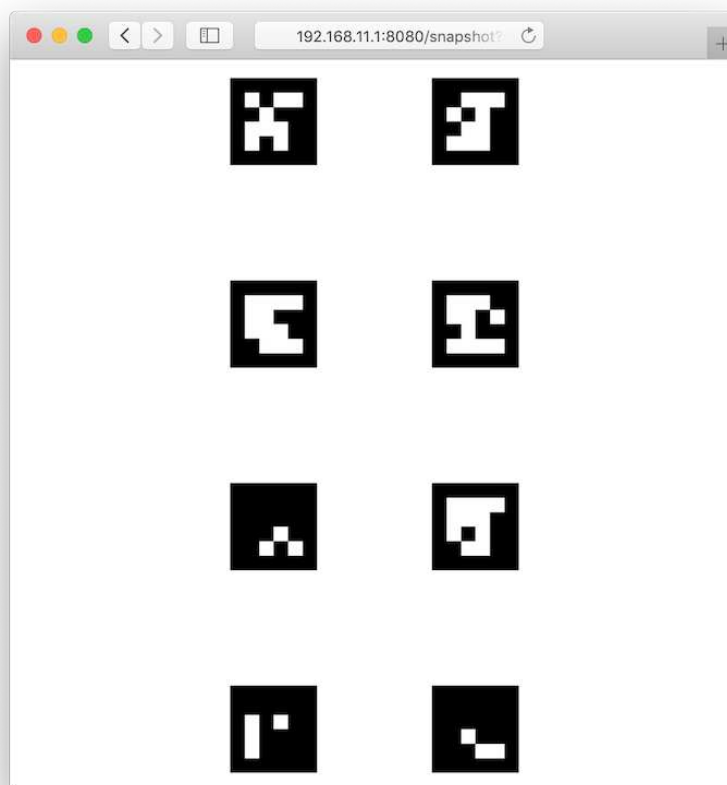
```
roslaunch aruco_pose genmap.py 0.33 2 4 1 1 0 -o test_map.txt
```

Дополнительную информацию по утилите можно получить по ключу `-h`:

```
roslaunch aruco_pose genmap.py -h .
```

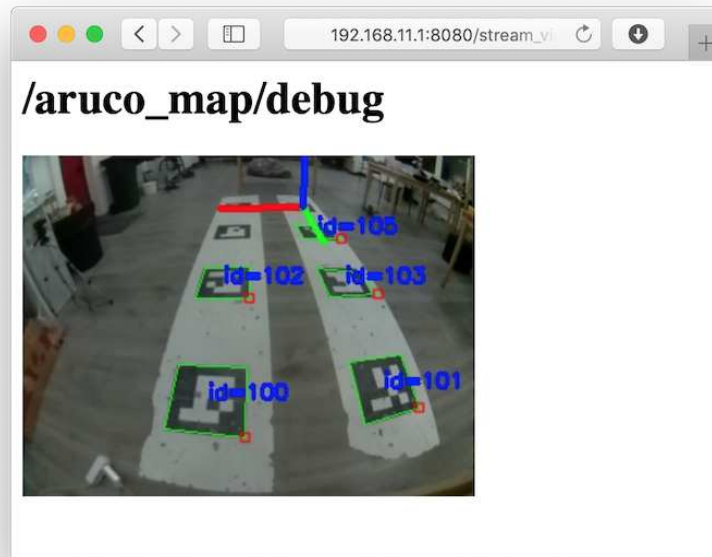
Проверка

Для контроля карты, по которой в данный момент дрон осуществляет навигацию, можно просмотреть содержимое топика `/aruco_map/image`. Через браузер его можно просмотреть при помощи [web_video_server](#) по ссылке http://192.168.11.1:8080/snapshot?topic=/aruco_map/image:



Дрон публикует текущую позицию распознанной карты в топик `aruco_map/pose`. Также публикуется **TF-фрейм** `aruco_map` (VPE выключен) или `aruco_map_detected` (VPE включен).

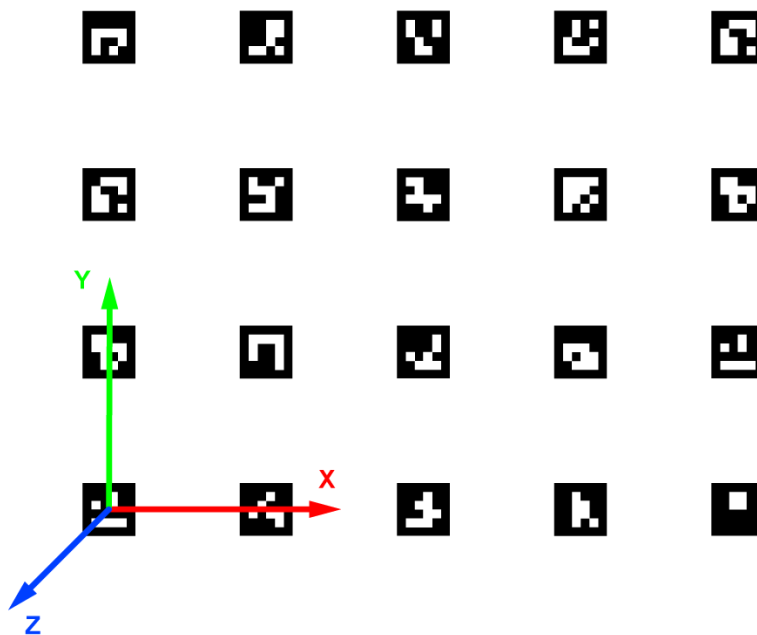
Наглядно позиция распознанной карты отображается в топике `aruco_map/debug` (просмотр доступен по ссылке http://192.168.11.1:8080/stream_viewer?topic=/aruco_map/debug):



Система координат

По [соглашению](#) в маркерном поле используется стандартная система координат ENU:

- ось **x** указывает на правую сторону карты маркеров;
- ось **y** указывает кверху карты маркеров;
- ось **z** указывает от плоскости карты маркеров.



Полет

При правильной настройке дрон начнет удерживать позицию в [режимах POSCTL и OFFBOARD](#) автоматически.

Для [автономных полетов](#) можно будет использовать функции `navigate`, `set_position`, `set_velocity`. Для полета в определенные координаты маркерного поля необходимо использовать фрейм `aruco_map`:

```
# Вначале необходимо взлететь, чтобы дрон увидел карту меток и появился фрейм aruco_map
navigate(x=0, y=0, z=2, frame_id='body', speed=0.5, auto_arm=True) # взлет на высоту 2 метра

time.sleep(5)

# Полет в координату 2:2 маркерного поля, высота 2 метра
navigate(x=2, y=2, z=2, speed=1, frame_id='aruco_map') # полет в координату 2:2
```

Полет в координаты по ID маркера

Доступны также полёты относительно отдельного маркера в карте, даже если дрон его не видит. По аналогии с [навигацией по отдельным маркерам](#) при настройке карты маркеров дрон сможет лететь в координаты относительно отдельного маркера, используя фрейм `aruco_ID` с соответствующим ID маркера.

Полет в точку над маркером 5 на высоту 1 метр:

```
navigate(frame_id='aruco_5', x=0, y=0, z=1)
```


Дополнительные настройки

Если дрон нестабильно удерживает позицию по VPE, попробуйте увеличить коэффициенты P PID-регулятора по скорости – параметры `MPC_XY_VEL_P` и `MPC_Z_VEL_P`.

Если дрон нестабильно удерживает высоту, попробуйте увеличить коэффициент `MPC_Z_VEL_P` или лучше подобрать газ висения – `MPC_THR_HOVER`.

Расположение маркеров на потолке



Для навигации по маркерам, расположенным на потолке, необходимо поставить основную камеру так, чтобы она смотрела вверх и [установить соответствующий фрейм камеры](#).

Также в файле `~/catkin_ws/src/drone/drone/launch/aruco.launch` необходимо выставить аргумент `placement` в значение `ceiling`:

```
<arg name="placement" default="ceiling"/>
```

При такой конфигурации фрейм `aruco_map` также окажется перевернутым. Таким образом, для полета на высоту 2 метра ниже потолка, аргумент `z` нужно устанавливать в 2:

```
navigate(x=1, y=2, z=1.1, speed=0.5, frame_id='aruco_map')
```

Автономный полет

Модуль `simple_offboard` предназначен для упрощенного программирования автономного полета дрона (режим `OFFBOARD`). Он позволяет устанавливать желаемые полетные задачи и автоматически трансформирует систему координат.

`simple_offboard` является высокоуровневым способом взаимодействия с полетным контроллером. Для более низкоуровневой работы см. `mavros`.

Основные сервисы – `get_telemetry` (получение телеметрии), `navigate` (полет в заданную точку по прямой), `navigate_global` (полет в глобальную точку по прямой), `land` (переход в режим посадки).

Использование из языка Python

Для использования сервисов, необходимо создать объекты-прокси к ним. Используйте этот шаблон для вашей программы:

```
import rospy
from drone import srv
from std_srvs.srv import Trigger

rospy.init_node('flight')

get_telemetry = rospy.ServiceProxy('get_telemetry', srv.GetTelemetry)
navigate = rospy.ServiceProxy('navigate', srv.Navigate)
navigate_global = rospy.ServiceProxy('navigate_global', srv.NavigateGlobal)
set_altitude = rospy.ServiceProxy('set_altitude', srv.SetAltitude)
set_yaw = rospy.ServiceProxy('set_yaw', srv.SetYaw)
set_yaw_rate = rospy.ServiceProxy('set_yaw_rate', srv.SetYawRate)
set_position = rospy.ServiceProxy('set_position', srv.SetPosition)
set_velocity = rospy.ServiceProxy('set_velocity', srv.SetVelocity)
set_attitude = rospy.ServiceProxy('set_attitude', srv.SetAttitude)
set_rates = rospy.ServiceProxy('set_rates', srv.SetRates)
land = rospy.ServiceProxy('land', Trigger)
```

Неиспользуемые функции-прокси можно удалить из кода.

Описание API

Незаполненные числовые параметры устанавливаются в значение 0.

get_telemetry

Получить полную телеметрию коптера.

Параметры:

- `frame_id` – система координат для значений `x`, `y`, `z`, `vx`, `vy`, `vz`.
Пример: `map`, `body`, `aruco_map`. Значение по умолчанию: `map`.

Формат ответа:

- `frame_id` – система координат;
- `connected` – есть ли подключение к `FCU`;
- `armed` – состояние `armed` пропеллеров (пропеллеры включены, если `true`);
- `mode` – текущий полетный режим;
- `x`, `y`, `z` – локальная позиция коптера (*м*);
- `lat`, `lon` – широта, долгота (*градусы*), необходимо наличие `GPS`;
- `alt` – высота в глобальной системе координат (стандарт `WGS-84`, не `AMSL!`), необходимо наличие `GPS`;
- `vx`, `vy`, `vz` – скорость коптера (*м/с*);
- `roll` – угол по крену (*радианы*);
- `pitch` – угол по тангажу (*радианы*);
- `yaw` – угол по рысканью (*радианы*);
- `roll_rate` – угловая скорость по крену (*рад/с*);
- `pitch_rate` – угловая скорость по тангажу (*рад/с*);
- `yaw_rate` – угловая скорость по рысканью (*рад/с*);
- `voltage` – общее напряжение аккумулятора (*В*);
- `cell_voltage` – напряжение аккумулятора на ячейку (*В*).

Недоступные по каким-то причинам поля будут содержать в себе значения `NaN`.

Вывод координат `x`, `y` и `z` коптера в локальной системе координат:

```
telemetry = get_telemetry()
print(telemetry.x, telemetry.y, telemetry.z)
```

Вывод высоты коптера относительно карты `ArUco`-меток:

```
telemetry = get_telemetry(frame_id='aruco_map')
print(telemetry.z)
```

Проверка доступности глобальной позиции:

```
import math
if not math.isnan(get_telemetry().lat):
    print('Global position is available')
else:
    print('No global position')
```

Вывод текущей телеметрии (командная строка):

```
rosservice call /get_telemetry "{frame_id: ''}"
```

navigate

Прилететь в обозначенную точку по прямой.

Параметры:

- `x`, `y`, `z` – координаты (м);
- `yaw` – угол по рысканью (радианы);
- `speed` – скорость полета (скорость движения setpoint) (м/с);
- `auto_arm` – перевести коптер в `OFFBOARD` и заармить автоматически (коптер взлетит);
- `frame_id` – система координат, в которой заданы `x`, `y`, `z` и `yaw` (по умолчанию: `map`).

Для полета без изменения угла по рысканью достаточно установить `yaw` в `NaN` (значение угловой скорости по умолчанию – 0).

Взлет на высоту 1.5 м со скоростью взлета 0.5 м/с:

```
navigate(x=0, y=0, z=1.5, speed=0.5, frame_id='body', auto_arm=True)
```

Полет по прямой в точку 5:0 (высота 2) в локальной системе координат со скоростью 0.8 м/с (рысканье установится в 0):

```
navigate(x=5, y=0, z=3, speed=0.8)
```

Полет в точку 5:0 без изменения угла по рысканью:

```
navigate(x=5, y=0, z=3, speed=0.8, yaw=float('nan'))
```

Полет вправо относительно коптера на 3 м:

```
navigate(x=0, y=-3, z=0, speed=1, frame_id='body')
```

Полет влево на 2 м относительно последней целевой точки полета дрона:

```
navigate(x=0, y=2, z=0, speed=1, frame_id='navigate_target')
```

Повернуться на 90 градусов по часовой:

```
navigate(yaw=math.radians(-90), frame_id='body')
```

Полет в точку 3:2 (высота 2) в системе координат маркерного поля со скоростью 1 м/с:

```
navigate(x=3, y=2, z=2, speed=1, frame_id='aruco_map')
```

Взлет на высоту 2 м (командная строка):

```
rosservice call /navigate "{x: 0.0, y: 0.0, z: 2, yaw: 0.0, speed: 0.5, frame_:
```

При программировании миссии дрона в терминах "вперед-назад-влево-вправо" рекомендуется использовать систему координат `navigate_target` ВМЕСТО `body`, чтобы не учитывать неточность прилета дрона в предыдущую целевую точку при вычислении следующей.

navigate_global

Полет по прямой в точку в глобальной системе координат (широта/долгота).

Параметры:

- `lat`, `lon` – широта и долгота (*градусы*);
- `z` – высота (*м*);
- `yaw` – угол по рысканью (*радианы*);
- `speed` – скорость полета (скорость движения setpoint) (*м/с*);
- `auto_arm` – перевести коптер в `OFFBOARD` и заармить автоматически (**коптер взлетит**);
- `frame_id` – **система координат**, в которой заданы `z` и `yaw` (по умолчанию: `map`).

Для полета без изменения угла по рысканью достаточно установить `yaw` в `NaN`.

Полет в глобальную точку со скоростью 5 м/с, оставаясь на текущей высоте (`yaw` установится в 0, коптер сориентируется передом на восток):

```
navigate_global(lat=55.707033, lon=37.725010, z=0, speed=5, frame_id='body')
```

Полет в глобальную точку без изменения угла по рысканью:

```
navigate_global(lat=55.707033, lon=37.725010, z=0, speed=5, yaw=float('nan'), 1
```

Полет в глобальную точку (командная строка):

```
rosservice call /navigate_global "{lat: 55.707033, lon: 37.725010, z: 0.0, yaw
```

set_altitude

Изменить целевую высоту полета. Сервис используется для независимой установки высоты (и системы координат для расчета высота) в режимах полета `navigate` и `set_position`.

Параметры:

- `z` – высота полета (*м*);
- `frame_id` – **система координат** для расчета высоты полета.

Установить высоту полета в 2 м относительно пола:

```
set_altitude(z=2, frame_id='terrain')
```

Установить высоту полета в 1 м относительно [маркерного поля](#):

```
set_altitude(z=1, frame_id='aruco_map')
```

set_yaw

Изменить целевой угол по рысканью (и систему координат для его расчета), оставив предыдущую команду в силе.

Параметры:

- `yaw` – угол по рысканью (*радианы*);
- `frame_id` – [система координат](#) для расчета угла по рысканью.

Повернуться на 90 градусов по часовой (продолжая выполнять предыдущую команду):

```
set_yaw(yaw=math.radians(-90), frame_id='body')
```

Установить угол по рысканью в ноль в системе координат [маркерного поля](#):

```
set_yaw(yaw=0, frame_id='aruco_map')
```

Остановить вращение по рысканью (при использовании [set_yaw_rate](#)):

```
set_yaw(yaw=float('nan'))
```

set_yaw_rate

Изменить целевую угловую скорость по рысканью, оставив предыдущую команду в силе.

Параметры:

- `yaw_rate` – угловая скорость по рысканью (*рад/с*).

Положительное направление вращения (при виде сверху) – против часовой.

Начать вращение на месте со скоростью 0.5 рад/с против часовой (продолжая выполнять предыдущую команду):

```
set_yaw_rate(yaw_rate=0.5)
```

set_position

Установить цель по позиции и рысканью. Данный сервис следует использовать при необходимости задания продолжающегося потока целевых точек, например, для полета по сложным траекториям (круговой, дугообразной и т. д.).

Для полета на точку по прямой или взлета используйте более высокоуровневый сервис `navigate`.

Параметры:

- `x`, `y`, `z` – координаты точки (м);
- `yaw` – угол по рысканью (радианы);
- `auto_arm` – перевести коптер в `OFFBOARD` и заармить автоматически (коптер взлетит);
- `frame_id` – система координат, в которой заданы `x`, `y`, `z` и `yaw` (по умолчанию: `map`).

Зависнуть на месте:

```
set_position(frame_id='body')
```

Назначить целевую точку на 3 м выше текущей позиции:

```
set_position(x=0, y=0, z=3, frame_id='body')
```

Назначить целевую точку на 1 м впереди текущей позиции:

```
set_position(x=1, y=0, z=0, frame_id='body')
```

set_velocity

Установить скорости и рысканье.

- `vx`, `vy`, `vz` – требуемая скорость полета (м/с);
- `yaw` – угол по рысканью (радианы);
- `auto_arm` – перевести коптер в `OFFBOARD` и заармить автоматически (коптер взлетит);
- `frame_id` – система координат, в которой заданы `vx`, `vy`, `vz` и `yaw` (по умолчанию: `map`).

Параметр `frame_id` определяет только ориентацию результирующего вектора скорости, но не его длину.

Полет вперед (относительно коптера) со скоростью 1 м/с:

```
set_velocity(vx=1, vy=0.0, vz=0, frame_id='body')
```

set_attitude

Установить тангаж, крен, рысканье и уровень газа (примерный аналог управления в режиме `STABILIZED`). Данный сервис может быть использован для более низкоуровневого контроля поведения коптера либо для управления коптером при отсутствии источника достоверных данных о его позиции.

Параметры:

- `roll`, `pitch`, `yaw` – необходимый угол по тангажу, крену и рысканью (*радианы*);
- `thrust` – уровень газа от 0 (нет газа, пропеллеры остановлены) до 1 (полный газ);
- `auto_arm` – перевести коптер в `OFFBOARD` и заармить автоматически (**коптер взлетит**);
- `frame_id` – **система координат**, в которой задан `yaw` (по умолчанию: `map`).

set_rates

Установить угловые скорости по тангажу, крену и рысканью и уровень газа (примерный аналог управления в **режиме ACRO**). Это самый низкий уровень управления коптером (исключая непосредственный контроль оборотов моторов). Данный сервис может быть использован для автоматического выполнения акробатических трюков (например, флипа).

Параметры:

- `roll_rate`, `pitch_rate`, `yaw_rate` – угловая скорость по тангажу, крену и рысканью (*рад/с*);
- `thrust` – уровень газа от 0 (нет газа, пропеллеры остановлены) до 1 (полный газ).
- `auto_arm` – перевести коптер в `OFFBOARD` и заармить автоматически (**коптер взлетит**);

Положительное направление вращения `yaw_rate` (при виде сверху) – против часовой, `pitch_rate` – вперед, `roll_rate` – влево.

land

Перевести коптер в **режим** посадки (`AUTO.LAND` или аналогичный).

Посадка коптера:

```
res = land()

if res.success:
    print('Copter is landing')
```

Посадка коптера (командная строка):

```
rosservice call /land "{}"
```

release

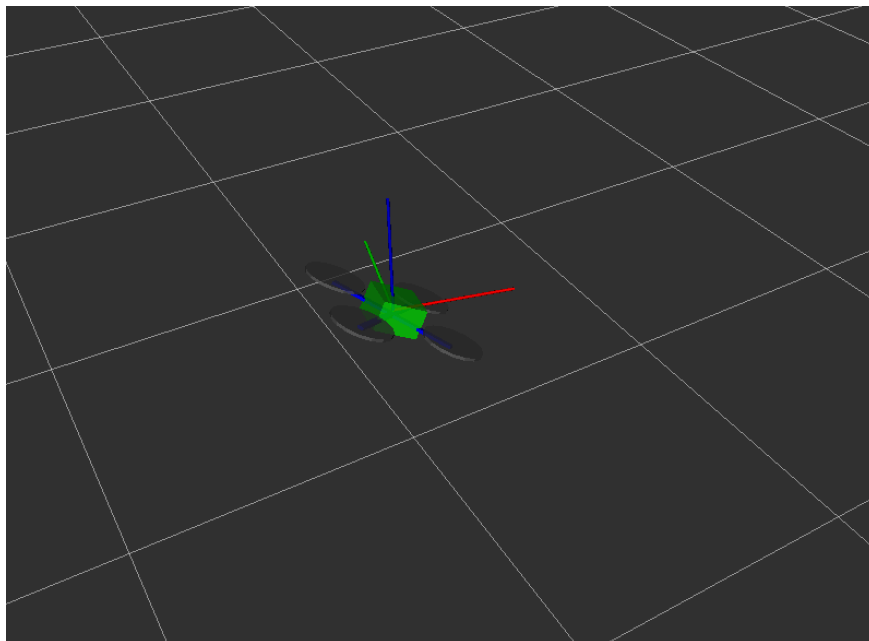
В случае необходимости приостановки отправки `setpoint`-сообщений, используйте сервис `simple_offboard/release`:


```
release = rospy.ServiceProxy('simple_offboard/release', Trigger)
release()
```

Дополнительные материалы

- [Полеты в поле ArUco-маркеров.](#)
- [Примеры программ и сниппеты.](#)

Системы координат (фреймы)



Основные фреймы:

- `map` — координаты относительно точки инициализации полетного контроллера: белая сетка на иллюстрации;
- `base_link` — координаты относительно дрона: схематичное изображение дрона на иллюстрации;
- `body` — координаты относительно дрона без учета наклонов по тангажу и крену: красная, синяя и зеленая линии на иллюстрации;
- `navigate_target` — координаты точки, в которую сейчас летит дрон (с использованием `navigate`);
- `terrain` — координаты относительно пола в текущей позиции коптера (см. сервис `set_altitude`);
- `setpoint` — текущий setpoint по позиции;
- `main_camera_optical` — система координат, [связанная с основной камерой](#).

При использовании [системы позиционирования по ArUco-маркерам](#) появляются дополнительные фреймы:

- `aruco_map` — координаты относительно [карты ArUco-маркеров](#);
- `aruco_N` — координаты относительно [маркера с ID=N](#).

В соответствии с [соглашением](#), для фреймов, связанных с коптером, ось X направлена вперед, Y — налево и Z — вверх.

tf2

Основная документация: <http://wiki.ros.org/tf2>

tf2 – это набор библиотек для языков программирования C++, Python и других, которые помогают работать с системами координат. ROS-ноды публикуют в топик `/tf` сообщения формата `TransformStamped`, которые содержат в себе трансформации между заданными системами координат в определенные моменты времени.

С помощью `simple_offboard` можно запросить расположение коптера в любой системе координат, используя аргумент `frame_id` сервиса `get_telemetry`.

Из Python можно использовать библиотеку tf2 для преобразования геометрических объектов (например, `PoseStamped`, `PointStamped`) из одной системы координат в другую.

Работа с GPIO

GPIO (General-Purpose Input/Output) – это тип пинов на Raspberry Pi, напряжение на которых можно программно подавать и измерять. Также на некоторых пинах реализован аппаратный ШИМ (PWM). Интерфейс GPIO может быть использован для управления различной периферией: светодиодами, электромагнитами, электромоторами, сервоприводами и т. д.

Используйте [распиновку](#), чтобы понять, какие из пинов на Raspberry Pi поддерживают GPIO и ШИМ.

Для того, чтобы не создавалось конфликтов при использовании портов GPIO в образе закрыт доступ для портов 0, 1, 2, 3, 14, 15, на которые выведены интерфейсы подключения I2C и UART.

Для работы с GPIO на [образе для RPi](#) предустановлена библиотека `pigpio`. Чтобы взаимодействовать с этой библиотекой, запустите соответствующий демон:

```
sudo systemctl start pigpiod.service
```

Для включения автозапуска демона `pigpiod` используйте команду:

```
sudo systemctl enable pigpiod.service
```

Пример работы с библиотекой:

```
import time
import pigpio

# инициализируем подключение к pigpiod
pi = pigpio.pi()

# устанавливаем режим 11 пина на вывод
pi.set_mode(11, pigpio.OUTPUT)

# включаем сигнал на 11 пине
pi.write(11, 1)

time.sleep(2)

# отключаем сигнал на 11 пине
pi.write(11, 0)

# ...

# устанавливаем режим 12 пина на ввод
pi.set_mode(12, pigpio.INPUT)

# считываем состояние 12 пина
level = pi.read(12)
```

Для определения номера пина используйте [распиновку Raspberry Pi](#).

Подключение сервоприводов

Большинство сервоприводов управляются с помощью ШИМ-сигнала, причем крайним положениям привода соответствуют сигналы шириной приблизительно 1000 и 2000 мкс. Значения для конкретного сервопривода могут быть определены экспериментально.

Подключите сигнальный провод сервопривода к одному из GPIO-пинов Raspberry. Для управления сервоприводом, подключенного к 13 пину, используйте такой код:

```
import time
import pigpio

pi = pigpio.pi()

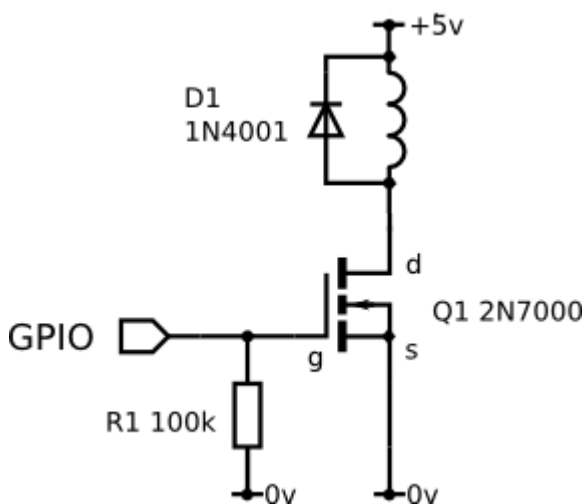
# устанавливаем режим 13 пина на вывод
pi.set_mode(13, pigpio.OUTPUT)

# устанавливаем на 13 пине ШИМ сигнал в 1000 мкс
pi.set_servo_pulsewidth(13, 1000)

time.sleep(2)

# устанавливаем на 13 пине ШИМ сигнал в 2000 мкс
pi.set_servo_pulsewidth(13, 2000)
```

Подключение электромагнита



Для подключения электромагнита используйте полевой транзистор (MOSFET). Подключите транзистор к одному из GPIO-пинов Raspberry Pi. Для управления магнитом, подключенным к 18 пину, используйте такой код:

Датчик температуры и влажности

```
import time
import pigpio

pi = pigpio.pi()

# устанавливаем режим 18 пина на вывод
pi.set_mode(18, pigpio.OUTPUT)

# включаем электромагнит
pi.write(18, 1)

time.sleep(2)

# отключаем электромагнит
pi.write(18, 0)
```

Примеры кода

Python

При использовании кириллических символов в кодировке UTF-8 необходимо добавить в начало программы указание кодировки:

```
# -*- coding: utf-8 -*-
```

#

Функция для полета в точку и ожидание окончания полета:

```
import math

def navigate_wait(x=0, y=0, z=0, yaw=float('nan'), speed=0.5, frame_id='', auto_arm=False):
    navigate(x=x, y=y, z=z, yaw=yaw, speed=speed, frame_id=frame_id, auto_arm=auto_arm)

    while not rospy.is_shutdown():
        telem = get_telemetry(frame_id='navigate_target')
        if math.sqrt(telem.x ** 2 + telem.y ** 2 + telem.z ** 2) < tolerance:
            break
        rospy.sleep(0.2)
```

Для того, чтобы определить расстояние до целевой точки, функция использует фрейм `navigate_target`.

Использование функции для полета в точку $x=3$, $y=2$, $z=1$ **относительно карты маркеров**:

```
navigate_wait(x=3, y=2, z=1, frame_id='aruco_map')
```

Эту функцию можно использовать и для взлета:

```
navigate_wait(z=1, frame_id='body', auto_arm=True)
```

#

Посадка и ожидание окончания посадки:

```
def land_wait():
    land()
    while get_telemetry().armed:
        rospy.sleep(0.2)
```

Использование:

```
land_wait()
```

#

Ожидание окончания прилета в [navigate](#)-точку:

```
import math

def wait_arrival(tolerance=0.2):
    while not rospy.is_shutdown():
        telem = get_telemetry(frame_id='navigate_target')
        if math.sqrt(telem.x ** 2 + telem.y ** 2 + telem.z ** 2) < tolerance:
            break
        rospy.sleep(0.2)
```

#

Функция определения расстояния между двумя точками (**важно**: точки должны быть в одной [системе координат](#)):

```
import math

def get_distance(x1, y1, z1, x2, y2, z2):
    return math.sqrt((x1 - x2) ** 2 + (y1 - y2) ** 2 + (z1 - z2) ** 2)
```

#

Функция для приблизительного определения расстояния (в метрах) между двумя глобальными координатами (широта/долгота):

```
import math

def get_distance_global(lat1, lon1, lat2, lon2):
    return math.hypot(lat1 - lat2, lon1 - lon2) * 1.113195e5
```

#

Дизарм коптера (выключение пропеллеров, **коптер упадет**):

```
# Объявление прокси:
from mavros_msgs.srv import CommandBool
arming = rospy.ServiceProxy('mavros/cmd/arming', CommandBool)

# ...

arming(False) # дизарм
```

#

Трансформировать позицию (`PoseStamped`) из одной системы координат ([фрейма](#)) в другую, используя [tf2](#):


```
import tf2_ros
import tf2_geometry_msgs
from geometry_msgs.msg import PoseStamped

tf_buffer = tf2_ros.Buffer()
tf_listener = tf2_ros.TransformListener(tf_buffer)

# ...

# Создаем объект PoseStamped (либо получаем из топика):
pose = PoseStamped()
pose.header.frame_id = 'map' # фрейм, в котором задана позиция
pose.header.stamp = rospy.get_rostime() # момент времени, для которого задана
pose.pose.position.x = 1
pose.pose.position.y = 2
pose.pose.position.z = 3
pose.pose.orientation.w = 1

frame_id = 'base_link' # целевой фрейм
transform_timeout = rospy.Duration(0.2) # таймаут ожидания трансформации

# Преобразовываем позицию из старого фрейма в новый:
new_pose = tf_buffer.transform(pose, frame_id, transform_timeout)
```

#

Определение, перевернут ли коптер:

```
PI_2 = math.pi / 2
telem = get_telemetry()

flipped = abs(telem.roll) > PI_2 or abs(telem.pitch) > PI_2
```

#

Расчет общего угла коптера к горизонту:

```
PI_2 = math.pi / 2
telem = get_telemetry()

flipped = not -PI_2 <= telem.roll <= PI_2 or not -PI_2 <= telem.pitch <= PI_2
angle_to_horizon = math.atan(math.hypot(math.tan(telem.pitch), math.tan(telem.roll)))
if flipped:
    angle_to_horizon = math.pi - angle_to_horizon
```

#

Полет по круговой траектории:

```
RADIUS = 0.6 # m
SPEED = 0.3 # rad / s

start = get_telemetry()
start_stamp = rospy.get_rostime()

r = rospy.Rate(10)

while not rospy.is_shutdown():
    angle = (rospy.get_rostime() - start_stamp).to_sec() * SPEED
    x = start.x + math.sin(angle) * RADIUS
    y = start.y + math.cos(angle) * RADIUS
    set_position(x=x, y=y, z=start.z)

    r.sleep()
```

#

Повторять действие с частотой 10 Гц:

```
r = rospy.Rate(10)
while not rospy.is_shutdown():
    # Do anything
    r.sleep()
```

#

Реакция на переключение режима на пульте радиоуправления (может быть использовано для запуска автономного полета):

```
from mavros_msgs.msg import RCIn

# Вызывается при получении новых данных с пульта
def rc_callback(data):
    # Произвольная реакция на переключение тумблера на пульте
    if data.channels[5] < 1100:
        # ...
        pass
    elif data.channels[5] > 1900:
        # ...
        pass
    else:
        # ...
        pass

# Создаем подписчик на топик с данными с пульта
rospy.Subscriber('mavros/rc/in', RCIn, rc_callback)

rospy.spin()
```

#

Сменить режим полета на произвольный:

```
from mavros_msgs.srv import SetMode

set_mode = rospy.ServiceProxy('mavros/set_mode', SetMode)

# ...

set_mode(custom_mode='STABILIZED')
```

#

Флип:

```
import math

PI_2 = math.pi / 2

def flip():
    start = get_telemetry() # memorize starting position

    set_rates(thrust=1) # bump up
    rospy.sleep(0.2)

    set_rates(pitch_rate=30, thrust=0.2) # pitch flip
    # set_rates(roll_rate=30, thrust=0.2) # roll flip

    while True:
        telem = get_telemetry()
        flipped = abs(telem.roll) > PI_2 or abs(telem.pitch) > PI_2
        if flipped:
            break

    rospy.loginfo('finish flip')
    set_position(x=start.x, y=start.y, z=start.z, yaw=start.yaw) # finish flip

print(navigate(z=2, speed=1, frame_id='body', auto_arm=True)) # take off
rospy.sleep(10)

rospy.loginfo('flip')
flip()
```

#

Произвести калибровку гироскопа:

```

from pymavlink import mavutil
from mavros_msgs.srv import CommandLong
from mavros_msgs.msg import State

send_command = rospy.ServiceProxy('mavros/cmd/command', CommandLong)

def calibrate_gyro():
    rospy.loginfo('Calibrate gyro')
    if not send_command(command=mavutil.mavlink.MAV_CMD_PREFLIGHT_CALIBRATION,
                        return False

    calibrating = False
    while not rospy.is_shutdown():
        state = rospy.wait_for_message('mavros/state', State)
        if state.system_status == mavutil.mavlink.MAV_STATE_CALIBRATING or state.system_status == mavutil.mavlink.MAV_STATE_CRITICAL:
            calibrating = True
        elif calibrating and state.system_status == mavutil.mavlink.MAV_STATE_STANDBY:
            rospy.loginfo('Calibrating finished')
            return True

calibrate_gyro()

```

В процессе калибровки гироскопов дрон нельзя двигать.

#

Динамически включать и отключать [распознавание ArUco-маркеров](#) (например, для экономии ресурсов процессора):

```

import rospy
import dynamic_reconfigure.client

rospy.init_node('flight')
aruco_client = dynamic_reconfigure.client.Client('aruco_detect')

# Выключить распознавание маркеров
aruco_client.update_configuration({'enabled': False})

rospy.sleep(5)

# Включить распознавание маркеров
aruco_client.update_configuration({'enabled': True})

```

#

Динамически изменить используемый файл с [картой ArUco-маркеров](#):

```

import rospy
import dynamic_reconfigure.client

rospy.init_node('flight')
map_client = dynamic_reconfigure.client.Client('aruco_map')

map_client.update_configuration({'map': '/home/pi/catkin_ws/src/drone/aruco_pos

```

#

Ожидать появления глобальной позиции (окончания инициализации [GPS-приемника](#)):

```
import math

while not rospy.is_shutdown():
    if math.isfinite(get_telemetry().lat):
        break
    rospy.sleep(0.2)
```

#

Считать параметр полетного контроллера:

```
from mavros_msgs.srv import ParamGet
from mavros_msgs.msg import ParamValue

param_get = rospy.ServiceProxy('mavros/param/get', ParamGet)

# Считать параметр типа INT
value = param_get(param_id='COM_FLTMODE1').value.integer

# Считать параметр типа FLOAT
value = param_get(param_id='MPC_Z_P').value.float
```

#

Изменить параметр полетного контроллера:

```
from mavros_msgs.srv import ParamSet
from mavros_msgs.msg import ParamValue

param_set = rospy.ServiceProxy('mavros/param/set', ParamSet)

# Изменить параметр типа INT:
param_set(param_id='COM_FLTMODE1', value=ParamValue(integer=8))

# Изменить параметр типа FLOAT:
param_set(param_id='MPC_Z_P', value=ParamValue(real=1.5))
```

Работа с камерой

Для работы с основной камерой необходимо убедиться что она включена в файле `~/catkin_ws/src/drone/drone/launch/drone.launch` :

```
<arg name="main_camera" default="true"/>
```

Также нужно убедиться, что камера [сфокусирована](#) и для нее [указано корректное расположение и ориентация](#).

При изменении launch-файла необходимо перезапустить пакет `drone` :

```
sudo systemctl restart drone
```

Для мониторинга изображения с камеры можно использовать [web_video_server](#).

Неисправности

Если изображение с камеры отсутствует, попробуйте проверить ее с помощью утилиты `raspistill` .

Остановите сервисы:

```
sudo systemctl stop drone
```

Получите картинку с камеры утилитой `raspistill` :

```
raspistill -o test.jpg
```

Если команда завершается с ошибкой, проверьте качество подключения шлейфа камеры к Raspberry Pi или замените его.

Настройки камеры

Ряд параметров камеры - размер изображения, максимальную частоту кадров, экспозицию - можно настроить в файле `main_camera.launch` . Список настраиваемых параметров можно [посмотреть в репозитории cv_camera](#).

Параметры, не указанные в этом списке, можно указывать через [код параметра OpenCV](#). Например, для установки фиксированной экспозиции добавьте следующие параметры в ноду камеры:

```
<param name="property_0_code" value="21"/> <!-- property code 21 is CAP_PROP_AI
<param name="property_0_value" value="0.25"/> <!-- property values are normaliz
<param name="cv_cap_prop_exposure" value="0.3"/> <!-- set exposure to 30% of ma
```

Компьютерное зрение

Для реализации алгоритмов компьютерного зрения рекомендуется использовать предустановленную на образ SD-карты библиотеку [OpenCV](#).

Python

Пример создания подписчика на топик с изображением с основной камеры для обработки с использованием OpenCV:

```
import rospy
import cv2
from sensor_msgs.msg import Image
from cv_bridge import CvBridge
from drone import long_callback

rospy.init_node('cv')
bridge = CvBridge()

@long_callback
def image_callback(data):
    img = bridge.imgmsg_to_cv2(data, 'bgr8') # OpenCV image
    # Do any image processing with cv2...

image_sub = rospy.Subscriber('main_camera/image_raw', Image, image_callback)

rospy.spin()
```

Обработка изображения может занимать значительное время. Это может вызвать [проблему](#) в библиотеке `rospy`, которая приведет к обработке устаревших кадров с камеры. Для решения этой проблемы необходимо использовать декоратор `long_callback` из библиотеки `drone`, как в примере выше.

Ограничение использования CPU

При использовании топика `main_camera/image_raw` скрипт будет обрабатывать максимальное количество кадров с камеры, активно используя CPU (вплоть до 100%). В задачах, где обработка каждого кадра не критична, можно использовать топик, где кадры публикуются с частотой 5 Гц:

```
main_camera/image_raw_throttled :
```

```
image_sub = rospy.Subscriber('main_camera/image_raw_throttled', Image, image_cb)
```

Публикация изображений

Для отладки обработки изображения можно публиковать отдельный топик с обработанным изображением:

```
image_pub = rospy.Publisher('-debug', Image)
```

Публикация обработанного изображения:

```
image_pub.publish(bridge.cv2_to_imgmsg(img, 'bgr8'))
```

Получаемые изображения можно просматривать используя [web_video_server](#).

Получение одного кадра

Существует возможность единоразового получения кадра с камеры. Этот способ работает медленнее, чем подписка на топик; его не следует применять в случае необходимости постоянной обработки изображений.

```
import rospy
from sensor_msgs.msg import Image
from cv_bridge import CvBridge

rospy.init_node('cv')
bridge = CvBridge()

# ...

# Retrieve a frame:
img = bridge.imgmsg_to_cv2(rospy.wait_for_message('main_camera/image_raw', Image
```

Примеры

Работа с QR-кодами

Для высокоскоростного распознавания и позиционирования лучше использовать [ArUco-маркеры](#).

Для программирования различных действий дрона при детектировании нужных [QR-кодов](#) можно использовать библиотеку [pyZBar](#). Она уже установлена в последнем образе для Raspberry Pi.

Распознавание QR-кодов на Python:


```
import rospy
from pyzbar import pyzbar
import cv2
from cv_bridge import CvBridge
from sensor_msgs.msg import Image
from drone import long_callback

rospy.init_node('cv')
bridge = CvBridge()

@long_callback
def image_callback(msg):
    img = bridge.imgmsg_to_cv2(msg, 'bgr8')
    barcodes = pyzbar.decode(img)
    for barcode in barcodes:
        b_data = barcode.data.decode('utf-8')
        b_type = barcode.type
        (x, y, w, h) = barcode.rect
        xc = x + w/2
        yc = y + h/2
        print('Found {} with data {} with center at x={}, y={}'.format(b_type,
                                b_data, xc, yc))

image_sub = rospy.Subscriber('main_camera/image_raw_throttled', Image, image_callback)

rospy.spin()
```

Запись видео

Для записи видео может быть использована нода `video_recorder` из пакета

`image_view` :

```
roslaunch image_view video_recorder image:=/main_camera/image_raw
```

Видео будет сохранено в файл `output.avi` . В аргументе `image` указывается название топика для записи видео.

Автозапуск ПО

systemd

Основная документация:

[https://wiki.archlinux.org/index.php/Systemd_\(Русский\)](https://wiki.archlinux.org/index.php/Systemd_(Русский)).

Все автоматически стартуемое ПО запускается в виде systemd-сервиса

```
drone.service .
```

Сервис может быть перезапущен командой `systemctl` :

```
sudo systemctl restart drone
```

Текстовый вывод ПО можно посмотреть с помощью команды `journalctl` :

```
journalctl -u drone
```

Для того чтобы запустить ПО непосредственно в текущей консольной сессии, вы можете использовать `roslaunch` :

```
sudo systemctl stop drone  
roslaunch drone drone.launch
```

Вы можете выключить автозапуск ПО с помощью команды `disable` :

```
sudo systemctl disable drone
```

roslaunch

Основная документация: <http://wiki.ros.org/roslaunch>.

Список объявленных для запуска нод / программ указывается в файле

```
/home/pi/catkin_ws/src/drone/drone/launch/drone.launch .
```

Вы можете добавить собственную ноду в список автозапускаемых. Для этого разместите ваш запускаемый файл (например, `my_program.py`) в каталог

```
/home/pi/catkin_ws/src/drone/drone/src . Затем добавьте запуск вашей ноды в drone.launch , например:
```

```
<node name="my_program" pkg="drone" type="my_program.py" output="screen"/>
```

Запускаемый файл должен иметь *permission* на запуск:

```
chmod +x my_program.py
```

При использовании скриптовых языков в начале файла должен стоять **shebang**, например:

```
#!/usr/bin/env python3
```

Работа с ROS из браузера

С помощью библиотеки `roslibjs` возможна работа со всеми ресурсами ROS (топики, сервисы, параметры) из JavaScript-кода внутри браузера, что позволяет создавать различные интерактивные браузерные приложения для коптера.

Все необходимое для работы с `roslibjs` предустановлено и настроено на образе для RPi.

Пример

Пример HTML-кода страницы, работающей с `roslib.js` :

```
<html>
  <script src="js/roslib.js"></script>
  <script type="text/javascript">
    // Establish roslibjs connection
    var ros = new ROSLIB.Ros({ url: 'ws://' + location.hostname + ':9090'

    ros.on('connection', function () {
      // Connection callback
      alert('Connected');
    });

    // Declare get_telemetry service client
    var getTelemetry = new ROSLIB.Service({ ros: ros, name : '/get_telemetry

    // Call get_telemetry
    getTelemetry.callService(new ROSLIB.ServiceRequest({ frame_id: 'map' }

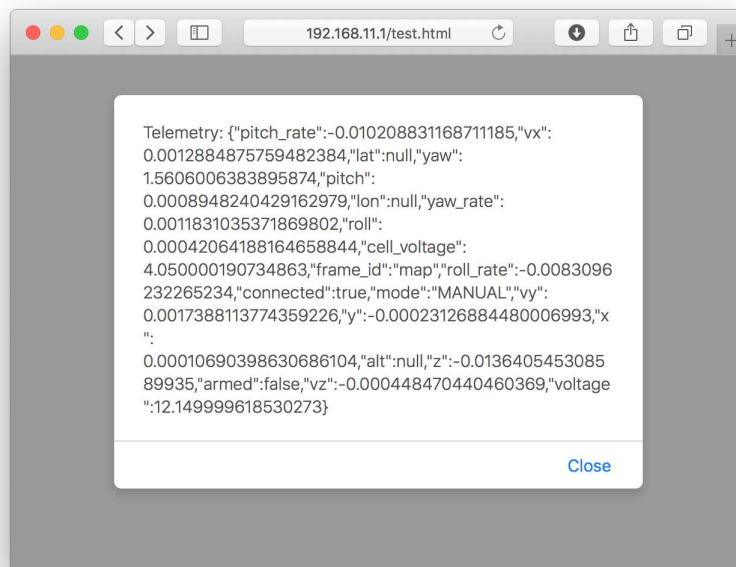
      // Service respond callback
      alert('Telemetry: ' + JSON.stringify(result));
    });

    // Subscribe to `~/mavros/state` topic
    var stateSub = new ROSLIB.Topic({ ros : ros, name : '/mavros/state', me
    stateSub.subscribe(function(msg) {
      // Topic message callback
      console.log('State: ', msg);
    });
  </script>
</html>
```

Взлет, посадка и все остальные операции могут быть осуществлены аналогичным образом.

Страница должна быть помещена в каталог `/home/pi/.ros/www/drone/`. После этого она станет доступна по адресу

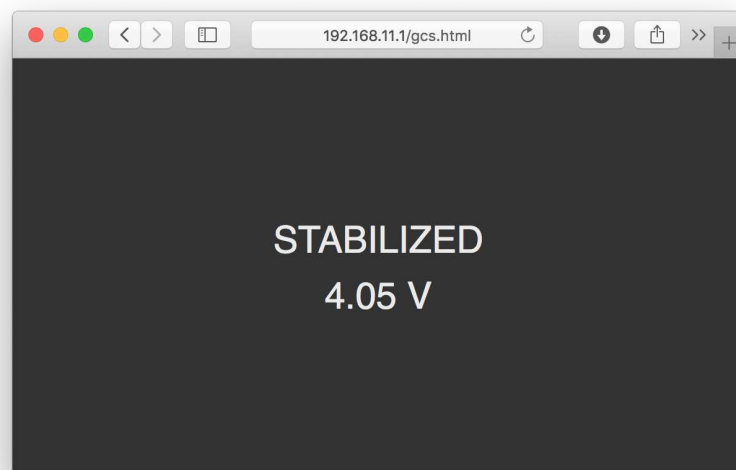
`http://192.168.11.1/drone/<имя_страницы>.html`. При открытии страницы браузер должен показать окно с телеметрией дрона, а также постоянно выводить состояние полетного контроллера в консоль.



Более подробную информацию смотрите в [туториале по roslibjs](#) .

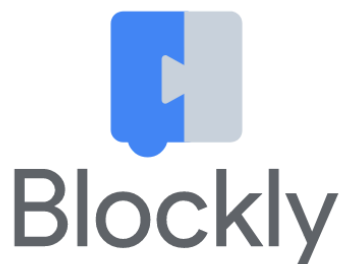
Браузерная GCS

Смотрите также пример реализации упрощенной браузерной наземной станции (GCS) по адресу <http://192.168.11.1/drone/gcs.html>.



Блочное программирование

Реализация блочного программирования основана на [Google Blockly](#). Интеграция Blockly позволяет понизить входной порог в программирование автономных полетов до минимального уровня.



Конфигурация

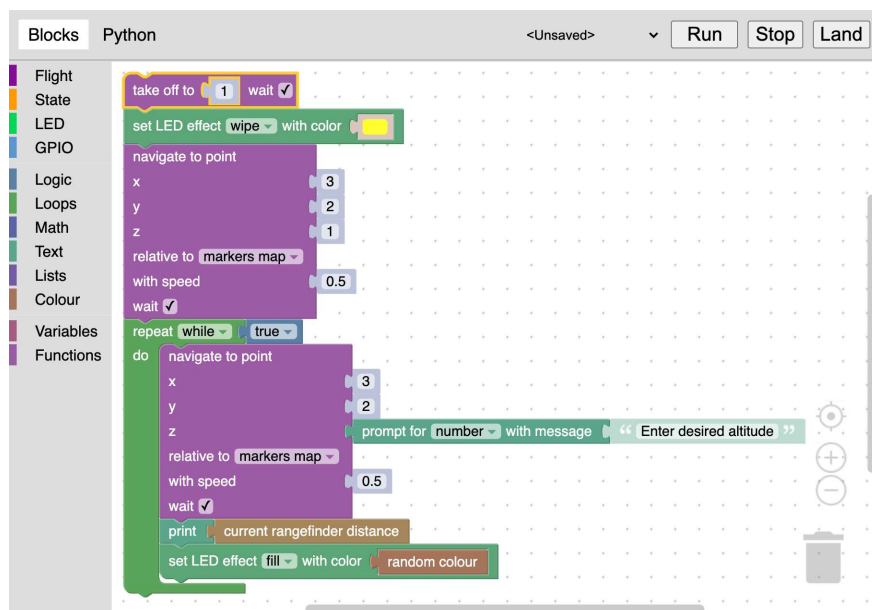
Для корректной работы работы блочного программирования аргумент `blocks` в `launch`-файле (`~/catkin_ws/src/drone/drone/launch/drone.launch`) **должен быть в значении** `true` :

```
<arg name="blocks" default="true"/>
```

Запуск

Для того, чтобы открыть интерфейс блочного программирования, [подключитесь по Wi-Fi](#) и перейдите на страницу http://192.168.11.1/drone_blocks/ либо нажмите ссылку *Blocks programming* на [основной веб-странице](#).

Интерфейс выглядит следующим образом:



Соберите необходимую программу из блоков в меню слева а затем нажмите кнопку *Run* для ее запуска. Также вы можете просмотреть сгенерированный код на языке Python, переключившись во вкладку *Python*.

Кнопка *Stop* позволяет остановить программу. Нажатие кнопки *Land* также останавливает программу и сажает дрон.

Сохранение и загрузка

Для сохранения программы откройте меню справа сверху, выберите пункт меню



Save и введите название программы. Название программы может содержать только латинские буквы, дефис, подчеркивание и точку. Все ранее сохраненные программы будут доступны в этом же меню.

На карте памяти сохраненные XML-файлы программ хранятся в каталоге `/catkin_ws/src/drone/drone_blocks/programs/`.

В этом же меню доступны примеры программ (подкаталог `examples`).

Блоки

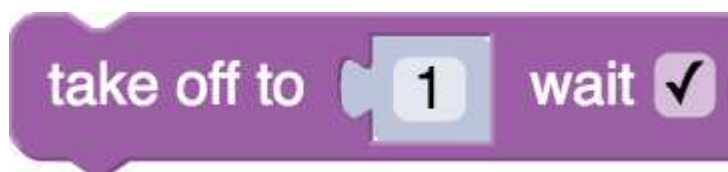
Набор блоков приблизительно аналогичен набору ROS-сервисов [API автономных полетов](#). В этом разделе приведено описание некоторых из них.

Блоки поделены на 4 категории:

- **Flight** – команды, имеющие отношение к полету.
- **State** – блоки, позволяющие получить те или иные параметры текущего состояния дрона.
- **LED** – блоки для управления [LED-лентой](#).
- **GPIO** – блоки для работы с [GPIO-пинами](#).

В остальных категориях находятся стандартные блоки Google Blockly.

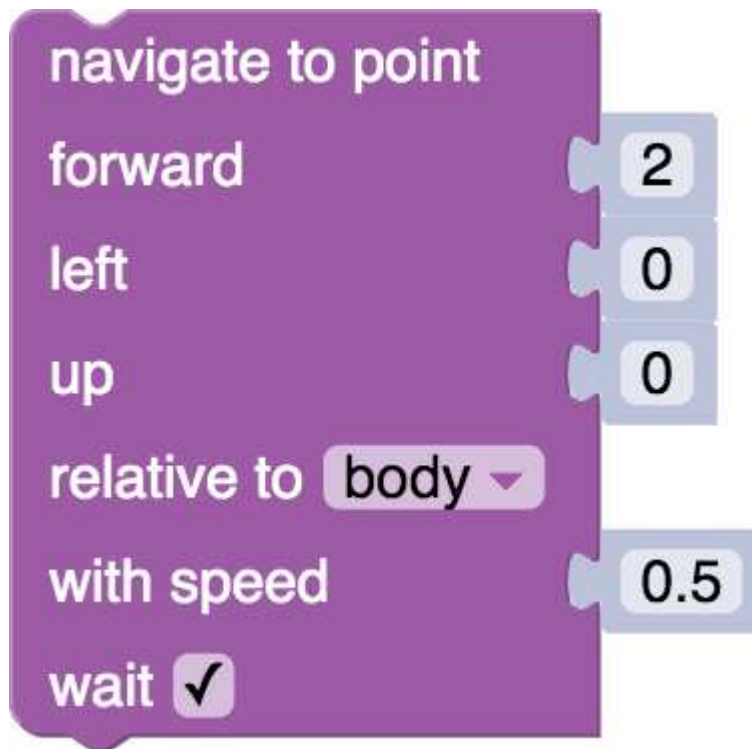
take_off



Взлететь на указанную высоту в метрах. Высота может быть произвольным блоком, возвращающим числовое значение.

Флаг `wait` определяет, должен ли дрон ожидать окончания взлета перед выполнением следующего блока.

navigate



Прилететь в заданную точку. Координаты точки задаются в метрах.

Флаг `wait` определяет, должен ли дрон ожидать завершения полета в точку перед выполнением следующего блока.

Поле *relative to*

В блоке может быть выбрана [система координат](#), в которой задана целевая точка:

- *body* – координаты относительно дрона: вперед (*forward*), влево (*left*), вверх (*up*).
- *markers map* – система координат, связанная с [картой ArUco-маркеров](#).
- *marker* – система координат, связанная с [ArUco-маркером](#); появляется поле для ввода ID маркера.
- *last navigate target* – координаты относительно последней заданной точки для навигации.
- *global* – глобальная система координат (широта и долгота) и относительная высота.
- *global, WGS 84 alt.* – глобальная система координат и высота в [системе WGS 84](#).

land



Произвести посадку.

Флаг `wait` определяет, должен ли дрон ожидать окончания посадки перед выполнением следующего блока.

wait



Ожидать заданное время в секундах. Время ожидания может быть произвольным блоком, возвращающим числовое значение.

wait_arrival



Ожидать, пока дрон долетит до целевой точки (заданной в [navigate](#)-блоке).

get_position



Блок позволяет получить позицию, скорость и угол по рысканью дрона в заданной [системе координат](#).

set_effect



Блок позволяет устанавливать различные анимации на LED-ленту аналогично [ROS-сервису set_effect](#) .

Пример использования блока для установки случайного цвета (блоки, связанные с цветами находятся в категории *Colour*):



Работа с GPIO

Категория [GPIO](#) содержит блоки для работы с GPIO. Обратите внимание, что для корректной работы этих блоков демон для работы с GPIO `pigpiod` должен быть включен:

Датчик температуры и влажности

```
sudo systemctl enable pigpiod.service  
sudo systemctl start pigpiod.service
```

Более подробную информацию о GPIO читайте в [соответствующей статье](#).

ROS

Основная документация:

<https://wiki.ros.org>.



ROS – это широко используемый фреймворк для создания сложных, распределенных робототехнических систем.

Установка

ROS уже установлен на [образе для RPi](#).

Концепции

Ноды

Основная статья: <https://wiki.ros.org/Nodes>.

ROS-нода¹ – это специальная программа (обычно написанная на Python или C++), которая взаимодействует с другими нодами посредством ROS-топиков и ROS-сервисов. Разделение сложных робототехнических систем на изолированные ноды дает определенные преимущества: понижается связанность кода, повышается переиспользуемость и надежность.

Очень многие робототехнические библиотеки и драйвера выполнены именно в виде ROS-нод.

Для того, чтобы превратить обычную программу в ROS-ноду, необходимо подключить к ней библиотеку `rospy` (Python) или `roscpp` (C++) и добавить инициализирующий код.

Пример ROS-ноды на языке Python:

```
import rospy

rospy.init_node('my_ros_node') # имя ROS-ноды

rospy.spin() # входим в бесконечный цикл...
```

Любая [программа для автономного полета](#) является ROS-нодой.

Топики

Основная статья: <https://wiki.ros.org/Topics>.

Топик – это именованная шина данных, по которой ноды обмениваются сообщениями. Любая нода может *опубликовать* сообщение в произвольный топик, а также *подписаться* на произвольный топик.

Для каждого созданного топика должен быть задан тип сообщений, которые по нему передаются. ROS включает в себя большое количество стандартных типов сообщений, покрывающих различные аспекты робототехники, но при необходимости возможно создание собственных типов сообщений. Примеры стандартных типов сообщений:

Тип сообщения	Описание
<code>std_msgs/Int64</code>	Целое число.
<code>std_msgs/Float64</code>	Число с плавающей точкой (дробное) двойной точности.
<code>std_msgs/String</code>	Строка.
<code>geometry_msgs/PoseStamped</code>	Позиция и ориентация объекта с заданной системой координат и временной меткой (широко используется для передачи текущей позиции робота и его частей).
<code>geometry_msgs/TwistStamped</code>	Линейная и угловая скорость объекта с заданной системой координат и временной меткой.
<code>sensor_msgs/Image</code>	Изображение (см. статью о работе с камерой)

Смотрите остальные стандартные типы сообщений в пакетах `common_msgs`, `std_msgs`, `geometry_msgs`, `sensor_msgs` и других.

Пример публикации сообщения типа `std_msgs/String` (строка) в топик `/foo` на языке Python:

```
import rospy
from std_msgs.msg import String

rospy.init_node('my_ros_node')

foo_pub = rospy.Publisher('/foo', String, queue_size=1) # создаем Publisher

foo_pub.publish(data='Hello, world!') # публикуем сообщение
```

Пример подписки на топик `/foo`:

```
import rospy
from std_msgs.msg import String

rospy.init_node('my_ros_node')

def foo_callback(msg):
    print(msg.data)

# Подписываемся. При получении сообщения в топик /foo будет вызвана функция foo
rospy.Subscriber('/foo', String, foo_callback)

rospy.spin() # входим в бесконечный цикл, чтобы программа не завершила работу
```

Вы можете прочитать данные из топика однократно, используя функцию

`wait_for_message` :

```
msg = rospy.wait_for_message('/foo', String, timeout=3) # ждать сообщения в т
```

Также существует возможность работы с топиками с помощью утилиты

`rostopic` . Например, с помощью следующей команды можно просматривать сообщения, публикуемые в топик `/mavros/state` :

```
rostopic echo /mavros/state
```

Команда `rostopic info` позволяет узнать тип сообщений в топике, команда

`rostopic hz` — частоту публикуемых в топике сообщений.

Сервисы

Основная статья: <https://wiki.ros.org/Services>.

Сервис – это некоторый аналог функции, которая может быть вызвана из одной ноды, а обработана в другой. У сервиса есть имя, аналогичное имени топика, и 2 типа сообщений: тип запроса и тип ответа.

Таким образом, сервисы реализуют паттерн *удаленного вызова процедур*.

Пример вызова ROS-сервиса из языка Python:

```
import rospy
from drone.srv import GetTelemetry

rospy.init_node('my_ros_node')

# Создаем обертку над сервисом get_telemetry пакета drone с типом GetTelemetry
get_telemetry = rospy.ServiceProxy('get_telemetry', srv.GetTelemetry)

# Вызываем сервис и получаем телеметрию дрона:
telemetry = get_telemetry()
```

С сервисами можно также работать при помощи утилиты `rosservice` . Так можно вызвать сервис `/get_telemetry` из командной строки:

```
rosservice call /get_telemetry "{frame_id: ''}"
```

Больше примеров использования сервисов для автономных полетов можно посмотреть в [документации ноды simple_offboard](#).

Имена

Основная статья: <https://wiki.ros.org/Names>.

Любой топик, сервис или параметр идентифицируется с помощью уникального имени. ROS-имя представляет собой иерархическую структуру с символом `/` в качестве разделителя (сходно с именами в файловой системе).

Примеры ROS-имен:

- `/` (глобальное пространство имен)
- `/foo`
- `/stanford/robot/name`
- `/wg/node1`

Эти имена являются глобальными (аналогично полному пути в файлу в файловой системе). На практике рекомендуется использование *приватных* или *относительных* имен.

Приватное имя

Каждая нода может использовать собственное приватное пространство имен (соответствующее имени ноды) для своих ресурсов. Например, нода `aruco_detect` может публиковать такие топики:

- `/aruco_detect/markers`
- `/aruco_detect/visualization`
- `/aruco_detect/debug`

Когда нода ссылается на свой приватный ресурс, вместо пространства имен (`/aruco_detect/`) используется символ `~`, например:

- `~markers`
- `~visualization`
- `~debug`

Таким образом, создание топика `foo` в приватном пространстве имен из Python будет выглядеть так:

```
private_foo_pub = rospy.Publisher('~foo', String, queue_size=1)
```

Относительное имя

Несколько нод также могут объединяться в общее пространство имен (например, при одновременной работе нескольких роботов). Для того, чтобы сослаться на топики с учетом общего пространства имен, в названии ресурса опускается начальный символ `/`.

Пример создание топика `foo` с учетом общего пространства имен:

```
relative_foo_pub = rospy.Publisher('foo', String, queue_size=1)
```

В общем случае всегда рекомендуется использовать приватные или относительные имена ресурсов и никогда не использовать глобальные.

Работа на нескольких машинах

Основная статья: <https://wiki.ros.org/ROS/Tutorials/MultipleMachines>.

Преимуществом использования ROS является возможность распределения нод на несколько машин в сети. Например, ноду, осуществляющую распознавание образов на изображении можно запустить на более мощном компьютере; ноду, управляющую коптером можно запустить непосредственно на Raspberrу Pi, подключенном к полетному контроллеру и т. д.

Дополнительные материалы

- Учебник по ROS от Voltbro - <http://docs.voltbro.ru/starting-ros/>.
- Другие книги по ROS - <https://wiki.ros.org/Books>.

¹. Также встречается перевод "узел". ↩

Дополнительные материалы

Этот раздел содержит статьи по той или иной проблематике, связанной с БПЛА.

Работа со светодиодной лентой

На образе [для RPi](#) предустановлены необходимые модули для работы с лентой. Они позволяют:

- управлять эффектами/анимациями на ленте;
- управлять лентой на низком уровне (переключением цветов отдельных светодиодов);
- настраивать реакцию ленты на полетные события.

Обратите внимание, что светодиодную ленту нужно питать от стабильного источника энергии. Если вы подключите питание напрямую к Raspberry, то это создаст слишком большую нагрузку на ваш микрокомпьютер. Для снятия нагрузки с Raspberry можно подключить питание к преобразователю ВЕС.

Высокоуровневое управление лентой

1. Для работы с лентой подключите ее к питанию +5v – 5v, земле GND – GND и сигнальному порту DIN – GPIO21.
2. Включите поддержку LED-ленты в файле

```
~/catkin_ws/src/drone/drone/launch/drone.launch :
```

```
<arg name="led" default="true"/>
```

3. Настройте параметры подключения ленты ws281x в файле

```
~/catkin_ws/src/drone/drone/launch/led.launch . Необходимо ввести верное количество светодиодов в ленте и GPIO-пин, использованный для подключения (если он отличается от GPIO21):
```

```
<arg name="led_count" default="58"/> <!-- количество светодиодов в ленте  
<arg name="gpio_pin" default="21"/> <!-- GPIO-пин для подключения -->
```

Высокоуровневое управления лентой позволяет управлять текущим эффектом (анимацией) на ленте. Для этого используется ROS-сервис

```
/led/set_effect . Параметры сервиса:
```

- `effect` – название необходимого эффекта.
- `r`, `g`, `b` – цвет эффекта в формате **RGB**. Значения изменяются от 0 до 255.

Список доступных эффектов:

- `fill` (или пустая строка) – залить всю ленту цветом;
- `blink` – мигание цветом;
- `blink_fast` – ускоренное мигание цветом;
- `fade` – плавное перетекание в цвет;

- `wipe` – "надвигание" нового цвета;
- `flash` – быстро мигнуть цветом 2 раза и вернуться к предыдущему эффекту;
- `rainbow` – переливание ленты цветами радуги;
- `rainbow_fill` – переливать заливку по цветам радуги.

Пример работы с сервисом из Python:

```
import rospy
from drone.srv import SetLEDEffect

rospy.init_node('flight')

set_effect = rospy.ServiceProxy('/led/set_effect', SetLEDEffect) # define proxy

set_effect(r=255, g=0, b=0) # fill strip with red color
rospy.sleep(2)

set_effect(r=0, g=100, b=0) # fill strip with green color
rospy.sleep(2)

set_effect(effect='fade', r=0, g=0, b=255) # fade to blue color
rospy.sleep(2)

set_effect(effect='flash', r=255, g=0, b=0) # flash twice with red color
rospy.sleep(5)

set_effect(effect='blink', r=255, g=255, b=255) # blink with white color
rospy.sleep(5)

set_effect(effect='rainbow') # show rainbow
```

Также лентой можно управлять из командной строки (Bash):

```
rosservice call /led/set_effect "{effect: 'fade', r: 0, g: 0, b: 255}"
```

```
rosservice call /led/set_effect "{effect: 'rainbow'"}
```

Настройка реакции ленты на события

Можно показывать LED-лентой текущее состояние полетного контроллера и сигнализировать о событиях. Данная функция настраивается в файле

`~/catkin_ws/src/drone/drone/launch/led.launch` в разделе *events effects table*.

Пример настройки:

```
startup: { r: 255, g: 255, b: 255 }
connected: { effect: rainbow }
disconnected: { effect: blink, r: 255, g: 50, b: 50 }
<!-- ... -->
```

В левой части таблицы указывается событие, на которая лента должна среагировать. В правой части указывается эффект (анимация), который необходимо включить при возникновении события.

Список поддерживаемых событий:

Событие	Описание	Эффект по умолчанию
startup	Запуск всех систем	Белый
connected	Успешное подключение к полетному контроллеру	Эффект радуги
disconnected	Разрыв связи с полетным контроллером	Мигание красным
armed	Переход в состояние Armed	
disarmed	Переход в состояние Disarmed	
acro	Режим Acro	Оранжевый
stabilized	Режим Stabilized	Зеленый
altctl	Режим Altitude	Желтый
posctl	Режим Position	Синий
offboard	Режим Offboard	Фиолетовый
rattitude , mission , rtl , land	Переход в соответствующие режимы	
error	Возникновение ошибки в ROS-нодах или полетном контроллере (<i>ERROR</i> -сообщение в топике /rosout)	Мигнуть красным
low_battery	Низкий заряд батареи (порог настраивается в параметре threshold)	Быстрое мигание красным

Для корректной работы сигнализации LED-лентой о низком заряде батареи необходимо корректная [калибровка электропитания](#).

Для того, чтобы отключить реакцию светодиодной ленты на события, установите аргумент `led_notify` в файле

`~/catkin_ws/src/drone/drone/launch/led.launch` в значение `false` :

```
<arg name="led_notify" default="false"/>
```

Низкоуровневое управление лентой

Для управления отдельными светодиодами используется ROS-сервис `/led/set_leds`. В параметрах задается массив номеров и RGB-цветов светодиодов, которые необходимо переключить.

Пример работы с сервисом из Python:

```
import rospy
from led_msgs.srv import SetLEDs
from led_msgs.msg import LEDStateArray, LEDState

rospy.init_node('flight')

set_leds = rospy.ServiceProxy('/led/set_leds', SetLEDs) # define proxy to ROS s

# switch LEDs number 0, 1 and 2 to red, green and blue color:
set_leds([LEDState(0, 255, 0, 0), LEDState(1, 0, 255, 0), LEDState(2, 0, 0, 255)
```

Сервис можно использовать из командной строки:

```
rosservice call /led/set_leds "leds:
- index: 0
  r: 50
  g: 100
  b: 200"
```

При использовании ленты в ROS-топике `/led/state` публикуется текущие цвета светодиодов. Просмотр топика из командной строки:

```
rostopic echo /led/state
```

Используя этот же топик можно получить общее выставленное в настройках количество светодиодов:

```
led_count = len(rospy.wait_for_message('/led/state', LEDStateArray, timeout=10))
```

Использование радиотелеметрии

Радиотелеметрический приемник позволяет выполнять подключение к полетному контроллеру без использования USB-кабеля или Wi-Fi. Это особенно полезно при полетах с GPS в режиме автономных миссий, так как дальность действия радиотелеметрии намного выше, чем у Wi-Fi.

Подключение радиотелеметрии

Комплект радиотелеметрии состоит из двух модулей, один из которых подключается к компьютеру, а другой – непосредственно к полетному контроллеру.



Перед использованием радиотелеметрии убедитесь, что на обоих модулях установлены антенны. Использование модуля без антенны может привести к выходу его из строя.

Подключите один из модулей к разъему "Telem 1" полетного контроллера.



Второй модуль подключите к компьютеру при помощи кабеля Micro-USB. Непрерывно горящий зеленый светодиод означает что связь между модулям установлена.

Моргающий красный светодиод означает, что между модулями передаются данные.

Откройте QGroundControl. Если всё было подключено верно, связь с дроном должна установиться без каких-либо дополнительных действий.

Пошаговая инструкция по настройке автономного полета

Данная инструкция содержит ссылки на другие статьи, в которых каждая из затронутых тем разобрана более подробно. Если вы столкнулись с трудностями во время прочтения одной из таких статей, рекомендуется вернуться к данной инструкции, так как здесь многие операции описаны пошагово, а также отсутствуют ненужные шаги.

Первоначальная настройка Raspberry Pi

- Скачайте образ системы по [ссылке](#).
- Запишите образ на MicroSD карту.
- Вставьте карту в Raspberry Pi.
- Установите Raspberry Pi и камеру на дрон.
- Подключите питание к Raspberry Pi и ожидайте появления Wi-Fi-сети. Для этого подключите Raspberry Pi к компьютеру через MicroUSB-кабель. На Raspberry Pi должен периодически мигать зеленый светодиод. Он сигнализирует о нормальной работе операционной системы.

Перед подключением Raspberry Pi к компьютеру по USB необходимо вытащить из Raspberry Pi провод питания (который идет от BEC). Иначе могут быть проблемы с питанием.

- Подключитесь к Wi-Fi и зайдите в веб-интерфейс ([статья](#)).

Во время первого включения сеть появляется не сразу. Нужно дождаться полной загрузки системы. Если в списке сетей долго не появляется нужной сети, закройте окно с выбором сети и откройте снова. Тогда список сетей обновится.

- Подключитесь к Raspberry Pi по SSH.

Самый быстрый способ – веб-доступ. Следуйте инструкциям в статье "[Доступ по SSH](#)".

- Если необходимо, можно поменять название и пароль сети. См. статью "[Настройка сети](#)". Остальные операции с сетью производить не нужно.
- Для редактирования файлов пользуйтесь редактором nano. [Инструкция по работе с редактором](#).

В редакторе перемещать курсор можно только стрелками на клавиатуре.

- Перезагрузите Raspberry Pi:

```
sudo reboot
```

Соединение временно закрывается, создается новая сеть. К ней надо подключиться заново.

- Убедитесь в корректной работе камеры. В браузере зайдите на адрес <http://192.168.11.1:8080> и выберите `image_raw`.

Более подробно можно прочитать в статье "[Просмотр изображений с камер](#)".

Если изображение размыто, необходимо сфокусировать линзу. Для этого покрутите объектив в одну или в другую сторону. Продолжайте крутить, пока изображение не станет четким.

На камере должен гореть красный светодиод: он означает, что камера в данный момент производит съемку. Если светодиод не горит: либо камера подключена неправильно, либо операционная система не загрузилась, либо в настройках допущена ошибка.

Базовые команды

Вам пригодятся основные команды Linux, а также специальные команды, чтобы уверенно работать в системе.

Показать список файлов:

```
ls
```

Перейти в папку с прописыванием пути к ней:

```
cd catkin_ws/src/drone/drone/launch/
```

Перейти в домашнюю директорию:

```
cd
```

Открыть файл `file.py`:

```
nano file.py
```

Открыть файл `drone.launch` с прописыванием полного пути к нему (сработает, если вы находитесь в другой папке):

```
nano ~/catkin_ws/src/drone/drone/launch/drone.launch
```

Сохранить файл (нажимать последовательно):


```
Ctrl+X; Y; Enter
```

Удалить файл или папку с названием name (ВНИМАНИЕ: операция выполнится без подтверждения. Будьте осторожны!):

```
rm -rf name
```

Создать папку с названием myfolder:

```
mkdir myfolder
```

Полная перезагрузка Raspberry Pi:

```
sudo reboot
```

Перезапуск только систем:

```
sudo systemctl restart drone
```

Выполнить самопроверку:

```
rosvun drone selfcheck.py
```

Остановить программу

```
ctrl+c
```

Запустить программу myprogram на Питоне:

```
python3 myprogram.py
```

Журнал событий процессов. Пролитывать список можно нажатием Enter или сочетанием клавиш Ctrl+V (пролистывает быстрее):

```
journalctl -u drone
```

Открыть файл sudoers от имени администратора (он не откроется без прописывания sudo. Через sudo можно запускать другие команды, если они не открываются без прав администратора):

```
sudo nano /etc/sudoers
```

Настройка параметров Raspberry Pi для автономного полета

Большинство параметров, необходимых для полета, хранится в папке

```
~/catkin_ws/src/drone/drone/launch/ .
```

- Зайти в папку:

```
cd ~/catkin_ws/src/drone/drone/launch/
```

Символ `~` обозначает домашнюю директорию вашего пользователя.

Если вы уже находитесь в ней, можно обойтись командой: `cd catkin_ws/src/drone/drone/launch/`

Клавишей `Tab` можно автоматически дополнить названия файлов, папок или команд. Нужно начать вводить желаемое название и нажать `Tab`. Если не будет конфликтов, название напишется полностью. Например, чтобы быстро ввести путь к папке с настройками, после ввода `cd` можно начать вводить следующую комбинацию клавиш: `c-Tab-s-Tab-c-Tab-c-Tab-l-Tab`. Таким образом можно сэкономить много времени при написании длинной команды, а также избежать возможных ошибок в написании пути.

- В этой папке необходимо сконфигурировать несколько файлов:

- `drone.launch`
- `aruco.launch`
- `main_camera.launch`

- Открыть файл `drone.launch` :

```
nano drone.launch
```

Вы должны находиться в папке, в которой располагается файл. Если вы находитесь в другой папке, файл можно открыть, прописав полный путь к нему:

```
nano ~/catkin_ws/src/drone/drone/launch/drone.launch
```

Если файл одновременно редактируют два пользователя, а также если в прошлый раз закрытие файла произошло некорректно, программа `nano` не отобразит файл сразу, а попросит дополнительное разрешение. Для этого нужно нажать клавишу `Y`.

Если содержимое файла все равно пусто, возможно, вы неверно ввели имя файла. Нужно обращать внимание на расширение и вписывать его полностью. Если вы вписали неверное имя или расширение, программа `nano` создаст пустой файл с этим названием, что нежелательно. Такой файл следует удалить.

- В файле `drone.launch` найти строчку:

```
<arg name="aruco" default="false"/>
```

и заменить `false` на `true` :

```
<arg name="aruco" default="true"/>.
```

Это активирует модуль распознавания ArUco-маркеров.

- Откройте файл `aruco.launch`.
- В нем нужно активировать несколько параметров. Подробнее в [статье](#).

Должно получиться:

```
<arg name="aruco_detect" default="true"/>
<arg name="aruco_map" default="true"/>
<arg name="aruco_vpe" default="true"/>
```

- Сгенерируйте поле с метками. Смотрите подробности в статье [Навигация по картам ArUco-маркеров](#). Для генерации меток нужно ввести команду с определенными значениями.

Пример команды для генерации поля, где:

- длина маркера = 0.335 м (`length`)
- 10 столбцов (`x`)
- 10 строк (`y`)
- расстояние между центрами меток по оси $x = 1$ м (`dist_x`)
- расстояние между центрами меток по оси $y = 1$ м (`dist_y`)
- номер первого маркера = 0 (`first`)
- название карты остается стандартным: `map.txt`
- нумерация идет с верхнего левого угла (ключ `--top-left`)

```
roslaunch aruco_pose genmap.py 0.335 10 10 1 1 0 > ~/catkin_ws/src/drone/aruc
```

В большинстве полей нумерация начинается с нулевой метки. Также в большинстве случаев нумерация начинается с верхнего левого угла, поэтому при генерации очень важно указывать ключ `--top-left`.

Если вы зададите другое имя для файла с картой, его нужно прописать в файле `aruco.launch`. Найдите строку `<param name="map" value="$(find aruco_pose)/map/map.txt"/>` и замените название `map.txt` на название вашего файла.

- Отредактируйте файл `main_camera.launch` для настройки камеры:

Подробнее в статье "[Настройка расположения основной камеры](#)".

В этом файле необходимо отредактировать строку с параметрами расположения камеры. Строка выглядит так:

```
<node pkg="tf2_ros" type="static_transform_publisher" name="main_camera_fr
```

В файле вы найдете много строк, похожих на эту, но большинство из них закомментированы (то есть не читаются) и только одна раскомментирована. Это заранее заготовленные настройки, из которых можно выбрать нужную вам.

Комментарий в языке XML — это символы `<!--` в начале строки и `-->` в конце строки. Пример закомментированной строки:

```
<!--<node pkg="tf2_ros" type="static_transform_publisher" name="main_camer
```

Пример незакомментированной строки (строка будет учитываться программой):

```
<node pkg="tf2_ros" type="static_transform_publisher" name="main_camera_fr
```

Над этими строками написано, какому расположению камеры соответствует настройка. Если шлейф от камеры выходит вперед относительно дрона, а камера направлена вниз, нужно выбрать настройку:

```
<!-- camera is oriented downward, camera cable goes forward [option 2] --
```

Чтобы выбрать нужную настройку, необходимо раскомментировать соответствующую строку, и закомментировать другую аналогичную строку, чтобы не возникло конфликтов.

- Сохранить изменения. Последовательно нажмите:

```
Ctrl+x; y; Enter
```

- Перезагрузите модуль:

```
sudo systemctl restart drone
```

Соединение полетного контроллера и Raspberry Pi

- Соедините Raspberry Pi и полётный контроллер. Кабель должен быть аккуратно плотно закручен и пропущен снизу дрона, чтобы не попасть в пропеллеры.
- Удаленно подключитесь к полётному контроллеру через QGroundControl. В системе уже выставлены нужные настройки, остается лишь создать новое подключение в QGroundControl, выбрать его и подключиться. Настраивается оно, как на картинке в статье "[Подключение QGroundControl по Wi-Fi](#)".

Настройка пульта

- Настройка полетных режимов описана в статье ["Полетные режимы"](#).

Канал 5 должен располагаться на переключателе SwC; Канал 6 - на SwA. Однако вы можете настроить эти каналы любым удобным для вас образом.

Выполнение автоматической проверки

Проверку следует выполнить, когда вы полностью настроили дрон, а также при возникновении неполадок. Подробно процедура описана в статье ["Автоматическая проверка"](#).

- Выполнить команду:

```
rosvun drone selfcheck.py
```

Написание программы

В статье ["Автономный полет"](#) описана работа с модулем `simple_offboard`, который создан для простого программирования дрона. В ней даны описания основных функций, а также примеры кода.

- Скопируйте из раздела "Использование из языка Python" пример кода и вставьте в редактор (например, в Visual Studio Code, PyCharm, Sublime Text, Notepad++).
- Сохраните документ с расширением `.py` для включения подсветки текста.
- Далее необходимо добавить полётные команды в программу. Примеры таких команд представлены в статье. Нужно написать функции для взлета и полета в точку, а также для посадки.
- Взлет.

Для взлета можно использовать функцию `navigate` :

```
navigate(x=0, y=0, z=1.5, speed=0.5, frame_id='body', auto_arm=True)
```

Добавьте эту строку внизу программы.

Также добавьте команду ожидания:

```
rospy.sleep(3)
```

Важно выделить время на выполнение команды `navigate`, иначе дрон, не дожидаясь выполнения предыдущей команды, сразу перейдет к выполнению следующей. Для этого используется команда `rospy.sleep()`. В скобках указывается время в секундах. Функция `rospy.sleep()` относится к предыдущей команде `navigate`, а не к последующей, то есть это время, которое мы даем на то, чтобы долететь до точки, обозначенной в предыдущем `navigate`.

- Зафиксировать положение дрона в системе координат маркерного поля.

Для этого нужно выполнить `navigate` и указать в нем необходимые координаты (например, $x=1, y=1, z=1.5$) и выбрать систему координат (`frame_id`):

```
navigate(x=1, y=1, z=1.5, speed=1, frame_id='aruco_map')
```

- В итоге должно получиться:

```
navigate(x=0, y=0, z=1.5, speed=0.5, frame_id='body', auto_arm=True)
rospy.sleep(3)
navigate(x=1, y=1, z=1.5, speed=1, frame_id='aruco_map')
```

Обратите внимание, что параметр `auto_arm=True` ставится только при первом взлете. В остальных случаях его выставлять нельзя, иначе возникнут проблемы с перехватом управления.

- Если вы хотите добавить другие точки для пролета, нужно дописать еще один `navigate` и `rospy.sleep()`. Время нужно вычислить отдельно для каждой точки в зависимости от скорости полета и расстояния между точками.

Например, если мы хотим полететь в точку (3, 3, 1.5):

```
navigate(x=3, y=3, z=1.5, speed=1, frame_id='aruco_map')
rospy.sleep(3)
```

Координаты не должны выходить за пределы вашего поля. Если поле имеет размер 4x4 метра, максимальное значение координат, которое стоит указывать, — 4.

- После пролета по точкам нужно приземлиться. Следующая строка ставится в конце программы:

```
land()
```

Запись программы на дрон

Самый простой способ – это скопировать текст программы, создать новый файл в командной строке и вставить текст программы в файл.

- Для создания файла `myprogram.py` введите команду:

```
nano myprogram.py
```

Название можно выбрать любое, однако не рекомендуется использовать пробелы и специальные символы. Также расширение у программы всегда должно быть `.py`.

- Вставить текст в поле ввода. Если вы пользуетесь веб-доступом Butterfly на Windows или Linux:

```
Ctrl+Shift+V
```

На Mac нажмите `cmd+v`.

- Сохранить файл:

```
Ctrl+x; Y; Enter
```

Запуск программы

- Необходимо тщательно подготовить дрон, пульт и программу. Запустите `selfcheck.py`. Убедитесь, что дрон летает в ручном режиме.
- Включите дрон и дождитесь, пока загрузится система. Красный огонек на камере означает, что систем загрузилась.
- Проверьте полет в режиме POSCTL.

Для этого взлетите над метками в режиме STABILIZED и переведите переключатель SwC в нижнее положение - режим POSCTL.

Будьте готовы сразу же переключиться обратно в режим STABILIZED в случае выхода дрона из-под контроля!

Установите левый стик (газ) в центральное положение. Дрон должен зависнуть на месте. В таком случае можно посадить дрон и переходить к следующему шагу. Если нет, нужно разобраться в проблеме.

- Установите переключатель SwC в центральное положение. С помощью него вы будете перехватывать дрон: стоит лишь переключить его в верхнее положение.
- Установите левый стик (газ) в центральное положение, чтобы в случае перехвата дрон не упал на пол.
- Запустите программу. Для этого выполните команду:

```
python3 my_program.py
```

Датчик температуры и влажности

После выполнения программы дрон может некорректно приземлиться и продолжать лететь над полом. В таком случае нужно перехватить управление.

Имя хоста

По умолчанию установлено имя хоста (hostname) `drone-xxxx`, где `xxxx` – случайные цифры. Имя хоста соответствует SSID точки доступа Wi-Fi.

Таким образом, дрон доступен на машинах, поддерживающих mDNS, под именем `drone-xxxx.local`. Вы можете использовать это имя для SSH-доступа:

```
ssh pi@drone-xxxx.local
```

Также это имя может быть использовано вместо IP-адреса для открытия страницы в браузере и т. д.

Изменение имени хоста

В некоторых ситуациях необходимо изменение имени хоста. Для это используйте утилиту `hostnamectl`:

```
sudo hostnamectl set-hostname newname
```

Где `newname` – новое имя машины. Утилита `hostnamectl` меняет имя в файле `/etc/hostname`.

Также необходимо прописывание нового имени в файл `/etc/hosts`:

```
127.0.1.1    newname newname.local
```

Прописывание `newname.local` необходимо, чтобы ROS смог разрешить это имя в ситуациях, когда все сетевые интерфейсы неактивны (отключение/разрыв связи Wi-Fi).

Изменение имени хоста не повлечёт за собой изменений SSID точки доступа Wi-Fi (и наоборот, изменение SSID точки доступа не меняет имя хоста).

Настройка PID регуляторов

В этой статье описаны методы и основные технологии настройки каскадного ПИД-регулятора. Приведенные советы и методики подходят для любых видов рам (Квадрокоптеров, Гексакоптеров, Октокоптеров и т.д.).

Усредненные рекомендованные настройки приведены в статье "[Первоначальная настройка](#)".

Эта статья только для продвинутых пользователей. Плохая или неполная настройка ПИД регуляторов может привести к сильным поломкам вашего дрона.

Введение

Пропорционально-Интегрально-Дифференцирующий (ПИД) регулятор является одним из самых распространенных методов управления.

Шаги настройки

В случае, если у вас нет опыта настройки ПИД регуляторов, придерживайтесь представленных ниже особенностей настройки.

- Все изменения значений регулятора должны быть постепенными, в противном случае могут появиться сильные осцилляции и вы совершенно потеряете контроль над аппаратом. Увеличивайте значения регулятора не более чем на 25-30% от настоящего значения, уменьшайте на 5-10% до окончательной настройки.
- Сажайте аппарат перед следующей итерацией смены коэффициентов. Перед следующим взлетом очень медленно увеличивайте стик газа для проверки аппарата на сильные осцилляции.

Регулятор угловых скоростей

Регулятор угловых скоростей является самым низкоуровневым в каскаде, он состоит из трех независимых ПИД-регуляторов, управляющих угловыми скоростями коптера по трем осям (крен, тангаж, рысканье).

- Регулятор угловых скоростей по крену (`MC_ROLLRATE_P` , `MC_ROLLRATE_I` , `MC_ROLLRATE_D`)
- Регулятор угловых скоростей по тангажу (`MC_PITCHRATE_P` , `MC_PITCHRATE_I` , `MC_PITCHRATE_D`)
- Регулятор угловых скоростей по рысканью (`MC_YAWRATE_P` , `MC_YAWRATE_I` , `MC_YAWRATE_D`)

Хорошая настройка коэффициентов регулятора угловых скоростей очень важна и влияет на все режимы полета. В то время как плохая настройка регуляторов будет заметна во всех режимах полета, к примеру в режиме Position вы можете видеть подергивания во время зависания аппарата.

Регулятор угловых скоростей можно настраивать как в режиме ACRO, так и в режиме STABILIZED.

Предпочтительнее настраивать регуляторы в режиме ACRO, поскольку вам будет легче визуально заметить произведенные изменения коэффициентов. Если вы собираетесь использовать этот режим отключите Expo-параметры и снизьте чувствительность стиков для облегчения управления.

- `MC_ACRO_EXPO = 0`, `MC_ACRO_EXPO_Y = 0`, `MC_ACRO_SUPEXPO = 0`,
`MC_ACRO_SUPEXPOY = 0`
- `MC_ACRO_P_MAX = 200`, `MC_ACRO_R_MAX = 200`
- `MC_ACRO_Y_MAX = 100`

Режим STABILIZED легче для управления но также в нем вам будет сложнее заметить изменения поведения вашего аппарата при настройке коэффициентов.

Если ваш аппарат вообще не летает обратите внимание на две основные вещи:

- Если вы видите сильные осцилляции при попытке взлета, уменьшайте все коэффициенты *P* и *D* до тех пор, пока аппарат не поднимется в воздух.
- С другой стороны, если аппарат почти не реагирует на управление передаваемое с пульта, увеличивайте коэффициент *P*.

Концепция настройки регуляторов одинакова как в режиме STABILIZED, так и в режиме ACRO. Итеративно с указанным шагом настраиваете коэффициенты *P* и *D* для крена и тангажа, а затем изменяете *I*.

Первоначально вы можете использовать одинаковые значения для крена, когда регуляторы настроены достаточно хорошо вы можете точно донстроить их по крену и тангажу отдельно (если ваш аппарат симметричен, можете оставить коэффициенты одинаковыми). Идея настройки регулятора рыскания идентична настройке крена и тангажа, за исключением того, что коэффициент *D* может оставаться нулевым.

Настройка коэффициента *P*

Коэффициент *P* (пропорциональный) используется для минимизации ошибки отслеживания и отвечает за скорость отклика, по этому должен быть установлен как можно выше, но без осцилляций.

При настройке коэффициента *P* пользуйтесь двумя основными наблюдениями:

- Если *P* слишком большой: вы увидите высокочастотные осцилляции.

- Если P слишком маленький:
 1. Аппарат медленно реагирует на входящее управление
 2. В режиме ACRO аппарат будет постоянно дрейфовать и вам нужно будет его корректировать, чтобы сохранить его уровень.

Настройка коэффициента D

Коэффициент D (дифференциальный) используется для демпфирования. Этот коэффициент должен быть как можно выше, но таким образом, чтобы не было "перестрелов" по управлению.

При настройке коэффициента D пользуйтесь двумя основными наблюдениями:

- Если D слишком большой: моторы могут подергиваться и сильно нагреваться во время полета, поскольку коэффициент D увеличивает шумы управления.
- Если D слишком маленький: возникнут "перестрелы" по входящему управляющему сигналу.

Настройка коэффициента I

Коэффициент I сохраняет "воспоминания" об ошибке. Это значит, что элемент I увеличивается в случае, если желаемая скорость не устанавливается в течении некоторого времени. Этот параметр важен для режима ACRO, а также оказывает достаточно сильное влияние на режимы POSITION и OFFBOARD.

- Если I слишком большой: вы можете увидеть медленные осцилляции
- Если I слишком маленький: можно заметить ошибку по выполнению управляющего воздействия. Также заниженный коэффициент I заметен на логах, это характеризуется тем, что на графиках желаемая скорость длительное время отличается от фактической.

Процедура тестирования

После изменения коэффициентов регулятора, для того, чтобы протестировать новые значения, подайте на вход аппарата быстрый ступенчатый ввод. Для этого быстро наклоните стик радиоаппаратуры в сторону, при этом коптер точно должен выполнять переданное управление, без осцилляций и "перестреливания". Поскольку обычно стики радиоаппаратуры подпружинены, в случае если их отпустить, они начинают колебаться, хорошо настроенный аппарат будет колебаться вместе со стиком.

Конфигурация логгера

Для получения более полной информации с графиков вы можете настроить логгер удобным вам образом. Для его настройки вы можете пользоваться параметрами:

- `SDLOG_PROFILE` - включение большого количества функций приводит к увеличению размера файла лога, а также к увеличению требований по скорости записи, перед началом работы убедитесь, что используете накопитель с достаточной пропускной способностью.
 1. `default set` - запись общих логов системы
 2. `estimator replay (EKF2)` - более подробное логирование при использовании EKF2
 3. `thermal calibration` - высокочастотные данные с IMU и барометра
 4. `system identification` - высокочастотные данные приводов и IMU
 5. `high rate` - высокочастотные данные радио, угловых скоростей и приводов
 6. `debug` - для записи пользовательских отладочных топиков
 7. `sensor comparison` - низкочастотные данные IMU, барометра и компаса, для сравнения показаний датчиков
- `SDLOG_MODE`
 1. `when armed until disarm` - лог записывается с момента армации копитера, до момента дизарма копитера
 2. `from boot until disarm` - лог записывается с момента запуска системы, до момента дизарма копитера
 3. `from boot until shutdown` - лог записывается с момента запуска системы, до момента выключения системы

Регулятор положения

Данный регулятор является вторым уровнем каскада и настраивается после регуляторов угловых скоростей. Он отвечает за угол наклона копитера и настраивается с помощью параметров:

- Регулятор угла по крену (`MC_ROLL_P`).
- Регулятор угла по тангажу (`MC_PITCH_P`).
- Регулятор угла по рысканью (`MC_YAW_P`).

В данном регуляторе настраиваемыми являются только пропорциональные значения и в большинстве случаев их стоит оставить со стандартными значениями.

Идея настройки данных параметров точно такая же, как и настройки P параметров регуляторов угловых скоростей. При высокочастотных осцилляциях уменьшите значение, при медленном выполнении команды увеличьте его.

Навигация по вертикальным ArUco-маркерам

Алгоритм навигации по визуальным ArUco-маркерам, реализованный в образе, поддерживает гибкую настройку положения маркеров в пространстве, что позволяет располагать их на любой поверхности, под любым углом.

Установка вертикального крепления камеры

Для более точного распознавания маркеров необходимо установить камеру вертикально таким образом, чтобы объектив был направлен параллельно горизонту.

Конфигурационный файл позволяет настраивать расположение камеры в пространстве относительно коптера любым образом. Для удобства далее будет рассматриваться вариант установки камеры под 90° к горизонту, по направлению носа коптера.

Настройка расположения камеры

Чтобы задать расположение камеры под необходимым углом, откройте файл

`main_camera.launch`, расположенный в `~/catkin_ws/src/drone/drone/launch/`.

```
nano ~/catkin_ws/src/drone/drone/launch/main_camera.launch
```

Версии 0.20+

В параметрах `direction_x`, `direction_y` установите пустые значения вручную или введите строки:

```
sed -i "/direction_z/s/default=\\.*/default=\\\"\\\"/" /home/pi/catkin_ws/src/drone/drone/launch/main_camera.launch
sed -i "/direction_y/s/default=\\.*/default=\\\"\\\"/" /home/pi/catkin_ws/src/drone/drone/launch/main_camera.launch
```

Отредактируйте одну из конфигурационных строк или добавьте строку, представленную ниже:

```
<node pkg="tf2_ros" type="static_transform_publisher" name="main_camera_frame"
```

Одновременно может использоваться только одна конфигурация камеры — если вы вставляете представленную выше строку, не забудьте закомментировать активную на данный момент. Для определения этого вам поможет подсветка синтаксиса — активная строка будет подсвечена отличным от комментариев цветом. Для комментирования в начало и конец строки добавьте символы `<!--` и `-->` соответственно.

Если на используемой вами карте, маркеры имеют равное расстояние по осям X и Y, можете воспользоваться [утилитой для создания карт](#) `gen_map.py`. В ином случае, вам потребуется задать их вручную — для этого перейдите в директорию `~/catkin_ws/src/drone/aruco_map/map` и создайте файл карты `map_name.txt`. Заполните вашу карту в соответствии с [синтаксисом карт](#). Пример карты маркеров со случайным расположением маркеров:

При введении карты выберите один из маркеров в качестве начала координат и относительно него отмеряйте расстояние до всех остальных маркеров. Если все ваши маркеры ориентированы одинаково, вы можете не указывать все 8 параметров, а указать только первые 5: индекс маркера, размер и его расположение в пространстве по осям x, y, z соответственно.

```
106 0.33 0 0 0
103 0.33 1.53 0.23 0
153 0.40 -0.56 1.36 0
```

После того, как вы заполните карту, необходимо применить ее — для этого отредактируйте файл `aruco.launch`, расположенный в `~/catkin_ws/src/drone/drone/launch/`. Измените в нем строку `<param name="map" value="$(find aruco_pose)/map/map_name.txt"/>`, где `map_name.txt` название вашего файла с картой.

При использовании маркеров, не привязанных к горизонтальным плоскостям (пол, потолок), необходимо также сделать пустым значение аргумента `placement` в том же файле:

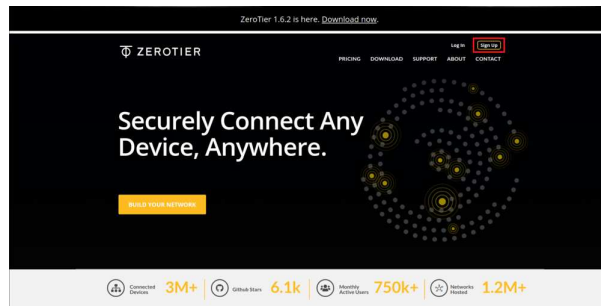
```
<arg name="placement" default=""/>
```

После всех настроек вызовите `sudo systemctl restart drone` для перезагрузки сервиса `drone`.

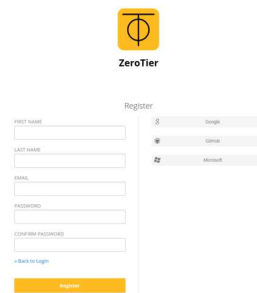
Создание виртуальной сети ZeroTier и подключение к ней

Создание и настройка сети ZeroTier

1. Зайдите на сайт [ZeroTier](#).

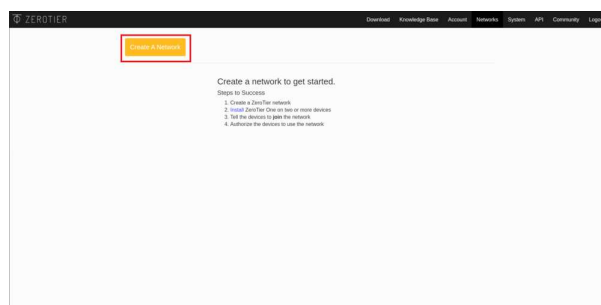


2. Зарегистрируйтесь в ZeroTier.

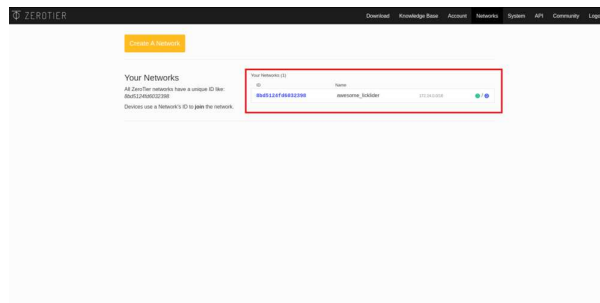


3. Зайдите в свой аккаунт.

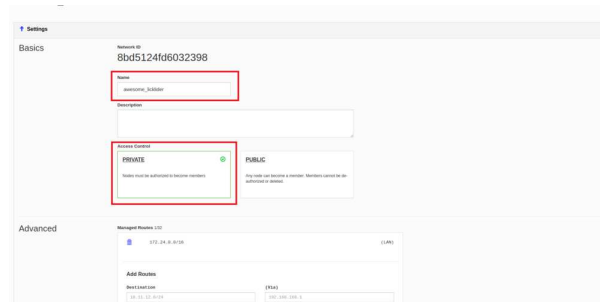
4. Нажмите кнопку *Create A Network*.



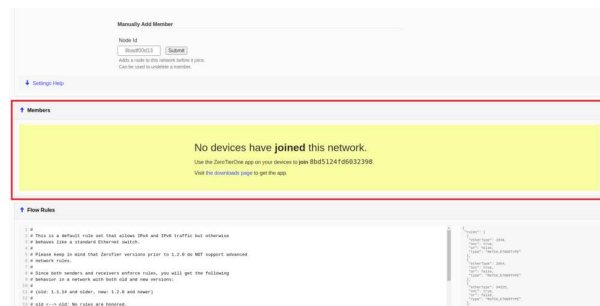
5. После этого вы увидите созданную вами сеть, ее ID и название. Для настройки сети нажмите на нее.



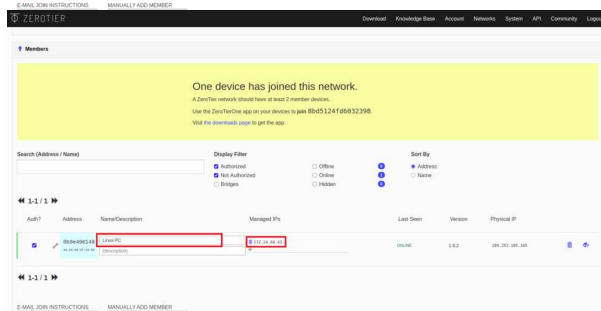
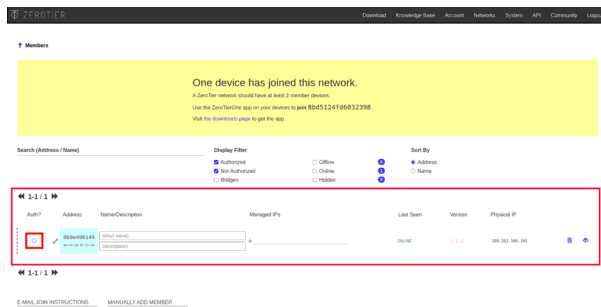
6. В открывшемся окне можно изменить имя сети и приватность подключения.



7. Пролитайте ниже, до графы *Members*. В ней будет написано о том, что в сети нет пользователей.



8. Устройства подключенные к сети будут отображаться в данной графе, для того, чтобы позволить им подключиться к сети, активируйте чекбокс *Auth?*. При этом, подключенному устройству автоматически выдаться внутренний IP адрес, в дальнейшем он будет использоваться для связи с данным устройством.



Указывайте имена для новых устройств, это поможет вам в дальнейшем отличать их друг от друга.

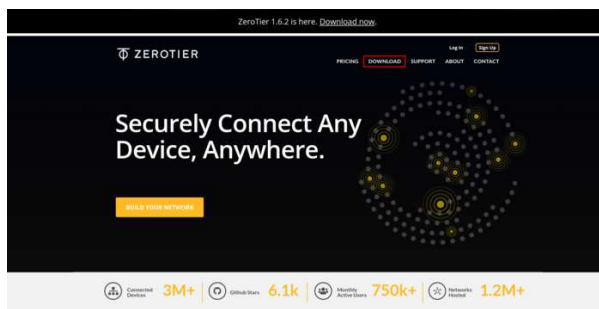
9. Повторите последний шаг для всех подключаемых устройств.

Сеть ZeroTier в случае бесплатного использования поддерживает до 50 пользователей одновременно.

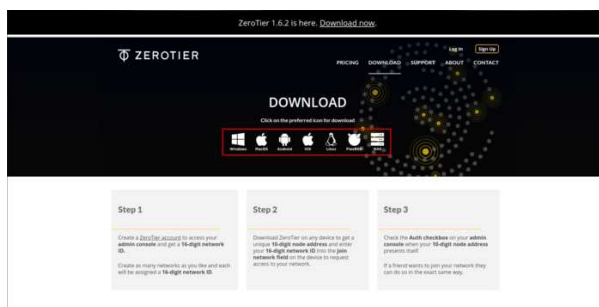
Настройка на Windows

Установка приложения

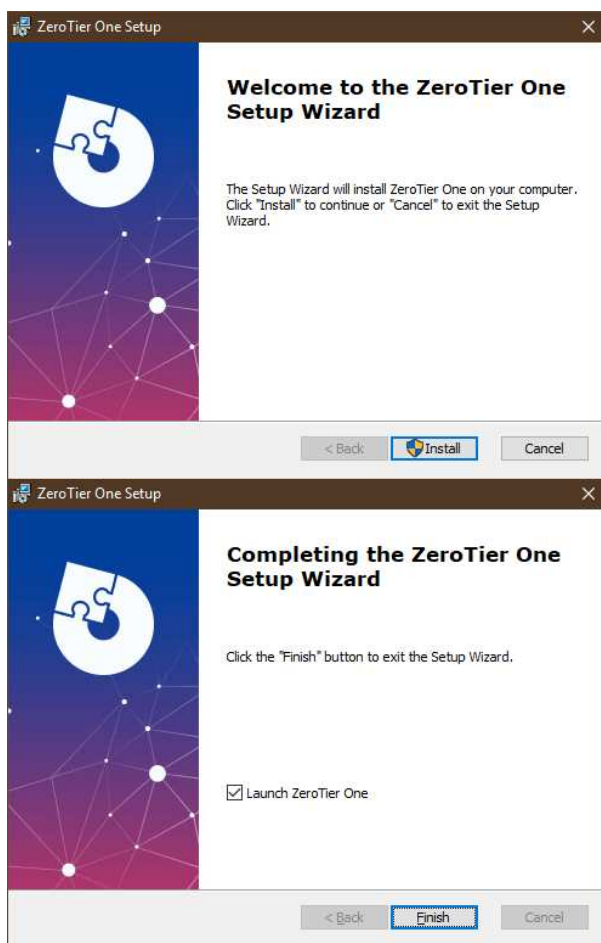
1. Перейдите на сайт ZeroTier.



2. Нажмите на иконку Windows.

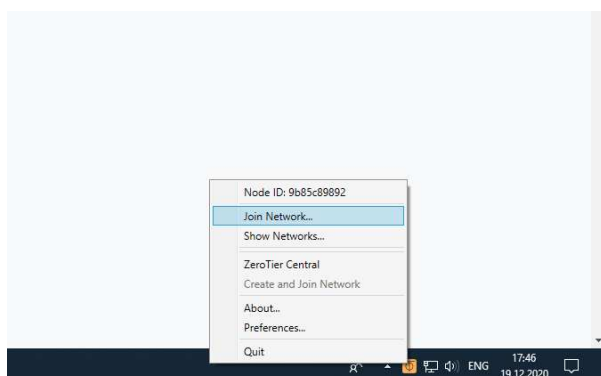


3. Скачайте и запустите файл `ZeroTier One.msi`.

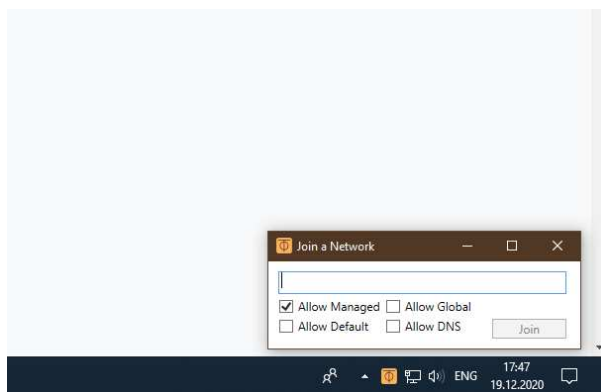


Подключение к сети

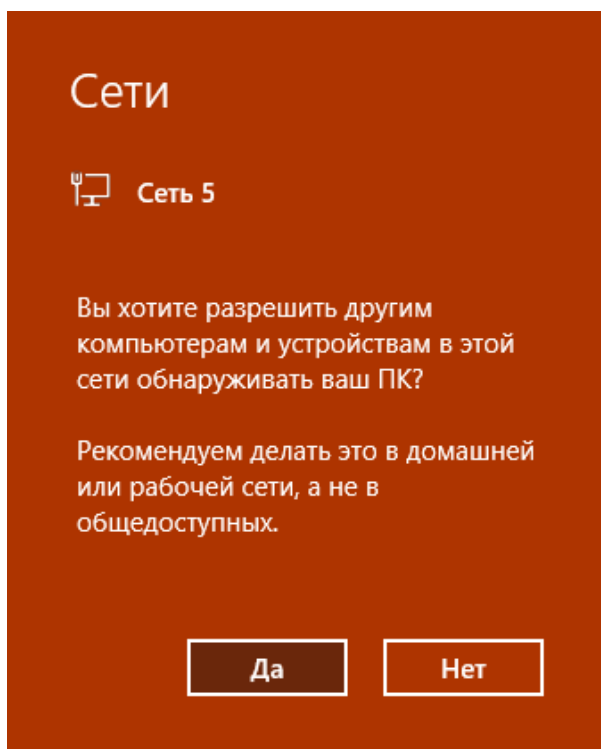
1. Запустите ZeroTier One.
2. Нажмите на иконку ZeroTier One в панели задач.
3. Нажмите на кнопку *Join Network...* для подключения к сети.



4. В появившемся окне введите ID вашей сети и нажмите кнопку *Join*.



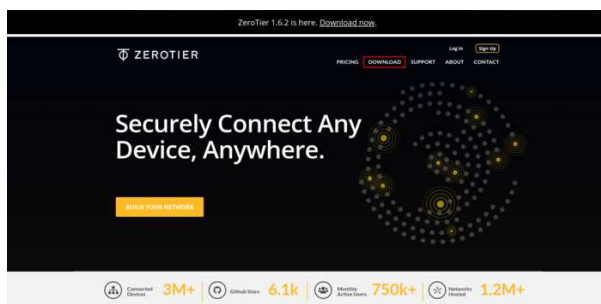
5. Разрешите использование новой сети.



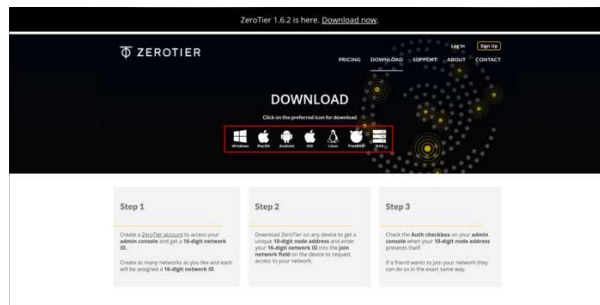
Настройка на iOS

Установка приложения

1. Перейдите на сайт ZeroTier.



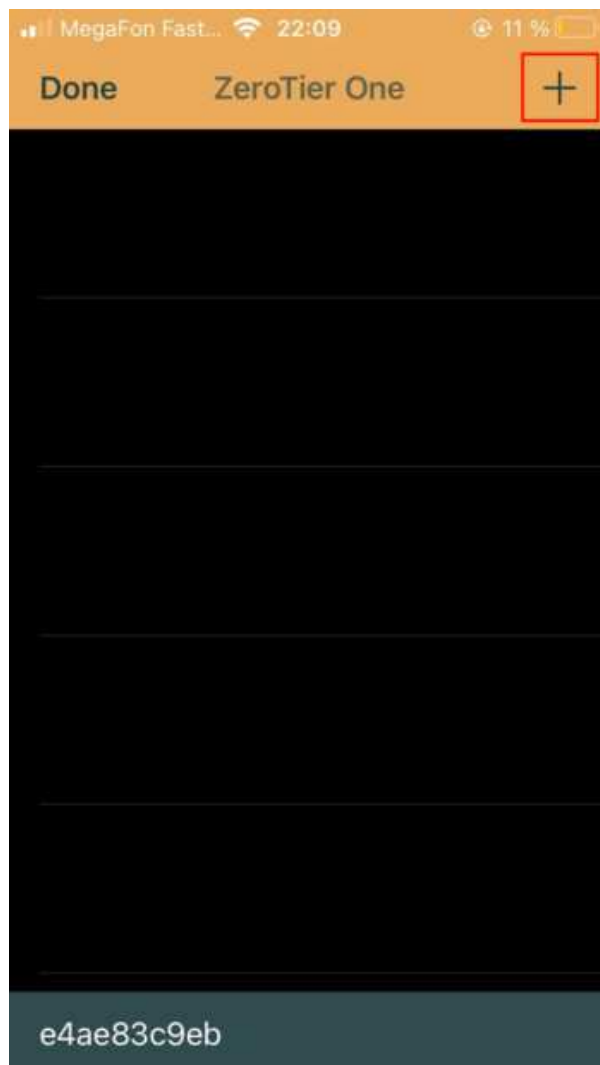
2. Нажмите на иконку iOS.



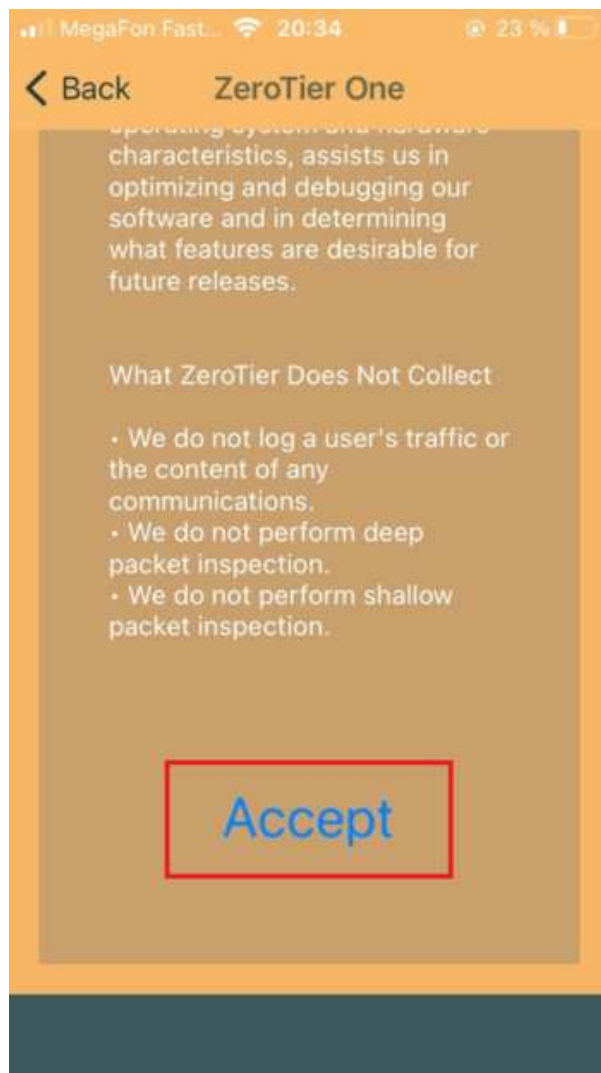
3. Установите приложение *ZeroTier One*.

Подключение к сети

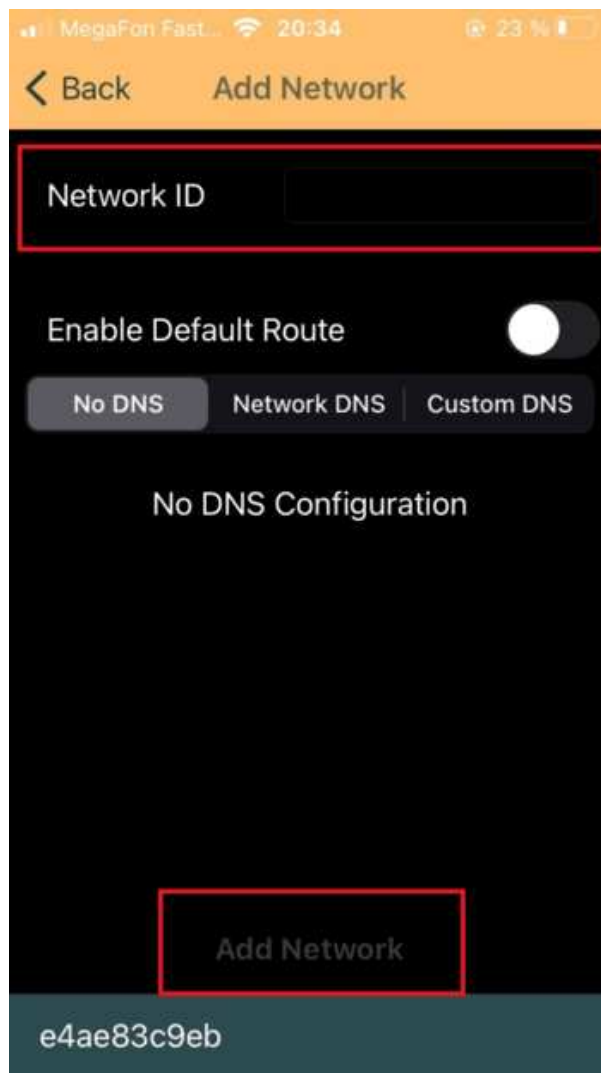
1. Запустите приложение *ZeroTier One*.
2. Нажмите на + для добавления нового подключения.



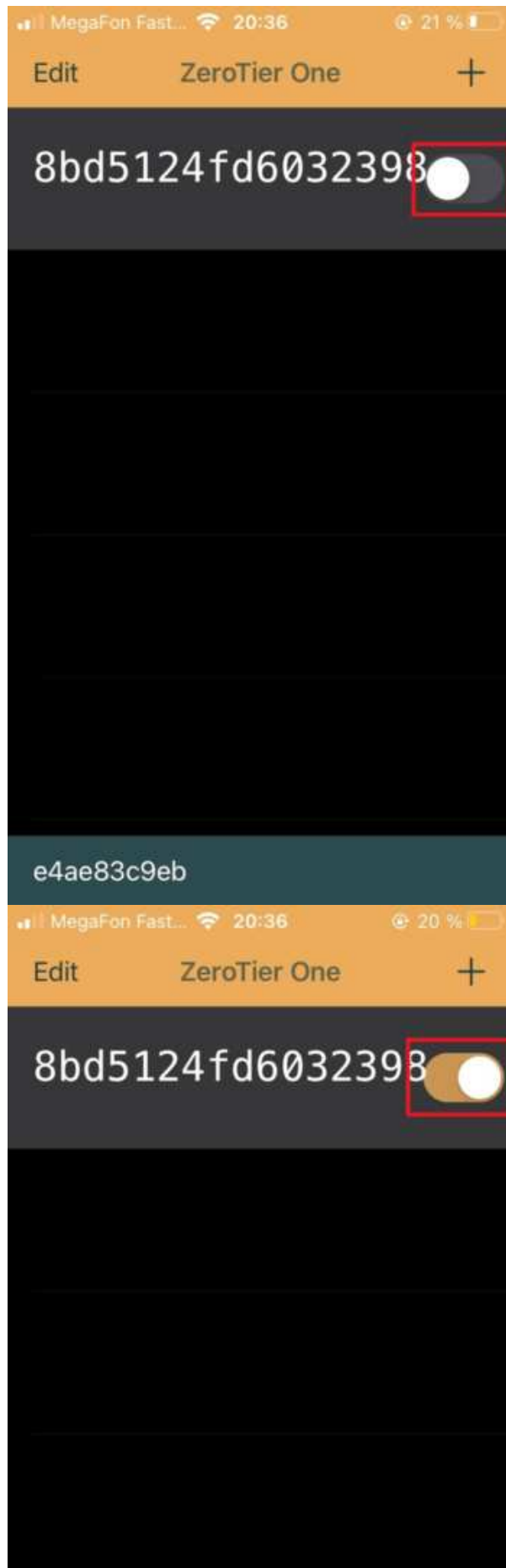
3. Подтвердите политику конфиденциальности.

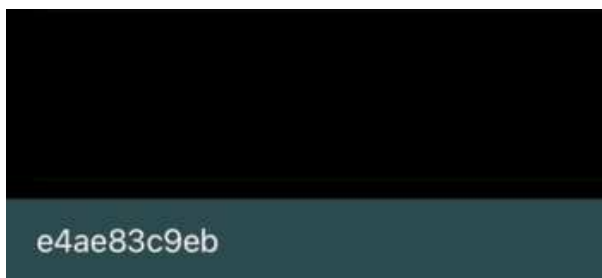


4. Введите ID вашей сети и нажмите кнопку *Add Network*.



5. Подтвердите добавление новой конфигурации VPN.
6. Подключитесь к VPN сети, сдвинув ползунок активации сети.





Настройка на Linux (PC, Raspberry Pi)

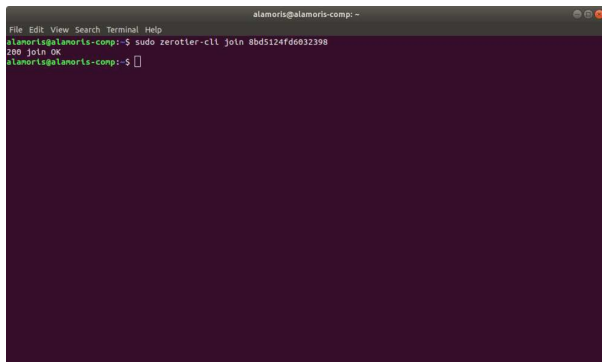
Установка приложения

1. Откройте консоль, для этого нажмите сочетание клавиш *Ctrl + Alt + T* или в строке поиска программ введите *Terminal*
2. Введите команду установки ZeroTier.

```
curl -s https://install.zerotier.com | sudo bash
```

Подключение к сети

1. Откройте консоль.
2. Введите команду `sudo zerotier-cli join network-id`, где `network-id` ЭТО ID вашей сети.

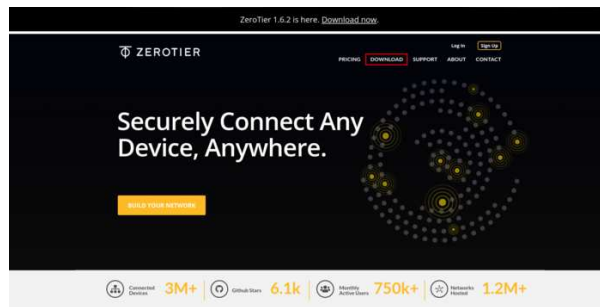


3. При успешном подключении, в консоль будет выведено соответствующее сообщение.

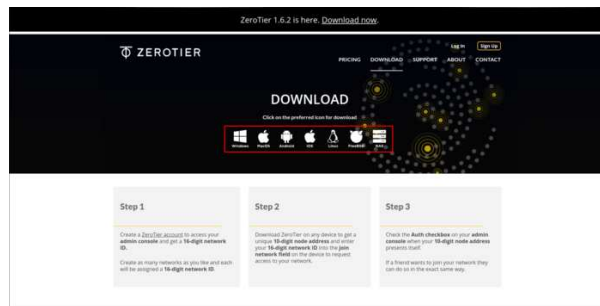
Установка и настройка на macOS

Установка приложения

1. Перейдите на сайт ZeroTier.



2. Нажмите на иконку macOS.



3. Скачайте и запустите файл ZeroTier One.pkg .

4. Установите приложение ZeroTier One.

Подключение к сети

1. Запустите приложение ZeroTier One.

2. В панели задач нажмите на иконку ZeroTier One.

3. В открывшемся окне нажмите *Join Network...*

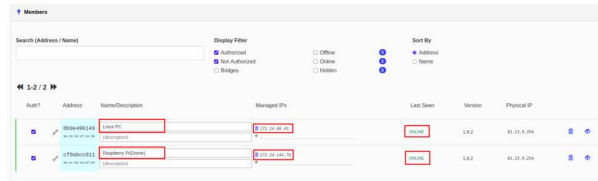


4. В поле *Enter Network ID* введите ID вашей сети.

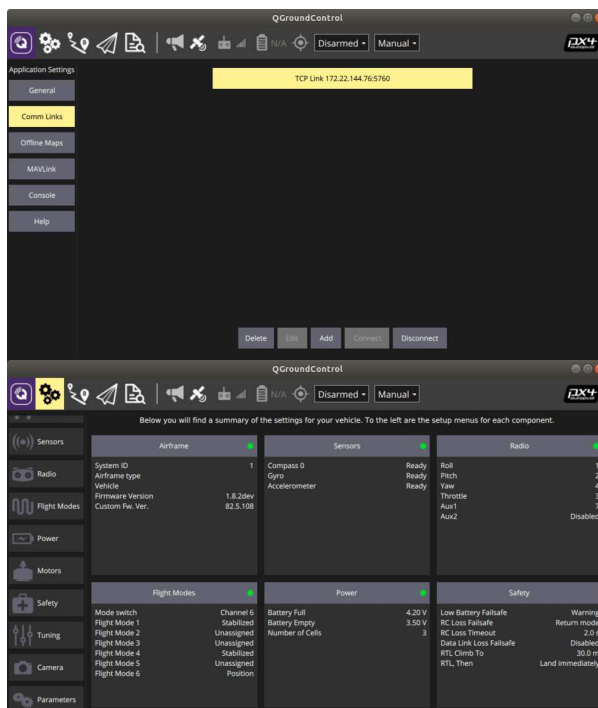


Подключение к коптеру

1. Убедитесь, что ZeroTier работает и имеет соединение с сетью на дроне и управляющем устройстве. Для этого убедитесь, что интересующие вас устройства имеют статус *Online*.



2. Убедитесь, что у всех устройств есть локальные IP адреса - *Managed IPs*.
3. Откройте QGC и во вкладке *Comm Links* добавьте TCP подключение, указав IP дрона. Подробнее про удаленное подключение читайте [тут](#).



Быстрое подключение к виртуальной сети

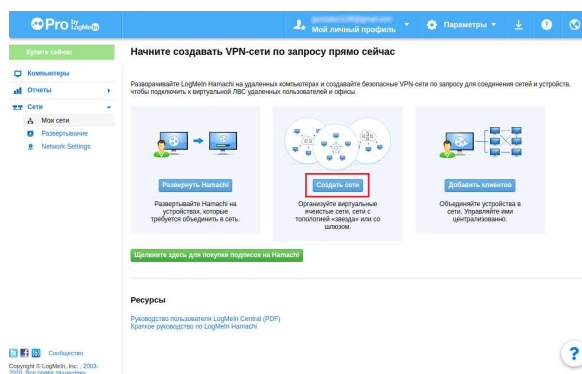
Для удаленного подключения и управления дроном или получения видеоизображения можно подключить его к виртуальной сети VPN.

Вы можете подключить вашу систему к любой доступной вам сети, если у вас есть соответствующие права доступа. В этой статье будет рассмотрен способ подключения к сети *LogMeIn*, как удобной и легкой в использовании.

Создание виртуальной сети

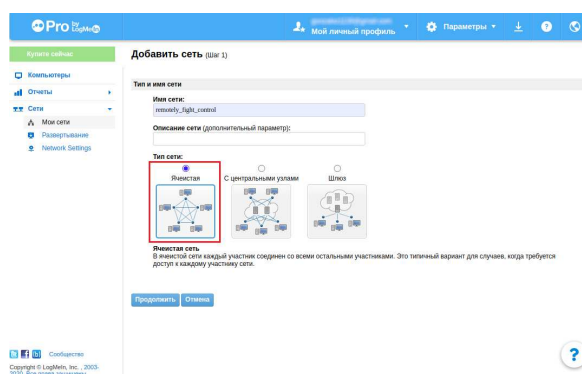
Создайте аккаунт и войдите на сайте [LogMeIn](https://www.logmeinpro.com).

После входа вы увидите основное меню управления вашими сетями.

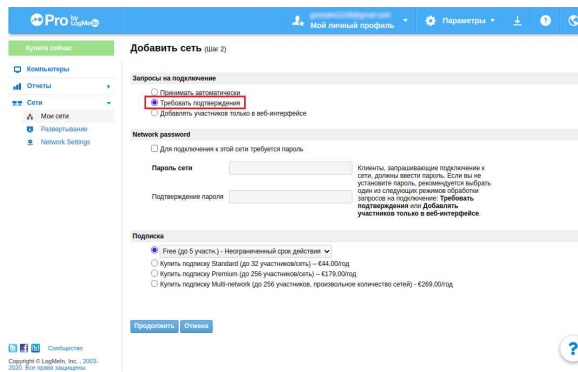


Выберите пункт *Создать сети*.

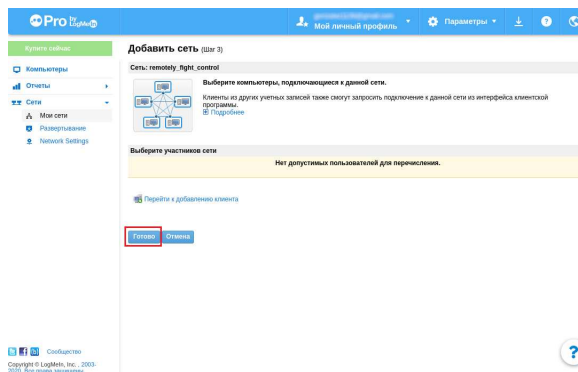
В открывшемся окне введите название сети и выберите тип *Ячеистая*.



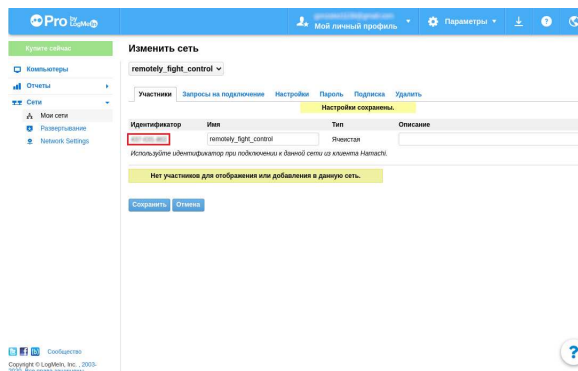
Далее меню *Запросы на подключение* выберите *Требовать подключение*.



Нажмите **Готово** и перейдите к настройке сети.



В открывшемся окне **Изменить сеть** необходимо запомнить значение поля **Идентификатор сети**, он будет использоваться в дальнейшем для подключения.



Установка менеджера Hamachi и подключение к сети

1. Скачайте Debian-пакет `logmein-hamachi`.

```
wget https://www.vpn.net/installers/logmein-hamachi_2.1.0.203-1_i386.deb
```

2. Установите пакет.

```
sudo dpkg -i logmein-hamachi_2.1.0.203-1_i386.deb
```

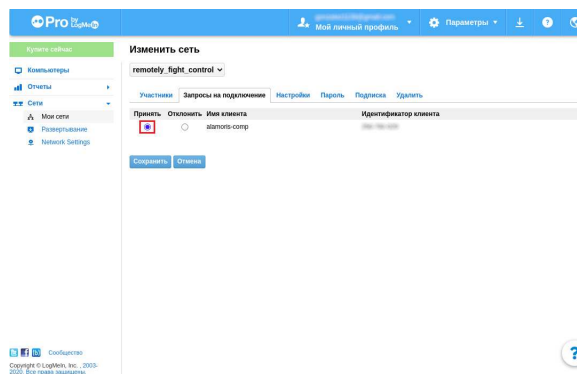
3. Подключите установленный модуль к сети.

```
hamachi login
```

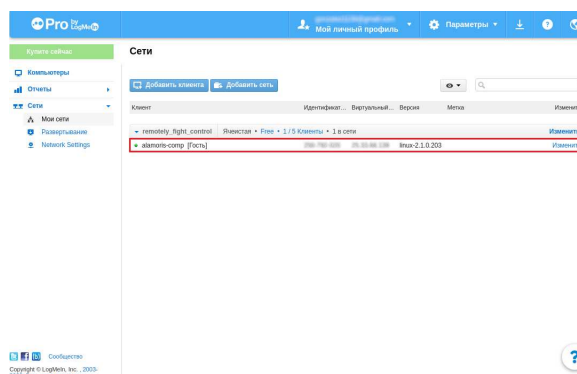
4. Подключитесь к сети используя ее идентификатор.

```
sudo hamachi do-join xxx-xxx-xxx
```

5. В поле ввода пароля нажмите *Enter*, если пароль не задан или введите его.
6. При успешном подключении вы увидите сообщение: *Joining 435-995-378 .. ok, request sent, waiting for approval.*
7. Подтвердите подключение к сети в меню *Изменить сеть*, во вкладке *Запросы на подключение*.



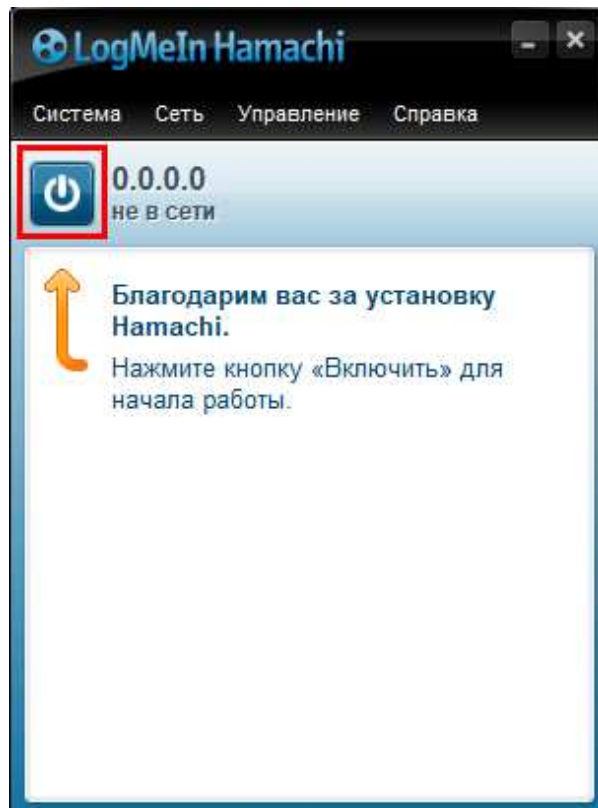
8. Можно проверить, удалось ли подключить пользователя в окне *Мои сети*.



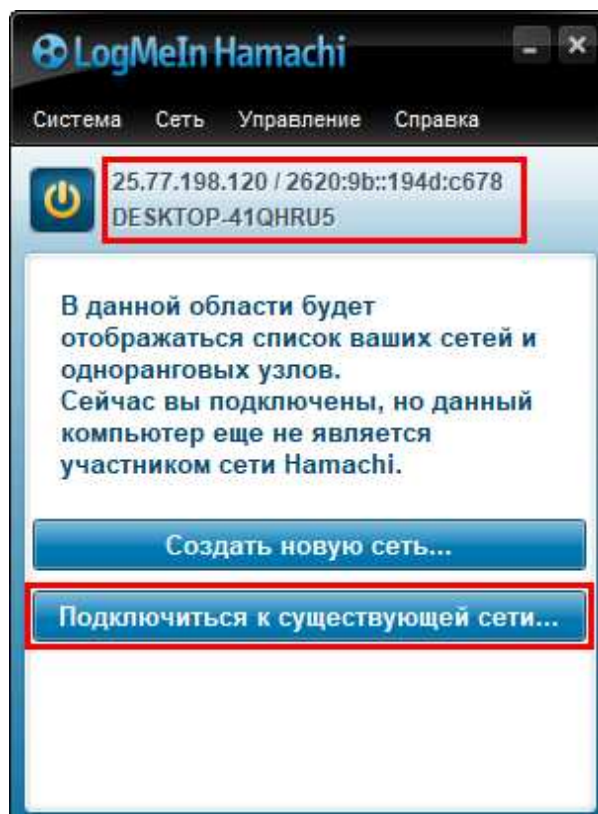
9. Повторите шаги 4–7 для подключения компьютера в случае, если вы пользуетесь операционной системой Linux, или обратитесь к инструкции для Windows.

Подключение к сети с помощью Windows

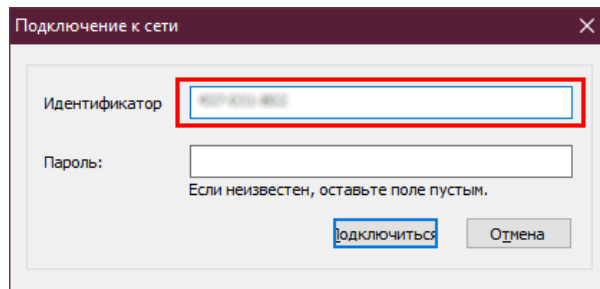
1. Установите приложение Hamachi.



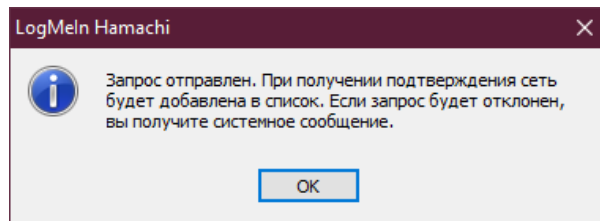
2. Запустите приложение Hamachi и нажмите кнопку включения. При необходимости введите свой логин.



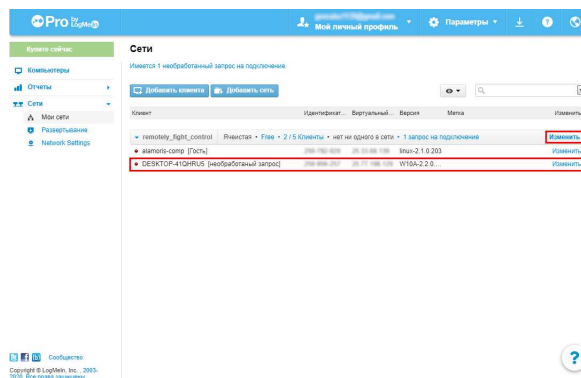
3. Если в шапке приложения отображается ваш виртуальный IP-адрес, подключитесь к сети, нажав соответствующую кнопку.



4. Введите идентификатор вашей сети.

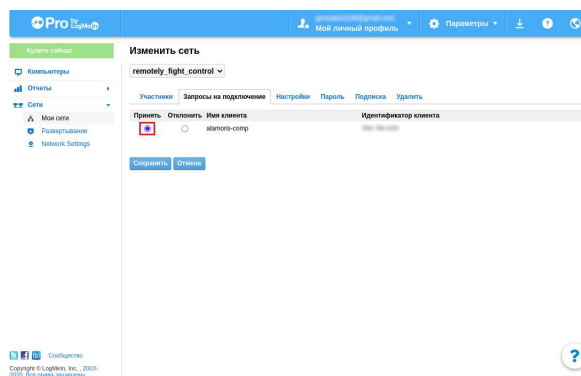


5. Если идентификатор введен верно, вы увидите соответствующее сообщение.



6. В меню вашей сети появится пользователь с префиксом "необработанный запрос".

7. Для подтверждения пользователя зайдите в меню *Изменить сеть* и во вкладке *Запросы на подключение* подтвердите подключение к сети.



См. также продолжение настройки для удаленного управления коптером и настройки стрима видео.

Управление при помощи 4G связи

Мобильная связь четвертого поколения удобный инструмент для передачи и получения информации на большой скорости. В настоящее время область покрытия мобильных операторов позволяет подключаться к интернету на большой скорости практически из любой точки.

Для передачи каких либо данных с вашего дрона на наземную станцию(QGroundControl) и обратно, вам необходимо настроить собственную VPN сеть.

Подключение 4G модема к Raspberry Pi

Подключите 4G модем с сим-картой в USB порт вашей Raspberry Pi.

Обратите внимание, что при подключении, некоторые модемы распознаются в системе как сетевая карта, без каких либо дополнительных настроек.

Пример 4g модема: *USB 4G Huawei E3372H*



Однако, некоторые модемы, например, *Quectel EP06*, автоматически не стартуют соединение. В этом случае необходимо воспользоваться утилитами `qmi-network` и `udhcpc`. Установить эти утилиты можно командой:

```
sudo apt install libqmi-utils udhcpc
```

Далее запустить соединение с интернетом следующими командами:

```
sudo ip link set wwan0 down
echo 'Y' | sudo tee /sys/class/net/wwan0/qmi/raw_ip
sudo ip link set wwan0 up
sudo qmi-network /dev/cdc-wdm0 start
sudo udhcpc -q -f -i wwan0
```

Более подробно прочитать про настройку сетевого подключения вы можете в [этой статье](#).

Проверить наличие соединения с интернетом можно командой:

```
ping -I wwan0 -c 5 8.8.8.8
```

Проверить скорость соединения с интернетом можно утилитой

speedtest :

```
sudo apt install speedtest-cli
speedtest
```

Подключение Raspberry Pi к VPN

Сформируйте необходимые ключи VPN сети, для подключения Raspberry Pi и наземной станции.

Для того, чтобы подключить Raspberry Pi к вашей сети, установите пакет

openvpn :

```
sudo apt-get install openvpn
```

Перенесите ваши ключи в директорию `/etc/openvpn/client` . Для удобства используйте графический SFTP интерфейс передачи данных, к примеру: WinSCP, FileZilla и т.д.

Для включения режима клиента, необходимо активировать переданные вами ключи. Ключи могут быть сформированы в различных форматах, к примеру:

`.ovpn` , `.conf` . Ключ или конфигурация использующийся на вашем дроне, должны быть строго в формате `.conf` .

Инициализируйте сервис применяющий ваши ключи для подключения в режиме клиента:

```
sudo systemctl enable openvpn-client@config-name
```

где `config-name` - название вашего конфигурационного файла.

Если все сделано правильно, при каждом перезапуске системы, сервис-клиент будет автоматически подключаться к вашей сети.

Перед началом работы не забудьте настроить и включить VPN подключение на вашем ПК.

В качестве удобной альтернативы вы можете воспользоваться VPN-сервисом [ZeroTier](#).

Управление дроном через QGroundControl

Для управления дроном предлагается использовать UDP протокол передачи, обеспечивающий меньшую задержку, ценой отсутствия гарантии получения пакета, что очень важно во время пилотирования дрона.

Убедитесь, что ваш дрон и наземная станция подключены к вашей сети.

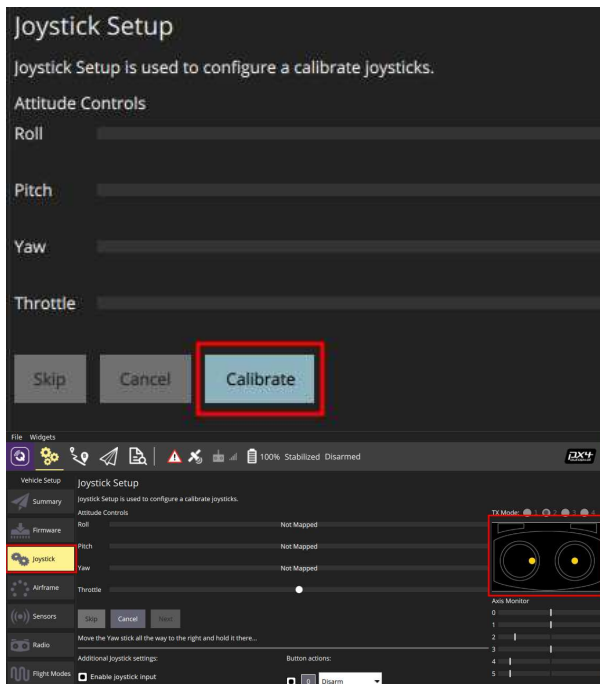
Для этого можете воспользоваться командой `ip addr`. Результатом ее выхода будет пронумерованный список активных сетей включенных на вашем устройстве. Обратите внимание стоит на подключение с префиксом `tun` и указанным вами IP адресом, если оно присутствует в вашем списке, ваш дрон подключился к сети.

В QGroundControl, аналогично [подключению по Wi-Fi](#), настройте подключение к вашему дрону по протоколу используемому в вашей сети: *UDP/TCP*. Рекомендуется использовать *UDP* подключение, за счет большей скорости передачи данных.

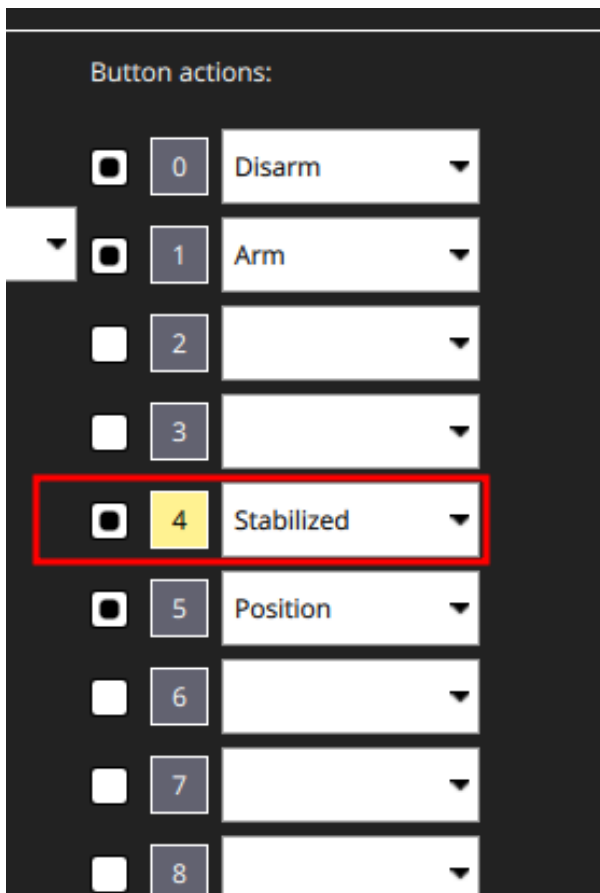
Если у вас появилась связь с дроном, подключите какой-либо джойстик к вашему ПК. Роль джойстика может выполнять как радио пульты, такие как, FlySky-i6X, Taranis x7 и т.д., так и джойстики от приставок или любые их эмуляции, которые распознаются системой.

Когда джойстик распознается системой, в колонке *Vehicle Setup* появится пункт *Joystick*, в случае, если он подсвечивается красным цветом, это значит, что требуется настройка.

Для калибровки джойстика, во вкладке *Joystick* нажмите кнопку *Calibrate* и следуйте инструкциям положения стиков пульта, указанных с левой стороны окна.



После успешной калибровки необходимо настроить полетные режимы. Чтобы это сделать несколько раз переключите необходимые тумблеры. В ходе переключения, вы увидите виртуальные каналы, на которых работают тумблеры. В активном положении будет подсвечиваться один из каналов.



При выборе джойстика, обратите внимание на количество рабочих каналов и на поддержку его, в QGroundControl(SDL2). Встречаются пульты поддерживающие всего 4 канала, что не удобно для такого типа управления.

Если изменения положения стиков отображается в окне QGroundControl, вам остается только применить параметр, определяющий, что управление дроном происходит с помощью джойстика, а не радиоаппаратуры:

`COM_RC_IN_MODE` - Joystick/No RC Checks

Поскольку мобильная связь не всегда бывает стабильна, рекомендуется увеличить таймаут на потерю сигнала управления до 5 секунд.



Дрон готов к полету!

Если дрон не армится при переведении левого стика в нижний правый угол, установите состояние Arm/Disarm на один из тумблеров.

Передача видео с камеры в QGroundControl

Передача видео возможна практически с любой камеры подключенной к вашей Raspberry Pi. Для этого вам потребуется установить или [собрать](#) пакет `gst-rtsp-launch`:

```
sudo apt update
sudo apt install gst-rtsp-launch
```

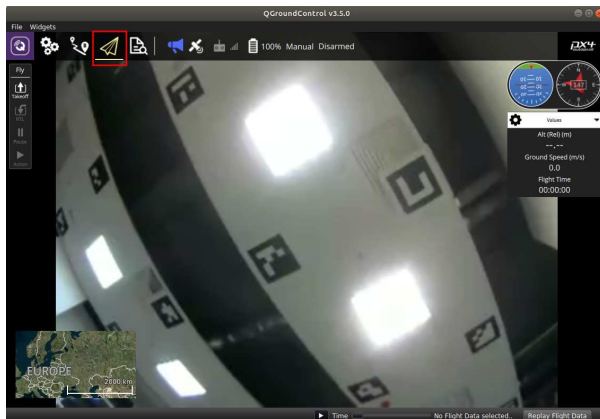
Чтобы запустить передачу изображений, необходимо ввести соответствующую командную строку:

```
gst-rtsp-launch "( v4l2src device=/dev/video0 ! video/x-raw, framerate=30/1, width=1280, height=720 ! rtspenc ! rtsp://192.168.1.100:5555 )" &&
```

Данная командная строка содержит параметры передачи видео, такие как: устройство передачи, частота кадров, высота/ширина изображения, метод кодирования и т.д. Подробнее о настройках [можно узнать тут](#).

```
Камера Raspberry Pi /dev/video0 может использоваться сервисом
drone и тогда gst-rtsp-launch не сможет получить к ней доступ.
Чтобы остановить сервис drone выполните команду sudo systemctl
stop drone . Также можно пустить трансляцию с USB-камеры, для этого
замените устройство передачи на /dev/video1 .
```

В приложении QGroundControl проверьте, что по ссылке `rtsp://192.168.11.1:8554/video` (IP-адрес вашей Raspberry Pi может отличаться) начался стрим изображения.



Автоматизация запуска передачи видео

Создайте файл и добавьте в него командную строку:

```
nano script_name.sh
```

Чтобы корректно запускать файл, необходимо присвоить ему соответствующие флаги доступа.

```
chmod a+x script_name.sh
```

Для того, чтобы передача видео запускалась каждый раз при включении системы, необходимо создать стартап-скрипт с помощью менеджера `systemd`. В директории `/etc/systemd/system` создайте файл `qgc_video.service` .

```
sudo nano /etc/systemd/system/qgc_video.service
```

В файл запишите соответствующие строки:

```
[Unit]
Description=VideoStream

[Service]
ExecStart=/bin/bash /home/pi/script_name.sh

[Install]
WantedBy=multi-user.target
```

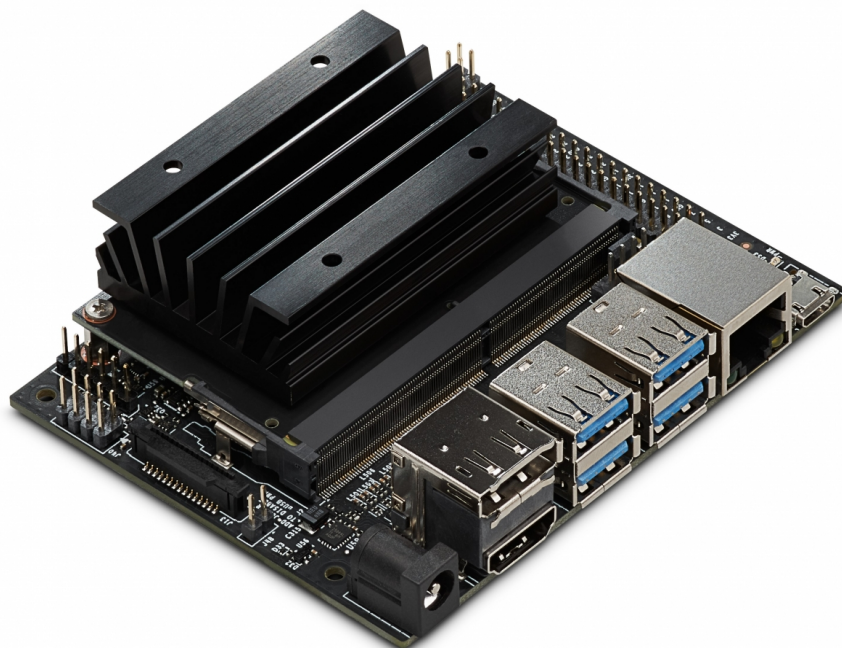
Осталось только инициализировать ваш скрипт в системе и он будет запускаться при каждом ее включении.

```
sudo systemctl enable qgc_video.service
```

Jetson Nano

О Jetson Nano

Jetson Nano – система на модуле (SoM), выпускаемая компанией Nvidia. Система построена на базе платформы Tegra X1 и несёт на себе четырёхядерный процессор ARM Cortex-A57 частотой 1.4 ГГц, 4 ГБ оперативной памяти и видеоядро на базе Nvidia Maxwell.



Набор для разработчиков Jetson Nano включает в себя как сам модуль, так и плату-носитель с портами USB 3.0, CSI, Ethernet и GPIO-пинами. Плата-носитель ненамного больше одноплатного компьютера Raspberry Pi и может быть использована в качестве бортового компьютера.

На плате-носителе изначально нет Wi-Fi-адаптера. Для организации беспроводного подключения к Jetson Nano следует использовать USB-адаптер или Wi-Fi карту стандарта M.2. Следует свериться со списком совместимого оборудования перед установкой адаптера.

Установка ПО

Nvidia предоставляет образ системы для Jetson Nano на базе Ubuntu 18.04. Эта версия системы поддерживается как база для ROS Melodic.

Начальная установка

Более подробные инструкции можно получить на [официальном сайте Nvidia](#) для Jetson Nano.

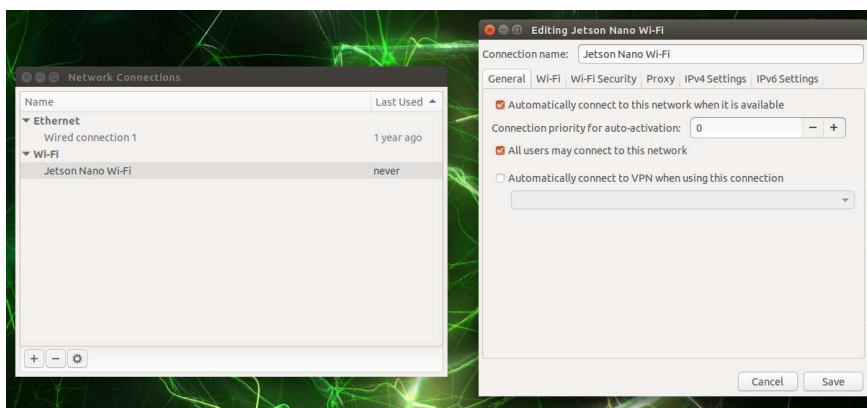
Для начальной установки требуется использование клавиатуры, мыши и HDMI-монитора. Скачайте [образ системы Jetson Nano](#) и запишите его на карту microSD (размером не менее 16 ГБ; рекомендуется использовать карту объёмом не менее 32 ГБ). Вставьте записанную карту в модуль Jetson Nano, подключите клавиатуру, мышь и монитор к плате-носителю и подайте питание на модуль.

Jetson Nano можно питать от кабеля microUSB, но для повышения надёжности рекомендуется использовать специальный разъём питания. При использовании этого разъёма следует поместить перемычку на пины J48 (расположены рядом с разъёмом CSI).

Примите лицензионное соглашение на использование системы и следуйте дальнейшим указаниям установщика. После установки система автоматически перезагрузится и покажет экран входа в систему. Выберите созданного вами пользователя и введите пароль, указанный на этапе установки.

Настоятельно рекомендуется выбрать английский язык как системный, дабы избежать возможных проблем с ROS.

Рекомендуется настроить автоматическое подключение к Wi-Fi сети. Для этого после установки системы нажмите на значок Wi-Fi подключения в верхней части экрана, выберите опцию "Edit Connections..." в выпадающем меню, выделите текущую Wi-Fi сеть в открывшемся списке и нажмите на кнопку с иконкой шестерёнки.



Во вкладке "General" поставьте галочку возле пункта "All users may connect to this network". Нажмите на кнопку "Save" для применения параметров и закрытия окна.

Убедитесь, что Jetson Nano доступен в сети. В образе по умолчанию установлен и включен SSH-сервер; для выполнения дальнейших операций рекомендуется подключиться к нему.

Установка ROS

Ubuntu 18.04 официально поддерживается дистрибутивом ROS Melodic, и поэтому на официальном сайте [есть подробная инструкция по его установке](#).

Добавьте ключи и репозитории OSRF в систему:

```
sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key C1CF6E91
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" >> /etc/apt/sources.list.d/ros-latest.list'
sudo apt update
```

Установите базовые ROS-пакеты (стек `ros-base`):

```
sudo apt install ros-melodic-ros-base
```

Активируйте окружение ROS и обновите кэш утилиты `rosdep`:

```
source /opt/ros/melodic/setup.bash
sudo rosdep init
rosdep update
```

Добавьте строчку `source /opt/ros/melodic/setup.bash` в конец файла `.profile` в своей домашней директории, чтобы не производить активацию окружения ROS каждый раз заново.

Установите пакетный менеджер `pip` для Python 2 (он требуется для установки некоторых зависимостей):

```
sudo apt install curl
curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py
sudo python ./get-pip.py
```

Ноды

Сборка и запуск нод предлагается оставить продвинутым пользователям и не включать в документацию.

Возможные проблемы

CSI-камеры

Jetson Nano не поддерживает старые камеры для Raspberry Pi (v1, на базе сенсора Omnivision OV5647). Камеры Raspberry Pi v2 (на базе Sony IMX219) поддерживаются, но не показываются в виде Video4Linux-устройств.

Изображения с этих камер можно захватывать с помощью GStreamer. Для последующей передачи этих изображений в ROS можно использовать ноды `gscam` или `jetson_camera`. Для запуска ноды `jetson_camera` потребуется собрать OpenCV из ветки 3.4 с поддержкой GStreamer.

Примеры конвейеров GStreamer для захвата изображения доступны в репозитории [JetsonHacksNano](#).

Изображение с камеры может становиться более красным по краям. Это можно исправить с помощью настройки процессора изображения. Эта процедура должна выполняться производителями камер; [вот пример файла настройки процессора изображения](#) от компании Arducam.

Настройка Wi-Fi

Wi-Fi адаптер на Raspberry Pi имеет два основных режима работы:

1. **Режим клиента** – RPi подключается к существующей Wi-Fi сети.
2. **Режим точки доступа** – RPi создает Wi-Fi сеть, к которой вы можете подключиться.

При использовании [образа для RPi](#) по умолчанию Wi-Fi адаптер работает в [режиме точки доступа](#).

Изменение пароля или SSID (имени сети)

1. Отредактируйте файл `/etc/wpa_supplicant/wpa_supplicant.conf` (используя [SSH-соединение](#)):

```
sudo nano /etc/wpa_supplicant/wpa_supplicant.conf
```

Измените значение параметра `ssid`, чтобы изменить название Wi-Fi сети, и параметра `psk`, чтобы изменить пароль. Например:

```
network={
    ssid="my-ssid"
    psk="dronewifi123"
    mode=2
    proto=RSN
    key_mgmt=WPA-PSK
    pairwise=CCMP
    group=CCMP
    auth_alg=OPEN
}
```

2. Перезагрузите Raspberry Pi.

Длина пароля для Wi-Fi сети должна быть **не менее** 8 символов.

При некорректных настройках `wpa_supplicant.conf` Raspberry Pi перестанет раздавать Wi-Fi!

Переключение адаптера в режим клиента

Для редактирования некоторых файлов, к которым у вас не будет доступа, необходимо использовать `sudo nano`, вместо `nano`.

1. Выключите службу `dnsmasq`.

```
sudo systemctl stop dnsmasq
sudo systemctl disable dnsmasq
```

2. Включите получение IP адреса на беспроводном интерфейсе DHCP клиентом. Для этого удалите из файла `/etc/dhcpd.conf` строки:

```
interface wlan0
static ip_address=192.168.11.1/24
```

3. Настройте `wpa_supplicant` для подключения к существующей точке доступа. Для этого замените содержимое файла

`/etc/wpa_supplicant/wpa_supplicant.conf` на:

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=GB

network={
    ssid="SSID"
    psk="password"
}
```

где `SSID` – название сети, а `password` – пароль.

4. Перезапустите службу `dhcpd`.

```
sudo systemctl restart dhcpd
```

Переключение адаптера в режим точки доступа

1. Включите статический IP адрес на беспроводном интерфейсе. Для этого добавьте в файл `/etc/dhcpd.conf` строки:

```
interface wlan0
static ip_address=192.168.11.1/24
```

2. Настройте `wpa_supplicant` на работу в режиме точки доступа. Для этого замените содержимое файла `/etc/wpa_supplicant/wpa_supplicant.conf` на:

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=GB

network={
    ssid="drone-1234"
    psk="dronewifi"
    mode=2
    proto=RSN
    key_mgmt=WPA-PSK
    pairwise=CCMP
    group=CCMP
    auth_alg=OPEN
}
```

где `drone-1234` — название сети, а `dronewifi` — пароль.

3. Включите службу `dnsmasq` .

```
sudo systemctl enable dnsmasq
sudo systemctl start dnsmasq
```

4. Перезапустите службу `dhcpcd` .

```
sudo systemctl start dhcpcd
```

Ниже вы можете узнать больше о том, как устроена работа с сетью на RPi.

Устройство сети RPi

Работа сети на [образе](#) поддерживается двумя предустановленными службами:

- **networking** — служба включает все сетевые интерфейсы в момент запуска.
- **dhcpcd** — служба обеспечивает настройку адресации и маршрутизации на интерфейсах, полученных динамически или указанных в файле настроек статически.

Для работы в режиме роутера (точки доступа) RPi необходим DHCP сервер. Он служит для автоматической выдачи настроек текущей сети подключившимся клиентам. В роли такого сервера может выступать `isc-dhcp-server` ИЛИ `dnsmasq` .

dhcpcd

Начиная с Raspbian Jessie настройки сети больше не задаются в файле `/etc/network/interfaces` . Теперь за выдачу адресации и настройку маршрутизации отвечает `dhcpcd` .

По умолчанию на всех интерфейсах включен dhcp-клиент. Настройки интерфейсов меняются в файле `/etc/dhcpd.conf`. Для того, чтобы поднять точку доступа необходимо прописать статический ip-адрес. Для этого в конец файла необходимо добавить следующие строки:

```
interface wlan0
static ip_address=192.168.11.1/24
```

Если интерфейс является беспроводным (wlan), то служба `dhcpd` триггерит `wpa_supplicant`, который в свою очередь работает непосредственно с wifi-адаптером и переводит его в заданное состояние.

wpa_supplicant

wpa_supplicant – служба конфигурирует Wi-Fi адаптер. Служба `wpa_supplicant` работает не как самостоятельная (хотя как таковая существует), а запускается как дочерний процесс от `dhcpd`.

Конфигурационный файл по умолчанию должен иметь путь

`/etc/wpa_supplicant/wpa_supplicant.conf`. Пример конфигурационного файла:

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=GB

network={
    ssid="my-drone"
    psk="dronewifi"
    mode=2
    proto=RSN
    key_mgmt=WPA-PSK
    pairwise=CCMP
    group=CCMP
    auth_alg=OPEN
}
```

Внутри конфига указываются общие настройки `wpa_supplicant` и параметры для настройки адаптера. Также конфигурационный файл содержит секции `network` – основные настройки Wi-Fi сети такие как SSID сети, пароль, режим работы адаптера. Таких блоков может быть несколько, но используется первый рабочий. Например, если вы указали в первом блоке подключение к некоторой недоступной сети, то адаптер будет настроен следующей удачной секцией, если такая есть. Подробнее о синтаксисе `wpa_supplicant.conf`.

wpa_passphrase

`wpa_passphrase` – утилита для создания секции `network`.

```
wpa_passphrase SSID PASSWORD
```

После выполнения команды скопируйте полученную секцию в ваш конфигурационный файл. Можно удалить закомментированное поле `psk` и оставить только поле с хешем пароля, либо наоборот.

```
network={
  ssid="SSID"
  #psk="PASSWORD"
  psk=c2161655c6ba444d8df94cbbf4e9c5c4c61fc37702b9c66ed37aee1545a5a333
}
```

Ссылки

1. habr.com: Linux WiFi из командной строки с `wpa_supplicant`
2. wiki.archlinux.org: WPA supplicant (Русский)
3. blog.hoxnox.com: WiFi access point with `wpa_supplicant`
4. dmitrysnotes.ru: Raspberry Pi 3. Присвоение статического IP-адреса
5. thegeekdiary.com: Linux OS Service 'network'
6. frillip.com: Using your new Raspberry Pi 3 as a Wi-Fi access point with `hostapd` (также здесь есть инструкция по настройке форвардинга для использования RPi в качестве шлюза для выхода в интернет)
7. expert-orda.ru: Настройка DHCP-сервера на Ubuntu (Настройка `isc-dhcp-server`)
8. academicfox.com: Raspberry Pi беспроводная точка доступа (WiFi access point) (Настройка маршрутов, `hostapd`, `isc-dhcp-server`)
9. weworkweplay.com: Automatically connect a Raspberry Pi to a Wifi network (Есть настройки для создания открытой точки доступа)
10. wiki.archlinux.org: WPA supplicant

Интерфейс UART

UART – последовательный асинхронный интерфейс для передачи данных, применяемый во многих устройствах. Например GPS-антенны, Wi-Fi роутеры или Pixhawk.

Интерфейс обычно содержит две линии: TX – линия для передачи данных, RX – линия для приёма данных. А также обычно использует 5-ти вольтовую логику.

Для соединения двух устройств необходимо линию TX первого устройства подать на RX второго. Аналогичную операцию нужно совершить с другой стороны, чтобы обеспечить двустороннюю передачу данных.

Необходимо синхронизировать уровни напряжений – соединить землю на двух устройствах.

Почитать больше про интерфейс и протокол можно в [этой статье](#).

Linux TTY

В Linux есть понятие Posix Terminal Interface (подробнее [здесь](#)). Это некоторая абстракция над последовательным или виртуальным интерфейсом, позволяющая работать с устройством несколькими агентам одновременно.

В качестве примера такой абстракции в Raspbian можно привести `/dev/tty1` – устройство вывода текста на экран подключенный по HDMI.

UART на Raspberry Pi 3

В Raspberry Pi 3 есть два аппаратных UART интерфейса:

1. `mini UART` (`/dev/ttyAMA0`) – для своей работы использует тактирование видеоядра RPi, в связи с чем ограничивает его частоту.
2. `PL011` (`/dev/ttyS0`) – полноценный UART интерфейс выполненный на отдельном блоке кристалла микроконтроллера.

Подробнее про UART на Raspberry Pi в [официальной статье](#).

Данные интерфейсы с помощью вентиля микроконтроллера можно переключать между двумя физическими выходами:

1. разъём UART на GPIO;
2. Bluetooth модуль RPi.

По умолчанию в Raspberry Pi 3 `PL011` подключен к Bluetooth модулю. А `mini UART` отключен с помощью значения директивы `enable_uart`, по дефолту равной `0`.

Надо понимать, что директива `enable_uart` меняет свое дефолтное значение исходя из того, какой UART подключен к Bluetooth модулю RPi с помощью директивы `dtoverlay=pi3-miniuart-bt`.

Для удобства работы с этими выходами в Raspbian существуют алиасы:

- `/dev/serial0` – всегда указывает на то TTY устройство, что подключено к GPIO портам.
- `/dev/serial1` – всегда указывает на то TTY устройство, что подключено к Bluetooth модулю.

Настройка UART на Raspberry Pi

Для настроек UART существуют директивы, которые находятся в

```
/boot/config.txt
```

Для включения UART интерфейса на GPIO:

```
enable_uart=1
```

Для отключения UART интерфейса от Bluetooth модуля:

```
dtoverlay=pi3-disable-bt
```

Для подключения mini UART к Bluetooth модулю:

```
dtoverlay=pi3-miniuart-bt
```

В случае отключения Bluetooth модуля следует отключить `hciuart` сервис:

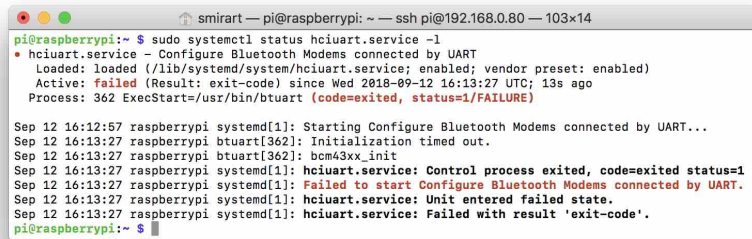
```
sudo systemctl disable hciuart.service
```

Настройка образа по умолчанию

На [образе для RPi](#) изначально выключен mini UART и Bluetooth модуль.

Bugs

Если использовать подключение mini UART к Bluetooth, `hciuart` падает с ошибкой:



```
pi@raspberrypi:~$ sudo systemctl status hciuart.service -l
● hciuart.service - Configure Bluetooth Modems connected by UART
   Loaded: loaded (/lib/systemd/system/hciuart.service; enabled; vendor preset: enabled)
   Active: failed (Result: exit-code) since Wed 2018-09-12 16:13:27 UTC; 13s ago
     Process: 362 ExecStart=/usr/bin/btuart (code=exited, status=1/FAILURE)

Sep 12 16:12:57 raspberrypi systemd[1]: Starting Configure Bluetooth Modems connected by UART...
Sep 12 16:13:27 raspberrypi btuart[362]: Initialization timed out.
Sep 12 16:13:27 raspberrypi btuart[362]: bcm43xx_init
Sep 12 16:13:27 raspberrypi systemd[1]: hciuart.service: Control process exited, code=exited status=1
Sep 12 16:13:27 raspberrypi systemd[1]: Failed to start Configure Bluetooth Modems connected by UART.
Sep 12 16:13:27 raspberrypi systemd[1]: hciuart.service: Unit entered failed state.
Sep 12 16:13:27 raspberrypi systemd[1]: hciuart.service: Failed with result 'exit-code'.
pi@raspberrypi:~$
```

В случае отключения Bluetooth

```
/dev/serial0 -> ttyAMA0
/dev/serial1 -> ttyS0
```

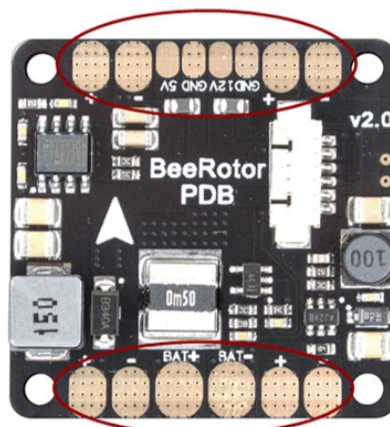
Использование мультиметра

Проверка цепей (прозвонка)

Проверяемый объект должен быть отключен от питания (обесточен)!

С помощью мультиметра проверить отсутствие короткого замыкания (прозвонить):

- Выставить мультиметр в режим прозвона.
- Проверить работу мультиметра путем замыкания щупов между собой. При корректной работе прибор издаст характерный звук.
- Попарно красный щуп прикладывается к “+” контакту, черный к “-” / “GND”. Если в цепи есть короткое замыкание, издается звук.



1. Прозвонить следующие цепи на НЕЗАМКНУТОСТЬ (отсутствие звукового сигнала мультиметра):

- “BAT+” и “BAT-”
- “12V” и “GND”
- “5V” и “GND”

2. Прозвонить следующие цепи на ЗАМКНУТОСТЬ (появление звукового сигнала мультиметра):

- “BAT-” с каждым контактом, обозначенным “-” и “GND”
- “BAT+”, с каждым контактом, обозначенным “+”

Проверка напряжения

С помощью мультиметра необходимо удостовериться, что преобразователи напряжения, расположенные на плате распределения питания, работают исправно и выдают напряжение 5В и 12В соответственно.

- Переключить мультиметр в режим "Измерение постоянного напряжения"

- Выбрать верхнюю границу измеряемого напряжения (в нашем случае, не более 20 В)
- Убедиться, что АКБ подключено
- Провести следующие измерения:
 1. Замерить напряжение АКБ (между ВАТ+ и ВАТ-). Оно должно лежать в пределах от 14.0В до 16.8В
 2. Замерить напряжение на выходе 5В. Оно не должно превышать 5.5В
 3. Замерить напряжение на выходе 12В. Оно не должно превышать 12.5В

После проведенных измерений:

- отключите АКБ
- выключите мультиметр

Неисправности радиоаппаратуры

Пульт заблокирован

Если пульт заблокирован, то на ЖК Экране будет отображено предупреждение: *Warning. Place all switches in their up position and lower the throttle.*

Для разблокировки пульта необходимо привести все стики и переключатели в исходное положение:

1. Левый стик (1) в центральной нижней позиции.
2. Переключатели А, В, С, D (2) в положение "от себя".
3. Правый стик (3) в центре.



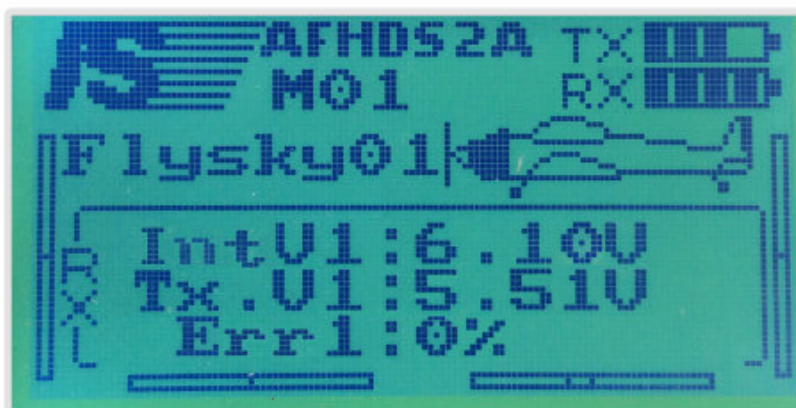
Нет связи с приемником

Для проверки соединения пульта с приемником, включите пульт и обратите внимание на индикацию на ЖК экране.

1. Соединение с приемником отсутствует:



2. Соединение с приемником установлено:

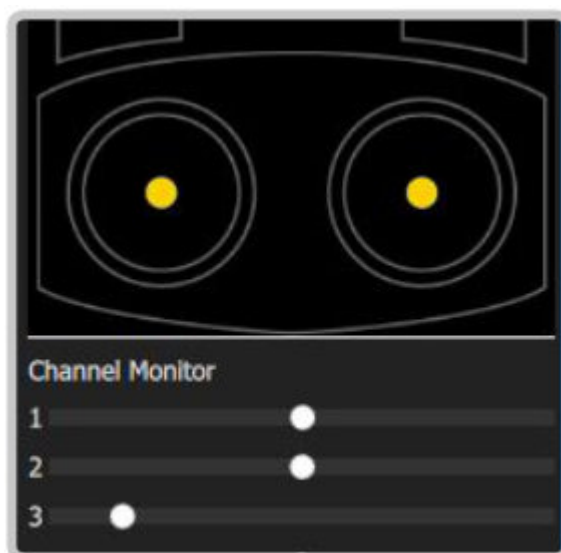


Если соединение отсутствует, то:

1. Проверьте, что приемник включен (моргает красный светодиод). Если светодиод горит непрерывно красным, то значит связь установлена с другим пультом.
2. Проведите процедуру сопряжения пульта и приемника.

Нет связи с полетным контроллером

Если нет связи с полетным контроллером, то на экране монитора компьютера в окне *Channel Monitor* не будут отображаться изменения положения слайдеров при перемещении стиками пульта.



1. Зайдите в меню (удерживайте нажатой кнопку *OK*).
2. Выберите меню *System setup* (Кнопки *Up/Down* - для навигации, кнопка *OK* — подтверждение выбора).
3. Выберите *RX setup > PPM OUTPUT > "On*.
4. Сохраните изменения (удерживайте нажатой кнопку *CANCEL*).

Прошивка ESC регуляторов с помощью BLHeliSuite

Хорошая статья, которая объясняет принцип работы ESC (Electric speed controller) регуляторов: <http://www.avmodels.ru/engines/electric/esc.html>

Зачем перепрошивать?

Иногда требуется поменять один из параметров регулятора, например направление вращения мотора, минимальная и максимальная скважности PPM сигнала на входе контроллера, уровень громкости звуковых сигналов, издаваемых мотором или время, через которое регулятор начинает напоминать, что он включён.

Программа для прошивки регуляторов

Для прошивки самых разнообразных ESC регуляторов существует программа [BLHeliSuite](#) (для Windows).

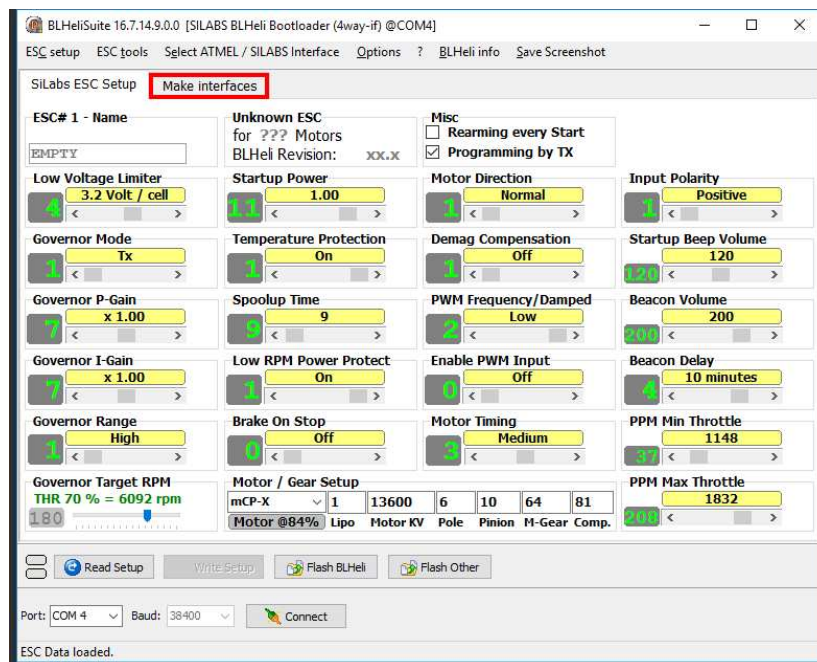
Для запуска программы (BLHeliSuite.exe) необходимо распаковать архивы BLHeliAtmelHEX.zip и BLHeliSilabsHEX.zip в папку с программой.

Программатор для прошивки регуляторов

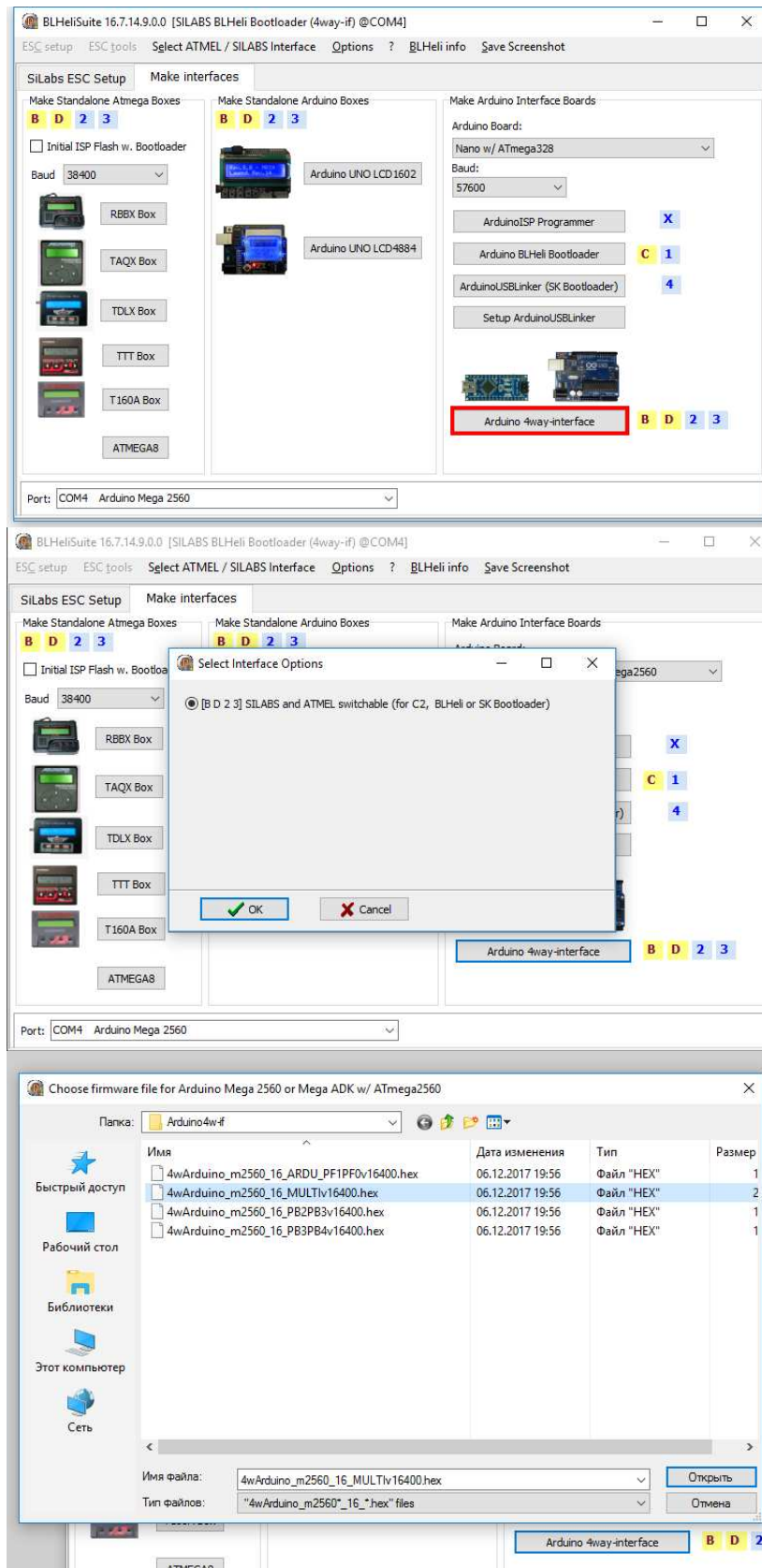
Чтобы прошить регулятор, необходим программатор, который умеет общаться с контроллером регулятора по 1-wire протоколу. Один из способов добыть программатор - взять подвернувшуюся под руку ардуинку и прошить её специальной прошивкой. В BLHeliSuite есть инструмент для создания интерфейсов программаторов.

Создание программатора на примере Arduino Mega.

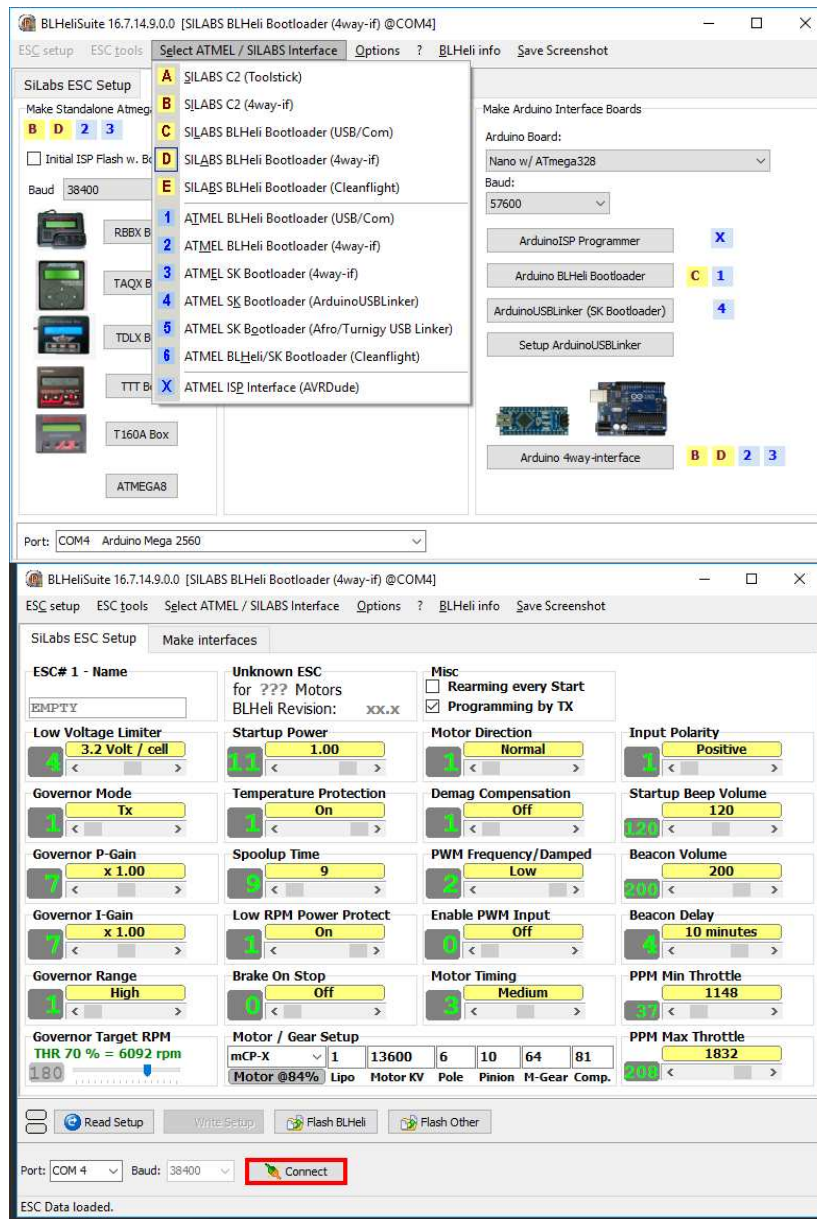
1. Запустите программу BLHeliSuite и выберите вкладку Make interfaces.



2. Подключите Arduino к компьютеру, при необходимости посмотрите в диспетчере устройств номер COM порта, к которому подключена плата.
3. Нажмите Arduino 4way-interface в разделе Make Arduino Interface Boards и выберите файл прошивки. После выбора файла начнётся прошивка контроллера.

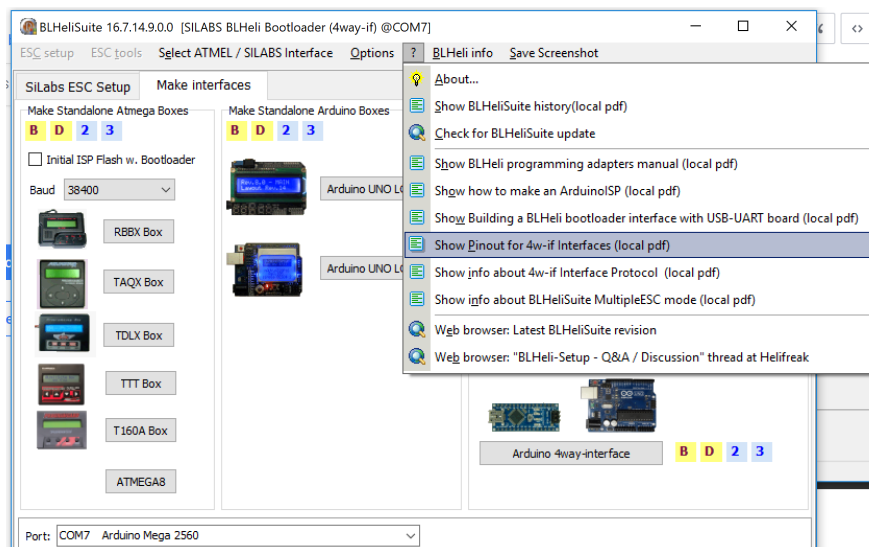


4. После прошивки Arduino вернитесь на вкладку Silabs ESC Setup и подключитесь к Arduino, предварительно выбрав интерфейс программатора 4way-if и COM порт Arduino.



Подключение ESC регуляторов к Arduino

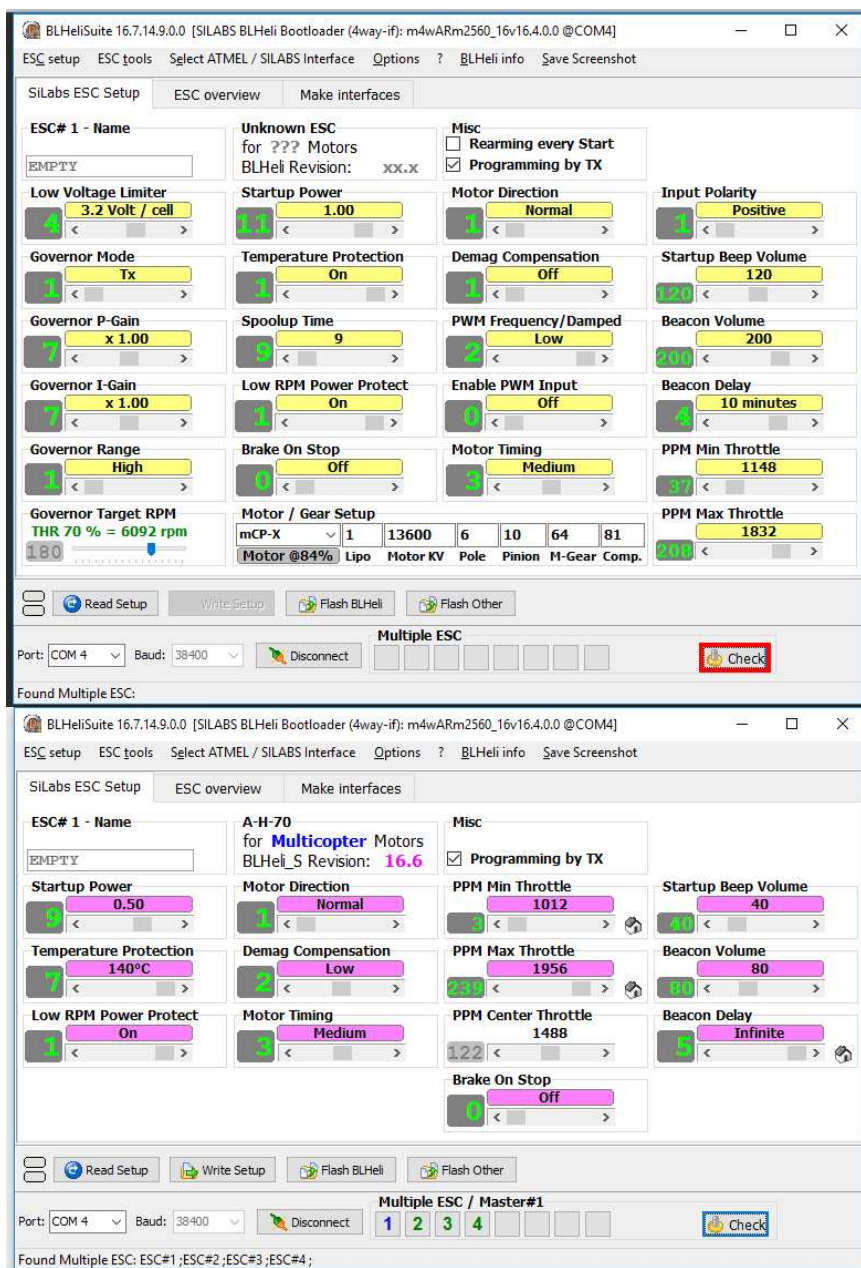
Для прошивки или изменения настроек регуляторов необходимо подключить сигнальные порты (обычно белого цвета) ESC регуляторов к портам Arduino, предварительно посмотрев в мануале (см. рисунок ниже), какие порты используются для соединения с регуляторами. Так же нужно соединить GND Arduino с землёй одного из регуляторов (обычно черного цвета). Регуляторы должны быть подключены к питанию, а если к регуляторам подключены моторы, **на них не должно быть пропеллеров.**



В случае с Arduino Мeга, сигнальные порты регуляторов подключаются к портам D43-D49 и D51.

Изменение настроек ESC регуляторов

Для загрузки информации о версии прошивки и настроек регуляторов нужно нажать на кнопку Check.

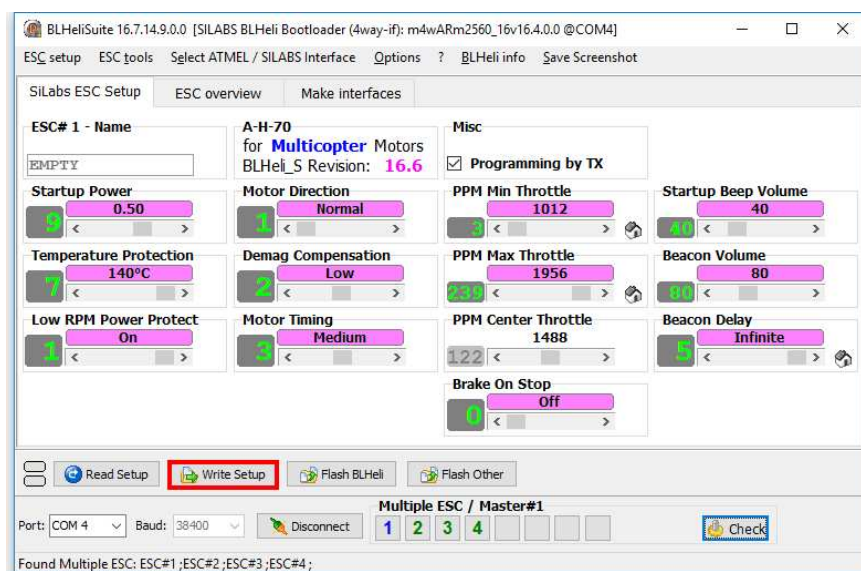


Основные параметры, которые нас интересуют, это:

- Motor Direction (Normal или Reversed) - отвечает за направление вращения моторов. Удобно настраивать, если нет желания перепаявать неправильно припаянный мотор.
- PPM Min и Max Throttle - отвечает за минимальный и максимальный сигнал газа
- Startup Beep Volume - громкость стартового сигнала. В версии прошивки 16.65 добавлена возможность изменения стартовой мелодии. Подробнее об этом написано [здесь](#). Например, можно в качестве мелодии запуска установить имперский марш из Звёздных Войн или главную тему Игры престолов.
- Beacon Volume - громкость обнаруживающего сигнала. Когда моторы не крутятся некоторое время и регулятор не используется, он начинает напоминать о себе писком моторов.

- Beacon Delay - время бездействия, после которого включается обнаруживающий сигнал. При разработке он может хорошенько надоедать, поэтому его можно выставлять в бесконечность.

Самый левый мотор в списке моторов (Multiple ESC) считается главным (мастер). Нажимая на номера моторов, можно включать/выключать возможность записи в них настроек. После изменения необходимых параметров можно записать в нужные моторы настройки, нажав на кнопку Write Setup.



Для отображения настроек со всех регуляторов одновременно можно воспользоваться вкладкой ESC Overview.

Прошивка ESC регуляторов

Файлы с прошивками регуляторов находятся [здесь](#).

Для перепрошивки регуляторов нужно нажать на кнопку Flash BLHeli и выбрать файл прошивки с типом контроллера, название которого указано в рамке названия прошивки и находится сверху во вкладке Silabs ESC Setup.

Для перепрошивки отдельного регулятора нужно сделать все остальные неактивными.

Видеоинструкция по перепрошивке ESC регуляторов

Для лучшего понимания того, что описано в статье, рекомендуем посмотреть наглядное руководство по подключению электроники и прошивке регуляторов на английском языке на [youtube](#).

Управление дроном с Arduino

Для взаимодействия с ROS-топиками и сервисами на Raspberry Pi можно использовать библиотеку [roserial_arduino](#). Эта библиотека предустановлена на [образе для Raspberry Pi](#).

Основной tutorial по roserial: http://wiki.ros.org/roserial_arduino/Tutorials

Arduino необходимо установить на дрон и подключить по USB-порту.

Настройка Arduino IDE

Для работы с ROS Arduino необходимо понимать формат сообщений установленных пакетов. Для этого [на Raspberry Pi](#) необходимо собрать библиотеку ROS-сообщений:

```
roslaunch roserial_arduino make_libraries.py .
```

Полученный каталог `ros_lib` необходимо скопировать в <папку скетчей>/libraries на компьютере с Arduino IDE.

Настройка Raspberry Pi

Для запуска `roserial` создайте файл `arduino.launch` в каталоге `~/catkin_ws/src/drone/drone/launch/` со следующим содержимым:

```
<launch>
  <node pkg="roserial_python" type="serial_node.py" name="serial_node" output="screen" />
  <param name="port" value="/dev/serial/by-id/usb-1a86_USB2.0-Serial-if00" />
</node>
</launch>
```

Чтобы единоразово запустить программу на Arduino, можно будет воспользоваться командой:

```
roslaunch drone arduino.launch
```

Чтобы запускать связку с Arduino при старте системы автоматически, необходимо добавить запуск созданного launch-файла в основной launch-файл (`~/catkin_ws/src/drone/drone/launch/drone.launch`). Добавьте в конец этого файла строку:

```
<include file="$(find drone)/launch/arduino.launch"/>
```

При изменении launch-файла необходимо перезапустить пакет `drone` :

```
sudo systemctl restart drone
```

Задержки

При использовании `rosserial_arduino` микроконтроллер Arduino не должен быть заблокирован больше чем на несколько секунд (например, с использованием функции `delay`); иначе связь между Raspberry Pi и Arduino будет разорвана.

При реализации долгих циклов `while` обеспечьте периодический вызов функции `nh.spinOnce`:

```
while(/* условие */) {  
    // ... Произвести необходимые действия  
    nh.spinOnce();  
}
```

Для организации долгих задержек используйте задержки в цикле с периодическим вызовом функции `nh.spinOnce`:

```
// Задержка на 8 секунд  
for(int i=0; i<8; i++) {  
    delay(1000);  
    nh.spinOnce();  
}
```

Работа с дроном

Набор сервисов и топиков аналогичен обычному набору в [simple_offboard](#).

Пример программы, контролирующей дрон по позиции, с использованием сервисов `navigate` и `set_mode`:


```

// Подключение библиотек для работы с roserial
#include <ros.h>

// Подключение заголовочных файлов сообщений пакета drone и MAVROS
#include <drone/Navigate.h>
#include <mavros_msgs/SetMode.h>

using namespace drone;
using namespace mavros_msgs;

ros::NodeHandle nh;

// Объявление сервисов
ros::ServiceClient<Navigate::Request, Navigate::Response> navigate("/navigate");
ros::ServiceClient<SetMode::Request, SetMode::Response> setMode("/mavros/set_mode");

void setup()
{
  // Инициализация roserial
  nh.initNode();

  // Инициализация сервисов
  nh.serviceClient(navigate);
  nh.serviceClient(setMode);

  // Ожидание подключения к Raspberry Pi
  while(!nh.connected()) nh.spinOnce();
  nh.loginfo("Startup complete");

  // Пользовательская настройка
  // <...>

  // Тестовая программа
  Navigate::Request nav_req;
  Navigate::Response nav_res;
  SetMode::Request sm_req;
  SetMode::Response sm_res;

  // Взлет на 2 метра:
  nh.loginfo("Take off");
  nav_req.auto_arm = false;
  nav_req.x = 0;
  nav_req.y = 0;
  nav_req.z = 2;
  nav_req.frame_id = "body";
  nav_req.speed = 0.5;
  navigate.call(nav_req, nav_res);

  // Ждем 5 секунд
  for(int i=0; i<5; i++) {
    delay(1000);
    nh.spinOnce();
  }

  nav_req.auto_arm = false;

  // Пролет вперед на 3 метра:
  nh.loginfo("Fly forward");
  nav_req.auto_arm = true;
  nav_req.x = 3;
  nav_req.y = 0;
  nav_req.z = 0;
  nav_req.frame_id = "body";
  nav_req.speed = 0.8;
  navigate.call(nav_req, nav_res);
}

```

```
// Ждем 5 секунд
for(int i=0; i<5; i++) {
  delay(1000);
  nh.spinOnce();
}

// Полет в точку 1:0:2 по маркерному полю
nh.loginfo("Fly on point");
nav_req.auto_arm = false;
nav_req.x = 1;
nav_req.y = 0;
nav_req.z = 2;
nav_req.frame_id = "aruco_map";
nav_req.speed = 0.8;
navigate.call(nav_req, nav_res);

// Ждем 5 секунд
for(int i=0; i<5; i++) {
  delay(1000);
  nh.spinOnce();
}

// Посадка
nh.loginfo("Land");
sm_req.custom_mode = "AUTO.LAND";
setMode.call(sm_req, sm_res);
}

void loop()
{
}
```

Получение телеметрии

С Arduino можно использовать сервис `get_telemetry`. Для этого надо объявить его по аналогии с сервисами `navigate` и `set_mode`:

```
#include <ros.h>

// ...

#include <drone/GetTelemetry.h>

// ...

ros::ServiceClient<GetTelemetry::Request, GetTelemetry::Response> getTelemetryClient;

// ...

nh.serviceClient(getTelemetryClient);

// ...

GetTelemetry::Request gt_req;
GetTelemetry::Response gt_res;

// ...

gt_req.frame_id = "aruco_map"; // фрейм для значений x, y, z
getTelemetryClient.call(gt_req, gt_res);

// gt_res.x - положение дрона по x
// gt_res.y - положение дрона по y
// gt_res.z - положение дрона по z
```

Проблемы

При использовании Arduino Nano может не хватать оперативной памяти (RAM). В таком случае в Arduino IDE будут появляться сообщения, типа:

```
Глобальные переменные используют 1837 байт (89%) динамической памяти, оставляя
Недостаточно памяти, программа может работать нестабильно.
```

Можно сократить использование оперативной памяти уменьшив размер выделяемых буферов для передачи и приема сообщений. Для этого **в самое начало** программы следует поместить строку:

```
#define __AVR_ATmega168__ 1
```

Можно уменьшить количество занятой памяти еще сильнее, если вручную настроить количество publisher'ов и subscriber'ов, а также размеры буферов памяти, выделяемой для сообщений, например:

Датчик температуры и влажности

```
#include <ros.h>

// ...

typedef ros::NodeHandle_<ArduinoHardware, 3, 3, 100, 100> NodeHandle;

// ...
NodeHandle nh;
```

Подключение GPS

При подключении GPS появляются следующие возможности:

- удерживание коптером позиции при полете на улице;
- программирование автономных миссий в программе QGroundControl;
- полеты на глобальные точки в автономном режиме при помощи модуля [simple_offboard](#).

Подключение

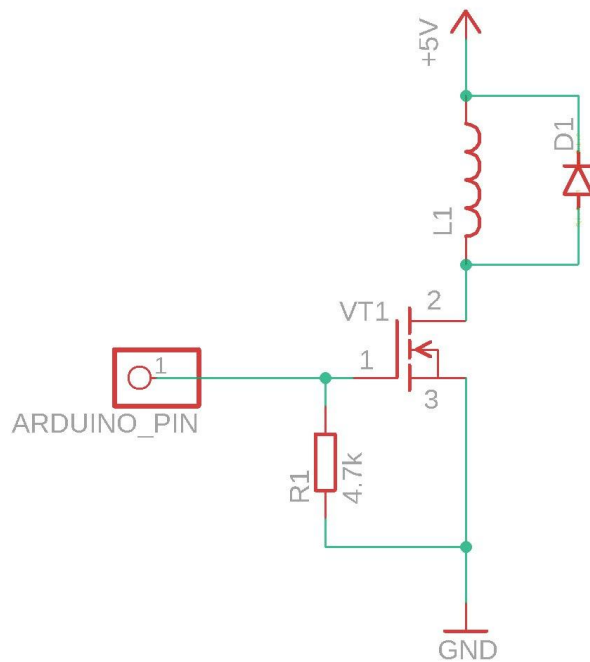
GPS-модуль подключается к разъемам "GPS" и "I2C" (компас) полетного контроллера.

При подключении GPS необходимо заново откалибровать магнитометры в программе QGroundControl, подключившись по [Wi-Fi](#) или USB.

Далее, необходимо включить GPS в параметре `EKF2_AID_MASK` (при использовании EKF2) или `LPE_FUSION` (при использовании LPE). При использовании LPE вес компаса должен быть больше нуля (`ATT_W_MAG = 0.1`).

Сборка и настройка электромагнитного захвата

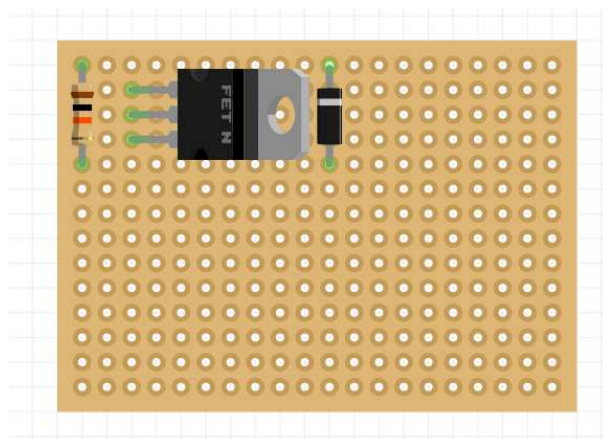
Магнитный захват можно собрать различными способами в соответствии с электрической схемой.



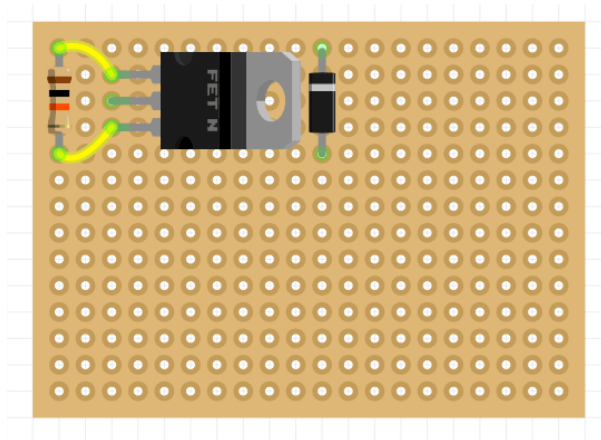
Ниже представлен пример сборки схемы электромагнитного захвата на макетной плате.

Рекомендуется проложить проводку между элементами с обратной стороны платы (на дальнейших изображениях проводка сделана поверх схемы, для наглядности).

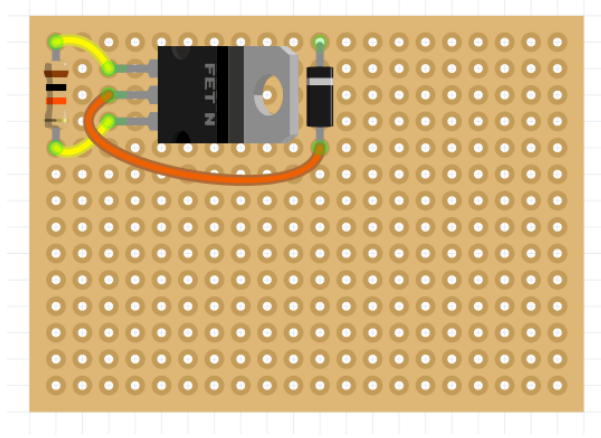
1. На паечной плате разместите диод Шоттки, резистор на 10 кОм и транзистор.



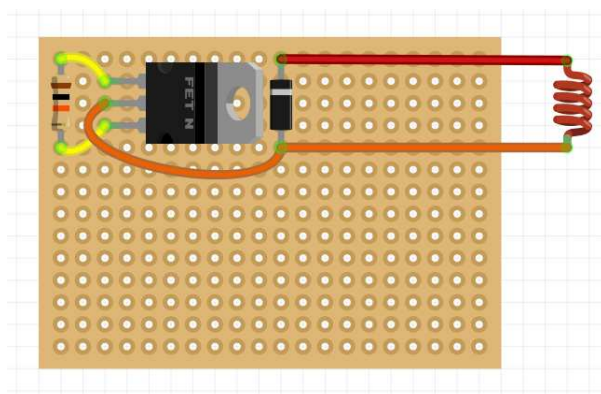
2. Припаяйте контакты с другой стороны платы и откусите оставшиеся ножки элементов.
3. Соедините контакты резистора и двух крайних ножек транзистора.



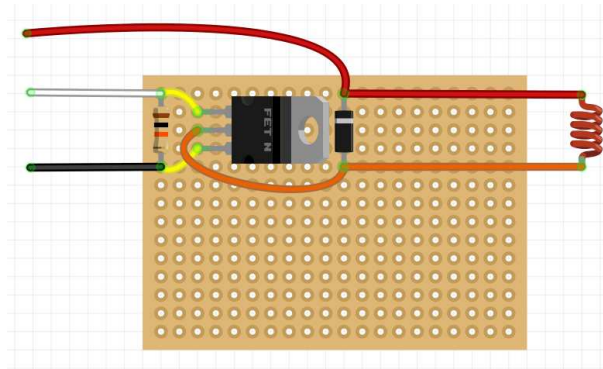
4. Соедините центральную ножку транзистора и ножку диода Шоттки (противоположную серой маркировочной полоске).



5. Обрежьте необходимое количество провода магнитного захвата и припаяйте его к контактам диода Шоттки.

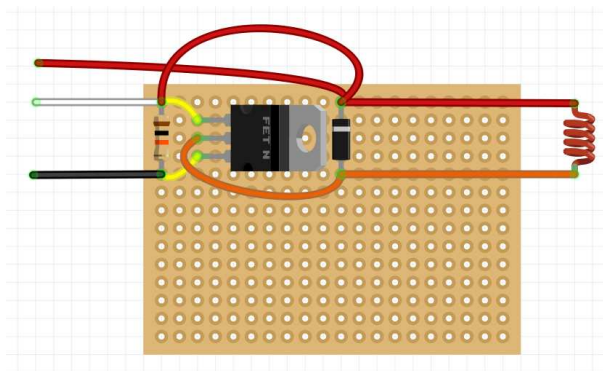


6. Припаяйте провода *Dupont*-папа к ножке транзистора и диода (красный, черный провода), а также провод *Dupont*-мама на противоположную ножку транзистора (белый провод).



Проверка работы электромагнитного захвата

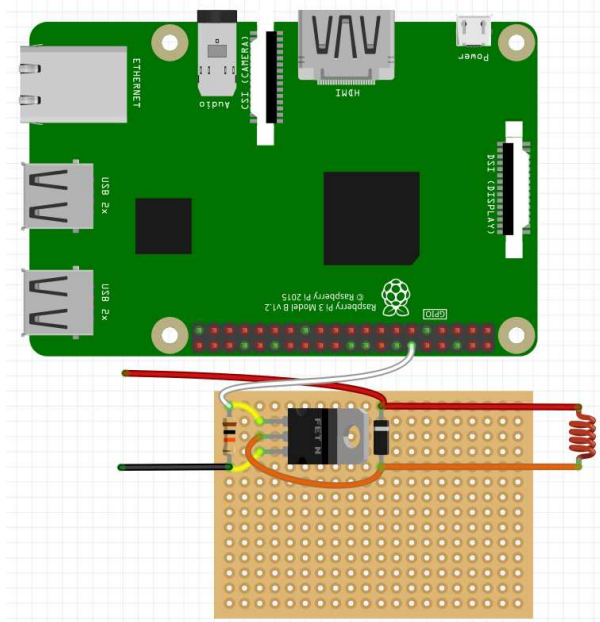
Для того, чтобы проверить работу захвата, подайте на сигнальный провод напряжение 5В. Для этого можно использовать провод *Dupont* папа-папа.



После подачи напряжения магнит должен включиться.

Подключение к Raspberry Pi

Подключите магнитный захват к плате Raspberry Pi для программного использования

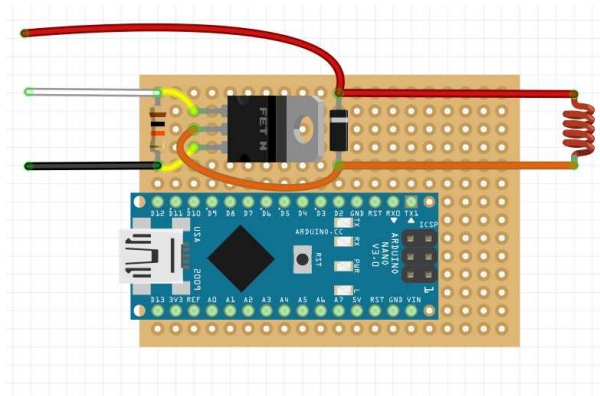


Пример кода, активирующего магнитный захват, можно посмотреть [тут](#).

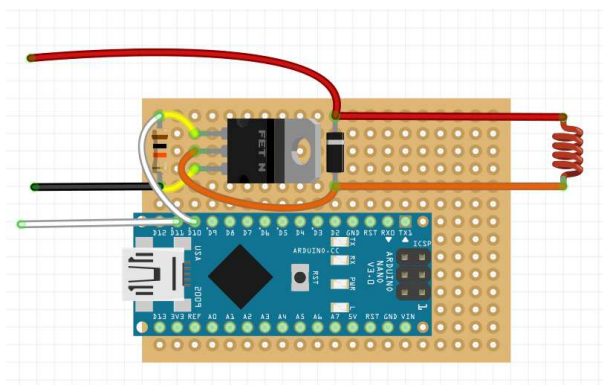
Подключение к Arduino

Подключите захват плате Arduino Nano, чтобы использовать его в ручном режиме.

Удобно ее располагать на той же паечной плате -- вставьте ее в подходящие отверстия и припаяйте с обратной стороны к плате.



Затем подключите сигнальный выход схемы к выбранному порту и припаяйте провод *Dupont*-мама к выбранному сигнальному порту на плате.



Установка электромагнитного захвата

1. В центральное отверстие на деке захвата установите электромагнит.
2. Стяжкой притяните собранную схему к обратной стороне деки.
3. Сигнальный вывод Arduino *D11* вставьте в один из выводов *AUX* на полетном контроллере.
4. Вставьте силовой вывод электромагнитного захвата в *JST 5B*.

Настройка электромагнитного захвата

Для управления магнитом через плату Arduino Nano, используйте код ниже:

```
void setup() {  
  pinMode(11, INPUT);  
  pinMode(13, OUTPUT);  
}  
  
void loop() {  
  if (int duration = pulseIn(11, HIGH) > 1200) {  
    digitalWrite(13, HIGH);  
  } else {  
    digitalWrite(13, LOW);  
  }  
}
```

Для управления магнитным захватом со светодиодной лентой через плату Arduino Nano используйте код ниже:

```
#include <Adafruit_NeoPixel.h>
#define NUMPIXELS 72
#define PIN 12
int pin = 11;
int led = 13;

unsigned long duration;
Adafruit_NeoPixel strip (NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);

void setup() {
  strip.begin();
  strip.setBrightness(10);
  Serial.begin(9600);
  pinMode(pin, INPUT);
  pinMode(led, OUTPUT);
}

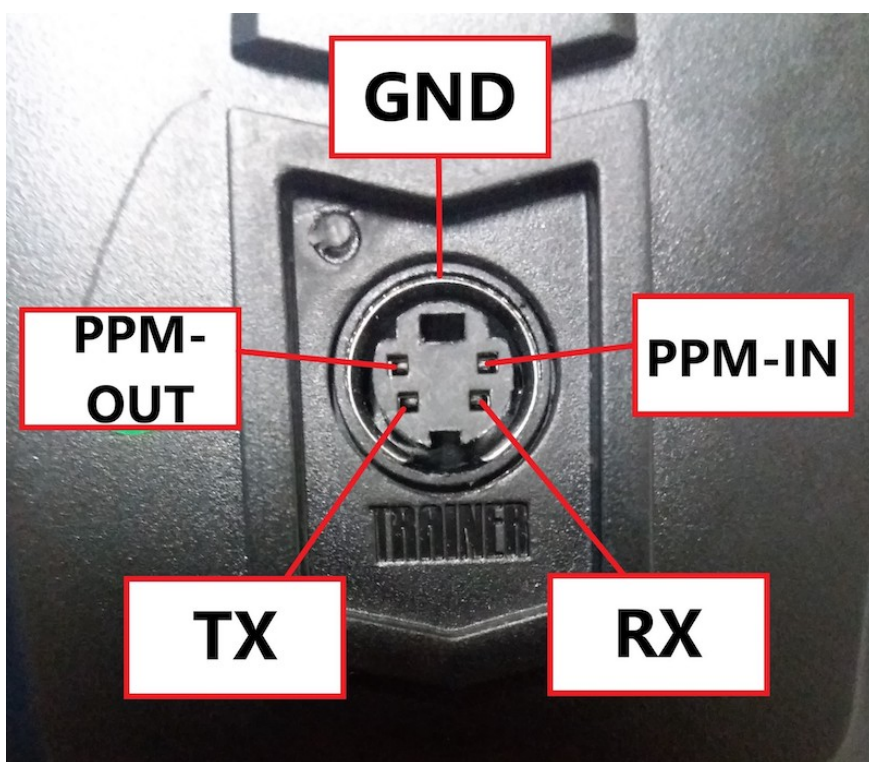
void loop() {
  duration = pulseIn(pin, HIGH);
  Serial.println(duration);
  delay(100);
  if (duration >= 1500) {
    digitalWrite(led, HIGH);
    for (int i = -1; i < NUMPIXELS; i++) {
      strip.setPixelColor(i, strip.Color(255, 0, 0));
      strip.show();
    }
  } else {
    digitalWrite(led, LOW);
    for (int i = -1; i < NUMPIXELS; i++) {
      strip.setPixelColor(i, strip.Color(0, 255, 0));
      strip.show();
    }
  }
}
```

Настройка режима тренера на пульте FlySky

Режим тренера используют для обучения пилотированию. Чтобы во время обучения ученик ничего не повредил, на перехвате стоит опытный пилот, чтобы при необходимости взять управление на себя.

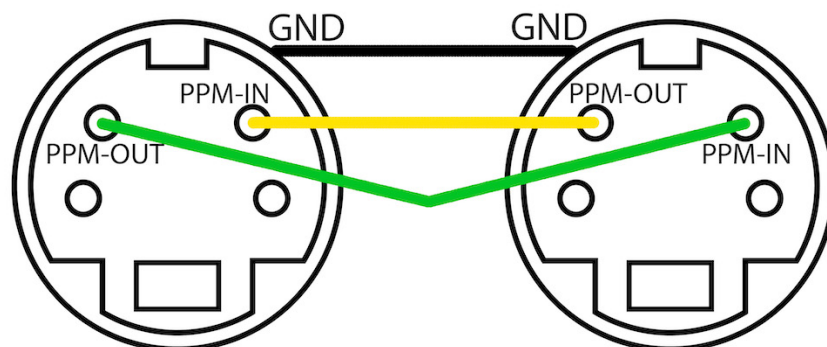
Для этого при помощи провода соединяют два пульта: один из них - пульт тренера, а другой – ученика.

Схема провода



Для создания канала общения между пультами используется разъём сзади корпуса (S-Video). Три контакта в разъёме отводятся для приёма, передачи информации и заземления. Контакт PPM-OUT (передача) должен быть соединён с PPM-IN (приём) и наоборот. Во избежание помех внешней среды контакты ground должны быть соединены между собой.

Таким образом нам нужно припаять провода к разъёму.



Настройка пульта учителя

Зайдите в настройки (зажмите кнопку ОК). Далее зайдите в системные настройки (System setup) и найдите (Up/Down) режим тренера (Trainer mode).

Активируйте режим: для этого в строке Mode должен стоять параметр On. Чтобы изменить параметр, используйте кнопки Up/Down. Чтобы сохранить параметр, нажмите ОК.

Теперь выбираем переключатель для передачи управления.

Это можно сделать в меню режима (Trainer mode). В строке Switch выберите (менять можно с помощью Up/Down) любую удобную клавишу (SwA, SwB, SwC, SwD). С помощью этой клавиши вы сможете перехватить управление. Убедитесь, что на эту клавишу не назначены другие функции.

Чтобы сохранить настройки зажмите Cancel.

На пульте учителя должен стоять тот же режим полёта, что и на пульте ученика.

Настройка пульта ученика

Зайдите в настройки, System setup и выберите режим ученика (Student mode). Далее нажмите ОК и при помощи Up/Down выберите подтверждение (Yes).

Зажмите Cancel, чтобы сохранить настройки.

Если всё настроено правильно, на главном экране появится буква S.

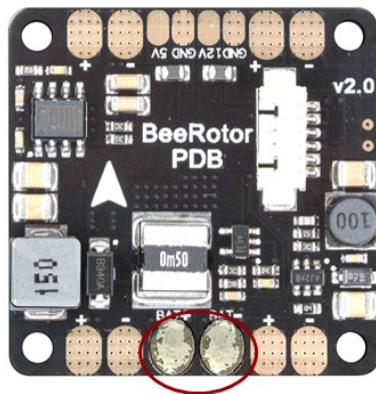
Лужение

Перед пайкой и лужением убедитесь, что провода и платы отключены от питания (обесточены)!

Лужение контактных площадок

Залудить контактную площадку значит:

1. Нанести флюс на контактную площадку.
2. Покрыть припоем контактную площадку.



1. Залудить контактные площадки VAT+ и VAT-
2. Залудить остальные контактные площадки

Лужение проводов

Залудить провод значит:

1. Зачистить провод — снять слой изоляции.
2. Скрутить оголенные провода.
3. Нанести флюс на скрученные оголенные провода.
4. Покрыть слоем припоя.



Типы силовых разъемов

XT-60

Один из самых надёжных силовых разъёмов, которые стараются применять на силовых аккумуляторах. Именно такие аккумуляторы используются на Li-Po аккумуляторах для дронов.



T-plug

Аналог XT-60. Имеет различные вариации для упрощения разъединения.



JST-XH или балансировочный разъем

Разъёмы данного типа часто применяются для балансировки отдельных элементов в составе сборки из нескольких литий-полимерных (Li-Pol), литий-ионных (Li-ion) или литий-фосфатных (LiFePO4) аккумуляторов. Подобные разъёмы с разным количеством штырьков устанавливаются в большинство современных зарядных устройств для балансировки литиевых элементов при заряде. Может использоваться в сочетании с Buzzer (пищалкой) для контроля заряда аккумулятора.



Gold Bullet Connector или "Бананы"

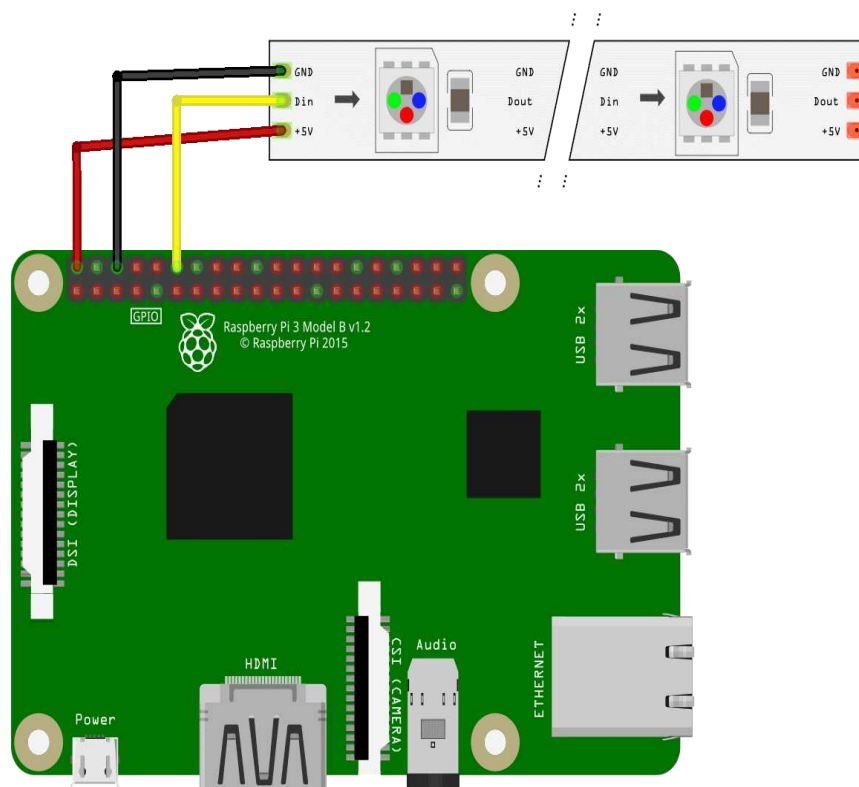
Существует великое множество штырьковых разъемов типа Gold Bullet Connector. Разъемы данного типа отличаются друг от друга диаметром и размером. Наиболее распространены разъемы с диаметром коннектора 2 мм, 3 мм и 4 мм. Часто используется для создания беспаячных соединений на PDB и моторах.



Работа со светодиодной лентой на Raspberry 3

Подключение и определение типа ленты

Подключите светодиодную ленту к питанию +5v - 5v, земле GND - GND и сигнальному порту DIN - GPIO30, **GPIO21**, GPIO18. По умолчанию нужно подключать к **GPIO21**.



Обратите внимание, что светодиодную ленту нужно питать от стабильного источника энергии. Если вы подключите питание напрямую к Raspberry, то это создаст слишком большую нагрузку на ваш микрокомпьютер. Для снятия нагрузки с Raspberry можно подключить питание к преобразователю `WES`.

При работе с [GPIO](#) следует подключать ленту к пину GPIO21. В противном случае управление LED-лентой будет работать некорректно.

Совместимость с ROS и Python

Чтобы корректно работать со светодиодной лентой вам нужно добавить в окружение необходимые пути к библиотекам Python и пакетам ROS, для этого необходимо добавить в файл `/etc/sudoers` следующие строки:

```
Defaults    env_keep += "PYTHONPATH"
Defaults    env_keep += "PATH"
Defaults    env_keep += "ROS_ROOT"
Defaults    env_keep += "ROS_MASTER_URI"
Defaults    env_keep += "ROS_PACKAGE_PATH"
Defaults    env_keep += "ROS_LOCATIONS"
Defaults    env_keep += "ROS_HOME"
Defaults    env_keep += "ROS_LOG_DIR"
```

Пример программы для светодиодной ленты на RPi3

Для проверки работоспособности ленты можете использовать приведенный ниже код, данный код поочередно зажжет первые 10 диодов 3 цветами и в конце их погасит.

```
import time

from rpi_ws281x import Adafruit_NeoPixel
from rpi_ws281x import Color

LED_COUNT    = 10      # Количество светодиодов в ленте
LED_PIN      = 21      # GPIO пин, к которому вы подключаете светодиодную ленту
LED_FREQ_HZ  = 800000  # Частота несущего сигнала (обычно 800 кГц)
LED_DMA      = 10      # DMA-канал для генерации сигнала (обычно 10)
LED_BRIGHTNESS = 255  # Яркость: 0 - наименьшая, 255 - наибольшая
LED_INVERT   = False   # True для инвертирования сигнала (для подключения через транзистор)
LED_CHANNEL  = 0       # '1' для GPIO 13, 19, 41, 45 или 53

strip = Adafruit_NeoPixel(LED_COUNT, LED_PIN, LED_FREQ_HZ, LED_DMA, LED_INVERT, LED_CHANNEL)

strip.begin()

def colorWipe(strip, color, wait_ms=50):
    """Заполнение ленты цветом по одному светодиоду."""
    for i in range(strip.numPixels()):
        strip.setPixelColor(i, color)
        strip.show()
        time.sleep(wait_ms/1000.0)

print('Color wipe animations.')
colorWipe(strip, Color(255, 0, 0), wait_ms=100) # Заполнение красным
colorWipe(strip, Color(0, 255, 0), wait_ms=100) # Заполнение зелёным
colorWipe(strip, Color(0, 0, 255), wait_ms=100) # Заполнение синим
colorWipe(strip, Color(0, 0, 0), wait_ms=100)  # Выключение ленты
```

Вы так же можете использовать тестовый код разработчиков данного модуля. Вы можете его [скачать](#) из репозитория разработчика.

Сохраните программу в ваш скрипт и запустите его используя права администратора:

```
sudo python YourScriptName.py
```

Основные функции используемые для работы со светодиодной лентой

Для подключения библиотеки и её корректной работы требуется подключить следующие модули: `Adafruit_NeoPixel` и `Color` - для работы ленты и `time` - для управления задержками.

```
from rpi_ws281x import Adafruit_NeoPixel
from rpi_ws281x import Color
import time
```

Для работы с лентой необходимо создать объект типа `Adafruit_NeoPixel` и инициализировать библиотеку:

```
# Создание объекта NeoPixel с заданной конфигурацией
strip = Adafruit_NeoPixel(LED_COUNT, LED_PIN, LED_FREQ_HZ, LED_DMA, LED_INVERT)
# Инициализация библиотеки, должна быть выполнена перед другими функциями
strip.begin()
```

Основные функции, которые используются для управления лентой:

- `numPixels()` - возвращает количество пикселей в ленте. Удобно для циклического управления всей лентой целиком.
- `setPixelColor(pos, color)` - устанавливает цвет пикселя в позиции `pos` в цвет `color`. Цвет должен быть 24 битным значением, где первые 8 бит - красный цвет (`red`), следующие 8 бит - зелёный цвет (`green`) и последние 8 бит - голубой (`blue`). Для получения значения `color` можно использовать функцию `Color(red, green, blue)`, которая составляет это значение из 3х компонент. Каждый компонент должен находиться в диапазоне 0-255, где 0 - отсутствие цвета, а 255 - наибольшая доступная яркость компонента в светодиодном модуле.
- `setPixelColorRGB(pos, red, green, blue)` - устанавливает цвет пикселя в позиции `pos` в цвет, состоящий из компонент `red`, `green`, `blue`. Каждый компонент должен находиться в диапазоне 0-255, где 0 - отсутствие цвета, а 255 - наибольшая доступная яркость компонента в светодиодном модуле.
- `show()` - обновляет состояние ленты. Только после её использования все программные изменения перемещаются на светодиодную ленту.

Почему именно так и можно ли по-другому?

Лента управляется по принципу: для массива светодиодов в ленте отправляется пакет данных по 24 бита на светодиод; каждый светодиод считывает первые 24 бита из пришедших к нему данных и устанавливает соответствующий цвет, остальные данные он отправляет следующему светодиоду в ленте. Нули и единицы задаются разными сочетаниями длительностей высокого и низкого уровня в импульсе.

Используемый тип ленты поддерживаются для управления библиотекой `rpi_ws281x`, при этом для управления используется модуль DMA (direct memory access) процессора Raspberry Pi и один из каналов передачи данных: PWM, PCM или SPI, что гарантирует отсутствие задержек в управлении, а управляется всё на многозадачной операционке, это важно.

Есть некоторые особенности работы с каналами, например при передаче данных с помощью PWM (ШИМ) перестаёт работать встроенная аудиосистема Raspberry Pi, при передаче данных по PCM блокируется использование подключенных цифровых аудиоустройств (при этом встроенная система работает), а при использовании SPI кстати, требуется специальная настройка размера буфера и частоты GPU Raspberry Pi для правильной работы, лента блокирует все остальные устройства, подключенные по этому каналу.

Есть некоторые особенности выбора канала DMA для управления лентой: некоторые каналы используются системой, поэтому их использование может привести к неприятным последствиям, например использование 5 канала рушит файловую систему Raspberry, т.к. этот канал используется при чтении-записи на SD карту. Безопасный канал – 10, он же установлен по умолчанию в приведённой выше библиотеке.

Поэтому сценарии использования LED-ленты следующие:

1. Если нам не важна работоспособность встроенного аудио на Raspberry Pi и мы его не используем, т. к. аудио и лента будут выдавать белиберду в этом случае, то можно использовать PWM канал для этого требуется подключить вход ленты к одному из следующих GPIO портов Raspberry Pi: 12, 18, 40, или 52 для PWM0 канала и 13, 19, 41, 45 или 53 для PWM1 канала.
2. Если нам не важно наличие на шине SPI других устройств, то можно управлять лентой по каналу SPI (GPIO на Raspberry Pi 10 или 38). Здесь требуется произвести следующие настройки (только для Raspberry Pi 3):
 - o увеличить размер буфера передачи данных для поддержки длинных лент, добавив строку `spidev.bufsiz=32768` в файл `/boot/cmdline.txt` ;
 - o установить частоту GPU для правильной частоты работы SPI, добавив строку `core_freq=250` в файл `/boot/config.txt` .
 - o перезагрузить вашу Raspberry, используя команду `sudo reboot`
3. Если нам важна и работа аудио, и подключение к SPI устройств кроме лед ленты, то можно управлять лентой по каналу PCM (GPIO 21 или 31). При этом никаких дополнительных манипуляций с Raspberry не требуется.

Исходя из вышеперечисленных способов управления лентой, наилучшим вариантом, позволяющим управлять лентой, сохранить работоспособность встроенной аудиосистемы и возможность подключения всяческих устройств и датчиков по SPI, является управление по каналу PCM (GPIO 21) с использованием 10 канала DMA.

Методические материалы

Здесь представлены методические материалы для ознакомления с основами БПЛА, которые помогут грамотно организовать процесс как самостоятельного изучения, так и в группах.

Введение

Данное методическое пособие призвано помочь преподавателю в обучении людей, желающих освоить сферу применения беспилотных летательных аппаратов и получить практические навыки в конструировании, пилотировании, настройке и программировании беспилотных летательных аппаратов.

Предполагаемый возраст обучающихся - не менее 12 лет (ожидается, что учащийся в этом возрасте уже имеет минимальный опыт работы с инструментом, компьютером и осознанно подходит к процессу обучения, с соблюдением всех правил техники безопасности).

Целью обучения является формирование компетенций в области беспилотных авиационных систем, развитие творческого и научно-технического потенциала учащихся, путем организации проектной деятельности, в рамках создания собственного беспилотного летательного аппарата. Предметными результатами изучения данного курса является формирование следующих знаний и умений:

Знать:

- об истории и тенденциях развития беспилотных летательных аппаратов; о том как можно улучшить их характеристики;
- правила техники безопасности при эксплуатации БПЛА;
- основные компоненты коптеров;
- конструктивные особенности различных моделей, сооружений и механизмов;
- компьютерные среды для настройки полетных контроллеров;
- основы аэродинамики полета;
- основы электричества, радиоэлектроники;
- основы программирования на языке Python;
- основы программирования автономных полетов коптеров;
- теорию FPV полетов;
- применение компьютерного зрения;
- конструктивные особенности различных БПЛА и их применение;
- способы настройки и подготовки коптера к полету;

Уметь:

- настраивать и калибровать полетные контроллеры разных производителей с применением специализированного ПО;

- осуществлять визуальное пилотирование беспилотного летательного аппарата и посредством FPV аппаратуры;
- создавать недостающие для реализации проектов элементы в средах 3D моделирования и осуществлять их печать на 3D принтере;
- взаимодействовать с микрокомпьютером Raspberry, обладать основами администрирования Linux;
- планировать и прописывать полетные задания и миссии;
- программировать и осуществлять автономные полеты. Проводить предполетную подготовку.

Перечень используемого оборудования и материалов:

1. Компьютеры с установленным необходимым ПО.
2. Наборы конструкторов для сборки программируемого дрона.
3. Лаборатория, оснащенная паяльными станциями, вытяжками и необходимыми инструментами.
4. Полетная зона или разрешение на полеты в открытом воздушном пространстве.

Промежуточный контроль

Промежуточный контроль знаний проводится путем проверки теоретических знаний с использованием контрольных вопросов и тестирования по каждой теме. Тестирования проводятся наравне с опросами по пройденным темам в начале занятия. Контрольные вопросы направлены на оценку степени освоения учащимися материала, пройденного на текущем занятии и задаются в конце занятия по каждой теме.

Темы

1. «Знакомство. Принципы проектирования и строение мультикоптеров»
2. «Основы электричества»
3. «Теория пайки»
4. «Аэродинамика полета. Пропеллер»
5. «Основы электромагнетизма. Типы двигателей»
6. «Бесколлекторные двигатели и регуляторы их хода»
7. «Принцип работы, типы и устройство аккумуляторов»
8. «Управление полетом мультикоптера. Принцип функционирования полетного контроллера. ПИД регуляторы»
9. «Основы радиосвязи. Принцип работы радиоаппаратуры управления»
10. «Аналоговая и цифровая видеотрансляция. Применяемые камеры, радиопередатчики и приемники»
11. «Техника безопасности при сборке и настройке коптеров, при подготовке к вылету. Техника безопасности при работе с аккумуляторами»
12. «Теория ручного визуального пилотирования»
13. «Техника безопасности при летной эксплуатации коптера»
14. «Обучение лётному мастерству»

15. «Основы радиоэлектроники, схемотехники и макетирования электрических схем»
16. «Основы работы с аналоговым и цифровым сигналом»
17. «Основы работы с лабораторным оборудованием»
18. «Теория FPV полетов»
19. «История автономных полетов. Развитие автопилотов в авиации»
20. «Основы программирование на языке Python»
21. «Знакомство с компьютером Raspberry Pi»
22. «Управление автономным дроном: теория»

Методика проведения занятий

Урок №1. Тема: «Знакомство. Принципы проектирования и строение мультикоптеров»

Цель урока: познакомить учеников с предстоящим курсом, сформировать учебную мотивацию, познакомить учеников с историей создания и развития коптеров, сформировать представления о функциях и возможностях современных коптеров, наглядно продемонстрировать виды и конфигурации коптеров, дать теоретические знания об управлении коптером и его составных частей, обучить основным терминам для качественного усвоения новых знаний.

№	Этап урока	Содержание
1	Знакомство	В начале занятия необходимо представиться классу.
2	Представление курса	Опрос учеников с какими ожиданиями они пришли на первое занятие, чего они хотят узнать и чему научиться в ближайшее время. Это поможет лучше познакомиться с классом, понять мотивацию детей. Необходимо прокомментировать их ответы, поддержать если ожидание ученика совпадает с программой курса. Дополнить ответы, перечислив интересные темы и практические занятия, которые ждут учеников в данном курсе.
3	Демонстрация современных коптеров	Чтобы ещё больше заинтересовать учеников, необходимо продемонстрировать им видео с современными коптерами, кадры с гонок, научных выставок, демонстраций.
4	История коптеров	<p>Опрос учеников, как давно, по их мнению, появились первые беспилотные летающие аппараты. Отметить учеников, которые были ближе всего в своих догадках, добавив что первые беспилотники появились ещё в 1890-х годах, например радиоуправляемая лодка Теслы (В 1899 году на выставке в Мэдисон-Сквер-Гарден инженер и изобретатель Никола Тесла продемонстрировал миниатюрное радиоуправляемое судно.).</p> <p>Продемонстрировать ученикам развитие беспилотных технологий:</p> <ul style="list-style-type: none"> • Kettering Bug («Жук» Каттеринга) (Экспериментальная беспилотная «воздушная торпеда», один из первых проектов предшественников современных крылатых ракет) • Вертолет Ботезата (Один из первых дронов (англ. quadcopter, четырехроторный вертолет), который реально оторвался от земли и мог держаться в воздухе, был создан Георгием Ботезатом и испытан в 1922 году). <p>Спросить учеников, какие недостатки, по их мнению, были у этих моделей и с какими трудностями сталкивались люди используя их. Показать ученикам современный беспилотник и современный коптер на фото или видео. Можно показать коптер, который учащиеся будут собирать во время обучения, чтобы у них сформировалось представление о результате, к которому они должны прийти после освоения курса.</p>

№	Этап урока	Содержание
5	Применение БПЛА самолетного типа	<p>Спросить у учеников, какие примеры использования БПЛА самолетного типа в современном мире они знают. Рассказать про преимущества применения БПЛА самолетного типа, особенно, про съемку с воздуха и мониторинг лесов и полей. Рассказать о сферах использования:</p> <ul style="list-style-type: none"> • Аэрофотосъемка местности (при необходимости отснять рельеф поверхности земли на протяженном расстоянии); • Военная сфера (тяжёлые БПЛА большой продолжительности полёта – взлётная масса более 1500 кг, дальность действия около 1500 км); • Сельское хозяйство (распространение удобрений, мониторинг полей); • Охрана и мониторинг местности. <p>Продемонстрировать свои примеры на фото и видео. Предложить ученикам привести ещё варианты, где могут применяться БПЛА самолетного типа.</p>
6	Применение коптеров	<p>Спросить у учеников, какие примеры использования коптеров в современном мире они знают. Постараться прокомментировать и дополнить каждый ответ. После чего задать ученикам вопросы, как, по их мнению, коптеры могут быть полезны в следующих областях:</p> <ul style="list-style-type: none"> • Телекоммуникационная сфера (использование двусторонней видео/аудио связи); • Транспортная сфера (транспортировка грузов); • Спасательные работы (исследование труднодоступных зон при стихийных бедствиях, терактах, поиск находящихся под завалами людей, транспортировка медикаментов, оказание первой помощи при несчастных случаях); • Рекламные кампании (применение робота для привлечения внимания на крупных выставках и т.п.); • Сфера СМИ (аэросъемка различных событий); • Видео/фото-съёмка (съёмка фильмов или рекламных роликов с высоты птичьего полёта); • Продовольственная сфера (доставка продуктов питания). <p>Продемонстрировать свои примеры на фото и видео. Предложить ученикам привести ещё варианты, где могут применяться коптеры. Поделиться своим собственным опытом использования коптеров, если есть такая возможность. Для эффектного завершения занятия можно показать ученикам фото и видео с гоночных соревнований дронов.</p>

№	Этап урока	Содержание
7	Виды и конфигурация дронов	<p>Рассказать ученикам о видах коптеров и зарисовать направление вращения моторов на каждом из них.</p> <ul style="list-style-type: none"> • трикоптер, • гексакоптер, • октокоптер, • квадрокоптер. <p>Спросить ребят, что, по их мнению, даёт разное количество пропеллеров, в конце сделав акцент на то, что квадрокоптеры самые простые из них в сборке и управлении. Показать ученикам два вида конфигурации дронов + и X и их отличие.</p>
8	Управление	<p>Показать классу пульт управления, перечислить название его управляющих элементов. Пусть ученики попробуют сами догадаться какое назначение у стиков и переключателей. Представить ученикам схему управления пультом. Познакомить их с необходимыми терминами и их значением:</p> <ul style="list-style-type: none"> • Throttle (газ). Газ мультикоптера – среднее арифметическое между скоростями вращения всех моторов. Чем больше газ, тем больше суммарная тяга моторов, и тем сильнее они тащат коптер вверх (проще говоря «Тапок в пол» здесь означает наискорейший подъем). Обычно измеряется в процентах: 0% – моторы остановлены, 100% – вращаются с максимальной скоростью. Газ висения – минимальный уровень газа, который необходим, чтобы коптер не терял высоту. • Оси коптера - углы тангажа, крена и рыскания (pitch, roll, yaw) – углы, которыми принято определять и задавать ориентацию мультикоптера в пространстве. • Yaw (рыскание). – «рыскание». Поворот носа мультикоптера. условно - вращение вправо влево. • Pitch (тангаж). В коптерах манипуляции с этим моментом силы позволяет коптеру двигаться вперед или назад за счет наклона носа в соответствующем направлении. • Roll (крен). Наклон мультикоптера влево вправо. Коптер за счет крена может двигаться боком в соответствующую сторону. <p>Рассказать классу принцип вращения пропеллеров, особенности направления их вращения. Продемонстрировать ученикам тестовую картинку, попросить их определить в каком направлении будет двигаться коптер. Позволить им самим свериться с правильными ответами и задать вопросы.</p>

№	Этап урока	Содержание
9	Элементы копитера	<p>Рассказать ученикам о составных элементах копитера:</p> <ul style="list-style-type: none"> • Полетный контроллер. Задача полетного контроллера – переводить команды от пульта управления в сигналы задающие обороты двигателя. Также в нем установлены инерциальные измерительные датчики, позволяющие следить за текущим положением платформы и выполнять автоматические регулировки. • ESC. В мультикопитерах используют специальные бесколлекторные электродвигатели, которые способны работать на очень больших оборотах. Для управления этими двигателями необходимо формировать трехфазное напряжение и относительно большие токи, чем и занимаются регуляторы оборотов. • Силовая рама • Электродвигатель • Пропеллеры • Радиоприемник • Радиопередатчик. Большинство передатчиков имеют частоту 2,4ГГц, так же на рынке представлен ряд других частот. • Аккумулятор. Самыми распространенными в применении на копитерах являются литий-ионные и литий-полимерные аккумуляторы. Основные характеристики аккумулятора (Емкость (мА*ч), Максимальный токоразряд (20С), Количество банок (2S,3S, 4S), Вес) <p>Зарисовать схему дрона и объяснить, как взаимодействуют между собой элементы. Спросить учеников, что, по их мнению, произойдет в случае отказа того или иного элемента.</p>
10	Резервное время	<p>Подведение итогов темы. Спросить у учеников, есть ли у них вопросы. Спросить, что из изученного на занятии было для них интереснее всего. Попросить учеников ответить на контрольные вопросы. Предложить ученикам по желанию провести в интернете дополнительное исследование на пройденную тему. Сообщить ученикам, какую тему они будут проходить на следующем занятии.</p>

Урок №2. Тема: «ОСНОВЫ ЭЛЕКТРИЧЕСТВА»

Цель урока: дать представления о природе и физических характеристиках электродвижущей силы. Формировать у учащихся представления о тепловом действии электрического тока и его причинах. Дать теоретическое представление ученикам о принципе строения электрических цепей.

№	Этап урока	Содержание
1	Повторение пройденного материала	Поприветствовать учеников. Спросить класс, кто хочет рассказать о том, что изучалось на прошлом уроке. Задать классу вопросы, по темам, которые не были озвучены учениками.
2	Введение	Спросить класс, кто может рассказать, что такое электрический ток. Выслушать ответы класса, чтобы узнать кто из учеников уже разбирается в этой теме. Отметить правильные ответы и, резюмировав их, перейти к объяснению нового материала. Рассказав классу о природе электродвижущей силы дать ученикам представление о значении терминов: • Разность потенциалов • Проводник • Диэлектрик • Электрический ток • Сопротивление • Величина тока. Для мотивации и актуализации внимания класса предложить ученикам, которые разбираются в этой теме, самим дать определение этих понятий, скорректировав их ответы при необходимости.
3	Закон Ома	Дать определение Закона Ома и вывести формулу. Объяснить значение её переменных. Для простоты применения формулы на практике продемонстрировать ученикам треугольник с формулой. Разобрать задачу с применением закона Ома. Спросить у класса есть ли у кого-то вопросы.
4	I закон Кирхгофа	Познакомить класс с понятиями узлов и ветвей электрической схемы. Сформулировать I закон Кирхгофа (сумма токов, подходящих к узловой точке электрической цепи, равна сумме токов, уходящих от этого узла), привести примеры его применения.
5	II закон Кирхгофа	Сформулировать II закон Кирхгофа (в замкнутом контуре электрической цепи сумма всех эдс равна сумме падения напряжения в сопротивлениях того же контура), приведя пример его формулы. Пояснить классу его значение и сферу применения этого закона. Убедиться, что у класса нет вопросов.
6	Закон Джоуля-Ленца	Перед началом можно показать классу видео с экспериментом, демонстрирующим закон Джоуля-Ленца. Рассказать классу, почему в законе присутствуют имена двух ученых. Рассказать, как формулируется закон. Для более качественного понимания провести сравнения сопротивления тока с сопротивлением трения и выделяющегося при этом тепла. Разобрать задачу с применением закона Джоуля-Ленца.
7	Заключение	Подвести итоги занятия, спросить, есть ли у класса вопросы. Задать контрольные вопросы по пройденному материалу. Предупредить класс, что на следующем занятии пройдет практическое занятие.

№	Этап урока	Содержание
8	Резервное время	Показать видео и рассказать классу интересные факты по пройденной теме, не вошедшие в программу.

Урок №3. Тема: «Теория пайки»

Цель урока: Обучить теории пайки, дать представление об инструментах и методике пайки.

№	Этап урока	Содержание
1	Введение	<p>Поприветствовать учеников. Сформулировать тему и цель урока. Постараться замотивировать учеников, акцентируя важность темы. Объяснить, что знания, которые они получают на этом уроке обязательно пригодятся им в дальнейшем на практических занятиях.</p>
2	Теория пайки	<p>Перечислить основные технологические операции, которые происходят во время пайки. Пайка сводится к следующим технологическим операциям: • Паяемые поверхности очищают от загрязнений, коррозионных корок и т.п. Зачищают до блеска, т.е. до отсутствия видимых следов окислов; • Покрывают флюсом – веществом, удаляющим остатки окисла и не допускающим окисления поверхностей в дальнейшем процессе. Для флюсовки под лужение предпочтительно использовать не жидкие или твердые флюсы, а флюс-пасты; • Затем поверхности лудят – наносят на них расплавленный припой (специально предназначенный для пайки сплав), он при этом растекается тонкой пленкой и химически соединяется с основным металлом; • Детали предварительно соединяют механически: скруткой, сжатием пинцетом, пассатижами, в тисках, струбциной и пр. Наносят еще флюс, чтобы не допустить окисления припоя под нагревом; • Наносят с прогревом еще припой (возможно, уже другой) до получения спая заданного качества; • Если пайка велась паяльником с луженым жалом, по ее окончании его очищают и покрывают неактивным флюсом. Чтобы пайки были качественными, обычный паяльник должен храниться с зафлюсованным жалом! Подробно описать процесс зачистки, лужения и пайки. Дать ученикам рассмотреть и подержать в руках паяльник, во время объяснения из чего он состоит. Рассказать об особенностях подготовки и пайки проводов. Научить зачистке и скрутке проводов. Каждый ученик должен провести эту операцию самостоятельно. Спросить, есть ли у класса вопросы.</p>

№	Этап урока	Содержание
3	Припои и флюсы	<p>Рассказать о припоях и флюсах. Припои от ПОС-90 до Авиа-2 – мягкие для низкотемпературной пайки. Гарантированно обеспечивают только электрический контакт. ПОС-30 и ПОС-40 паяют медь, латунь, бронзу с неактивными флюсами, а их же со сталью и сталь со сталью – с активными. ПОССр-15 можно паять оцинковку с неактивными флюсами; другие припои при этом разъедают цинк до стали и пайка скоро отваливается. 34А, МФ-1 и ПСр-25 припои твердые, для высокотемпературной пайки. Припоем 34А можно паять алюминий в пламени (см. далее, о пайке алюминия) со специальными флюсами, см. там же. Припоем МФ1 припаивают медь к стали с активированным флюсом. «Низкие требования к прочности» в данном случае значит, что прочность спая ближе к прочности меди, чем стали. ПСр-25 при пайке сухим паяльником пригоден для пайки ювелирных изделий, витражей тиффани и т.п. Паяльные флюсы делятся на нейтральные (неактивные, бескислотные), химически с основным металлом не взаимодействующие или взаимодействующие в ничтожной степени, активированные, химически действующие на основной металл при нагреве, и активные (кислотные), действующие на него и холодными. Заменители технического ацетона – растворители 646 и 647. Хлористый цинк в активированных флюс-пастах часто заменяют тетраборнокислым натрием – бурой. Соляная кислота – высокотоксичное химически агрессивное летучее вещество; хлорид цинка также токсичен, а при нагреве сублимирует, т.е. улетучивается не плавясь. Бура безопасна, но при нагреве выделяет большое количество кристаллизационной воды, что немного ухудшает качество пайки. Лучше, чтобы у в наличии были образцы припоев и флюсов, чтобы ученики смогли взять их в руки и рассмотреть.</p>
4	Виды пайки	<p>Рассказать об основных видах пайки: ● Мелкая пайка ● Пайка радиоэлектронных компонент на печатную плату ● Микросхемы. Для более наглядного понимания технического процесса показать ученикам пайку микросхем, или продемонстрировать обучающее видео.</p>
5	Заключение	<p>Подвести итоги занятия, спросить, есть ли у класса вопросы. Задать контрольные вопросы по пройденному материалу. Если останется время рассказать классу о подставках для паяльников и основную технику безопасности.</p>
6	Резервное время	<p>Показать видео и рассказать классу интересные факты по пройденной теме, не вошедшие в программу.</p>

Урок №4. Тема: «Аэродинамика полета. Пропеллер»

Цель урока: Сформировать знания основных принципов аэродинамики пропеллеров. Дать представление о главных характеристиках винта, и их влияние на полетные качества коптера. Научить ориентироваться в технических таблицах по подбору пропеллеров и моторов.

№	Этап урока	Содержание
1	Введение	Поприветствовать учеников. Провести тестирование по пройденным темам. Сформулировать тему и цель урока.
2	Аэродинамика пропеллера	Объяснить учащимся за счет чего образуется подъемная сила, которая позволяет коптеру совершать полет. Привести пример с воздушной подушкой. Раздать каждому ученику или на команду пропеллер и объяснить его строение. Показать видео с вращением пропеллера и его воздействием на воздушные потоки. Изучить схему вращения моторов дрона и позволить ученикам правильно разместить воздушные пропеллеры.
3	Параметры пропеллеров	Показать пример пропеллеров разного диаметра и с различным количеством лопастей. Более крупные пропеллеры требуют большей мощности от мотора на свою раскрутку. Нужно убедиться, что мотор может развивать нужную мощность. Также, большие и тяжелые пропеллеры обладают большей инерцией, поэтому они не смогут мгновенно ускориться, что отразится на маневренности коптера. Разобрать основные характеристики пропеллеров и предложить ученикам рассчитать, какие лучше использовать пропеллеры для 3 коптеров разных габаритов. Расчет и подбор воздушного винта к двигателю, а также к конкретному коптеру – сложная и тонкая задача. Исходными данными для подбора пропеллеров для самодельных конструкторов обычно являются мощность двигателя $N_{дв}$ (Вт), частота вращения воздушного винта $TВ$ (об/мин), максимальная скорость движения (полета) $V_{макс}$ (м/с).
4	Тестирование воздушных пропеллеров	Провести исследование влияния выбора пропеллеров. Провести летные испытания с использованием 3 типов пропеллеров: • 2 лопастной большого диаметра • 3 лопастной большого диаметра • 3 лопастной малого диаметра. Засечь время полета, скорость и подъемную силу (путем подъема дополнительного груза).
5	Заключение	Подвести итоги занятия, спросить, есть ли у класса вопросы. Спросить, что из изученного на занятии было для них интереснее всего. Попросить учеников ответить на контрольные вопросы. Предложить ученикам по желанию провести в интернете дополнительное исследование на пройденную тему. Сообщить ученикам, какую тему они будут проходить на следующем занятии.
6	Резервное время	Показать видео и рассказать классу интересные факты по пройденной теме, не вошедшие в программу.

Урок №5. Тема: «Основы электромагнетизма. Типы двигателей»

Цель урока: сформировать теоретические знания по основным законам электромагнетизма. Дать представление о строении и функционале популярных моделей электромоторов. Актуализировать использование бесколлекторных двигателей при создании коптеров.

№	Этап урока	Содержание
1	Введение	Поприветствовать учеников. Провести опрос по пройденным темам. Сформулировать тему и цель урока. Спросить, как, по их мнению, работают моторы и что заставляет их вращаться.
2	Основные законы электромагнетизма	Описать основные законы электромагнетизма и их формулировки: <ul style="list-style-type: none"> • Закон Ампера (закон взаимодействия электрических токов. Впервые был установлен Андре • Мари Ампером в 1820 для постоянного тока. Из закона Ампера следует, что параллельные проводники с электрическими токами, текущими в одном направлении, притягиваются, а в противоположных – отталкиваются) • Закон Ома (физический закон, определяющий связь электродвижущей силы источника (или электрического напряжения) с силой тока, протекающего в проводнике, и сопротивлением проводника. Установлен Георгом Омом в 1826 году и назван в его честь) • Закон Кулона (это закон, описывающий силы взаимодействия между неподвижными точечными электрическими зарядами). Привести примеры использования этих законов в повседневной жизни. Спросить, есть ли у класса вопросы.
3	Типы двигателей	Показать классу (на фото и видео, если нет в наличии) популярные виды электродвигателей: <ul style="list-style-type: none"> • Двигатель постоянного тока • Универсальный коллекторный двигатель • Асинхронный электродвигатель • Синхронный электродвигатель. Коллекторные развивают меньшую скорость. Бесколлекторные двигатели способны развить большую скорость, а также более износостойкие. Попросить класс привести примеры, где они встречали данные типы двигателей.
4	Сравнение двигателей	Дать наглядное сравнение коллекторных и бесколлекторных двигателей. Спросить учеников, какой двигатель будет правильно использовать на коптерах и почему. Спросить, есть ли у класса вопросы по пройденной теме.
5	Заключение	Подвести итоги занятия, спросить, есть ли у класса вопросы. Спросить, что из изученного на занятии было для них интереснее всего. Попросить учеников ответить на контрольные вопросы. Предложить ученикам по желанию провести в интернете дополнительное исследование на пройденную тему. Сообщить ученикам, какую тему они будут проходить на следующем занятии.
6	Резервное время	Показать видео и рассказать классу интересные факты по пройденной теме, не вошедшие в программу.

Урок №6. Тема: «Бесколлекторные двигатели и регуляторы их хода»

Цель урока: закрепить теоретические знания о строении и работе бесколлекторных электродвигателей. Сформировать знания о работе и настройке регуляторов хода, технике безопасности. Обучить решению проблем, возникающих в случае некорректной работы регуляторов.

№	Этап урока	Содержание
1	Введение	Поприветствовать учеников. Провести опрос по пройденным темам. Сформулировать тему и цель урока. Сделать упор на то, какие типы двигателей и почему применяются в коптерах.
2	Принцип работы бесколлекторного электродвигателя	Описать принцип работы бесколлекторного электродвигателя. Показать видео работы. Разобрать такие понятия как: ● Обмотка ● Ротор ● Статор ● Фаза (Трехфазные бесколлекторные двигатели приобрели наибольшее распространение. Но они могут быть и одно, двух, трех и более фазными. Чем больше фаз, тем более плавное вращение магнитного поля, но и сложнее система управления двигателем. 3-х фазная система наиболее оптимальна по соотношению эффективность/сложность, поэтому и получила столь широкое распространение). Попросить учеников объяснить, почему в бесколлекторных электродвигателях три фазных провода. При необходимости скорректировать ответы.

№	Этап урока	Содержание
3	Основные характеристики	<p>Спросить у класса, какими характеристиками, по их мнению, должен обладать бесколлекторный электродвигатель. Рассказать о датчиках и контроллерах (регуляторах оборотов или ESC). Рассказать о функциях и характеристиках ESC:</p> <ul style="list-style-type: none"> • Максимальный постоянный ток (указывает, какой ток контроллер способен держать продолжительное время. Как правило, этот параметр входит в обозначение контроллера (например Jes -18, Phoenix -10). Иногда указывают величину "кратковременного" тока, допустимого в течении нескольких секунд) • Максимальное рабочее напряжение (указывается, с каким количеством NiCd или литий-полимерных банок можно использовать контроллер. Для контроллеров с ВЕС-м, эта величина может быть разная, в зависимости от количества сервомашинки) • Максимальные обороты (программное ограничение максимальных оборотов. Всегда указывается для двухполюсного двигателя. Для многополюсных моторов это число надо разделить на количество пар полюсов) • Внутреннее сопротивление (полное сопротивление силовых ключей контроллера, без учета проводов. Чем мощнее контроллер, тем меньше его внутреннее сопротивление. Как правило, сопротивление проводов сравнимо с внутренним сопротивлением контроллера и вносит до 30% потерь) • Частота импульсов контроллера (как правило, составляет 7- 8 КГц. У "продвинутых" контроллеров частоту регулирования можно запрограммировать на другие значения- 16 и 32 КГц. Эти значения применяется в основном для высокооборотных 3-4-х витковых моторов с малой индуктивностью, при этом улучшается линейность регулирования частоты вращения). Спросить у учащихся, что осталось непонятным разобрать то, что оказалось сложным для восприятия еще раз.

№	Этап урока	Содержание
4	Применение	<p>Дать каждому ученику или команде мотор и регулятор оборотов. Указать на основные ошибки при установке моторов на луч (использование слишком длинных пропеллеров при установке, что приводит к повреждению обмотки). Разобрать каким образом мотор присоединяется к регулятору оборотов, а регулятор к полетному контроллеру и плате распределения питания. Рассказать про настройки, которыми обладает контроллер:</p> <ul style="list-style-type: none"> • Напряжение выключения мотора • Тип выключения мотора • Тормоз • Опережение • Режим старта • Время акселерации или задержка акселерации • Ограничение тока • Режим газа • Реверс <p>Объяснить основные проблемы при использовании регуляторов оборотов и какие действия помогают их разрешить.</p>
5	Заключение	<p>Подвести итоги занятия, спросить, есть ли у класса вопросы. Спросить, что из изученного на занятии было для них интереснее всего. Попросить учеников ответить на контрольные вопросы. Предложить ученикам по желанию провести в интернете дополнительное исследование на пройденную тему. Сообщить ученикам, какую тему они будут проходить на следующем занятии.</p>
6	Резервное время	<p>Показать видео и рассказать классу интересные факты по пройденной теме, не вошедшие в программу.</p>

Урок №7. Тема: «Принцип работы, типы и устройство аккумуляторов»

Цель урока: сформировать знания о принципе работы аккумуляторов. Сформировать и актуализировать знания о видах и специфике литий-полимерных аккумуляторах, техники безопасности.

№	Этап урока	Содержание
1	Введение	<p>Поприветствовать учеников. Сформулировать тему и цель урока. Сформировать учебную мотивацию на занятии, отметив, что на уроке будет рассказан принцип работы с элементом питания, который в дальнейшем будет использоваться при сборке коптеров.</p>
2	Устройство аккумуляторов	<p>Дать определение аккумулятора. Рассказать о сферах их применения. Продемонстрировать схему его устройства, процесс зарядки и разрядки аккумулятора. (Принцип работы аккумулятора: когда к электродам подключена нагрузка, например, лампочка, то создается замкнутая электрическая цепь, через которую протекает ток разряда. Он формируется движением электронов в металлических частях и анионов с катионами в электролите)</p>
3	Способы соединения аккумуляторов	<p>Рассказать о правилах и особенностях при параллельном и последовательном соединении аккумуляторов.</p>
4	Аккумуляторы для коптеров	<p>Рассказать о моделях аккумуляторов, которые применяются при сборке коптеров. Описать основные характеристики аккумуляторов: ● Ёмкость (Записывается в ампер-часах. Это такой ток который до полного разряда может выдавать аккумулятор в течении часа. Например, если емкость аккумулятора 3А/ч, то значит он может в течении одного часа выдавать ток 3А. При токе 1А его хватит на 3 часа, а при токе 30А он разрядится за 6 минут) ● Максимальный разрядный ток Указывается во сколько максимальный разрядный ток превышает емкость. Например значение «30-40С» для аккумулятора с емкостью 3А/ч означает, что он кратковременно может выдавать ток 90-120А. Естественно, при выборе аккумулятора необходимо руководствоваться меньшим значением. ● Напряжение (Зависит от количества «банок» или ячеек аккумулятора. Напряжение одной ячейки LiPo аккумулятора составляет порядка 3,7В. Соответственно, чем больше ячеек, тем больше напряжение. Соединяя аккумуляторы последовательно можно набрать достаточно большое напряжение, как это делают, например, в электровелосипедах). Раздать всем учащимся или на команду аккумуляторы. Попросить измерить напряжение на одном аккумуляторе, а также на аккумуляторах соединенных последовательно и параллельно. Попросить сделать выводы о величине напряжения при различных способах их соединения. Скорректировать их при необходимости.</p>

№	Этап урока	Содержание
5	Зарядка аккумулятора	Рассказать о правилах зарядки LiPo аккумуляторов, и необходимых для этого устройствах. Показать, как поставить аккумулятор на зарядку. Попросить каждого ученика проделать это.
6	Применение аккумуляторов	Рассказать о непосредственном применении аккумулятора в сборке коптера. Продемонстрировать классу установленный на коптере аккумулятор. Показать устройство его подключения к плате распределения энергии. (Для подключения аккумуляторов используют специальные коннекторы. Диаметр пистонов в них 4мм и они дополнительно подпружинены для обеспечения большой площади контакта. Еще для подключения используют специальные провода в силиконовой изоляции, которая способна выдерживать высокие температуры).
7	Техника безопасности	Рассказать о правилах эксплуатации и хранения аккумуляторов, и последствия, к которым может привести нарушение техники безопасности. Показать видео, демонстрирующие последствия механического повреждения аккумулятора .
8	Заключение	Провести тест по пройденным темам. Подвести итоги занятия. Спросить, есть ли у класса вопросы. Задать контрольные вопросы.
9	Резервное время	Если останется время, показать классу видео, более подробно демонстрирующее процесс зарядки аккумулятора.

Урок №8. Тема: «Управление полетом мультикоптера. Принцип функционирования полетного контроллера. ПИД регуляторы»

Цель урока: Закрепить знания о принципе управления коптером.

Сформировать знания о особенностях работы полетного контроллера и ПИД регулятора. Дать представление о принципах расчётов ПИД-регуляторов.

№	Этап урока	Содержание
1	Введение	Поприветствовать учеников. Сформулировать тему и цель урока. Повторяя пройденный материал, спросить у учеников по какому принципу вращения пропеллеров происходит полет коптера и изменение его движения, какую роль занимает полетный контроллер.
2	Принцип функционирования полетного контроллера	Напомнить ученикам еще раз назначение и функции полетного контроллера. Перечислить его основные задачи: <ul style="list-style-type: none"> • Собирает информацию с датчиков (встроенные, либо внешние: гироскопы, акселерометры, GPS, магнитометр); • Рассчитывает свое положение в пространстве, по показаниям датчиков; • Собирает информацию о внешних воздействиях, таких как отклонения стиков пилотом, алгоритм программы; • Вносит корректировку с помощью коэффициентов ПИД (Пропорционально-Интегрально-Дифференциальные); • Отправляет управляющие сигналы на регуляторы оборотов (ESC). Рассказать подробнее о связи полетного контроллера с ESC.
3	ПИД-регуляторы	Расскажите о функциях ПИД регулятора. (При работе с мультикоптерами, вам придется столкнуться с настройкой ПИД-регулятора, этот математический аппарат применяется почти во всех задачах стабилизации: стабилизация углов дрона в воздухе, полет и удержание позиции по GPS, удержание высоты по барометру). По порядку объясните математические расчёты по которым происходит контроль над полетом коптера. Привести примеры сайтов, на которых можно рассчитать ПИДы для конкретной системы. Научить этим пользоваться. Спросить у класса, есть ли вопросы.
4	Заключение	Подвести итоги. Задать контрольные вопросы. Наверняка у многих учеников появятся вопросы после этого урока. Показать или предложить ученикам самостоятельно посмотреть видео, объясняющее простым языком функционирование ПИД-регулятора.
5	Резервное время	Если останется время, показать видео полета коптера с плохо настроенными пидами. Предложить ребятам подумать, каким образом это можно исправить.

Урок №9. Тема: «Основы радиосвязи. Принцип работы радиоаппаратуры управления»

Цель урока: Дать представление об основах радиосвязи, работе передатчика и приёмника сигнала. Ввести в терминологию модуляции сигналов.

№	Этап урока	Содержание
1	Введение	Поприветствовать учеников. Сформулировать тему и цель урока. Провести тестирование по пройденным темам.
2	Основы радиосвязи	Дать определение понятию «Радиосвязь». (Радиосвязь - наиболее распространенный способ передачи информации на расстояние. Сотовые телефоны, спутниковая связь, телевидение - все это работает на основе передачи сигналов через электромагнитные колебания определенной частоты). Спросить класс, где в повседневной жизни они могут наблюдать применение радиосвязи. Рассказать по какому принципу происходит передача сигнала от передатчика к приемнику.
3	Принцип работы радиоаппаратуры управления	Рассказать о принципе работы передатчика и приемника (Аппаратура радиопередачи состоит из передатчика, который находится у пилота, и размещенных на модели приемника и полетного контроллера, который и управляет дроном через регуляторы мощности). Продемонстрировать схему взаимодействия передатчика и приемника в устройстве коптера. Спросить учеников, есть ли у них вопросы по этой схеме.
4	Передатчик	Рассказать о видах пультов управления. (Различают 2 основных вида пультов - джойстиковые и пистолетные. Для дронов используют джойстиковый пульт). Продемонстрируйте схему модуляции PPM сигнала. Показать настройки PPM (PPM сигнал имеет фиксированную длину периода $T=20\text{мс}$. Это означает, что информация о положениях ручек управления на передатчике попадает на модель 50 раз в секунду, что определяет быстродействие аппаратуры управления. Как правило, этого хватает, поскольку скорость реакции пилота на поведение модели намного меньше. Все каналы пронумерованы и передаются по порядку номеров. Значение сигнала в канале определяется величиной временного промежутка между первым и вторым импульсом - для первого канала, между вторым и третьим - для второго канала и т.д.) и PWM - шим импульс (Для дронов минимальное количество каналов - 4: управление газом, угол крена, угол тангажа, угол рысканья. Положение каждого из стиков пульта кодируется при помощи ШИМ импульса) на пульте радиопередачи).

№	Этап урока	Содержание
5	Приёмник	<p>Дать определение приемнику. (Устройство, служащее для осуществления радиоприема, т.е. для выделения сигналов из радиоизлучения. Приёмник устанавливается на квадрокоптере, принимает сигнал с пульта и передаёт его в полетный контроллер) Продемонстрировать схему работы приёмника, рассмотреть конкретный пример.</p>
6	Заключение	<p>Подвести итоги занятия, спросить, есть ли у класса вопросы. Спросить, что из изученного на занятии было для них интереснее всего. Попросить учеников ответить на контрольные вопросы. Предложить ученикам по желанию провести в интернете дополнительное исследование на пройденную тему. Сообщить ученикам, какую тему они будут проходить на следующем занятии.</p>
7	Резервное время	<p>Показать видео и рассказать классу интересные факты по пройденной теме, не вошедшие в программу.</p>

Урок №10. Тема: «Аналоговая и цифровая видеотрансляция. Применяемые камеры, радиопередатчики и приемники»

Цель урока: Дать представление о принципах работы видеосвязи. Раскрыть основы работы аналоговой и цифровой видеосъемки. Сформировать знания применение съёмки на беспилотных летательных аппаратах.

№	Этап урока	Содержание
1	Введение	Поприветствовать учеников. Провести опрос или тестирование по пройденным темам. Сформулировать тему и цель урока.
2	Видеокамеры аналогового типа	Рассказать об истории аналоговых камер и сфере их применения.
3	Работа аналоговых видеокамер	Продемонстрировать классу схему устройства аналоговой камеры, описать принцип ее работы. (Световой поток, проходя сквозь линзы объектива, попадает на матрицу ПЗС, где он преобразуется в видеосигнал). Попросить учеников зафиксировать основные элементы схемы и запомнить её.
4	Видеокамеры цифрового типа	Рассказать о появлении цифровых камер и области их использования.
5	Работа цифровых видеокамер	Продемонстрировать классу схему устройства цифровой видеокамеры, описать принцип ее работы (световой поток, отраженный от предметов попадает на чувствительную к свету матрицу устройства, которая преобразует его, но уже в сигнал электрический. Далее этот электрический сигнал, посредством процессора IP камеры обрабатывается, и лишь тогда кабелем «витая пара» или «коаксиальным кабелем», а возможно и средствами беспроводной связи (Wi – Fi), обработанный видеосигнал поступает на вход цифрового видеорегистратора). Попросить класс зафиксировать и запомнить её.
6	Дальность полёта	Рассказать классу о дальности полета коптера при использовании камеры. Перечислить особенности, на которые надо обратить внимание: <ul style="list-style-type: none"> • Дальность передачи видеосигнала зависит от количества помех в зоне полета • Разные системы передачи сигнала имеют различную способность огибать препятствия • Дальность ограничивается лишь емкостью батареи, но для реализации всего потенциала современных технологий необходима наземная станция. • Моделью довольно сложно управлять по камере. Так или иначе вы будете зависеть от погоды. Спросить у учеников, какие, по их мнению, еще трудности могут возникнуть.
7	Качество изображения	Рассказать о характеристиках современных камер. Продемонстрировать классу несколько видео с установленной на коптер камеры.

№	Этап урока	Содержание
8	Заключение	Подвести итоги занятия, спросить, есть ли у класса вопросы. Спросить, что из изученного на занятии было для них интереснее всего. Попросить учеников ответить на контрольные вопросы. Предложить ученикам по желанию провести в интернете дополнительное исследование на пройденную тему. Сообщить ученикам, какую тему они будут проходить на следующем занятии.
9	Резервное время	Показать видео и рассказать классу интересные факты по пройденной теме, не вошедшие в программу.

Урок №11. Тема: «Техника безопасности при сборке и настройке коптеров, при подготовке к вылету. Техника безопасности при работе с аккумуляторами»

Цель урока: Дать учащимся понимание основных правил техники безопасности при конструировании и эксплуатации коптеров и почему их следует соблюдать.

№	Этап урока	Содержание
1	Введение	Поприветствовать учеников. Провести опрос или тестирование по пройденным темам. Сформулировать тему и цель урока.
2	Безопасность при работе с Li-Po аккумуляторами	Рассказать правила техники безопасности при работе с Li-Po аккумуляторами: <ul style="list-style-type: none"> • Обращаться с аккумуляторами бережно. Не допускать падений, ударов деформаций. • При подключении (отключении) аккумуляторов держаться только за разъёмы, тянуть или дергать за провода запрещается. • В случае обрыва разъемов, обнаружения целостности изоляции или корпуса аккумулятора, не трогая его, немедленно сообщить преподавателю. Объяснить почему происходят возгорания аккумуляторов. Подкрепить рассказ показом видеороликов.
3	Техника безопасности при работе с паяльником	Вместе с учащимися вывести список правил, которые следует выполнять в лаборатории и во время пайки: <ul style="list-style-type: none"> • Следите за тем, чтобы нагретая часть паяльника не прикасалась в ходе пайки к электрическому проводу. Жало обладает очень высокой температурой, поэтому изоляция будет повреждена в считанные мгновения. • Далее последует короткое замыкание. • Перед началом работы проверьте целостность проводки и штепсельной вилки. Повреждения могут привести к тому, что ток замкнет непосредственно на вас. • При работе с горячим паяльником необходимо использовать подставку. Ее обычно изготавливают из деревянного бруска и металлических держателей. Так вы сможете безопасно расположить инструмент и не бояться, что он упадет на горючие материалы. • Как канифоль, так и сам припой при плавлении выделяют большое количество вредных веществ. Работать в респираторе вас никто не принуждает, но проветривать помещение после каждой пайки просто жизненно необходимо. Через каждые 30 минут делайте небольшие перерывы со сквозным проветриванием помещения и не забудьте при этом отключать паяльник. • Держите паяльник только за ручку. Проследить, чтобы учащиеся записали их в тетрадь и предупредить, что чтобы допустить до пайки, каждый должен будет отчитаться по этим правилам.

№	Этап урока	Содержание
4	Техника безопасности в аудитории	<p>Рассказать правила техники безопасности и поведения в аудитории:</p> <ul style="list-style-type: none"> • Не бегать рядом с рабочими столами • Вешать сумки на ручку кресла (а лучше вообще убрать в шкаф или оставить в другой части аудитории) • Не трогать оборудование (резак, 3д принтер) без преподавателя • Не вставлять пальцы и другие части тела и вещи в розетку • Еда, вода только на перемене (или по разрешению преподавателя) • Брать в руки демонстрационные дроны, пульты и так далее только с разрешения преподавателя. <p>Назначить ответственного за соблюдением ТБ в классе.</p>
5	Правила техники безопасности при летной эксплуатации коптеров	<p>Рассказать что включает в себя предполетная подготовка. Раздать листы предполетной подготовки каждому учащемуся или на команду и дать возможность учащимся полностью провести предполетную подготовку с занесением отметок о выполнении в лист предполетной подготовки. Правила ТБ при подготовке к полетам:</p> <ul style="list-style-type: none"> • Убедиться, что Li-Po аккумуляторы заряжены. • Убедиться, что аккумуляторы или батарейки в аппаратуре управления заряжены. • Устанавливать пропеллеры только перед вылетом. • Проверить надёжность следующих узлов: Затянутость гаек пропеллеров. Крепление и целостность защит пропеллеров. Надёжность крепления проводов Отсутствие болтающихся проводов. • Подключать Li-Po аккумулятор только перед вылетом. • Располагать зрителей за спиной пилота или за линией, проходящей через оба плеча пилота за спиной пилота. • Не допускать выхода зрителей в полусферу перед лицом пилота. • Знать и помнить время полёта, на которое рассчитан данный коптер и его аккумулятор. • ДО подключения Li-Po аккумулятора включить аппаратуру управления • (пульт), перевести стик газа в нулевое положение. • Подключать Li-Po аккумулятор только перед взлётом, отключать сразу после взлёта. • Стоять на расстоянии не менее 3 м от коптера. • Взлетать с земли с ровной площадки, на расстоянии не менее 3 метров от препятствий.

№	Этап урока	Содержание
6	Безопасность в полете	<p>Вместе с учащимися вывести правила техники безопасности в полете. ● Выполнять все указания преподавателя или лётного инструктора. ● Заранее обозначить зону пилотажа. Летать только в обозначенной зоне и не допускать вылета за её пределы. Не залетать за собственную спину. ● При обучении полётам летать на уровне ниже собственного роста. ● Летать рядом с собой на расстоянии, на котором вам видна ориентация коптера в пространстве. Не улетать далеко от себя. В случае сомнений в ориентации коптера немедленно выполнить посадку на месте. Не пытаться взлететь. Подойти ближе к коптеру и выполнить взлёт. ● При управлении все движения стиками выполнять аккуратно и плавно. Не допускать резких движений. При необходимости изменить направление полёта двигать стиками следует энергично, но не резко. ● Летать следует осторожно и выполнять только те элементы, в которых нет сомнений. Запрещается выполнять фигуры пилотажа, в успехе которых возникают сомнения и фигуры, связанные с риском. ● Соблюдать скоростной режим. Скорость полета коптера держать в пределах скорости идущего человека. ● Вернуть коптер к месту посадки к рассчитанному времени, не допускать полной разрядки аккумулятора в полёте. ● Посадку выполнять только на ровную открытую площадку вдали от препятствий. Проследить, чтобы учащиеся записали правила в тетрадь. Предупредить, что допуск к полетам будет даваться только после сдачи правил техники безопасности.</p>
7	Заключение	<p>Подвести итоги занятия, спросить, есть ли у класса вопросы. Спросить, что из изученного на занятии было для них интереснее всего. Попросить учеников ответить на контрольные вопросы. Предложить ученикам по желанию провести в интернете дополнительное исследование на пройденную тему. Сообщить ученикам, какую тему они будут проходить на следующем занятии.</p>
8	Резервное время	<p>Показать видео и рассказать классу интересные факты по пройденной теме, не вошедшие в программу.</p>

Урок №12. Тема: «Теория ручного визуального пилотирования»

Цель урока: Сформировать представление у учащегося о принципах визуального пилотирования. Разобрать принципы управления коптером с помощью пульта радиуправления.

№	Этап урока	Содержание
1	Введение	Поприветствовать учеников. Провести опрос или тестирование по пройденным темам. Сформулировать тему и цель урока.
2	Предполетная подготовка	Дать задание каждому учащемуся или команде провести предполетную подготовку и поставить отметки в листе предполетной подготовки.
3	Базовые процедуры	Проверить, чтобы на всех коптерах были сняты пропеллеры. Рассказать о том, как выполнить биндинг пульта, арм, дизарм, kill switch. Прodelать это несколько раз вместе с учащимися. Спросить, есть ли у учащихся вопросы. Под запись: ● Arm (англ. - "вооружить") – разблокировать моторы коптера, перевести коптера в "боевое" состояние, после чего коптер начинает реагировать на движения стика газа. ● Disarm (англ. - "разоружить") – заблокировать моторы коптера, после чего коптер перестает реагировать на движения стика газа. ● Процедура включения – последовательность действий после установки коптера на взлетную площадку перед взлетом. ● Визуальное пилотирование - тип пилотирования, при котором коптер находится в зоне прямой видимости ● FPV (англ. - "вид от первого лица") - полет по камере, вид от первого лица - тип пилотирования, при котором управление коптером осуществляется по дополнительному видео-радиоканалу с передачей изображения с камеры, установленной на борту коптера.
4	Пульт управления	Разобрать основные принципы управления коптером с пульта (газ, рыскание, крен, тангаж). Рассказать о полетных режимах и о том, как их менять на пульте. Пройтись по меню настроек пульта.
5	Подготовка к пилотированию	Полеты в симуляторе: каждый учащийся летает в симуляторе, привыкая к управлению коптером. Полет производится в режиме Stabilize в режиме визуального пилотирования (не FPV).
6	Заключение	Подвести итоги занятия, спросить, есть ли у класса вопросы. Спросить, что из изученного на занятии было для них интереснее всего. Попросить учеников ответить на контрольные вопросы. Предложить ученикам по желанию провести в интернете дополнительное исследование на пройденную тему. Сообщить ученикам, какую тему они будут проходить на следующем занятии.
7	Резервное время	Показать видео, где БПЛА выполняют различные трюки.

Урок №13. Тема: «Техника безопасности при летной эксплуатации коптера»

Цель урока: Сформировать у учащегося понимание опасности коптеров и заложить основы правильного поведения при летной эксплуатации коптера.

№	Этап урока	Содержание
1	Введение	Поприветствовать учеников. Провести опрос или тестирование по пройденным темам. Сформулировать тему и цель урока.
2	Предполетная подготовка	Дать задание каждому учащемуся или команде провести предполетную подготовку и поставить отметки в листе предполетной подготовки. Чеклист включает в себя следующие пункты: 1. Провода аккумулятора уложены так, что, будучи подключенными, не помещают полётам. 2. Вращению пропеллеров ничего не мешает. 3. Защиты пропеллеров целы и закреплены. 4. Проверить дальность видеопередатчика и обозначить как полетную зону 5. Все присутствующие люди находятся за спиной. На расстоянии 10 метров спереди и сбоку нет людей.
3	Полетная зона	Вместе с учащимися проводится осмотр и подготовка полетной зоны. Провести зачет по технике безопасности при полетах и предполетной подготовке. Показать примеры полетов с объяснениями по способам пилотирования. Показать как действовать в аварийных ситуациях и при поломках. Смоделировать такие ситуации и проверить, как дети поведут себя в таких ситуациях. Скорректировать их действия при необходимости.
4	Заключение	Подвести итоги занятия, спросить, есть ли у класса вопросы. Спросить, что из изученного на занятии было для них интереснее всего. Попросить учеников ответить на контрольные вопросы. Предложить ученикам по желанию провести в интернете дополнительное исследование на пройденную тему. Сообщить ученикам, какую тему они будут проходить на следующем занятии.
5	Резервное время	Показать видео и рассказать классу интересные факты по пройденной теме, не вошедшие в программу.

Урок №14. Тема: «Обучение лётному мастерству»

Цель урока: Развить навыки визуального пилотирования.

№	Этап урока	Содержание
1	Введение	Поприветствовать учеников. Сформулировать тему и цель урока. Провести тестирование по пройденным темам.
2	Взлет / посадка.	Провести летные испытания коптера. Каждый учащийся выполняет не менее 5 взлетов - посадок, стараясь совершить посадку как можно мягче.
3	Висение	Учащийся должен научиться зависать над точкой не менее чем на 10 секунд хвостом к себе. (Упражнение 1. Висение хвостом к себе. Выполняется на уровне колен над центральным перекрестием зоны полётов).
4	Полеты вперед / назад	Выполнение упражнения "полеты вперед - назад по прямой хвостом к себе" не менее 5 раз
5	Полет по кругу	Выполнение упражнения "полет по кругу хвостом к себе" не менее 5 раз
6	Висение боком	Учащийся должен научиться зависать над точкой не менее чем на 10 секунд боком к себе.
7	Заключение	Подвести итоги занятия, спросить, есть ли у класса вопросы. Спросить, что из изученного на занятии было для них интереснее всего. Попросить учеников ответить на контрольные вопросы. Предложить ученикам по желанию провести в интернете дополнительное исследование на пройденную тему. Сообщить ученикам, какую тему они будут проходить на следующем занятии.
8	Резервное время	Показать видео, где БПЛА выполняют различные трюки. Разобраться, каким образом их можно выполнить.

Урок №15. Тема: «Основы радиоэлектроники, схемотехники и макетирования электрических схем»

Цель урока: Сформировать у учащегося представление о том, как работают различные платы, схемы и из каких элементов они состоят. Научить составлять принципиальные электрические схемы.

№	Этап урока	Содержание
1	Введение	Поприветствовать учеников. Провести опрос по пройденным темам. Сформулировать тему и цель урока. Сделать упор на то, что знание основ радиозлектроники поможет учащимся усовершенствовать свой летательный аппарат, путем подключения датчиков и светодиодов / светодиодных лент.
2	Электричество	Дать под запись основные определения, касающиеся разделов радиозлектроника и схемотехника. <ul style="list-style-type: none"> • Электричество (природное явление, подтверждающее существование, взаимодействие и движение электрических зарядов. Электричество впервые было обнаружено еще в VII веке до н.э. греческим философом Фалесом) • Электрон (элементарная частица, имеет отрицательный заряд примерно равный $-1,602 \cdot 10^{-19}$ Кл (Кулон). Обозначается "e") • Напряжение • Электрический ток (это физический процесс направленного движения заряженных частиц под действием электромагнитного поля от одного полюса замкнутой электрической цепи к другому. В качестве частиц, переносящих заряд, могут выступать электроны, протоны, ионы и дырки) • Проводник • Электрическое сопротивление (физическая величина, определяющая свойство проводника препятствовать (сопротивляться) прохождению тока. Единица измерения сопротивления – Ом) • Сила тока • Электронная схема (это сочетание отдельных электронных компонентов, таких как резисторы, конденсаторы, индуктивности, диоды, транзисторы и интегральные микросхемы, соединённых между собой) • Типы плат Каждое понятие подкреплять коротким видеороликом. Спросить у учащихся, что осталось непонятным.
3	Закон Ома	Прорешать несколько простых задач на закон Ома (от 3 до 5). Раздать каждому учащемуся подобную задачу на самостоятельное решение. Объяснить что такое последовательное и параллельное соединение.
4	Электронные компоненты	Раздать учащимся электронные компоненты и дать перерисовать обозначения этих компонентов. Учащиеся самостоятельно или в командах собирают схемы: <ul style="list-style-type: none"> • Последовательное и параллельное подключение светодиодов на макетной плате • Светодиод с кнопкой на макетной плате • Светодиод с переменным резистором на макетной плате. К каждому опыту зарисовать схему подключения.

№	Этап урока	Содержание
5	Заключение	Подвести итоги занятия, спросить, есть ли у класса вопросы. Спросить, что из изученного на занятии было для них интереснее всего. Попросить учеников ответить на контрольные вопросы. Предложить учащимся решить простую задачу на закон Ома. Предложить ученикам по желанию провести в интернете дополнительное исследование на пройденную тему. Сообщить ученикам, какую тему они будут проходить на следующем занятии.
6	Резервное время	Показать примеры интересных схем. Предложить учащимся собрать одну из понравившихся схем дома.

Урок №16. Тема: «Основы работы с аналоговым и цифровым сигналом»

Цель урока: сформировать у учащихся понимание типов хранения передачи видео, их достоинств и недостатков друг перед другом.

№	Этап урока	Содержание
1	Введение	Поприветствовать учеников. Провести опрос по пройденным темам. Сформулировать тему и цель урока.
2	Сигналы	Спросить у учащихся как они понимают определение “сигнал”. Скорректировать их ответы и дать под запись правильное определение. Зарисовать графики цифрового и аналогового сигнала.
3	Аналоговые сигналы	Объяснить что такое аналоговый сигнал, как выглядит и каким образом передается. Дать определение аналогового сигнала под запись. (Носитель информации, используемый для передачи сообщений в системе связи. Сигнал может генерироваться, но его приём не обязателен, в отличие от сообщения, которое рассчитано на принятие принимающей стороной, иначе оно не является сообщением. Сигналом может быть любой физический процесс, параметры которого изменяются (или находятся) в соответствии с передаваемым сообщением). Выделить основные достоинства и недостатки данного вида сигнала. Дать определение АЦП. (Аналого-цифровой преобразователь (АЦП, англ. Analog-to-digital converter, ADC) – устройство, преобразующее входной аналоговый сигнал в цифровой сигнал. Аналоговый сигнал является непрерывной функцией времени, в АЦП он преобразуется в последовательность цифровых значений)
4	Цифровые сигналы	Объяснить что такое цифровой сигнал, как выглядит и каким образом передается. Дать определение цифрового сигнала под запись. (Сигнал, который можно представить в виде последовательности цифровых значений. В наше время наиболее распространены двоичные цифровые сигналы (битовый поток) в связи с простотой кодирования и использованием в двоичной электронике. Для передачи цифрового сигнала по аналоговым каналам (например, электрическим или радиоканалам) используются различные виды манипуляции (модуляции)). Выделить основные достоинства и недостатки данного вида сигнала.
5	Заключение	Подвести итоги занятия, спросить, есть ли у класса вопросы. Спросить, что из изученного на занятии было для них интереснее всего. Попросить учеников ответить на контрольные вопросы. Сообщить ученикам, какую тему они будут проходить на следующем занятии.
6	Резервное время	Разобрать схемы передачи цифрового и аналогового видео.

Урок №17. Тема: «Основы работы с лабораторным оборудованием»

Цель урока: выработать понимание и навыки работы с лабораторным оборудованием.

№	Этап урока	Содержание
1	Введение	Поприветствовать учеников. Провести опрос по пройденным темам. Сформулировать тему и цель урока.
2	Мультиметр	Под запись дать определение мультиметра. (Универсальный комбинированный измерительный прибор, который сочетает в себе функции нескольких измерительных приборов, то есть может измерять целый диапазон электрических величин). Рассказать об основных режимах работы и правилах установления верхней границы измерений. Попросить учащихся измерить различные характеристики трех типов аккумуляторов или батареек, если есть возможность. Объяснить как обнаружить короткое замыкание или разрыв в цепи при помощи мультиметра.
3	Осциллограф	Дать определение осциллографа и способов его применение. (Прибор, показывающий форму напряжения во времени. Также он позволяет измерять ряд параметров сигнала, такие как напряжение, ток, частота, угол сдвига фаз. Но главная польза от осциллографа – возможность наблюдения формы сигнала. Во многих случаях именно форма сигнала позволяет определить, что именно происходит в цепи). Показать работу прибора на видео или на практике, если есть такая возможность.
4	Заключение	Подвести итоги занятия, спросить, есть ли у класса вопросы. Спросить, что из изученного на занятии было для них интереснее всего. Попросить учеников ответить на контрольные вопросы. Сообщить ученикам, какую тему они будут проходить на следующем занятии.
5	Резервное время	Предложить учащимся измерить сопротивление предметов, находящихся под рукой и сделать выводы о сопротивлении различных материалов.

Урок №18. Тема: «Теория FPV полетов»

Цель урока: Развить навыки FPV пилотирования.

№	Этап урока	Содержание
1	Введение	Поприветствовать учеников. Провести опрос или тестирование по пройденным темам. Сформулировать тему и цель урока.
2	Подключение FPV	Провести подключение и настройку оборудования для FPV полета. Провести предполетную подготовку коптера. Повторить правила техники безопасности при полетах. Уточнить особенности управления при FPV полетах.
3	Полеты в симуляторе	Каждый учащийся не менее 20 мин летает в режиме FPV в симуляторе, после чего допускается до полета на настоящем коптере.
4	Полеты вперед / назад	Выполнение упражнения "полеты вперед - назад по прямой хвостом к себе" не менее 5 раз
5	Полет по кругу	Выполнение упражнения "полет по кругу хвостом к себе" не менее 5 раз
6	Заключение	Подвести итоги занятия, спросить, есть ли у класса вопросы. Спросить, что из изученного на занятии было для них интереснее всего. Попросить учеников ответить на контрольные вопросы. Предложить ученикам по желанию провести в интернете дополнительное исследование на пройденную тему. Сообщить ученикам, какую тему они будут проходить на следующем занятии.
7	Резервное время	Показать видео, где БПЛА выполняют различные трюки в FPV режиме или транслирует красивые пейзажи с высоты.

Урок №19. Тема: «История автономных полетов. Развитие автопилотов в авиации»

Цель урока: сформировать понимание у учащихся что такое автономные системы, какими они могут быть и что требуется для их создания.

№	Этап урока	Содержание
1	Введение	Поприветствовать учеников. Провести опрос или тестирование по пройденным темам. Сформулировать тему и цель урока.
2	Первые автономные системы	Рассказать учащимся что такое автономные системы и автопилоты. (Автопилот – это устройство или программно-аппаратный комплекс, который может вести вверенное ему транспортное средство по заданной траектории). Привести примеры первых автономных систем: ● Тележка Леонардо да Винчи ● Торпеды Александровского и Уайтхеда ● Ракета «Фау-2» Стэнфордская тележка. Объяснить, с помощью чего автопилот может ориентироваться в пространстве. Попросить учащихся привести примеры знакомых им современных автономных устройств и систем.
3	Деловая игра	Попросить учащихся продумать концепт автономного БПЛА, который будет включать в себя описание: ● Применения этого устройства ● Способа и устройства навигации ● Используемого оборудования ● Технических характеристик ● Социальной значимости ● Себестоимости. Каждый учащийся или команда защищает свой проект перед классом, остальные учащиеся задают вопросы и пытаются скорректировать идею. По завершении анонимным голосованием выбирается лучший проект.
4	Заключение	Подвести итоги занятия, спросить, есть ли у класса вопросы. Спросить, что из изученного на занятии было для них интереснее всего. Попросить учеников ответить на контрольные вопросы. Предложить ученикам по желанию провести в интернете дополнительное исследование на пройденную тему. Сообщить ученикам, какую тему они будут проходить на следующем занятии.
5	Резервное время	Рассказать про интересные способы применения автономных БПЛА. Показать видео.

Урок №20. Тема: «Основы программирование на языке Python»

Цель урока: разобрать основы синтаксиса языка Python. Выработать навыки написания простейших программ.

№	Этап урока	Содержание
1	Введение	Поприветствовать учеников. Провести опрос или тестирование по пройденным темам. Сформулировать тему и цель урока. Сделать упор на то, что программирование является неотъемлемой частью работы любого современного инженера и создание автономных коптеров является популярной задачей настоящих разработчиков крупных компаний.
2	Введение в Python	Рассказать про типы языков, а именно расшифровать понятия: <ul style="list-style-type: none"> • Объектно-ориентированный язык • Язык программирования высокого уровня • Язык программирования низкого уровня • Компилируемый язык • Интерпретируемый язык. Рассказать, к каким типам относится язык Python. Привести примеры программ и систем, которые можно написать на языке Python. Объяснить зачем нужны библиотеки. (В составе Python поставляется большое число собранных и переносимых функциональных возможностей, известных как стандартная библиотека. Эта библиотека предоставляет Вам массу возможностей, востребованных в прикладных программах, начиная от поиска текста по шаблону и заканчивая сетевыми функциями. Python допускает расширение как за счет ваших собственных библиотек, так и за счёт библиотек, созданных другими разработчиками). Привести примеры библиотек в языке Python (например, math, позволяющая производить сложные математические операции). Спросить у учащихся, какие появились вопросы, их должно быть достаточно. Объяснить что такое блок-схемы. Научить ребят их строить.
3	Условная инструкция if-elif-else	Рассказать про важность отступов в питоне. Дать под запись синтаксис конструкции. Привести пример небольшой программы с использованием этой конструкции. Запустить программу на исполнение. Нарисовать блок-схему программы.
4	Цикл for	Дать под запись синтаксис конструкции. Привести пример небольшой программы с использованием этой конструкции. Запустить программу на исполнение. Нарисовать блок-схему программы.

№	Этап урока	Содержание
5	Цикл while	Дать под запись синтаксис конструкции. Привести пример небольшой программы с использованием этой конструкции. Запустить программу на исполнение. Нарисовать блок-схему программы.
6	Операторы break и continue	Дать под запись синтаксис конструкции. Привести пример небольшой программы с использованием этой конструкции. Запустить программу на исполнение. Нарисовать блок-схему программы.
7	Программирование	Дать несколько простых задач, требующих использования минимум двух конструкций языка. Попросить учащихся написать программу для их решения. Проверить и скорректировать программу вместе с обучающимися.
8	Заключение	Подвести итоги занятия, спросить, есть ли у класса вопросы. Спросить, что из изученного на занятии было для них интереснее всего. Попросить учеников ответить на контрольные вопросы. Предложить ученикам по желанию провести в интернете дополнительное исследование на пройденную тему. Сообщить ученикам, какую тему они будут проходить на следующем занятии.
9	Резервное время	Привести примеры популярных программ и систем, написанных на языке Python.

Урок №21. Тема: «Знакомство с компьютером Raspberry Pi»

Цель урока: дать понимание о способах применения микрокомпьютеров на примере Raspberry Pi 3. Обучить взаимодействию с unix-подобными системами.

№	Этап урока	Содержание
1	Введение	Поприветствовать учеников. Провести опрос или тестирование по пройденным темам. Сформулировать тему и цель урока. Рассказать о типах компьютеров, сделав упор на применение микрокомпьютеров.
2	Знакомство с Raspberry Pi 3	Дать под запись основные характеристики микрокомпьютера Raspberry Pi 3: – Вес – 45 грамм, – тактовая частота 700 МГц; – есть графическое ядро в процессоре Broadcom BCM2835; – оперативная память – 512 Мб; – есть USB -разъемы (один или два в зависимости от модели). Рассказать о сферах применения микрокомпьютеров, показать видео готовых проектов, использующих Raspberry Pi 3. Попросить учащихся придумать системы, в которых было бы актуальным применение Raspberry Pi 3.
3	Первое подключение Raspberry Pi 3	Установить микрокомпьютер и камеру на коптер. Провести подключение к полетному контроллеру с помощью USB или по UART. Обучить ребят записывать образ ОС на microSD. Подключиться к Raspberry по Wi-Fi. Рассказать об SSH и существующих SSH клиентах.
4	Использование ОС Raspbian	Объяснить что такое командная строка и как ей пользоваться. Дать под запись основные команды и примеры их использования: • ls • cd • nano • mv • mkdir • rm. Дать задания, требующие использования этих команд (просмотр файлов в каталоге, переименования файла, переноса файла, создание нового файла и т.п.)
5	Меняем SSID	Рассказать, что такое SSID. Научить изменять имя Wi-Fi сети. Объяснить что такое демоны и в какой момент они запускаются. Проработать с конфигурацией одного из них.
6	Используем права суперпользователя	Рассказать о типах и правах пользователей системы. Показать примеры использования sudo.
7	Подготовка копитера к автономным полетам	Проверить подключенное оборудование для автономных полетов. Убедиться в работоспособности подключения можно выполнив на Raspberry Pi: <code>rostopic echo /mavros/state</code>
8	Использование QGroundControl через Wi-Fi	Настроить беспроводное соединение для работы в QGroundControl. Предложить учащимся установить новую прошивку, которая подходит для автономных полетов и откалибровать копитер при беспроводном подключении.

№	Этап урока	Содержание
9	Заключение	Подвести итоги занятия, спросить, есть ли у класса вопросы, их должно быть много, нужно заранее продумать ответы на них. Попросить учеников ответить на контрольные вопросы. Предложить ученикам по желанию провести в интернете дополнительное исследование на пройденную тему. Сообщить ученикам, какую тему они будут проходить на следующем занятии.

Урок №22. Тема: «Управление автономным дроном: теория»

Цель урока: дать понимание способов программирования автономного коптера. Сформировать навыки написания кода на языке Python и закрепить навыки по использованию unix-подобных систем.

№	Этап урока	Содержание
1	Введение	Поприветствовать учеников. Провести опрос или тестирование по пройденным темам. Сформулировать тему и цель урока. Сделать упор на то, что на занятии будет решаться задача написания программы для настоящего автономного полета.
2	Системы координат	Рассказать про локальную и глобальную систему координат. Объяснить почему ориентации по локальной системе координат недостаточно. Рассказать о дополнительных системах координат, которые появляются благодаря получению дополнительной информации с камеры и внешних датчиков.
3	Включение и использование камеры	Необходимо попросить учащихся убедиться, что в launch-файле (~/.catkin_ws/src/drone/drone/drone.launch) включен запуск aguco_pose и нижней камеры для компьютерного зрения: При изменении launch-файла необходимо перезапустить пакет drone: <code>sudo systemctl restart drone</code> Проверить видео, полученное с камеры перейдя по адресу http://192.168.11.1:8080/ .
4	Распознавание меток	Рассказать каким образом распознаются метки и как они образуют систему координат. Спросить у учащихся есть ли вопросы на текущий момент.
5	Программирование и автономный полет	Предложить учащимся написать программу для взлета над меткой и посадки на нее. Рассказать о структуре программы, которая может выполнить эту задачу и привести пример кода с использованием функций: <code>• navigate</code> , <code>• set_position</code> <code>• set_velocity</code> . Проверить и скорректировать программы, написанные учащимися. Рассказать, каким образом осуществляется перехват копитера в ручное управление. Протестировать написанные программы.
6	Заключение	Подвести итоги занятия, спросить, есть ли у класса вопросы, их должно быть много, нужно заранее продумать ответы на них. Попросить учеников ответить на контрольные вопросы. Предложить ученикам по желанию провести в интернете дополнительное исследование на пройденную тему. Сообщить ученикам, какую тему они будут проходить на следующем занятии.
7	Резервное время	Попросить учащихся написать программы для полета по интересным траекториям и протестировать их.

Контрольные и проверочные материалы

Проверочные задания включают в себя:

- Контрольные вопросы к каждой теме
- Тесты текущего контроля знаний по темам
- Тест итогового контроля знаний

Контрольные вопросы по темам

Знакомство. Принципы проектирования и строение мультикоптеров

1. В какое время появился первый дрон, и в чём был его недостаток?
2. Чем отличаются БПЛА самолетного типа от обычных самолетов?
3. В каких сферах можно использовать БПЛА самолетного типа?
4. В каких сферах можно использовать коптеры?
5. Какие конфигурации дронов бывают?
6. Перечислите название осей коптера.
7. По какому принципу вращаются пропеллеры коптера?
8. За что отвечает полётный контроллер?
9. Для чего нужен ESC?
10. Какой вид электродвигателей применяется в коптерах? В чём их преимущество?
11. Какими тремя параметрами обладают воздушные пропеллеры?
12. Может ли дрон летать в вакууме?

Основы электричества

1. Что такое электродвижущая сила?
2. Как найти сопротивление в проводнике используя закон Ома.
3. Чем отличается проводник от диэлектрика?
4. Где применяется первый закон Кирхгофа?
5. Из-за чего в проводнике происходит выделение тепла при прохождении тока?

Теория пайки

1. Какое вещество не допускает окисление?
2. Перечислите основные этапы пайки.
3. Что такое лужение?
4. В каких случаях пайку использовать нельзя?
5. Какой флюс лучше использовать при пайке микросхем.

Аэродинамика полета. Пропеллер

1. За счёт чего образуется сила тяги в пропеллере?
2. Как узнать шаг пропеллера по названию его марки?
3. Что такое пропеллерная константа?
4. Для чего в конструкции коптера одновременно используются пропеллеры, вращающиеся по и против часовой стрелки?
5. Что является исходными данными для подбора винта в коптере?
6. Какие характеристики пропеллера нужны для быстроходного и тихоходного коптера?

Основы электромагнетизма. Типы двигателей

1. Как, следуя закону Ампера, ведут себя проводники с электрическими токами?
2. По закону Кулона как взаимодействуют относительно друг друга два точечных заряда в вакууме.
3. В чём основное различие коллекторных и бесколлекторных электродвигателей?
4. По каким характеристикам бесколлекторные электродвигатели подходят для использования их на дронах?

Бесколлекторные двигатели и регуляторы их хода

1. Зачем нужны датчики в бесколлекторных электродвигателях?
2. На что влияет количество фаз в бесколлекторном электродвигателе?
3. Перечислите основные характеристики контроллеров.
4. Какие ошибки при подключении контроллеров возможно допустить?
5. К каким последствиям могут привести эти ошибки?
6. Перечислите возможные настройки контроллера.

Принцип работы, типы и устройство аккумуляторов

1. Какие устройства называют аккумуляторами?
2. За счёт каких процессов в аккумуляторе накапливается энергия?
3. Что происходит в аккумуляторе во время его заряде и разряде?
4. Опишите два способа соединения аккумуляторов.
5. Какие аккумуляторы применяются при сборке коптеров?
6. Перечислите основные характеристики аккумуляторов.

Управление полётом мультикоптера. Принцип функционирования полетного контроллера. ПИД регуляторы

1. По какому принципу работает полётный контроллер?
2. Перечислите основные задачи полётного контроллера.
3. Сформулируйте принцип работы ПИД-регулятора.

Основы радиосвязи. Принцип работы радиоаппаратуры управления

1. Как происходит передача радиосигнала от передатчика к приёмнику?
2. Чем отличается АМ и FM модуляция передачи сигнала?
3. Почему передатчики радиоуправления делают многоканальными?
4. Какая модуляция используется в пультах управления коптерами?
5. По какому принципу работает приёмник радиосигнала?

Аналоговая и цифровая видеотрансляция. Применяемые камеры, радиопередатчики и приёмники

1. Опишите принцип работы аналоговой камеры
2. Опишите принцип работы цифровой камеры.
3. В чём основное отличие аналоговой и цифровой камеры?
4. На какой дистанции можно производить видеосъёмку.
5. Что позволит увеличить дистанцию приёма видеосигнала.
6. Что ещё может повлиять на дальность полёта?

Техника безопасности

1. Назовите меры предосторожности при использовании LiPo аккумуляторов.
2. Чего не следует делать при работе с паяльником?
3. Какие действия нужно выполнить перед взлётом?
4. Что запрещено делать во время полета?

Теория ручного визуального пилотирования

1. Что такое “Arm” и как его выполнить?
2. Что такое “Disarm” и как его выполнить?
3. Что включает в себя предполетная подготовка?

Техника безопасности при летной эксплуатации коптера

1. Какие пункты включает в себя “чеклист”?
2. Назовите правила техники безопасности при полете.
3. Что делать в случае падения и повреждения коптера в полете?

Обучение лётному мастерству

1. Как заармить коптер?
2. Как будет двигаться коптер, если левый стик поднять вверх на 50%, а правый Переместить назад?
3. Какие действия нужно выполнить стиками, чтобы развернуть коптер на 180 градусов?
4. Какие стили полета бывают?
5. Как задизармить коптер?

Основы радиоэлектроники, схемотехники и макетирования электрических схем

1. Каким образом было обнаружено существование электричества и как обосновать это с физической точки зрения?
2. Что такое сопротивление и в чем оно измеряется?
3. Как звучит закон Ома для участка цепи?
4. Объясните, в чем отличие аналоговых схем от цифровых?
5. Назовите самые часто встречающиеся компоненты в радиоэлектронных схемах.
6. В чем разница между микрокомпьютером и микроконтроллером?
7. Зачем нужны макетные платы?

Основы работы с аналоговым и цифровым сигналом

1. Какие типы сигналов бывают и чем они отличаются?
2. Объясните, зачем нужны АЦП?

Основы работы с лабораторным оборудованием

1. Что можно измерить при помощи мультиметра?
2. Можно ли при помощи обычного мультиметра измерить напряжение в розетке?
3. Что такое фаза?

Теория FPV полетов

1. Какое оборудование используется при FPV полетах?
2. Какими стиками чаще всего происходит управление при FPV полетах?
3. Какие действия стоит проделать стиками, чтобы полететь вправо?

История автономных полетов. Развитие автопилотов в авиации

1. Приведите пример первых автономных систем и принципы их работы.
2. Как работает автопилот?
3. Какие приборы задействовали изобретатели при создании первых автономно управляемых торпед?
4. Какими углами определяется положение летательного аппарата в пространстве?
5. Почему нельзя было использовать радиосигналы для управления стенфордской тележкой?
6. Как ориентировался первый полностью автономный наземный автопилот?

Основы программирование на языке Python

1. К какому типу языков программирования относится Python?

2. Зачем нужны библиотеки?
3. Объясните, что означает термин “переносимость” программ.
4. Приведите пример кода с использованием оператора ветвления.
5. Какие формы записи могут принимать логические “ложь” и “истина”?
6. Приведите пример кода с использованием цикла for.
7. Приведите пример кода с использованием цикла while.
8. В каком случае используются операторы break и continue?

Знакомство с компьютером Raspberry Pi

1. Что такое микрокомпьютер? Приведите примеры известных вам микрокомпьютеров.
2. Какие устройства можно подключить к Raspberry Pi 3?
3. Каким образом можно соединить полетный контроллер и Raspberry?
4. Какое напряжение требуется для корректной работы Raspberry Pi 3?
5. Откуда происходит загрузка операционной системы при включении Raspberry Pi 3?
6. Что такое SSH клиент и для чего используется?
7. Перечислите основные команды при работе с командной строкой ОС.
8. В какой момент запускаются демоны?
9. Как система обрабатывает комментарии в коде, оставленные программистом?
10. Каким образом можно получить права суперпользователя?

Управление автономным дроном: теория

1. Почему нельзя летать в помещении, используя GPS координаты?
2. Можно ли автономно летать используя только локальные координаты коптера?
3. Какие устройства нужно установить на коптер для ориентации по специальным меткам?
4. Что включает в себя предполетная подготовка к автономному полету?

Тесты текущего контроля знаний по темам

Знакомство. Принципы проектирования и строение мультикоптеров

1. Кто создал первое беспилотное судно?

1. Альберт Эйнштейн
2. Никола Тесла
3. Исаак Ньютон
4. Чарльз Кеттеринг

2. Как называется коптер с 6 моторами?

1. Пентакоптер

2. Октокоптер
3. Трикоптер
4. Гексакоптер

3. Что такое “тангаж”?

1. Наклон коптера вперед-назад
2. Наклон коптера вправо-влево
3. Вращение коптера вокруг своей оси
4. Набор скорости

4. Где расположены датчики, отвечающие за определение положения коптера в пространстве?

1. В регуляторе оборотов
2. В плате распределения питания
3. В полетном контроллере
4. В пульте радиопередачи

5. Какие типы аккумуляторов бывают?

1. Литий-ионные
2. Литий-полимерные
3. Свинцово-кислотные
4. Никель-металл-гидридные

Основы электричества

1. Как обозначается сопротивление в законе Ома?

1. I
2. R
3. U
4. S

2. Как обнаружить короткое замыкание в цепи?

1. “Прозвонить” мультиметром
2. Измерить напряжение во включенном состоянии
3. Измерить сопротивление в цепи
4. Измерить напряжение в выключенном состоянии

3. При каком типе соединения аккумуляторов напряжение складывается?

1. Последовательное
2. Параллельное
3. Смешанное
4. Замкнутое

4. Электрический ток это -

1. Движение заряженных частиц (электронов).
2. Движение заряженных частиц (протонов).
3. Движение заряженных частиц (бозонов).

4. Движение заряженных частиц (нейтронов).

5. Сумма токов, подходящих к узловой точке электрической цепи, равна

1. Разности токов приходящих к узлу и уходящих от него
2. Полусумме токов, уходящих от этого узла
3. Сумме токов, уходящих от этого узла
4. Произведению токов, уходящих от этого узла

6. Что отражает закон Джоуля-Ленца

1. Направление силы тока и силовых магнитных линий
2. Переход электрической энергии в тепловую
3. Связь электродвижущей силы источника (или электрического напряжения, с силой тока, протекающего в проводнике, и сопротивлением проводника)
4. Соотношение между токами и напряжениями в разветвленных электрических цепях

Теория пайки

1. Чего нельзя делать во время пайки?

1. Соприкасаться жалами двух работающих паяльников
2. Трогать жало паяльника
3. Очищать жало паяльника при помощи металлической губки
4. Паять на температуре свыше 400 градусов

2. Что нужно сделать с проводами перед тем, как спаять их между собой?

1. Изолировать
2. Зачистить
3. Залудить
4. Скрутить

3. За какую часть следует держать паяльник?

1. Фартук
2. Ручка
3. Корпус
4. Жало

4. На каком этапе используется флюс?

1. Лужение
2. Процесс спаивания двух поверхностей
3. Зачистка
4. Скручивание многожильных проводов

5. Какой флюс следует использовать с осторожностью при пайке микросхем?

1. Нейтральный
2. Активированные

3. Пассивный
4. Активный

Аэродинамика полета. Пропеллер

1. К чему ведет увеличение диаметра пропеллера?

1. Уменьшению расхода заряда аккумулятора
2. Увеличению подъемной силы
3. Ускорению набора скорости вращения
4. Замедлению набора скорости вращения

2. Пропеллер с каким количеством лопастей создает наибольшую подъемную силу

1. 2
2. 3
3. 4
4. Подъемная сила не зависит от количества лопастей

3. Что будет если пропеллеры установить в перевернутом виде?

1. Коптер перевернется
2. Коптер будет лететь вниз
3. Коптер взлетит, но с меньшей скоростью
4. Коптер начнет вращаться вокруг своей оси

4. При каких дефектах на воздушном винте нельзя совершать полеты?

1. Трещина на лопасти
2. Лопасть сколота на 20%
3. Лопасть имеет зазубрины
4. Лопасть искривлена

5. В соответствии с какими параметрами моторов БПЛА подбираются пропеллеры?

1. Количество обмоток
2. Мощность двигателя
3. Токопотребление
4. Частота вращения

Основы электромагнетизма. Типы двигателей

1. Какие моторы чаще всего используются в коптерах?

1. Коллекторные
2. Асинхронные
3. Бесколлекторные
4. Синхронные

2. Отметьте преимущества коллекторных двигателей:

1. Высокий КПД
2. Низкий вес двигателя

3. Продолжительный срок службы
4. Низкая стоимость

3. Отметьте преимущества бесколлекторных двигателей

1. Высокий КПД
2. Низкая стоимость
3. Высокая максимальная скорость
4. Высокая износостойкость

4. Как можно изменить направление вращения бесколлекторного двигателя на коптере?

1. Поменять "+" и "-"
2. Перепрощить регулятор оборотов
3. Поменять между собой 2 фазных провода
4. Это невозможно

5. Как можно изменить направление вращения коллекторного двигателя на коптере?

1. Подать на оба провода ток "+"
2. Поменять "+" и "-"
3. Подать на оба провода ток "-"
4. Это невозможно

Бесколлекторные двигатели и регуляторы их хода

1. Что необходимо использовать для работы бесколлекторного двигателя?

1. Систему охлаждения
2. Стабилизатор напряжения
3. Регулятор оборотов
4. Виброразвязку

2. Как подается ток на обмотки трехфазного бесколлекторного двигателя?

1. Парно подается ток + и - на обмотки
2. Парно подается ток - и - на обмотки
3. Парно подается ток + и + на обмотки
4. Ток подается на все обмотки сразу

3. Какой кратности должно быть число обмоток в бесколлекторном моторе?

1. 2
2. 3
3. 5
4. 7

Принцип работы, типы и устройство аккумуляторов

1. Какая характеристика аккумуляторов влияет на скорость вращения моторов?

1. Емкость
2. Максимальный разрядный ток
3. Напряжение
4. Токоотдача

2. На что влияет емкость аккумулятора

1. На время работы
2. На максимальное выдаваемое напряжение
3. На время заряда аккумулятора
4. На величину тока, которым можно заряжать аккумулятор

3. Каким напряжением можно запитать зарядное устройство Li-Po аккумуляторов для коптеров?

1. 5В
2. 12В
3. 100В
4. 220В

4. Что произойдет в случае прокола Li-Po аккумулятора

1. Вытекание кислоты
2. Возгорание
3. Вздутие аккумулятора
4. Ничего не произойдет

5. Как обозначается трехбаночный аккумулятор?

1. 3C
2. 3S
3. 3V
4. 3G

Управление полетом мультикоптера. Принцип функционирования полетного контроллера. ПИД регуляторы

1. Что является “мозгом” коптера?

1. Регулятор оборотов (ESC).
2. Плата распределения питания
3. Полетный контроллер
4. Радиоприемник

2. Какие функции не выполняет полетный контроллер?

1. Рассчитывает свое положение в пространстве, по показаниям датчиков
2. Прием сигналов с пульта
3. Вносит корректировку с помощью коэффициентов ПИД
4. Распределяет питание на моторы

3. Что обозначает P в формуле ПИД-регулятора

1. Мощность двигателя
2. Дифференциальная составляющая
3. Погрешность датчиков
4. Пропорциональная составляющая

4. Как обозначаются ШИМ-импульсы?

1. TX
2. PPM
3. PWM
4. RX

5. Как обозначается угол крена?

1. throttle
2. roll
3. force
4. spin

Основы радиосвязи. Принцип работы радиоаппаратуры управления

1. На какой частоте работает аппаратура радиоуправления копитера

1. 0-1 ГГц
2. 1-2 ГГц
3. 2-3 ГГц
4. 3-4 ГГц

2. Какое минимальное количество каналов управления нужно для дрона?

1. 2
2. 4
3. 6
4. 8

3. Как обозначается фазово-импульсная модуляция?

1. TX
2. PPM
3. PWM
4. RX

4. Какого типа бывают каналы управления?

1. Импульсные
2. Дифференциальные
3. Дискретные
4. Пропорциональные

5. Куда передаются сигналы с радиоприемника в дрон?

1. На регуляторы оборотов
2. На моторы
3. На полетный контроллер
4. На плату распределения питания

Аналоговая и цифровая видеотрансляция. Применяемые камеры, радиопередатчики и приемники

1. Укажите преимущества аналоговых видеокамер перед цифровыми.

1. Помехозащищенность
2. Высокая совместимость
3. Просмотр видео в режиме реального времени
4. Высокая надежность

2. Что не относится к возможностям цифровых камер?

1. Возможность работы в паре с датчиком движения
2. Просмотр видео в режиме реального времени
3. Запись видео с точностью до долей секунд
4. Использование встроенного динамика и микрофона

3. Выберите верные утверждения.

1. Дальность передачи видеосигнала не зависит от количества помех в зоне полета
2. Разные системы передачи сигнала имеют различную способность огибать препятствия
3. Дальность полета не зависит от погоды
4. Дальность полета, в большинстве случаев, ограничивается лишь емкостью батареи, но для реализации всего потенциала современных технологий необходима наземная станция

4. Что не относится к схеме работы цифровой камеры?

1. Блок сжатия
2. АЦП
3. Блок оцифровки
4. ПЗС - матрица

5. Что относится к схеме работы аналоговой камеры?

1. Линза
2. Светофильтр
3. Блок оцифровки
4. Блок сжатия

Техника безопасности при сборке и настройке коптеров, при подготовке к вылету. Техника безопасности при работе с аккумуляторами

1. В какой момент нужно устанавливать пропеллеры на коптер?

1. Перед установкой моторов
2. При сборке защиты коптера
3. При настройке коптера
4. Перед взлетом

2. Что запрещается делать с Li-Po аккумуляторами?

1. Устанавливать на холоде
2. Подключать и отключать держась за разъемы
3. Наносить механические повреждения
4. Нарушать целостность изоляции

3. Выберите неверное утверждение.

1. Паяльник следует хранить в подставке
2. Паять можно только при естественном освещении
3. Нельзя паять включенные в сеть электроприборы
4. Во время пайки следует использовать пинцет и “третью руку”

4. Вы заармировали коптер. Пропеллеры коптера вращаются, но он не взлетает. Что следует проверить?

1. Заряд аккумуляторов
2. Правильность установки воздушных пропеллеров
3. Затянутость гаек на моторах
4. Уровень сигнала с пульта радиоуправления

5. Произошла аварийная ситуация и коптер упал. Что следует сделать в первую очередь?

1. Попытаться взлететь снова
2. Убрать коптер с полетной зоны
3. Disarm
4. Проверить целостность защиты

Теория ручного визуального пилотирования

1. Как называется процедура разблокировки моторов коптера?

1. Disarm
2. Kill Switch
3. Arm
4. FPV

2. Что должно произойти в первую очередь при FPV пилотировании?

1. Включение FPV шлема
2. Включение пульта управления
3. Включение питания коптера
4. Включение моторов

3. Что не включает в себя предполетная подготовка

1. Укладка проводов таким образом, чтобы они не попадали под пропеллеры

2. “Прозвонка” платы распределения питания
3. Проверка целостности рамы коптера
4. Правильная установка пропеллеров

4. В какой момент включается пульт радиоуправления?

1. Перед полетом после подключения аккумуляторов
2. Во время предполетной подготовки
3. Перед полетом до подключения аккумуляторов
4. Правильный ответ отсутствует

5. Как называется процедура блокировки (выключения моторов?)

1. Disarm
2. Kill Switch
3. Arm
4. FPV

Техника безопасности при летной эксплуатации коптера

1. Зачем нужен чеклист?

1. Чтобы записать показания заряда аккумуляторов
2. Чтобы отметить время полета
3. Чтобы отметить дальность полета
4. Чтобы верно провести предполетную подготовку

2. На каком минимальном расстоянии от коптера должен находиться пилот во время полета?

1. 0-1 м
2. 1-2 м
3. 2 - 3 м
4. Более 3 м

3. Где находятся зрители во время полета?

1. Слева от пилота, если пилот правша
2. Спереди от пилота на расстоянии 3-5 метров
3. За спиной пилота
4. Справа от пилота, если пилот правша

4. Чего нельзя допускать во время полета?

1. Резких движений стиками
2. Полной разрядки аккумуляторов
3. Полетов выше своего роста
4. Полетов далее 3 метров от себя

5. Укажите правильную последовательность действий при аварийной посадке.

1. Прекратить полёт. Посадить коптер на землю. Выключить пульт. Disarm (стик YAW влево вниз на 3 секунды.. Отключить аккумулятор на коптере.

2. Прекратить полёт. Посадить коптер на землю. Посадить коптер на землю. Отключить аккумулятор на коптере. Disarm (стик YAW влево вниз на секунды). Выключить пульт.
3. Прекратить полёт. Посадить коптер на землю. Disarm (стик YAW влево вниз на 3 секунды). Отключить аккумулятор на коптере. Выключить пульт.
4. Прекратить полёт. Посадить коптер на землю. Disarm (стик YAW влево вниз на 3 секунды). Выключить пульт. Посадить коптер на землю.

Обучение лётному мастерству

1. Как полететь вправо или влево?

1. Переместить стик в нужную сторону по яву
2. Переместить стик в нужную сторону по крену
3. Переместить стик в нужную сторону по газу
4. Переместить стик в нужную сторону по тангажу

2. Как полететь вперед или назад?

1. Переместить стик в нужную сторону по яву
2. Переместить стик в нужную сторону по крену
3. Переместить стик в нужную сторону по газу
4. Переместить стик в нужную сторону по тангажу

3. Как развернуть коптер вокруг оси, проходящей перпендикулярно плоскости коптера через его центр?

1. Переместить стик в нужную сторону по яву
2. Переместить стик в нужную сторону по крену
3. Переместить стик в нужную сторону по газу
4. Переместить стик в нужную сторону по тангажу

Основы радиоэлектроники, схемотехники и макетирования электрических схем

1. В каких единицах измеряется сила тока?

1. [Вольт]
2. [Кулон]
3. [Ампер]
4. [Ом]

2. Какого типа электронных схем не существует?

1. Гибридные
2. Пропорциональные
3. Цифровые
4. Аналоговые

3. Укажите электронный компонент, позволяющий ограничить ток.

1. Светодиод
2. Резистор

3. Конденсатор
4. Трансформатор

4. Укажите электронный компонент, служащий для накопления заряда и энергии электрического поля.

1. Светодиод
2. Резистор
3. Конденсатор
4. Трансформатор

5. Какого типа печатных плат не существует?

1. Замкнутые (ЗПП)
2. Односторонние (ОПП)
3. Двусторонние (ДПП)
4. Многослойные (МПП)

Основы работы с аналоговым и цифровым сигналом

1. Что не может измерить мультиметр?

1. Сопротивление
2. Напряжение
3. Силу тока
4. Правильный вариант ответа отсутствует

2. Какое значение измеряемой величины следует устанавливать на мультиметре?

1. Максимальное
2. Немного меньше предполагаемого значения
3. Немного больше предполагаемого значения
4. Минимальное

3. Зачем нужен режим “прозвонки”

1. Чтобы обнаружить разрывы в цепи
2. Чтобы обнаружить короткое замыкание
3. Чтобы измерить напряжение
4. Чтобы измерить силу тока

4. Что не измеряет осциллограф?

1. Угол сдвига фаз
2. Угловая скорость
3. Частота
4. Напряжение фазы по отношению к земле

5. Какое сопротивление покажет омметр, если соприкоснуть щупы между собой?

1. 0 Ом
2. 1 Ом

3. -1 Ом

Теория FPV полетов

1. Какой стик является основным для позиционирования при FPV полетах?

1. Roll
2. Pitch
3. Yaw
4. Throttle

2. Каким стиком удерживается высота?

1. Roll
2. Pitch
3. Yaw
4. Throttle

3. Что такое FPV пилотирование?

1. Полеты с ориентацией “от первого лица”
2. Полеты с грузом
3. Полеты в помещении
4. Полеты на большой высоте

История автономных полетов. Развитие автопилотов в авиации

1. Автопилот - это

1. БПЛА, который может лететь в заданную точку
2. Устройство или программно-аппаратный комплекс, который может вести вверенное ему транспортное средство по заданной траектории
3. Программа, заставляющая БПЛА лететь в заданную точку
4. Правильный вариант ответа отсутствует

2. Кто изобрел первый автономно управляемый аппарат?

1. Георгий Ботезат
2. Братья Райт
3. Леонардо да Винчи
4. Альфред Уайтхед

3. Какой прибор помогает определить ориентацию летательного аппарата?

1. Осциллограф
2. Гироскоп
3. Барометр
4. Гигрометр

4. Что такое “Фау-2”

1. Коптер

2. Самолет
3. Ракета
4. Спутник

5. Как называется первый полностью автономный наземный автопилот?

1. Фау-2
2. Тележка Леонардо да Винчи
3. Стэнфордская тележка
4. Торпеда Александровского

Основы программирование на языке Python

1. К каким типам языков относится Python?

1. Компилируемый
2. Низкоуровневый
3. Объектно-ориентированный
4. Высокоуровневый

2. Укажите правильную конструкцию.

```
1. if test1:
    state1
elif test2:
    state2
else:
    state3
```

```
2. if test1:
    state1
else:
    state2
elif test2:
    state3
```

```
3. a = int(input(a))
else a < -5:
    print('Low')
elif -5 <= a <= 5:
    print('Mid')
if:
    print('High')
```

```
4. a = int(input(a))
if a < -5:
    print('Low')
else -5 <= a <= 5:
    print('Mid')
elif:
    print('High')
```

3. Укажите верные обозначения логической истины.

1. 1
2. 0
3. True
4. False

4. Укажите верное обозначение логического оператора “и”.

1. elif
2. or
3. if
4. and

5. Что не является оператором?

1. for
2. continue
3. break
4. while

Тест по теме: «Знакомство с компьютером Raspberry Pi

1. Что такое Raspberry Pi 3?

1. Операционная система
2. Микрокомпьютер
3. Микроконтроллер
4. Процессор

2. Укажите количество ядер микрокомпьютера Raspberry Pi3 model B?

1. 1
2. 2
3. 4
4. 6

3. С помощью какой команды можно перейти в предыдущую директорию?

1. cd ~
2. cd /
3. cd ..
4. cd -

4. Укажите команду, используемую для перехода в директорию.

1. mkdir
2. nano
3. ls
4. cd

5. Как получить права суперпользователя?

1. sudo
2. nano
3. rm
4. ls -l

Управление автономным дроном: теория

1. Какой полетный режим используется для автономных полетов?

1. STABILIZED
2. OFFBOARD
3. ACRO
4. ALTHOLD

2. Почему нельзя ориентироваться только на показания датчиков при автономном полете?

1. Датчики не синхронизированы между собой
2. Коптеру недостаточно информации, получаемой с датчиков, чтобы определить свое положение в пространстве
3. Быстро накапливается ошибка
4. Для полета обязательно нужны глобальные координаты, получаемые с GPS спутника

3. Как называются метки, по которым ориентируется дрон?

1. QR
2. ArUco
3. ID
4. Map

4. Что измеряют сонары?

1. Температуру
2. Расстояние
3. Освещенность
4. Излучение

5. Что такое ROS?

1. Фреймворк
2. Редактор
3. Операционная система для роботов
4. Компилятор

Тест итогового контроля знаний

1. Как называется коптер с 8 моторами?

1. Пентакоптер
2. Октокоптер
3. Трикоптер
4. Гексакоптер

2. Как обозначается напряжение в законе Ома?

1. I
2. R
3. U
4. S

3. При каком типе соединения аккумуляторов напряжение не складывается?

1. Последовательное
2. Параллельное
3. Смешанное
4. Замкнутое

4. Какие типы флюсов следует использовать при пайке микросхем?

1. Нейтральный
2. Активированные
3. Пассивный
4. Активный

5. В соответствии с какими параметрами моторов БПЛА подбираются пропеллеры?

1. Количество обмоток
2. Мощность двигателя
3. Токопотребление
4. Частота вращения

6. Какие моторы редко используются в коптерах?

1. Коллекторные
2. Асинхронные
3. Бесколлекторные
4. Синхронные

7. Отметьте преимущества бесколлекторных двигателей

1. Высокий КПД
2. Низкая стоимость
3. Высокая максимальная скорость
4. Высокая износостойкость

8. Какой кратности должно быть число обмоток в бесколлекторном моторе?

1. 2
2. 3
3. 5
4. 7

9. Как обозначается трехбаночный аккумулятор?

1. 3C
2. 3S
3. 3V
4. 3G

10. Что является “мозгом” коптера?

1. Регулятор оборотов (ESC)
2. Плата распределения питания

3. Полетный контроллер
4. Радиоприемник

11. Какое минимальное количество каналов управления нужно для дрона?

1. 2
2. 4
3. 6
4. 8

12. Как обозначаются ШИМ-импульсы?

1. TX
2. PPM
3. PWM
4. RX

13. Что не относится к возможностям цифровых камер?

1. Возможность работы в паре с датчиком движения
2. Просмотр видео в режиме реального времени
3. Запись видео с точностью до долей секунд
4. Использование встроенного динамика и микрофона

14. В какой момент нужно устанавливать пропеллеры на коптер?

1. Перед установкой моторов
2. При сборке защиты коптера
3. При настройке коптера
4. Перед взлетом

15. Как называется процедура разблокировки моторов коптера?

1. Disarm
2. Kill Switch
3. Arm
4. FPV

16. Где находятся зрители во время полета?

1. Слева от пилота, если пилот правша
2. Спереди от пилота на расстоянии 3-5 метров
3. За спиной пилота
4. Справа от пилота, если пилот правша

17. В каких единицах измеряется напряжение?

1. [Вольт]
2. [Кулон]
3. [Ампер]
4. [Ом]

18. Что не является оператором?

1. for

Датчик температуры и влажности

2. continue
3. break
4. while

19. Какой полетный режим используется для автономных полетов?

1. STABILIZED
2. OFFBOARD
3. ACRO
4. ALTHOLD